Rule-based meta-modelling for bio-curation

Russell Harmer

École Normale Supérieure de Lyon

Mémoire d'habilitation à diriger des recherches
présenté et soutenu publiquement le 22 mai 2017

Rule-based meta-modelling for bio-curation

Russell HARMER

Rapporteurs :
Pierre-Louis CURIEN
François FAGES
Barbara KÖNIG

Examinateurs :
Pierre-Louis CURIEN
Thomas EHRHARD
François FAGES
Jasmin FISHER
Olivier GANDRILLON
Barbara KÖNIG
Pawel SOBOCINSKI

This page intentionally left blank.

Dossier

As explained in the main text, my scientific work has been carried out in two quite (technically) distinct fields: game semantics and rule-based modelling. For my *habilitation* dossier, I have chosen only papers from the latter field as this reflects the majority of my work subsequent to my PhD thesis. Out of my work in that field, I have focussed on the particular theme of *bio-curation* as this has always been my principal personal interest in rule-based modelling and is also the subject of my research project—as summarized in the final chapter.

[17] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. *CONCUR 2007–Concurrency Theory*, pages 17–41, 2007.

[27] V. Danos, R. Harmer, and G. Winskel. Constraining rule-based dynamics with types. *MSCS*, 23(2):272–289, 2013.

[28] * V. Danos, J. Feret, W. Fontana, R. Harmer, J. Hayman, J. Krivine, C. Thompson-Walsh, and G. Winskel. Graphs, Rewriting and Pathway Reconstruction for Rule-Based Models. In *FSTTCS 2012*, volume 18 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 276–288. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2012.

[37] R. Harmer, V. Danos, J. Feret, J. Krivine, and W. Fontana. Intrinsic information carriers in combinatorial dynamical systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(3):037108, 2010.

[26] * V. Danos, R. Harmer, R. Honorato-Zimmer, and S. Stucki. Deriving rate equations for site graph rewriting systems. *To appear in Electronic Notes in Theoretical Computer Science*, 2014.

*Paper for which I was not lead author but played a significant rôle.

[20] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling and model perturbation. In *Transactions on Computational Systems Biology XI*, pages 116–137. Springer, 2009.

[36] R. Harmer. Rule-based modelling and tunable resolution. *EPTCS*, 9:65–72, 2009.

[6] A. Basso-Blandin, W. Fontana, and R. Harmer. A knowledge representation meta-model for rule-based modelling of signalling networks. *EPTCS*, 204:47–59, 2016.

Acknowledgements

Special thanks to Pasquale, Guy, Vincent, Walter, Jean & Jérôme for helping me to get here; and to Rebecca & Winston for making it worthwhile.

There are many others too, of course; I trust you know who you are.

Ce mémoire est dédié à la mémoire de mon ami Benoît Carreau.

This page intentionally left blank.

Contents

Perspective

My current research concerns the development of a general, graph rewriting-based framework for knowledge representation, modelling and analysis that I call *rule-based meta-modelling* as it incorporates the entire modelling process itself as a modulable component of the framework. My particular interest at present lies in a specific instantiation of this general methodology to build a conceptual and software platform for *bio-curation* in the context of *cellular signalling* and its *deregulation* in disease states.

In this chapter, I provide a high-level overview of this specific project. Further detail and discussion can be found in the final chapter.

I Cellular signalling and rule-based modelling

Cellular signalling is the decentralized mechanism through which the cells of an organism coordinate during (embryonic and childhood) development, wound healing and adult homeostasis. Many diseases of the developed world, notably type II diabetes and many cancers, arise from the deregulation of signalling. A significant fraction of the human genome, estimated at 20–25% of the whole, codes for proteins dedicated to signalling; these proteins control the state of the cell through the agency of receptor proteins that detect the presence of various hormones in the cell's environment and which transduce these 'signals' into the modulation of gene expression. However, the result of signal tranduction also depends, to some extent, on the internal state of the cell—most notably, the current cell cycle phase—as this also substantially affects gene expression. In this way, the cell can be induced to grow, divide, bide its time, specialize its function or even die, depending on its own state and its external context. In particular, the 'same' signal can affect the 'same' cell differently at different times; and a single signal can affect cells of different types, *e.g.* liver versus skin, quite differently.

The 'wiring diagram' of signal transduction involves many thousands of proteins and tens, perhaps even hundreds, of thousands of individual protein-protein interactions (PPIs). These PPIs principally control (i) the transient association of proteins into so-called *complexes*; and (ii) the activation or inhibition of the enzymatic action of certain proteins to modify the state of other proteins. The modulation of gene expression depends on exactly which proteins get modified (and how) and on which complexes are formed which itself depends on prior modifications and complex formations and, ultimately, on the cell's state and external context.

Most protein modifications significant to signalling are of a binary 'present or absent' nature; however, it is not at all unusual for a single protein to have many [up to a couple of dozen] such sites of modification which means that, even in isolation, the protein can exist in many thousands of distinct states. Once the protein's multiple binding partners—which can themselves exist in many distinct states in isolation and have many binding partners—are additionally taken into consideration, the number of distinct complexes that can exist rapidly becomes truly *astronomical*. This fact rules out the use of traditional modelling methodologies such as ODEs, Petri nets and multi-set rewriting which rely on an a priori listing of all possible complexes.

In the last decade or so, an approach called *rule-based modelling* has emerged where protein complexes are represented explicitly as the connected components of a (very) large graph representing the whole system. In this setting, a model is a collection of *graph rewriting rules*, each representing a particular PPI; importantly, a rule expresses only the (known) necessary conditions for its corresponding interaction to occur. This avoids the need to enumerate all possible complexes in advance and enables the scalable representation and simulation of very high-dimension systems [22, 43].

In addition to stochastic simulation, the rule-based modelling approach in the **Kappa** [17] language enables further analyses: static analysis using abstract interpretation [23] and, especially, causal analysis [28] based either on the extraction of *causal traces* from execution traces of the simulator; or, when feasible, by a static enumeration of all possible such causal traces for a given collection of rules and initial state. After appropriate post-processing, these causal traces bear a very close resemblance to what biologists call *signal transduction pathways*. However, while biologists generally try to 'figure out' these pathways in their heads, the rule-based approach, given a collection of PPIs formalized as (graph) rewriting rules, reconstitutes these pathways automatically from the dynamics provided by the simulator.

II The bio-curation problem

The rule-based approach provides a solution to the problem of combinatorial explosion of the number of possible complexes. However, in order to take advantage of this, the modeller still needs to *collate* all pertinent information for each PPI of their desired model; and, while a great deal has been learned about signalling PPIs over the last two or three decades, a number of major obstacles remain before this knowledge can be fully exploited.

The first obstacle has a rather banal, but non-negotiable, nature: the overwhelming majority of knowledge has been reported only in the scientific literature and only in non-machine-readable format. This literature has been organized and made effectively searchable by the NCBI's PubMed repository (and related tools) but the actual knowledge remains imprisoned in those papers in the form of a combination of text and image. The ongoing DARPA 'Big Mechanism' programme [10], which I am involved in, aims to automate the extraction of textual knowledge by the use of natural language processing. While progress has undoubtedly been made, I think it is fair to say that there is little immediate prospect of removing human supervision of such a machine curation process; moreover, the programme makes no attempt to treat image-based knowledge, not all of which is reified in the text, which would clearly require its own specialized curation pipeline. As such, for the foreseeable future, the need for human intervention implies an inevitable bottleneck for any attempt to extract knowledge from PubMed on a large scale.

Even for a curation process sufficiently restricted in scope that the human bottleneck can be neglected, a second obstacle arises: the knowledge is highly dispersed and typically heterogeneous. A single paper may contain partial details about several PPIs; and the information pertinent to a single PPI is not contained in a single paper but, rather, partial details will be found across many papers. Moreover, some papers contain highly—even overly—detailed mechanistic descriptions while others provide far less detail. This situation imposes the need for a robust aggregation process where, as a paper is read, all the *pertinent* details it contains can be *immediately* incorporated into a curated format. In this way, the necessary details for PPIs can be gradually 'stitched together' from a variety of sources. In the absence of such a process, my own experience has shown that the human curator becomes rapidly overwhelmed by the apparent need to read an ever-increasing pile of papers: in effect, they are trying to stitch the details together in their head and, without machine assistance, the cognitive burden is simply too great.

This second obstacle is further complicated because the partial details one reads in scientific papers have varied epistemic status: sometimes they refer directly to *experimentally observed* facts; sometimes they *infer* evolutionarily plausible hypotheses from observed facts; sometimes they state *phenomenological observations* about the overall effect of many PPIs in some particular context; and sometimes they propose purely *hypothetical* PPIs, advanced as possible explanations of phenomenological observations. Clearly, these varied kinds of information cannot be used in the same way; but they do all have a rôle to play in the curation process: the first two and the last provide raw material, of differing quality, while the third provides assertions that can be investigated through the use of simulation, static and causal analysis.

Let us now summarize by stating our basic **bio-curation problem**: we want a platform that enables the machine-assisted collation and aggregation of the knowledge that is pertinent to the construction of rule-based models of signal transduction and its deregulation in disease.

Philosophical aside Modelling in physics generally occurs *in the head* of the physicist and [what is usually called] the 'model'—typically a collection of ODEs or PDEs—is but a transcript of that mental construct. The overwhelming complexity of signal transduction renders this rather opaque modelling process completely infeasible; the sheer number of moving parts and the huge variety of ways that they can perturbed simply cannot fit in anyone's head*. As such, it seems that the modelling process itself must be reified explicitly, providing an audit trail and the means to test hypotheses and backtrack when necessary. In other words, modelling as a *tool for discovery* and not merely the codifying of achieved understanding in an artefact.

In practice, this means that the modelling process must emphatically *not* seek to 'figure out' how the model 'ought' to work, *i.e.* hard-wire its responses to known perturbations, but rather provide only the raw material required to construct, at any given time, a model whose behaviour can then be investigated with whatever analyses† we have at hand. In its conception of a model as a collection of lots of formalized PPIs, the rule-based approach fulfils this basic requirement; and, once coupled with additional machinery addressing the above concerns, should bring us closer to this conception of modelling as a tool for discovery.

*People have tried, of course, but the resulting models are brittle and unevolvable.

†cf. the above discussion of pathways and causal analysis

Trajectory

I give a high-level account of my scientific career from my PhD onwards. I have tried to avoid technical details, as far as reasonably possible, while maintaining a rigorous discourse; nonetheless, given the fairly broad scope, there may be a few moments where I must ask for a little indulgence. References are provided as entry points to the literature for the interested reader.

I Game semantics

In linguistics, the term '*semantics*' refers to the relationship between a class of *signifiers* and their *denotations*. A *formal* language provides a precise definition of 'a class of signifiers' which may correspond, among many other possibilities, to a fragment of some *natural* language or to a *programming* language. In *formal* semantics, the *process of assigning* a denotation to a signifier can itself be described as a particular signifier in an appropriate formal language. For example, in the case of particular interest to us here, the process of assigning a denotation to terms of a given programming language is generally defined as a *syntax-directed* process, *i.e.* 'by cases' according to the syntactic structure provided by the programming language, which could itself be written—were we so inclined—as a term in a second formal language that provides sufficient composition operators to build all required denotations. Let us note that this 'second formal language' is generally left implicit with only the assignment process itself being written down *in situ*, *i.e.* in the terms of the general mathematical universe* within which denotations lie: composition of functions, evaluation of a function at an argument, *&c.*

*In modern programming language semantics, this almost always means a *category*—and if said category has an *internal language*, the case of λ-calculus for Cartesian closed categories [47] being the exemplar, then this precisely provides an explicit definition of our second formal language.

The field of *game semantics* provides means to construct mathematical universes appropriate for defining the formal semantics of a wide variety of programming languages. Similarly to the sequential algorithms [15] approach that preceded it, these mathematical universes are generally (although not exclusively) Cartesian closed categories but are *not* extensional: they provide denotations that cannot be viewed as mathematical functions but rather as intensional descriptions of functions that explicitly include information about how the function is to be deployed, *e.g.* order of evaluation of arguments.

These denotations are called *strategies*, following the underlying metaphor that views the behaviour of a program as a collection of rules, a 'crib sheet', for a two-player game telling one of the players how to respond to the moves made by the other (representing the program's context of execution). A *play* of a game, an execution trace, is an alternating sequence of moves beginning with a move by the context, so that a strategy can be defined as a set of plays—possibly subject to certain constraints.

The key particularity of game semantics lies in its *definability* theorem which establishes a one-to-one correspondence between the (finite, η-long) normal forms of PCF—an idealized functional programming language: the typed λ-calculus with `boo`lean and `nat`ural numerals plus constructs, `if` and `case`, to manipulate local control flow—and a particular class of strategies satisfying a suite of technical constraints: *determinism*, *visibility*, *innocence* and *well-bracketing* [45]. In other words, in addition to the usual process assigning a strategy to each PCF term, we can further assign a PCF normal form to each (deterministic, visible, innocent and well-bracketed) strategy in such a way that going there-and-back-again returns the *normal form* of the starting term without performing any syntactic normalization whatsoever; a kind of 'normalization by evaluation'.

This definability theorem was refined and extended to richer settings by a number of researchers; perhaps the most notable landmarks were FPC [53], an extension of PCF with product, sum and recursive types, maintaining the correspondence with the same class of strategies; and μPCF [42], PCF extended with non-local control flow, which removed the constraint of well-bracketing while retaining the others. My subsequent work with V. Danos [24] deconstructed these results in order to reconstitute a new constraint on strategies, called *rigidity*, to establish definability even in the absence of local control flow; and my further work with O. Laurent [39] systematized all of the above into a single setting that even allowed a statement of definability in the absence of numerals through a further decomposition of rigidity.

