Paolo Dini, Chrystopher L. Nehaniv, Attila Egri-Nagy and Maria J. Schilstra

Exploring the concept of interaction computing through the discrete algebraic analysis of the Belousov–Zhabotinsky reaction

Article (Accepted version)
(Refereed)

Original citation:
Dini, Paolo, Nehaniv, Chrystopher L., Egri-Nagy, Attila and Schilstra, Maria J. (2013) *Exploring the concept of interaction computing through the discrete algebraic analysis of the Belousov–Zhabotinsky reaction.* Biosystems, 112 (2). pp. 145-162.
DOI: 10.1016/j.biosystems.2013.03.003

Exploring the Concept of Interaction Computing through the Discrete Algebraic Analysis of the Belousov-Zhabotinsky Reaction

Paolo Dini[a,b,*], **Chrystopher L Nehaniv**[a], **Attila Egri-Nagy**[a,c], **Maria J Schilstra**[a]

[a]*Royal Society Wolfson BioComputation Research Lab, Centre for Computer Science and Informatics Research, University of Hertfordshire, Hatfield, Hertfordshire, United Kingdom*

[b]*Department of Media and Communications, London School of Economics and Political Science, London, United Kingdom*

[c]*School of Computing, Engineering and Mathematics, University of Western Sydney, NSW, Australia*

Abstract

Interaction Computing (IC) aims to map the properties of integrable low-dimensional non-linear dynamical systems to the discrete domain of finite-state automata in an attempt to reproduce in software the self-organizing and dynamically stable properties of sub-cellular biochemical systems. As the work reported in this paper is still at the early stages of theory development it focuses on the analysis of a particularly simple chemical oscillator, the Belousov-Zhabotinsky (BZ) reaction. After retracing the rationale for IC developed over the past several years from the physical, biological, mathematical, and computer science points of view, the paper presents an elementary discussion of the Krohn-Rhodes decomposition of finite-state automata, including the holonomy decomposition of a simple automaton, and of its interpretation as an abstract positional number system. The method is then applied to the analysis of the algebraic properties of discrete finite-state automata derived from a simplified Petri Net model of the BZ reaction. In the simplest possible and symmetrical case the corresponding automaton is, not surprisingly, found to contain exclusively cyclic groups. In a second, asymmetrical case, the decomposition is much more complex and includes five different simple non-abelian groups whose potential relevance arises from their ability to encode functionally complete algebras. The possible computational relevance of these findings is discussed and possible conclusions are drawn.

Keywords: Interaction, discrete dynamical systems, non-linear dynamics, algebraic automata theory, algebraic invariance, systems biology, open non-equilibrium systems, functional completeness

1. Introduction

The research reported in this paper aims to understand dynamical self-organizing behaviour in sub-cellular systems through an algebraic perspective as part of a wider research programme aiming to achieve self-organizing behaviour in software through algebraic automata theory. In this context, even though the Belousov-Zhabotinsky (BZ) reaction [40] is not biological, it is relevant in several other ways, as we explain below. The objective of understanding the mechanisms and structural properties underpinning self-organization in biochemical systems, therefore, is to a large degree motivated by the desire to replicate such mechanisms or properties in software. Even if self-organizing, the software will have to satisfy specific design or user requirements, implying the presence of a synthetic or engineering design element to the research, in addition to the more familiar analytical perspective that characterizes science in general. Finally, as a longer-term objective, the insights and the greater ability to express complex behaviour on the part of the software that we expect to

[*] Corresponding author

Email addresses: p.dini@lse.ac.uk (Paolo Dini), c.l.nehaniv@herts.ac.uk (Chrystopher L Nehaniv), a.egri-nagy@uws.edu.au (Attila Egri-Nagy), m.j.1.schilstra@herts.ac.uk (Maria J Schilstra) come out of this work may lead to more flexible and scalable modelling frameworks for systems biology than are currently available. As a consequence, it may help the reader to keep in mind that the discussion that follows does not only build on multiple disciplinary perspectives, but is also motivated by a mixture of theoretical and applied research interests.

1.1. Posing the Problem

In self-organizing systems, the “self-” or autonomous aspect is provided by the fall towards equilibrium, which serves as the driver or energy source. As a consequence, a system that needs to maintain self-organizing behaviour indefinitely must be open, since, if it were closed, once it had reached equilibrium it would stop functioning. In order to keep going a self-organizing system must thus be open and connected to a source of (free) energy that can keep it ‘far from equilibrium’, to use Prigogine’s famous phrase [38], even while it is continually falling towards it. All living organisms are open and are kept far from equilibrium by food, implying that, in this context, equilibrium is death. For the biosphere, the Sun is the (free) energy source that is captured by photosynthesis and directly or indirectly drives most of the biological activity on Earth.[1] At the other end, the energy flows out of the biosphere in the form of high-entropy molecular motion as heat, exhausted of its order-construction potential.

The kind of self-organization that we aim to understand in this work is associated with cellular processes – regulatory and metabolic pathways – rather than with evolution, neural processes, or population dynamics. Cell metabolic/regulatory systems are able to self-organize and/or construct order dynamically, through random interactions between their components and based on a wide range of possible inputs. In many systems biology models, the biochemical rate equations of such cellular processes are expressed as systems of ordinary differential equations (ODEs) in the species concentrations, under an assumption of spatially uniform concentration in a well-stirred vessel. Although this is a simplifying assumption and partial differential equations are increasingly being used, the ODE approach has been the workhorse of systems biology for decades and remains the predominant modelling methodology.

The fact that the ODEs populating such models are generally coupled and non-linear lends them a great deal of expressiveness which can capture very complex dynamical behaviour. At the same time, these same properties imply that systems biology is plagued with problems which, in most cases, cannot be solved analytically and, to date, can only be approached with numerical methods. The development of an explanatory and predictive mathematical theory of cellular self-organization, therefore, faces a challenging paradox: the properties that seem to underpin this particular kind of self-organization – openness and non-linear coupling – also make such systems extremely difficult if not impossible to solve analytically. Although numerical methods can be very useful, they are mainly descriptive and cannot form the basis for an explanatory theory. Given a sufficiently large number of molecular species and a sufficiently large number of tunable parameters complex biochemical behaviour can often be reproduced, but the structural principles that are responsible for such behaviour remain difficult to infer. Understanding the causal links between structure and behaviour becomes crucially important in an applied context, whether it be software design or drug development, which motivates us to dig deeper in order to arrive at a proper definition of the problem we are trying to solve.

The contribution of many disparate and granular system components at a small scale of description to an emergent system-wide trend or behaviour at a larger scale of description can be seen as a metaphor for mathematical integration. Regardless of whether we perform a formal integration step or we follow a qualitative/intuitive inductive argument, a “constant of integration” is produced or discovered in the process, which provides an additional, and deeper, level of causation [39]. For example, in frictionless mechanical systems the observed system-wide motion can be understood to be caused by the forces acting on each part of the system at each instant in time through Newton’s 2nd Law, which formalizes the proximate cause-effect

[1] The hydrothermal vents issuing from the mid-Atlantic ridge at the bottom of the ocean is another example of an energy source driving a whole ecosystem, which in this case does not depend on the Sun at all.

relationship between force and motion. However, in the process of integrating Newton's Law (when it is expressed in differential form), we find that one of the constants of integration is the total energy of the system. Knowing that the total energy is constant gives us a deeper and global "cause" for the observed behaviour, along with the feeling that the system behaviour has been "explained" in a more satisfactory way. Analytical solutions of (differential) mathematical models, therefore, are important because they give us insight into the "fundamental" workings of the phenomena they model *and* because such insight then improves our ability to create new applied systems that satisfy specific requirements through engineering design or applied biomedical research.

Predictive and explanatory power can only be as good as the original differential model is expressive, however. Consistently with a growing trend in systems biology, this has motivated us to adopt a critical and open-minded attitude towards ODE models, and to try to imagine how they could be generalized and made more expressive. The problem is not that they are not sufficiently non-linear – in fact, ODE systems can easily exhibit chaotic behaviour – but that they are spatially homogeneous in the species concentrations. By accounting for spatial variation, partial differential equations (PDEs) go a step further, but they still assume that the same species are present throughout the domain. What is needed is a way to break up the domain so that different processes are allowed to take place in different sub-domains, and are patched at the boundaries. Put otherwise, what is needed is a way to represent non-linear behaviour in smaller sub-domains, where different species dominate, and to allow the sub-domains to interact, perhaps also non-linearly. Thus, rather than dealing with a single completely non-integrable and chaotic system, a better approach – that reflects the ordered and stable behaviour of healthy organisms – may be to aim for multiple *mildly* non-linear systems that are "integrable" in some suitable sense and that interact with each other in a similar manageably non-linear way.[2] In other words, it seems that the complexity should be lower at any individual scale of description, but distributed over multiple scales and across different interacting systems.

Whereas a possible approach could be to use different ODE systems in each of the separate sub-regions, coupled through suitable boundary conditions, the problem with such an approach is that its analytical complexity increases very quickly with the addition of each new ODE system and sub-domain. It would be very unlikely for analytical solutions to be obtainable, or for a theory to be developed that can account for the inherent multi-scale structure of such a composite domain simultaneously with the analytical solution of each of the domains. A numerical approach would be the only sensible option, but as we discuss at various points in this paper we remain unconvinced about the explanatory power and generalizability of numerical solutions.

It is partly for this reason that in this line of research we have sought to complement more traditional systems biology and dynamical systems modelling methods with the radically different discrete perspective of computer science. That dynamical systems map to automata has been known for a long time [27], but the part that is more difficult to model is the coupling between different degrees of freedom or between different sub-domains. We are postulating that it should be possible for the behaviour of cellular systems to be reproduced in a controllable way through interacting finite-state automata. If successful, this research programme will lead to a radically new model of "bottom-up computation" with equal applicability to computer science and systems biology that we call Interaction Computing (IC). This paper discusses the current status and the latest results of our research towards this goal.

[2] We should not lose track of the importance of the system's energy. For example, in the Elastica non-linear Hamiltonian system [5] as one explores higher-energy shells (by choosing suitable higher-energy initial conditions) the response changes from periodic to quasi-periodic to chaotic. In open, non-linear dissipative systems, as the energy flow through the system increases the response similarly tends towards chaos. So in the development of a given modelling strategy one should keep both factors in mind: the functional form of the coupling terms and the system's energy.

1.2. Symmetry

The concept of regularity as an expression of order can be made more precise by relying on the mathematical concept of symmetry.[3] Since the mathematical formalization of invariance is expressed through the language of algebra, IC research benefits from the merging of two research traditions: dynamical systems theory rooted in physics and informing much of modern-day systems biology, and theoretical computer science rooted in algebraic automata theory. In other words, since self-organizing behaviour in biological systems is a consequence of regularities of physical laws, the IC concept has been developed mainly from a statistical physics, systems biology, and dynamical systems point of view in a number of EU projects[4] since 2003 [8, 9, 25]. On the other hand, the concept of interacting automata stretches back to the origins of algebraic automata theory in the 1960s [30, 29, 43], and has greatly benefited from recent work in the same mathematical tradition [37, 14, 17, 13, 11, 7].

In addition to openness and non-linearity, a third very challenging property of biological systems from the point of view of model construction is their nested structure, which is to some extent mitigated by the scale invariance of some of their dynamical properties. In other words, some of the abstract non-linear properties appear to show up in similar ways at sub-cellular scales as well as on the scale of whole populations of organisms, as demonstrated by the similarity in the corresponding (non-linear) ODE models of such systems. The manner in which different scales depend on each other, on the other hand, is a rather more difficult phenomenon to understand and model, as discussed below in the context of the cascade decomposition of finite-state automata. In any case, the assumption of sufficiently similar order construction processes at widely different scales enables us to draw correspondingly disparate theories together under the same conceptual framework.

For example, as shown metaphorically in Figure 1, in software systems the role of energy is played by information, and the source of information are the users. Thus, the collection of all human minds interfaced to the system serves a role analogous to the sun for the biosphere or food for a biological organism. If we replace the biosphere by an individual cell, a similar framework is assumed to apply, and similar interaction rules are assumed to be derivable from the biology and mappable to finite-state automata modelling or implementing much smaller digital systems. In other words, the regularities observed in biological systems are formalized as symmetries in the structure and behaviour of the corresponding discretized models, which are then to be expressed as constraints on the possible structure and behaviour of the software. This paper begins to develop some possibilities for what this might mean specifically.

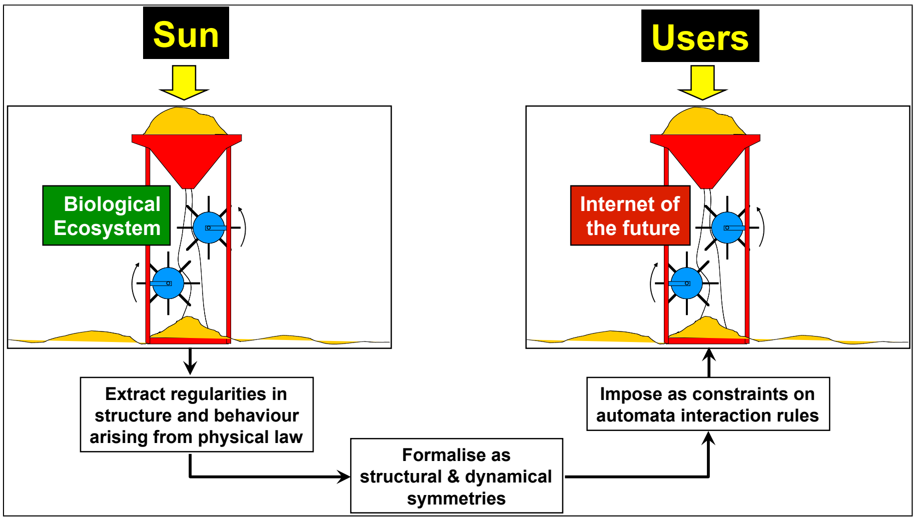

Sun
Users
Biological Ecosystem
Internet of the future
Extract regularities in structure and behaviour arising from physical law
Formalise as structural & dynamical symmetries
Impose as constraints on automata interaction rules

Figure 1: **The beach-toy metaphor**

[3] A symmetry is an invertible transformation that leaves some feature of a mathematical object invariant. The set of invertible transformations of a mathematical object that leave some feature of that object invariant always forms a group.

