Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 313 (2015) 47–64

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Coarse-graining the Dynamics of Ideal Branched Polymers

Vincent Danos[a,1] Ricardo Honorato-Zimmer[b,2]
Sebastián Jaramillo-Riveri[c,3] Sandro Stucki[d,4]

[a] *School of Informatics*
University of Edinburgh
Edinburgh, United Kingdom

[b] *Informatics Life-Sciences Institute*
University of Edinburgh
Edinburgh, United Kingdom

[c] *SynthSys*
Edinburgh EH9 3JD, United Kingdom

[d] *PPS Laboratory*
Université Paris Diderot – Paris 7
Paris, France

Abstract

We define a class of local stochastic rewrite rules on directed site trees. We give a compact presentation of (often countably infinite) coarse-grained differential systems describing the dynamics of these rules in the deterministic limit, and study in a simple case finite approximations based on truncations to a certain size. We show an application to the modelling of the dynamics of sugar polymers.

Keywords: rule-based modelling, fragmentation, coarse-graining, ideal branched polymers

1 Introduction

There are well-understood limitations to using rate equations in the modelling of chemical and biochemical networks of reactions. Often, in realistic conditions, the number of molecules is small, stochastic effects become important and differential methods lose relevance. In this paper we will be concerned with another limitation induced by the *constructive complexity* of biochemical systems. To illustrate this idea

[1] Email: vdanos@inf.ed.ac.uk
[2] Email: s1066652@sms.ed.ac.uk
[3] Email: sjaramil@staffmail.ed.ac.uk
[4] Email: sandro.stucki@epfl.ch

http://dx.doi.org/10.1016/j.entcs.2015.04.018

consider proteins that can bind together via specific domains [17,14] and create a number of different species which is so enormous that it calls for different methods, perhaps even different questions and perspectives on the protein world [10]. Enters *rule-based modelling*. The idea is that basic molecules are now described as sets of domains, and the various reactions in which they can take part are pooled together into reaction classes or *rules*. This gives means to describe highly constructive systems in a compact form and without having to enumerate a (possibly infinite) set of species. This idea has been captured, studied, and implemented in two languages which are essentially the same, namely Kappa and the BNG language [1]. Other formalisms, in the tradition of process algebras, use rules of less direct expressivity but broadly similar in their intentions, and have the same fundamental ability to cope with unbounded sets of species [2].

Let us make the idea of rules a bit more precise by considering a molecule (or agent) A with two domains x, y. We can write a generic rule $A(x), A(y) \to A(x^1), A(y^1)$ expressing the possibility of forming a bond between x and y (the bond is denoted by the shared superscript) provided they are both free. This rule will apply *regardless* of the larger context in which the two agents A are found. It follows that this single rule pools or subsumes countably many reactions. Indeed, the two agents of type A can themselves be part of two chains, perhaps formed by the repeated application of the same rule. Suppose now A has a third domain similar to x, say z, with a similar binding rule $A(z), A(y) \to A(z^1), A(y^1)$. Taken together these rules will lead to the formation of branched polymers. Note that in this paper, there is no attempt at representing the actual geometry of complexes or polymers. This corresponds in the vocabulary of polymer sciences to *ideal* or *spherical polymers*. Of course, not every polymer is like that. For polymers where geometry is important (as in some of the cell molecular machinery such as the proteasome [9]) other and richer methods are called for (e.g. see Ref. [5], same volume). Given the title of this paper, one might be tempted to conclude that we are interested in the macro structure of such polymers. This is not the case. Rather, we are interested in the fine details of the connectivity between monomers at the micro scale. At larger scales, other interesting phenomena occur such as those treated in Ref. [7].

What then, do we mean by coarse-graining? Due to the constructive complexity mentioned above, the old biochemical horizon of a few hundreds of species is broken. This is specially true for polymers, where the number of possible species generated by a single polymerisation rule is infinite. To deal with such systems, coarse-graining methods have been developed recently which can reduce considerably the number of variables to be considered. In the context of the present paper, we are interested in one of them, called *fragmentation*. This method is suited to rule-based models and allows one to extract a smaller dimensional, more efficient representation of the dynamics [11,12,3]. We will adopt the method presented in Ref. [12] and explore the implications of using it in a new setting. In this method, one starts with a set of observables S and the fragmentation will generate a differential equation that describes the evolution of each observable (or *fragment*) in S as a function of some other observables $T_1, \ldots, T_n$. One can apply the same treatment to these new observables T_i, in the hope that at some point the procedure will stop generating new observables. If that happens, one ends up with a finite and closed set of differential equations for a set of observables $\mathcal{F}(S)$, from which any reference to the actual state of the system has been eliminated.

So far, this methodology has been successful at dealing with rule sets that generate only a finite number of species. Indeed, in this case, one is certain that the above expansion will terminate, as there cannot be more fragments than there are partial species. The overall goal of this paper is to take fragmentation to new territories where the number of species generated by the dynamics is unbounded. In this case, there is no guarantee that the procedure terminates. The "lower dimensional" system describing the evolution of fragments in $\mathcal{F}(S)$ might not be finite. It is even doubtful in general whether it is decidable that a given observable T is in $\mathcal{F}(S)$. Note that in this work we are dealing with a different kind of infinity than that of infinite-state continuous-time Markov chains as presented in [13]: Here we have an unbounded number of species (i.e. dimensions), not only an unbounded number of tokens for one or more species.

To explore this new universe cautiously, we start with a rather tame kind of polymers, namely the branched trees alluded to above. Nodes are taken from an alphabet Σ of agent types, each type having a distinguished *input* site. A tree over Σ is said to be *directed* if all links it contains link an output site of a node to an input site of another node. Note that such trees are allowed to be partial, meaning that nodes do not have to exhibit all their sites. Dealing with acyclic and directed polymers is one strong restriction. We shall add another one by considering only simple local rules of the following three types: branching (B), extension (E), and deletion (D). We refer to the combination of these types of rules as the *BED rule class*.

The problem is still interesting despite those simplifications and the BED rule class exhibits a rich phenomenology in relation to the fragmentation problem as we will see below. We ask essentially three questions. Firstly, we ask if there is a simple presentation of $\mathcal{F}(S)$, given S. The answer is affirmative. The presentation can be given by a simple set of rewrite rules that will generate all fragments in $\mathcal{F}(S)$. This set is in general infinite and we give a series of simple examples, although there are interesting particular cases where it is finite. These rewrite rules used to generate fragments are slightly more general than the Kappa syntax allows because they deal with partial objects (as opposed to Kappa rules that deal with complete objects, i.e. mixtures). Secondly, we ask whether one can decide if $\mathcal{F}(S)$ is finite, or more generally if a given T is in $\mathcal{F}(S)$. We give partial answers to this second question. Thirdly, we investigate whether one can define consistent *truncations* of the set of differential equations $\mathcal{E}(T)$, that is to say finite self-consistent subsets of equations, which will approximate the behaviour of the original system $\mathcal{F}(S)$ even if it is infinite. This is really two questions in one. We give an example of truncation in a very simple case and leave the general problem for future work.