The very tight correspondence between strategies and programs implied by these definability results left game semantics vulnerable to the criticism that it was nothing but 'glorified syntax'. Indeed, an earlier development of T. Coquand [11] and the independent, but contemporary with Hyland & Ong, work by H. Nickau on PCF [56] remained tightly anchored to syntax: in both cases, the 'semantic' notion of composition of strategies[†] retained the asymmetric character of its 'syntactic' correspondent, *i.e.* substitution. However, the work of Hyland & Ong precisely enabled an 'unfolding' of this vision into a properly symmetrized setting with an elegant and general notion of composition—based on parallel composition plus hiding à la CCS [54]—while retaining the property of definability. This result did not follow from purely syntactic considerations and required real proof.

Let us note that the work of Abramsky, Jagadeesan and Malacaria [2] independently provided the same definability result for PCF; however, this setting proved less 'modular' than that of Hyland & Ong and gradually fell from favour as a result. Indeed, although at the time game semantics was largely marketed as solving various long-standing *full abstraction* problems[‡], its real significance only became apparent *a posteriori* in the unanticipated, serendipitous unravelling (c. 1996–1999) of the interpretation of PCF: each of the above constraints (determinism, visibility, innocence and well-bracketing) required for PCF definability turned out to correspond to a specific, known sequential programming feature. We have already mentioned that dropping the constraint of well-bracketing corresponds to admitting non-local control flow. In a similar vein, although only proved indirectly by *factorization* rather than through a direct definability result, dropping innocence corresponds to introducing references for numerals [3]; and dropping visibility leads to full-blown references (including references to references) [1].

My PhD work contributed to this effort by analysing the role of the *determinism* constraint. Unsurprisingly, removing this corresponds to adding a finite non-deterministic choice construct; however, a more subtle analysis was also possible, inspired by the concept of *divergences* in CSP [7], so as to take account of the possibility of divergence as well as all possibilities of convergence, leading to an interpretation of Idealized Algol with respect to may- and must-convergence testing [40] (with G. McCusker).

[†]Coquand called them *E-dialogues*.

[‡]We choose not to comment this particular debate. The interested reader can find a detailed and measured discussion in [16].

My PhD manuscript [35] remains a standard reference for the state of the art in 1999, providing a clean general framework and resuming the results of this unravelling process for all the above constraints. My early post-doctoral work with V. Danos [25] extended this with the possibility to express *probabilistic* strategies. Unlike my PhD work—where non-determinism arises due to a program's lack of knowledge of a nonetheless deterministic context—this setting corresponds to the situation where programs are themselves making non-deterministic choices. Indeed, the resulting interpretation of Idealized Algol has a flavour of *fairness*: a program that, under the interpretation given in my PhD work may diverge, would now converge with probability 1; or, to put it differently, the possibility of divergence results entirely from the behaviour of a malicious (deterministic) scheduler in the context.

My final works in game semantics concerned a deeper and finer analysis of certain aspects of innocent strategies. One line of work, in conjunction with my PhD student Pierre Clairambault[§], provided an analysis of the class of *total* innocent strategies, where a strategy always has a valid response irrespective of what its context does, and their relation to *proofs* in logic as opposed to programs [9]. This work provided the technical basis for the subsequent developments, principally concerning inductive and co-inductive types, in Clairambault's thesis [8]. Another line of work, in collaboration with M. Hyland and P.-A. Melliès, investigated the combinatorial underpinnings of the composition of innocent strategies [38]. Indeed although, as noted above, this can be written in a generic 'parallel composition plus hiding' form, this presentation leaves a great deal of additional structure, particular to the innocent case, implicit. This work showed one way of making this additional structure explicit, at the cost of (apparently) restricting the applicability of the approach to defining composition *only* of innocent strategies.

I still retain a distant interest in game semantics, primarily with respect to two unresolved questions upon which it may yet shed some light: whether or not contextual equivalence in *sesqui* PCF[¶] is decidable; and the precise relationship between (innocent) strategies and sequential algorithms. The first question is intriguing because, in the case of sesqui PCF, the techniques, due to Loader [50] and Schmidt-Schauß [60], to establish decidability of unary PCF fail; and likewise for the technique of Loader [51] to prove undecidabilty of boolean PCF.

[§]defended 02/2010, now CR at the CNRS (LIP, ENS Lyon)

[¶]unary PCF with a single error value

The second question arose out of initial attempts to solve the first. The principal difference between strategies and sequential algorithms concerns their treatment of multiple calls to a function: sequential algorithms provide a far more subtle account than game semantics where functions are called at most once and, if a second evaluation proves necessary, this occurs via a *backtracking* mechanism. I have shown how, in restricted cases, this backtracking interpretation can be obtained incrementally from an innocent strategy via a *cellularization* procedure inspired by that of Schmidt-Schauß [60] (and subsequently simplified by Loader[‖]). In the case of unary PCF, the result of this procedure can be optimized to recover a well-bracketed strategy that is the denotation of the canonical representative of the contextual equivalence class of the original term; the hope, as yet unrealized, was that this might extend to full sesqui PCF.

A more general approach could be to try to adapt the ideas of Hofmann, Streicher and Reus [44,63], and as later extended by Selinger [62], to a setting appropriate for *delimited* control operators [29]. This would provide a richer formal language for defining denotations with support for the reification of 'bits of code' as composable entities whose subsequent recomposition could include sophisticated backtracking behaviour. In such a setting, the *process* of cellularization could potentially be formalized abstractly, thus avoiding the (huge) concrete complications that arise from working within the particular mathematical universe of game semantics. *On verra bien ; à l'occasion.*

II Rule-based modelling

2007–2009 My first work in the field of rule-based modelling consisted in the building of a large proof-of-concept model of a real cellular signalling system. The aims were rather prosaic: to test the behaviour and performance of the early Kappa simulator and, in particular, the extraction of causal traces from execution traces [17]. I started building this model by refactoring an existing ODE model [61], a process that rapidly forced me to turn to the primary (biological) literature in order to fill in a variety of details that had been abstracted away—in order to simplify—in the transcription of the ODEs. Although at the time it seemed more like a fun puzzle that could be 'solved' by reading enough and thinking hard, with hindsight this was my first exposure to the bio-curation problem.

[‖]Unpublished note: `http://homepages.ihug.co.nz/~suckfish/papers/atomic.pdf`

This thread of work continued until late 2009, culminating in a proposed meta-language for Kappa to relieve some of the cognitive burden of writing large collections of rules [20, 36]. In particular, it enabled the definition of *generic* rules that can express PPIs that are *shared* by a family of proteins—and provided their automatic expansion into the implied full set of Kappa rules—and loss- (but not gain-)of-function mutations where a generic rule could be expanded only for a specified subset of a family of proteins. This work enabled the further extension of the model of [17] which was validated, using a very simple class of assertions in conjunction with the static and causal analyses of Kappa, with respect to the experimental data of [64]. Concretely, this provided the assurance that the Kappa model was capable of reproducing the qualitative behaviour reported in [64]—an illustration of the most basic requirements of bio-curation.

2008–2013 In parallel with this thread of nascent bio-curation, I worked on a number of theoretical questions about Kappa. The first, in collaboration with V. Danos and, subsequently, also E. Murphy, concerned the notion of rule *refinement*—the splitting of a rule into a collection of non-overlapping, and usually exhaustive, sub-cases—as the motor for a systematic process of deriving model variants from a starting model [18, 19, 55].

Several years after this initial work, in collaboration with V. Danos and R. Honorato-Zimmer, the refinement methodology was applied to construct automatically a thermodynamically-correct collection of Kappa rules given a collection of *generator* rules and a set of *energy patterns*. Thermodynamic correctness, in this context, means that each rule has a fixed, statically-determined impact on the number of instances of each energy pattern (that can be written as a vector of integers). In order to achieve this, the generator rules must be systematically refined to reveal sufficient additional context so that each energy pattern is either definitely included, and so impacted, or definitely excluded.

The second question concerned the definition, given a collection of Kappa rules, of a system of ODEs whose solution corresponds to the thermodynamic limit of the stochastic dynamics of the rules. In other words, the stochastic behaviour converges to the ODEs as the size of the system is increased. One approach to this question, in which I played a minor rôle, used abstract interpretation to identify all information flows implied by the rules from which a system of ODEs can be determined [21, 34].

A second approach, in which I played a major rôle, specified the required class of graphs as a category and exploited the categorical structure in order to identify, given the starting collection of Kappa rules and a collection of desired observables, a collection of graphs that are sufficient to keep track of the average stochastic dynamics of the system [26, 37]. Given this, a system of ODEs can easily be written that captures the average behaviour of the desired observables.

The third question addressed the definition of the (stochastic) dynamics of a collection of Kappa rules, and the subsequent extraction of causal traces, in terms of graph rewriting. Initial work, principally in collaboration with G. Winskel, used the double push-out approach to describe the side-effect-free fragment of Kappa [27]; subsequent work, led by J. Hayman but in which I played a significant rôle, generalized this to full Kappa using the single push-out approach [28] and provided the first categorically-based treatment of the extraction of causal traces from the graph rewriting dynamics.

In the next chapter, I give a condensed presentation of all aspects of graph rewriting that have been required for rule-based modelling to date. This subsumes, and generalizes, the various approaches to the theoretical questions cited above and will be used, in the subsequent chapter, as a *post hoc* reading guide to the papers of my *habilitation* dossier.

2011– After a hiatus where I was occupied with the theoretical questions discussed above, I returned to the theme of bio-curation, initially in the context of a short exploratory project with NIBR[**], during my sabbatical period at Harvard Medical School. Although this collaboration, with S. Jaeger and J. Loureiro (NIBR) and W. Fontana (HMS), gave rise to no publication, it led to the idea that bio-curation critically requires a 'staging area' that plays a double rôle: it represents knowledge *and* provides an executable model.

This vision has subsequently begun to be realized—in the context of two DARPA programmes: Big Mechanism and Communicating with Computers—as an instance of a general methodology that I call *rule-based meta-modelling*. This approach again relies on graph rewriting; however, usual rewriting rules are now reified as graphs which are themselves subject to rewriting. This provides a setting where the knowledge/executable rules can be continually updated in the light of new information. The basic mathematical framework appeared recently [6] and will be further discussed in the final chapter.

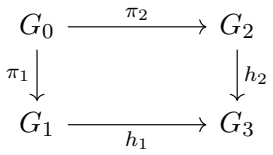
[**]Novartis Institute of Basic Research, Cambridge, MA

This page intentionally left blank.

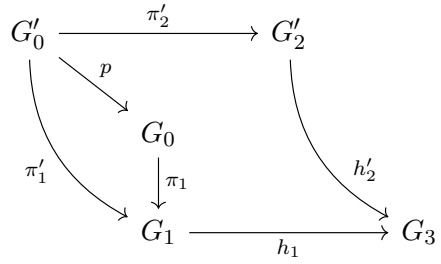
Graph rewriting

Preamble

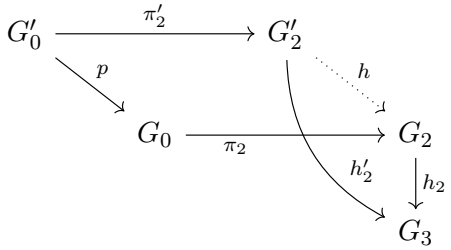
The *pull-back complement* (PBC) [31] of any pair of composable arrows in a category, $\pi_1 : G_0 \to G_1$ and $h_1 : G_1 \to G_3$, is a pull-back



satisfying [its universal property]: for any pull-back and arrow $p : G'_0 \to G_0$



such that $\pi_1 \circ p = \pi'_1$, there exists a unique arrow $h : G'_2 \to G_2$



such that $h_2 \circ h = h'_2$ and $h \circ \pi'_2 = \pi_2 \circ p$.

In words, the PBC gives the largest, *i.e.* least general, G_2 for which the completed square is nonetheless a pull-back. In the simplest imaginable case—the category of sets and set inclusions—the PBC of $A \subseteq B \subseteq D$ is $C := D - (B - A)$: all the other candidates are included in C (and include A).

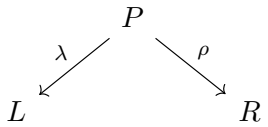
In this chapter, we present a rapid technical introduction to *sesqui-push-out* graph rewriting [13]. The basic setting is a category* $\mathbf{C}$ with all pull-backs, all push-outs and all pull-back complements *over monos*, *i.e.* the second arrow, h_1, in the above definition is a mono. The simplest model of these requirements is the category **Set** of sets and total functions—although it gives rise to a rather uninteresting setting where, effectively, we can only rewrite multi-sets over a singleton base set.

The pull-back, push-out and PBC constructions are all preserved† by the *slice* category construction $\mathbf{C}/T$ which, in practice, is a rich source of additional models, *e.g.* $\mathbf{Set}/T$ gives rise to multi-set rewriting over T.

In general, assuming that we consider $\mathbf{C}$ to be some category of 'graphs' and appropriate homomorphisms, an object of $\mathbf{C}/T$ can be viewed as a graph *typed by* T in the sense that each node acquires a type, the node of T to which it maps, and every edge must respect the 'admissible' edges provided by T; moreover, all arrows of the slice category must preserve typing in the natural sense. We will continually exploit this notion of typing in what follows.

