Spiky RBN:
A Sub-symbolic Artificial Chemistry

Michael Krastev

PhD
University of York
Computer Science
September 2018

Abstract

We design and build a sub-symbolic artificial chemistry based on random boolean networks (RBN). We show the expressive richness of the RBN in terms of system design and the behavioural range of the overall system. This is done by first generating reference sets of RBNs and then comparing their behaviour as we add mass conservation and energetics to the system. The comparison is facilitated by an activity measure based on information theory and reaction graphs but tailored for our system.

The system is used to reason about methods of designing complex systems and directing them towards specific tasks.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to express my deepest thanks to:

- Susan Stepney and Angelika Sebald for guiding me with enthusiasm for 3 years and dragging me through the writing of this document;
- Leo Caves for his insightful and productive questions;
- Monsterz and Sausage for moral support nondescript.

'From the simple comes the complex, and from the complex comes a different kind of simplicity. Chaos is order in a mask'

Terry Pratchett: Theft of Time

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. Parts of this work have been published [61, 60, 36]. All sources are acknowledged as References.

Chapter 1

Introduction

One of the most basic questions still open in the natural sciences is how life originated on the planet. At some point in the past the pre-biotic molecules found themselves in an environment which allowed them to develop mechanisms like self-replication and self-maintenance. There are many views on how this came about and what biologically relevant component was the first to emerge [40], but no definite answer.

From the computer science point of view the question is: what are the properties of real chemical systems that have allowed for these complex behaviours to emerge? Chemistry is combinatorically huge with self-organising behaviours, emergence of functionality, and adaptability to environments being present at almost every level of abstraction, from molecules to social networks.

The natural world has long provided a seemingly bottomless well of inspiration for computer scientists aiming to develop computational models. Observations of the organised behaviour of ants, termites and other swarm creatures solving computationally difficult tasks has made many people interested in how they do it. By understanding how they perform the tasks, like solving the travelling salesman problem, models are created for doing the same [27]. And these systems are a high-level emergent property of nature.

This work aims to explore what makes chemical systems so open-ended. More

specifically: can inspiration from real chemistry guide us in creating a complex open-ended model? And, what are the mechanisms this model should have?

Understanding of how atoms behave and interact, at least conceptually, is possible [91]. However our ability to accurately simulate the system over any large timescale is lacking.

Interest in chemical systems from a computer science point of view has a long history and is closely related to the questions posed by biology [4]. Artificial Life (ALife) and Artificial Chemistry (AChem) have produced many models for both computation and simulation [25]. These models are extremely high-level abstractions from the natural world [8]. They abstract out many of the mechanisms present in chemistry in order to exhibit specific behaviour. This makes sense in many cases where the aim is to produce abstract models for exploring specific real world problems [68].

Exploring exactly those mechanics that are generally left out is part of this work. Conservation of mass and energy, kinetics of reactions, and molecular structure are just some properties that are fundamental to our physical world. This work explores what effect they might have on an Artificial Chemistry model.

In order to do so, we require a model to which we can apply these concepts. Sub-Symbolic Artificial Chemistries (ssAChem) [33] are a good fit given our aim. Unlike most AChem systems in which reactions are specified by the developer, ssAChems use an underlying rich system to represent particles. Here we use the richness of a dynamical system. The ability of particles to react is then a function of the underlying dynamic of the reactants. This is a closer analogue to physical reality, while still being computationally tractable.

There are many concepts missing from the original RBN-World ssAChem, and this works creates a new ssAChem using RBN-World as a starting point. We introduce emergent bonding and kinetics to our system. More importantly we do this in a way that fits our understanding of how these mechanism exist in the physical world.

Creating a system with rich complex behaviours is only part of the challenge. In

order to be able to evaluate the system and set meaningful targets we require measures of complexity. There is a wealth of definitions of complexity and associated measures [86]. Some have been applied to chemistry [104]. Our aim is to build an indicative measure which will be based on information theory but tailored to our system.

This work begins with a literature review of the field in Chapter 2 followed by a statement of the research Hypothesis in Chapter 3. Chapter 4 introduces the Core model of our ssAChem which is then extended with environmental constraints in Chapter 5 and with energetics in Chapter 6. We then reason about complex systems in general and reflect on the one we have built in Chapter 7 by introducing an instance model of a complex system. Chapter 8 contains a conclusion and suggestions for future work.

Chapter 2

Literature Review

This chapter reviews a range of complex systems and emergent properties, how they have been abstracted and captured in computational systems, and how their complexity can be measured. It then focusses on natural chemistry as a complex system, and the related computational field of Artificial Chemistry.

2.1 Complex Systems

Exploration of complex systems and their behaviours is of interest to a wide range of disciplines. In the natural world there are many examples of complex systems that produce interesting emergent behaviours, from the emergence of life in chemical systems to the creation of complex structures like hives in swarm systems. Artificial system also exhibit emergent properties, for example, financial systems are emergent institutional systems comprised of many smaller systems and human elements. Methods of studying these are important to better understand and interact with them in a consistent manner.

The natural world has inspired many computational models which exploit emergent properties to perform computation or control behaviour.

2.1.1 Swarm Optimisation

One area of particular interest is swarm behaviour. Swarms of animals, be it birds, insects or fish, can perform a number of complex tasks without any centralised control mechanism. Moreover swarm systems are robust to failure of individuals and, since, each individual is only concerned with its immediate vicinity, the agents can be relatively simple.

2.1.1.1 Ant Colony Optimisation

Algorithms based on swarm behaviour have seen applications in optimisation problems. Ant Colony Optimisation (ACO), first explored by Dorigo [27], is based on the forging behaviour of ants and the stigmergic communication methods they use to achieve this. Stigmergy is a method of indirect communication used by a number of species. It relies on modifying the environment in order to provide a communication channel. Ants exhibit this characteristic by releasing pheromone. In ACO algorithms 'ants' traverse a landscape, a connected graph, picking their route probabilistically based on the amount of 'pheromone' present. This pheromone represents The quantity of pheromone on a path is subject to evaporation and addition. Evaporation removes a portion of the pheromone along every path. When ants walk over a path they add fresh pheromone, increasing the intensity along that path. Solutions are based on which paths have the most pheromone. There are many variants on ACO using different update rules for pheromones. One often used and cited example is Max Min Ant System, developed by Stutzle and Hoos [101, 99, 100]. This system updates only paths which give the best solution, either in the current iteration or overall. Rank Based Ant Systems [14, 66] allows only a few of the best solutions found in the current iterations to deposit pheromones.

Dorigo first showed his ACO algorithm being applied to the travelling salesman problem [26]; ACO algorithms have been used for many other optimisation problems since. Examples include the Set Covering Problem [64], Job Scheduling [17, 113],

and many others. Mohan et al [75] review ACOs being used in a variety of domains as well as some other algorithms to solve these problems.

2.1.1.2 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO), first introduced by Eberhart and Kennedy [30], is another approach to optimisation based on the behaviour of swarms. This time inspiration comes from bird flocking and fish schooling. Particles explore the parameter space of a problem, and have a velocity that moves them through the space. The velocity changes at each time step based on both personal knowledge (the location of the best solution found by this particle) and global knowledge (the location of the current global best). Like ACO, a large number of variations of PSO exist; depending on the class of problem and type of landscape many different variants have been developed. Some introduce new factors to the velocity update mechanism [95], others explore different initialisation techniques [89], and others still introduce the concept of mutation to the particles [108, 46]. Imran et al [51] give a short overview of a number of PSO variants that fit into these categories. Khare and Rangnekar [58] give a more comprehensive overview of different approaches, with a view on specific applications of PSOs.

2.1.1.3 Other approaches

Since the introduction of ACO and PSO there have been many other swarm optimisation algorithms inspired by different natural examples. Parpinelli and Lopes [83] overview some based on bees, bats, cockroaches and other natural swarm systems, as well as providing examples of these algorithms being used.

Many of the modifications to ACO and PSO tend to stem from a requirement to improve the algorithm's performance with regard to a specific problem domain or fitness landscapes, such as to increase speed of convergence or (more often) prevent early convergence at a local minimum. There are so many variation of ACO and

PSO, as well as many other swarm optimisation approaches that use different methods, that it is difficult to speak in general terms. There is much literature comparing variants with each other as well as other approaches such as genetic and evolutionary algorithms [53, 31, 106]. There is unlikely to be one best option overall: individual implementations have strengths and weaknesses both in terms of performance and efficiency.

2.1.1.4 Parameter setting and exploration

With this class of algorithms comes a requirement to find good parameter values. For a given problem finding such good parameter values for the optimisation algorithm is a computational optimisation problem in itself. This issue is further compounded by the stochastic nature of the algorithm. The solution is an emergent property of the system, not simply attributable to the behaviour of the particles, and so there is no deterministic way to set the parameters. In fact there is no solid deterministic way to predict how parameter sets will interact and behave overall. This issue is especially pronounced in swarm optimisers with a high number of parameters: Glowworm swarm optimisation [62], GSO, has 9 parameters that have to be set by the user.

There are a number of algorithms proposed for tuning parameters [76, 11]. Yuan [114] explore various methods of tuning applied to both ACO and PSO algorithms. Even so, the tuning problem exposes one difficulty when dealing with complex systems. Due to the stochastic nature of the system and the unclear relationships between local and global behaviour, predicting and reasoning about performance can be difficult. As such these systems are fragile: good parameter values for solving one problem are not necessary good values for solving another. In nature we can think of natural selection as being the tuning algorithm. Better parameters, for example a pheromone that is well suited to a particular environment, result in higher chance of survival. Of course nature has a very large time scale over which to explore the parameter search space.

2.1.2 Swarm robotics and engineering

Closely related to swarm intelligence is the field of swarm robotics. As discussed earlier, natural swarms have a variety of desirable properties. A swarm is capable of performing complex tasks such as optimisation and path finding. Its members can cooperate in order to create complex structures and to perform tasks not possible for single members. For example, ants join together to form a living bridge or to kill predators. This makes the swarm as a whole adaptable and flexible to changes in the environment. This is achieved without centralised control, making the system robust to loss of individuals and scalable. These properties are attractive to people exploring robotic systems. There is also the added benefit that the swarm as a whole can theoretically perform actions the individual robots cannot. This means the individual robots can be relatively simple, meaning cheap to produce and replace.

While the inspiration is similar to the swarm optimisation algorithms, swarm robotics pose quite a different set of challenges in terms of implementing a complex system. In swarm optimisation there is a direct link between the desirable emergent behaviour and the observable behaviour of the natural system. Ants perform foraging using pheromone trails, so it makes sense that an implementation of ant exploration based on pheromones would have the desired effects. With swarm robotics however the desired behaviours are much more specific, for example, self assembly into a bridge structure or organisation in order to perform a common task such as moving a large object. These are tasks that real swarms can perform, however the mechanisms that they use to perform them are not as well understood. Unlike the pheromone trail model, which has very clear mechanisms, we do not understand other behaviour to the same extent. This makes it much more difficult to replicate the behaviour, and specifically to engineer it.

This raises the question: how do we go from desired global behaviour to the behaviour of the individual robot? Since we are dealing with an emergent system the issue is not clear. There is no clear mechanism for determining emergent properties from individual behaviour outside of analysis of simulations.

The same observation is also problematic in the other direction. A system can exhibit an emergent behaviour that is undesirable (a bug), preventing the correct behaviour from emerging. Finding the cause of the bug and resolving it is difficult. Proving that the system will never exhibit undesirable behaviour is even more difficult. While the individual robots can be shown to work correctly, there is no guarantee that their collective behaviour is also correct. In software engineering there are mechanisms for ensuring correctness. Testing individual units of code against the specification and ensuring that behaviour is as expected is one method. This segmentation of the problem into easier to assess parts is useful. It is why concepts like test driven development are successful and popular. Of course, even in software this methodology is not perfect or foolproof. There are features that would be considered desirable that cannot be guaranteed per code block. Therefore software engineering requires testing of the whole system to ensure requirements such as safety and scalability are met.

The problems in developing specific emergent behaviour exist in all complex systems. Just like the issue of setting correct parameters in swarm optimisation algorithms, the problem stems from the difficulty of linking cause and effect: how individual behaviour influences global emergent properties.

In swarm robotics, exploration of how to design, develop and verify emergent behaviour is key. Kazadi [57] introduces the term "swarm engineering" to describe a two step process. The first step defines the problem to be tackled in a way that leads to conditions on the behaviour of the individual robots. The second step is producing behaviour for the robots such that they satisfy the conditions. This seems like quite a simplistic view of the problem of designing emergent behaviour. Kazadi himself states that there is no formal methodology for accomplishing the first step. There is also very little mention of testing and verification methodology in the general sense. The important thing however is that Kazadi recognises the need for a more formal methodology to developing swarm systems. Winfield et al [111] also use the term "swarm engineering", this time with a slightly better defined goal of using engineering practices to design, analyse and test swarm systems. This is done

with a view on dependability and use in safety critical scenarios. Winfield talks about possible approaches to design, analysis and testing of both individual agents in the swarm and, more importantly, the swarm as a whole. This is a new and open field of research. Winfield's paper is mostly suggestions of possibly applicable methodologies.

Another approach to design is to model the agents as probabilistic finite state machines that transition between behaviours. Because the behaviours of the individual agents at any given time depend on their observable environment, it is possible to construct a probabilistic model. This model determines what can be observed and therefore how the agents will act. This is quite a coarse grained approach, but useful to show that in general the system will exhibit desirable emergent behaviour. One issue is that it could ignore unlikely but important critical cases. The requirement to build a probabilistic model of a complex system is also a difficult one. Correctly identifying realistic probability distributions for the model can be extremely hard. Especially if the robot has to interact with a real world environment. The approach is used to develop behaviour based architectures for swarms. Soysal et al [98, 97] use it to develop aggregation behaviour. They recognise a number of different possible control strategies based on different finite state machines. Nouyan [81] follows a similar design principle to develop path formation behaviour. In these examples the probabilistic model is used to control the individual robots, however a probabilistic approach can also be used to describe and analyse the swarm as a whole. Lerman [63] gives an example of this using rate equations, using a method first suggested by Martinoli et al [70]. Lerman uses the method to derive the model of the swarm as a whole from the behaviours of the individuals.

Barchrach [1] has developed a more programmatic approach to designing swarm behaviour. Using the language Proto, which automatically relates global properties to local behaviour, Barchrach shows it is possible to program global behaviour of a swarm. This global behaviour translates to actual usable instructions for agents and the work shows robots controlled by it. The approach allows for behaviours to be composed in order to create even more complex behaviours. It is a top-down

approach, whereas the previous examples are bottom up, developing the code for the individual agents and then seeing the emergent behaviour.

There is a limitation to the design methodologies, specifically that robots and their sensors are imperfect and there is a level of noise generated by the agents of the system. This adds a low level of complexity to the system. It is feasible that a faulty sensor could push an agent into an incorrect state that could produce an undesirable high level emergent behaviour. This is compounded by the inherent difficulty of tracing a behaviour to its source.

Swarm robotics is a relatively new field and the requirement for more robust mechanism for development of emergent properties is still being actively explored. Much of the current work relies on specific development to fit the required behaviours, with more general mechanisms needing to emerge. Brambilla et al [13] review swarm robotics from the swarm engineering perspective. They cover design, analysis and testing methodology examples. One issue that emerges from the review is the lack of general understanding of how microscopic, agent level, properties translate into higher emergent properties. Another issue is comparability of different systems. Brambilla et al state "Despite the great number of analysis methods, performing verification and validation of a swarm robotics system and comparing one system with another are still very difficult tasks. The reason behind this is the lack of well defined metrics and testbed applications.". This statement is telling of the lack of understanding of how a complex system is actually defined. Exploring this problem is key to understanding the properties of complex systems and may lead to methods for analysing and developing them.

2.1.3 Cellular Automata

Cellular Automata (CA) are another class of complex systems that have often been explored in the context of classifying complexity. Most generally a CA can be defined as consisting of a number of cells arranged in a lattice. Each cell has a state (often binary) and at time t the state is a function of the states of neighbouring

cells at time $t-1$. This is a very simple model of a complex system, yet it shares a key property with other complex systems like swarms. Each cell performs a simple function as a result of communication with a subset of all cells (here the cells in its neighbourhood).

One of the first explorations of a CA was performed by von Neumann, published after his death by Burks [107]. Von Neumann was interested in creating self replicating automata. He proposed a cellular automaton where each cell can be in one of 29 states. From a defined initial configuration of cell states this CA is capable of producing a replication of the initial pattern. In fact the automata is a universal constructor: based on the pattern in the "tape" portion of the initial configuration, the automaton is capable of creating any other automaton. This CA has also been shown to be capable of universal computation. Later Codd then Banks [2] simplified the automaton and showed the same behaviour with an 8 state CA, and then a 4 state CA, respectively. Other authors have developed CAs based on different mechanisms that have also shown universal computation. This shows the power of CAs.

There are examples of CA being used for a variety of computational tasks, such as pattern recognition [112], and modelling of real world phenomenon from galaxies [90] to traffic [32]. In most of these, an initial pattern is set that corresponds to the problem being solved. The CA is allowed to run until it reaches a state considered final. At this point the pattern is read and interpreted as the result. This is a somewhat similar approach to the swarm optimisation above. The system is given an initial configuration and allowed to run through a number of dynamic states. Then the resulting pattern is observed providing the solution.

This focus on only the final state under-uses the potential power of complex systems. The dynamics of the system are more important than the final state. A system that is in a dynamic equilibrium is more powerful because it reacts to new stimuli. If we can define a system in such a way that the dynamic stable state is desirable then perturbation results in a set of transitional states that get us back to a desirable stable one. With swarm robotics this is more evident. We want the system to perform a function (the transitional states) that will get us to a stable state. But

we also want the system to be able to react to a new input after it has reached the stable state, in effect pushing it back into transitional states. Using CAs for modelling dynamic systems like the traffic example stated above seems a better use of their inherent properties.

2.1.3.1 Game of Life

One of the most recognisable examples of a CA is the Game of Life (GoL) [38], developed by John Conway. It is a two dimensional CA with binary states (alive and dead), and eight neighbours. The transition rules are:

1. A live cell with less then 2 live neighbours transitions to the dead state
2. A live cell with more then 3 live neighbours transitions to the dead state
3. A live cell with 2 or 3 live neighbours stays alive
4. A dead cell with 3 live neighbours transitions to the live state

In GoL the Moore neighbourhood is used, so there are 9 total cells involved in a state transition. The rules above are one of a possible 2^9 rules.

GoL exhibits a number of interesting properties. Firstly the relatively simple rule set produces complex emergent patterns. This is not necessarily surprising: there are CA implementations on 1D "strips" of cells with significantly fewer rules that also exhibit complex patterns and are capable of computation.

What is more interesting is the capacity of the system to support complex patterns that have emergent behaviour. It is quite easy to find simple stable patterns, arrangements that if left undisturbed will not change their behaviour. These patterns can be thought of as "frozen". A simple example is the "block", 4 live cells arranged in a square surrounded by dead cells; this arrangement is stable and will not transition to different states. The pattern can also be oscillating between a number of states. Other patterns, referred to as spaceships, can be thought of as moving.

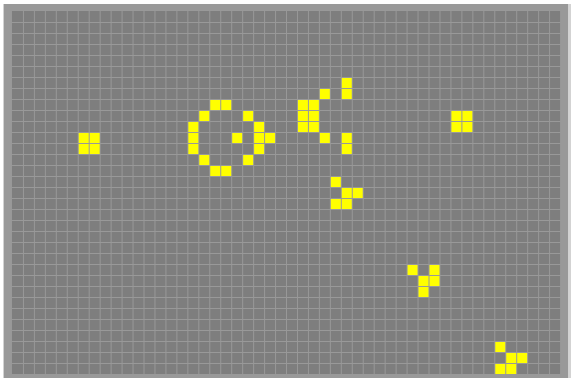


Figure 2.1: The Gosper Gun producing gliders. Note the block on either side of the gun internals

The initial pattern propagates live cells in one direction while destroying previous cells, so the pattern can be observed as moving across the lattice. These are all organisations that can be observed to emerge from a random initial state. These organisations can also be stably generated by other patterns.

The "glider" is one of the simplest spaceship patterns. It is also possible to construct a stable oscillating pattern, a "glider gun", that generates gliders at a constant rate. This can be thought of as a higher level of emergence. One of the simplest guns in GoL is the Gosper Gun, shown in figure 2.1. It consists of two stable non-oscillating blocks and a specific pattern between them. We have the stable block as a pattern which emergently persists; now in the context of the gun that pattern also performs a function, of redirecting the repeating internal pattern that generate the gliders. The specific configuration of the block pattern makes it suitable for the job (figure 2.2). Other "frozen" patterns would be destroyed as a result of the interaction with the other gun components. So we can talk about the block as having properties of its own (in this case stability), and we can also talk about the block having properties in terms of its interactions with other patterns.

The gun pattern as a whole also has properties other then the generation of gliders. For example, the Gosper gun can interact with an incoming glider, consuming it and preventing the generation of its own glider. This is effectively a NOT operation on the stream of gliders.

This is an example of emergent properties giving rise to new emergent behaviour

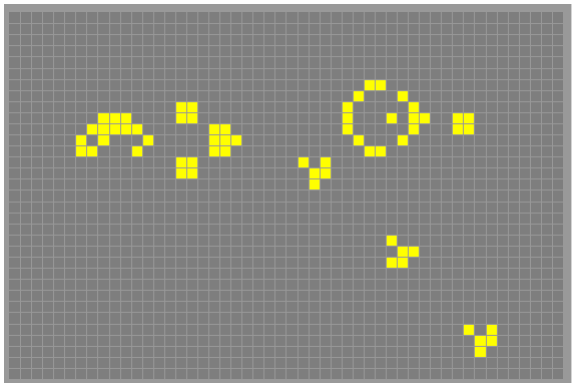


Figure 2.2: A different stage in the Gosper Gun pattern. The "frozen" block pattern on the left is currently in a transitional state as it interacts with the left part of the gun internal pattern.

and a growth in complexity. With GoL and CAs in general this type of constructive behaviour is very difficult to generate spontaneously. While there are many examples of CAs showing complex patterns and behaviours, finding patterns that produce interesting behaviour is difficult. Most initial configuration never produce this type of higher level interactivity. The system behaviour is very dependant on the right, carefully engineered, initial configuration. The same is true from the other direction. These interesting patterns are generally quite fragile. Introducing a random state change, for example by manually switching cell states often results in the destruction of the entire stable pattern.