This study increases our familiarity with fragmentation in general conditions, and advances the concept of approximate finite truncations as a way to deal with infinite fragmentations. This is not to say that the purpose of this work is entirely theoretical and methodological. It turns out that one of the simple examples where we have a finite expansion can be seen as a stylised version of sugar polymers. To the extent that one can describe their dynamics without explicit mention of geometries, our methodology offers a simple system of differential equations to explore the competition one observes between branching and elongation in the fine structure of some of such polymers. This preliminary work also demonstrates that there are many basic questions that are not yet understood about the coarse-graining of rule-based systems. We return to these questions in the conclusion.

2 Preliminaries

The site graph rewriting system which we are going to use is called Kappa. We will keep our Kappa reminder informal because it has been well-explained in other references (e.g. [16]) and the rest of the paper will give several examples of actual rule sets. As said in the introduction, agents (or nodes) of a certain type have a fixed set of sites and they can assemble into site graphs by using sites to build connections. Sites can also hold one of finitely many states. This is convenient to represent allosteric states and/or post-translational modifications, but we will not use it in this paper. For instance, if we return to the divalent agent $A(x, y)$ of the introduction, with the bond forming rule $A(x), A(y) \to A(x^1), A(y^1)$, we can form chains of any length, and even rings $A(y^0, x^1), A(y^1, x^2), A(y^2, x^0)$. Complete (meaning all agents have all their sites) connected site graphs correspond to the usual notion of species (also known as complex in biology).

Site graphs and embeddings form a category and we write $[x; y]$ for the set of embeddings of a site graph x into another site graph y. Our notion of *embedding* (sometimes called matching as well) is one-one on nodes and has to respect free sites: a free site can only be matched to a free site (unlike in traditional graph morphisms). This category has a fundamental property, namely it has a notion of minimal glueing. This means that for any two site graphs s_1, s_2, a pair of embeddings $m_1 \in [s_1; x]$, $m_2 \in [s_2; x]$ can always be factored uniquely up to unique isomorphism through a ‘smaller’ pair of embeddings that constitute a minimal glueing as follows:

$$\begin{tikzcd}
s_1 \arrow[r, "m_1'"] \arrow[dr, "m_1"'] & t \arrow[d] & s_2 \arrow[l, "m_2'"'] \arrow[dl, "m_2"] \\
& x &
\end{tikzcd}$$

One simply takes for t the union of the images of m_1, m_2 in x. There can be many minimal glueings depending on the pair m_1, m_2 one starts from. For example, in the case of $s_1 = s_2 = A(x^1), A(y^1, x^2), A(y^2)$, there are nine different non-isomorphic minimal glueings (it is an interesting exercise to find them all). Importantly, a glueing g is a diagram, specifically a pair of embeddings with the same codomain (also known as a co-span). We write $\hat{g}$ for that codomain (x and t respectively in the diagram above). Non-isomorphic minimal glueings can have the same codomain; so the $\hat{\cdot}$ map is *not* injective on (minimal) glueings. In other words, of all the (finitely) many minimal and non-isomorphic ways to glue two given site graphs s_1, s_2, some might lead to the same codomain t. This can cause some confusion and should be kept in mind for equation 1 below.

Rules are triples (s, α, k) consisting of a partial site graph s (the left hand side or lhs), a rule action α which is a composition of atomic actions such as erasing or adding an edge or a node (and changing internal states), and a rate $k \geq 0$. The notion of embedding guarantees that these instructions carry over through any embedding to any complete site graph x, that is, that the action of the rule is applicable to x. Each embedding of s in x a complete site graph can give rise to an event and is given rate k. This automatically defines the stochastic semantics of a set of rules as a continuous-time Markov chain which can be implemented efficiently [4]. Despite the said efficiency, and for other reasons we will mention in the next section, it is sometimes advantageous to use another semantics based on differential equations. One can do this on species, but for even moderately complex rule sets the number of species can be prohibitive, and the differential approximation of the natural stochastic semantics irrelevant. One solution to this is to write differential equations on partial observables.

Given a partial connected site graph F (also called a fragment) and a rule set R, the average rate of change in the number of embeddings for F in x can be expressed as follows:

$$\frac{d}{dt}[F; x] = -\sum_{r \in R} \sum_{C \in g(F,r)} k(r) \cdot [\hat{C}; x] + \sum_{r \in R} \sum_{P \in g'(F,r)} k(r) \cdot [(\hat{P})^\star; x] \tag{1}$$

where $g(F, r)$ is the set of *minimal glueings* of F and the lhs of rule r such that the action of the rule $\alpha(r)$ would modify F (called relevant glueings for short), $g'(F, r)$ is the same but to the rhs of r, and $(\hat{P})^\star$ denotes the result of applying the inverse rule $r^\star$ to $\hat{P}$.

One has a similar equation for the observables $[\hat{C}; x]$, and $[(\hat{P})^\star; x]$ which in turn will produce new observables. The process of *fragmentation* consists in the repeated application of the equation above until a fixed point, possibly infinite, is reached [12]. Note that the procedure can be made relative to the reachable observables. There is no need to generate an equation for an observable that one knows will never be present in the system.

2.1 Directed site trees

We will assume throughout this paper that reachables are site *trees* built on *directed agents* $(i, s_1, \ldots, s_n)$ with a unique *input site* and any number of output ones (Fig. 1). To simplify notations, we suppose agent types are encoded in their unique i input site. So $i \in I$ where I indexes agent types. We write $\sigma(i)$ for the set of *outputs* of agent i, so that the complete interface of this agent type is $\{i\} \cup \sigma(i)$. We fix once and for all the alphabet Σ of directed agents we work with, and will consider rule sets where only bonds between inputs and outputs can be formed.

We write $\mathcal{T}(\Sigma)$ for *directed site trees* over Σ, and $\mathcal{T}_0(\Sigma) \subseteq \mathcal{T}(\Sigma)$ for the complete ones, and $\mathcal{T}_0(\Sigma)^\star$ for finite disjoint sums of trees in $\mathcal{T}_0(\Sigma)$. Every tree, complete or not, has a natural order induced by the local orientation of agents. This order has a minimum node which we call the *root* of the tree, and a set of maximal nodes which we call the *leaves* of the tree (Fig. 1).

Given trees T, T_1 in $\mathcal{T}(\Sigma)$, we write $T_1 \leq T$ if T_1 can obtained from T by repeatedly pruning leaves off of T (leaving the output site to which the leaf was attached free in the node below the leaf); by convention, this *includes* the case where T_1 is the empty tree (all leaves have been pruned so to speak). This is stronger than the usual notion of sub-site-tree as one is not allowed to erase free sites.