I Rewriting

From now on, we assume that $\mathbf{C}$ is a category with all pull-backs, push-outs and PBCs over monos. In this setting, a *rewriting rule* is a span



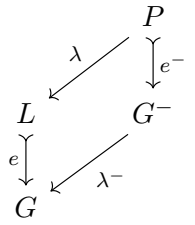
Unlike in the double push-out (DPO) [14] or the single push-out (SPO) [32] approaches, we do not require that λ be a mono. A rule is *linear* iff both legs of its span are monos.

*We do not give a self-contained introduction to category theory; the interested reader should consult one (or more) of [4, 49, 52]. In particular, we assume knowledge of basic concepts such as 'monos', 'pull-backs'/'push-outs' and 'slice' categories.

†see section I of the Appendix for full details

Cloning and deleting

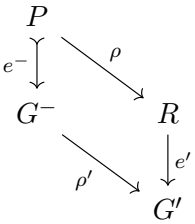
In general, nodes and edges in a graph can be cloned and deleted by a step of rewriting; the exact details depend on the class of graphs under consideration. This 'negative' phase of a rewriting step is defined by the left leg of the span $\lambda : P \to L$. An instance—also often called a *matching*—of this in a graph G is a mono $e : L \rightarrowtail G$ and the rewriting step is defined by taking a pull-back complement.



In concrete settings, such as **Set** or **Grph**, the category of simple graphs, deletion[‡] arises from failures of surjectivity of λ while cloning—impossible under DPO or SPO rewriting—arises from failures of injectivity.

Merging and adding

The right leg of the span $\rho : P \to R$ defines the adding and merging of nodes and edges. Again, the precise details depend on the exact class of graphs under consideration; but this 'positive' phase of rewriting is always defined by taking a push-out.



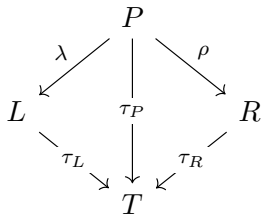
Concretely, failures of surjectivity in ρ give rise to addition while failures of injectivity give rise to merging.

Note that e' has no abstract reason to be a mono; in many concrete settings, this is however the case. Certain abstract settings, based on various notions of 'adhesivity' [46], can also be used to guarantee this property that 'push-outs reflect monos'.

[‡]An edge can also be deleted implicitly as the *side-effect* of the deletion of a node to which it is incident.

Typing

If our category is a slice category, a rewriting rule necessarily respects typing in the sense that

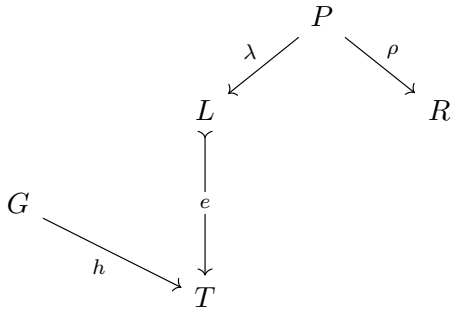


commutes; and likewise for the matching $e : L \rightarrowtail G$. After the negative phase of rewriting, G^- is still typed by T by pre-composition[§]: $\tau_{G^-} := \tau_G \circ \lambda^-$. By a routine diagram chase, we find that $\tau_R \circ \rho = \tau_{G^-} \circ e^-$ and, by the universal property of the push-out (of the positive phase), we have a unique arrow $\tau_{G'} : G' \to T$ which types G'. In words, rewriting preserves typing.

Retyping

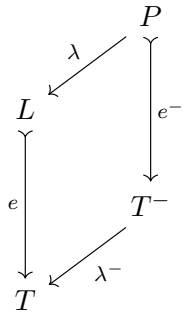
What if we wished to rewrite the type T? We might reasonably expect this to have a *generic* effect on all graphs G typed by T, *e.g.* if we delete a node of T, this should delete *all* nodes of G typed by that node of T.

Clearly, within the slice category framework, we cannot rewrite T itself directly as it is hard-wired into the slice category; instead, we must examine the situation in the underlying category by considering an untyped rule r with a matching e into T and an arbitrary arrow $h : G \to T$. This will allow us to 'rewrite' T by changing base.

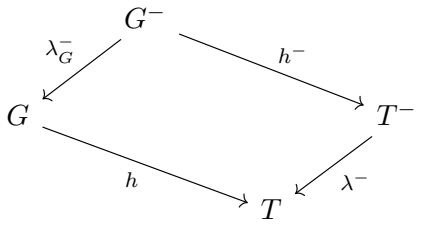


[§]cf. the post-composition 'change of base' functor [31]

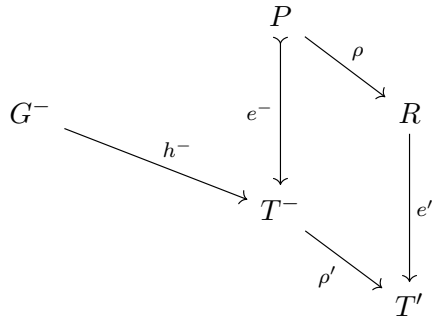
The negative phase of rewriting effected by r on T can be propagated[¶] canonically to G by building the PBC of λ and e



and then taking the pull-back[‖] of h and λ^- to obtain



with the final arrow 'typing' G^- defined by $h' := \rho' \circ h^- : G^- \to T'$.



There is *no* canonical propagation of this positive phase of rewriting to G^-; the effect is rather to change the *interpretation* of G^-, *e.g.* nodes in T^- can be merged so that differently-typed nodes in G^- have the same type in T'.

We can thus rewrite on a *typing tower* $G_1 \to \cdots G_n$ where a rule typed at level i canonically rewrites all levels $j < i$, leaving all levels $j \geqslant i$ unaffected as rewriting preserves typing.

[¶]We give an alternative, but equivalent, definition in section II of the Appendix.

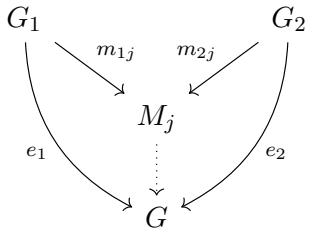
[‖]we are essentially evaluating the λ^- 'change of base' functor at h

II Matchings

Multi-sums

The notion of *multi-sum*—and, more generally, of *multi-co-limit*—is due to Y. Diers [30]. It generalizes the standard notion of *co-product* (and co-limit) to situations where the universal property fails in a 'not too serious' fashion: instead of having a single universal co-span, we ask for a *family* of co-spans that are *collectively* universal: any co-span factors uniquely through a unique family member.

Specifically, the multi-sum of two objects G_1 and G_2 consists of a family of co-spans $G_1 \xrightarrow{m_{1i}} M_i \xleftarrow{m_{2i}} G_2$ such that any co-span $G_1 \xrightarrow{e_1} G \xleftarrow{e_2} G_2$ factors through *exactly one* member M_j of the multi-sum—and *does so uniquely*.



Concrete examples of multi-sums often arise when we restrict attention to subcategories of monos, *e.g.* the co-product in **Set**—disjoint union—precisely splits into a multi-sum in the subcategory of injective functions. In this case, which is representative of all situations of concern to us here, the members of the multi-sum enumerate the ways in which the *images* of two functions can overlap: the requirement that the mediating arrow be a mono prevents us from taking simply the disjoint union.

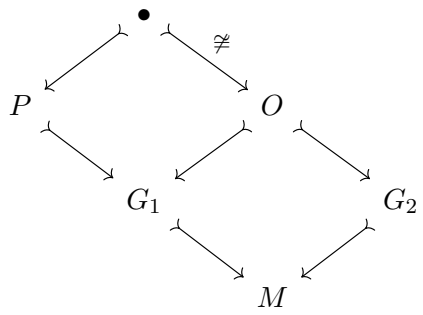
More generally, multi-sums provide us with some abstract machinery with which we can formalize relationships between matchings and, by extension, (instances of) rules. In effect, if we assume that the sub-category $\mathbf{C}_{>}$ of monos has all multi-sums, these provide us with an abstract enumeration of all *types of pairs* of matchings.

In the next section, we exploit this abstract enumeration in order to define two relations[**] between *rules* that capture the intuitive notions of co-operation and conflict.

[**]of an *intensional* character, as we make the witnesses explicit

Activation and inhibition

A mono $P \rightarrowtail G_1$ *influences* an object G_2 iff, for some $G_1 \rightarrowtail\!\!\!\!\!\!\rightarrow M \leftarrow\!\!\!\!\!\!\prec G_2$ in the multi-sum of G_1 and G_2 (in $\mathbf{C}_>$) with pull-back $G_1 \leftarrow\!\!\!\!\!\!\prec O \rightarrowtail\!\!\!\!\!\!\rightarrow G_2$, the arrow $\bullet \to O$ of the further pull-back



is *not* an isomorphism. In words, O is *not contained* in P so something in the overlap O of G_1 and G_2 is in G_1 but not P.

Given two *linear* rules $r_1 := L_1 \leftarrowtail P_1 \rightarrowtail R_1$ and $r_2 := L_2 \leftarrowtail P_2 \rightarrowtail R_2$, we say that r_1 *activates* r_2 iff $\rho_1 : P_1 \to R_1$ influences L_2 [via the *activation witness* $R_1 \rightarrowtail M \leftarrowtail L_2$]; and r_1 *inhibits* r_2 iff $\lambda_1 : P_1 \to L_1$ influences L_2 [via the *inhibition witness* $L_1 \rightarrowtail M \leftarrowtail L_2$].

In words, if r_1 activates r_2 then rewriting via r_1 *may* create new instances of r_2; and, if r_1 inhibits r_2 then rewriting via r_1 *may* destroy existing instances of r_2. Note that inhibition is *not* necessarily symmetric; note also that there may be *several* witnesses for a given activation or inhibition. We will exploit the additional information provided by the witnesses in the next section.

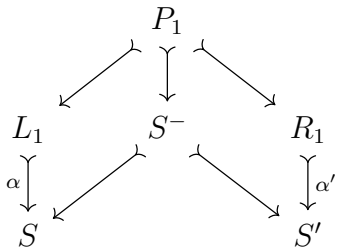
Application conditions

Given a rule $r := L \leftarrow P \rightarrow R$, an *application condition* (AC) is simply any mono $\alpha : L \rightarrowtail S$. An application condition can be thought of as being *negative* if, by default, we consider any matching of L in some graph G as inducing an admissible rewrite but take as exceptions, *i.e.* sub-cases where the rewrite is inadmissible, those matchings that factor through such an α. We abbreviate 'negative application condition' by NAC.

Dually, an AC can be thought of as *positive* if, by default, no matching is considered admissible unless it factors through such an α; we abbreviate 'positive application condition' by PAC.

The use of (negative or positive) application conditions may result in the *loss* of activation or inhibition witnesses: given a mono $P \rightarrowtail G_1$ that influences an object G_2 through witness $G_1 \overset{m_1}{\rightarrowtail} M \overset{m_2}{\leftarrowtail} G_2$ and two sets of monos, A_1 and A_2, from G_1 and G_2 respectively to be thought of as NACs, the witness is *eliminated* if m_1 factors through some $\alpha_{1i} : G_1 \rightarrowtail S_{1i} \in A_1$ *or* if m_2 factors through some $\alpha_{2j} : G_2 \rightarrowtail S_{2j} \in A_2$. Dually, taking A_1 and A_2 as PACs, if m_1 factors through no $\alpha \in A_1$ *or* m_2 factors through no $\alpha \in A_2$ then the witness never applies and is eliminated.

Given two *linear* rules $r_1 := L_1 \leftarrowtail P_1 \rightarrowtail R_1$ and $r_2 := L_2 \leftarrowtail P_2 \rightarrowtail R_2$ with sets A_1 and A_2 of NACs from L_1 and L_2 respectively, we therefore only need to keep those inhibition witnesses $L_1 \rightarrowtail M \leftarrowtail L_2$ that are not eliminated by any $\alpha \in A_1$ or $\alpha \in A_2$; and we need only keep those activation witnesses that are not eliminated by the rewrite $\alpha' : R_1 \rightarrowtail S'$



of any $\alpha : L_1 \rightarrowtail S \in A_1$ or by any $\alpha \in A_2$. Note that we are assuming here that push-outs reflect monos.

If *all* inhibition (resp. activation) witnesses for a given pair of rules are eliminated in this way, this changes the overall inhibition (resp. activation) relation.

III Implicit state simulation

We now define the (qualitative) dynamics of a *system* specified by a finite set of *linear* rules $\mathcal{R} := \{r_1, \ldots r_n\}$ and an object G—the current state. We must further require that push-outs in the ambient category *reflect* monos.

The event horizon

The $\mathcal{R}$-*event horizon* in G is the set of all matchings of all $\mathcal{R}$-LHSs into G; we denote this set by $\mathcal{E}_{\mathcal{R}}(G)$. As a convenient notation, we write $e_{i,k}$ for an event in $\mathcal{E}_{\mathcal{R}}(G)$ which is the kth matching of rule $r_i \in \mathcal{R}$.

Given $e_{i,k} \in \mathcal{E}_{\mathcal{R}}(G)$, we can perform the associated rewrite, obtaining a new object G'. In an implementation, such as that provided by the Kappa simulator, this state update is generally performed in-place.

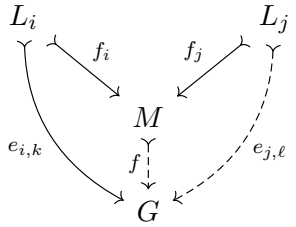
The new event horizon $\mathcal{E}_{\mathcal{R}}(G')$ is typically only mildly different from $\mathcal{E}_{\mathcal{R}}(G)$ and is also generally updated in-place. The changes can involve both the *removal* [negative update] and *addition* [positive update] of matchings from/to the event horizon; and even those that have been 'preserved' must still be updated to reflect the rewrite, *i.e.* to *define* the new event horizon mathematically, we must calculate the *residual* $e_{j,\ell}/e_{i,k} \in \mathcal{E}_{\mathcal{R}}(G')$ of every 'preserved' $e_{j,\ell} \in \mathcal{E}_{\mathcal{R}}(G)$.