[4] http://www.opaals.eu, http://bionets.eu, http://internet-science.eu

In the remainder of the paper, Section 2 discusses the analytical framework we have developed for building a bridge between cell biology and computer science and the rationale for discretizing biochemical processes to obtain the corresponding finite-state automata. Section 3 introduces some of the key concepts from algebraic automata theory needed for the analysis of the BZ reaction in Section 4. Section 5 discusses some of the possible interpretations of the results of the mathematical and computational algebra analysis, along with the likely next steps in the research. Finally, Section 6 draws some conclusions.

2. Research Strategy and Methodology

The computational interpretation of the algebraic structure of automata derived from biochemical systems, as presented in the algebraic automata theory references given above, is far from obvious. Therefore, one of the purposes of this paper is to compare well-understood features of dynamical systems to features of the algebraic structure of the automata derived from the *same* biochemical systems in the hope that the former will shed some light on the latter. For example, the fact that periodic oscillations are probably the most clearly understood property of dynamical systems was the main (epistemological) motivation for choosing the BZ reaction as an object of study. In other words, although the BZ reaction is not necessarily the simplest chemical oscillator, it is simple enough that it could be said to play a role for open and non-linear chemical systems that is analogous to the role the simple harmonic oscillator (SHO) of Classical Physics plays for non-linear mechanical oscillators.

Thus, rather than breaking up the physical domain immediately, our strategy is first to strengthen our understanding of how the mathematical properties of ODE systems map to the mathematical properties of the discrete automaton formulation of the same (bio)chemical equations. The strategy is to use this mapping to strengthen the explanatory theories of dynamical and computational behaviour in the two domains, respectively, and to establish a virtuous circle between them whereby each theory informs the other, and can be made more sophisticated and expressive with each iteration. This is the second (methodological) reason why in this paper we focus on a relatively simple ODE systems such as the BZ reaction as the starting point of the research.

2.1. Analytical Framework

The BZ reaction has been studied extensively [40] because it was the first reaction to exhibit sustained oscillations even in an isolated system, although they do die down eventually. Before Belousov's discovery in the 1930s and Zhabotinsky's confirmation of the phenomenon in the 1960s, species concentrations were believed to vary monotonically unless driven by a periodic forcing function. By contrast, in a constant-flow stirred-tank (CFST) reactor the BZ oscillations are periodic and, counter-intuitively, can be sustained indefinitely as long as the inflow and outflow are kept *constant*. This qualifies the BZ reaction as an open system far from equilibrium. In this paper we analyse a simplified version of an ODE model of the BZ reaction, the almost equally famous 'Oregonator' model, developed by Field and Noyes at the University of Oregon [19].

Oscillations have been observed also in cellular pathways such as the p53-mdm2 regulatory cycle. As discussed at some length in [6], in [32] fluorescent labelling was used to monitor simultaneous changes in expression of p53 and mdm2 following irradiation, and sustained oscillations were reported. However, the experimental work in [6] shows that, somewhat unexpectedly, the labelling of mdm2 renders it functionally inactive. This raises the possibility that the oscillations observed in the single-cell fluorescence experiment [32] might be due to artefacts induced by the abnormalities in mdm2 function introduced by the labelling procedure. Nevertheless, there is good evidence [33, 6] that the p53/mdm2 system *does* oscillate – but that these oscillations are damped. There are several possible explanations for such damped oscillations – for example that p53 and mdm2 operate in a simple feedback loop but with a discrete time delay [35].

Be that as it may, the point is that oscillations appear to play a central role in one of the most important regulatory pathways of the cell. Further, ideally we would like our theory to be able to handle not only periodic or damped oscillations, but more complex behaviour that may not be solvable analytically by present methods. In other words, the definition of the kinds of problems to be solved should be derived from biological systems that are functioning and "healthy", even if such systems happen to correspond to mathematical models that we cannot currently solve by elementary analytical methods. We assume that it may be possible to define formally the boundary between healthy and pathological behaviour; since the theory is not advanced enough for such formal definition, however, in the meantime we use the intuitive concept of "dynamical stability" as a synonym for "health". The indications from dynamical systems theory are that the notion of dynamical stability links to what we said above about models that are larger and scalable, but with a wide range of possible behaviours between rigid periodicity and chaos.

Dynamical stability is an intuitive concept that can be associated with a middle ground between entirely predictable and totally chaotic behaviour. In other words, it pushes the boundary of integrability of the dynamical governing equations of a system. Integrability of ODEs is intimately linked with their Lie group structure.[5] Thus, a way in which the dynamical stability of the systems of ODEs associated with cellular pathways could be studied is by searching for so-called global symmetries through a Lie group analysis. Finding Lie symmetries in systems of ODEs is a very complex task; however, luckily it is amenable to algorithmic formalization [46, 41, 26]. Several open source implementations of Lie's method have been developed and are available for a number of mathematical analysis environments such as *Mathematica* [2].

The problem with Lie symmetries is that they are global: to each global symmetry of a system there corresponds a globally conserved quantity [39] (i.e. the "constant of integration" we mentioned above). When this happens the differential mathematical model can be integrated one step towards its analytical solution, i.e. its order can be reduced by a whole unit. This is great but, for the same reason, it does not happen very often. And when it does happen, it can be as much a consequence of mathematical idealization in developing the model as of the presence of a truly fundamental conserved quantity. Another way to put it is that Lie symmetries are "rigid": they either apply to the whole system or not at all. By contrast, biological systems are full of compromises. Ordered structures or behaviour may apply to a particular sub-system, or over a certain parameter range, or at certain times, but never for all time, space, and parameter values. Therefore, it is likely that Lie's theory will need to be generalized, for example from Lie groups to Lie groupoids [47], which is the analytical analogue of the domain fragmentation we mentioned above. In any case, we think that a critical and iterative dialogue between continuous and discrete models of dynamical systems may well be the best way to advance the mathematical theories of both. Thus, while on the one hand IC research is necessarily concerned with the study of universal mathematical properties of dynamical systems, on the other hand we have also placed a great deal of attention on how sub-cellular systems can be mapped to discrete automata [14, 17].

With this context, we can finally show in Figure 2 the analytical framework that underpins the emerging mathematical theory of IC in general and the work discussed in this paper in particular. The purpose of this diagram at this point of the paper is mainly motivational; it is explained more fully below, in the discussion of the results. In the rest of the paper we focus mainly on the right-hand side of this diagram. In the figure, 'SNAG' stands for 'simple non-abelian group' and will be discussed later in the paper.

2.2. Discretization Method

To be able to analyse the (Oregonator model of the) BZ reaction from the perspective of algebraic automata theory we must find a way to discretize it and we must convince ourselves that doing so retains the main features of the dynamics of the original system. Cell metabolism relies on ultimately undirected bottom-up and random/stochastic processes that can only "execute" through the spontaneous interaction of the various components. The interactions are driven by a combination of electrostatic forces (usually conceptualized as minimizing the potential energy of interaction) and most probable outcomes (maximization of entropy). In spite of this, however, a healthy cell behaves in an organized and finely balanced way that is more evocative

[5] Lie groups are continuously parameterized transformations of the dependent and independent variables of a problem that leave the functional form of the ODEs invariant; thus, each element of a Lie group is, by definition, a (Lie) symmetry. Because they are continuous, Lie groups can also be visualized as manifolds (i.e. (hyper)surfaces) defined in the same space as the solution of the ODEs.

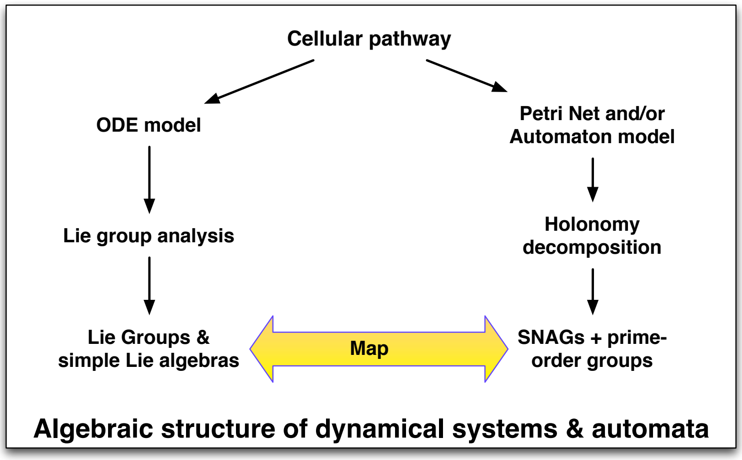

Cellular pathway
ODE model
Petri Net and/or Automaton model
Lie group analysis
Holonomy decomposition
Lie Groups & simple Lie algebras
Map
SNAGs + prime-order groups
Algebraic structure of dynamical systems & automata

Figure 2: **Analytical framework for mathematical research in Interaction Computing**

of a deterministic, even if very complex, machine than of random chaos. The cell in fact has a definite physical structure and executes well-defined "algorithms" in the form of cellular processes such as metabolic or regulatory biochemical pathways. This suggests a description and modelling of the internal dynamics of the cell at a level of abstraction that is higher than the molecular, and through mechanisms or constraints that are complementary to stochastic processes.

In particular, our perspective views the stochastic nature of cell biochemistry mainly as a mechanism of dimensional reduction that does not necessarily need to be emulated in detail. For example, a gene expresses hundreds of mRNA molecules which, in turn, engage hundreds of ribosomes for no other reason than to maximize the probability that a particular, single genetic instruction will be carried out, such as the synthesis of a particular enzyme. As a consequence of this dimensional reduction (hundreds to 1), a higher level of abstraction than that at which stochastic molecular processes operate is justified in the modelling approach – in particular, a formalization that retains, and builds on, the discrete properties of cell biology. In this manner, the practically continuous distribution of millions of molecules per species is discretized into a handful of species concentrations. However, even the resulting lower-dimensional system can't plausibly be imagined to perform the complexity of a cell's functions driven simply by a uniform distribution of interaction probability between its (now fewer) components. Additional structure and constraints must be at play.

Not surprisingly, additional structure that is consistent with a low-dimensional description of the biochemical process is provided by casting the problem as a low-dimensional dynamical system derived from the biochemical rate equations expressed as (usually coupled and non-linear) ODEs in the species concentrations. Furthermore, the system can be discretized also in the time dimension according to explicit and implicit time models ([43]: 38). Transformations can simply be induced by the passage of time in discretized steps, so in this case time appears explicitly as the input of the system. However, there is another interpretation where the inputs of the system are perturbations to steady states and the state transition finishes when the system settles again to a new steady state, regardless of the time it takes. Thus time only appears implicitly and the dynamical system has a more natural discrete structure. On the strength particularly of the latter view, cellular processes can be treated as finite-state automata or discrete low-dimensional dynamical systems. This appears to be the most appropriate level of abstraction and entry point to understand biological construction of order in a way that is relatively easy to transfer to computer science. A good way to achieve this is with a Petri net (PN) model, although Kauffman's Boolean networks [28, 14] and Rhodes's reaction digraphs [43] are also useful possibilities, all amenable to algebraic automata-theoretic methods.

The PN notation, invented to describe interaction and transformation in discrete distributed systems [42, 13], is highly suitable to depict the structure of biochemical reaction networks at the level of interaction between molecules, but can also account for discretized species concentration levels. In combination with kinetic information, PN models are useful tools in the derivation of the coupled ODE systems that describe the dynamic behaviour of these networks [44] (or vice versa, as we are doing here). Once such a PN has been obtained, it is straightforward to derive a finite-state automaton by treating each possible marking of the PN as a different state of the automaton. In order to obtain a finite-state automaton the number of tokens is bounded, and the bound is called the "capacity" of the place. This can be viewed as discretizing concentration levels or bounding the number of molecules of each type. Since the resulting number of states can be very large, normally the markings-to-states mapping is done for a specific choice of initial conditions, which yields a subset of the global automaton of all possible states.

In the next section we present the basic concepts of algebraic automata theory that we need in order to begin to understand the algebraic analysis of the computation in the BZ system.

3. Krohn-Rhodes Theory

A finite-state automaton can be defined mathematically as a finite set of transformations acting on a finite set of states Q. In general, these transformations can be composed by following one transformation by another one. The set of all finite sequences of transformations thus yields a set of tranformations that satisfies the axioms of a *semigroup*, meaning that it is closed with respect to a multiplication law (here, functional composition) that satisfies the associative property. However, not all of its members need have an inverse. By convention, the empty sequence of transformations yields an identity transformation so, including this, what we have is actually a monoid (i.e. semigroup with identity element). For mathematical convenience we may always include the identity transformation, although we continue to use the term 'semigroup'. Further, all semigroups we will be concerned with will be finite. Thus, the algebraic version of a finite-state automaton is a 'transformation semigroup', or 'ts' for short, defined as

$$X = (Q, S), \tag{1}$$

where Q is the set of states and S is the semigroup acting on Q. S can be regarded as a subsemigroup of all maps from Q to itself under composition. The ts is a direct generalization of the permutation group concept to semigroups. As for permutations groups, a potential cause of confusion arises from the fact that one element s of the semigroup S of a given ts X acts as an operator on *all* of the states $q \in Q$ "simultaneously". In other words, the element $s \in S$ should be seen as the whole function $s : Q \to Q$ that is defined over the whole state set at once. By contrast, the execution of a single step of a given algorithm implemented by a given automaton yields a 'state transition' and should be seen as a single *value* of such a function, for a given starting state q:

$$s(q) = q'. \tag{2}$$

We revisit this point below.