What must be true of the system in order for complex structures to emerge? In the case of GoL, for example, what must be changed to allow for stable patterns to become more prominent and more robust to perturbation? One possible issue is that CAs and swarms have no constructive mechanism for increasing complexity. Methods like genetic algorithms can be used to provide an external constructive system, assuming a fitness function can be developed that rewards complexity. There are examples of genetic algorithms being used to develop CAs for specific computational tasks. Mitchell et al [74, 73] show a genetic algorithm developing a CA in order to compute density classification and synchronisation. However this is an easier task than the more general question of developing a system with complex emergent structures. The natural world is rather good at this, natural systems are not only complex but have built-in mechanisms that seem to grow complexity.

2.2 Definition and Measurements of Complexity

In order to develop methods for creating and evaluating complex systems there is a requirement for a measure of complexity. Provided such a measure exists we might be able to use methods like a genetic algorithm in order to create a new complex system by optimising this complexity measure. Such a measure would also allow us to compare complex systems as well as reason about their behaviour over time. For example, is there a "growth in complexity"?

Developing such a metric is a difficult task since it requires a solid definition of what complexity is. Although people tend to be quite good at distinguishing between a simple, complex and random system, quantifying how they came to the conclusion is difficult. There are some things that can be said about the behaviour of a complexity measure. Lopex-Ruiz et al [67] discuss that intuitively a complexity measure should tend to zero for ordered systems and for random systems. This is illustrated by talking about a particle system and its ideal states. When in a crystalline state the particles are arranged in a particular repeating structure and are perfectly ordered. On the other hand maximum chaos is achieved in an ideal gas state where there is equal probability of the particles being in any arrangement.

2.2.1 Algorithmic complexity

One approach to measuring complexity is to measure how difficult it is to describe the system. Intuitively, a simple system is easy to describe while a complex system is harder. This is the concept behind algorithmic complexity [59, 15], also referred to as Kolmogorov, Solomonoff, or Chaitin complexity. The algorithmic complexity $K(s)$ of a string s is defined as the length of the shortest input program I to a Turing machine that will produce s. For well structured strings the minimum program is short and for longer more complex string it will grow. For random strings however I will be maximal, so $K(s)$ is maximised for random strings.

There are other definitions of complexity that can be derived from this minimum program concept. Bennett's "logical depth" [7] is a different measure of complexity again relying on the construction of a minimum program. Bennett argues that complexity depends on the amount of work required to reproduce s, in other words the time it takes the shortest program I to produce s. This complexity measure is not maximal for random strings but instead maximal for "deep" objects: strings that require a long computational time to produce.

There is serious issue with both these measures: they are uncomputable. While these measures are theoretically interesting from the perspective of describing systems, they are of limited practical application, especially for large systems like chemistry. Furthermore, these measures define complexity for a single string. Our interest lies in the dynamics of the system, whether complexity grows in a system inspired by chemistry.

2.2.2 Crutchfield's epsilon-machines

The idea of defining complexity in terms of the length of system description is attractive, since it makes intuitive sense. A different approach to constructing the description is given by Crutchfield in his ϵ-machine framework [18]. In this framework the complexity measure is statistical complexity C_μ, defined as the size of the reconstructed ϵ-machine. This ϵ-machine is defined as the *"minimal model at the least computationally powerful level yielding a finite description"*. It is constructed iteratively by fitting it to the observed data until it accurately predicts new data. Ss the accuracy of the ϵ-machine tends towards 1, the size of the machine could tend to infinity. To deal with this the framework has a hierarchy of models. As an ϵ-machine reaches the limit of computational resource a new ϵ-machine is created at the next model level by factoring out regularities in the current level machine. A model hierarchy is suggested in the paper consisting of:

0. Data Stream

1. Tree

2. Finite Automata

3. String Production machine

If for example the Tree ϵ-machine reaches the limit of the available computational resource while still not successfully predicting new measurements then a finite automata ϵ-machine is constructed by grouping tree nodes. This new machine is then iteratively improved. Eventually at some model level an ϵ-machine is constructed which is finite and predicts accurately. So this method includes a concept of different expressive power being required to adequately describe different systems. Crutchfield shows the difficulty the framework has when dealing with a probabilistic system. The deterministic finite automata grows in size indefinitely since it cannot accurately describe the system. Shalizi et al [93] use a stochastic automaton as a model to construct ϵ-machines for Markov processes. This suggests that selection of the model for the ϵ-machine is a key part in generating useful complexity data, and that there is some requirement for prior knowledge of the observable complex system.

C_μ has the desired property of tending to 0 for both ordered and chaotic/random systems. C_μ tends to be maximal around the transition between an ordered and chaotic state. It is computationally intensive to calculate, however still a plausible computational metric. Another property of this approach is how C_μ varies with scale around the critical region. When the ϵ-machine is observing a dynamic system it consumes a string of measurements of length L. This can be thought of as the granularity at which the ϵ-machine observes the system's history. What Crutchfield [18] notes is that as L increases the ϵ-machine and therefore C_μ grows and is unbounded at this transition region. This suggests that a "coarser" view of the system requires a higher level model. The later work by Shalizi[93] actually uses a variable length L parameter between 0 and L_{max} in order to construct the stochastic automata. He also observes that increasing L_{max} causes an increase in size of the ϵ-machine. Shalizi's work also involves a finite set of measurements and he points out that a correct

value for L_{max} is dependent on the number of time series measurements available.

The framework's ability to extract hidden Markov models from data has been used for a variety of purposes: as a complexity measure to explore chaotic systems [43], as an anomaly detector for an electronic circuit [88], and for generating network protocol models which can be used to generate dummy data [110]. Because the ϵ-machine is conceptually understandable the statistical complexity can be used to talk about other features of a complex system. For example, Shalizi et al [92] define self organisation as self-driven increase in statistical complexity.

2.2.3 Entropy based measures

2.2.3.1 Shannon entropy

One of the key aspects almost always discussed in the literature on complexity measures is the concept of Shannon Entropy $H(X)$ [94]. This entropy measure calculates the number of bits needed in order to encode X, often referred to as the amount of information in the system.

$$H(X) = -K \sum_{x \in X} p(x) \log_2 p(x) \tag{2.1}$$

where x is a state of the system X; $p(x)$ is the probability of the system being in that state; K is a constant that defines the units of the entropy measure and is generally set to 1.

The Shannon Entropy has a number of properties. If one state x_i is a certainty, so $p(x_i) = 1$, then H is minimal: $H = 0$. If the probability distribution of all possible states is uniform, then $H = \log_2 N$ where N is the number of possible states, and this is the maximum value of H. H is additive for independent events: for two *independent* systems X and Y the entropy of the combination of the two systems is the sum of the individual entropies, $H(X, Y) = H(X) + H(Y)$. If X is fully determined by Y, then $H(X, Y) = H(Y)$.

We can directly calculate the joint entropy from the joint probabilities:

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 p(x,y) \tag{2.2}$$

We can calculate the conditional entropy of one system given another:

$$H(X|Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 \frac{p(x)}{p(x,y)} = H(X,Y) - H(X) \tag{2.3}$$

We can calculate how much information is shared between two systems. This is referred to as *mutual information*, which can be defined in terms of probability, joint entropy or conditional entropy:

$$\begin{aligned} I(X;Y) &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 \frac{p(x)p(y)}{p(x,y)} \\ &= H(X) + H(Y) - H(X,Y) = H(Y) - H(Y|X) \end{aligned} \tag{2.4}$$

There are conceptual similarities between H and the model constructing methods discussed above. In both cases there is an attempt to describe the system either by constructing a machine or defining the amount of information in it. Shannon entropy, like algorithmic complexity, is maximal for random systems. Since a lack of pattern equates to more information required to fully describe the system a random pattern requires a maximum amount of information. However, H is significantly simpler to calculate, provided the system can be described as a set of states each with a probability of being observed. This is by no means a trivial requirement. A system has to be partitioned and accurate probabilities have to be found in order for the measure to make sense. So the measure can be said to be subjective to some extent since the partitioning of the system (especially continuous multi-particle systems) into states depends on the user, and calculating meaningful probabilities can be difficult.

Shannon entropy has been widely used to reason about complex systems since it is

calculable. Interpretation of the measurements, in terms of information content and transfer, is intuitive. Prokopenko [86] gives primer on the use of information theory and Shannon entropy in computing complexity, self-organisation and emergence.

2.2.3.2 Grassberger's measures

Conceptual definitions of complexity can be expressed in terms of the Shannon entropy approach. In this way Grassberger[44] proposes three complexity measures: Set Complexity (SC), True Measure Complexity (TMC) and Effective Measure Complexity (EMC). In the work the measurements being observed are a sequence of valid characters. The system being explored is a grammar defining valid and invalid character sequences as well as a probability distribution defining the likelihood of observing a valid string.

Set Complexity is defined as the smallest average amount of information (Shannon entropy) required about past strings in order to verify the correctness of future strings. That is, how much information do we need about the system in order to identify incorrect strings. SC is constructed by creating a finite directed graph from the observed string. The graph has nodes and labelled edges, the labels being members of the alphabet of the observed string. As a measurement is read we move to the next node by travelling along an edge from the current node labelled by the read character. SC is defined as:

$$SC = -\sum_{nodes} p(i) \log_2 p(i) \tag{2.5}$$

where $p(i)$ is the frequency of visitations to node i in the graph so far.

There is a difficulty in generating the directed graph which to traverse. This approach is similar to algorithmic complexity in that we cannot create the model based on observation. Even if the grammar of the underlying system is known and algorithmic complexity can be determined (a minimal model created), this does not mean that SC can be determined: a larger connected graph (with more nodes) that

describes the system can lead to a smaller value of SC by having fewer nodes being visited frequently.

Grassberger's TMC is the minimum amount of information required about previous states in order to predict the probability distribution of the next state. That is, given an observed sequence $\{s_i, s_{i-1}, ...\}$, how much information do we need to predict the conditional probability of the next state $p(s_{i+1}|s_i, s_{i-1}, ...)$? This measure is more powerful than SC, which can be thought of as the amount of information needed to predict which characters have a probability of occurring equal to 0. TMC is no less difficult to calculate then SC.

Grassberger's EMC is the value of past information in predicting future states.

$$EMC = \sum_{N=0}^{\infty}(h_N - h) \tag{2.6}$$

where $h_N$is the Nth order block entropy, which is the amount of additional information needed to predict the next state given that we have N previous observations. This is calculated as follows. The amount of entropy present in a sub-string of previous observations of length N, $S = \{s_i, ...s_{i+N-1}\}$, is given by:

$$H_N = -\sum_{S} p_N\{S\} log_2 p_N\{S\} \tag{2.7}$$

h_N can then be calculated as the difference between the entropy of strings $\{s_i, ...s_{i+N-1}\}$ and $\{s_i, ...s_{i+N-1}, s_{i+N}\}$:

$$h_N = H_{N+1} - H_N \tag{2.8}$$

The h term is the limit of the block entropy as we reach an infinite string:

$$h = \lim_{N\to\infty} h_N \tag{2.9}$$

EMC is calculable since there is no requirements for knowing the underlying grammar of the system. Grassberger [44] calculates EMC for cellular automata. He

suggests that a slow approach of block entropy with increasing N towards the limit h is indicative of complex behaviour, as opposed to a linear approach. That is, each new piece of information does not contribute as much to the overall description of the system.

2.2.3.3 Predictive information

This interpretation of complexity in terms of descriptive or predictive power is discussed by Bialek [10]. In an ordered system each new observation does not tell us anything new about the system, so increasing the length of observation does not increase the entropy of the observation. That is, $h_N = 0$ because no new information is needed to predict the next state. In a chaotic system the past has no influence on the future: each new observation increases the overall entropy by the same amount since no previous information is useful. Complex systems are then in between these two behaviours; each new observation does provide new information, however the information observed so far also contributes. This means that as more of the system is observed new measurements provide less and less useful information.

Bialek talks about this in terms of predictive information I_{pred} and argues that complexity is related to how much information is useful for prediction of future events. I_{pred} can be described as the mutual information between the past and future of the system.

$$\begin{aligned} I_{pred}(T, T') = &-\langle \log_2 p(x_{\text{future}}) \rangle - \langle \log_2 p(x_{\text{past}}) \rangle \\ &- [-\langle \log_2 p(x_{\text{future}}, x_{\text{past}}) \rangle] \end{aligned} \tag{2.10}$$

where $\langle ... \rangle$ denotes averaging over the distribution p; T is the length of observation in the past; T' is the length of observation in the future; x_{future} and x_{past} are the observations of the past and predictions for the future over the time periods T and T' respectively.

Bialek suggests that the behaviour of I_{pred} with respect to increasing T can also

help with defining complexity. He defines three distinct possible behaviours and related systems that could produce them. If I_{pred} is constant it suggests that there is only a finite amount of information about the past needed to predict the future. For example ordered periodic systems will have a constant I_{pred} as T grows. In this case I_{pred} scales with respect to the size of the period. Longer periods mean more information must be stored before the system can be fully described. For complex systems I_{pred} will diverge logarithmically as T grows. The coefficient of this divergence can be thought of as estimating the number of rules in the underlying system model. This means comparison between two complex systems can be done by comparing the coefficients of their divergence. A larger coefficient suggests more rules and therefore a more complex system. I_{pred} can even grow at a sub linear power law rate. Bialek suggests that continuous systems can exhibit this type of behaviour. The conceptual understandability of this measure (the amount of information about the past that has to be stored to predict the future) means interpretation of the measure can be easier.

Predictive information has been discussed by many authors under different names. Crutchfield and Feldman [19] give a number of sources that use the same concept with various names. They themselves call it excess entropy. They interpret the quantity in terms of a number of different concepts, and provide a number of examples interpreting the measurements.

2.2.3.4 Conclusion

The malleability of Shannon entropy, combined with the reasoning power associated with the concept that the entropy exposes (information in the system) makes it a good choice for exploring complexity. This is further helped by the computability of the measure.

2.2.4 Chemical Complexity in terms of Graphs

From the perspective of our work, interest in defining a measure of complexity comes from our requirement to analyse the system we develop.

We can explore complexity of a chemical system in multiple ways depending on the scale of the model. We could explore the complexity of molecular structure, how complex are the structures that emerge in chemistry. We could explore the complexity of the interactions in the system specifically the possible reactions that can occur. In either case we can represent our system as a graph: defining molecular shape as a structure graph, relating reactants and products in a reaction network graph.

The concept of the informational content of graphs in relation to Shannon entropy was first discussed by Rashevsky [87] and Trucco [104]. The vertices of a graph are grouped based on the equivalence classes of G under automorphism: if two vertices can be said to be indistinguishable in terms of their topological description they belong to the same set. Each set can then be viewed as a single state. If we pick a random vertex from the graph the probability $p(j)$ of it belonging to a set j is then given by the proportion of total vertices that are in the set j:

$$p(j) = \frac{n_j}{n} \tag{2.11}$$

where n_j is the number of vertices in the set j; n is the total number of vertices.

We calculate the information content of a graph G with m distinct vertex orbits:

$$^{V}I(G) = -\sum_{j=1}^{m} p(j) \log_2 p(j) \tag{2.12}$$

The same can be done for edges of a graph or any invariant. Bertz [9] notes that for molecular exploration it is vital that the correct invariant is chosen in order to distinguish molecules. Rashevsky's[87] work is specifically motivated by exploration of molecular structural complexity.

In a series of papers [77, 78, 79, 80], Mowshowitz give a detailed exploration of this measure of graph information. He also proposes a different partitioning of the graph vertices. Instead of automorphism, graph colouring is used, which exposes structural features that are different from those exposed by automorphism. Different invariants used to define the partitioning yield different measures of information based on different features.

Karrenman [54] used the graph information measure to explore how information changes as a result of a reaction (between reactant molecules and product molecules). One drawback of this measure is that $I(G)$ is always 0 if the sets of invariants have equal probability. So a graph with 2 sets is just as complex as a graph with > 2 sets, if they have equal probability. Bertz [9] also uses Rashevsky's measure to explore molecular complexity, modified to distinguish between such graphs with equally likely sets. He also uses a different invariant for partitioning based on the number of two edge subgraphs. Various other measures have been proposed based on partitioning of the graph according to different invariants. Bonchev and Trinajstić [12] combine multiple graph features, assigning weight to each to produce a compound measure. This allows the measure to more accurately distinguish between different graph structures. Dehmer et al [22] overview graph measure of complexity as well as their applications to chemistry and other domains.

2.3 Chemistry as a Complex system

The natural world is often used as inspiration when developing complex systems in an attempt to mimic desirable behaviours like self organisation, concurrent computation and adaptability. It is interesting to look at Chemistry itself from the perspective of a complex system. From that perspective everything from biological systems like animals and plants to phenomena like electricity are emergent products of the interaction between molecules and atoms. As such chemistry as a complex system exhibits a vast array of emergent properties in an open-ended manner where observable systems become more complex over time.

2.3.1 Self organisation within chemistry

The first interesting behaviour that chemistry exhibits from the perspective of a complex system is self organisation. This behaviour can be observed at multiple levels and can result from different interaction mechanisms within the system.

For the most part elements outside of the noble gases do not exists in their atomic state naturally. Most atoms are part of molecules. Therefore we can think of molecules as the self organisation of atoms into structures that are energy efficient. The mechanism for this self organisation is the making and breaking of chemical bonds between atoms. For example we can talk about about water molecules, H_2O, as the product of the self organising behaviour when the system starts off with hydrogen and oxygen atoms.

Not every atom can bind to every other atom, and bonds between different atoms have their own specific properties like the length of the bond and its strength. These properties are functions of underlying properties of the participating atoms. For example the strength and length of an H–H bond is different from the strength and length of an H–O bond. Atoms are themselves a dynamic system of subatomic particles, and it is the interaction between these two dynamic systems that gives rise to the bond's properties. As such we can view our atoms as being both a datum describing the internal dynamics and a function on another atom.

2.3.1.1 Geometry

Outside of the properties of bonds, molecules also have shape. The geometric properties of molecules are a function of the atoms involved. For example the angle between the O atoms in CO_2 is 180^o; carbon dioxide is a linear molecule. The angle between the H atoms in H_2O is around 104^o degrees; water is not linear. The structure of a molecule plays a key role in its behaviour. The molecular structure is also not unique for the involved atoms. Isomers are molecules which have the same composition of atoms but the structure of the molecule is different, result-

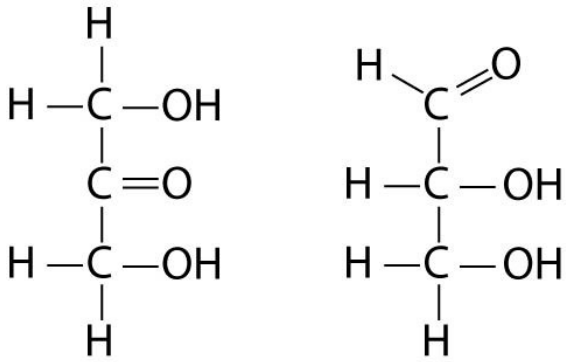


Figure 2.3: An example of two structural isomers that have different functional groups; notice the position of the oxygen atom

ing in different properties. There are two main types of isomers: structural and spatial. In structural isomers the atoms are connected in different ways between the isomers. In spatial isomers the atoms are connected in the same way but their spatial arrangement is different. Both types of isomer result in different molecular properties.

In organic chemistry a structural isomer can result in different functional groups. For example the arrangement between O, H and C in an organic molecule can produce a hydroxyl group or an ethyl group (figure 2.3. While these two isomers have the same number and type of atoms, their properties are very different. In spatial isomers the structure is the same however the geometric position of the atoms is different. For example two molecules that are mirror images of each other but not superimposable on each other. Again this has an effect on the properties of the molecule and possible interactions it can participate in.

Other emergent properties also exists in molecules. The atoms in the molecule have rotational and vibrational energy. There are a large number of different normal modes that the molecule can be in and it can transition between them by absorbing or releasing energy.

Study of each of these phenomenon, reaction chemistry, molecular geometry and vibrational behaviour are fields of science unto themselves. This is also true of everything else discussed in this section. From the perspective of chemistry as a

dynamic system the key feature is that the interactions between atoms allows for the creation of structures. We can think of these structures as being a product of self organising behaviour. This behaviour not only generates structures, it also enhances the descriptive language of the system by introducing new properties that were not present in the atoms alone. These new properties are still a function of the atoms involved in the molecule, however they are not trivial to derive simply by looking at the constituents. Similarly the interactions themselves are not trivial; not all atoms can bind to all other atoms or molecules yet the number that can is still vast, and therefore the system is interesting. Lastly the presence of isomers shows that the resulting organisations are not simply dependent on the initial conditions. There are many valid arrangements and the ones that are observed depend on the sequence of reactions and the environment.

Just by talking about the set of possible molecules that can emerge from an initial condition we already observe a combinatorially huge search space of possible final states. Moreover the act of generating these organisations gives us new properties that themselves play a part in the interactions.

2.3.1.2 Regularity in molecular structure

In chemistry the self organising behaviour can be seen at multiple levels. For example, instead of talking about molecules as the self organisations of atoms we can talk about the internal structure of the molecules themselves as being an organisation.

The simplest example of this would be polymers. A polymer is a large molecule comprised of repeating sequences of atoms. Here we have a molecule that has a regular internal structure, so not only have atoms combined to find a stable state but that stable state has a definable repeating structural unit. The repeating unit is called a monomer; it has its own structure that can be quite complex, resulting in various types of polymer which have very different properties. The type of polymer being created and the process also has an impact on the average length of the chain. This in turn has an effect on the properties of the material such as its stiffness.

Many examples of polymers have carbon based backbones: carbon atoms in the monomers connect together to form the polymer. There are other examples, like organosilicon compounds, where the monomer unit has a -[Si–O]- backbone. Si does not form strong bonds with itself but does with O; so a backbone Si–O is preferred over Si–Si. In DNA and RNA the backbone is made up of an even larger and more complex monomer unit containing oxygen, phosphorus and carbon. Different properties of the atoms in the backbone results in different properties of the polymers themselves, and finally different properties for the materials.