In practice, the calculation of residuals generally occurs *implicitly* via the in-place update of G to G' since a matching into the current state will not be affected by a rewrite that does not touch the image of the matching. However, the negative and positive updates require considerable effort [22] and are still the subject of ongoing research[††].

Negative update

Recall that an *inhibition witness* is a co-span of the form $L \rightarrowtail M \leftarrowtail L'$ in the multi-sum of L and L' where L and L' range over LHSs of rules in $\mathcal{R}$. As a convenient notation, given $\mathcal{R}$, we denote by $\mathcal{I}_i$ the set of all inhibition witnesses (modulo ACs) where L is the LHS of r_i.

Given $e_{i,k} \in \mathcal{E}_{\mathcal{R}}(G)$, *for all* inhibition witnesses $f_i : L_i \rightarrowtail M \leftarrowtail L_j : f_j \in \mathcal{I}_i$ and *for all* factorizations $e_{i,k} = f \circ f_i$, the matching $f \circ f_j =: e_{j,\ell} \in \mathcal{E}_{\mathcal{R}}(G)$ of the LHS of r_j must be removed from $\mathcal{E}_{\mathcal{R}}(G')$.



The existence of one or more such fs means that $e_{i,k}$ matches G in a way that is compatible with the refinement[‡‡] M of L_i and L_j. Every such f *necessarily* gives rise to an $e_{j,\ell}$ that must be removed: the algorithmic difficulty lies in efficiently identifying the $e_{j,\ell}$s despite the double universal quantification.

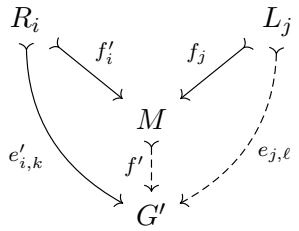
[††] J. Krivine and P. Boutillier, private communication.

[‡‡] specialization to a sub-case

Positive update

An *activation witness* is a co-span of the form $R \rightarrowtail M \leftarrowtail L$ in the multi-sum of R and L where R (resp. L) ranges over RHSs (resp. LHSs) of rules in $\mathcal{R}$; let $\mathcal{A}_i$ be the set of all activation witnesses (modulo ACs) for the RHS of r_i.

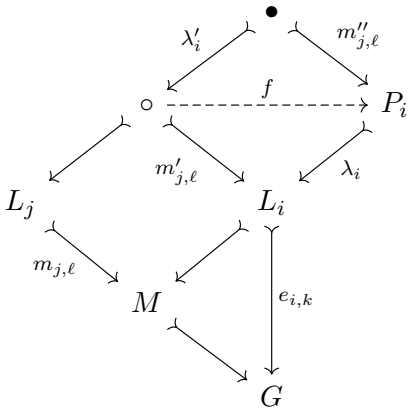
Given $e_{i,k} \in \mathcal{E}_{\mathcal{R}}(G)$, we obtain $e'_{i,k} : R_i \rightarrowtail G'$ after rewriting; so, *for all* activation witnesses $f'_i : L_i \rightarrowtail M \leftarrowtail L_j : f_j \in \mathcal{I}_i$ and *for all* factorizations $e'_{i,k} = f' \circ f'_i$, the matching $f' \circ f_j =: e_{j,\ell} \in \mathcal{E}_{\mathcal{R}}(G')$ must be added to $\mathcal{E}_{\mathcal{R}}(G')$.



Residuals

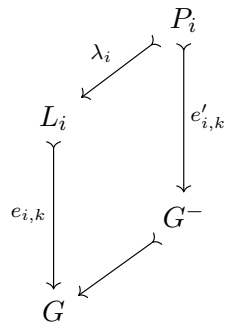
The remaining case we need to consider arises for all $e_{j,\ell} \in \mathcal{E}_{\mathcal{R}}(G)$ for which the left leg of r_i, $\lambda_i : P_i \to L_i$, does *not* influence L_j. Intuitively, this leaves $e_{j,\ell}$ 'unchanged' but, mathematically speaking, we still need to define the *residual* $e_{j,\ell}/e_{i,k}$ since the co-domain changes (from G to G').

Given $e_{i,k}, e_{j,\ell} \in \mathcal{E}_{\mathcal{R}}(G)$, they have a unique factorization through a unique member $L_i \rightarrowtail M \leftarrowtail L_j$ of the multi-sum of L_i and L_j. The pull-back $L_i \leftarrowtail \circ \rightarrowtail L_j$ of $L_i \rightarrowtail M \leftarrowtail L_j$ is therefore also the pull-back of $L_i \rightarrowtail G \leftarrowtail L_j$ and, since $\lambda_i : P_i \to L_i$ has no influence on L_j, the arrow $\lambda'_i : \bullet \to \circ$ in the further pull-back

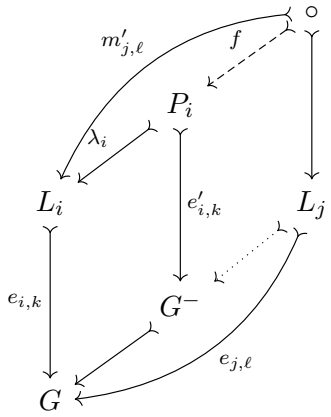


is an isomorphism and we thus obtain the dashed arrow $f := \lambda_i'^{-1} \circ m''_{j,\ell}$.

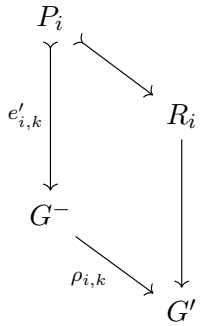
We can now calculate $m'_{j,\ell} = m'_{j,\ell} \circ \lambda'_i \circ \lambda'^{-1}_i = \lambda_i \circ m''_{j,\ell} \circ \lambda'^{-1}_i = \lambda_i \circ f$, so the pull-back complement



factorizes the pull-back of $L_i \to G \leftarrow L_j$ and, by its universal property,



we obtain the dotted arrow $\mathrm{e}^-_{j,\ell} : L_j \rightarrowtail G^-$ [which is a mono because $e_{j,\ell}$ is]. After post-composition with $\rho_{i,k}$ from the push-out,



we define the residual $e_{j,\ell}/e_{i,k} := \rho_{i,k} \circ e^-_{j,\ell} : L_j \to G'$.

The property that push-outs reflect monos implies that $\rho_{i,k}$ is a mono; so $e_{j,\ell}/e_{i,k}$ is also a mono. More generally, it would be necessary to generalize the notions of activation and inhibition—indeed, the underlying notion of influence—to all arrows, not just monos, in order to extend the negative and positive updates to non-linear rules that can merge and clone nodes: a clone, although specified by the left leg of the span, impacts upon the positive update; and a merge, dually, on the negative update.

However, this would take us beyond the intended scope of this manuscript as linear rules suffice for the graph-based formulation of Kappa. Nonetheless, it would be an interesting direction for future research to consider how to extend implicit state simulation to full-blown sesqui-push-out rewriting and to even more general approaches, *e.g.* the AGREE framework [12].

Reinterpretation

In this chapter, I provide a *reinterpretation* of the papers in my dossier in terms of a general framework that I call *rule-based meta-modelling*. This serves the double purpose of (i) setting those papers in a unified technical context that hopefully aids the reader to understand the progression of ideas and to appreciate, with the benefit of hindsight, the technical difficulties that motivated various developments, *i.e.* a kind of 'reader's guide' to the dossier; and of (ii) allowing me to situate my research project, as discussed in the (next and) final chapter, in the same technical context so as to emphasize that it can be seen as *one* logical progression from what came before.

Rule-based meta-modelling is not so much a technical idea as a 'way of thinking' that emerges from the previous chapter, the intention being to fully exploit the preservation of the categorical structure required to define graph-like rewriting, and activation and inhibition, by various constructions—most notably slice categories—and the ability to propagate rewriting in a 'tower' of graphs. This enables a workflow where an ambient 'universe' can be defined simply by writing down a concrete 'graph'*—the *meta-model*—to serve as the root of a tower whose upper levels are then automatically 'framed' in terms of the nodes and edges of that meta-model.

To take an overly simple example, the meta-model for multi-set rewriting is a singleton set representing formally the *concept* of a 'kind of element' of a multi-set; in chemical kinetics, this would be the concept of 'molecular species'. The next level up is a *model* which is the set of all *actual* elements—the molecular species—relevant to a given system which, in turn, types all the rewriting rules that collectively define the system (and its implied dynamics). A rewrite at one level, *e.g.* to delete a molecular species, can be automatically propagated, *i.e.* to delete all mention of that species in the rules.

*In our case, in the category of graphs with multiple, disjoint edge structures [alternatively, where edges as well as nodes can have attributes] and their homomorphisms.

As we will see, the meta-model for Kappa is a bit less trivial and, more generally, adopting this way of thinking enables a more incremental approach to modelling: one starts with a basic meta-model and, as experience reveals its defects, it can be updated—through a rewrite—to allow the expression of richer and/or refined concepts; critically, this update propagates upwards, providing a canonical and intrinsic notion of backwards compatibility. In contrast, even in a purely mathematical development, if the meta-model has been implicitly hard-wired into multiple fundamental definitions, this already obfuscates its transversal rôle and greatly complicates the effort of 'updating the meta-model' as and when the need arises.

Beyond such (meta-)mathematical considerations, in collaboration with various interns and post-docs, I have developed the `ReGraph` Python library[†] which provides support precisely for the definition of towers of graphs, rooted by a meta-model, and sesqui-push-out rewriting with upward propagation. As I will discuss in my project in the final chapter, this library provides the basic functionality upon which the bio-curation machine is being built. This work will be presented in detail in a forthcoming paper.

I Semantics of Kappa rules

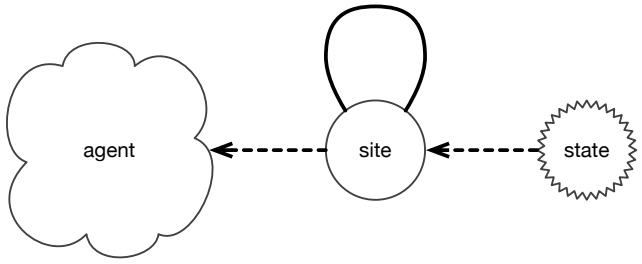
The first mathematical interpretation of Kappa[‡] rules, qua graph rewriting rules, and the dynamics induced by a collection of Kappa rules made use of the double push-out approach to rewriting and, as such, necessarily had to restrict to a side-effect-free fragment of Kappa [27]: the *mass-preserving* linear rules, *i.e.* no deletion/addition of agents, plus addition/deletion of fully-specified connected components with respect to the contact graph.

Although earlier work had already defined a category of realizable site graphs and embeddings [19, 55], the work in [27] marked the first time that a general notion of site graph and homomorphism was defined, giving rise to a category with the structure required to define rewriting. Another technical novelty of the paper was the explicit specification of a *contact graph* as a type and the use of its slice category to control the kinds of graphs and rewriting rules under consideration; this was the first, faint sign of the general meta-modelling framework.

[†] `https://github.com/Kappa-Dev/ReGraph` (with, in alphabetical order, Adrien Basso-Blandin, Ismaïl Lahkim Bennani, Yves-Stan Le Cornec and Ievgeniia Oshurko)

[‡] Refer to section III of the Appendix for a brief guide to the concrete syntax of Kappa.

Indeed, with hindsight, the entire basic mathematical framework could have been more compactly defined by specifying a simple meta-model



[where the dotted edges represent a 'belongs to' relation and the solid edge represents Kappa bonds] and imposing some mild constraints to ensure that (i) every site (resp. state) belongs to an agent (resp. site); and (ii) no site (resp. state) belongs to more than one agent[§] (resp. site). Further constraints could have been applied more selectively to obtain sub-classes of graphs of interest, *e.g.* realizable and (realizable and) fully-specified. Let us also note that this meta-model represents states as nodes, exactly as in [27]; but, with hindsight, it would have been more natural to use attributes[§] on sites.

Aside In my opinion, the key advantages of the meta-modelling approach—'state your concepts and the constraints they should satisfy'—are (i) that the explicit statement of constraints, de-convolved from concepts, makes explicit the 'wiggle room' that you still have without needing to enrich your concepts; and (ii) provides cognitive support for the fateful day when you realize that you do need to upgrade your meta-model, by at least enabling you to bring along your work to date. As noted above, this is even useful for 'purely mathematical' modelling, *i.e.* everyday mathematical work [59], but becomes completely indispensable in the wild open world of knowledge representation.

The above meta-model does have one slight difference to the framework of [27] in that it lacks the 'free site' predicate. However, this does not result in any loss of expressivity—at least with respect to defining an interpretation of Kappa rules that correctly describes their *intended* actions—because a 'free site' condition can instead be expressed by a collection of NACs—one for each possible incident (solid) edge in the contact graph.

[§]The `ReGraph` library provides such functionality.

This issue of whether a 'free' site should be represented explicitly in some way or implicitly via NACs becomes central once we attempt to extend our mathematical interpretation to full-blown Kappa [28]. If, as in [27], we have a 'free site' predicate in the structure of our graphs, this implies that (usually) all sites in a contact graph satisfy this predicate—as well as having multiple incident edges. This makes sense because, read as a type, this merely asserts that each site *may* be free but also *may* be bound in various specified ways.