3.1. The Krohn-Rhodes and Holonomy Theorems

The Krohn-Rhodes prime decomposition theorem for finite automata [30] has been discussed, explained, and applied in a large number of books and articles since the theorem was published in 1965 ([1] and many others). The language and formalism found in the proofs in the various sources can be either from automata theory or from algebra, the former possibly being more accessible to a wider audience. Therefore, Table 1 shows the statement of the theorem along with the main definitions and concepts as expressed in these two "languages". In the rest of the paper, where possible, we continue to use both notations, hoping that the two descriptions side-by-side will make the concepts more understandable than they would be if presented in either formalism by itself.

In this table, $\mathcal{A}$ is the symbol for a (semi-)automaton, which is defined as a triplet[6] composed by an input

[6] We do not explicitly consider the output of the automaton as it can be recovered from the state and the input symbol.

Automata Theory	**Algebra**
Any deterministic automaton can be decomposed into a cascade of two kinds of elementary automata: prime group automata and two-state identity-resets (flip-flops).	Any transformation semigroup is the homomorphic image of one embedding in the wreath product of two kinds of fundamental algebraic objects: simple groups and flip-flop monoids.
$\mathcal{A} = (A, Q, \delta)$	$X = (Q, S)$
Automaton	Transformation semigroup
Cascade	Wreath product
Cascade decomposition	Homomorphic image of subsemigroup of wreath product
Elementary automaton	Irreducible semigroup
Group automaton (e.g. counters, loop)	Finite permutation group
Flip-flop, memory overwrite	Identity-Reset semigroup, constant map

Table 1: **Translation table between automata theory and algebra terms**

alphabet A, a set of states Q, and a state transition function $\delta : Q \times A \rightarrow Q$ such that

$$\delta(q, a) = q', \qquad q, q' \in Q, \quad a \in A \tag{3}$$

In the algebraic notation, by contrast, the corresponding object is a ts $X = (Q, S)$ as we described above with semigroup S generated by all the transformations $a \in A$.

Figure 3 shows different types of action that a semigroup element $s \in S$ can induce on a set of six states. In Case 3a, s is a member of a group that is a subgroup of S; in Case 3b, s is a constant map, which is non-invertible; the remaining two cases are other examples of non-invertible transformations. Thus, in cases 3b, 3c, and 3d, s cannot belong to a group action that permutes the states. Maler states it in a somewhat different way: 'A permutation is reverse-deterministic, while in resets the degree of reverse non-determinism is maximal' [34]. The attribute 'fundamental' that we used in Table 1 refers to the fact that the elementary automata[7] or algebraic structures cannot be decomposed further, so that they can be regarded as analogous to the prime factors of an integer – hence the name of the theorem. Furthermore, permutation groups and identity-resets could be regarded as "mutually orthogonal" in the sense that no combination of one type can yield a member of the other type. In other words, they are analogous to a generalized basis into which an automaton can be decomposed.

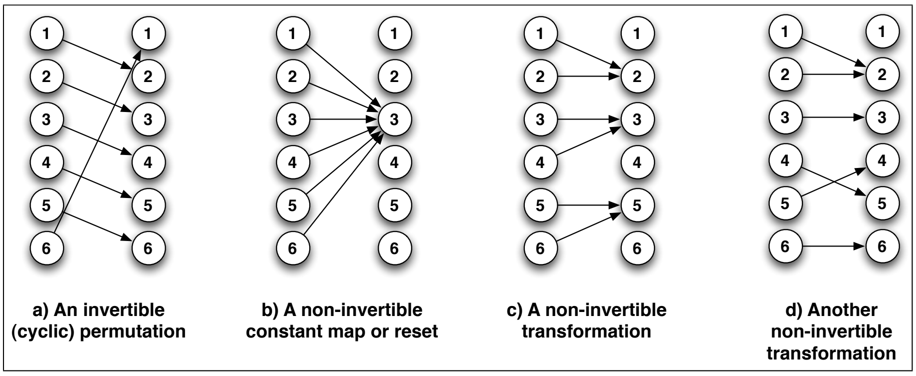

1
2
3
4
5
6
a) An invertible (cyclic) permutation
b) A non-invertible constant map or reset
c) A non-invertible transformation
d) Another non-invertible transformation

Figure 3: **Different kinds of transformations of 6 states (after [34])**

In 1967 Zeiger [48] proved a variant, called the holonomy decomposition (HD) theorem, according to which any finite automaton can be decomposed into a cascade of permutation-reset automata arising from a study of how inputs act on certain subsets of the powerset of the state set. A permutation-reset automaton is one whose semigroup has only two kinds of elements: the first kind of element is a permutation and therefore belongs to a group, which is a subgroup of S and therefore is analogous to Case 3a; the second kind of element is a constant map and analogous to Case 3b. Any other kind of non-invertible transformation, such as 3c or 3d, cannot occur

[7] These turn out to be automata whose transition semigroups are irreducible, i.e., as Krohn and Rhodes proved, simple groups or subsemigroups of the flip-flop (See [43]).

in a permutation-reset automaton. The permutation-reset automata can then be further decomposed into (finite and discrete) groups and two-state identity-reset automata (also known as flip-flops).[8] Finally, using the Jordan-Hölder theorem ([3]: 244) each group can be further subdivided into a sequence of simple groups,[9] known as its composition factors, recovering the Krohn-Rhodes decomposition into irreducible "atomic" groups (simple groups) and combinatorial semigroups (cascades of banks of flip-flops). One of the clearest and most accessible explanations and proofs of the holonomy theorem is due to Maler [34].

The HD has continuously been improved in efficiency over the years (e.g., [18, 23, 10, 34, 12], finally leading to computer algebraic realization ([17], which has more recently been reimplemented in `GAP` [20] as `SgpDec` [16]), making possible the decomposition and analysis of structures previously well beyond human capacity to analyse.

3.2. Hierarchical Structure

One of the reasons the holonomy (or Krohn-Rhodes) theorem is difficult to grasp is that the decomposition is hierarchical: the hierarchy refers to a particular description of the action of the semigroup on its set of states, whereby each level indicates the action of subsets of semigroup elements on certain sets that are collections of subsets of states. The latter are the dynamic analogues to the physical subdomains of the previous section. 'Subsets of semigroup elements acting on certain sets that are collections of subsets of states' is analogous to identifying a ("coarse-grained") sub-automaton.[10] Thus, the components are arranged in a manner that can be viewed as a partly nested hierarchy of sub-automata, or groups and identity-reset semigroups acting on nested sets of states. This means that a state that is being acted upon by a group element at one level, for example, could in fact be a macrostate composed of multiple states, each of which belongs to a lower level and is acted upon in another way by semigroup elements at that lower level. This structure is recursive and the number of levels reached can be very high. We say 'partly nested' because at any one level multiple groups and identity-reset semigroups could be present, each of which can belong to different and partly overlapping nested hierarchies of states and transformations, i.e., each of which can act upon different and partly overlapping collections of (macro)states at the same level. This implies that each level of the hierarchy can be interpreted as separate parallel machines or sub-automata operating at a different "scale".

Another important property of the holonomy decomposition is that a part of the "resolution hierarchy" relates higher levels to lower levels. This is called the "loop-free" property in the sense that the unidirectional downward mappings imply that lower levels depend on higher ones, but not the reverse. A very useful example to clarify this point is the unidirectional dependence of the components of a human arm with respect to position in space: if only the fingers move, this action has no effect on the position of the upper arm, whereas if the upper arm moves then the fingers will necessarily have to move with it. Although some levels are not reachable from the levels above, as one moves down the hierarchy the subsets of states shrink in size. This is why the hierarchy can also be interpreted as a "nested" cascade of automata, where the resolution of the automata increases as one moves down the hierarchy (the macrostates get smaller until they correspond to single states on the bottom level). The algebraic construction that formalizes these properties is the wreath product. It turns out that the wreath product of the elementary components of an automaton encodes a "larger" automaton than what one started with (this is shown explicitly through a detailed example by Maler [34]). This is why we say that a transformation semigroup "embeds" into the wreath product of its holonomy decomposition components. More precisely, we can also say that the original transformation semigroup is a homomorphic image of a sub-transformation semigroup of a wreath product of its irreducible components. Alternatively, this can be stated as: An automaton can be emulated by a subautomaton of the cascade of elementary automata that correspond

[8] In an *identity-reset automaton* with state set Q each input either acts as the identity map or as a constant map (reset) on Q.

[9] A simple group has no nontrivial normal subgroups and therefore cannot be decomposed further (see, e.g. [3] for more details).

[10] Actually the inputs to the "sub-automaton" may be words, not necessarily letters, of the original automaton. Moreover, the action in the holonomy decomposition using these words, say, is to map subsets of states to other subsets of states in a manner such that at the level of subsets each such mapping is either a permutation or a reset (for details see any reference on the holonomy decompostion, e.g. [43, 10, 18]).

to irreducible “pieces” of its semigroup.

If a group is present at any one level, it indicates a permutation group that acts on subsets of states from the levels below. For example, if A_5 were present in the decomposition of a given automaton, it would indicate that there are elements of the semigroup of that automaton that satisfy the group axioms and permute subsets of states in the levels below, and necessarily this subgroup of the semigroup would either be isomorphic to A_5 itself or map homomorphically onto A_5 acting faithfully on the permuted subsets.

An example of a single permutation is shown in Figure 3a in the form of a single element of the cyclic group C_6. The fact that a group may be operating on a (sub)set of states has several interpretations algorithmically. For example, if we define a system state as a singleton (which is the default definition in applications), by such definition of “state” a given system cannot be in more than one state at any one time. In other words, the 6 states shown in Figure 3 (for any of the 4 cases) cannot all transition at the same time. Therefore, in this case the most natural interpretation of the action of each group or semigroup element operating on a given subset of states is as state transitions that are induced by the same element but that can potentially occur *at different points in time*. In other words, according to this interpretation the same semigroup element happens to be responsible for the transitions between different pairs of states that might otherwise be regarded as entirely unrelated. This concept hints at the possibility that using semigroup elements explicitly could lead to a more efficient formalization or encoding of algorithms.[11] In addition, the presence of groups embedded within the semigroup of a given automaton obviously implies that (1) such subsets of transitions comprising the group are intimately related by the closure property with respect to functional composition – a fact that to be sure applies also to subsemigroups – and (2) each of the transformations belonging to one such group has an inverse. In other words, that automata whose semigroups have subgroups are generally characterized by pools of reversibility embedded in a larger “sea” of irreversible computation [37, 14, 43].

3.3. Visualization

It is most fruitful to regard a HD as a particular *representation* of the original finite-state machine. In other words, the fact that a particular automaton is in a particular state can be shown by highlighting that state in a normal state transition diagram, but – in principle – it can also be shown graphically on the decomposition diagram of that same automaton. Thus, although a direct visualization of a HD does not yet exist, Figure 4 is an attempt to show this intuitively rather than with a formally precise diagram because the latter would be too complex, defeating the purpose to clarify the most basic concepts. In particular, Figure 4 shows two states of the original automaton, q and q', and the transition between them. Assuming that the original automaton splits into three components, at three different levels, each level is in principle a separate machine with its own set of states and set of transformations acting on the states. However, when the interdependencies shown in Figure 4 are enforced, these three machines taken *together* are closely related to the original automaton. Thus, the starting state of the original automaton, q, could be thought of as a vector that is “projected” onto its components (3 in this case), yielding a component representation as $q = (q_1, q_2, q_3)$. The transition in the original automaton (not shown) corresponds to the three transitions that are shown here, ending in a hypothetical $q' = (q'_1, q'_2, q'_3)$. The semigroup elements acting on the three sub-machines are shown as s_1, s_2, and s_3. As shown in the figure, whereas s_1 depends only on the input symbol, s_2 and s_3 are given by dependency functions taking increasing more arguments with each level.

For more realistic examples we have developed a visualization that we call the “skeleton” of the HD and that shows the essential features of the structure. Skeleton diagrams do not show an automaton’s semigroup explicitly or, if any, its embedded groups. They show the subsets of states that are being acted upon by these different semigroup elements or by these different kinds of groups. Before discussing an example in more detail we need to introduce an additional concept whose importance cannot be overstated.

[11] Alternatively, action on a subset by the elements of the semigroup can be interpreted as acting in parallel, simultaneously, on multiple copies of the system via a natural diagonal embedding. Algorithmically, this could lead to parallel computation possibilities.