The properties of carbon make it a good option for polymer backbones, while silicon does not have the same option. However the product of an interaction between silicon and oxygen gives us a compound which exhibits similar behaviour to carbon (at least in terms of ability go create polymers). So the results of interactions give new avenues of exploration in terms of the search space. This is a contributing factor to the open-ended complexity of chemistry. Interactions expand the "functionality" of the interacting particles.

Polymers are not the only example of organisation within molecules. For example adding a second monomer gives is a copolymer. Here the resulting structure is made up of multiple units that appear throughout it. The organisation can vary, again depending on the properties of the monomers and the environment. The copolymer can have regularly alternating monomers, a larger repeating unit made up of a sequence of monomers, a random distribution of monomer types within the polymer, or periodically changing sequences of only 1 monomer. This gives an even larger combinatoric space. Not only the monomer type and the polymer size have an effect on larger properties of the material but also the specific internal organisation of the copolymers present.

Even carbon alone can show organisation into structures that have vastly different properties. Graphite and diamond are isomers of carbon, the difference is in the arrangement of the carbon lattice. Graphite has a flat lattice while diamond has a 3D lattice structure. Many other organisations of carbon have been synthesised and explored: tube structures, spheres of various sizes and other lattice organisations. So

even at the simplest level of only one type of atom, chemistry can show an explosion of possible end states each with very different properties at both the micro and macro scale.

2.3.1.3 Non Covalent bonding

So far we have discussed one type of interaction, covalent bonding, and shown the huge combinatoric search space that results from it. Even in this we have been brief: there are different types of covalent bond that can exist depending on the atoms interacting. This also has an impact on the properties of the molecule.

In chemistry apart from covalent bonds other bond types are defined, specifically ionic and metallic bonding. In covalent bonding two atoms "share" an electron. In ionic bonding one atom donates an electron to the other resulting in an attraction between the resulting ions. In metallic bonding electrons become de-localised, with the metal atoms "floating" in a "sea of electrons". Each of these is a different type of interaction between atoms and they produce very different properties. They do not exist in isolation from each other: de-localisation of electrons can also occur in organic molecules for example benzene rings. So the system has multiple methods of interaction; the atoms not only define the outcome of the interaction but also define the interaction mechanism itself.

Even this does not sufficiently describe the interactions between atoms and molecules. There are other types of interactions weaker then bonding; these are referred to as intermolecular forces. While weaker then the forces that form bonds they still influence molecules, especially their spatial arrangement. One of the results of these interactions is a very different type of self organising behaviour then the one discussed above.

The strongest intermolecular force is hydrogen bonding. Due to the distribution of electrons in molecules certain atoms in the molecule could be slightly electro-positive or electro-negative. In hydrogen bonding an electro-positive hydrogen atom is attracted to a lone pair of another atom (which is electro-negative). This attraction is

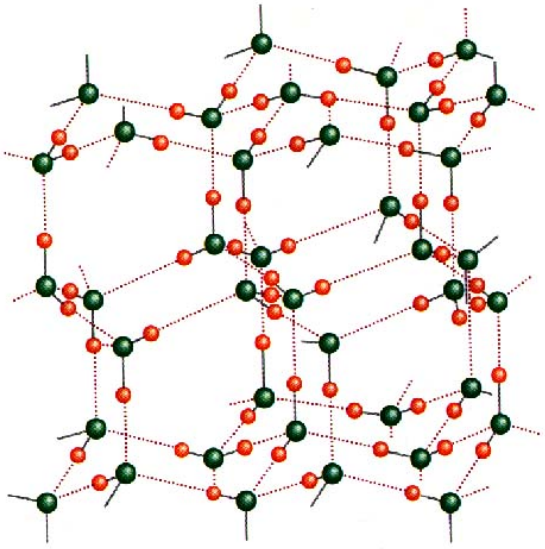


Figure 2.4: A visualisation of water molecules in solid state , dotted lines are hydrogen bonds.
Source: `http://gallery4share.com/h/hydrogen-bonds-ice.html`

quite strong although not actually a bond. In water molecules for example the –H of one molecule would be attracted to the lone pairs of electrons around the oxygen atom of another molecule. This interaction is important in understanding the properties of water. For example, the shape of the water molecule and the presence of this hydrogen bonding attraction means that at low temperature water forms a sparse structure of molecules held together by hydrogen bonding (figure 2.4). This results in the solid state of water being less dense then the liquid state, so ice floats on water. At a higher temperature some of the molecules have enough energy to overcome the hydrogen bonding force meaning the structure is less sparse making the liquid state denser.

There are other types of intermolecular forces described in chemistry, such as dipole-dipole forces and London dispersion forces (van der Waals forces). These rely on there being variances in the electro-negativity in the molecule. A dipole is a molecule that is electro-positive at one end and electro-negative at the other. London dispersion forces appear when the electrons around an atom are momentarily clustered tighter then usual, resulting in an instantaneous dipole.

Intermolecular forces also give us another type of observable self organisation. At low

temperatures there is not enough energy in the system to ignore the intermolecular forces, so they dictate the organisation in the solid. The atoms in the molecule and the shape of the molecule dictate how the intermolecular forces are distributed and therefore dictate the crystalline structure of the solid. This crystalline structure determines the properties of the solid, strength, brittleness etc.

The intermolecular forces also determine other properties of the substance, for example, the temperature at which state transitions occur, viscosity, and surface tension.

These forces are also responsible for multiple self organising behaviours which are extremely important in biology. Cells membranes are semi-permeable barriers which are made up of a lipid bilayer. The bilayer is made up of phospholipid molecules. The molecules are attracted to water (hydrophilic) at one end and repelled by it (hydrophobic) at the other, due to the intermolecular forces resulting from their structure. As a result of these forces the phospholipids arrange into a layer such that the hydrophilic ends face "outward". This can result in a number of different arrangements: a liposome, a micelle, or a bilayer.

These are structures consisting of multiple molecules, organisations that are the result of a different type of interaction between atoms. However these organisations do depend on the intramolecular forces discussed above. The fact that a molecule can be created which has the necessary shape and properties allow for these types of higher level organisations.

So we can say that the system as a whole has multiple types of possible interactions between molecules. The interacting molecules determine what type of interaction will occur. The products of the interaction and the properties of the product are also molecule dependent. The system exhibits a combinatorial explosion of properties and possible interactions: as the molecules grow they can be thought of as gaining properties. These new properties give rise to new interaction possibilities.

2.3.2 Emergence of functionality

The membrane example illustrates another property of chemistry as an emergent system. The self organisation of phospholipids is interesting in itself, however the resulting structure can also be discussed in terms of its function. The membrane as a whole essentially segments the environment. Because of how the layer interacts with other molecules, allowing some through while stopping others, we can talk of it as a filter. So chemistry produces organisations which can be thought of as having a function. The result of the interactions, both intramolecular and intermolecular, and interaction with other molecules in the environment is a new organisation that can be thought of as having a role, in this case filter.

This property of chemistry is what allows for growth in complexity and is essential to biological systems.

Most of biology at a molecular level is talked about in terms of the function that the particular molecules perform. The processes themselves are multi stage mechanisms with specific molecules performing specific functions.

2.3.2.1 DNA Replication

DNA and the processes surrounding it are probably the most interesting example. DNA itself is formed of two polymers made of monomers called nucleotides which are attracted to each other by hydrogen bonding between specific nucleotide types. The helix is structured since hydrogen bonding occurs between specific pairs of nucleotide meaning that DNA has a well defined internal structure.

Replication is a multi stage process with different molecules performing specific tasks (figure 2.5). Firstly replication begins at set locations along the DNA strand. A specific type of molecule, called a helicase, breaks the hydrogen bonds and produces two strands ready for replication. Another type of molecule, called DNA polymerase, then assembles a new strand by "reading" the existing strand and adding the correct nucleotides to create the stable hydrogen bond. There are other molecules involved;

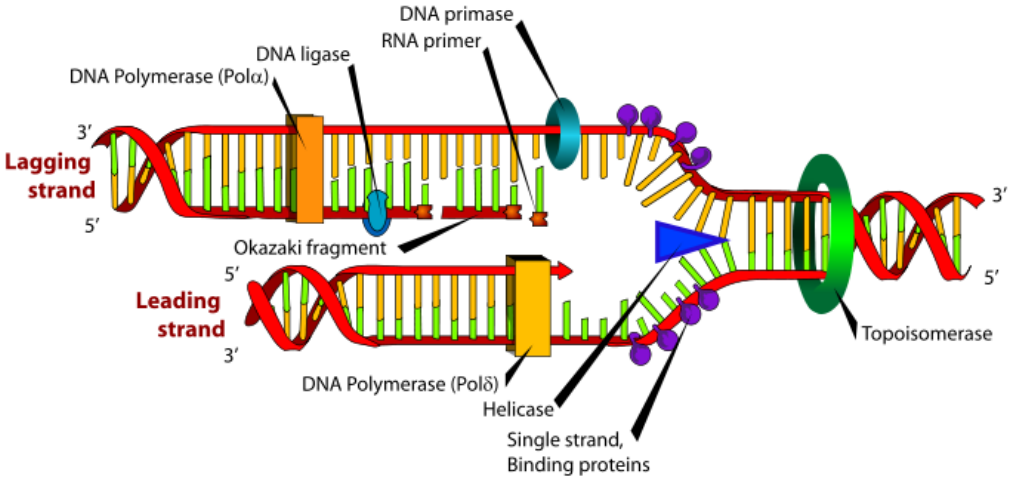


Figure 2.5: A diagrammatic representation of DNA replication. Source: `http://replicationdna.blogspot.co.uk/2011/03/structure-and-diagrams-of-dna.html`

for example, DNA clamps serve to speed up the process of adding new nucleotides. There are even molecules referred to as clamp loaders, which place the clamps on the strand. Because of the structure of DNA one strand can directly be read and a complementary strand assembled. However the other strand must be read "backwards". To accommodate this a specific fragment is added as a "break" to the backward strand. Nucleotides are then added between two break fragments and finally the break fragments are replaced by the relevant nucleotides.

This is an example of molecules not only performing specific functions but of mechanisms involving multi-stage processes. The properties that allow for these types of processes are essentially the same as those involved in forming membranes, although more complex. The molecules involved in the above process and many other biological process are proteins.

2.3.2.2 Protein shape and structure

A protein is a large heteropolymer, the monomers being amino acids. Because of its size, the intermolecular forces impose a shape. This is called protein folding (2.6. The structure of the molecule governs the characteristics of its intermolecular forces.This means different proteins will fold in different ways, and two instances of

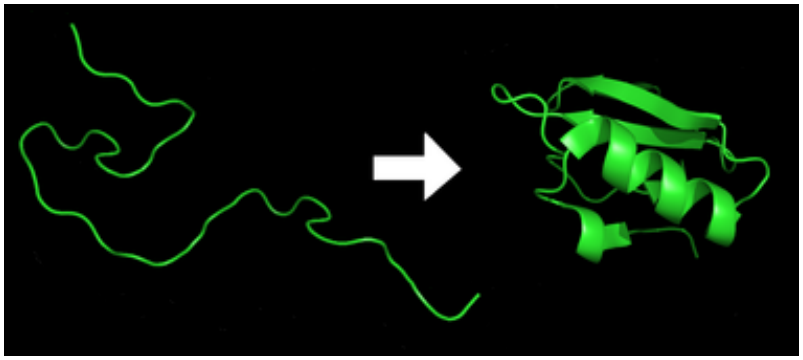


Figure 2.6: The effect of folding on a protein structure
Source: `https://en.wikipedia.org/wiki/Protein_folding`

the same protein will generally fold in the same way. The folded protein can take part in interactions with other molecules, however its specific shape restricts the possible interactions it can take part in. The molecule's interaction with itself restricts the interactions it can take part in with other molecules. This is an important part of the emergence of function. By restricting the possible interactions, the molecule can be seen as doing only one specific thing where only a few other molecule can interact with it successfully. There are many examples in biology; one of particular importance are enzymes. An enzyme acts as a catalyst for reactions, for example, breaking down other molecules like lactose into products that can be absorbed into the body.

2.3.2.3 Catalysts

Catalysts can be thought of as the simplest example of the emergence of functionality in chemical systems. While enzymes are large complex catalysts in biological systems, inorganic chemistry has many examples of catalytic reactions. A catalyst is a molecule involved in a reaction which is not used up during the reaction. It appears as both a reactant and a product of the reaction. Catalysts provide reaction paths that are more economical in terms of energy. This means that products are reachable at much lower temperatures. This again shows the complexity of the possible state space.

For example two molecules are inert at room temperature. The initial and final state

of the system are equivalent. However by introducing a new molecule which catalyses the reaction we now have a vastly different final state. The catalyst itself is still present in both initial and final state, however it has allowed access to transitional states that were inaccessible before.

2.3.3 Environmental interactions

The concept of catalysts leads to another interesting characteristic of chemistry. The environment, specifically temperature and pressure, play a key role in restricting the possible state space. The making and breaking of bonds requires energy, so the reactivity of the system and therefore its ability to exhibit interesting behaviour is directly tied to the energy in that system.

2.3.3.1 Effect on reactivity, equilibrium and organisation

In order for a reaction to occur a certain amount of energy is needed. The amount is based on the properties of the particles involved and is called the activation energy. If the activation energy can be reached then the reaction can take place and we move from reactants to products. Catalysts give us a reaction path which requires a lower activation energy. Most reactions are also bidirectional. If there is enough energy for the products to reach activation energy then we can move back from products to reactants.

The bidirectional nature of reactions means that the state of the system can reach an equilibrium. By changing the temperature we can push this equilibrium either towards reactants or products.

Conservation of energy however shows that no energy must be gained or lost, just converted. This means that if the products are of a lower energy then the reactants the difference must be present in the environment. The environment heats up. Conversely if the products are of a higher energy then the reactants the environment cools down. This can be viewed as a stigmergy mechanism. Interactions between

particles alter the environment, which could result in an increase or decrease of reactivity overall. Burning is an example of this: a small localised change to temperature is applied resulting in a reaction which in turn increases temperature allowing for others to also react. As the fuel burns we see a propagation of the initial change to the system.

Even without discussing reaction mechanics, the environment has an important impact on the system. As discussed earlier some self organising behaviour is present in solids like ice but not present in liquids or gasses. The environment not only dictates what reactions can occur but how ordered the system is. The more energy in the system the less important intermolecular forces become.

2.3.3.2 Segmentation of the environment

Outside of the influence of temperature on the system, chemical systems show an ability to modify their environment by segmentation, as discussed earlier. This is another key feature. Membranes allow for only some molecules to be present inside a specific region. This has the effect of focusing the search space of possible reactions to only those between particles that can permeate the membrane. Essentially the system inside is protected from external influence. This is very important for larger molecules. Proteins for example are very susceptible to changes in environment. The folded shape can be disturbed by a number of factors like temperature, pressure or presence of other molecules like salts.

So the ability for chemical systems to influence the environment can be thought of as an indirect interaction mechanism.

2.3.4 Simulation and modelling

In chemistry all of the interactions and properties described here can be derived from the position and behaviour of the electrons around atoms and molecules. There are many models that aim to predict certain properties and how particles will interact.

The model of electron sharing was proposed by Gilbert N Lewis [65] in order to explain how atoms bound together and what properties they must have for this to happen. The model does not however discuss other properties of interest, such as bond strength and length or molecular shape.

Shape is addressed by the VSEPR model [42]. Here the concept of electron pairs is used. Based on how many electron pairs there are around an atom the shape is derived by minimising the repulsive forces between the pairs. The model predicts quite exact bond angles, however experimental data shows that real angles have more variation, for example the model predicts the bond in water is around 109^o while experiments show it is 104^o.

There is a question as to what degree of accuracy is required. Given that our aim here is not simulation of real chemistry but exploration of a complex system, this error is perfectly within tolerance. The specific angle is not important, the important thing is that an angle can be calculated dynamically based on what is known about the particles participating in the bond. Of course bond properties like strength and length are still ignored here.

The Lewis model [65] also gives us some ability to reason about intermolecular forces. Since we know of the location of the lone pairs we can talk about the electronegativity of the molecule. Again the accuracy is extremely low but this is not as much of an issue.

In order to fully describe the properties of molecules a more accurate model of electron behaviour is required. This can be done by solving the Schrödinger equation, [91] which describes the wave function of the electrons. The wave function shows the probability of finding an electron at a specific point in space. However, the equation is not analytically solvable for more then one electron orbiting a nucleus (ie a hydrogen atom). Very accurate approximations can be made and are often used in order to model and predict chemical behaviour.

The field of computational chemistry is actively trying to produce higher quality models to predict chemical behaviour. This is extremely important in may areas

of chemistry and biology. Models for protein folding for example are actively being studied. Since proteins are such large molecules the interactions involved in the folding process are complex and numerous. The methodology varies depending on what property is being explored. Modelling quantum mechanics is extremely expensive computationally and as such not suitable in many cases. Molecular mechanics aims to model molecules using classical mechanics methods, these are computationally cheaper allowing for much larger molecules to be studied.

This modelling approach is not appropriate for our research. Our aim is to show an emergent complex system, not to model chemistry. Computational resource should be used to explore the dynamics of the system as a whole, so accuracy is not a concern. The important concept is that of different types of interaction should exist in the system, producing non trivial properties allowing a feedback mechanism to the environment. It is not important that we model water molecules or hydrogen bonding, but that the system can produce particles of various sizes and shapes and that those particles can interact in multiple ways depending on who is involved in the interaction and the environment.

2.4 Artificial Chemistries

Artificial chemistries (AChems) are designed to explore specific model chemical reaction systems, initially motivated by research into Artificial Life (ALife), with the hope of exploring self organisation in pre-biotic reaction systems. AChems have stretched beyond this initial motivation and algorithms inspired by the chemical model have been used for computational tasks.

2.4.1 Definition

A broad definition of AChems defines them as the tuple $\langle S, R, A \rangle$; this is the definition used by Dittrich et al [25] in their review of AChems.

S denotes the set of possible molecules. This set defines all valid molecules that the system can produce. It includes both the initial set of particles and all possible combinations that can result from the reaction rules. This does mean that S can be an infinite set. For example if the AChem is built to show polymerization then S can contain the monomer A and all polymers created by combining monomers eg AA, AAA etc. The type of members of S is unspecified: a molecule may be represented by a symbol, a string of symbols, a binary vector, or any other data structure depending on the specific AChem.

Generating members of S is done by applying the reaction rules defined in the set R. Each reaction rule has reactants and products. For example, R may contain the rule $A + B \rightarrow C$ which states that if an A and a B are present they are replaced by a C. Reactions rules can be explicit and deterministic, as in this example, or can be probabilistic and algorithmically defined. Rules can be conditional on environmental factors, the simplest being proximity: A and B have to next to each other for a reaction to occur. Other conditions such as requiring the temperature in the environment to be above a threshold for a reaction to occur are also possible.

The environment itself is defined by the reactor algorithm A. This algorithm defines when rules can be applied to the current molecules. It may be a 2D lattice where particles move randomly and rules are applied when two particles are next to each other. A simpler "well-mixed" algorithm may have a multiset of current particles from which subsets are chosen to test for reaction.

This is a very broad definition of AChems and allows for both very specific modelling of real chemical systems and very abstract systems that fit the definition but are not intended for reasoning about specific chemical processes. The second type of AChems are useful for exploring the expressive and computational power of chemistry-like systems. The definition is so accepting that even cellular automata with self replicating patterns can be viewed as AChems [25].

2.4.2 Rewrite Chemistries

A very simple way to define an AChem is as a rewriting system. Molecules are defined as symbols or strings of symbols. Rules are rewriting rules which replace symbol patterns and the environment can be a string of symbols or more simply a multiset.

The Chemical Abstract Machine (CHAM) developed by Berry and Boudol [8] is an example of this type of AChem. The motivation behind CHAM is to develop a model for concurrent programming inspired by the highly parallel nature of molecular interactions.

Within CHAM molecules are defined as terms of an algebra. Reactions are then rewrite rules. The environment is defined in terms of multisets called solutions. A solution S is a multiset of molecules. Additionally a solution can be viewed as a molecule which appears as a sub-solution in another solution S'. This sub-solution notion gives the idea of a "membrane". Within a membrane rules are applied independently of the wider environment and therefore sub-solutions can evolve independently of each other. Lastly the notion of an "airlock" is added, which allows molecules to enter or leave a sub-solution.

These concepts are defined as four general laws. The Reaction law defines when a rule can be applied. The Chemical law states that reactions can be applied to any solution. The Membrane law states that sub-solutions evolve independently of environmental context. The Airlock law states when molecules can enter or leave solutions.

Defining rewrite rules for the system is then done depending on what the instance of CHAM should do. The rewrite rules are classified by CHAM into three kinds: heating, cooling, and reaction rules. Heating rules decay molecules (a large molecule decomposes into constituent parts). Cooling rules compose molecules (a set of smaller molecules bind to form a larger one). Reaction rules transform usually ions, which are defined as molecules which cannot be heated further. Heating and cooling are

seen as reversible and are generally written together; reaction rules are seen as non-reversible.

CHAMS are inherently parallel: any rule can be applied at any time provided the reactants are not being used by another rule. The membrane law means that subsets can act independently with no interference yet still communicate via the airlock law. CHAMS are non-deterministic: there are no laws defining which rules are applied if multiple rules are applicable.

Specific CHAMS have been built to reason about concurrent systems, for example by Inverardi and Wolf [52] for analysing architectures like concurrent compilers.

A system called ARMS (Abstract Rewriting System on Multisets) [103], developed by Suzuki and Tanaka, is similar to CHAMS but is built to explore how rule ordering effects system behaviour. Like CHAMS, ARMS defines molecules as abstract symbols and reaction rules as rewrite operations over the reactor. Again the reactor is a multiset of molecules. There is no concept of sub-solutions, however ARMS has a concept of input strings which probabilistically adds molecules to the reactor, as well as an order for reaction rule application. By varying the precedence of the reaction rules and frequency of input the behaviour of the system can be varied. Suzuki and Tanaka explore these and classify different behaviours of ARMS. Implementations of ARMS have been studied that model real chemical oscillation reactions like the Belousov-Zabotinsky reaction [102]. In that paper there is also an exploration of how the ratio between heating and cooling reactions present in the system effects the dynamics of the system.