A Kappa rule such as

```
A(s!_) -> A(s)
```

applied to such a graph would pick an edge incident to `s`, delete it and make `s` 'free', despite leaving all other incident edges in place; but, most of all, *not* make the unnamed site at the *other* end of the deleted edge 'free'. This, however, is *emphatically not* the intended action of the rule applied, as it properly should be, to a realizable graph: the action then is to unbind `s` from whatever (unique) thing it is bound to and make the sites at *both* ends[¶] explicitly 'free'. In other words, the *intended* meaning of this rule convolves pure graph rewriting aspects with the constraint of realizability which insists that all sites maintain a well-defined binding state.

The approach in [28] amounted to performing an upgrade of the meta-model of [27], with additional explicit structures for binding wild-cards (and binding-types) and the extra constraints they necessitated, then forcing the (SPO) rewriting process to maintain these constraints—even when it really had no business doing so. The resulting framework is, not unexpectedly, complicated and fragile; however, its real weakness is that, in doing this, it becomes rather a mere *transcription* of Kappa into bespoke-tailored graphs than an insightful re-situating in a more general mathematical setting.

One possible approach to easing this criticism would be not to enforce the maintenance of constraints but rather include, in the rule, a *post hoc* 'application condition' intended to be satisfied *after* applying the rule; and, if not, this should induce an automatic *administrative*[‖] rewrite to reestablish any violated constraints. This is actually how the operational semantics of Kappa [21] deals with the issue, albeit in a term-rewriting setting. However, NACs (and PACs for the dual encoding of the `!_` wild-card) can achieve the same effect using, moreover, a far simpler notion of graph so, in the absence of any other motivating need, there seems little point in doing this.

[¶]Such *side-effects* also arise through deletion of an agent bound on some of its sites.

[‖]cf. administrative reductions in CPS

II Causality, activation and inhibition

The first account of how 'pathways', *i.e. causal* traces, could be extracted from the execution traces of a rule-based simulation appeared in [17]. The treatment was largely informal, being based on the empirical use of (an early version of) the Kappa simulator in the context of a reasonably large model, built by myself, as a way of providing a proof of concept. One of the key (meta-)findings was the need for the automatic *compression* of causal traces in order to remove irrelevant digressions[**] due to binding equilibria and futile cycles of modification.

A second key point of the paper was the use of activation and inhibition between rules to characterize influences *between* causal traces. Most notably, these took the form of feed-back and/or feed-forward within, or between, causal traces. However, the notions of activation and inhibition were defined in a state-dependent fashion—'there exists a state in which the firing of this rule creates/destroys instances of that rule'—which were only later related formally to the state-independent definitions in terms of multi-sums.

The real purpose of [28] was not to provide a definitive interpretation of Kappa rules but to give the first rigorous treatment of this extraction—from execution traces, as formalized with SPO rewriting[††]—of causal traces and their compression in order to obtain something close to what biologists call 'signalling pathways'. As such, the paper critically needed to take wild-cards and, more generally, side-effects into account and, although application conditions (mainly NACs) could have been employed for this purpose, the explicit reification of structures for wild-cards into the graphs was in practice rather more direct and convenient.

In addition, although it was not treated explicitly in the paper, a second motivation of [28] was to have a graphical interpretation of rules suitable for the design of static analyses: in that case, far from wanting a general mathematical universe within which to situate, and investigate the possible generalizations, of a class of rules, for static analysis we seek a finely-tuned, highly specific interpretation with which we can extract the most information possible [23].

[**]The causal traces presented in the paper were compressed *de facto* by manipulating the system's state and ignoring any non-compressed traces that still arose.

[††]In this paper, execution traces were not described à la 'implicit state simulation' but just in pedestrian fashion as a sequence of matchings of LHSs into the current system state. A reformulation in terms of witnesses could provide further insight.

A completely different application of activation and inhibition arose from the observation that they allow us to characterize all ways in which a graph, a pattern, can be produced and consumed by the action of a collection of rules. Indeed, one can use activation and inhibition to define an ODE describing the limiting trajectory[‡‡] of a pattern; however, this ODE must keep track of the entire system state. In earlier work, J. Feret showed [21,34] how to compute, via an abstraction interpretation, a finite set of coarse-grained patterns—called *fragments*—giving rise to a *self-consistent* description of their limiting trajectories: the ODE for each fragment requires access only to the current values of (some of) the other fragments. In this way, the dependence on the whole system state is suppressed and the resulting description generally gives rise to a significant reduction in the number of variables.

This idea of computing a self-consistent coarse-graining, relative to a given set of rules and observables, was revisited through the lens of the abstract categorical combinatorics of production and consumption provided by the multi-sum formulation of activation and inhibition [37]. The setting was a simplified Kappa with no internal states and no wild-cards—although with hindsight there is no reason why these could not be incorporated—with the further key assumption that the contact graph of the system is acyclic. This enabled a description of the (finite) set of fragments and their ODEs via a *saturation* procedure whose termination was implied by acyclicity. This assumption was suppressed in the follow-up paper [26] at the cost of requiring a less computationally intuitive mathematical definition of (in general) an infinite set of fragments and their associated ODEs.

These methods complement those based on abstract interpretation: the abstract combinatorics provides a mathematically optimal coarse-graining; whereas static analysis[§§] seeks to efficiently compute a good approximation to the optimal solution.

These papers made use of several fairly idiosyncratic notions in order to formulate the abstract combinatorics: production and consumption squares, minimal gluings and relative push-outs. However, ultimately, all the abstract machinery required follows from the existence of all multi-sums in the sub-category of monos; in particular, the notions of production and consumption squares correspond directly to the definitions of activation and inhibition and a minimal gluing is a restatement of the notion of an element of a multi-sum.

[‡‡] *i.e.* in the thermodynamic limit

[§§] see F. Camporesi's forthcoming PhD thesis for a definitive account

III Towards knowledge representation

The 'proof of concept' Kappa model in [17] rapidly acquired a double status as being, on the one hand, the *source code* of an executable model; yet, on the other hand, somehow—and despite a clear dose of arbitrary encoding—a representation of the domain-specific (biological) knowledge upon which the model is based. As the model was expanded[¶¶], certain commonalities began to emerge between rules: in effect, some rules felt like they could be expressed at a more generic level and then instantiated into various concrete cases. The motivations for this were partly pragmatic—keeping the model smaller—but also derived from biological arguments: many PPIs have generic character as they involve regions of proteins conserved across orthologs [between species] and paralogs [between distinct genes of a single species].

In collaboration with J. Feret, I designed and built a definitional meta-language for Kappa rules, called MetaKappa, in which the user could define (forest) hierarchies of agents and rules that could refer to agents at any level of the hierarchy [20]. The sites of an agent could be duplicated or deleted in its descendants; so a rule written at one level might be duplicated for some of its children and disappear completely for others. Given a hierarchy and a collection of rules, MetaKappa automatically translated into Kappa[***]. The ability, through the deletion of sites, to express the effects of loss-of-function (LOF) mutations enabled a new use of static and causal analysis: for a simple class of 'reachability' assertions[†††] we could use these analyses to validate our proof of concept model against empirical data [20] that precisely exploits engineered LOF mutations in order to characterize the necessary conditions for PPIs.

A rather paradoxical effect of this *definitional* approach was that it forced the user to keep in mind the 'intended' Kappa rules; as such, it still required the bulk of the actual modelling work to be carried out 'in the head' of the user and only intervened, after the fact, in order to provide a more humanly-readable presentation of the model. Moreover, a given collection of intended rules can—in the absence of other constraints—generally be expressed in multiple ways, even in a simple framework like MetaKappa, so the approach suffers from an intrinsic lack of canonicity.

[¶¶] ultimately to roughly 300 rules

[***] This was not done using graph rewriting although, with hindsight, it would have been very natural to formulate MetaKappa in these terms.

[†††] can this pattern be built? and, if so, how?

The follow-up paper [36] continued in this definitional vein, allowing even more powerful and even less canonical definitions by generalizing hierarchies to being DAGs. In hindsight, this was a completely wrong turn. However, the very fact that this more powerful framework inevitably enabled many more ways of defining the same collection of Kappa rules brought into sharper focus the deeper underlying opposition between *specification* and *definition*: a specification-based approach simply lists, in an unstructured fashion, all the cases; whereas a definitional approach seeks to find some regularity. The ultimate aim of the latter approach is lofty and amounts to being an attempt to provide a preliminary *theory* of PPIs: not merely documenting what we know but trying to predict what should be the case. With all due respect to the progress made in recent decades in biochemistry, such a theory remains a distant hope. As such, and despite its being the most initially tempting to a computer scientist, it seems that the definitional approach is, at best, extremely premature.

My recent work has reverted to the specification-based approach—as was initially implicit in the use of Kappa to 'represent knowledge'—but in a graph-based setting [6]. The aim is simply to represent knowledge and, to this end, employs a domain-specific meta-model. As before, automatic translation into Kappa is provided. The graph-based formalism of [6] falls squarely[‡‡‡] into the rule-based meta-modelling framework—although it was not presented in that way—and possesses considerable, as yet untapped, additional representational power for capturing phenomena such as gain-of-function mutations and negative constraints (such as allosteric inhibition)[§§§]. This work relies heavily on the `ReGraph` library which is still under active development. Some future directions for this line of work are explored in the next chapter.

[‡‡‡]indeed, this marked the birth of the very idea

[§§§]This extension of [6] will be presented in a forthcoming paper.

Project

Context

I have already begun work on the bio-curation problem, in the context of DARPA's 'Big Mechanism' and 'Communicating with Computers (bio-curation use case)' programmes, in collaboration with W. Fontana at the Harvard Medical School. So far, this has focused on a general approach to solving the aggregation problem using graph rewriting relative to a domain-specific *meta-model* defining precisely the kinds of information of interest [6]. A prototype system KAMI* has been built to enable the incremental collation of knowledge, its aggregation and automatic translation into the rule-based modelling language Kappa [17] for the purposes of simulation†.

More recent work has begun to formalize key domain-specific properties of our meta-model so as to support a *semantically normalized* representation of proteins. Similar work has been done by others in the past, most notably in the BioPAX project [33], but the particularity and principal novelty of our approach is to restrict our domain of interest to signalling while insisting on a more detailed representation of proteins so that the effects of mutations on PPIs can be automatically accounted for in the translation into Kappa.

An important further consequence of semantic normalization is to ease the integration of separate human curation projects into a larger whole: until the human bottleneck of 'high-throughput curation' has been resolved, large-scale curation can only realistically occur through the merging of smaller-scale human-led efforts. Such merges may very well yield an inconsistent whole—typically different versions of the same PPI—but normalization would at least guarantee, and indeed identify, that the different versions refer to, and are *intended* to refer to, the same entities.

*__K__nowledge __A__ggregator & __M__odel __I__nstantiator: `https://github.com/Kappa-Dev/Kami`

†KaSim: `http://dev.executableknowledge.org`

I Multiple models

short- to medium-term

The first axis of my project aims to address this issue of inconsistency and, more generally, the need to *support and evaluate* multiple models. The first step is to provide a robust infrastructure that enables KAMI to maintain distinct, but interlinked, models by making explicit their common parts; and to enable the evolution of those *linkages* as the models are updated over time. This would enable (i) small-scale variants of a single model where some of the PPIs have several versions; and (ii) more significant variant models that propose alternative collections of PPIs to explain some experimental observations; but also (iii) connections between models for different species (*e.g.* human vs. mouse) that formalize evolutionary *homology* relations.

The first case would provide machine support for keeping track of small variants, such as those that could arise from merging, that would otherwise be the modeller's responsibility; while the second extends this to more substantial variant models—for different cell types, *e.g.* skin vs. liver, employing different collections of PPIs; or proposing alternative hypotheses—so as to compare and contrast competing explanations of experimental data.

The third case exploits the same underlying idea—connecting two models by specifying their common parts—with a rather different aim: to formalize the notion of knowledge transfer 'by similarity' frequently used by biologists. Specifically, if some PPI has been experimentally observed in one species, *e.g.* mouse, and the (relevant regions of the) proteins in question are conserved in (say) humans then, *by similarity* and in the absence of data to the contrary, we would hypothesize that 'the same' PPI occurs in humans. The linkage between the mouse and the human model defines precisely what we mean by 'the same'. This formalization of knowledge transfer implies that we also need to augment our platform with the ability to assign epistemic status to PPIs (as discussed briefly above) so as to distinguish between 'experimentally observed fact', 'inferred by similarity' and 'pure hypothesis'.

This first step must be accompanied by a significant use case designed to test the above functionality. A good candidate would be the development of a collection of models of the activation of [the protein] Raf, recently reviewed in [48]. Raf activation is a highly complex, and only partially understood, process involving a number of proteins and PPIs. Its literature is highly complicated, with multiple competing hypotheses, and the data have been gathered across many human cell types as well as in other species.

As such, the curation of this literature would potentially benefit greatly from precisely the kind of infrastructure we are proposing to build; and should moreover usefully inform the very design of that infrastructure. Furthermore, Raf protein mutations are implicated in certain cancers, notably melanoma, so this use case has further interest for the KAMI system as a whole.