Given a tree S, we write ∂S for the set of its free output sites; given a sequence of N *complete* trees, $\boldsymbol{T}$, including possibly empty ones, and a tree S with $|\partial S| = N$, we write $S \circ \boldsymbol{T}$ for the tree obtained by binding each free output site of S to the root of its associated tree in $\boldsymbol{T}$ (if it is not empty, and doing nothing if it is).

We will restrict further our interest to rule sets under which the set $\mathcal{T}_0(\Sigma)^\star$ is closed, that is, rule sets where the result of applying any rule to a complete site tree is again a complete site tree.

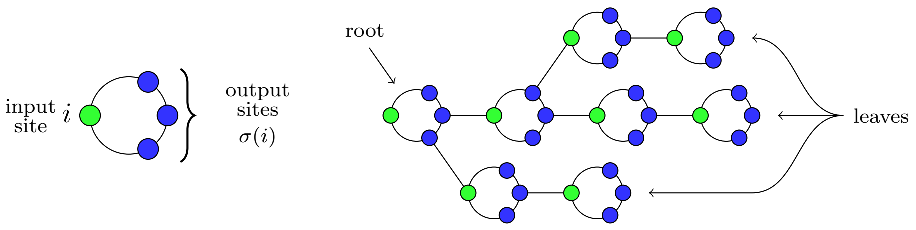


Figure 1. Directed agent on the left and directed tree on the right.

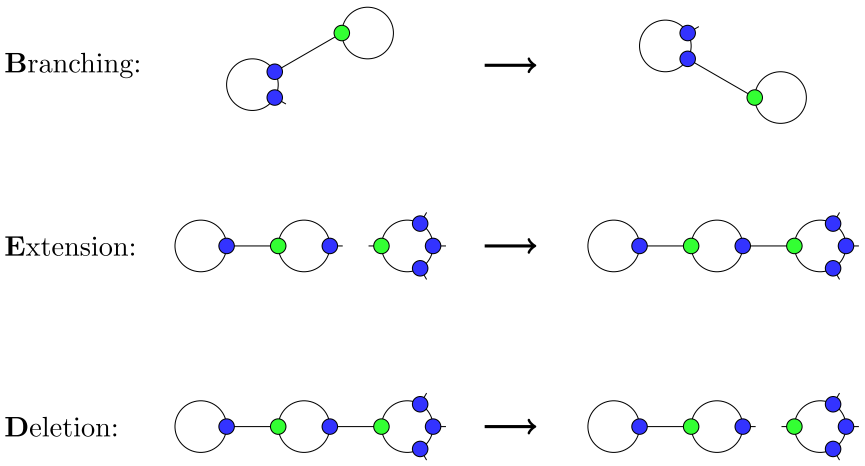


Figure 2. BED rule set.

2.2 BED rule set

Our chosen BED rule set is as follows.

$$
\begin{array}{lll}
B(s,t,i) & := (s^1,t),(i^1) \to (s,t^1),(i^1) & \textit{Branching} \\
E(s,i,t,j) & := (s^0),(i^0,t),(j,\sigma(j)) \to (s^0),(i^0,t^1),(j^1,\sigma(j)) & \textit{Extension} \\
D(s,i,t,j) & := (s^0),(i^0,t^1),(j^1,\sigma(j)) \to (s^0),(i^0,t),(j,\sigma(j)) & \textit{Deletion}
\end{array}
$$

Fig. 2 shows a graphical representation of these rules. Branching is allowed anywhere including at a root, and not just at the leaves (i.e. not only at the 'surface' of the tree, hence we say this is the 'volume' version of the branching rule). Extensions and deletions on the other hand must happen on leaves, so at the surface. Extensions are obtained by binding a free output site t to an agent $(j,\sigma(j))$ with its input site j free. This can happen only provided t's node is itself input bound via a specific (s,i) bond (called later sometimes the 'witness'). This condition forces the node to not be the root - so we need at least a dimer *seed* to start up a polymer. This condition also allows for dependencies between the added edge (t,j) and the witness one (s,i), depending on whether the witnessing edge (s,i) can be any edge type or a subset of these.

Leaf deletion depends on a witnessing edge as well. This means that a seed dimer cannot be dissociated, thus avoiding the deadlock state in which we do not have a tree to grow.

These rules were chosen as a simple class of rules that allows nevertheless a stylised version of the action of different classes of enzymes involved in the polymerisation and management of large sugar polymers. See next section for more motivations and explanations about the biological context. Having said that, in the context of this paper these rules serve mostly as a testbed to investigate infinite fragmentations as said earlier in the introduction.

It is easily seen that:

Lemma 1 $\mathcal{T}_0(\Sigma)^\star$ *is closed under the action of BED rules.*

The fact that we can restrict to systems where the only reachable complexes are directed trees over Σ will greatly simplify our analysis of fragmentation.

3 Basic surface model

Before we move on to the general theory, let us investigate a simple surface model where we have only extension and *surface branching*. See Fig. 3 for the rules in graphical notation. In our model Σ consists of only one type of agent $G(s_1,s_4,s_6)$ where s_1 is the distinguished input (see Fig. 4 for an example directed site tree generated by this model). This agent is an abstract version of the glucose molecule which has three binding sites in sugar polymers. The indices 1, 4, 6 refer to the position of the atom used in the formation of a new bond in the molecule. The rule themselves abstract the behaviour of enzymes extending and rebranching sugar polymers. Rebranching enzymes are known to rebranch with longer distances between the section node (white) and the insertion node (light blue), but one can imagine that an s_4–s_1 bond stands for several monomers in a chain.

Branching is directed from s_4 to s_6. Note that in this basic system, once a (white) node has chosen to extend, it can *no longer* branch, because B is only enabled at leaves.

Extension requires the s_1 site of the left (white) node to be bound; otherwise free nodes can be used as seeds to generate new trees. This condition is expressed in Kappa by the use of a "semilink" which states that a site has to be bound. Semilinks are denoted by a thick line sticking out of a site in graphical notation and a star ($\star$) superscript in text notation. If we do not impose such condition on extensions, the formation of polymers is no longer a unique growth where monomers are added (for this kind of free polymerisation, the reader can consult Ref. [6]).

To analyse this basic model we define the following observables:

$$\begin{aligned} T &= G(s_1^\star, s_4) \\ G &= G(s_1, s_4, s_6) \\ T_f &= G(s_4^1, s_6), G(s_1^1, s_4, s_6) \end{aligned}$$

Then, the extension and branching activities are:

$$\begin{aligned} \alpha_E &= k_E[G][T] \\ \alpha_B &= k_B[T_f] \end{aligned}$$

where $[G]$ is the number of free G agents and $[T_f] \le [T]$ is the number of leaves u standing above a node free at s_6. In Fig. 5, we can see an example tree where $[T_f] = 0$, hence a deadlock state for this basic model.