There are many other computational models inspired by the chemical metaphor. Paun and Rozenberg's P System [84] take inspiration from cell-like organisation. Like CHAM there is the concept of a membrane as well as a subsolution giving a hierarchy of membranes. However in P systems, membranes contain not only a multiset of molecules (again defined as symbols) but also rules that act within that membrane on the multiset.

2.4.3 Particles as Model and Function

There are abstract AChems that do not use explicit rewrite rules as the above. Fontana's AlChemy [37] is based on λ-calculus. Particles are λ-calculus expressions and the product of a reaction is defined as one particle acting on another. For example if particles A and B interact the interaction is defined as the function $A(B)$. When this function is reduced to normal form, which Fontana describes as stable, we get the products of the reaction which are then added back into the reactor. Note that $A(B)$ and $B(A)$ do not necessary produce the same result. Fontana explores this model showing self replication and self maintenance. He shows that the emergent behaviour can best be described by emergent laws not connected to the underlying mechanics of λ-calculus.

Fenizio [23] modifies AlChemy by using combinators instead of λ terms for particles. Reactions are still handled in the same way since combinators are treated both as operations and data. This model also has a concept of conservation, that there is a finite number of atomic particles. Reactions never generate new particles or destroy existing ones.

The notion of reactions as operators over particles is also explored by other AChems. For example, Banzhaf et al [3] build an AChem where particles are binary vectors. When two particles (A and B) interact one is folded into a matrix A'. The product of the reaction is then result of applying a mathematical operation on A' and B to produce C, the binary vector product of the reaction. Different reactors are defined and explored, including a multiset where particles are chosen randomly for interaction, and a 2D lattice structure where particles are chosen from the Moore neighbourhood randomly or probabilistically based on a Gaussian function.

Dittrich and Banzhaf [24] produce another AChem similar to the above. Again particles are binary stings. A is folded into an automaton by mapping 4-bit strings from A into instructions for the automaton. B is then fed into the automaton, and the product of the reaction is the output of the automaton.

As with Fontana, in the above models reactions are non-commutative.

In these types of AChems the reactions are not explicitly defined, instead a reaction mechanism is defined, analogous to the physics behind molecular interaction. One drawback of this approach is that the behaviours of particles (for example which are active or inactive within the system) cannot be specified and must either be experimentally explored or explained systematically. Finding sets of particles that interact in interesting ways can be difficult, especially if set of possible particles is very large.

2.4.4 Lattice Based Environments

AChem development was initially motivated by research into ALife as a mechanism for exploring chemical systems. The models surveyed thus far are very abstract, and more interested in exploring the computational characteristics of chemistry-like systems. There is however a number of AChems that explore specific chemical reaction systems with a view on showing self-organising and self-maintaining behaviours.

The Substrate, Catalyst, and Link (SCL) AChem originally described by Varela et al [105] and later revisited by McMullin [72] is an example of a relatively simple AChem aiming to show self repair of cell walls. This is also an example of an AChem where the reactor is a 2D lattice structure. The particles in the systems are represented by a symbol and reactions are rewrite rules. The reactor is a lattice and a reaction can occur when the reactants are next to each other (Moore neighbourhood). The model defines 4 particles L, $L+$, K and S, and 6 reaction rules. The rules are written to show that cell wall, made up of linked L particles, can be repaired if ruptured. There is also a set of conditions on the movements of particles. The cell wall can only be passed by S particles which are said to diffuse through it. K particles cannot exit the cell. Linked L particles have to stay within one square of the connected particles.

SCL is quite a simple model and as a result very limited. In order to explore more complex phenomena such models must also become more complicated.

Hutton's Squirm3 [47] model has similar design principles to SCL, however is more complex. Squirm3 was originally developed to show self replication and information copying of molecular sequences [47]. The particles in the system are defined as having a type (represented by a letter) and a state (represented by an integer). Quite unlike most of the systems we have explored so far the particle cannot change type. This means reaction rules of the kind $A + B \rightarrow C$ are not valid. Reaction can however change the state of a particle. Particles can also form and break bonds through reaction rules. While the reaction rules are explicitly defined, most use variables instead of specific particle types. For example rule 4 in the original paper is:

$$x3 + y6 \rightarrow x2y3 \tag{2.13}$$

This rule states that if particle of any type with state 3 interacts with a particle of any type with state 6, the two particles bond and their states become 2 and 3 respectively.

The particles move around freely and randomly in the 2D space; bound particles have to stay in each other's Moore neighbourhood and only one particle can occupy a space at any given time.

Hutton [47] reports that experiments show the system running as expected, producing copies of molecular sequences (figure 2.7).

There is some emergent behaviour reported: different molecular sequences emerge which are themselves capable of replication. Hutton then goes on to vary the environment by deleting half of the 2D space and reseeding it with random particles, to explore if the replicators adapt. Indeed shorter replicators become prevalent. Lastly Hutton shows that replications can emerge from a completely random, inert soup of particles given a sporadic outside event which excites a particle changing its state.

Hutton [47] notes the absence of complexity growth and competitive behaviour within his system. One of the properties of natural chemistry is that as a system there is growth of complexity over time. This is a key feature when discussing

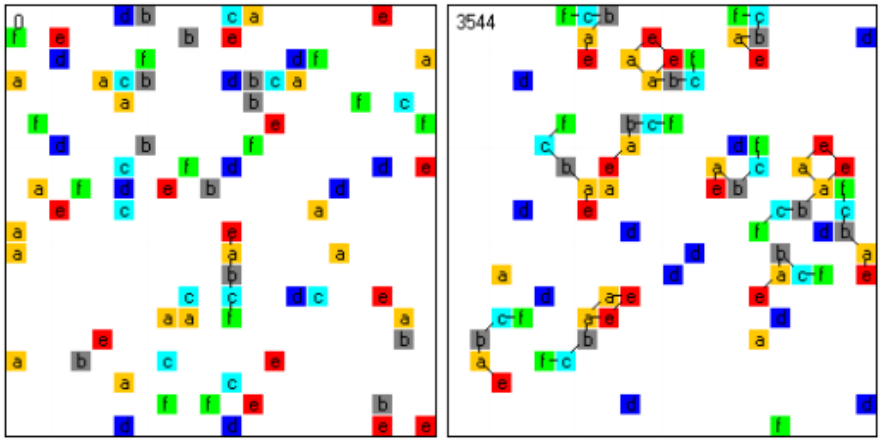


Figure 2.7: Squirm3.(a) shows a reactor with one replicator (e-a-b-c-f) after run (b) shows replication and mutation of the original string into many instances[47]

the question about origins of life, yet creating open ended AChems that can replicate it is very difficult.

Squirm3 is none the less useful for investigating how replicators behave under different conditions. Lucht [68] uses the model to investigate if replicators can survive in the presence of parasitic behaviour.

Later work by Hutton [50, 48, 49] adds more rules in order to replicate cell membranes and introduce cell replication as well as competition for resources. In these papers the initial 8 rules of Squirm3 become 41 rules, and new states for particles are added. This speaks against explicit reaction rule systems if the goal is open ended complex systems. Adding rules in order to increase complexity seems like explicitly engineering a system as opposed to defining rules which exhibits the desired emergent behaviours. Explicit rules in essence limit the possible search space by defining tight roles for the particles; like the SCL system each particle has a function defined by the reactions it is part of. A more fruitful approach could be systems like AlChemy where a generic mechanism for reactions is defined, as opposed to specific reactions. This means it is more feasible for a reaction to produce a new type of molecule that is better suited to the environment, and which opens up new, more efficient reaction paths for the system.

Ono, Naoaki and Takashi [82] describe an AChem model with a more complex environment, and define more ways for particles to interact outside of reactions. Like Squirm and SCL, particles are represented by a symbol. In this case there are 5 particle types. There are also 6 explicit reaction rules, which convert particles to different types, for example $A + A \rightarrow A + E$. There is no concept of a bond between particles like in Squirm.

Unlike the previous models a more probabilistic approach is taken to reactions. Each reaction has a rate constant, which defines the likelihood of the reaction occurring. In real chemical systems the kinetics of the system determine how likely a reaction is to occur as a function of the energy in the system. Here the reaction rates are not a function of some global temperature, but are directly chosen by the authors, however it is still a more realistic approach.

While the model is based in a lattice like the previous two examples, the lattice itself is different. Each cell can hold a multiset of particles, as opposed to only one particle. As such the environment can be viewed as a lattice of "buckets". Particles can move between neighbouring cells as a result of two processes, diffusion and repulsion. Each cell contains exactly 100 particles so when a particle moves it effectively swaps with a particle from the cell it is moving to.

The diffusion process means that each particle has a chance to move to a neighbouring cell with a constant probability. The repulsion process probabilistically moves a particle depending on the other particles in the current cell. To achieve this participles are defined as being hydrophilic or hydrophobic. In the model M is hydrophobic and A, E and X are hydrophilic. That means that M will move to a neighbouring cell with a probability proportional to the number of A, E and X particles present in current cell and vice versa.

The repulsion mechanism adds a way for particles to influence each other that is not based on a reaction occurring. This behaviour is also important in real chemistry, especially in producing self organising structures.Later in the work they create amphiphilic molecules by binding a hydrophobic and hydrophilic particle together and

calculating the force exerted on these particles by others representing water. With this they show formation of micelles and membranes as well as self maintenance and fission of cells after the introduction of reactions.

In the original paper [82] the authors also show behaviour that can be interpreted as cell division within a 1D lattice of multisets. Later in the same paper they extend this this to 2D lattices with cells again containing more then one particle, and again experiments show cell maintenance and replication as well as emergence from a homogeneous initial state. Yet another paper from the group [69] extends this idea to a 3D model; again cell formation, replication and self-maintenance is observed. The authors also report that some of the initial configurations do not produce cell like organisations but completely different structures (figure 2.8). These structures are stable and within them particles can move around but they are not cell-like at all.

The various Ono papers show that allowing the particles to interact in different ways, not just by reacting, can yield interesting results. It shows that reaction rules do not have to be the only driving force behind the behaviours of the system.

There are AChems that specifically focus on weaker interactions between particles, rather than reaction rules. Mayer et al [71] model only hydrophobic and hydrophilic forces between particles. This is achieved by defining an automaton to simulate the forces between particles. Again the environment is a 2D lattice and particles move based on the force exerted on them by the surrounding particles. Like the Ono work, self-organisation is observed, however there is no mechanism for a cell to grow so fission is not observable.

2.4.5 Graph Based Approach

The Mayer [71] model is much closer to the mechanics of real chemical systems, at least in regard to intermolecular forces. There are other AChems which aim to be closer to the real chemical systems.

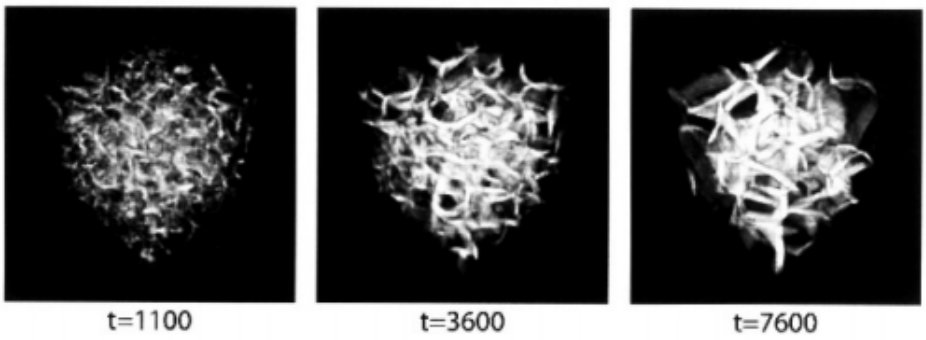


Fig. 5. From a random homogeneous initial configuration, filaments form and begin to organize

Fig. 6. From an initial configuration of two parallel membranes, tubular structures form to connect them

Figure 2.8: Resulting self organisation from [69]

Benko et al [6] develop an AChem based on graphs that aims to be much more realistic while still being computationally tractable. The main aim of the system is to be able to generate realistic reaction network graphs for exploring real chemical systems. This is done by using valence shell electron pair repulsion (VSEPR[41]) rules to predict molecular geometry. The model is based on the minimizing the repulsion between electron pairs and provides realistic molecular shapes.

In the Benko et al model molecules are defines as graphs with vertices being atom orbitals and edges being overlapping orbitals. The orbitals described by the vertices are very simplified versions of their real world counterparts. The work does however show they are sufficient approximations for what is required. Reaction rules are defined as graph rewrite rules. This approach means that many reaction paths may be possible for two molecules interacting. Because the representation encodes information about the interaction of the orbitals, the model can make an informed decision about which reaction path to take based on which one is less energy intensive. An extension to the model breaks down the interaction between molecules when they are attempting to bond, in essence making a reaction a non atomic step.

Benko [5] shows that the model can create realistic reaction network graphs. The model uses experimental data about strength and length of bonds to perform these calculations. It remains to be seen if the model can be made to work with artificial atoms whose properties are not realistic. The level of simulation involved in the model is important, given the aims of the system. However, in the context of exploring the computational power and open-ended nature of chemistry like systems strict adherence to the mechanics of the real world is not required. The important thing is that the system has mechanisms that perform similar functions to real world counterparts. For example, explicitly modelling electron orbitals and overlaps is not as important as having a concept of bond being the result of interactions between particles. Simulating how a bond forms or breaks due to the electron interactions is not as important as having the reaction mechanism be non-atomic and non-deterministic.

2.5 Sub-Symbolic Artificial Chemistry

AChems can be used for exploring mechanisms in biological or pre-biotic systems. Because the main requirement is to explore how these systems behave, a number of underlying mechanics are simplified or ignored all together. In many of the models described above, instead of creating mechanisms that allow for a reaction to take place, the set of possible reactions is often predetermined to only those of interest. They also completely ignore molecular size and structure. Many tend to have reactions as rewrite rules where a particle of type A becomes a particle of type B, which effectively results in a system that has no concept of conservation of mass.

Molecular structure is an important component in the behaviour of molecules. However, by reducing a large complex molecule, or even an atom, to a single symbol, and the interactions between molecules to those predefined by the reaction set, the effects of structures are ignored.

Another common omission is the energetics mechanics of a reaction. Many systems do not have an environmental temperature, and the reactions do not include the need for energy input or output. In real chemical systems energetics leads to multiple reaction pathways and gives rise to the emergent property of a catalyst, which allows for a reaction to occur at a cheaper energetic cost.

All of these omissions and simplifications make sense in the original systems, given the goal of most AChems is not to simulate reactions but instead to explore already defined reaction networks or processes.

If the goal however is to explore the complexity of systems like chemistry and their inherent combinatoric power then all of these omissions severely limit the scope of system. In real chemistry the properties of the system are emergent from the underlying interactions between atoms. Because atoms themselves have internal structure that gives rise to their dynamics, these interactions are non trivial. So if we wish to create a system that has a similar scope for complexity and emergence the basic building blocks should have their own non trivial properties and should

be able to interact. Additionally, the interaction itself should in some way change the properties of the particles interacting. In real chemistry the dynamics of the electrons around the atom is the basic property from which others (geometry of molecules, ability to bond, strength of bond etc) are derived.

This suggests that we would like to build a system such that the basic building block has exploitable properties from which we can derive all others. That is the core concept of Sub symbolic Artificial Chemistries in general [36] and more specifically RBN-World [33, 34, 35].

2.5.1 Random Boolean Network

In RBN-World the basic building block (atomic particle) is a random boolean network (RBN). RBNs were first discussed by Kauffman [56] as a method for modelling gene regulatory networks in an abstract, high level and computationally tractable way. Since then RBNs have been used as tools for a number of different tasks that require a computationally tractable dynamic system with interesting, non trivial properties [55].

An RBN is a directed graph. The graph has N nodes, each of which has a binary state. Each node changes state according to a randomly-chosen boolean function. There are exactly K incoming edges for every node and arbitrary number of outgoing edges. For each node, the state of the nodes connected by incoming edges is the input to the boolean function, so each boolean function has exactly K inputs. RBNs are synchronous; at each timestep t every node updates its current state based on its inputs and its boolean function. The inputs are the states of the input nodes at time $t-1$. Overall the state of an RBN at time t can be defined as a boolean vector of the current state of all nodes.

More formally an RBN is a finite directed graph X with node set $v[X] = \{1, 2, ...N\}$. Each node has a state $s_i \in \{0, 1\}$. The state of the RBN is then the N-tuple $\{s_1, s_2...s_N\}$. At each times step t each node state s_i is updated by an associated

function $f_i(s_{j_1}, s_{j_2}, \ldots s_{j_k})$ where $j_1, \ldots j_k$ are the nodes which input to node i.

The topology of the network is unspecified apart from requiring exactly K incoming edges to each node, where the source nodes are distinct. A node can have an incoming edge from itself.

There are a number of different modifications to the standard RBN model; Gershenson [39] gives an overview. Some use an asynchronous update rules, either randomly updating some nodes or using a set schedule for which nodes to update at each time step. There are even extensions that do not use boolean functions and binary states, or that use probabilistic update functions.

The properties of RBNs are being explored and there is a wealth of literature on the subject [96, 29, 45]. RBNs are dynamic systems which gravitate towards attractors in the state space as a result of different initial conditions. These attractors are a set of states which repeat forming a cycle in the state space. Based on the parameter K the RBN can be classified as ordered ($K < 2$), chaotic ($K > 2$) or critical ($K = 2$) based on how changes to the initial state propagate through the system.

RBNs themselves do not behave like real world atoms. If we were aiming to create a simulation of real chemistry this would be a problem and probably make RBNs unsuitable. However the goal here is a system that exhibits similar emergent and complex properties to chemistry. What we need from our basic particle is the ability to fulfil the same role as atoms do in chemistry. That is to provide a set of properties which govern interaction and have the ability to combine with instances of themselves. And the result of a combination should influence the properties of the components. RBNs fit this description well. It is easy to combine RBNs, which results in changes to behaviour. In the critical RBN region these changes are not chaotic, and have a chance to influence the behaviour of the RBN.

2.5.2 Binding Mechanism

Explicitly defined reaction rules limit the scope of the system. In RBN-World, instead of defining the possible reactions like in most AChems, what is defined is the reaction mechanism. This mechanism defines:

- A pre-condition defining what must be true of the reactants before an interaction can occur.
- An interaction algorithm which defines how two reactants interact.
- A post-condition, what must be true after the interaction has occurred.
- A binding algorithm which defines how the two reactants bind.
- A stability condition which defines what must be true of the resulting product.

If the stability condition is not met then the product decomposes until the condition is met; this allows for reactions like substitution where A + B–C $\rightarrow$ B + A–C.

In order to allow for RBNs to interact and bind, the original RBN-World uses bRBNs. A bRBN is defined like a normal RBN with the addition of special "binding nodes". These nodes do not perform a boolean function, instead they have a set binary state. They are not connected to the RBN in the same way normal nodes are; a binding node has exactly 1 outgoing connection to a non binding node in the bRBN. A binding node that is not part of an interaction has its state set to 0.

The conditions that make up the reaction mechanism are a function of a binding property of the bRBN. In the original thesis for RBN-World a number of different binding properties are investigated, as well as different conditions. To illustrate the mechanism here, we use the cycle length of the RBN as a binding property, and equality of cycle lengths for the binding condition:

1. Two reactants are chosen (A and B) and a binding node on each is chosen (a on reactant A and b on reactant B)

2. Pre-condition: The binding properties (cycle length) of A and B or a sub-molecule that contains the selected binding nodes (A' B') are equal.

3. Interaction Algorithm: The chosen binding nodes (a and b) are set to 1. This results in a change in the bRBNs denoted by $A-$ and $-B$ so the binding property is recalculated. If a sub-molecule was chosen then the binding property is recalculated for that sub-molecule $A'-$ and $-B'$

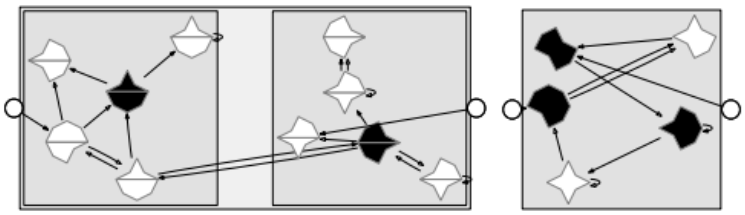
4. Post-condition: The binding properties of $A'-$ and $-B'$ are equal.

5. Binding Algorithm: Product A–B is created by removing the binding nodes a b, and directly connecting the nodes that the binding nodes where outputting to. $A-$ and $-B$ are recorded so that the product can be decomposed back to A, B as well as for the stability condition.

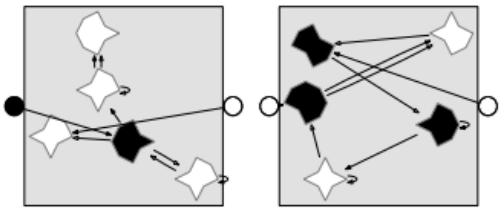
6. Stability Condition: The binding property of all sub-molecule in the product are equal. Any bond where the condition does not hold is broken.

If the two reactants A and B are atomic, the stability condition is not required. However the reaction mechanism allows two molecules to react, not just atomic particles (figure 2.9).

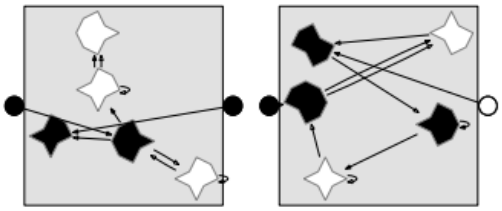
As an example reactant A could be the molecule X–Y and reactant B the molecule G–H. The chosen binding nodes are a binding node on Y and a binding node on G. Now the pre-condition is checked from the smallest sub-molecule that contains the chosen binding nodes. That is we first check if the cycle length of $-Y$ is equal to the cycle length of $G-$. If they are not equal we also check the binding condition for X–Y equal to $G-$ and X–Y equal to G–H. If the condition passes we then set the binding node to 1 and recalculate. For example if the pre-condition is true for X–Y and $G-$ we recalculate the binding property for X–$Y-$ and $-G-$. The post condition is then checked against these and if it is true then a bind occurs. In the example that would result in a bind between X–$Y-$ and $-G-$ giving us a product of (X–Y–G)–H. Here the parenthesise denote a sub-structure. There is a difference between X–Y–G–H and (X–Y–G)–H.