The second step is concerned with enhancing this infrastructure to provide support for the *evaluation* of the explanatory power of models. As mentioned above, a great deal of knowledge takes the form of assertions that apply in a certain context, *i.e.* a particular cell type of a particular species; these may be compatible with some models and incompatible with others. I plan to formalize assertions within KAMI, by generalizing its meta-model, so that they can be evaluated in an automated fashion via the Kappa simulator: if (say) an assertion claims that treating cells with a certain hormone leads to Raf activation, this scenario can be 'run' across all appropriate variant models in order to identify which ones validate the assertion (and which ones don't). The exact class of assertions we need remains an empirical question at this time; however, since KaSim provides full-blown stochastic simulation, it could in principle be used to validate quite sophisticated assertions about the timing and the degree of activation of a signal transduction pathway.

The completion of the above would already provide a powerful platform for bio-curation, allowing the user to construct plausible variant models in an organized fashion, transfer knowledge across species and perform semi-automated evaluation of their relative explanatory power. However, the third (and, for this part of my project, final) step should take the platform one step further to provide support for distillation of *consensus* models for species by the gradual merging of smaller models into a larger-scale, but still human-directed, curation effort. Specifically, differences in behaviour between (say) skin and liver cells only arise because the cells express different repertoires of proteins; their behaviour is obviously blind to whether they live in a skin or a liver cell[‡]. So, ultimately, a human skin model is one instantiation of a *single* consensus human model; and a human liver model is a different instantiation of that same consensus model. The identification of such consensus models hinges on a *decontextualization* process that prunes spurious, typically cell type-specific, conditions on PPIs, leading to a canonical structure for KAMI consisting of a collection of consensus species models, each connected to its variants for cell types (and any outstanding competing hypotheses).

[‡]cf. the philosophical aside at the end of the first chapter

Technical approach

The approach is grounded in the mathematical formalism of graphs and graph rewriting [6]. Each PPI is represented by a graph detailing the nature of the interaction—usually binding, unbinding or enzymatic modification—, the participant proteins and any further necessary conditions, *e.g.* one of the proteins must be phosphorylated in some way and the other must be bound to some 'chaperone' protein. A model consists of a collection of such formalized PPIs together with additional disambiguating information specifying when actions are in conflict, *e.g.* a protein can engage in two binding actions but only one-at-a-time because the two binding sites overlap physically. This additional information is itself represented as a graph. Updates to a PPI are expressed by applying rewriting rules that modify the necessary conditions. This can be either the removal of conditions newly revealed to be spurious; or the addition of newly discovered conditions.

Mathematically speaking, a *linkage* between two graphs is precisely a rewriting rule from one to the other: it expresses what they have in common; and what must be removed and added from the one to obtain the other. If we rewrite one of the graphs with some second rewriting rule, this induces a canonical rewrite on the linkage itself which, in turn, induces a canonical rewrite on the other graph. This fundamental formal choreography provides the underpinnings for how we propose to represent and maintain multiple evolving model variants: it is enough to represent one graph explicitly—and in non-volatile fashion so we will require some database infrastructure, based on RDF or similar, well-matched to our graphical formalism—and the others implicitly as 'rewrites' of the first; in practice, the explicitly represented graph should ideally be our current best candidate for consensus model. The same notion of linkage can be used to represent conserved regions of proteins, *i.e.* homology relations, and, in this case, the implied rewrites correspond exactly to our desired notion of knowledge transfer between species.

All our graphs are constrained in form by our choice of meta-model. In order to accommodate assertions, we will generalize the existing meta-model to allow the representation of 'actions at a distance' that correspond to what biologists call *pathways*. In other words, such an assertion could represent a statement such as 'hormone X leads to Raf activation' which does not correspond to any direct notion of causality; on the contrary, there can be a complicated chain of PPIs required to fulfil the claim. There could even be several distinct ways by which hormone X leads to Raf activation.

As mentioned previously, the KaSim simulator, in addition to providing execution traces, can also perform post hoc causal analysis of traces in order to detect the existence of such causal chains between processes. We would consider the discovery of such a chain to be a validation of the corresponding assertion. We can complement this with static analysis, based on abstract interpretation, which can establish the definite non-existence of any such explanatory causal trace. Although potentially computationally expensive, in principle these analyses enable the automated 'screening' of many variant models with respect to a collection of assertions.

The greatest point of difficulty in this project lies in the elucidation of the decontextualization process. The basic idea is to excise—from PPIs and from assertions—details that have proven to be spurious. For example, in one cell type, a necessary phosphorylation in the middle of some pathway might occur constitutively whereas, in a different cell type, it might be regulated by some other process and, in some third cell type, by yet another process. Depending on which context one uses to curate, one might begin by assuming that those auxiliary processes themselves, not just their ultimate phosphorylation action, belong to the overall pathway. This would then be contradictory with the first cell type where no such auxiliary process occurs. Decontextualization would remove these spurious dependencies, replacing them with INUS[§]-style causal relations within and between pathways. The proper formalization and automation of this procedure will have to be informed by our experience of first applying the ideas—by hand—in our use case.

II Automation

medium- to long-term

The second axis of my project aims to build an automated curation pipeline. More precisely, it seeks to sharpen the requirements that we make on machine reading that are indispensable for the specific notion of bio-curation discussed in this manuscript. The basic problem can be stated in rigorous semantic terms[¶] as follows: given an input to the bio-curation machine and given the current state of that machine, *calculate* the denotation of that input, *i.e.* the rewriting rule that must be applied to update the current state—in the light of what the input contains.

[§]'Insufficient but Necessary, Unnecessary but Sufficient', due to J. L. Mackie

[¶]as in our earlier discussion of game semantics

At the time of writing, the current status of the project is: we have built generic machinery for performing such rewriting; and we have identified the kinds of entities and edges that can exist in a state of the specific bio-curation machine[‖]. The former is provided by the general purpose `ReGraph` Python library discussed in the previous chapter; the latter by the definition of a specific meta-model, *i.e.* a particular graph that types all other graphs in the machine, that expresses a collection of concepts relevant to bio-curation for cellular signalling. On the other hand, the calculation of the denotation of an input must still be performed *in the head* of the user of the machine; and, at present, the *incarnation* of that rewriting rule in the machine can only occur through the user *transcribing* it via the agency of a user interface.

The automation of this task can only realistically be tackled incrementally and will require considerable introspection on how, as humans, we calculated these updates during the development of the use case. Certain aspects of the problem fall naturally into the category of 'background knowledge' that any expert bio-curator could be expected to know, *e.g.* the name and standard symbols for amino acids, their chemical properties pertinent to signalling and standard notation for mutations; or more sophisticated properties built on top of this such as 'only serine, threonine and tyrosine residues can be phosphorylated by an active kinase domain'.

Further *semantic* properties specific to signalling—*i.e.* with respect to the corresponding real-world entities: the domain-specific semantics—such as the binding properties of the various modular 'binding domains' would also play an important rôle; as would more pedestrian (typically syntactic) notions such as synonyms and near synonyms, *e.g.* 'bind' verses 'associate' or 'form a complex'. Ultimately, the determination of all relevant factors can only be considered an open-ended empirical question.

Technical approach

Although we are evidently a long way from calculating denotations with a program written in the internal language of some highly-structured category, this rigorous framing of the problem does at least provide some conceptual scaffolding that will hopefully prevent the project from degenerating into an imbroglio of 'chewing gum and string'.

[‖]There is also the automatic translation into Kappa rules for simulation and analysis but that is not germane to this discussion.

It is already clear that we need to write a (complicated) program; the above discussion also makes it clear that its behaviour must depend on the current state of the machine: not only does the update need to be *situated*—we must calculate not just a rewriting rule but also its *matching*, *i.e.* where it is to be applied—but the denotation itself, and potentially even the *way* it is calculated, will also vary depending on the current state. For (a simple) example, the denotation of a sentence mentioning some entity, *e.g.* a protein, differs depending on whether or not that entity is already represented in the machine: we only wish to add entities that do not already exist.

As far as 'anatomical' aspects of proteins are concerned, a great deal of ground work has already been done, by the bioinformatics community, in providing databases that *ground* entities: UniProt gives unique identifiers for genes and documents their protein products; PFAM details regions and domains of proteins; and PhosphoSite has a lot of detail on post-translational modifications (primarily phosphorylation) relevant to signalling. In all likelihood, it will be necessary to complement these concrete groundings with additional *phenomenological* properties, typically enzymatic 'activity', that are sufficiently systematic to be incorporated into an *ontology*. As suggested above, the combination of such properties—either by hand in *ad hoc* fashion; or more systematically using an OWL Description Logic-based ontology [5]—already provides the basis for a step of [domain-specific] semantic validation: if an input sentence states non-sense, we want to detect this, if at all possible.

The situation is far less clear when it comes to PPIs: to date, no standard grounding exists**. In consequence, when an input sentence speaks of some interaction, we cannot simply 'look up' whether or not it already exists in our machine's representation. Instead, we can only rely on domain-specific semantic properties so as to identify (at least a sub-class of) the cases where the interaction does already exist in the machine, *e.g.* sequence constraints or the *localization* of an interaction at some protein domain. We are not aware of any previous efforts of this kind and, as such, the early development will likely have a rather artisanal character. However, we can hope that broader principles might begin to emerge over time.

In any case, once we have identified what is new and what already existed in the machine, given an input sentence, we can construct its denotation—the rewriting rule and its matching—and subsequently apply it in order to update the machine's state.

**Indeed, a notable side-effect of this project is precisely to provide such a grounding.

One final point concerns syntactic issues. In the early phase of the project, and as tacitly assumed above, we do not intend to perform syntactic parsing ourselves; we will rather provide the system with pre-annotated sentences in order to focus our initial attention on developing domain-specific semantic aspects. However, as the system matures, we intend to incorporate syntactic parsing and an even earlier phase of machine learning in order to detect which sentences we actually need to read: the biological literature is highly stylized and most sentences fall into one (or sometimes two) of only a small number of 'types' of sentences, only some of which ever contain actionable content.

Summary

My project has two broad and complementary axes: the first is primarily concerned with aspects of bio-curation that exploit having a human expert 'in the loop'; whereas the second aims to increase the overall throughput of curation by incorporating machine reading technology.

In the longer term, this project could evolve towards the development of a very general (rule-based) meta-modelling [41] framework. This could include abstracting the Kappa-specific simulator to being meta-model independent, thus enabling the user to execute directly any collection of rules written over their own meta-model. Such a framework would enable more flexible representations for biology—*e.g.* by capturing pertinent notions of *locality*, without going all the way to 3d-coordinates, if diffusion-limited or membrane-associated processes are implicated; or to coordinate multi-level modelling to capture the effects of signalling at the tissue organization level—or, indeed, for a wide variety of modelling domains.

An entirely different generalization could focus instead on the linguistic side in order to extend its scope, beyond that of bio-curation, towards other target domains or even general semantics. This might enable the forging of connections with work that seeks to analyse the semantics of natural language via 3d temporal simulations [57, 58] to capture context-specific aspects of semantics that natural language does not (explicitly) express[††]. In particular, the dependence of such work on 'Cartesian 3d-space' could potentially be suppressed by the use of a meta-model-independent simulation machinery. As such, there should be scope for synergy with general meta-modelling.

On verra bien.

[††]cf. the use of Kappa to investigate the cell type-dependence of a collection of PPIs

Bibliography

[1] S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *Proceedings, 13th Annual IEEE Symposium on Logic in Computer Science*, pages 334–344. IEEE, 1998.

[2] S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.

[3] S. Abramsky and G. McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions. In *Algol-like languages*, pages 297–329. Springer, 1997.

[4] S. Awodey. *Category theory*. Oxford University Press, 2010.

[5] F. Baader *et al. The Description Logic handbook: theory, implementation and applications*. Cambridge University Press, 2003.

[6] A. Basso-Blandin, W. Fontana, and R. Harmer. A knowledge representation meta-model for rule-based modelling of signalling networks. *EPTCS*, 204:47–59, 2016.

[7] S. D. Brookes and A. W. Roscoe. An improved failures model for communicating processes. In *International Conference on Concurrency*, pages 281–305. Springer, 1984.

[8] P. Clairambault. *Logique et interaction: une étude sémantique de la totalité*. PhD thesis, Université Paris-Diderot-Paris 7, 2010.

[9] P. Clairambault and R. Harmer. Totality in arena games. *Annals of pure and applied logic*, 161(5):673–689, 2010.

[10] P. R. Cohen. DARPA's Big Mechanism program. *Physical biology*, 12(4):045008, 2015.

[11] T. Coquand. A semantics of evidence for classical arithmetic. *Journal of Symbolic Logic*, 60:325–337, 1995.

[12] A. Corradini, D. Duval, R. Echahed, F. Prost, and L. Ribeiro. Agree–algebraic graph rewriting with controlled embedding. In *International Conference on Graph Transformation*, pages 35–51. Springer, 2015.

[13] A. Corradini, T. Heindel, F. Hermann, and B. König. Sesqui-pushout rewriting. In *International Conference on Graph Transformation*, pages 30–45. Springer, 2006.

[14] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation, part I: Basic concepts and double pushout approach. In *Handbook of Graph Grammars*, pages 163–246, 1997.

[15] P.-L. Curien. *Categorical combinators, sequential algorithms, and functional programming*. Birkhaüser Boston, 1993.

[16] P.-L. Curien. Definability and full abstraction. *Electronic Notes in Theoretical Computer Science*, 172:301–310, 2007.

[17] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. *CONCUR 2007–Concurrency Theory*, pages 17–41, 2007.

[18] V. Danos, J. Féret, W. Fontana, R. Harmer, and J. Krivine. Investigation of a biological repair scheme. In *International Workshop on Membrane Computing*, pages 1–12. Springer, 2008.

[19] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling, symmetries, refinements. In *Formal methods in systems biology*, pages 103–122. Springer, 2008.