We have a mostly branching regime when:

$$\kappa := \frac{\alpha_B}{\alpha_E} = \frac{k_B}{k_E} \cdot \frac{1}{[G]} \cdot \frac{[T_f]}{[T]} \gg 1$$

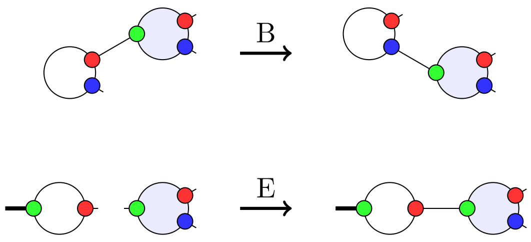


Figure 3. In this basic model, one can either branch (B) or extend (E). In both cases the right (light blue) agent has to be free on s_4 and s_6 to guarantee the absence of cycles. The initial state needs a seed chain, i.e. a chain of length at least 1 (i.e. 2 agents), else no rule applies. Note the use of a semilink, denoted by the thick line, on site s_1 of the left agent in rule E.

3.1 Fragmentation

To study the basic model, in particular the ratio $\kappa = \alpha_B/\alpha_E$, we can fragment it using as seeds the observables defined above, i.e. G, T, and T_f. The closure of this triple is given by:

$$\begin{aligned}
X_1 &= G(s_1^\star, s_4) && = T \\
X_2 &= G(s_1^\star, s_4, s_6) && \\
X_3 &= G(s_1, s_4, s_6) && = G \\
Y_1 &= G(s_4^1, s_6), G(s_1^1, s_4, s_6) && = T_f \\
Y_2 &= G(s_1^\star, s_4^1, s_6), G(s_1^1, s_4, s_6) &&
\end{aligned}$$

Indeed the fragmentation gives:

$$\begin{aligned}
[\dot{X_1}] &= -k_E[X_1][X_3] + k_E[X_1][X_3] + k_B[Y_2] &&= k_B[Y_2] \\
[\dot{X_2}] &= -k_E[X_2][X_3] + k_E[X_1][X_3] &&= k_E[X_3]([X_1] - [X_2]) \\
[\dot{X_3}] &= -k_E[X_1][X_3] + \phi && \\
[\dot{Y_1}] &= -k_E[Y_1][X_3] - k_B[Y_1] + k_E[X_1][X_3] &&= k_E[X_3]([X_1] - [Y_1]) - k_B[Y_1] \\
[\dot{Y_2}] &= -k_E[Y_2][X_3] - k_B[Y_2] + k_E[X_2][X_3] &&= k_E[X_3]([X_2] - [Y_2]) - k_B[Y_2]
\end{aligned}$$

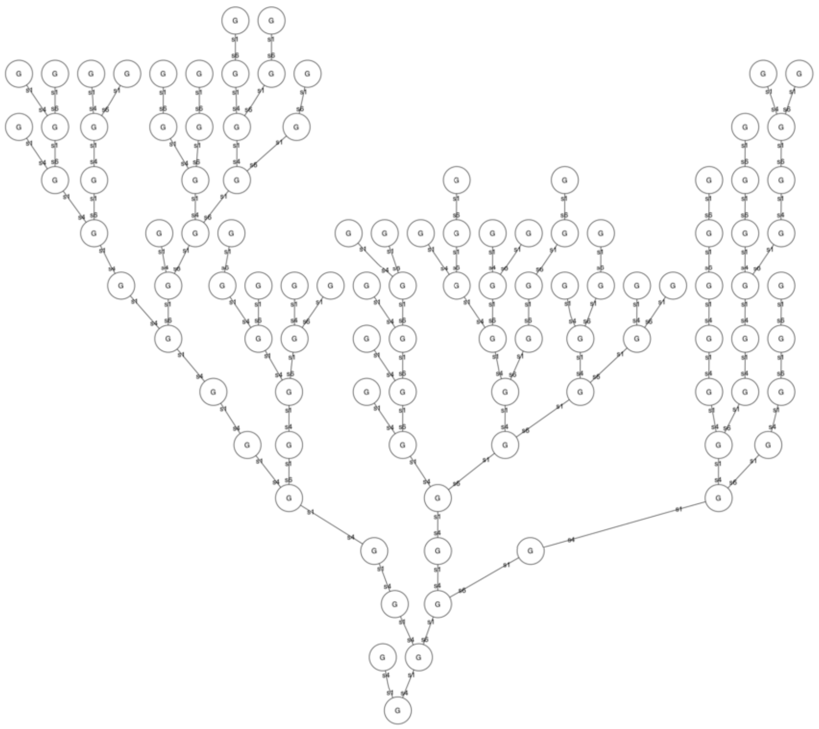


Figure 4. An example of a site tree generated by the basic model; with our simple branching steps, we can have nodes with an s_6 bond and no s_4 bond.

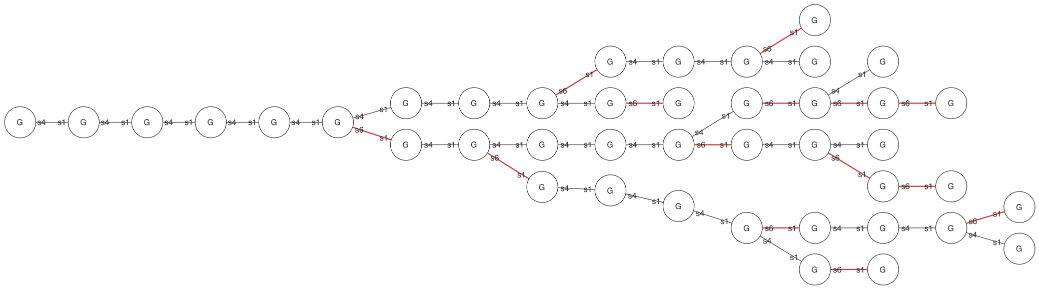


Figure 5. A site tree x over Σ; s_6–s_1 bonds are indicated in red; no leaf can induce a branching, in other words $\alpha_B(x) = 0$.

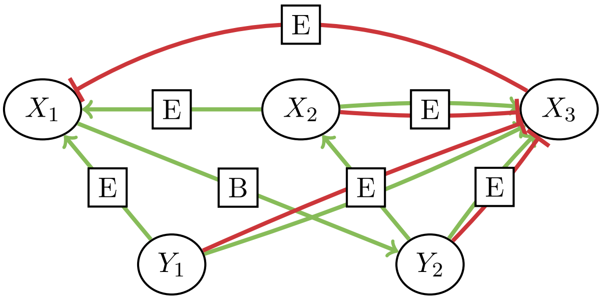


Figure 6. Each node is a pattern and an arrow represent a dependency in the fragmentation labelled by the rule and coloured green if it is an activation and red if an inhibition.