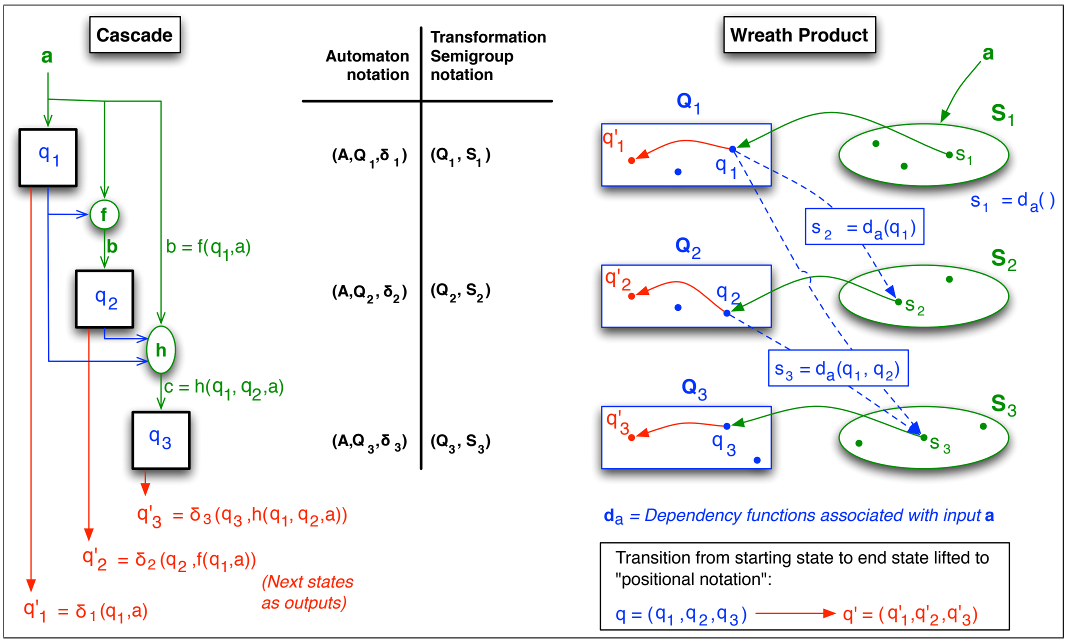


Figure 4: **Comparison of cascade and wreath product decompositions**

3.4. Automata as Generalized Number Systems

A crucially important point for understanding what the HD says about an automaton is that the *whole* hierarchy is needed to indicate each state of a machine, with each level as one moves downwards narrowing down the resolution. Although this point may at first sound difficult to grasp, we all first learnt it in elementary school: we are only talking about the expansion of the state set of an automaton into a generalized positional "number" system in a way that is formally and precisely analogous to the expansion of a given integer into a positional number system in a given base. For example, if we were to say that to represent in base-10 the integer expressed as the following list of sticks

$$| \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \quad | \tag{4}$$

we need to use *both* the units and the tens positions of the decimal system in order to write down '14', nobody would find such a statement particularly strange. In other words, just as a given "number" is a Platonic concept that can have many different incarnations or representations (positional, Roman, etc), so can the state of an automaton be represented in different ways. The HD is a way to make explicit, through a generalized positional notation (of uncertain and variable base!), a generalized number system that a given automaton *implicitly* defines [43]. For this reason the HD of an automaton is also referred to as its 'coordinatization' [36, 37]. Informally, a coordinate system can be considered to be a notational system for some object or process. Every state and transformation of the modelled system is described by an ordered list of coordinate values coming from simpler basic building blocks, the components. By using coordinates we can address the components separately and also their relations in a decomposition. Therefore, a coordinate system creates the possibility of developing simple algorithms to manipulate the original structure via the coordinates. For instance, in the Cartesian coordinate system we can uniquely specify any point of the n-dimensional Euclidean space by n coordinates and define vector operations coordinatewise. The Cartesian coordinate system is an example of an inherently non-hierarchical coordinate system for a totally homogenous system, therefore the coordinates can be reordered arbitrarily. More generally, however, different coordinate positions have different roles, addressing parts of the system different in size or in function. If the ordering of the coordinate positions is not arbitrary then we talk about hierarchical coordinate systems. A prominent example is our positional number notation system, a coordinate system built from copies of C_{10}, modulo-10 counters. It is hierarchical since shuffling the coordinate positions yields different numbers, and it is loop-free since the carry goes only in one direction. The algorithms to calculate with these coordinatized numbers are so simple and powerful that they are taught in primary schools. In general these coordinate systems can be called cognitive tools for understanding and manipulating transformation semigroups and the processes modelled by them.

3.4.1. Decomposition of Modulo-16 Counter as a 4-Bit Counter Automaton

To explain these concepts in more detail we analyse a modulo-16 counter, which we can think of as the right-regular representation of the cyclic group C_{16}, i.e. as C_{16} expressed as a permutation group acting on itself. For this simple example, we will show how the familiar 4-bit binary representation arises as a 'coordinatization'. Table 2 shows how such an object could be rendered as an automaton and as a permutation group. Figure 5 shows the automaton version in binary and decimal notation, where we treat each number reached by the counter as a different state.

Automata Theory: $\mathcal{A} = (A, Q, \delta)$	**Algebra**: $X = (Q, S)$
$A = \{0, 1\}$	$S = C_{16} = \{0, +1, +2, +3, +4, +5, +6, +7, +8, +9, +10, +11, +12, +13, +14, +15\}$ (mod 16)
$Q = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$	$Q = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$
(For example:) $\delta(2, 0) = 2$, $\delta(2, 1) = 3$, etc	Identity = 0 Generator = +1

Table 2: **16-counter as automaton and as permutation group**

The HD is not needed and strictly does not apply since we started with a group. However, this example is useful for introducing the graphical notation used to depict the different levels of the HD, which we will use below in the analysis of rather more complex automata. The Jordan-Hölder theorem tells us that C_{16} can be decomposed using the composition series $C_{16} \trianglerighteq C_8 \trianglerighteq C_4 \trianglerighteq C_2 \trianglerighteq \{e\}$, with the quotient of any two successive series components in this case equalling the same factor C_2, which is trivially simple. Now, Figure 6 shows how the four C_2 factors can be arranged as four binary counters to match the binary positional notation of any number within the original set of 16. Further, Figure 6 also shows how the elementary operation $13 + 1 = 14$ can be expressed as the result of these four elementary automata or binary counters acting together. The rule by which they do so is the familiar 'carry-bit rule', which we are all able to do automatically although here it is formalized as a complicated-looking set of dependency functions between the levels.

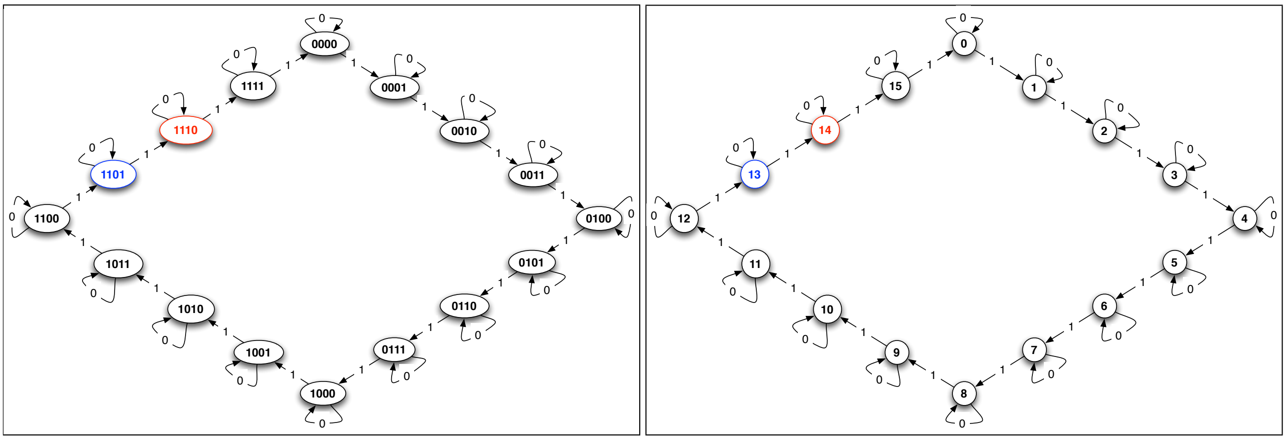


Figure 5: **Automaton representation of 16-counter in binary and decimal notation**

More specifically, when using the carry-bit rule, the input is fed only to the least significant bit (LSB), and it is "carried" to the next bit only if the LSB = 1. At the second bit the same rule is applied again, and so on. When using dependency functions, on the other hand, we can envisage the input +1 as being applied separately to all 4 levels at the same time. The state transition of any of the binary counters at a given level then depends on the states of the levels above, implying that the highest level is free to transition subject only to the input symbols. Thus, when +1 is fed to the counter, the top level (LSB) will always toggle between 0 and 1. The level below, on the other hand, will switch from 1 to 0 or from 0 to 1 *only* if the top level is in state 1 when the +1 input symbol arrives. This is shown by the string '(1)' next to the label of the transition arc. Level 3 transitions only when *both* the levels above are at 1, shown as '(1, 1)'. Finally, the lowest level (most significant bit, MSB) will toggle only when *all three* levels above are simultaneously at 1, shown as '(1, 1, 1)'. The dependency functions

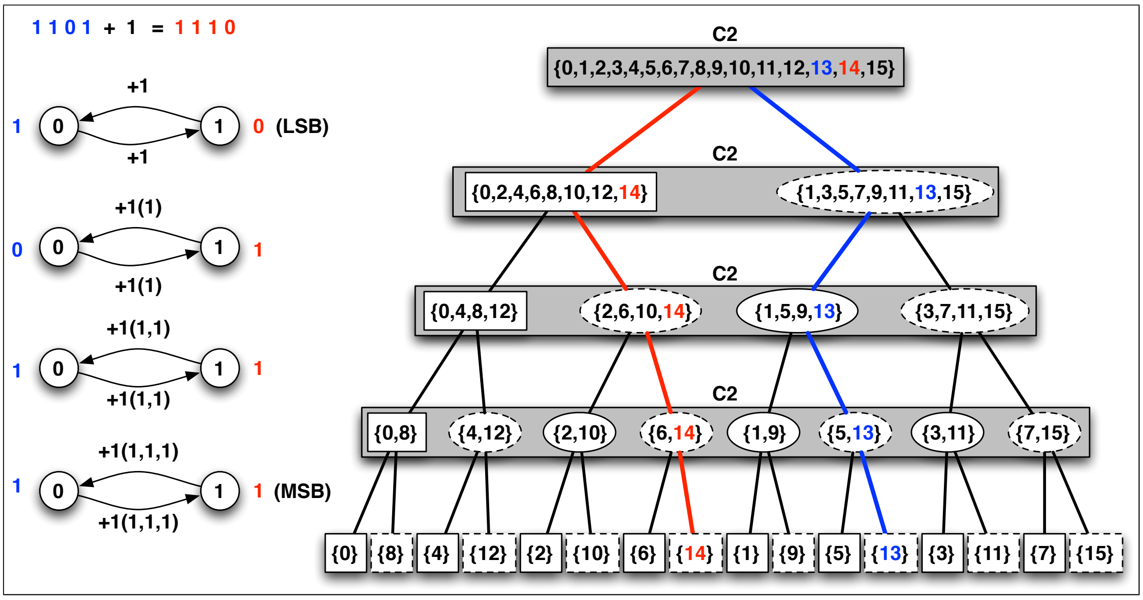


Figure 6: **Left: Binary positional notation for non-negative integers $<$ 16 expressed as a composition of binary counters, with dependency conditions (carry bit) shown explicitly. Right: Corresponding group coordinatization in decimal notation.**

at each level could in principle be anything, i.e. '$(1, 0, 1)$', but if we want the counter to behave correctly, i.e. in a manner equivalent to the carry rule, they need to be implemented as explained and as shown.

On the right of the figure, the numbers 13 and 14 are represented as *states* of an automaton that can take on any value from the original set of states. This representation is highlighted with Blue and Red lines, respectively, connecting the subsets of the original set, of increasing resolution, that contain the number in question at the different levels. The top box corresponds to the coarsest resolution for the number 13: all it says is that 13 is somewhere in the set of numbers $\{0, \cdots, 15\}$. The 'C2' label above the box refers to the fact that there is a constituent subgroup of C_{16}, C_2 in fact, that operates on *two* subsets of states, or two macro-states. The same is then true also for each of these two, which therefore fork again, and so forth.

'C_2 operates on two subsets of states' means, more precisely, that there exists a two-element group which maps the two subsets to each other. Since one element must be the identity, that leaves only one transformation. Thus, the C_2 shown above the top level refers to the group $\{0, +1\}$ acting on the two sets of states $\{0, 2, 4, 6, 8, 10, 12, 14\}$ and $\{1, 3, 5, 7, 9, 11, 13, 15\}$. This means that if $+1$ (mod 16) is added to any of the states in one subset an element from the other subset is obtained, which is certainly true. The second level *also* has a C_2 label above it, indicating that each of its two subsets, in turn, splits into two more subsets. Furthermore, the same (new) C_2 group will map the two 4-state subsets in each pair to each other, swapping them; in this case this (different) $C_2 = \{0, +2\}$ acts, e.g. on 'macro-states' $\{0, 4, 8, 12\}$ and $\{2, 6, 10, 14\}$, by swapping them. The next group, operating on the next level with macro-states (called 'cosets' in group theory) $\{0, 8\}$ and $\{4, 12\}$, is $C_2 = \{0, +4\}$, and finally $C_2 = \{0, +8\}$ permuting $\{0\}$ and $\{8\}$.[12] These four instances of C_2 are indeed different groups, but since they are all isomorphic the same symbol is used: from the point of view of the algebraic structural analysis that group theory enables they are as good as identical.

The fact that the *same* group applies multiple times at any one level is shown by the subsets at each level (oval or rectangular tiles) appearing within the *same* wider rectangle. Further, the two kinds of lines the tiles are drawn in, solid and dotted, highlight that when the states are expressed in binary notation, as shown in Figure 7, an isomorphic copy of exactly the same group $C_2 = \{0, +1\}$ is operating not only within each level but also at different levels. In other words, a single binary automaton is sufficient to perform the action of each level, on any of the solid/dotted subset pairs. Within each long rectangle, the rectangular tile plays a role similar to a coset representative. In fact, each such tile contains the 0 state or identity, making it analogous to

[12] Each of these C_2's also permutes the other cosets at the same level of the diagram in the same way, by swapping pairs of them.

a subgroup. The ovals then become the remaining cosets. The analogy is particularly convincing here because the automaton states (shown) and the group elements acting on them (not shown) are indistinguishable (since elementary counters such as this are implemented with groups using their right regular representation).