[20] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling and model perturbation. In *Transactions on Computational Systems Biology XI*, pages 116–137. Springer, 2009.

[21] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Abstracting the differential semantics of rule-based models: exact and automated model reduction. In *Logic in Computer Science (LICS), 2010 25th Annual IEEE Symposium on*, pages 362–381. IEEE, 2010.

[22] V. Danos, J. Feret, W. Fontana, and J. Krivine. Scalable simulation of cellular signaling networks. In *Asian Symposium on Programming Languages and Systems*, pages 139–157. Springer, 2007.

[23] V. Danos, J. Feret, W. Fontana, and J. Krivine. Abstract interpretation of cellular signalling networks. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 83–97. Springer, 2008.

[24] V. Danos and R. Harmer. The anatomy of innocence. In *International Workshop on Computer Science Logic*, pages 188–202. Springer, 2001.

[25] V. Danos and R. Harmer. Probabilistic game semantics. *ACM Transactions on Computational Logic (TOCL)*, 3(3):359–382, 2002.

[26] V. Danos, R. Harmer, R. Honorato-Zimmer, and S. Stucki. Deriving rate equations for site graph rewriting systems. In *To appear in Electronic Notes in Theoretical Computer Science*, 2014.

[27] V. Danos, R. Harmer, and G. Winskel. Constraining rule-based dynamics with types. *MSCS*, 23(2):272–289, 2013.

[28] V. Danos, Feret J., W. Fontana, R. Harmer, J. Hayman, J. Krivine, C. Thompson-Walsh, and G. Winskel. Graphs, Rewriting and Pathway Reconstruction for Rule-Based Models. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012)*, volume 18 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 276–288. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2012.

[29] O. Danvy and A. Filinski. Representing control: A study of the CPS transformation. *Mathematical structures in computer science*, 2(04):361–391, 1992.

[30] Y. Diers. *Familles universelles de morphismes*, volume 145 of *Publications de l'U.E.R. mathématiques pures et appliquées*. Université des sciences et techniques de Lille I, 1978.

[31] R. Dyckhoff and W. Tholen. Exponentiable morphisms, partial products and pullback complements. *Journal of Pure and Applied Algebra*, 49(1-2):103–116, 1987.

[32] H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. Algebraic approaches to graph transformation, part II: Single pushout approach and comparison with double pushout approach. In *Handbook of Graph Grammars*, pages 247–312, 1997.

[33] E. Demir *et al.* The BioPAX community standard for pathway data sharing. *Nature biotechnology*, 28(9):935–942, 2010.

[34] J. Feret, V. Danos, J. Krivine, R. Harmer, and W. Fontana. Internal coarse-graining of molecular systems. *Proceedings of the National Academy of Sciences*, 106(16):6453, 2009.

[35] R. Harmer. *Games and full abstraction for non-deterministic languages*. PhD thesis, Imperial College of Science, Technology and Medicine, 1999.

[36] R. Harmer. Rule-based modelling and tunable resolution. In S. B. Cooper and V. Danos, editors, 5th Workshop on Developments in Computational Models — *Computational Models From Nature*, volume 9 of *Electronic Proceedings in Theoretical Computer Science*, pages 65–72. Open Publishing Association, 2009.

[37] R. Harmer, V. Danos, J. Feret, J. Krivine, and W. Fontana. Intrinsic information carriers in combinatorial dynamical systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(3):037108, 2010.

[38] R. Harmer, M. Hyland, and P.-A. Mellies. Categorical combinatorics for innocent strategies. In *22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*, pages 379–388. IEEE, 2007.

[39] R. Harmer and O. Laurent. The anatomy of innocence revisited. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 224–235. Springer, 2006.

[40] R. Harmer and G. McCusker. A fully abstract game semantics for finite nondeterminism. In *Logic in Computer Science, 1999. Proceedings. 14th Symposium on*, pages 422–430. IEEE, 1999.

[41] B. Henderson-Sellers. *On the mathematics of modelling, metamodelling, ontologies and modelling languages*. Springer Briefs in Computer Science, 2012.

[42] H. Herbelin. Games and weak-head reduction for classical PCF. In *International Conference on Typed Lambda Calculi and Applications*, pages 214–230. Springer, 1997.

[43] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, R. G. Posner, M. Hucka, and W. Fontana. Rules for modeling signal-transduction systems. *Sci STKE*, 2006(344):re6, 2006.

[44] M. Hofmann and T. Streicher. Continuation models are universal for $\lambda\mu$-calculus. In *Logic in Computer Science, 1997. LICS'97. Proceedings., 12th Annual IEEE Symposium on*, pages 387–395. IEEE, 1997.

[45] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. *Information and Computation*, 163(2):285–408, 2000.

[46] S. Lack and P. Sobociński. Adhesive categories. In *International Conference on Foundations of Software Science and Computation Structures*, pages 273–288. Springer, 2004.

[47] J. Lambek and P. J. Scott. *Introduction to higher-order categorical logic*. Cambridge University Press, 1988.

[48] Hugo Lavoie and Marc Therrien. Regulation of RAF protein kinases in ERK signalling. *Nature Reviews Molecular Cell Biology*, 16(5):281–298, 2015.

[49] T. Leinster. *Basic category theory*. Cambridge University Press, 2014.

[50] R. Loader. Unary PCF is decidable. *Theoretical Computer Science*, 206(1):317–329, 1998.

[51] R. Loader. Finitary PCF is not decidable. *Theoretical Computer Science*, 266(1):341–364, 2001.

[52] S. Mac Lane. *Categories for the working mathematician*. Springer-Verlag, 1998.

[53] G. McCusker. Games and definability for FPC. *Bulletin of Symbolic Logic*, 3(03):347–362, 1997.

[54] R. Milner. *Communication and concurrency*. Prentice Hall, 1989.

[55] E. Murphy, V. Danos, J. Feret, R. Harmer, and J. Krivine. Rule-based modeling and model refinement. *Elements of Computational Systems Biology*, pages 83–114, 2009.

[56] H. Nickau. Hereditarily sequential functionals. In *International Symposium on Logical Foundations of Computer Science*, pages 253–264. Springer, 1994.

[57] J. Pustejovsky. Dynamic event structure and habitat theory. In *6th International Conference on Generative Approaches to the Lexicon*, pages 1–10. ACL, 2013.

[58] J. Pustejovsky and N. Krishnaswamy. Generating simulations of motion events from verbal descriptions. *Lexical and Computational Semantics*, pages 99–109, 2014.

[59] R. Rosen. On models and modeling. *Applied Mathematics and Computation*, 56(2):359–372, 1993.

[60] M. Schmidt-Schauß. Decidability of behavioural equivalence in unary PCF. *Theoretical Computer Science*, 216(1):363–373, 1999.

[61] B. Schoeberl, C. Eichler-Jonsson, E. D. Gilles, and G. Müller. Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. *Nature biotechnology*, 20(4):370–375, 2002.

[62] P. Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(02):207–260, 2001.

[63] T. Streicher and B. Reus. Classical logic, continuation semantics and abstract machines. *Journal of functional programming*, 8(06):543–572, 1998.

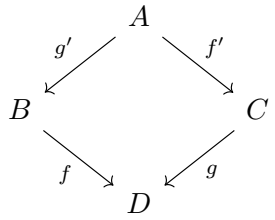
[64] X. Zhang, J. Gureasko, K. Shen, P. A. Cole, and J. Kuriyan. An allosteric mechanism for activation of the kinase domain of epidermal growth factor receptor. *Cell*, 125(6):1137–1149, 2006.

Appendix

I Slice categories

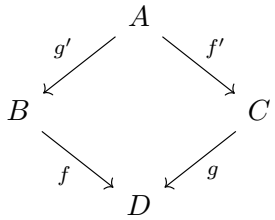
Given a category $\mathbf{C}$ and an object T, the slice category $\mathbf{C}/T$ has, as objects, all arrows of the form $f : X \to T$ where X is any object of $\mathbf{C}$, and, as arrows from $f : X \to T$ to $g : Y \to T$, all arrows $h : X \to Y$ of $\mathbf{C}$ such that $f = g \circ h$.

A commuting square



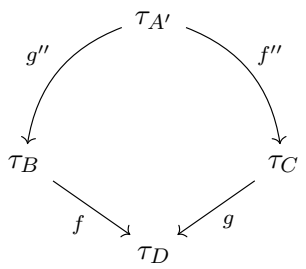
in $\mathbf{C}$ and an arrow $\tau_D : D \to T$ uniquely determines a commuting square in $\mathbf{C}/T$ (by setting $\tau_B := \tau_D \circ f$, $\tau_C := \tau_D \circ g$ and $\tau_A := \tau_B \circ g' = \tau_C \circ f'$). This clearly extends to a bijection between commuting squares in $\mathbf{C}/T$ and pairs consisting of a commuting square in $\mathbf{C}$ and an arrow $\tau_D : D \to T$ from its sink object to T, *i.e.* all commuting squares in $\mathbf{C}/T$ arise in this way.

Lemma 1. *The bijective correspondent of a pull-back*

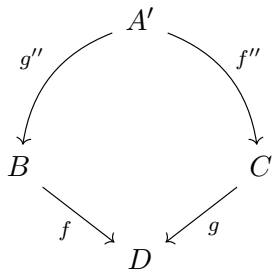


in $\mathbf{C}$ *and an arrow* $\tau_D : D \to T$ *is a pull-back in* C/T.

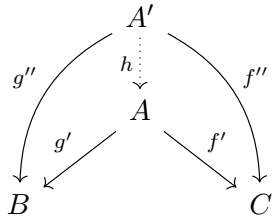
Proof. Any span $\tau_B \xleftarrow{g''} \tau_{A'} \xrightarrow{f''} \tau_C$ in $\mathbf{C}/T$ for which



commutes gives rise, by the above bijection, to a commuting square in $\mathbf{C}$



and, by the universal property of the pull-back in $\mathbf{C}$, we have a unique $h : A' \to A$ in $\mathbf{C}$ making



commute. Since $\tau_{A'} = \tau_B \circ g'' = \tau_B \circ g' \circ h = \tau_A \circ h$, we can conclude that h is also an arrow of $\mathbf{C}/T$. □

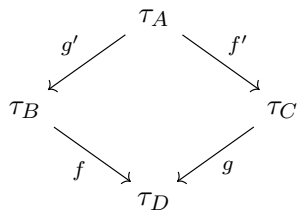
This is already enough to establish that the property of $\mathbf{C}$ having all pull-backs carries over to $\mathbf{C}/T$.

Corollary 2. *If* $\mathbf{C}$ *has all pull-backs then so does* $\mathbf{C}/T$.

Proof. A co-span in $\mathbf{C}/T$ determines a co-span in $\mathbf{C}$ whose pull-back in $\mathbf{C}$, by Lemma 1, determines a pull-back in $\mathbf{C}/T$. □

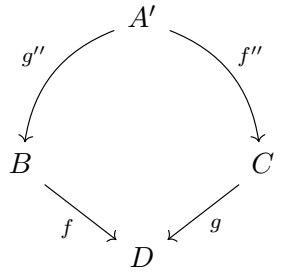
Analogous to the case for commuting squares, there is a 'converse' to lemma 1 that establishes that the bijection restricts to pull-backs, *i.e.* all pull-backs in $\mathbf{C}/T$ arise in this way.

Lemma 3. *The bijective correspondent of a pull-back*

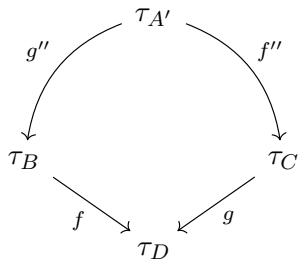


in $\mathbf{C}/T$ *is a pull-back in* $\mathbf{C}$*.*

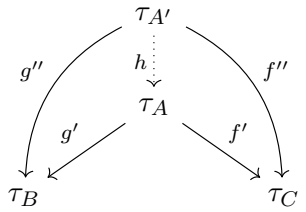
Proof. Any span $B \xleftarrow{g''} A' \xrightarrow{f''} C$ in $\mathbf{C}$ for which



commutes gives rise to a commuting square in $\mathbf{C}/T$ where $\tau_{A'} := \tau_B \circ g''$:

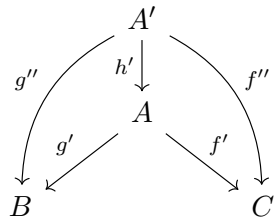


By the universal property of the pull-back in $\mathbf{C}/T$, there is a unique $h : \tau_{A'} \to \tau_A$ in $\mathbf{C}/T$ making



commute.

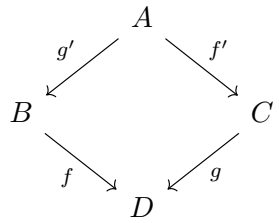
We can conclude by noting that any arrow $h' : A' \to A$ in $\mathbf{C}$ making



commute can be construed as an arrow of $\mathbf{C}/T$ from $\tau_{A'}$ $[\tau'_{A'} := \tau_B \circ g' \circ h' = \tau_B \circ g'' = \tau_{A'}]$ and so, by the universal property of h in $\mathbf{C}/T$, we have $h' = h$ which transfers that universal property of h back to $\mathbf{C}$. □

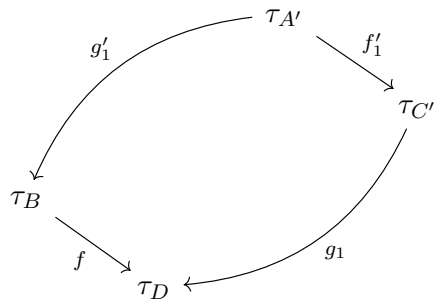
This completes the proof that the bijective correspondence—between pairs consisting of a commutative square in $\mathbf{C}$ and an arrow from its sink object to T and commuting squares in $\mathbf{C}/T$—restricts to pull-backs. We make use of the second half of this result in the following lemma that implies that all pull-back complements in $\mathbf{C}$ transfer to $\mathbf{C}/T$.