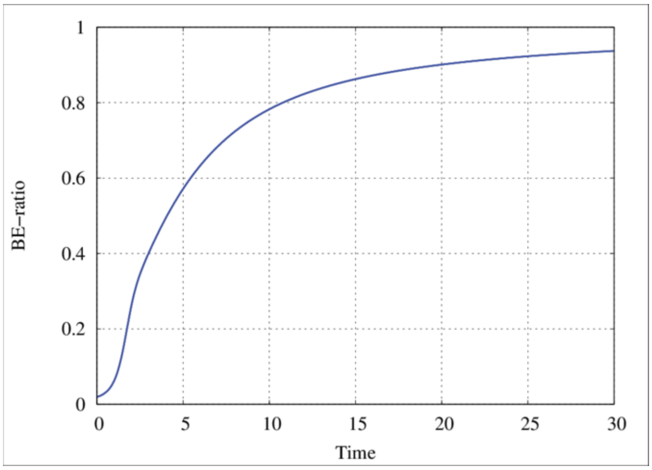


Figure 7. The evolution of the BE ratio starting in the E-regime. Initial conditions are $[X_1] = [X_2] = [Y_1] = 1$, $[Y_2] = 0$, $k_E = k_B = 1$, $\phi = 10$, and $[X_3] = 50$.

with ϕ the external flow of G. Note that $[X_1] \geq [X_2] \geq [Y_1] \geq [Y_2]$.

Note that $\{X_1, X_2, X_3, Y_2\}$ is already closed, but we also need Y_1 to obtain κ. This can be visualised with the rule-labelled dependency diagram shown in Fig. 6.

With initial conditions: $[X_1] = [X_2] = [Y_1] = 1$, $[Y_2] = 0$, $k_E = k_B = 1$, $\phi = 10$, and $[X_3] = [G] = 50$, the initial BE ratio is $\kappa = k_B[Y_1]/k_E[X_1][X_3] = 1/50$ well in the extension regime and Fig. 7 shows its evolution and asymptotic convergence to a more balanced regime.

3.2 Aside on the system of differential equations

The advantage of working with a system of differential equations is not just that of obtaining a coarse-grained and therefore more efficient representation of the system. One can also take the opportunity of using techniques from qualitative dynamical systems. For instance, one can compute the Jacobian of F for $\dot{\mathbf{X}} = F(\mathbf{X})$ the ODE system above:

$$J_F = \begin{pmatrix} 0 & 0 & 0 & 0 & k_B \\ k_E[X_3] & -k_E[X_3] & k_E([X_1]-[X_2]) & 0 & 0 \\ -k_E[X_3] & 0 & -k_E[X_1] & 0 & 0 \\ k_E[X_3] & 0 & k_E([X_1]-[Y_1]) & -k_B - k_E[X_3] & 0 \\ 0 & k_E[X_3] & k_E([X_2]-[Y_2]) & 0 & -k_B - k_E[X_3] \end{pmatrix}$$

and verify that the Thomas necessary condition for the existence of an oscillation is satisfied [15]. Such oscillations might be one of the sources of the alternating microstructure of starch. We can also compute the time derivative of our BE-ratio:

$$\dot{\kappa} = \frac{k_B}{k_E} \cdot \frac{[Y_1]}{[X_1][X_3]}\left(\frac{[\dot{Y_1}]}{[Y_1]} - \frac{[\dot{X_1}]}{[X_1]} - \frac{[\dot{X_3}]}{[X_3]}\right)$$

Note that a serious modelling effort of this system, which we intend to pursue in later work, will need to take into account the actual mediation by enzymes and the saturation effect for extension and branching. This is all very interesting, but adding rule D creates countably many new fragments, which means that to do the kind of analysis (e.g. of the ratio κ) we have suggested one would have either to work in a regime where deletions are negligible, or else deal with an infinite system of differential equations. If one chooses the latter more theoretical option, it becomes even unclear how to describe these equations. This question is what we address in the next section.

4 Analysis of BED fragmentations

The goal of this section is to characterise simply the set of fragments of any BED rule set on directed trees. We suppose we are given a starting observable S and want to describe the set $\mathcal{F}(S)$ of fragments generated by S.

As said, adding deletions changes things dramatically in that fragmentations become infinite in general. In fact, even with just Es and Bs it is sometimes infinite as we will see. As we will also see, unless one is ready to ask very specific constraints, *chains are not closed* under fragmentation; specifically, the right glueings on D described below allow branching on any free site, and B glueings will generate such sites. So one strong conclusion of this section is that the simple BE fragmentation space, which we have studied above, is somewhat atypical and far from documenting the complexity of the general problem.

Before we look at the fragment-generation rules, we have to fix some notation and conventions. We will present $\mathcal{F}(S)$ by using rules acting on partial directed trees. The interpretation of these generative rules is that we can apply them in any context, and the claim is that $\mathcal{F}(S)$ is the (smallest) set of trees obtained by repeatedly applying these rules until a (smallest) fixed point is reached. It turns out that these rules are almost Kappa rules except for two facts: 1) one is allowed to *add absent sites* in the action of rules, and 2) one can *test for the absence of specific sites*. Specifically, we will use the notation $\neg x$ to mean that a site x is not present in the agent interface it is matched to. Clearly, this addition to pattern matching is only useful when matching against partial objects. When matching against complete objects, by definition all sites are present as in their full interface, and a $\neg x$ clause is never satisfied, unless x is not in the node interface in which case it is always satisfied. Implementation-wise, this is a minor generalisation which changes little to the matching engine and could be of direct modelling interest, e.g. for modelling protein cleavage.

Using this variant of Kappa, we now present our generative rules. For each rule type, the lhs glueings are presented first (with a $-$ superscript) and the rhs ones second (with a $+$ superscript). By convention, we present for each rule and side the generative rules in the order of *increasing relevant intersections*. So-called *self-glueings* where the intersection of the fragment and the lhs component is the component itself, and therefore the fragment under glueing is already big enough relative to that glueing, are not shown. They do not generate new variables in the final ODE (but, they do generate new terms). In each case the rhs of the generative rule is the lhs of the rule of interest, by definition of a glueing.

Here are the generative rules indexed by BED rules over Σ.