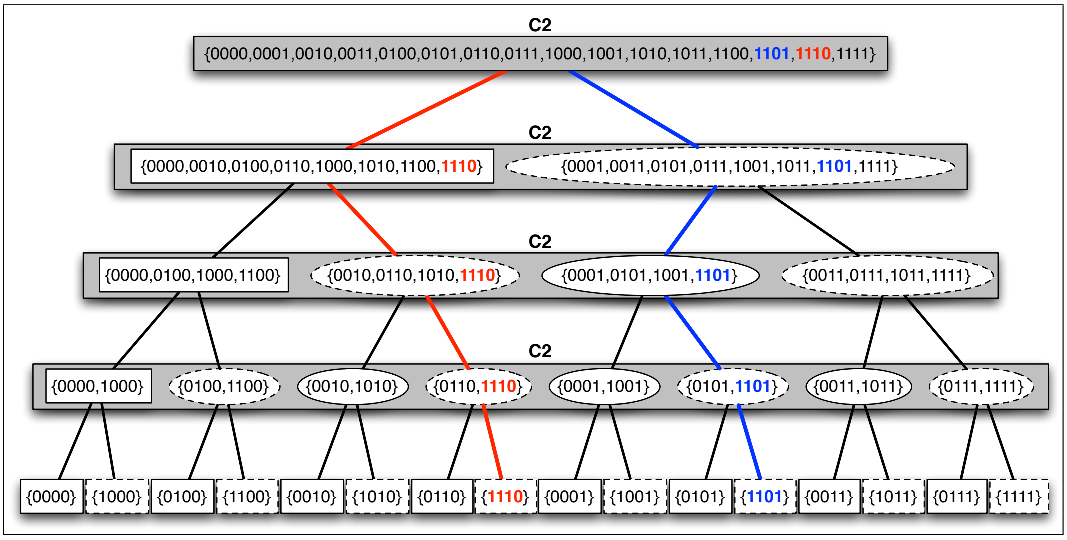


Figure 7: **Group coordinatization for non-negative integers** <16 **in binary notation**

3.4.2. *Example of Simple Automaton*

The previous automaton example has the simplifying property, once it is decomposed into 4 binary counters, that the state labels are the same as the input symbols: both can only take on values of 0 or 1. Furthermore, there is only one sub-automaton per level, all the levels have the same identical sub-automaton, and all the sub-automata have identical state labels. A holonomy decomposiiton for a more general automaton, although it will still provide a hierarchical coordinate system, will not usually have any of these nice properties. For example, Figure 8 shows a randomly generated automaton with 5 states $Q = \{1, 2, 3, 4, 5\}$ and an input alphabet $A = \{x, y\}$. Figure 9 shows the holonomy decomposition of this automaton on the right, obtained with `SgpDec` [16], and its decomposition into different sub-automata at different levels on the left.

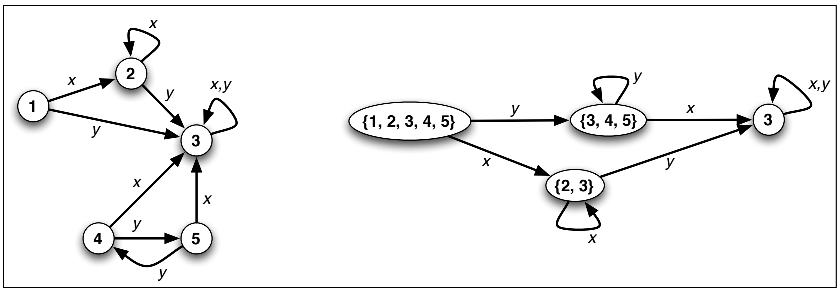


Figure 8: **Small, randomly generated automaton and its top-level state image sets**

First, the skeleton or tiling diagram depicting the holonomy decomposition (center of Figure 9) shows several differences from the previous example (Figure 6):

- Most of the rectangles are white rather than dark gray. This means that most transformations in the coordinate system are identity-resets, i.e. trivial or irreversible. For example, the map, under either x or y, from the original full set of 5 states to the subset $\{2, 3\}$ is not reversible. Once the system is in this subset, it can never recover States 1, 4, or 5.
- Moreover, State 1 is a “Garden of Eden” state: once the system leaves it, it can never come back. The dotted line from the top level to State 1 in the lowest level indicates that this state is not the image of $\{1, 2, 3, 4, 5\}$, under any

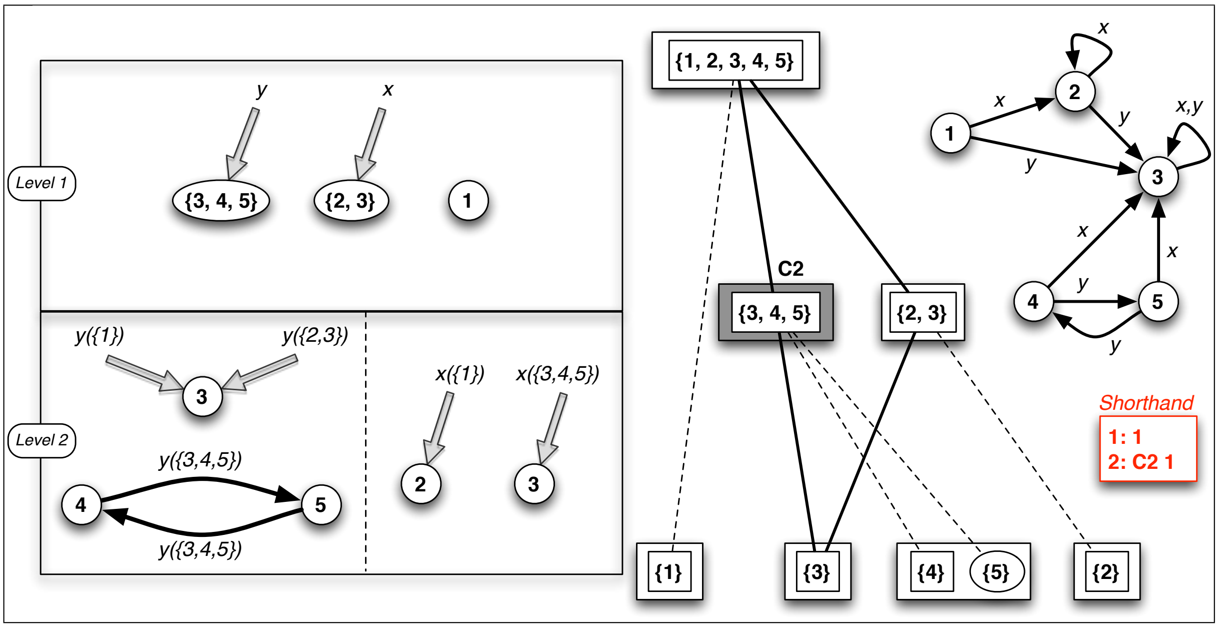


Figure 9: **Holonomy decomposition of the small automaton, with its sub-automata. The compact notation used on the left side of this figure highlights how action at each level of an HD can only be given by dependency functions whose values are either resets or permutations. Gray block arrows show resets within a given level. Black arrows indicate permutations. If arrows for possible dependency functions are not shown, then these have no effect (like the identity map), e.g. at level 2, $x(\{2,3\})$ is the identity permutation, and $y(\{3,4,5\})$ fixes all states but** 4 **and** 5**.**

transformation in the semigroup (since such a transformation would have to have 1 in its range). Within the subset $\{2,3\}$, State 2 plays a similar role, which is why it too is connected by a dotted line. Finally, neither $\{4\}$ or $\{5\}$ is the image of $\{3,4,5\}$.

- The subset $\{3,4,5\}$ is in a gray rectangle denoting that the automaton harbours a group acting on its sub-macrostates (in this case the singletons $\{3\}$, $\{4\}$, and $\{5\}$); in fact, this group is another C_2, even though the action is on 3, rather than 2 states. Thus, the action is reversible, meaning that there exists a symbol string (e.g. y) permuting the subsets under $\{3,4,5\}$ with order 2. It will take the system back to 4 (or 5) if it starts in 5 (or 4), while fixing 3. Note that in general a group (or semigroup) element need *not* correspond to a single input symbol, but normally a string of input symbols is needed to encode such elements; for example, yy realizes the identity permutation on $\{3,4,5\}$ in this C_2. Note also that such strings realizing transformations could be quite long.

Now, for the sub-automata, we notice the following differences from Figure 6:

- At each level the sub-automata are different.
- At level 2, we have *two* sub-automata in which the state of the second coordinate can lie.
- For each sub-automaton, the state labels are quite different.
- A block arrow in gray denotes a constant map taking the constant value pointed to. That is, it resets any state at the given level of the decomposition to the value pointed to. Black arrows denote permutations (which fix all other states).

But there are also two crucial similarities:

- When an input x or y arrives the system changes state, just as before. Thus, feeding an input symbol to an automaton can be seen as a generalization of adding two numbers.
- Whether and how a particular sub-automaton at a lower level transitions upon the arrival of an input symbol depends on the state(s) of the sub-automata above it in the diagram. This is shown with dependency functions similar to those of the 4-bit coordinatization of the C_{16} counter. Thus, the dependency functions in the wreath product decomposition of an automaton are a generalization of the carry rule in simple addition.

Finally, the red box labelled ‘Shorthand’ on the right of the figure is a very compact way to represent the HD skeleton for this example. It is shown here because for the larger systems to be discussed below the full skeleton or tiling diagram is too large (esp. too wide) and if shrunk to fit the page width would be unreadable. This very small automaton, therefore, provides us with the opportunity to show both versions, to help the interpretation of the larger examples expressed in shorthand notation in the next section. The 1s indicate either identity-reset components (whose transformations are just the identity mapping or constant maps), and the group C_2 is shown with its symbol. Note that the shorthand does not show what sets these groups and identity-reset semigroups act on, nor does it determine the cardinality of these sets.

Intriguingly, the decomposition of automata derived from metabolic and regulatory pathways often uncovers the presence of groups at different levels in the hierarchy. Since every group is associated with one or more symmetries and since biological systems exhibit – in fact, depend on – many symmetries in their structure and behaviour, it seems inescapable that the groups appearing in the holonomy decomposition of biological automata have something to do with their self-organizing properties. Indeed, as mentioned above, these groups are symptomatic of pools of reversibility in the dynamics of the biological system. Although in this paper we will not discuss any such biological systems, we now have enough tools at our disposal to present the algebraic analysis of the BZ reaction as a reference example that further work with cellular pathways will be able to build on.

4. Analysis of BZ Reaction

The BZ reaction has been analysed by many people over the last 50 years. In addition to the Field, Noyes, and Körös work already cited [40, 19] a useful reference is Scott [45], on whom we mostly rely. The BZ reaction is very complex, but can be described in broad outline as comprising three main processes, whose “resultant” reactions are:

$$\text{Process A:}\qquad BrO_3^- + 2Br^- + 3H \longrightarrow 2HOBr \tag{5}$$

$$\text{Process B:}\qquad BrO_3^- + HBrO_2 + 2Ce(III) + 3H^+ \longrightarrow 2HBrO_2 + 2Ce(IV) + H_2O \tag{6}$$

$$\text{Process C:}\qquad 2Ce(IV) + CH_2(COOH)_2 + BrCH(COOH)_2 \longrightarrow fBr^- + 2Ce(III) + \text{(other products)} \tag{7}$$

The three processes feed into each other in a cycle so that, as long as the input and output flows are held constant, they will repeat indefinitely. Process A reduces Bromide ion (Br^-), producing Bromous Acid ($HBrO_2$).[13] When the Bromide ion concentration falls below a certain threshold, the main reaction channel switches to Process B, which is very fast because it is autocatalytic (in $HBrO_2$) and produces Cerum IV ($Ce(IV)$). This causes Process C to start, which reduces Cerum IV to generate the Bromide ion again and restarts the cycle. As the system cycles through these three processes the colour of the solution changes cyclically, whereas if the reaction is taking place in a Petri[14] dish the familiar concentric colour waves of the BZ reaction can be observed.

4.1. *Oregonator Model*

Following [45], the Oregonator chemical equations are:

$$A + Y \rightarrow X + P \qquad rate = k_3AY \tag{8}$$

$$X + Y \rightarrow 2P \qquad rate = k_2XY \tag{9}$$

$$A + X \rightarrow 2X + 2Z \qquad rate = k_5AX \tag{10}$$

$$2X \rightarrow A + P \qquad rate = k_4X^2 \tag{11}$$

$$B + Z \rightarrow (1/2)fY \qquad rate = k_cBZ \tag{12}$$

X, Y, and Z correspond to the three compounds that undergo periodic oscillations under steady-state boundary conditions, meaning inside the CFST reactor. A, B, and P, by contrast, do not vary as a function of time, a consequence of the CFST reactor setup. The three colours Red, Green and Blue shown correspond to the three

[13] Reaction (5) is the resultant of two other reactions, not shown here, that produce $HBrO_2$. $HBrO_2$ is one of the intermediate products of the BZ reaction whose concentration is of interest here due to its oscillations. Reaction (5) only shows the production of hypobromous acid ($HOBr$), whose concentration is kept constant in a CFST reactor by a constant rate of extraction of this compound.

[14] A different Petri from the one who invented the Petri nets.

processes the BZ reaction is conceptually divided into.[15] Table 3 describes the variables in question, and Figure 10 shows the Petri net corresponding to the Oregonator model.