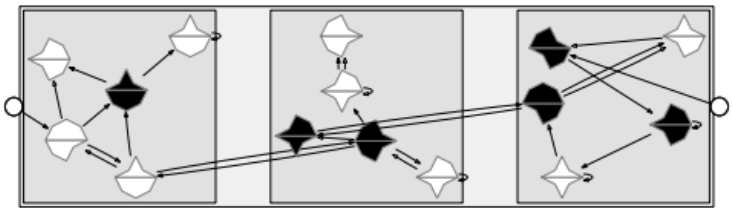
(a) Reactants (A–B) + C



(b) Initial interaction –B) + C



(c) Filling of the selected bonding sites in each reactant represented as –B)– and –C



(d) Composition of the product (A–B–C)

Figure 2.9: Representation of binding mechanism in RBN-World, from [33]

The stability condition now has to be checked. Because Y is now bonded on both sides for the molecule to be stable the binding property of $X-$ has to be equal to $-Y-$. The same is true for G, the binding property of $-G-$ has to be equal to $-H$. If one of these is no longer true the bond is broken. For example if $X-$ does not equal $-Y-$ then the X–Y bond breaks leaving the products $X + (Y\text{–}G)\text{–}H$, note the parentheses stay because the generated molecule was $(X\text{–}Y\text{–}G)\text{–}H$. If we had created $X\text{–}Y\text{–}G\text{–}H$ then the product would be $X + Y\text{–}G\text{–}H$.

So as a result of the stability condition our products could be $X + (Y\text{–}G)\text{–}H$, $X + H + Y\text{–}G$ or $(X\text{–}Y\text{–}G)\text{–}H$

2.5.3 Sub-Structures

Lastly a note on the importance of sub-structure. Assume our product is $(X\text{–}Y\text{–}G)\text{–}H$ and we wish to react it with another particle N, which is atomic. The chosen binding nodes are one on N and one on X.

Now we would first check the pre-condition for N and $X-$. If it does not hold we then check the pre-condition for N and $(X\text{–}Y\text{–}G)-$. If this also does not hold we check the pre-condition for N and $(X\text{–}Y\text{–}G)\text{–}H$. Assuming no decomposition our possible products are:

1. $(N\text{–}X\text{–}Y\text{–}G)\text{–}H$ if the first pre-condition holds
2. $(N\text{–}(X\text{–}Y\text{–}G))\text{–}H$ if the second pre-condition holds
3. $N\text{–}((X\text{–}Y\text{–}G)\text{–}H)$ if the third pre-condition holds.

If however our initial reaction produced $X\text{–}Y\text{–}G\text{–}H$, we can only check the pre-condition for N and $X-$ and the pre-condition for N and $X\text{–}Y\text{–}G\text{–}H$. So our possible products are:

1. $(N\text{–}X)\text{–}Y\text{–}G\text{–}H$ if the first condition holds

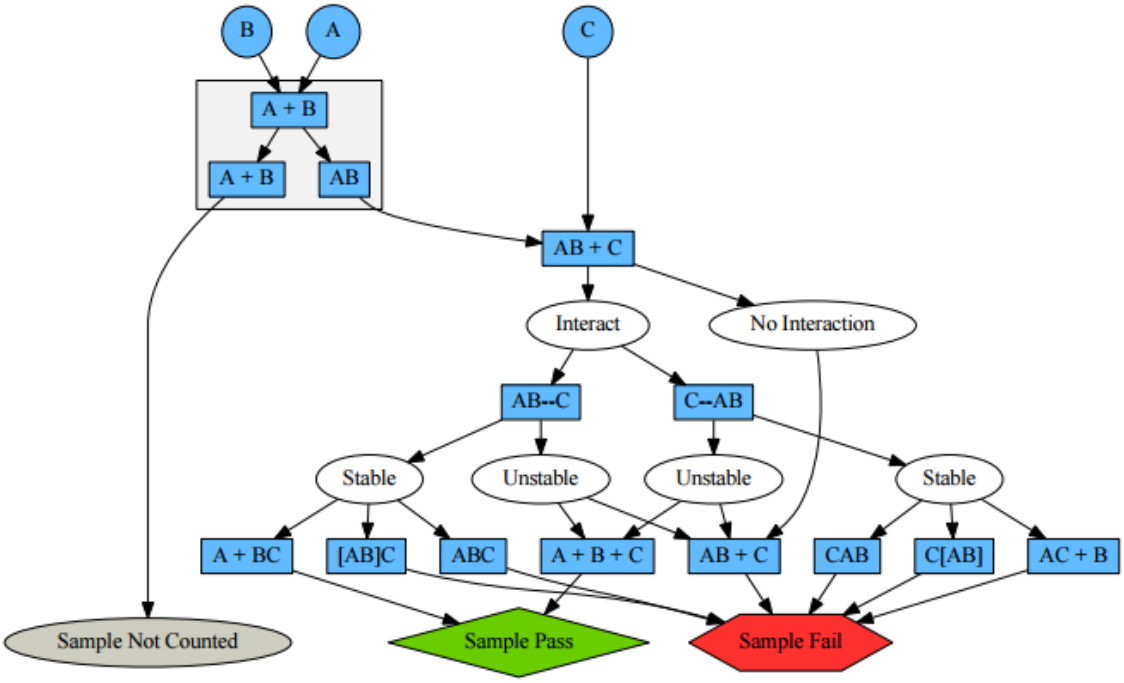


Figure 2.10: Possible reaction paths, from [34]

2. N–X–Y–G–H if the second condition holds

The reaction mechanism allows for a wide range of possible reactions as well as a number of reaction paths leading to the same results (figure 2.10). The mechanism also allows for any arbitrary reactants to be able to react, provided conditions are met. This is important since the set of possible RBNs that can be used as atomic particles is huge.

Exploring the RBN space in order to find useful atomic bRBN sets that generate interesting reaction networks is also an issue. In the original RBN-World work the exploration of this space is performed by a genetic algorithm. The fitness function is based on the length of the largest loop within the reaction network generated. The GA chromosomes are the individual bRBNs that make up each set. Mutation is done by allowing one change to either the structure of a bRBN or the truth table of its nodes. Alongside this if an atomic set does not interact or takes too much computation to build the reaction network that individual set is replaced by a new random set of bRBNs.

In the results that thesis shows that RBN-World can generate interesting reaction networks. It is reported that one set of bRBNs produced 363 synthesis reactions, 515 decomposition reactions, 517 directly catalysed reactions, and 86 auto-catalytic reactions. Overall the reaction network had 1,286 reactions and a maximum loop length of 8 and 63 unique molecular structures.

This all suggests that the reaction mechanism gives rise to features in the resulting reaction network that are not hard coded by explicit rules. Features such as auto-catalytic sets emerge from the system given the right set of initial bRBNs. This is encouraging and suggests that the sub-symbolic approach can produce systems that show emergent properties and maybe even growths in complexity.

There are still a number of omissions from the RBN-World model. While the reaction mechanism is based on properties of the bRBNs involved the number of possible connections per bRBN (the set of binding nodes) is set to two. This means that structurally the system can only ever produce chain composites. Another side effect of this is that the composites created always have two possible binding sites meaning that you can't have a composite which cannot bind any more. This problem can be dealt with by adding more binding nodes to the bRBN and having bRBNs with different numbers of binding nodes involved in the same reaction set. However the number of binding nodes to add is still an arbitrary decision. In the real world atoms have a valency that is a function of the number and arrangement of electrons. It should follow that in a sub symbolic AChem the number of possible bonds is a function of some underlying property of the RBN.

Related to the above there is also no concept of shape. The placement of the binding nodes is arbitrarily decided. Again in real chemistry the angle between formed bonds is a function of the electron arrangement. The shape of molecules that results from this has an impact on how they behave and what they can or cannot interact with.

RBN-World only has a notion of one type of bond. Weaker intermolecular interactions such as hydrogen bonding play a key part in the emergence of self organisation within real molecular systems.

Even with just covalent bonds, RBN-World ignores concepts such as bond strength. Within RBN-World all reactions that can happen do happen, and decomposition of bonds is only a result of the introduction of a new reactant. The concept of energetics would add an environmental factor so that a reaction that can occur will occur with a probability based on the energy in the environment and of the reactants. This also means that a molecule could decompose spontaneously if there is enough energy for it break its own bond.

2.5.4 Kinetics in RBN World

One solution to the kinetic issue was explored in a Masters project, later reported in [36]. In the project a concept of energy is introduced. Each reactant and product has a latent energy and an excited energy. The latent energy is defined as the transient length of the bRBN from an initial all 0 configuration (initial values of all nodes are 0). The excited energy is the transient length from an all 1 configuration.

Using these energies we can calculate the potential energy of the reactants and products as well as the activation energy for the reaction. In general for a reaction A + B $\rightarrow$ AB

Potential energy of reactants : $A_{t0} + B_{t0}$

Activation energy : $A_{t1} + B_{t1} + AB_{t1}$

Potential energy of products: AB_{t0}

From this we can calculate the amount of energy needed for the reaction in both directions. For the forward reaction energy needed is *activation energy − potential energy of reactants*. For the reverse reaction the energy needed is *activation energy − potential energy of products*.

Using a Boltzmann factor we can then calculate the probability of the reaction occurring in both the forward and reverse directions. The Boltzmann factor is a function of both the energy needed and the temperature of the system.

The reaction mechanics work in the same way as before with two additions. First in step (5) the binding algorithm occurs with a probability based on the forward reaction Boltzmann factor. If the bind occurs then in step (6) before we check for decomposition we first see if the new bond instantly degrades due to the energy of the system with a probability based on the back reaction Boltzman factor.

Experiments on this extension show that it has an effect on the overall reactivity of the system. At low temperatures the system is non-reactive and at very high temperatures any products quickly decompose back to reactants. In the middle of the extremes, experiments show that different reactions are possible at different ambient temperatures. Even with the same initial set of bRBNs at different temperatures different atomic particles are most active.

Overall this extension shows that adding a kinetic mechanism and an environmental component (temperature) changes the behaviour of the system leading different types of compounds to become prevalent based on environmental conditions.

The extension does have problems however; the definition of potential reactant energy and activation energy leaves the chance that the activation energy may be smaller then the potential reactant energy. In other words there exist reactions that require a negative energy input to occur.

Feedback to the environment is also not addressed. While a reaction can easily be classified as exothermic or endothermic (releasing or absorbing energy) and the amount of energy can be quantified, that energy is never added or subtracted from the temperature of the system. This omission means that some behaviours which would be interesting to observe are not possible. Specifically if a reaction is highly exothermic there is a chance that it would heat up the environment enough to allow other reactions, currently not possible to occur.

This extension shows the malleability of the RBN-World model and more specifically of RBNs themselves. While being computationally cheap RBNs offer a large number of exploitable properties which are non trivial, and at the same time not random. This suggests that there is scope for extending the original RBN-World to address

some of the omissions, or creating a new sub-symbolic AChem based on RBNs.

In this review we have shown a number of different complex systems and looked at chemistry from an emergent system perspective. We have also looked at some of the tooling which allows us to understand complexity and emergence as well as discussed some of the difficulties around working with such systems. AChems aim to replicate some of the dynamics of real chemistry and we find that a sub-symbolic approach gives us an expressive system while still being computationally tractable. In the rest of this work we will describe a sub-symbolic system based on this approach.

Chapter 3

Research Hypothesis

The main aim of this work is to construct a system with rich dynamic behaviours. For this we take inspiration from natural systems and build on artificial chemistries, specifically the sub-symbolic approach.

We postulate that we can construct a rich dynamic system through a consistent design approach:

- We design interactions between particles and then explore resulting behaviour.
- The interactions act exclusively on properties of the particles and the environment.
- Properties are derived only from the sub-symbolic representation.
- Properties and interactions are not the result of arbitrary design decisions.

We postulate that using RBNs as a sub-symbolic system will provide us with enough richness to enable the desired design approach.

Chapter 4

Spiky RBN

4.1 Motivation and Objectives

The basis for this work is to create and explore a complex system which draws inspiration from the mechanisms involved in real world chemistry. This system should exhibit properties similar to those observed in chemical systems. It is hoped that this system would also exhibit emergent properties like those observed in real chemistry. For example the emergence of complex structures, self organisation of particles and compositions, and complexity of reaction networks. The created system should show signs of the growth in complexity observed in real chemical systems. In order to achieve this we believe that our system must incorporate a number of mechanisms from chemistry such as reaction kinetics, molecular geometry and the ability for particles to interact in multiple ways and to varying degrees.

We propose that sub-symbolic artificial chemistries are a good conceptual model from which to build our system. In order to develop mechanisms like bonding, reaction kinetics and geometry in our system we require that our basic particles have a number of exploitable properties. This properties should be:

1. Non arbitrary but dependent on the specific particle.

2. Variable as a result of interaction between particles in non trivial ways.

3. Computationally tractable even for large compositions of particles.

In ssAchems, particles are defined as dynamic systems which can be composed. The properties of these dynamic systems depend on the systems themselves so are non arbitrary. Composing the dynamic system changes the properties in non trivial ways which meets requirement 2 above.

The tractability of computing these properties is then dependent on the specific dynamic system used. In RBN-World [35], RBNs are used as the dynamic system. RBNs have a number of different properties both local (liveliness of nodes) and global (size of attractor, number of attractors for a given RBN). These properties are efficiently computable, and RBNs when combinabled result in changes to the properties in non trivial ways. Therefore we aim to use RBNs as particles due to their computational tractability and large number of calculable, non arbitrary properties.

There is no requirement for our system to produce "realistic" behaviour: there is no requirement that the compositions or reaction networks that we observe are plausible in real world chemistry. We define particles which have exploitable properties and mechanism which modify the particles as a function of some of the properties.

4.1.1 Design principles

There are a number of high level design principles which we would like to adhere to when developing our system.

Computational Tractability and Scalability

The dynamic system used to represent a particle should be computationally tractable. Properties of the particle should be computationally scalable with respect to size of the system.

Traceability of properties

No property used by an interaction should be arbitrary, or a mere function of the computational representation (such as taking the first item in a list in arbitrary order). Every property should be a function of the underlying particle dynamics or the environment. This ensures that observable behaviour can be interpreted in terms of the properties of the particles and their underlying structure. Note this does not require interactions to be fully deterministic. Reaction energetics for example are a probabilistic function of the environmental temperature and the properties of the bond being formed/broken.

Non trivial composability of properties

When two particles interact, for example by bonding, the resulting change in their properties should be non trivial. For example, not simply summing properties together, and not simply assigning a new random value.

No arbitrary interactions

The outcome of interactions such as binding should be dependent on the properties of the interacting particles and the environment. No outcome of an interaction should be due to an arbitrary decision.

4.2 Core Model

4.2.1 Atomic Particle

4.2.1.1 RBN

As discussed in the motivations RBNs are used as the core dynamic system from which we derive our particles. The concept of bRBNs used in RBN-World does not fit with our requirements because the location and number of bonding sites is arbitrarily selected. Instead our system uses pure RBNs without any additional

components. RBNs are composed of N nodes, each of which has an associated boolean state and boolean function. Each node has exactly K incoming edges and determines its state by applying its boolean function to its current state and the states of nodes with incoming edges to this node; the Literature Review has a more detailed definition.

The topology of the RBN and the boolean functions associated with each node are randomly defined when the RBN is created and do not change. We place no restrictions on the random initialisations.

In general RBNs have two associated parameters, K defining how many inputs to each node, and N defining how many nodes in an RBN.

In our system we use $K = 2$ because this is classed as the critical region for RBN dynamics. Our requirements state that a composite particle should not lead to a trivial composition of properties, yet at the same time we do not want a composite particle to have completely new random properties. So we do not want the RBN behaviour to be ordered nor do we want it to be chaotic.

The size of the RBNs (N) is also important. Large values of N may lead to long transient states, meaning it takes a long time to compute the dynamics. Since our atomic RBNs will be combined to make composite particles, the size of the composite RBN will grow linearly with the size of the particle. So, we would like relatively small atomic particles (small N) so that we can comfortably combine into large particles without a substantial slow down in computation. However because of our definition of Interaction Lists for bonding (discussed below) we also require that N to be large enough to provide a variety of different structures. We choose $N = 12$ for our atomic RBNs. This provides a relatively small atomic particle, meaning it is easy to calculate, while still allowing for a variety of Interaction List sizes which allows for a variety of possible number of bonds.

So our atomic particles are composed of RBNs with $K = 2$, $N = 12$.

Data: N: the set of all nodes in the RBN
while N *is not empty* **do**
    Remove a node n from N;
    Create new Interaction List IL_i;
    Add n to IL_i ;
    while $\exists n' \in N$ *where* n *is an input to* n' **do**
        Remove n' from N;
        Add n' to IL_i;
        $n \leftarrow n'$;
    end
    i++ ;
end

Algorithm 1: Building Interaction Lists; arbitrary choice of start

4.2.1.2 Interaction List

One of our core requirements is that all particle properties are a function of the underlying RBN properties. This requirement is not compatible with the RBN-World concept of bonding sites. bRBNs have exactly two bonding sites arbitrarily connected to the RBN. In our system the number of possible bonds a particle is capable of and how they effect the particle should not be randomly determined. Instead those particle properties should derive in some way from the RBN properties.

In order to achieve this we segment the RBN into units capable of forming a bond. These are referred to as Interaction Lists (IL). Each IL is a list of RBN nodes where each subsequent node has a direct input from the previous node in the IL except the first node for which input edges are not specified. An RBN node is part of exactly one IL and every RBN node is part of an IL.

By this construction a RBN now has a number of IL each consisting of a number of nodes. The number of ILs and their size are fully dependent on the topology of the RBN. So overall our particle has a number of possible bonding sites each of which is distinct. Different topologies will lead to different numbers of bonding sites.

We can construct such ILs according to Algorithm 1. However, this algorithm still has aan arbitrary component, specifically the choice of first node and the choice between possible subsequent nodes (since node n can output to an arbitrary number

of other nodes). This is contrary to our design principles and can lead to problems. In this case one particular issue that could arise is that two otherwise identical RBNs could be segmented into different IL, therefore having different properties. This seems conceptually wrong: two identical particles should behave in the same way.

To deal with this we need to select n based on some property of the nodes. Two properties are suggested, influence and liveliness. Liveliness is defined as the number of timesteps that a node is in the "True" state over the attractor cycle. Influence is defined as the number of outgoing edges a node has: the more outgoing edges the higher the influence. Liveliness is a property of the RBN dynamics, and influence is a property of the RBN topology.

In each case we can choose most or least lively or influential node to be first in the IL, and choose between the possible next nodes using the same criteria.

We explore how the possible choices affects the mean size of the largest IL in each RBN, Figure 4.1; we also record the largest observed interaction list across the 10,000 generated RBNs. The first observation is that ILs of maximum size are possible: we can find particles with exactly one IL containing all $N = 12$ nodes. This occurs when we pick least influential or most live first. In the other cases we still see very large IL of size 11 are possible. The mean number of nodes per largest IL suggest that picking least influential tends to create larger IL sizes then picking most live.

We also explore how the mean number of ILs per RBN changes depending on the picking criterion, Figure 4.2. Here we observe that the choice of least influential has by far the lowest number of ILs on average. This combined with the high average max IL size suggest that least influential tends to create one large IL and two smaller ones. Most influential has the highest average number of IL per RBN with around 6. Since the mean number of nodes per max IL is around 6 this suggest that on average RBNs have 1 large IL of size 6 with 5 more IL of size 1 or 2.

One of the persistent issues with creating this system has been in defining what constitute desirable properties. In this case should ILs be small and numerous or

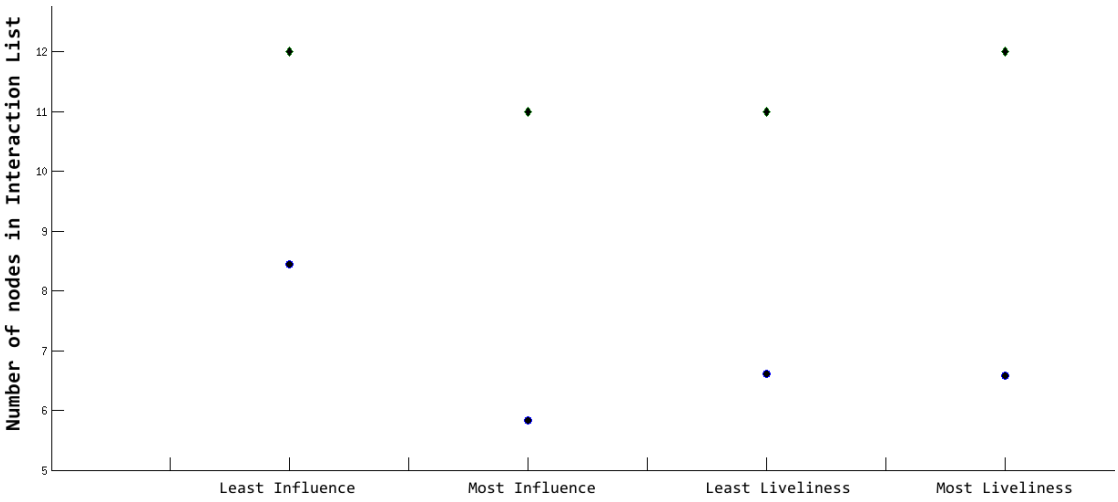


Figure 4.1: Mean and maximum IL size of largest IL, observed under different construction scenarios: least influence, most influence, least liveliness, most liveliness

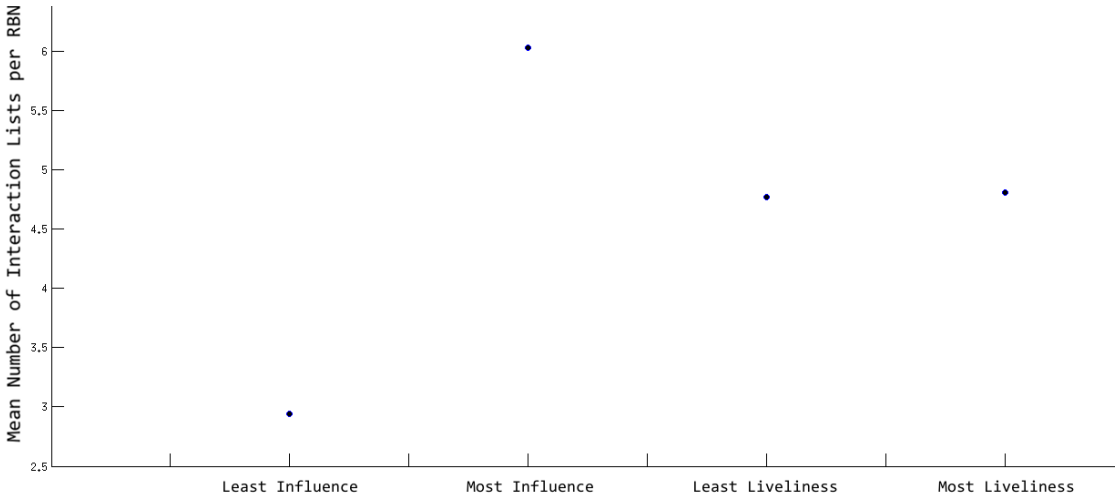


Figure 4.2: Average number of interaction lists per RBN under different construction scenarios: least influence, most influence, least liveliness, most liveliness

Data: N: list of all nodes in the RBN ordered by least influential first
while N *is not empty* **do**
    Remove first node n from N;
    Create new Interaction List IL_i;
    Add n to IL_i ;
    while $\exists\ n' \in N$ *where* n *is an input to* n' **do**
        Remove n' from N;
        Add n' to IL_i;
        $n \leftarrow n'$;
    end
    i++ ;
end

Algorithm 2: Building Interaction Lists; non-arbitrary choice of start

large and fewer? It is difficult to make an informed decision on what effect each choice will have without going through a long exploration process.