Lemma 4. *A pull-back complement*

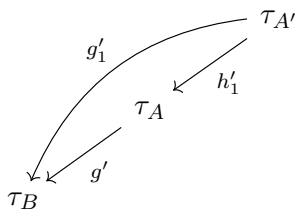


in $\mathbf{C}$ *and arrow* $\tau_D : D \to T$ *uniquely determines a pull-back complement in* C/T.

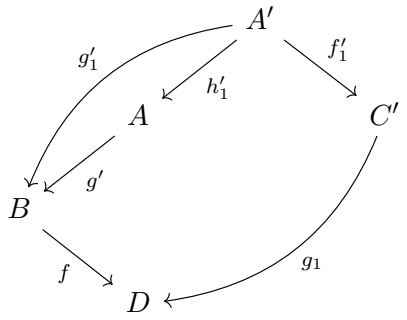
Proof. Any pull-back



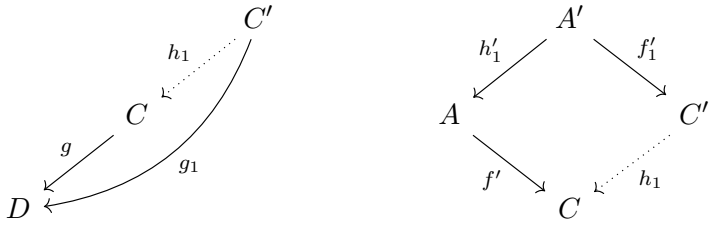
together with any arrow $h'_1 : \tau_{A'} \to \tau_A$ in $\mathbf{C}/T$ such that



commutes gives rise, by lemma 3, to a pull-back in $\mathbf{C}$ together with a factorization $g'_1 = g' \circ h'_1$



and, by the universal property of the pull-back complement in $\mathbf{C}$, we have a unique arrow $h_1 : C' \to C$ in $\mathbf{C}$ such that



commute.

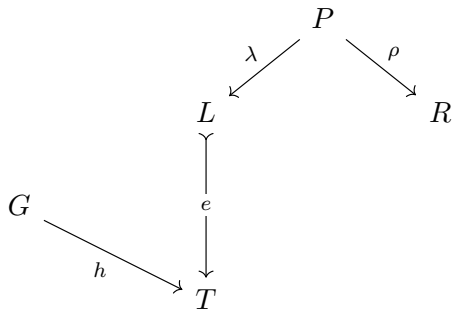
Since $\tau_{C'} = \tau_D \circ g_1 = \tau_D \circ g \circ h_1 = \tau_C \circ h_1$, we can conclude that h_1 is also an arrow of $\mathbf{C}/T$. □

Corollary 5. *If* $\mathbf{C}$ *also has all pull-back complements (over monos) then so does* $\mathbf{C}/T$.

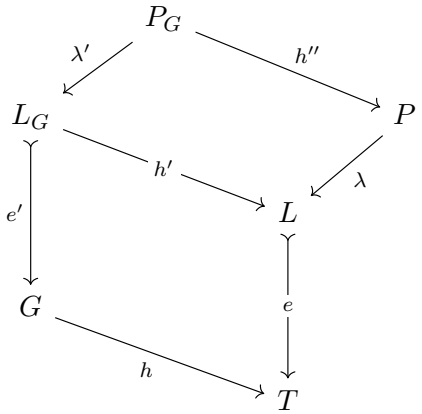
It is also immediate that co-limits in $\mathbf{C}$ are preserved in $\mathbf{C}/T$: all canonical typing arrows arise as universal arrows in $\mathbf{C}$. This easily extends to multi-co-limits. In particular, and for our specific purposes, push-outs in $\mathbf{C}$ and multi-sums in $\mathbf{C}_>$ are preserved in $\mathbf{C}/T$.

II Propagating rewriting

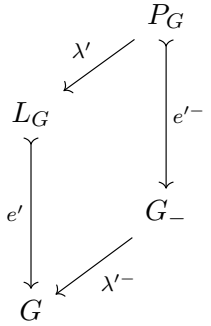
Recall the starting point for propagating rewriting: we have an object G typed by T and an instance of a rule r in T.



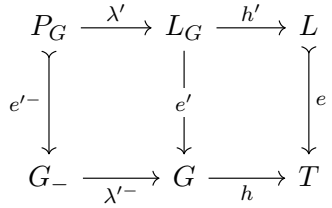
Instead of rewriting T and using that to construct G^-, we can rewrite [the left leg of] r itself by taking two successive pull-backs



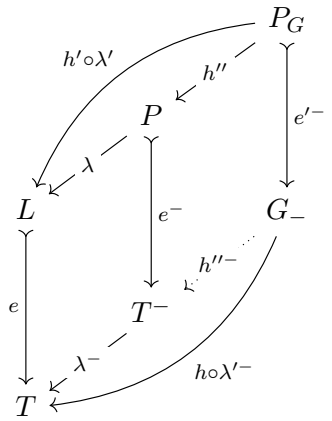
so as to obtain [the left leg of] a new rule that has expanded r into all its instances in G according to the typing h. We thus obtain a propagation of r to G by taking the PBC.



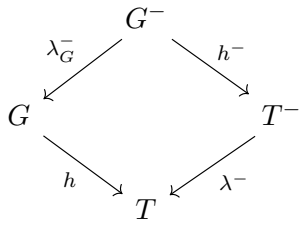
By the pasting lemma, we find that the outer rectangle of



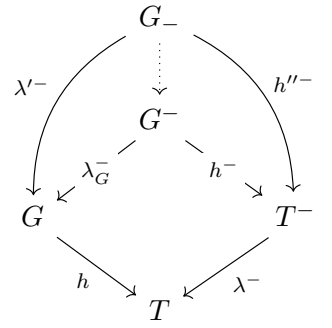
is a pull-back and, by a simple diagram chase, we can apply the universal property of the PBC $P \rightarrowtail T^- \to T$ to obtain the dotted arrow h''^-.



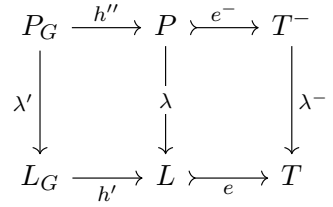
We can thus apply the universal property of the pull-back



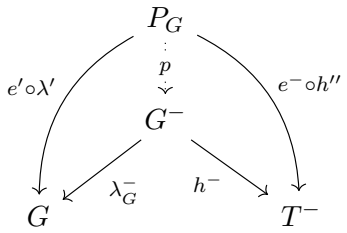
to obtain the dotted arrow.



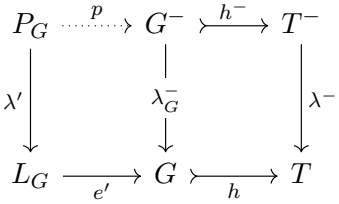
A second application of the pasting lemma implies that the outer rectangle of



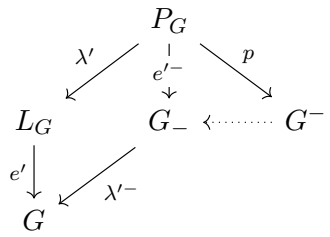
is a pull-back. The bottom line can be 'refactorized' through G since $e \circ h' = h \circ e'$. Moreover, $h \circ e' \circ \lambda' = \lambda^- \circ e^- \circ h''$ by a simple diagram chase, so we can apply the universal property of the pull-back



and hence 'refactorize' the top line, yielding

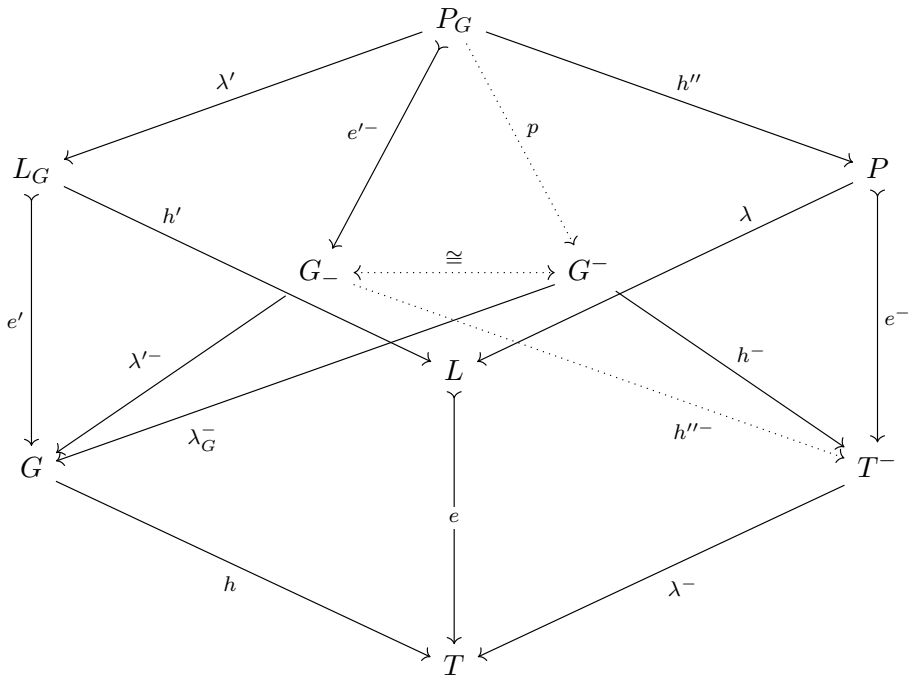


where the right-hand inner square is a pull-back. A final application of the pasting lemma therefore implies that the left-hand inner square is a pull-back and, by the universal property of the PBC, we obtain the dotted arrow



from which, by standard categorical reasoning, we can conclude that $G^- \cong G_-$.

This establishes the equivalence of this definition of propagation with that given in the main text. To aid the reader, we provide a single diagram with all the arrows to situate the argument in a single global context:



III Concrete Kappa

We give a brief guide to the concrete syntax of Kappa [17].

A Kappa *agent* has an *interface* of *sites*, each of which may have a single internal *state* (with multiple possible values) attached. Agent(type)s are declared in the following way:

```
%agent: A(s~0~1,t)
%agent: B(s)
```

In words, this declares an agent type `A` that has two sites, `s` and `t`, the former of which has a state taking value `0` or `1`; and a second agent type `B` with a single site `s` which, despite having the same 'name', has no connection with the site `s` of `A`. The default value of a state is the value mentioned after the first ~in the declaration.

A Kappa *rule* is specified by a LHS and RHS (but without explicitly stating the preserved region) of a rewriting rule:

```
A(s~1,t), B(s) -> A(s~1,t!2), B(s!2)
A(s~0) -> A(s~1)
```

The first rule says that an agent of type `A` whose site `s` is in state `1` and whose site `t` is free, *i.e.* unbound, can bind to an agent of type `B` whose site `s` is free. The resulting edge between the two free sites is represented explicitly in the RHS with the `!2` syntax: the `2` is just a label that allows us to identify the two end-points of the edge.

The second rule says that an agent of type `A` whose site `s` is free and in state `0` can change that state to `1`; the rule does not mention `t` and, as such, applies irrespective of its binding state. We could instead write

```
A(s~0?) -> A(s~1?)
```

if we want it to be independent of the 'free or bound' binding state of `s`.

In general, a rule can delete/add edges, change states and delete/add agents. The preserved region of a rule is determined by the so-called 'longest common prefix' convention: the longest common prefix of the LHS and RHS—ignoring the *values* of states and binding states—is considered to be preserved. In the two rules above, all agents (and so sites) are preserved. If we instead wrote

```
A(s~1,t), B(s) -> B(s!2), A(s~1,t!2)
```

then *nothing*, not even the existence of the agents, would be preserved and the effect of this rule would be to delete both agents and recreate them—including any missing sites and states with their default values—with states as per the RHS.

The specification of a Kappa system is completed by providing an *initial state*. Unlike in rules, all agents in the initial state must be fully-specified: display all their sites, according to their declaration, and specify a single well-defined value for each binding and internal state. These invariants must be preserved by all rule applications; we will see that this is, occasionally, non-trivial and a source of considerable complication.

Given a collection of agent declarations and rules and an initial state, their *contact graph* summarizes all the admissible edges. (This is effectively an annotation of the `%agent` declarations with `!2`-like syntax for edges.)

As such, the contact graph plays the rôle of *model*, analogous to the set of molecular species in chemical kinetics: it lays out everything that is possible, *i.e.* all rules and states project onto it. However, the contact graph is not subject to the *realizability* constraint applied to all other Kappa graphs: in a rule, or a state, a site can have at most one incident edge and, if it has a state, that must have a single value.

The final piece of syntax is the *binding wild-card* `!_` which, in the 'world-closing' context of a given contact graph, is the *negation* of being free: `s!_` tests that `s` is bound but cares not to what. The use of this construct is restricted to rules, *i.e.* it cannot be used when defining an initial state, and, even in rules, can only appear in a RHS if it does so as part of the preserved region of the rule, *i.e.* a rule cannot create a binding wild-card.

For further details on the concrete syntax of Kappa, its simulator `KaSim`, and its static analyser `KaSA`, please refer to the evolving online manual*.

*`http://dev.executableknowledge.org/docs/KaSim-manual-master/KaSim_manual.htm`