Oregonator Variable	Chemical Compound	Chemical Symbol	Association with BZ Reaction Process
X	Bromous Acid	$HBrO_2$	Process B (reduces X, generates Z)
Y	Bromide Ion	Br^-	Process A (reduces Y, generates X)
Z	Cerum 4	$Ce(IV)$	Process C (reduces Z, generates Y)
A	Bromate	BrO_3^-	All 3 processes
B	Malonic Acid	$CH_2(COOH)_2$	Process C
P	Hypobromous Acid	$HOBr$	Process A and C

Table 3: **Summary of Oregonator variables**

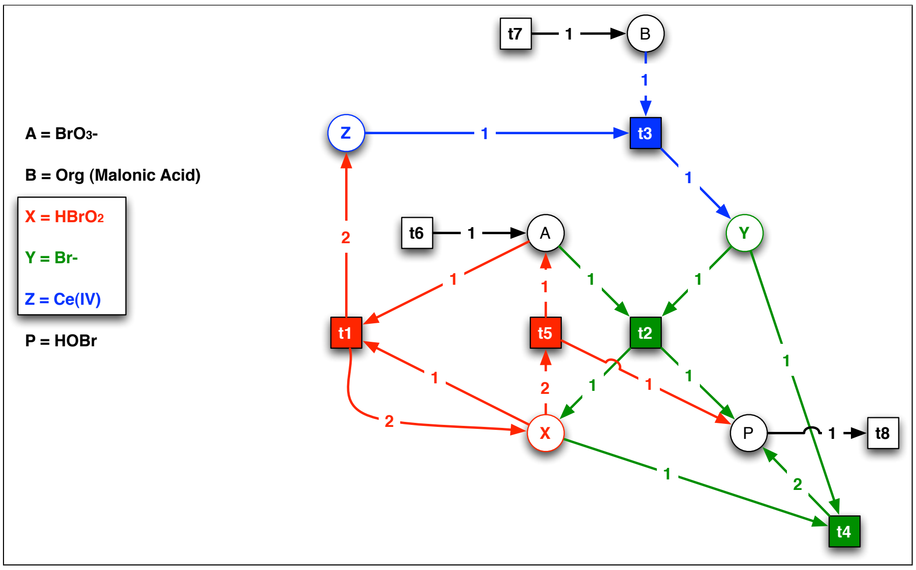


Figure 10: **Petri net corresponding to Eqs. 8-12**

From Eqs. 8-12 the rate equations are easily derived as a set of three ODEs [45]:

$$\frac{dX}{dt} = k_3AY - k_2XY + k_5AX - 2k_4X^2 \tag{13}$$

$$\frac{dY}{dt} = -k_3AY - k_2XY + 1/2fk_cBZ \tag{14}$$

$$\frac{dZ}{dt} = 2k_5AX - k_cBZ, \tag{15}$$

where the colours indicate the contribution to the rates of change of the three variables due to the three processes A, B, and C.

4.2. *Further Simplifications*

The size of the HD tends to increase exponentially with automaton size. Thus, although SgpDec has made it possible to analyse systems that are immensely greater than what could previously be done only by hand, we still need to keep the systems analyzsd as small as possible. Therefore, although the Oregonator model is already a significant simplification relative to the full set of chemical equations of the BZ system, we had to simplify it further in order to bring the corresponding automaton to a size amenable for algebraic analysis. In fact, since,

[15] Using the potentially confusing accepted notation, the names of the processes have nothing to do with the letters A, B, and P assigned to three of the compounds.

as we shall see below, the HD results can be very difficult to interpret even with very simple automata, it is important to start with the simplest possible system. In order to see behaviour that is close to a non-equilibrium dynamical system's we needed each place to have a capacity of at least 4 tokens, so we reduced the number of places to the smallest number possible, i.e. 3. The justification for eliminating the three compounds A, B, and P lies in the fact that for a CFST reactor they remain constant. Thus, including them greatly increases the size of the state space of the automaton derived from the PN in spite of their not contributing to the dynamics we are investigating. Furthermore, we eliminated also k_2 and k_4 in Eqs. 13-15. The motivation is that although they are important for reproducing the shape of the BZ oscillations, the oscillations themselves can be generated with a simpler system, which is preferable for now. Similarly, the factor of 2 in front of the first term on the right-hand side of Eq. 15, which comes from the factor of 2 in front of Z in Eq. 10, is also ignored in constructing the PN. This will need to be brought back in at a later stage because it is responsible for the characteristically fast growth of the Z compound (in Process A).

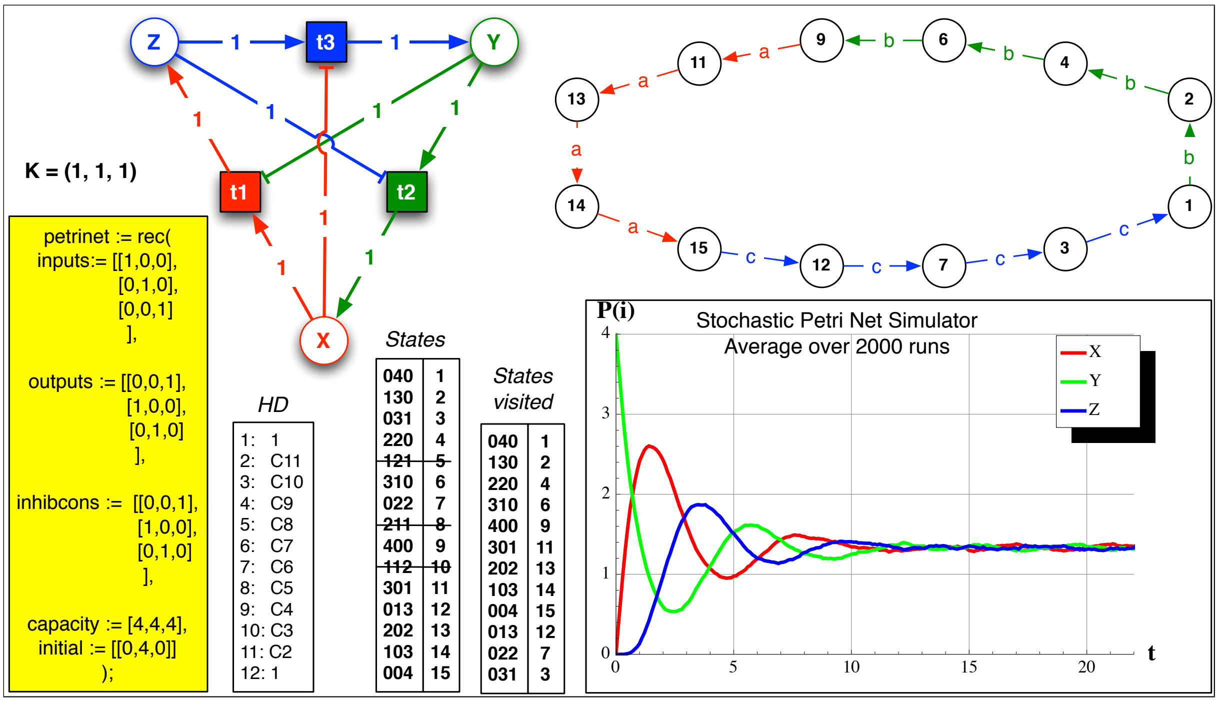
The problem with these modifications is that the resulting system is so different from the original BZ system that it may not even oscillate. This is remedied by artificially introducing inhibition, which is applied cyclically around the three active compounds. Unfortunately once this step is taken it becomes impossible to compare directly the resulting PN to the original ODE system, even if simplified. It is actually possible to extend the ODE model by including each instance of inhibition as an algebraic equation using a sigmoid function that approximates the step function of the PN. However, this is an artificial addition to the system in the sense that it models a desired effect rather than its physical cause, just like the inhibition arc in the PN. Whereas our derivation of the simplified PN is motivated by our desire to investigate the algebraic structure of an automaton corresponding to an oscillating system, obtained even through the artificial addition of inhibition arcs, in the case of the physical system we can afford to have higher aims because the rate equations embody a great deal of dynamical information at a more fundamental level. But, if an analytical study of the ODE system (e.g. through Lie groups) is to provide useful insights about its physical behaviour, the relationship between the oscillations and the original rate equations needs to be preserved. By radically simplifying the latter we have already lost a lot of valuable physical information that an artificial sigmoid function will not regain for us. In this sense, therefore, we feel there is less to be gained at this point from recovering some ODE system that matches in some formal way the radically simplifed PN system. As our ability to analyse larger PN systems improves, we will become better able to match the original ODE systems without much simplification. For now we are more focused on the more modest aim of understanding the discrete algebraic structure, and therefore of relating it to oscillatory behaviour obtained even artificially. From this point of view the radical simplification of the PN is actually a good thing since, as we will see below, even this extremely simplified system can still exhibit staggering algebraic complexity.

In summary, our primary interest is to relate the algebraic structure of oscillatory dynamical systems (which we understand fairly well) to the computational properties of the algebraic structures contained in the corresponding finite automata (which we do not yet understand). Therefore, using the BZ reaction as little more than initial inspiration to arrive at a much simpler oscillatory system should be seen as a necessary step in the analysis at this stage of the research. These simplifications are further justified by a fundamental and extremely useful property of stochastic systems demonstrated by Gillespie [21]: since the average place concentrations obtained with a stochastic PN simulator will converge to the ODE results as the number of runs approaches infinity,[16] we can still analyse the resulting system as if we *did* know the governing equations. The result of all these simplifications and modifications is an extremely simple and highly symmetrical PN, shown in Figure 11. A weight n of the inhibition arc between, for example, Place Y and Transition t_1 means that t_1 is inhibited if Y contains n or more tokens.

[16] It is relevant in this context to note that also Gillespie analysed the BZ reaction with a discrete stochastic model [22], reproducing the qualitative features of the physical system. However, in his case he used a number of molecules on the order of 8000. Although our stochastic Petri net simulator can handle systems of this size, the number of states associated with such a large capacity would be on the order of $8,000^3 = 512,000,000$. It is unlikely that the algebraic analysis of such a large automaton could ever be performed, thus demonstrating the value of seeking mathematical/theoretical insight.

4.3. Results

Figure 11 shows the GAP input file prepared for our PN package [15] and the 15-state automaton corresponding to the initial condition $(0, 4, 0)$. States 5, 8 and 10 are not reachable and are not shown. Rather, the figure also lists the states that are reachable and in the order in which the system visits them, which matches the automaton shown. An in-depth discussion of the mapping from a Petri net to its corresponding automaton is provided in [13]. At the bottom of Figure 11 the output of our stochastic PN simulator coded in *Mathematica* can be seen as a time series of the token values in the three places of this PN. The traces shown are the average of 2000 runs. Damped oscillations are clearly visible, as well as the fact that this PN conserves mass $(1.3 \cdot 3 \approx 4)$. The states of the automaton shown are the possible markings of the PN from the given initial condition. The rate constants for this example are all 1 $(K = (1, 1, 1))$. Finally, SgpDec was used to generate the HD, revealing 12 cascaded permutation-reset levels, whereby '1' indicates either an irreversible component or a trivial group, and the other groups are shown in standard notation. It is not very surprising that ins this extremely simple and symmetric example most of the levels of the decomposition are groups, and they are particularly simple groups (in the colloquial sense of the term), since they are all cyclic groups.



```
petrinet := rec(
inputs:= [[1,0,0],
          [0,1,0],
          [0,0,1]
          ],

outputs := [[0,0,1],
            [1,0,0],
            [0,1,0]
            ],

inhibcons := [[0,0,1],
              [1,0,0],
              [0,1,0]
              ],

capacity := [4,4,4],
initial := [[0,4,0]]
);
```

HD

1:	1
2:	C11
3:	C10
4:	C9
5:	C8
6:	C7
7:	C6
8:	C5
9:	C4
10:	C3
11:	C2
12:	1

States

040	1
130	2
031	3
220	4
~~121~~	~~5~~
310	6
022	7
~~211~~	~~8~~
400	9
~~112~~	~~10~~
301	11
013	12
202	13
103	14
004	15

States visited

040	1
130	2
220	4
310	6
400	9
301	11
202	13
103	14
004	15
013	12
022	7
031	3

Figure 11: **Stochastic simulation and HD analysis of very simple, symmetric BZ-like system**

The behaviour of the automaton is also oscillatory. This can be seen from Figure 12a, which is the frequency distribution over 2000 runs, and from Figure 12b, which shows a few representative traces. Whereas Figure 11 seems to say that the time evolution of the concentration levels (number of tokens) approaches the same constant for all three places, in fact this is only the average response. Figure 12 does not involve any averaging and shows that after some initial transient behaviour the system settles into a fairly stable pattern (i.e. independent of t) involving the 12 reachable states. The sampling time-step[17] is 0.1 seconds. The rise in frequency around States 6, 14, and 3 (ordinal 4th, 8th and 12th states from the initial marking) can be explained by the fact that the probability of firing increases when more tokens are present in the place feeding a given transition. Thus, as the tokens are used up the time-to-fire becomes longer, on average. This is not shown directly along the time axis, but at any given time the probability that the system is in one of these states is higher simply because it takes longer for it to transition out of them (see Figure 12b, esp. the slowest trace). Hence the frequency distribution shows higher values there. Figure 12b also shows the average of the 2000 state traces. The fact

[17] Since the stochastic simulation involves a decaying probability distribution of firing for each of the transitions, the times at which the system changes state are not evenly distributed. Thus, a uniform-time array is created after the stochastic run to sample the system state at even time intervals to simplify the tasks of calculating the average values and displaying the results.

that it approaches a constant indicates that the system settles into a periodic steady-state mode after the initial transient response. This can also be seen from the state traces, which show the system visiting the states shown in the automaton of Figure 11, in order, and cycling back. In other words, the system settles into a limit cycle whose average over many runs shows up as the limiting value of the damped oscillation of Figure 11.