$E^-(s,i,t,j):$	$(\neg i, t)$	$\Rightarrow$	$(s!0),(i!0,t),(j,\sigma(j))$	*co-grow*
$E_1^+(s,i,t,j):$	$(\neg i, t!1),(j!1,\sigma,\neg\sigma')$	$\Rightarrow$	$(s!0),(i!0,t),(j,\sigma(j))$	*co-grow-and-cut-leaf*
$E_2^+(s,i,t,j):$	$(s!0),(i!0,t!1),(j!1,\sigma,\neg\sigma')$	$\Rightarrow$	$(s!0),(i!0,t),(j,\sigma(j))$	*cut-leaf*
$D_1^-(s,i,t,j):$	$(\neg i, t!1),(j!1,\sigma,\neg\sigma')$	$\Rightarrow$	$(s!0),(i!0,t!1),(j!1,\sigma(j))$	*co-grow-and-leaf completion*
$D_2^-(s,i,t,j):$	$(s!0),(i!0,t!1),(j!1,\sigma,\neg\sigma')$	$\Rightarrow$	$(s!0),(i!0,t!1),(j!1,\sigma(j))$	*leaf-completion*
$D_1^+(s,i,t,j):$	$(\neg i, t)$	$\Rightarrow$	$(s!0),(i!0,t!1),(j!1,\sigma(j))$	*co-grow-and-grow leaf*
$D_2^+(s,i,t,j):$	$(s!0),(i!0,t)$	$\Rightarrow$	$(s!0),(i!0,t!1),(j!1,\sigma(j))$	*grow-leaf*
$B_1^-(s,s',i):$	$(s',\neg s)$	$\Rightarrow$	$(s',s!1),(i!1)$	$+s$, $+si$
$B_2^-(s,s',i):$	$(\neg s', s!1),(i!1)$	$\Rightarrow$	$(s',s!1),(i!1)$	$+s'$
$B_1^+(s,s',i):$	$(\neg s', s)$	$\Rightarrow$	$(s',s!1),(i!1)$	$+si$, $+s'$
$B_2^+(s,s',i):$	$(s'!1,\neg s),(i!1)$	$\Rightarrow$	$(s',s!1),(i!1)$	$-s'i$, $+s$, $+si$
$B_3^+(s,s',i):$	$(s'!1,s),(i!1)$	$\Rightarrow$	$(s',s!1),(i!1)$	$-s'i$, $+si$

If we have a rule r of the BED class, we write $\Gamma(r)$ for the set of generative rules associated to r.

Let us discuss each rule in turn.

First, we see that E rules generate $(j,\sigma(j))$, a complete leaf of type j, as a fragment. To simplify the reasoning below, we will assume hereafter that all complete leaves are in the current fragmentation. This is without real loss of generality because any application of the E rules will add them, and apart from degenerate examples, E rules will always apply to the fragment S of which one is computing the fragmentation $\mathcal{F}(S)$.

The rule E^- "co-grows" an input-less node if this node has an extensible free output, and co-grows the witness of that extension (the term co-growth comes from the fact that the tree is growing by the root); it generates also a complete leaf of type j. As all of the above rules bearing a $\neg i$ constraint on their root node, namely E^-, E_1^+, D_1^-, D_1^+, this rule can only be applied at the *root* of a fragment. Indeed, this is the only node in a fragment where there might not be an input (by connectivity of the fragment and the fact that the reachables can only connect via inputs and are trees). We note also that the result does not depend on j (the leaf added) except for adding that leaf to the current fragment.

For E^+ and D^- rules, σ is any subset of $\sigma(j)$ and $\sigma' = \sigma(j) \smallsetminus \sigma$ (recall $\sigma(j)$ is the complete output interface of agent j); in other words, all sites of our leaf are free[5]. One could, equivalently, write $(s!0), (i!0, \sigma, \neg(\sigma(i) \smallsetminus \sigma))$. There is one such rule for every subset σ of $\sigma(j)$.

Note that D^+, D_1^+, and D_2^+ can generate branching if the free site s, or t, sits on a node that already has an outgoing edge. And, as B rules can add such sites, unless one is very particular models, the set of chains will not be closed under BED fragmentation. Branching rules induce branches in the fragments as well.

All B^+ instances incorporate a 'co-swap', swapping from s' back to s with the creation of missing material in the fragment intersection.

Proposition 4.1 *Given $S \in \mathcal{T}(\Sigma)$, and a rule set R of the BED class over Σ, the fragmentation of S, $\mathcal{F}(S)$, is the (smallest) set of trees obtained as a fixed point of the set of generative rules $\cup_{r\in R}\Gamma(r)$.*

Proof Clearly, it is enough to restrict to glueings of one single component: this remark only applies to left glueings to E, and right ones to D which are the only cases where one has more than one component; in both cases, it is impossible to glue to two components at once as one of them is an isolated node.[6]

The proof can then proceed by inspection of all possible relevant intersections of a directed tree in $\mathcal{T}(\Sigma)$ with one BED rule component (left or right). Let us consider the case of a right glueing to B as an illustration. Suppose T is a tree in $\mathcal{T}(\Sigma)$. For a glueing of T on the rhs of a B rule to be relevant, the intersection must contain one of the three sites that are modified by the rule, s, s' and i. If the intersection is restricted to s on a node $u \in T$, this means two things: 1) s cannot be present in u (else it would either grow the intersection or make it incoherent and contradict the existence of a glueing), 2) the glueing will add the missing s' and an edge to a new node at i. It remains to apply the inverse of the action of the rule and we obtain B_1^+. The intersection cannot be limited to s' since s' is bound in B's rhs, but it can

[5] This is not a Kappa rule, and if we were to write simply $(j!1, \sigma)$ it would be a wrong Kappa rule, as one could cut a non-leaf.

[6] In fact, one can prove much more generally, that double glueings in arbitrary rules with components in $\mathcal{T}(\Sigma)$), which are possible with more complex rules, would induce cycles in fragments, and are therefore unnecessary with rule classes under which $\mathcal{T}_0(\Sigma)^\star$ is closed. A consequence is that glueings are enumerated by one point of identification (by the rigidity of site graph embeddings).

be limited to $(s'!1), (i!1)$ which gives the second case, and finally the intersection can contain the entire rhs which gives the third and last case. □

4.1 Exercise in composition

The *refinement* of leaf completion, the rule $D_2^-(s, i, t, j)$ where the fragment has a co-edge (s, i), can be simulated (assuming j-leaves are in the current fragmentation) by a combination of $E_2^+(s, i, t, j)$ and $D_2^+(s, i, t, j)$:

$$\begin{array}{r@{\;}c@{\;}l} E_2^+(s,i,t,j) & \dfrac{(s!1),(i!1,t!0),(j!0,\sigma,\neg\sigma')}{(s!1),(i!1,t)} & \textit{cut-leaf} \\ D_2^+(r,h,s,i) & \dfrac{}{(s!1),(i!1,t!0),(j!0,\sigma(i))} & \textit{grow-leaf} \end{array}$$

Similarly, $E_1^+(s, i, t, j)$ and $D_2^+(s, i, t, j)$ simulates $D_1^-(s, i, t, j)$. So from the strict point of view of which fragments are generated, there is never a need to take into account the D^- rules. They do come into play to compute the differential system though. We will use this remark to simplify our reasoning below.