We can in some ways reason about what the different choices mean for our bonding mechanism (described below). Because bonding results in swapping edges between ILs the larger the ILs involved in the bond the more likely the bond is to result in a change to system behaviour (since more edges will be swapped and the system will be more perturbed). So fewer larger ILs mean particles are more likely to change behaviour as a result of bonding.

Another consideration is that the liveliness property is a function of RBN dynamics while the influence property is a function of topology. The IL is the connection between the RBN topology and the particle's topology which means that conceptually it makes sense to use a topological property.

Using liveliness brings up another issue. When a particle bonds the dynamics can change; the liveliness used to construct the ILs could now be different. In order to be consistent then we would have to recalculate the ILs after each bonding. This recalculation could result in the IL involved in the bond no longer existing, which then brings up the question of whether we keep or dissolve the bond.

Using influence bypasses this issue: the topology of the particles does not change outside of the ILs involved in the bond.

Based on these argument, we use the least influential criterion, meaning we construct ILs using the refined Algorithm 2. We now have non-arbitrary bonding sites based on an underlying topological property of the RBNs.

4.2.1.3 Spikes: The Bonding Property

In order to bond particles together we need a particle property that we can use to determine if a bond is possible.

The formation of a bond should result in a change to the behaviour of the reactant particles. By constructing ILs using influence rather then liveliness, we have ensured they will not change as a result of bonding. The bonding property should be a function of the RBN dynamics, which may change as a result of bonding.

We would also like the bonding property to be localised to the specific ILs involved in the bond. In other words bonds between particles should only be possible on specific ILs.

To meet the above two criteria the bonding property cannot be the cycle or transient length of the RBN since those are global properties of the RBN.

Instead we first considered the mean or sum of node liveliness (defined as the number of times a node is a"True" state over the cycle). This provides a positive integer value bounded by nc, where n is the number of nodes in the IL and c is the cycle length. If we used mean node liveliness then the value is bounded by c.

The bonding property should be expressive enough to allow for a variety of different interactions, not just bonding. One thing in particular that we wish the bonding property to be able to do is to allow particles to behave like dipoles, or more generally to allow for attractive or repulsive behaviour. (While the work has not progressed to the point where we make use of this, at design time it was a requirement we wished to have.)

So our bonding property needs to include not only a magnitude but also a sign.

The bonding property value is calculated over the attractor cycle as follows.

The value of node x at RBN state s, $x_{s_{value}}$ is 1 if it is in a 'true' state and -1 if it is in a 'false' state.

$$x_{s_{value}} = \begin{cases} 1, & \text{if } x_{s_{state}} = T \\ -1, & \text{if } x_{s_{state}} = F \end{cases} \tag{4.1}$$

The value of node x over one cycle of the attractor x_{value} is

$$x_{value} = \sum_{s=t}^{s=t+c} x_{s_{value}} \tag{4.2}$$

where t is the first state of the attractor and c is the attractor length.

The bonding property for IL_1 of particle A, S_{A1}, is the sum of all node values for nodes in that IL.

$$S_{A1} = \sum_{x \in IL_1} x_{value} \tag{4.3}$$

This gives us a property with both a magnitude and sign. It is constrained by the attractor length c and the number of nodes in IL, $IL_{A1_{size}}$:

$$-IL_{A1_{size}}c \leq S_{A1} \leq IL_{A1_{size}}c \tag{4.4}$$

We calculate the attractor of a RBN from an initial state of all 'false'.

We refer to this bonding property as a *Spike*, due to the diagrammatic representation that we use, Figure 4.3.

4.2.2 Linking Algorithm

The last component of the core model is the core interaction we wish particles to be able to perform: linking. This is the equivalent of bond formation in chemistry. We define the requirement for our linking algorithm.

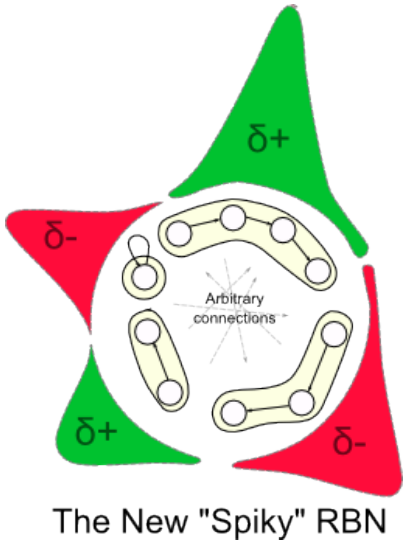


Figure 4.3: Diagram of the Spiky RBN. Nodes are separated into Interaction Lists (ILs) of varying size based on Algorithm 2. Edges in solid lines are those which are part of the IL and can be used in linking. Dashed line edges are part of the RBN topology that cannot be changed due to linking. Note that there is an IL with only one node and no edges; it cannot form a link. Each IL's spike can be seen, with colour denoting sign and size denoting magnitude.

- A link should be able to form between two ILs if a condition over their Spikes is met
- Linking should result in a change in dynamics of constituent RBNs
- Links should have a concept of stability
- Unstable Links should be broken
- Past Linking should not change atomic behaviour

The last requirement is added to ensure that linking does not have a lingering effect. That is, two identical atomic particles should behave identically, irrespective of how many and what types of links they have previously had.

Much like with the original b-RBNs, we construct links by changing the RBN topology (swapping node inputs).

The b-RBN model used the special binding node to allow RBNs to be composed. We could introduce binding nodes, creating one per IL and connecting it to a node

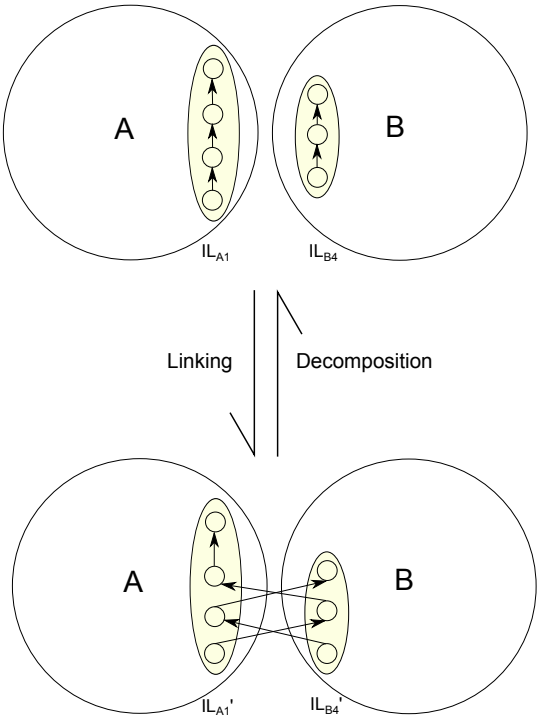


Figure 4.4: An example of the structure of a link between particles A and B. Note that two pairs of edges are swapped since IL_{B4} has a size of 3.

within the IL. The choice of where to connect it is again arbitrary so some method of picking a node would have to be found. More importantly the binding node mechanism means that a link is a very binary property (it either exists or does not).

Instead, we would like a link to be a more expressive concept: there should be stronger and weaker links, as well as possible multiple links. To allow for this, we use a different method for composing RBNs. A link between interaction lists of different particles is created by swapping pairs of edges between equivalent nodes in the IL. Here equivalent means position within the IL: the edge between first and second node on interaction list A is swapped with the edge between first and second node in interaction list B, and so on, for the length of the shorter IL (figure 4.4).

There is a difference between the first and last node in an interaction list in terms of how changing their input can affect the RBN. The last node in the list can be said to be most susceptible to changes in the IL. This is because swapping any link prior to it in the IL can result in the propagation of change to that node. Inversely the

first node in the list is the one least likely to change its behaviour since its inputs are from the rest of the RBN.

4.2.2.1 Stability and Collision Criterion

Before a link can be formed we need to provide a criterion for linking so that not all possible spikes can produce a link. We also need a criterion for stability, so that links can be broken. The collision criterion dictates what must be true in order for a link to form. The stability criterion dictates what must be true in order for a link to continue to exist.

A link can form between ILs chosen from two different particles. The ILs must have size > 1 in order for a link to be formed by rewiring. How candidate particles and ILs are chosen depends on the reactor type. In an aspatial well-mixed reactor two random particles can be chosen and a random IL on each. In a 2D lattice reactor particles in adjacent sites and the nearest ILs can be chosen.

The collision criterion states that:

$$S_{iA} + S_{jB} = 0 \tag{4.5}$$

where S_{iA} is the spike of the i^{th} IL of particle A. So equal and opposite spikes can form links. If the collision criterion is not met, the collision is considered elastic and the two particles do not form a link. If the collision criterion is met then a link forms as described below. Link formation results in a change in RBN topology and consequently in a possible change to linking properties. After the link construction all linking properties are recalculated and used to check against the stability criterion.

Like collision criterion, the stability criterion states that:

$$S'_{iA} + S'_{jB} = 0 \tag{4.6}$$

where S'_{iA} is the spike of the i^{th} IL of particle A after the link has been formed.

So equal and opposite spikes are stable (spike values can change on linking). The stability criterion is checked not only for the newly formed link, but for every pair of ILs that are part of a link in the new composite particle. Decomposition results in a particle splitting into two or more fragments. Each IL that was part of a now broken link is free again. This means the topology has changed such that stability criterion is checked recursively until all links in all fragments meet the criterion. Because the stability criterion is checked for all links it holds true for every composite particle that is not currently attempting to link.

4.2.2.2 Link Structure and Formation

If the collision criterion holds then a link is formed. This is done by swapping pairs of nodes inputs between the two ILs as shown in Figure 4.4. Edges which are part of the IL are swapped, starting with the edge outputting from the first node in the IL.

The maximum number of swaps possible is $n - 1$, where n is the size of the smaller of the two ILs.

A link can be constructed between any two ILs of size > 1. ILs of size 1 cannot link because they have no edges to swap. (It is possible to have a node that takes input from itself. We do not consider these.)

After link formation the spikes of all ILs in the new composite particle are recalculated, since link formation results in a change in the underlying RBN topology. For any links that do not meet the stability criterion the link is decomposed by reversing the swaps. When a link decomposes the break results in two new particles; again the spikes of the ILs are recalculated, and any further decomposition needed is performed. This process continues until the products are stable (see the stability criterion), hence a single interaction between two reactants can result in multiple product particles. The algorithm for two atomic particles is shown in Figure 4.5.

Our links have a richer structure than those in the original RBN-World [35]. Small

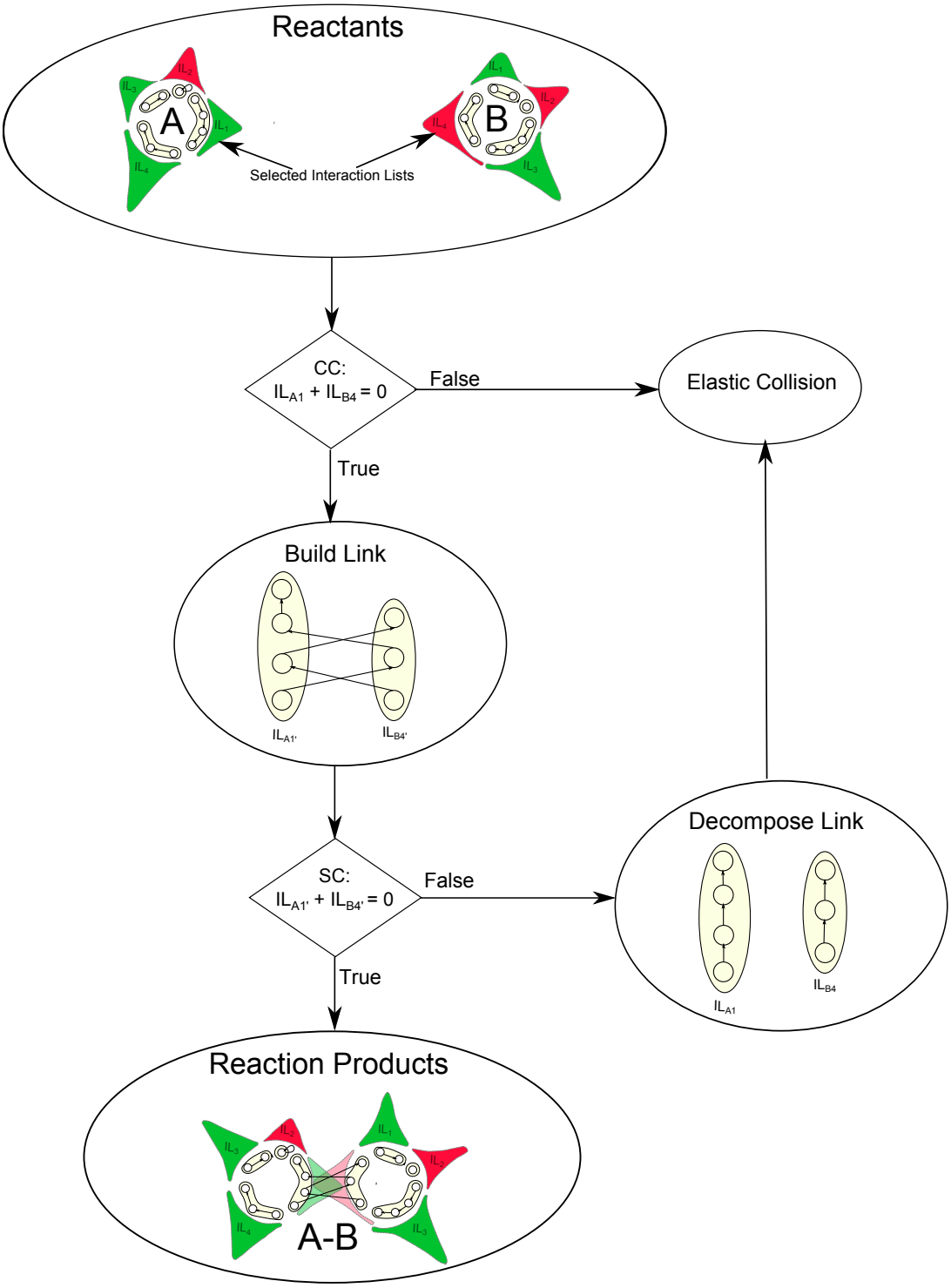


Figure 4.5: Reaction algorithm between two atomic particles. First an IL is selected on each. The collision criterion is checked. If it passes then a link is formed (producing particle AB). The stability criterion is checked for the ILs that are part of the link. If it fails then the link decomposes. Note that ILs not involved in links may have had their spikes changed due the effect of the link on the new composite RBN.

ILs mean fewer swaps to form a link, resulting in less perturbation to the linking particles. This implies a higher chance that the spikes do not change and the link is stable. Larger ILs produce a larger change in topology and are therefore more likely to result in different spike values and so link instability.

4.3 Generating Atomic Sets

Exploring this new "Spiky RBN" system in terms of behaviour is quite challenging. There is a huge search space of possible atomic particles (every possible RBN with $K = 2, N = 12$) and from this, a huge combinatoric explosion of possible composite particles which we need to look through for interesting dynamic behaviour.

An exhaustive search is not feasible. Even when selecting a small subset of atomic particles it is not practical to test all possible links.

In order to reason about the system, especially when adding components to the core model such as in the following chapters, we need a reference set of particles which to use. These particles should represent a range of possible system behaviours and act as a reference set which we can use to explore different extensions to the core model.

We could select these sets of particles by hand or randomly generate them. However we choose to generate them by using an evolutionary algorithm, growing the atomic sets to meet some fitness function. One nice property of an evolutionary approach is that at each generation there is a population of candidates which are themselves internally ranked. This fits our requirements quite well since we do not want one single set of atomic particles but a range with hopefully different behaviours.

4.3.1 Reactor

For this experiment we use an aspatial reactor initialised with 20 unique atomic particles. The reactor attempts 1,000 links by picking two particles at random,

picking an IL on each at random, and attempting a link. If the reaction is successful all reactants *and* products are added to the reactor.

At any time the reactor contains one copy of each composite particle that has been generated so far, plus the initial 20 atomic particles. In effect our system is a well stirred reactor with equal concentrations of all particles.

This reactor strategy is chosen because our interest is in the range of possible composites and reactions the atomic set can exhibit. The reactor will bias towards making large composite particles since as soon as a large particle is discovered it has an equal probability of being picked for reaction.

4.3.2 Defining a Fitness

In order to generate the atomic sets we are after, we need some definition of a fitness function describing the kinds of behaviour we would like our atomic sets to produce.

Defining this is pretty tricky since we are unsure of exactly what range of behaviours the system is capable of, and we do not have a good definition of what we would like to observe. In fact, we do not want to impose a strong fitness on the system since the generated atomic sets will then only show the specific behaviour we envisioned. Instead we want to use this as a way of exploring the range of behaviours.

While it is difficult to state specifically what we do want we can state undesirable characteristics:

- Overly restrictive: most reactions are elastic and do not result in larger composite particles. The system quickly reaches a stable state with no reactions occurring.

- Overly permissive: almost all reactions result in stable links. The system quickly congeals into a single large composite particle.

- Chaotic: almost all reactions between composite particles result in decomposition of links. The system is reactive but larger composites are quickly destroyed.

Our fitness function is based on trying to avoid the above undesirable characteristics. There are five measures on which we base our fitness function:

C**:** number of unique composite particles that the system creates after a set number of reaction attempts

V**:** variance in observed composite particle size

L**:** variance in number of links per atomic particle in a composite particle

R**:** number attempted reactions for which the new bond is stable

P**:** number of unique links that an atomic particle has been observed as forming. For example if atomic particle A has only ever formed bonds with atomic particle B and itself, then $P = 2$.

These characteristics are rather permissive. By using variances for V and L we ensure that the system will not bias too much towards large composite particles with very dense branching. We none the less want the system to be reactive and produce a large number of composites (C) and we want to ensure that the composites are stable (R). The last characteristic gives a way of ensuring that composites are not just long chain polymers of simple monomers ie that most particles are involved in some reaction or other.

These characteristics form the basis of our fitness function. C, V, L and R provide an overall fitness for our seed set. P provides an individual fitness for each atomic particle within the seed set.

One issue with this kind of multi variable fitness is that generally a good set of weight parameters is required to balance one against another. This is especially important since the characteristics we describe have very different possible value ranges. Here

we could use Pareto optimization to find a good fitness function. However at this point the aim is to find a range of reactor variants and not specifically to find ''optimal'' atomic sets so there is no reason to spend the time going through a full analysis.

We bypass this by using a ranking system to generate an overall fitness value. By ranking members of the population based on each of these characteristics we can bring the values into a range. This is a crude way of combining the fitnesses, however it is effective in ensuring that each characteristic is equally important. Adding weights to this would allow us to bias the fitness function towards one characteristic or another. However, since we do not have a strong definition of a perfect system we cannot make an informed decision about what we should bias, so we do not.

Overall we define a fitness function for a reactor and for a particle within a reactor.

The fitness of reactor i is:

$$fr_i = Rank(C_i) + Rank(V_i) + Rank(L_i) + Rank(R_i) \tag{4.7}$$

where $Rank(C_i)$ is the rank of reactor i when the reactors are ordered by lowest to highest. Since our population is made of 20 individuals and there are four ranks, fr_i is constrained to

$$4 \leq fr_i \leq 80 \tag{4.8}$$

The fitness of atomic particle j in reactor i is the number of unique bonds it can form, P_{ij}

$$f_{ij} = P_{ij} \tag{4.9}$$

With a seed set of 20 atomic particles f_{ij} is constrained to

$$0 \leq f_{ij} \leq 20 \tag{4.10}$$

The mutation function replaces the atomic particle j in reactor i with a new random

one with a probability proportional to M_{ij}, where

$$M_{ij} = \frac{84 - fr_i}{1 + f_{ij}} \tag{4.11}$$

That is, fitter reactors and fitter particles are mutated less.

$$0.19 < M_{ij} \leq 80 \tag{4.12}$$

4.3.3 Experiment: Generating atomic sets

To generate atomic particle sets we use an algorithm similar to clonal selection [20, 21] (see Figure 4.6). Our population is made of 20 reactors, each reactor containing 20 unique atomic particles. Unlike normal clonal selection, each population member produces exactly one clone by mutating atomic particles based on M_{ij}. This is because our aim is not specific optimisation but rather exploration for possible seed sets with favourable behaviours. In order to ensure that good seeds propagate through the generations we include a low 5% crossover chance. The crossover function replaces the three lowest participation particles in a set with the three highest participation particles from another set. When crossover occurs we ensure that the resultant seed set has 20 unique atomic particles by making sure the incoming particles are not already in the seed set. The crossover probability is kept low because again we are interested in diversity. Also, due to the nature of the system, high fitness of a particle in one reactor does not necessarily imply high fitness in another reactor. This is a desirable trait since high fitness in all reactors would suggest the particle is overly permissive and can bond with almost everything.