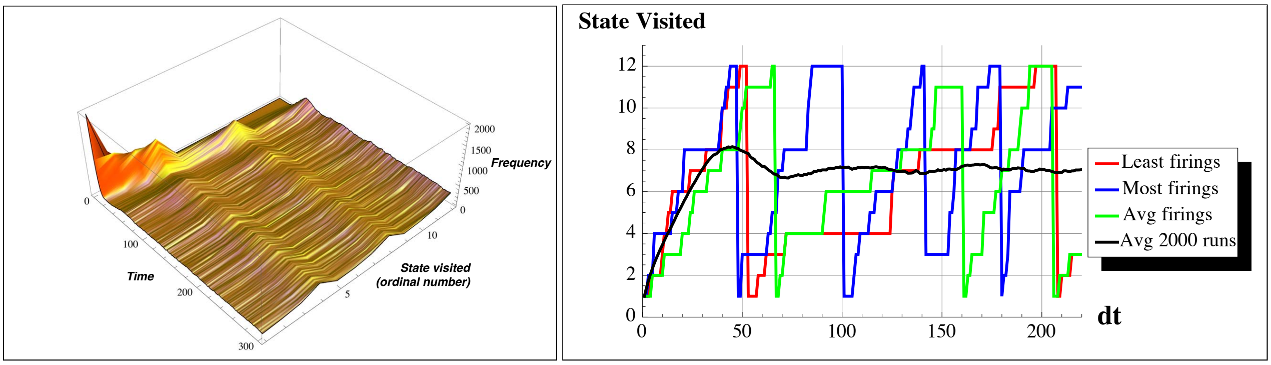


Figure 12: **Frequency distribution of the states visited, and representative state traces for the case of Figure 11**

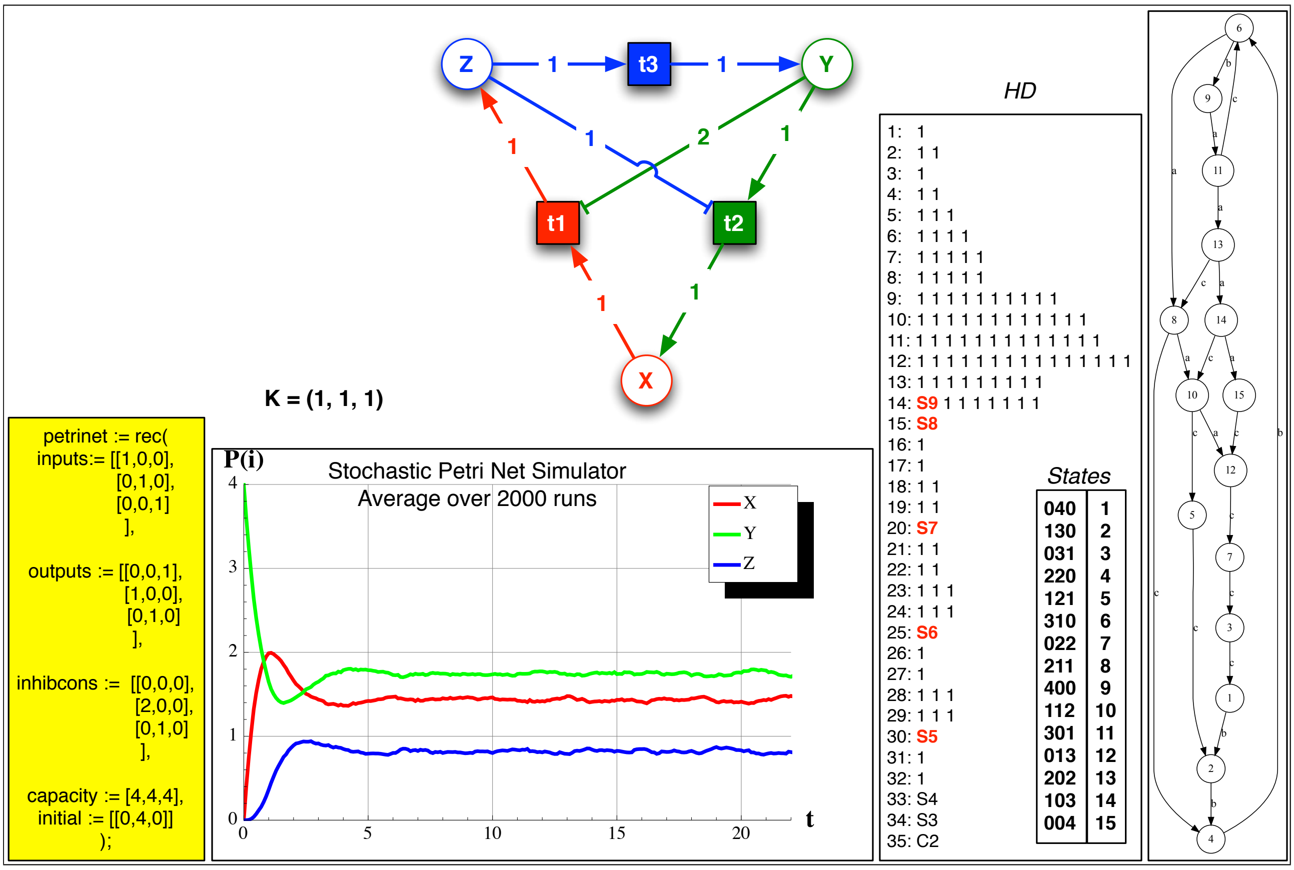
Although these considerations are beginning to shed some light into possible interpretations of the abstract concept of algebraic structure, we are still far from imagining how it could be applied. Fortunately, as discussed in Section 3.4, using the coordinatization point of view the different levels of the Krohn-Rhodes decomposition of an automaton become analogous to the different positions in our positional number systems. The decomposition then becomes an "expansion" of a given automaton into an "abstract number system" that is defined by the automaton itself: each state is expressed as a different multi-digit "number", where the position of each "digit" corresponds to a level in the decomposition. The significance of this insight is that the coordinatization perspective gives us at once powerful cognitive and calculational tools for manipulating an automaton in our mind or with possible software support, and also gives us the starting point of a general computer science methodology.

Figure 13 shows a rather different story. Here the PN is not symmetrical since one of the inhibitions is missing, and the other two have different values. The stochastic simulation shows strongly damped oscillations, and the automaton is more complex. The automaton has the same states as the previous example, and all of them are reachable. Since multiple paths can be followed through this automaton we have not attempted to reorder the states. To put things in perspective, the total number of possible transformations of 15 states is $15^{15} = 4.38 \times 10^{17}$ (approximately). The semigroup acting on the automaton of Figure 11 is a small subset, with 12,558,414 elements. For Figure 13, despite the fact that we could decompose the automaton, the number of elements of its semigroup could not be computed by `GAP` with 14GB of RAM.

The most interesting output is by far the HD. The decomposition shows 35 levels, with a fair number of groups. This case is remarkable because of the presence of very large groups (the symmetric group S_9 has $9! = 362,880$ elements acting on subsets of the states shown, at level 14). The groups shown in red all contain simple non-abelian groups (SNAGs): the alternating groups A_9, A_8, A_7, A_6, A_5, each a subgroup of the corresponding symmetric group S_n appearing in the HD (Figure 13).[18]

Finally, Figure 14 shows the frequency distribution and the state traces for the asymmetric case. The frequency distribution does not present surprises, it is qualitatively similar to the symmetric case but with emphasis on different states, as we might expect. The state traces, on the other hand, are different because they are not all monotonic within one cycle. This is because the automaton is more complex, with multiple paths possible, which implies that there is no single "natural" ordering of the states analogous to the symmetric case.

[18] In the holonomy decomposition, at any one level there can be many isomorphic copies of a given (permutation) group acting on different sets of subsets of states. The holonomy decomposition allows one to use just one such representative group from an equivalence class of these. In particular, for each group symbol in the HD of the two cases discussed here there are in fact many copies of that group in the semigroup and at the same level.



```
petrinet := rec(
  inputs:= [[1,0,0],
            [0,1,0],
            [0,0,1]
           ],

  outputs := [[0,0,1],
              [1,0,0],
              [0,1,0]
             ],

  inhibcons := [[0,0,0],
                [2,0,0],
                [0,1,0]
               ],

  capacity := [4,4,4],
  initial := [[0,4,0]]
);
```

```
1:  1
2:  1 1
3:  1
4:  1 1
5:  1 1 1
6:  1 1 1 1
7:  1 1 1 1 1
8:  1 1 1 1 1
9:  1 1 1 1 1 1 1 1 1 1
10: 1 1 1 1 1 1 1 1 1 1 1 1
11: 1 1 1 1 1 1 1 1 1 1 1 1 1
12: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
13: 1 1 1 1 1 1 1 1 1 1
14: S9 1 1 1 1 1 1 1
15: S8
16: 1
17: 1
18: 1 1
19: 1 1
20: S7
21: 1 1
22: 1 1
23: 1 1 1
24: 1 1 1
25: S6
26: 1
27: 1
28: 1 1 1
29: 1 1 1
30: S5
31: 1
32: 1
33: S4
34: S3
35: C2
```

040	1
130	2
031	3
220	4
121	5
310	6
022	7
211	8
400	9
112	10
301	11
013	12
202	13
103	14
004	15

Figure 13: **Analysis of asymmetric variant of our very simple BZ-like system**

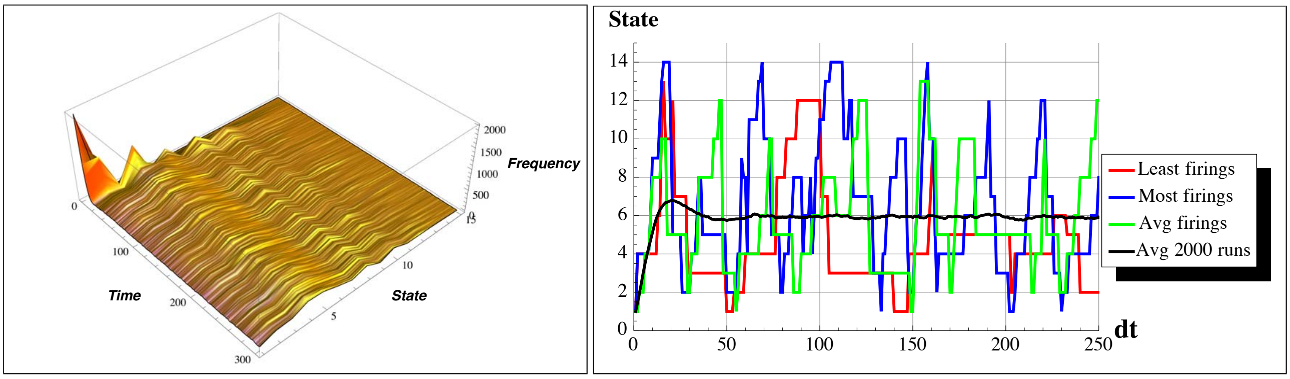


Figure 14: **Frequency distribution of the states visited for the case of Figure 13, and state traces**

5. Discussion and Outlook

The algebraic automata theory perspective is so general that one might struggle to imagine how its results could be narrowed down to the point where they can be harnessed even for relatively simple applications. For example, the purposefully idealized and simplistic symmetric case, which only has 12 states, is acted upon by more than 12 million possible transformations between these states! The fact that this is approximately a millionth of the total possible number of transformations between 12 states (8.9×10^{12}) does not improve things much.

However, the HD also suggests that the great majority of what the 12 million semigroup elements are doing can be described using group elements at multiple levels, and the groups are shown at the different levels of the decomposition. Therefore, there is a huge level of redundancy simply due to the highly symmetric nature of this system: there are many combinations of subsets of states that behave in exactly the same way. The fact that all the groups present are cyclic matches our intuition about the physical behaviour of this system, which can actually be played in one's mind by simply reading the state markings downwards, in the 'States Visited' table of Figure 11. For example, upon a t_2 transition the initial state (0,4,0) maps to (1,3,0). The fact that Place Y still has more than 1 token means that t_1 is inhibited, and since there are no tokens in Place Z the only thing that can happen is that Y fires again. And so it does, moving the system to the next state, (2,2,0). The next state is (3,1,0), and finally (4,0,0). At this point t_1 is not inhibited any longer, and Place X is "loaded", so t_1 begins firing. The pattern repeats as the tokens make their way around the PN to recover the initial state, at which point the whole cycle starts over. Morever, running these transitions *in reverse order* actually gives the cyclic generator of the corresponding group acting on the subsets of states in a manner that cyclically permutes them. (For details of how this works, see the editorial footnote by Nehaniv in [43, p. 160]).

The symmetric case is a bit contrived in the sense that it is so simple that it is difficult to imagine what kind of "algorithm" it might implement beyond the wave-like cyclic movement of the 4 tokens around the 3 places. So the fact that so many potential transformations between the states happen to satisfy the group axioms is not necessarily very useful or relevant in practical terms, especially since cyclic groups have a very simple structure. A similar statement could be made about the groups found in the asymmetric case, except that now even such a small modification apparently renders the system so much more complex that it is much more difficult to "see" what it is doing or what it can do physically. Thus, the presence of a wide variety of different groups containing simple non-abelian groups (SNAGs) is both intimidating – how many million combinations are we dealing with? – and reassuring, as the algebraic structure appears to be rather sophisticated and, therefore, potentially quite "expressive".