4.2 Divergences

It is easy to generate countably many fragments by using the simple D^+ rule. For the refined case, one can use any periodic sequence of $D_2^+(s_n, i_n, s_{n+1}, i_{n+1})$ rules. But in fact, divergent fragmentations can be obtained even without using the D rule, and using only E^-, and B^+ as follows:

$$\begin{array}{r@{\;}c} E^-(s_1,i_1,s_0,\_) & \dfrac{(s_0)}{(s_1^1),(i_1^1,s_0)} \\ B_2^+(t_1,s_1,i_1) & \dfrac{}{(s_1,t_1^1),(i_1^1,s_0)} \\ E^-(s_2,i_2,s_1,\_) & \dfrac{}{(s_2^2),(i_2^2,s_1,t_1^1),(i_1^1,s_0)} \\ B_2^+(t_2,s_2,i_2) & \dfrac{}{(s_2,t_2^2),(i_2^2,s_1,t_1^1),(i_1^1,s_0)} \end{array}$$

For this to work, however, one needs to be able to swap *to* a site that one can also co-extend. This would not be the case with our earlier sugar polymers example (previous section) even with volume branching since one swaps away from the edges generated by E rules. Indeed the fragmentation of the volume variant without deletion is finite in this case.

4.3 The complexity of $\mathcal{F}(S)$ in the ED class

Here we address the next important question which our first result leaves open, namely that of the complexity of the fragment set generated by an observable S.

Lemma 2 *Let S be a tree in $\mathcal{T}(\Sigma)$ and R be a rule set of class ED, any $T \in \mathcal{F}(S)$ can be written as $T = e \circ T_1 \circ \boldsymbol{T}$ for some sequence $\boldsymbol{T}$ of complete trees in $\mathcal{T}_0(\Sigma)$, and e either empty or a rootless edge with one output site, and some $T_1 \leq S$.*

Proof First, we do not need to consider rule D_1^-, D_2^- as they can be simulated. Suppose now that S has an input i (necessarily free) in its root interface. Then none of the remaining co-growth rules E^-, E_1^+, and D_1^+ can be used. Remain E_2^+, and D_2^+. Applying D_2^+, and then E_2^+ at the same node is the identity transformation. So one can assume that all E_2^+ cut-leaf rules are done first. This gives $T_1 \leq T$ by definition, and the remaining (optional) rule applications of type D_2^+ define the substitution vector $\boldsymbol{T}$.

Consider now the case where S's root, u, does not have an input. Then one can use rules E^-, E_1^+, and D_1^+ at u. A careful inspection shows that all cases can be traced back to the E^- one. For E_1^+ the leaf cut could have been cut by E_2^+ and this only changes the definition of T_1 and we are back in the E^- case. For D_1^+, the leaf grown can be grown later by D_2^+ after co-growth, so one can apply E^- directly without changing the above decompositions T_1, $\boldsymbol{T}$. Remains E^- which co-grows by adding an edge to u's input (and adding that input at the same time) from a new root u^-. That new root can support no further rewrite from E^-, and D_1^+ as u^- has no free site. It can support an application of E_1^+ but for this u has to be a leaf in T_1. In which case $T = e = (s!0), (i!0, t)$ a shape which is fixed under the action of E_1^+ and we conclude with T_1 the empty tree. □

Proposition 4.2 *Let S be a tree in $\mathcal{T}(\Sigma)$ and R be a rule set of class ED, whether $T \in \mathcal{F}(S)$ is decidable.*

Proof Suppose again S has an input at the root. Then T must have one as well. One can then enumerate the decomposition of T as $T_1 \circ \boldsymbol{T}$, which is easily done as there are finitely many T_1s such that $T_1 \leq T$. For each decomposition, it remains to see if R does contain enough rules to take the necessary E_2^+, and D_2^+ steps as in the proof above.

If S has no input at the root. Then T should have none either. One can then try to decompose $T = T_1 \circ \boldsymbol{T}$ as above, or $T = e \circ T_1 \circ \boldsymbol{T}$ for some edge e if S's root has a free site, or barring that $T = e \circ T'$ for a complete T'; all of the above decompositions are easily enumerated and tested against R, as explained in the proof of the preceding lemma. □

It follows immediately that deciding whether a given monomial is in the differential system associated to S is decidable. Clearly, the decision procedure delineated above could be made more efficient. The style of proof might extend to cover the B case but we have not done it yet.

5 Truncation

Now that we know that most fragmentations are infinite and we have found a way to describe them compactly, the next step is to understand how one can truncate them. We will not address this question in the context of this preliminary investigation but we can work out a consistent truncation for a simple divergent fragmentation for the ED system:

$$\begin{aligned} A(y) &\xrightarrow{E} A(y^1), A(x^1, y) \\ A(y^1), A(x^1, y) &\xrightarrow{D} A(y) \end{aligned}$$

Suppose the initial state is a chain with n links (if $n = 0$ this means no link and exactly one free $A(x, y)$ node). Then the system is equivalent (because there is only ever one chain) to a continuous-time Markov chain on integers, namely, $n \rightarrow_{k_e} n+1$, and $n + 1 \rightarrow_{k_d} n$. It is easy to verify that this chain has a probabilistic equilibrium which is a geometric distribution with $\alpha := k_E/k_D$ which we now suppose.

Set $F_0 := A(y)$, $F_n := A(y^1), \ldots, A(x^n, y)$ for $n \geq 1$. Suppose we want to compute the average number of nodes in the chain. We take as our unique seed $A()$ and get the following fragmentation:

$$\begin{aligned} \dot{A}() &= k_e[A(y)] - k_d[A(y^1), A(x^1, y)] &&= k_e F_0 - k_d F_1 \\ \dot{F}_{n+1} &= k_e(F_n - F_{n+1}) - k_d(F_{n+1} - F_{n+2}) \end{aligned}$$

So, $\mathcal{F}(A())$ is infinite, and we get a simple recurrence for the associated countable ODE system, $\mathcal{E}(A())$. That $\mathcal{E}$ is countable is not surprising because $F_n = 1$ iff the chain is of length $\geq n$.

Interestingly we can truncate this system at stage $n + 1$ in two ways:

$$\begin{aligned} \dot{F}_{n+1} &= k_e F_n - k_d F_{n+1} \\ \dot{F}_{n+1} &= k_e F_n - (k_e + k_d) F_{n+1} \end{aligned}$$

As can be seen in Fig. 9b the second truncation, which is purely syntactic (we set to zero the fragments we do not want) is worse than the first one in Fig. 9a compared with the true stochastic simulation given Fig. 8.