4.3.4 Exploratory Algorithm Performance

The exploration run produces the desired results: a range of atomic sets which show varying reactivity and particle composition.

Figure 4.7 shows how the values of C, V, L and R change over the generations. Over

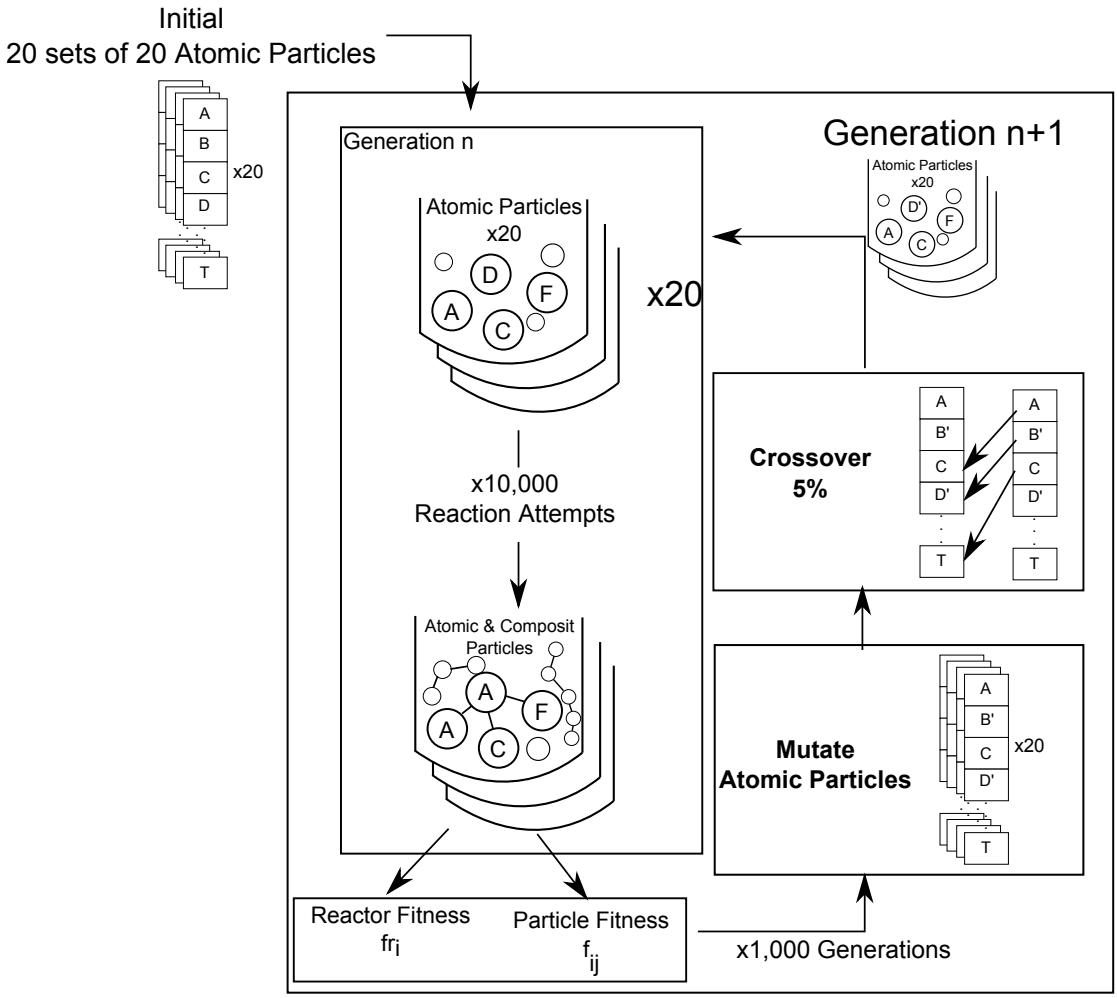


Figure 4.6: Exploration algorithm. The algorithm is initialized with 20 sets of 20 atomic particles each. Each reactor then attempts 1,000 reactions. We then calculate reactor fitness and particle fitness, mutate and perform crossover to get the new population for gen $n + 1$. This repeats for 100 generations.

the generations there is an increase in the distributions' upper quartiles, suggesting some reactors are improving and finding better seed sets.

The median values in each graph fluctuate without showing a strong positive trend. This is to be expected: even if no mutation or crossover is experienced there is no guarantee that a successful reactor will reproduce its behaviour in the following generation. One reason for this is that reactions are randomly chosen, it is possible that a very reactive composite particle is not generated even if one is possible.

The median variance in size (Figure 4.7b) stays very low for most of the experiment. This is due to reactors producing only composite particles of one size (most commonly size 2) giving a low variance score. These are almost always reactors which are not very active. Our reaction algorithm can only compose two particles, which at the beginning of the run means atomic particles reacting to produce particles of size 2. Low variance suggests that particles bigger then that are not found so the reactors quickly reach a 'dead' state. Because each reactor runs for a set number of reaction attempts it is also less likely that a highly reactive product was missed in these cases.

Generation 68 shows a large increase in variance of particle size (Figure 4.7b) compared to the previous generation. For that generation there is also an increase in median number of unique particles (Figure 4.7a), and number of bonds formed (Figure 4.7d) compared to the previous generation. The median variance in number of bonds per particle (Figure 4.7c) is lower then the previous generation however.

The reactor metrics give us a good idea about the dynamics in that generation as well as showing how the metrics involved in the fitness interact. C, V and R are all strongly linked and are likely to all point in the same direction. Discovering more unique particles requires the system to be able to form bigger particles (after exhausting all possibilities with smaller particles) which also directly requires more links per particle.

We observe that this generation suffers from a lack of branching (indicated by the low values of Figure 4.7c). From the perspective of fitness score the low branching

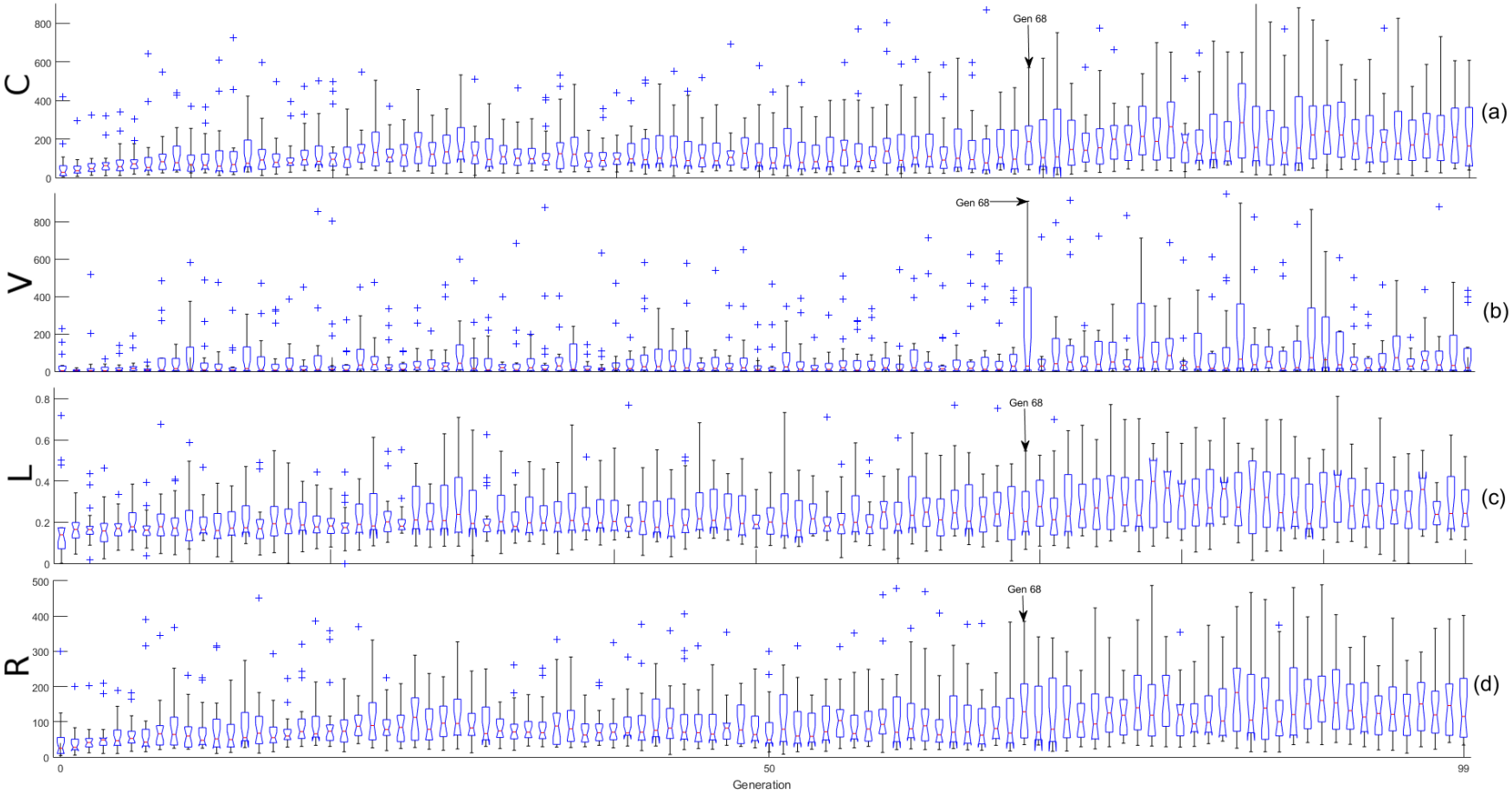


Figure 4.7: Distribution of the reactor measures per generation; some outliers are omitted. (a) C, number of unique particles; (b) V, variance of particle size; (c) L, variance of number of links; (d) R, stability. Generation 68 is highlighted for reference.

acts in opposition to the other three metrics in this case. However that does not have to be generally true, it is simply the case that the branching metric is not as strongly correlated with the others. This illustrates why the branching metric is used. Our interest is in diverse behaviour, not just in terms of reactivity, but also in terms of geometry. Generation 68 produced a large number of unique particles which are, for the most part, straight ribbons of repeating particles with low branching.

The large number of outliers shows that while these low-reactivity systems are common we can also find more interesting examples. Because of the interdependence of our metrics it is highly unlikely that a reactor can produce an outlier in only one metric meaning we are not producing reactors which are specialising in individual components of the fitness.

The reduction in outlier numbers towards the end of the run together with an increase in distribution variance suggests that fit individuals have a positive influence on the population. However we do not see a strong fitness improvement over the generations. This points to the 'fragility' of the system: high fitness is not easily transferred to another member of the population simply by taking particles that contributed. In other words the dynamics of the system are not easy to replicate.

In order to validate that the exploratory algorithm is producing positive evolutionary activity we use the quantitative non-neutral (QNN) measure of evolutionary activity [28]. Figure 4.8 shows the QNN distribution per reactor over the generations. Large increases in the QNN distribution, especially towards the latter third of the experiment, suggest a period of strong evolutionary activity for most reactors. Most likely this is a result of a positive mutation that enables a wide range of new reactions substantially increasing the fitness of a reactor in a short time. Because mutation is more likely to occur in less fit particles within a reactor we can think of it as a mechanism for finding new particles that complement the dynamics already present in the reactor. If the reactor can produce large particles using 3 atomic particles and mutation introduces a new atomic particle which can interact with any of the products then the fitness will jump substantially. The crossover also provides an interesting component to this. If the set of particles being crossed over can produce

a very large number of reactions just between themselves you will see a substantial jump in fitness. Essentially cross over provides us with a mechanism of introducing a minimum set of reactive particles so that mutation can then attempt to add other atomic particles that compliment the dynamics of that set.

So we have produced atomic sets which are fit for our purpose, showcasing a range of the dynamic behaviours our system can generate. However the experiment shows the difficulty that such behavioural range brings.

4.3.5 Atomic sets

Table 4.1 shows the reactor metrics for each reactor at the end of the last generation. In the final run four of the 20 reactors produced over 500 unique composite particles. Of these the best (based on f_r) was reactor 3 producing 586 particles of which 307 had newly created stable bonds. Reactor 8 is a close second producing less particles with a lower variance in links but with a higher variance in particle size and higher bond stability. Mostly these large reactors produce one type of particle, generally different combinations of a small set of atomic particles which can react to generate a variety of particle sizes and geometries.

Figure 4.9 shows a graphical representation of the reaction networks constructed using graph-tool [85]. Particles are represented by red circles and reactions by green squares. Particles are arranged vertically by size so the top row of particle represented the atomic set, the third row represents composites of size 3 and so on. Figure 4.9 shows a range of reaction networks observed, from very active on the left to very sparse on the right.

Figure 4.10 shows one of the larger generated composite particles. The main chain consists of T and K atomic particles. We see branching along the chain showing that the T atom is capable of three links. We also see that atom L allows other, non-K or -T atoms to join the chain (specifically F, M and S). This gives the product compositional diversity. Like the T atom, L is capable of forming up to three links.

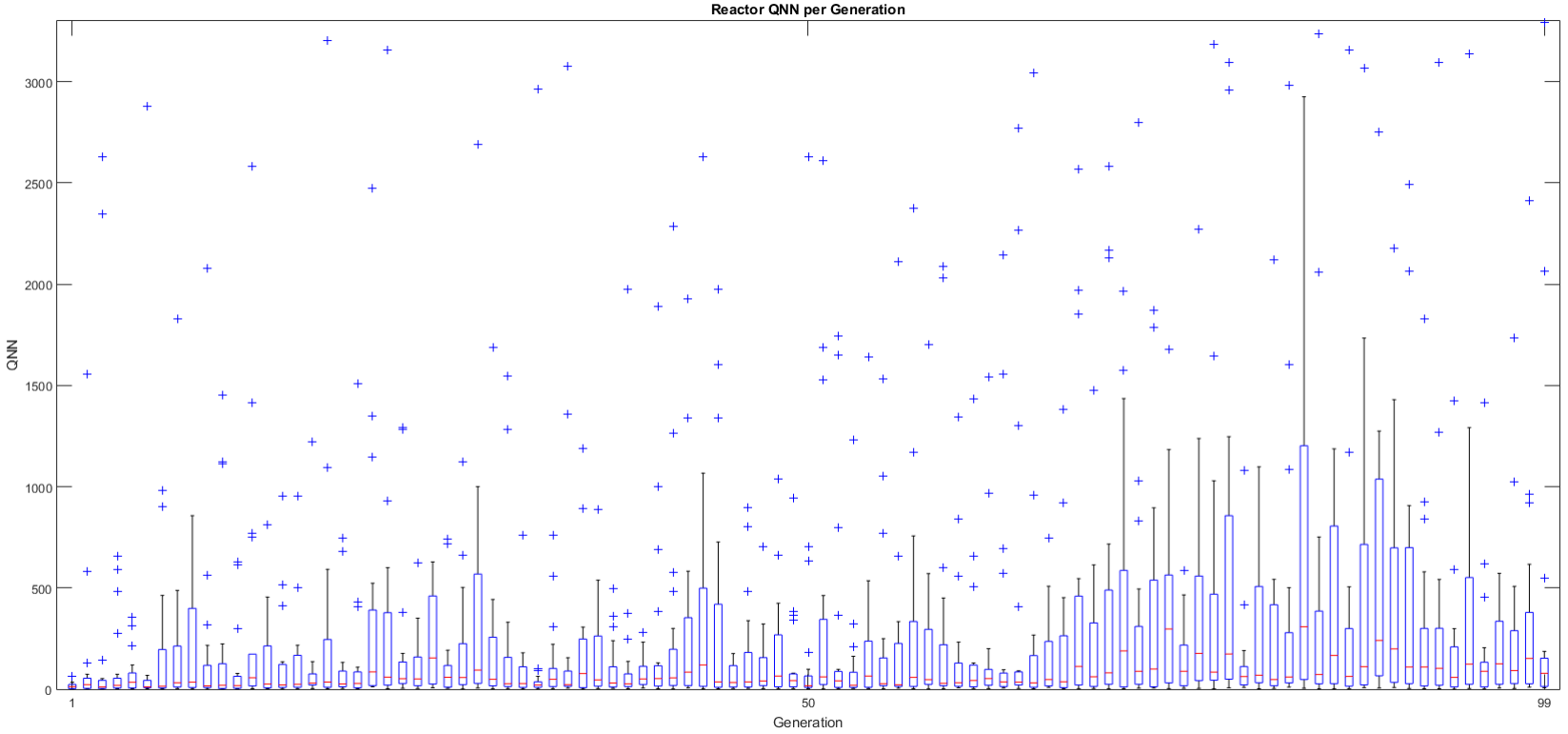


Figure 4.8: Distribution of the evolutionary activity measure QNN per generation for each reactor.

Reactor	C	V	L	R	fr
0	46	2.3667	0.18255	40	11
1	63	4.3401	0.19970	59	22
2	129	8.3091	0.20915	83	31
3	586	430.3	0.51866	307	72
4	76	7.6058	0.17437	58	21
5	229	21.694	0.38975	137	50
6	59	1.8248	0.17602	54	12
7	139	6.8823	0.27666	116	34
8	501	1607.6	0.36968	402	70
9	453	128.54	0.27755	298	57
10	149	16.221	0.20390	93	34
11	39	2.5063	0.12235	34	4
12	566	116.41	0.36923	280	61
13	274	398.26	0.11454	166	45
14	608	368.44	0.29997	338	66
15	266	31.143	0.28607	158	50
16	183	25.329	0.34497	140	48
17	58	4.797	0.18159	45	16
18	179	61.056	0.48978	114	50
19	50	2.2724	0.12439	38	6

Table 4.1: Gen 99 Reactors

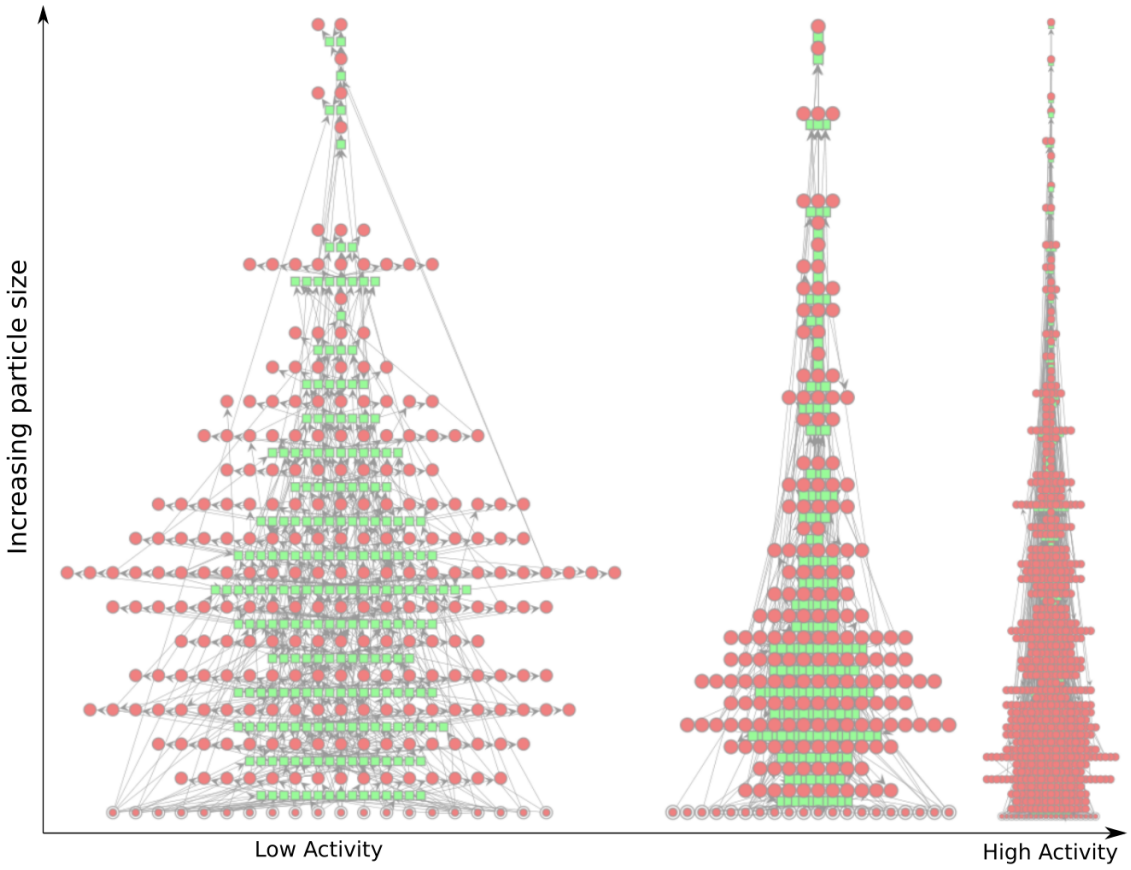


Figure 4.9: A graphical representation of some of the reaction networks. Most reactive on the left and least reactive on the right.