In fact, a very important property of the SNAGs is that they are exactly the functionally complete groups [31, 24] and are also considered to be related to error correction [43]. The former property of SNAGs makes them a natural candidate for realizing an analogue of "universal computation" within the finite realm [43, 31, 24, 8]. Therefore, we can conclude that the appearance of SNAGs indicates that even such a simple system is capable of functionally complete computation (i.e. like Boolean algebra). In particular, this implies that there are reversible subsystems of this system whose dynamics realize this (SNAG) computation by permuting certain "higher-level" (or macro-) states. Natural emerging research questions now are: (1) How, in detail, do oscillatory systems such as the simple BZ-like one analysed here realize this kind of (functionally complete) computation in terms of the dynamics of specific reversible subsystems? And (2) How might the finitary universal computational potential of these BZ subsystems be harnessed? The first problem can be addressed by more detailed study of generators and states of the permutation groups arising in the decompositions, to find insights from particular example oscillatory systems like those studied in this paper into the roles that SNAGs can play. It is likely that the solution to the second, more practical question will benefit from a more in-depth understanding of the analytical framework shown in Figure 2. This statement is based on our experience analysing and modelling non-linear dynamical systems: although often non-integrable, when some analytical solution – even to an approximate form of a given system – can be found, usually a great deal of insight is gained. Our research is aiming at extending the reach of current analytical methods by making a formal bridge between the symmetry properties of ODE systems, formalized through Lie groups, and the symmetry properties of their corresponding discrete automata, formalized through the Krohn-Rhodes or holonomy decomposition of their semigroups. In particular, we seek to understand the formal relationship between:

- Lie group structure of the dynamical systems modelling a given cellular pathway expressed as a system of ODEs; and

- Algebraic structure of the automata modelling the same cellular pathway, in particular the presence of the functionally complete SNAGs.

An introduction to some of the basic concepts underlying Lie groups is given in [25], and detailed discussions can be found in [41, 46, 26] and many others. In very simple terms, Lie groups are groups of continuous transformations acting on the dependent and independent variables of a given differential problem. For a single ODE, they can be visualized as smooth curves that cross a given solution curve of the ODE at some transversal angle and, since they leave the functional form of the ODE invariant, they map such a solution to a different solution of the same ODE. For this reason they are also called ‘symmetry curves’. It turns out that this property greatly facilitates the integration of the ODE. In fact, the reason Lie groups are effective, in some cases, at solving non-linear problems is that their continuous structure makes it possible to work with their infinitesimal generators, which break the original problem down into linear sub-problems. The infinitesimal generators can be visualized as vectors tangent to the symmetry curves. The tangent vector space in which these tangent vectors live is called the Lie algebra of the given Lie group.

As first discussed by Chevalley [4], the construction of certain groups of automorphisms of certain Lie algebras over finite fields rather than over the real or complex number fields yields SNAGs; and the modern classification of SNAGs asserts that ‘most’ of them arise in exactly this or closely related ways. Therefore, as shown in Figure 2, our next steps will involve a study of the relationship between a Lie group (and its corresponding Lie algebra) that helps solve the ODE problem of a particular cellular system and the corresponding finite groups, whose irreducible building blocks are SNAGs and prime-order cyclic groups, embedded in the transformation semigroup of the automaton derived from the discretization of the same biochemical system.

6. Conclusion

The main conclusion we can derive from the foregoing is that even though the precise algorithmic significance of the groups present in the decomposition of our model of the BZ reaction is not yet clear, the interpretation of the HD as an abstract positional number system provides a way to understand its “computation” in the most elementary sense of the term, i.e. as an abstract form of “counting”. In other words, the Krohn-Rhodes or holonomy decomposition has enabled us to imagine the execution traces of automata as meanderings of abstract “counters” through a complex terrain of reversible and irreversible families of transformations on their state sets. Further, since a Krohn-Rhodes decomposition for each automaton implicitly defines a different abstract number system expanded in a different generalized base that can vary at each level, a starting point for the formal investigation of Interaction Computing could be hypotheses expressed as compatibility conditions on the “number systems” defined by automata that are meant to interact or on the “bases” used at particular levels by automata that are meant to communicate.

From the point of view of systems biology quite a lot of work still needs to be done to develop an appropriate modelling formalism based on Interaction Computing with which to construct discrete models that reflect dynamical and/or self-organizing properties that are relevant to sub-cellular systems. The motivation and the hope is that if we are able to map non-linear dynamical properties from ODEs modelling biochemical systems to discrete primitives we will have a much more powerful “language” at our disposal with which to formalize, analyse, and model systems biology problems.

Acknowledgements

The work reported in this article was funded in part by a number of EU projects: DBE, contract number INFSO-FP6-507953; BIONETS, contract number INFSO-FP6-027748; OPAALS, contract number INFSO-FP6-034824; EINS, contract number INFSO-FP7-288021; and BIOMICS, contract number CNECT-ICT-318202. Their support is gratefully acknowledged.

References

[1] M A Arbib, editor. *Algebraic Theory of Machines, Languages, and Semigroups.* Academic Press, 1968.

[2] G Baumann. *Symmetry Analysis of Differential Equations with Mathematica.* Springer-Verlag, Berlin, 2000.

[3] P Cameron. *Introduction to Algebra (2nd Ed.).* Oxford University Press, Oxford, 2008.

[4] C Chevalley. Sur certains groupes simples. *The Tohoku Mathematical Journal*, 7:14–66, 1955. 2nd Series.

[5] J P Cusumano and F C Moon. Chaotic non-planar vibrations of the thin elastica parts I & II. *Journal of Sound and Vibration*, 179(2):185–208, 209–226, 1995.

[6] P Dini, G Briscoe, I Van Leeuwen, A J Munro, and S Lain. *D1.3: Biological Design Patterns of Autopoietic Behaviour in Digital Ecosystems*. OPAALS Deliverable, European Commission, 2009. Available from: http://files.opaals.eu/OPAALS/Year_3_Deliverables/WP01/.

[7] P Dini, C L Nehaniv, A Egri-Nagy, and M J Schilstra. Algebraic Analysis of the Computation in the Belousov-Zhabotinsky Reaction. In M Lones, S L Smith, S Teichmann, F Naef, J A Walker, and M A Trefzer, editors, *IPCAT2012: 9th International Conference on Information Processing in Cells and Tissues*, pages 216–224, Cambridge, UK, 30 March-2 April, 2012. Springer LNCS 7223 (Short paper).

[8] P Dini and D Schreckling. A Research Framework for Interaction Computing. In Fernando A B Colugnati, Lia C R Lopes, and Saulo F A Barretto, editors, *Digital Ecosystems: Proceedings of the 3rd International Conference, OPAALS 2010*, pages 224–244, Aracaju, Sergipe, Brazil, 22-23 March, 2010. Springer LNICST.

[9] P Dini, D Schreckling, and G Horváth. Algebraic and Categorical Framework for Interaction Computing and Symbiotic Security. In E Altman, P Dini, D Miorandi, and D Schreckling, editors, *Paradigms for Biologically-Inspired Autonomic Networks and Services: The BIONETS Project eBook*, 2010. Available from: http://www.bionets.eu.

[10] P Dömösi and C L Nehaniv. *Algebraic Theory of Automata Networks*. SIAM, Philadelphia, 2005.

[11] A Egri-Nagy, P Dini, C L Nehaniv, and M J Schilstra. Transformation Semigroups as Constructive Dynamical Spaces. In Fernando A B Colugnati, Lia C R Lopes, and Saulo F A Barretto, editors, *Digital Ecosystems: Proceedings of the 3rd International Conference, OPAALS 2010*, pages 245–265, Aracaju, Sergipe, Brazil, 22-23 March, 2010. Springer LNICST.

[12] A Egri-Nagy, J D Mitchell, and C L Nehaniv. Algorithms for the Efficient Calculation of the Holonomy Decomposition. In P Dömösi and S Iván, editors, *Automata and Formal Languages: Proc. 13th Intern. Conference AFL2011*, Debrecen, Hungary, 17-22 Aug, 2011. Institute of Mathematics and Informatics, College of Nyíregyháza.

[13] A Egri-Nagy and C L Nehaniv. Algebraic properties of automata associated to Petri nets and applications to computation in biological systems. *BioSystems*, 94(1-2):135–144, 2008.

[14] A Egri-Nagy and C L Nehaniv. Hierarchical coordinate systems for understanding complexity and its evolution with applications to genetic regulatory networks. *Artificial Life*, 14(3):299–312, 2008. (Special Issue on the Evolution of Complexity),.

[15] A Egri-Nagy and C L Nehaniv. PN2A: Petri Net Analysis GAP Package, 2010. Available from: http://sourceforge.net/projects/pn2a/.

[16] A Egri-Nagy, C L Nehaniv, and J D Mitchell. *SgpDec – software package for hierarchical coordinatization of groups and semigroups, implemented in the GAP computer algebra system, Version 0.6.67*, 2012. http://sgpdec.sf.net.

[17] A Egri-Nagy, C L Nehaniv, J L Rhodes, and M J Schilstra. Automatic analysis of computation in biochemical reactions. *BioSystems*, 94(1-2):126–134, 2008.

[18] S Eilenberg. *Automata, Languages, and Machines, Vol B*. Academic Press, New York, 1976.

[19] R J Field and R M Noyes. Oscillations in chemical systems IV. Limit cycle behavior in a model of a real chemical reaction. *Journal of Chemical Physics*, 60(5):1877–1884, 1974.

[20] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.5.6*, 2012. http://www.gap-system.org.

[21] D T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[22] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[23] W Holcombe. *Algebraic Automata Theory*. Cambridge University Press, 1982.

[24] G Horváth. *Functions and Polynomials over Finite Groups from the Computational Perspective*. The University of Hertfordshire, PhD Dissertation, 2008.

[25] G Horváth and P Dini. Lie Group Analysis of p53-mdm2 Pathway. In Fernando A B Colugnati, Lia C R Lopes, and Saulo F A Barretto, editors, *Digital Ecosystems: Proceedings of the 3rd International Conference, OPAALS 2010*, pages 285–304, Aracaju, Sergipe, Brazil, 22-23 March, 2010. Springer LNICST.

[26] P E Hydon. *Symmetry Methods for Differential Equations: A Beginner's Guide*. Cambridge University Press, 2000.

[27] R E Kalman, P L Falb, and M A Arbib. *Topics in Mathematical System Theory*. McGraw-Hill, New York, 1969.

[28] S Kauffman. *The Origins of Order: Self-Organisation and Selection in Evolution*. Oxford University Press, Oxford, 1993.

[29] K Krohn, R Langer, and J Rhodes. Algebraic Principles for the Analysis of a Biochemical System. *Journal of Computer and System Sciences*, 1:119–136, 1967.

[30] K Krohn and J Rhodes. Algebraic Theory of Machines. I. Prime Decomposition Theorem for Finite Semigroups and Machines. *Transactions of the American Mathematical Society*, 116:450–464, 1965.

[31] Kenneth Krohn, W. D. Maurer, and John Rhodes. Realizing complex boolean functions with simple groups. *Information and Control*, 9(2):190–195, 1966.

[32] G Lahav, N Rosenfeld, A Sigal, N Geva-Zatorsky, A J Levine, M B Elowitz, and U Alon. Generation of oscillations by the p53-mdm2 feedback loop: a theoretical and experimental study. *Natural Genetics*, 36:147–150, 2004.

[33] R Lev Bar-Or, R Maya, L A Segel, U Alon, A J Levine, and M Oren. Generation of oscillations by the p53-mdm2 feedback loop: a theoretical and experimental study. *Proceedings of the National Academy of Science USA*, 97(21):11250–11255, 2000.

[34] O Maler. On the Krohn-Rhodes Cascaded Decomposition Theorem. Web publication. Available from: `http://www-verimag.imag.fr/~maler/`.

[35] N A M Monk. Oscillatory expression of Hes1, p53, and NF-kappaB driven by transcriptional time delays. *Current Biology*, 13(16):1409–1413, 2003.

[36] C L Nehaniv. Algebraic models for understanding: coordinate systems and cognitive empowerment. In *Proceedings of the Second International Conference on Cognitive Technology, 1997: Humanizing the Information Age*, pages 147–162, Aizu-Wakamatsu City, Japan, 1997.

[37] C L Nehaniv and J L Rhodes. The Evolution and Understanding of Hierarchical Complexity in Biology from an Algebraic Perspective. *Artificial Life*, 6:45–67, 2000.

[38] G Nicolis and I Prigogine. *Self-Organization in Nonequilibrium Systems*. Wiley, New York, 1977.

[39] E Noether. Invariante Variationsprobleme. *Math-Phys Kl*, pages 235–257, 1918. English Translation: Transport Theory and Stat. Phys., 1:186-207, 1971.

[40] R M Noyes, R J Field, and E Körös. Oscillations in Chemical Systems I. Detailed Mechanism in a System Showing Temporal Oscillations. *Journal of the American Chemical Society*, 94(4):1394–1395, 1972.

[41] P Olver. *Applications of Lie Groups to Differential Equations*. Springer, 2nd edition, 1993.

[42] C A Petri. Kommunikation mit Automaten. *Schriften des IIM*, 2, 1962.

[43] J Rhodes. *Applications of Automata Theory and Algebra via the Mathematical Theory of Complexity to Biology, Physics, Psychology, Philosophy, and Games*. World Scientific Press, 2010. Foreword by M W Hirsch, edited by C L Nehaniv (Original version: University of California at Berkeley, Mathematics Library, 1969).

[44] M J Schilstra and S R Martin. Simple stochastic simulation. In L Michael and B Ludwig, editors, *Methods in Enzymology*, pages 381–409. Academic Press (Elsevier), 2009.

[45] S Scott. *Oscillations, Waves, and Chaos in Chemical Kinetics*. Oxford University Press, Oxford, 1994.

[46] H Stephani. *Differential equations: Their solution using symmetries*. Cambridge University Press, Cambridge, 1989.

[47] A Weinstein. Groupoids: unifying internal and external symmetry. *Notices of the American Mathematical Society*, 43:744–752, 1996.

[48] H P Zeiger. Cascade synthesis of finite-state machines. *Information and Control*, 10(4):419–433, 1967.