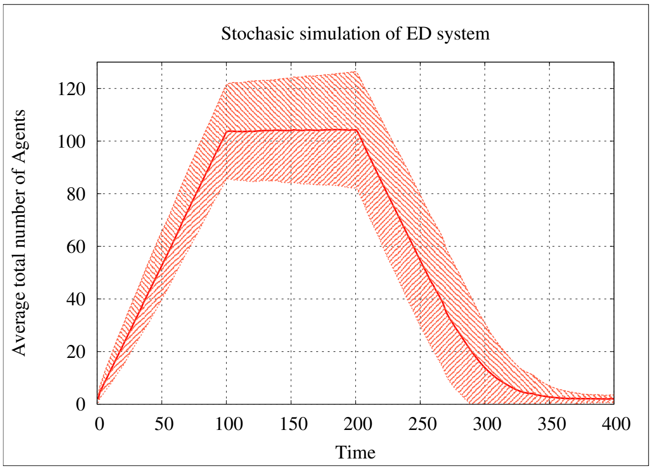
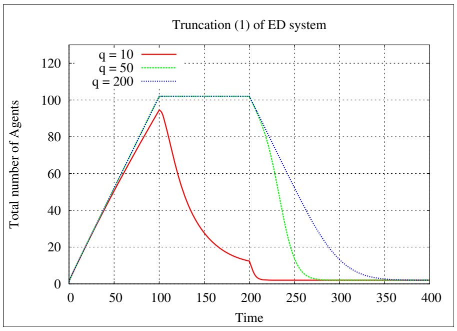
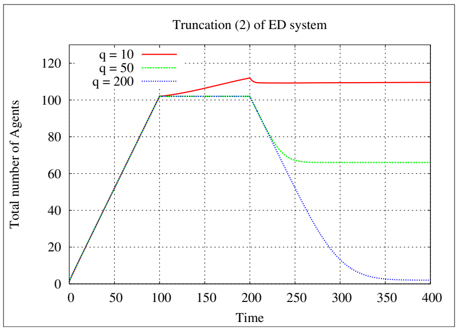


Figure 8. Average and standard deviation for the number of As in the chain from 500 stochastic simulations, with a chain of length 1 as initial state. Rates k_e and k_d are set to 2 and 1 initially, then k_e is set to 1 at time 100, and k_d to 2 at time 200.



(a) Truncation 1



(b) Truncation 2

Figure 9. Numerical integration of truncations 1 and 2 up to F_{10}, F_{50}, and F_{200}. Rates and initial states are as in the stochastic case. The quality of the approximation for truncation 2 degrades clearly for F_{50} compared to truncation 1.

6 Conclusion

We have defined a simple but expressive class of Kappa rules operating on directed polymers and which generates a type of coarse-graining called fragmentations, which lead in general to infinite ODE systems. We have shown on an example that such coarse-graining can be invaluable in the study of the dynamics of our ideal tree polymers. We have also captured the generation of fragments (the variables featuring in the coarse-grained ODE system) in a simple set of rewrite rules which use an extension of Kappa to the manipulation of partial objects (trees in our case). Finally we have shown that the ED class generates decidable fragmentations.

There are many questions this first investigation suggests. For one thing, can we decide the BED class? Can we replay this analysis with a larger rule class? Here one has to be a bit careful as it is easy to map Turing machines as simple systems of local rules and possibly the fragmentation of an observable could reveal information about termination.

But perhaps the most important question before one moves to a larger class is that of how to define a truncated version. Decidability should ensure that one can write syntactic truncations but as we have seen they might be entirely useless because they fail to propose consistent boundary conditions. Dually, one would like an appropriate version of Kurtz theory [8] which explains why truncations are sometimes working very well as approximations.

Finally, there are questions related to the algebraic aspects of the fragmentation expansion. For one thing, one would like to know in which generic rewriting universe the operations underpinning the construction of the generative rules are possible and for what specific rule class. Another point is that the fragmentation process as we have defined it is entirely syntactical, and will produce observables that are not all linearly independent. When that happens, there is no need to incorporate the new fragment into the current stock. So a natural question is whether we can recognise such situations and use it to simplify the above generation scheme.

References

[1] Blinov, M. L., J. R. Faeder, B. Goldstein and W. S. Hlavacek, *Bionetgen: software for rule-based modeling of signal transduction based on the interactions of molecular domains*, Bioinformatics **20** (2004), pp. 3289–3291.

[2] Cardelli, L., E. Caron, P. Gardner, O. Kahramanogullari and A. Phillips, *A process model of actin polymerisation*, Electronic Notes in Theoretical Computer Science **229** (2009), pp. 127–144.

[3] Danos, V., 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*, IEEE, 2010, pp. 362–381.

[4] Danos, V., J. Feret, W. Fontana and J. Krivine, *Scalable simulation of cellular signaling networks*, Programming Languages and Systems (2007), pp. 139–157.

[5] Danos, V., R. Honorato-Zimmer, S. J. Riveri and S. Stucki, *Rigid geometric constraints for kappa models*, in: *Third International Workshop on Static Analysis and Systems Biology (SASB 2012)*, 2012, to appear in ENTCS (Electronic Notes in Theoretical Computer Science).

[6] Danos, V., H. Koeppl and J. Wilson-Kanamori, *Cooperative assembly systems*, DNA Computing and Molecular Programming (2011), pp. 1–20.

[7] Daoud, M. and Joanny, J.F., *Conformation of branched polymers*, J. Phys. France **42** (1981), pp. 1359–1371.
URL http://dx.doi.org/10.1051/jphys:0198100420100135900

[8] Darling, R. and J. Norris, *Differential equation approximations for Markov chains*, Probability Surveys **5** (2008), pp. 37–79.

[9] Deeds, E. J., J. A. Bachman and W. Fontana, *Optimizing ring assembly reveals the strength of weak interactions*, Proceedings of the National Academy of Sciences **109** (2012), pp. 2348–2353.

[10] Deeds, E. J., J. Krivine, J. Feret, V. Danos and W. Fontana, *Combinatorial complexity and compositional drift in protein interaction networks*, PLoS ONE **7** (2012), p. e32032.

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

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

[13] Henzinger, T. A., M. Mateescu and V. Wolf, *Sliding Window Abstraction for Infinite Markov Chains*, Technical report (2009).

[14] Kapp, G., S. Liu, A. Stein, D. Wong, A. Reményi, B. Yeh, J. Fraser, J. Taunton, W. Lim and T. Kortemme, *Control of protein signaling using a computationally designed gtpase/gef orthogonal pair*, Proceedings of the National Academy of Sciences **109** (2012), pp. 5277–5282.

[15] Kaufman, M., C. Soule and R. Thomas, *A new necessary condition on interaction graphs for multistationarity*, Journal of Theoretical Biology **248** (2007), pp. 675–685.

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

[17] Pawson, T. and P. Nash, *Assembly of cell regulatory systems through protein interaction domains*, Science's STKE **300** (2003), p. 445.