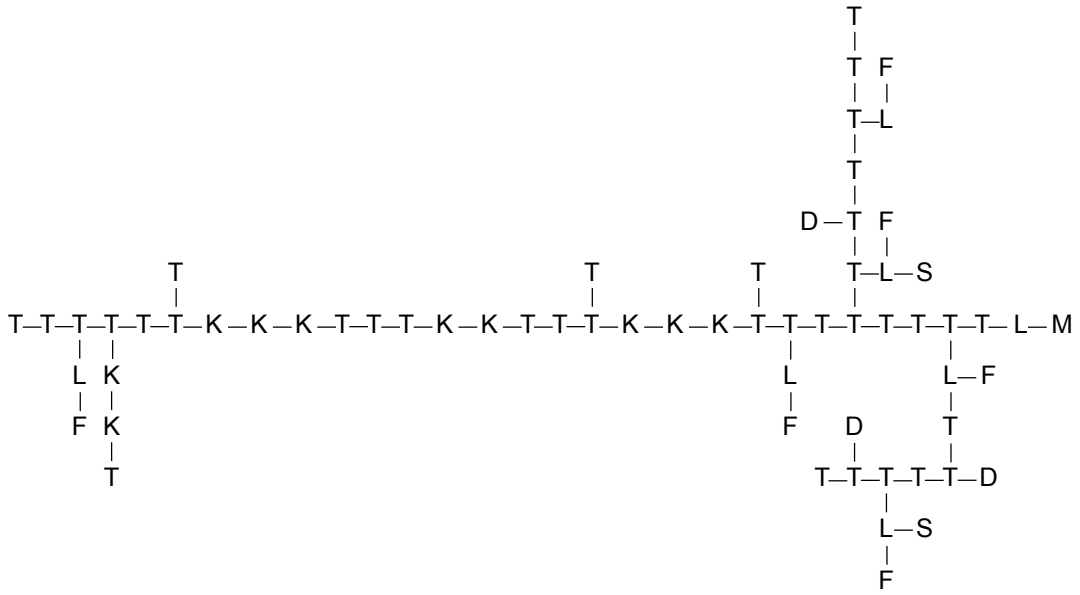


Figure 4.10: A large composite particle produced by reactor 3, made of 65 atomic particles from 7 different species (T,K,L,S,F,M,D).

However the reactor does not contain any long chain L composites, suggesting that L-L links are unstable. While the particle in Figure 4.10 is stable there are still T atoms with only one link, suggesting that it could grow even further. The L atom also seems to be able to participate in 3 distinct types of bonds: T-L-SF, T-L-M and T-L-FT.

This particle is an example of exactly the kinds of structures we hoped to find. It is not composed of a random collection of all atoms, meaning bonding is not overly permissive. Yet it shows variation to its structure due to the addition of specific substructures. We can see the beginning of functional structures emerging. K-T particles form long "spines" that are capable of branching due to the T atom. The T atom can also be terminated by D or L atoms. While the D atom seem not to be reactive, the L atoms can lead to a variety of end structures. A closer look at the reactions shows that L atoms are unable to bind to the K-T chains; instead, a previous reaction is required in order to push the L particle into a reactive state, in this case L-F or L-M reactions.

This particle is a product of 47 unique reactions. Most produce exactly one new unique composite particle. For the most part these are reactions that combine two

shorter T-K chains together. However three of the reactions produce two unique particles meaning splitting into previously unseen substructures. Again these are mostly structural variations on T-K chains with different placement of branches and L/D particles. The reactor does not count non-unique products; a reaction that produces one new structure and one existing one would record only the one unique structure. Figure 4.11 shows a selection of other particles produced by the reactors; these plots are generated using graph-tool[85]. We observe various types of particle structure, in terms of number of constituent atomic particles, organisation, size, and degree of branching.

Branching is common in the final reactors: 15 out of 20 reactors have at least one composite where an atom has three links. All reactors that produce a wealth of composites show similar branching to the one we see in Figure 4.10. However, we have not observed a particle that can form four or more links. Most likely this is a consequence of the size of atomic particles and the way Interaction Lists are constructed, which tends towards 2 or 3 ILs capable of bonding. It is possible that changing the way ILs are constructed to ensure that each particle has at least 4 linking sites of size > 2 would give more branching. However, engineering particles in such a way is contrary to the core principle of having emergent properties. A more consistent approach would be to find naturally occurring particles of that nature and introduce them into the seed set, or possibly to increase the size of the atomic particles so that the existing IL generation algorithm is more likely to produce more ILs.

4.3.6 Evaluating Reactor

We use a very "unnatural" reactor type for these experiments. It has no concept of mass conservation and considers that all particles observed exist in equal quantity, meaning concentration are equal.

This has an effect on the reaction dynamics of the system. In general this reactor biasses towards large composite particles of one species (variations on the same

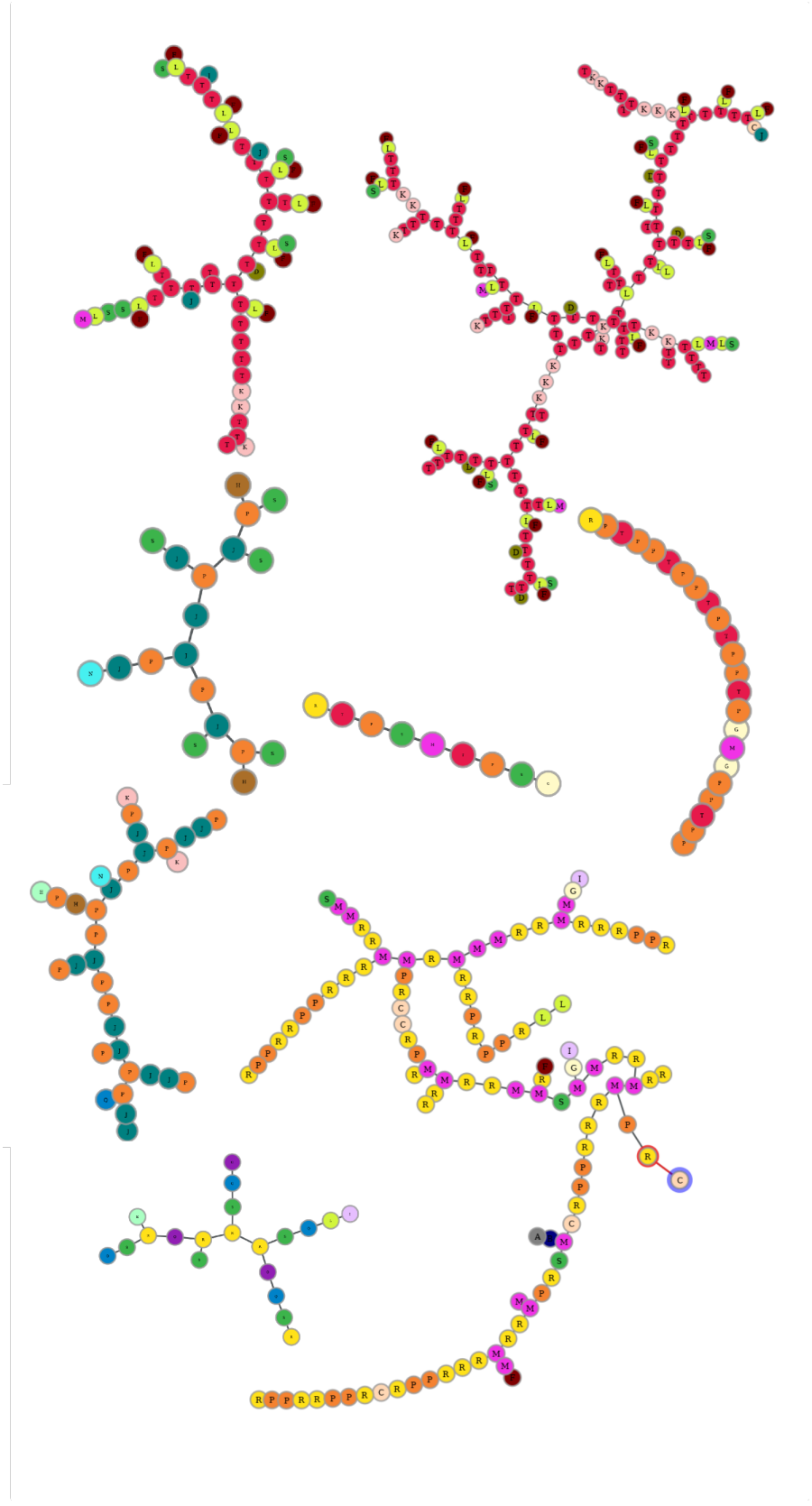


Figure 4.11: A selection of other observed particles. Generated with graph-tool [85]

general structure and composition). This is a result of the equal concentrations. If one reactive set is found, geometric variations of it will quickly come to dominate the overall concentration of particles. Particles which are readily reactive produce new larger particles which bias the selection probability towards that particle species. The probability of attempting reactions with atomic particles decreases substantially as the reactor discovers new composites of the easy to bond type. Overall the reactor is likely to explore the variation within that one composite type as oppose to the variation in composite types.

In a system which conserves mass, concentration will play a very different role. New large composites will still have a low probability of participating in a reaction since they exist in very low concentration, so the system will tend towards exploring small particle dynamics more.

4.3.7 Conclusion

We have described a basic model of a sub-symbolic artificial chemistry. The ssAChem consists of a definition of a particle which derives its structure and dynamics from an underlying system, in our case a random boolean network. We show how particles can be composed in a way that allows them to effect each others properties and results in new properties (those defined within the bond). We also propose the rules that govern this composition as our linking algorithm. This algorithm allows particles to form and break their interaction with each other based purely on the properties that derive from our particle definition.

We have generated a set of particles aiming for behavioural variance in order to further explore the dynamics of our system. We observe that the linking algorithm is capable of producing particles of various size and with strong internal structures.

The reactor rules we use for this exploration are highly permissive; they ignore concentration and mass conservation as well as other mechanisms we might wish to introduce. Moving forward we aim to see the generated atomic sets behave under

different, more realistic constraints and with the introduction of new mechanisms.

Chapter 5

Analysis methodology and reactor variants

5.1 Analysis Methodology

One of the aims of this work is to build on the core model and explore how different extensions and constraints affect the system behaviour. The previous chapter covers how we generate atomic particle sets to use as the basis for this comparison. Alongside this baseline we also need a structured way to analyse the system.

Analysing and quantifying the effects of different rules on the observed systems can be challenging. In AChems with defined reaction rules there is often a desired behaviour that the rules aim to produce, so analysis can be geared towards finding that behaviour. With the Spiky-RBN Chemistry, the aim is to explore the dynamics and activity of ssAChems and how they are affected by different rules.

One approach to analysis is quantifying the complexity of the system. There are multiple definitions of complexity, and from there, multiple approaches to measuring it. Within our system we could look at the complexity of the RBNs [16, 109], or of the composite particle geometries [87, 54]. We are interested in the reaction networks our system produces. There are metrics for chemical reactions specifically

[54, 12], and for graph complexity [22]. However, we want our analysis tooling not only to provide a numeric quantity, but also to allow us to analyse and reason about the system behaviour. Therefore, we move away from these more generic complexity measures, and define our own metric, which is designed to accommodate the reaction processes of our Spiky-RBN system, while still being based on core concepts of complexity.

Overall we can say that our tooling should:

- Help identify components in the system that might have interesting behaviour
- Help in comparing different instances within the context of the system
- Allow us to differentiate between uninteresting and interesting reactions

The final requirement is needed in order to help us "prune" the reaction networks that our system generates. Defining what constitutes an interesting reaction is very much down to what specific things are being looked for, so it is important to carefully state what we consider interesting and be mindful of what information we are losing as we prune.

5.1.1 Defining System Complexity

The dynamics of our system can be described as a reaction network: a directed graph where nodes are particles in the system, and observed reactions are directed edges connecting reactants to products. Defining where a complexity measure should be maximum can be difficult. We can state that our measure should tend to zero when the behaviour is 'uninteresting', by which we mean when the system is either ordered or random in terms of reaction possibility and particle properties. The particle properties (the spikes in our system) are the only thing that particles can observe about each other, and the main thing that dictates what reactions can take place. We therefore think of these spikes as a descriptor of the system. This does not fully describe our system, since it is capable of degeneracy: two different RBNs could

have the same spikes produced by different underlying dynamics. In this case their current properties would be the same, however their different underlying dynamics could become visible after they have reacted and formed different composites.

A purely random system has no structure for defining which reactions occur, or the resulting properties of the products. Every possible reaction attempt would be successful, and any two particles could create a composite linked between any of their spikes. Additionally, there would be no connection between the property of the product and the properties of the reactants that formed it. In effect we would see a huge number of possible unique particles with almost no internal structure. Eventually the system would tend to one single composite containing all atomic particles.

A fully ordered system can take two forms. The simplest is a completely inert system where no reactions are possible at all. Alternatively, some reactions are possible, but no reaction changes particle properties—composites have identical spikes to their reactants—and therefore no reaction changes the behaviour in the system. In this case the set of possible reactions never changes. In practice this would be a system which produces only "polymers", particles with a rigid internal structure which are only ever capable of the same reactions they are a product of.

"Interesting" systems lay in between these extremes of order and randomness. Some reactions maintain the behaviour of particles and composites, while other reactions result in changes in behaviour causing products to have different spikes from their reactants, therefore changing the possible set of reactions.

An open ended system is one where a comparatively small but still unlimited number of reactions cause changes in behaviour. The occurrence of such reactions would open up new possible interactions to the system, possibly leading to new particle types. Our metric is designed to measure such behaviours in terms of spike properties.

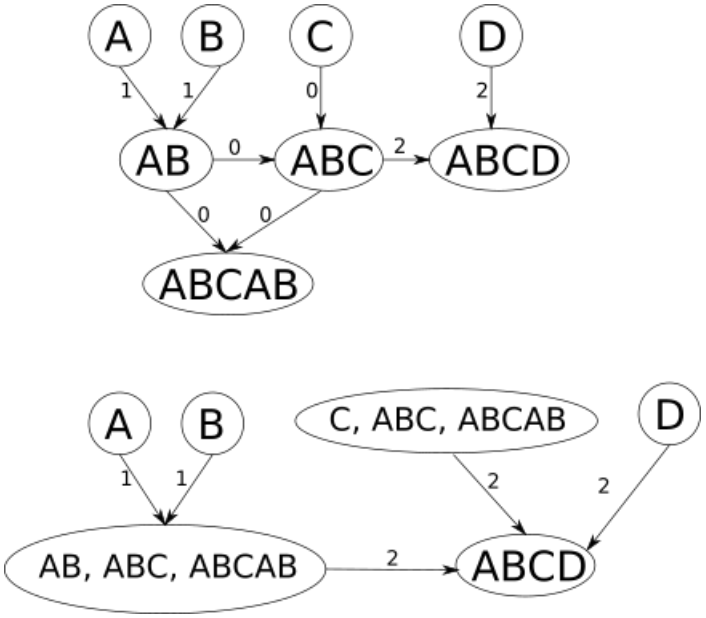


Figure 5.1: Set of reactions expressed as a reaction graph (top), and a functional grouping (bottom). Edges represent reactions labelled with how many spikes were changed by the reaction.

5.1.2 Functional Groupings

In order to generate a complexity measure we begin by reducing our reaction network to include only reactions that result in spike change.

We start from a complete reaction network, a graph describing all reactions that have occurred in the reaction vessel we are analysing.

We reduce this full reaction network graph to include only those reactions which result in a change to spikes, where the spikes of the product are different from the (non-linked) spikes of the reactants. This focusses on products that have new linking properties. We do this by merging particle nodes where reactions do not change spikes. Figure 5.1 shows an example of this grouping, reactions that produce ABC and $ABCAB$ do not cause a change in spikes therefore we merge the reactants with the products.

These groupings represent sets of particles that share a common root structure and for which there exist reactions that do not change the reactant's spikes.

If we describe the state of the system as the set of possible reactions that the system can undergo, reactions within a group do not change the system state. Edges between groups represent reactions that change unbound spikes. A change of unbound spikes means a change in possible reactions and therefore a change in system state.

The above reasoning holds true, a change to a spike means it can no longer participate in the same reactions that it could previously. However in practice the spike change will only have an observable effect if the newly changed spike is used for a reaction. This is not guaranteed simply due to the dynamics of the system, none of the experiments run in this work exhaustively attempt all possible reactions. The issue of whether or not the effect of spike change is observable is even more pronounced when we introduce a mass conservation concept (as later in this chapter). Reactions that result in spike change could be so rare and their products have such a low concentration that the effects of the spike change are not observable.

While these are limitations that we should be mindful of they are not sufficient to render this method useless. We can work around them by manipulating concentrations and initialisations so that the system is more likely to exploit a spike change.

This method of 'functional grouping' also allows us to reason about the bonding spikes themselves. If we observe that most successful reactions involving a given spike result in no change to unbound spikes we can think of that bonding spike as "frozen". Alternatively if any link on a given spike changes the properties of the other unbound spikes then we can consider that bonding spike as "active". We can then reason about the likelihood of reactions causing changes to the system based on what spikes are involved in the link and how frozen or active those spikes are.

In most cases a reaction will change only some spikes, not all of them. This allows us not only to label spikes as frozen or active, but to more precisely say how perturbing they are to the system, how many other spikes they change. This also gives us a degree of control over the "resolution" of our functional groupings. We can for example put in a cut off of 2 spikes so that any reactions that changes 2 or fewer

spikes are grouped as if they changed no spikes. In this way we can see only the most perturbing reactions if we wish.

In general we label edges on the reaction network with how many spikes are changed as a result of the reaction.

Originally we considered also tracking how much the spike changes by recording the change of spike sign and magnitude. However, this is not a useful measure to record for the current system. Because the stability and collision criterion (discussed in the previous chapter) are binary any spike change, irrelevant of the magnitude, results in a change to possible reactions. What the magnitude of the change dictates is the new set of reactions that are now possible, however we cannot find that set without exhaustively attempting reactions, which is not feasible.

5.1.3 Activity Measures in Terms of Functional Grouping

We reason about the dynamics of our system in terms of the identified functional groups as follows. As the number of groups approaches the total number of unique particles, we describe the system as random, since each particle has properties different from the reactants that made it. Conversely, as the number of groups approaches one we have an ordered system, since all found particles share the same properties.

We plot the activity A of the system in terms of the ratio between the number of groups and total number of unique particles produced:

$$A = \frac{FG(t)}{U(t)} \tag{5.1}$$

where $FG(t)$ is the number of functional groups after t bonding attempts; $U(t)$ is the number of unique particles after t attempts.

Clearly, $0 < A \leq 1$. As A tends towards 1 the system is more random, and as it tends towards 0 the system is more ordered. This can more clearly be observed as

change in activity, ΔA, over a period of $t + n$ reaction attempts.

$$\Delta A_n^t = \frac{FG(t+n) - FG(t)}{U(t+n) - U(t)} \tag{5.2}$$

Mapping these values over the runtime of a reactor and comparing them across different reactor types provides us with a way to reason about the system dynamics. It also highlights moments of increased activity, which we can then explore in more detail.

5.2 Experiment: Reactor Variants

As discussed at the end of the last chapter the reactor method used in the exploration was not a very realistic representation of a reactor. While realism is not a core aim of this work we none the less use real systems as inspiration and attempt to reason about what effect real world considerations have on the system.

The main omission so far has been mass conservation. The previous chapter pointed out that by ignoring this and stating that all particles are of equal concentration we biased the system's dynamics towards large particles of a single type.

A mass-conserving reactor, in which concentrations change as a result of reactions, is closer to reality. From a system's perspective, it also provides pressures on the types of particles that can be seen, and could lead to more diversity in the smaller composite particles generated. Reactant concentrations play an important role in reaction dynamics, so it is interesting to compare different reactor rules and their effects on observed system dynamics.

Here we explore how different types of mass-conserving reactors affect the dynamics of our system. We use 4 different reaction rules sets which we consider more 'more realistic'.

- Mass conserving

- Flow Random
- Flow Food
- Decant

We run each reactor scheme with each of the 20 atomic particle sets found in the previous chapter. Every reactor begins with 40,000 atomic particles, 200 of copies of each atomic particle in the set. We attempt 100,000 reactions, and repeat each instance 10 times.

5.2.1 Mass Conserving

The first reactor variant is a simple mass conserving reactor. A reaction attempt consists of removing two particles at random, attempting a reaction, and returning to the reactor whatever the products of the reaction are: a new particle for a successful attempt, or the original reactants for an unsuccessful attempt. Since no reaction can create or destroy atoms, only change links, the reactor conserves its mass.

As the reactor proceeds the concentrations of atomic particles decrease as more are used up to produce composites.

5.2.2 Flow Food and Flow Random

A flow reactor proceeds in the same way as the mass conserving reactor, by selecting particles to attempt reactions and returning the products. Periodically (every 5,000 reactions) the flow reactor injects new atomic particles into the system. In order to conserve mass a random set of particles with equal total mass is removed from the reactor. For example if 10 atomic particles are added to the reactor a set of particles with a total size of 10 is removed, either 10 atomic particles, 5 composite particles of size two, or any other combination that adds up.

The first variant of the flow reactor injects those atomic particles that have been involved in bonding over the period. It 'feeds' reactions that have already been observed. This should encourage more composite particles of similar types and more closely explore a dominant particle type.

The second variant injects random atomic particles from the initial atomic set. The aim here is to encourage diversity in composite particle types.

5.2.3 Decant

The decanting reactor outflows the largest 40 unique particles that are generated through a run to a second reactor. This second reactor is populated with 100 copies of each of these outflow particles, and is then run (again to 100,000 reaction attempt). This can be thought of as decanting the heavier particles and allowing them to interact separately. The aim is to observe if larger composite particles have different activity from smaller particles.

5.2.4 Results and discussion

Figures 5.2–5.5 show results for six of the 20 reactors, the two with the highest number of unique particles (reactors 13 and 18), two with the lowest (8 and 0) and two with a median number (7 and 2). Table 5.1 shows the average number of unique particles found under each reactor type. A unique particle is defined as one with a structure or composition that has never previously been seen in the run. In Spiky RBN, spikes are uniquely identifiable so structural uniqueness also includes particles that have the same geometric structure but the bonds are between different spikes. We consider reactors with a high number of unique particles to be more reactive and those with a smaller number more inert.

Figures 5.2–5.5 show the activity measure ΔA (Equation 5.2) for each reactor with an $n = 10,000$. Each boxplot represents the ΔA of the 10 repetitions of the reactor. So a low median would indicate activity over those 10,000 reactions was low (most

Reactor	Mass Conserving	Food Flow	Food Random	Decanted
13	1445	2924	1478	1819
18	1286	2915	1302	2112
2	488	1117	486	1263
7	468	787	459	1156
0	182	251	180	307
8	93	181	97	293

Table 5.1: Average unique particles discovered

reactions do not affect binding property), a high median represents high activity (most reactions do change binding property). The spread of each boxplot shows the difference between the 10 runs, high spread indicates the system is very dependant on the order of particles selected for binding, low spread the system is less dependant on the other of selection.

5.2.4.1 Mass Conserving

Figure 5.2 shows results for the mass conserving reactor. The reactors with low numbers of unique particles (reactor 8 and 0) show a much higher spread in activity between runs. This is likely because the overall activity of the system is much more reliant on which specific particles emerge. Reactor 8 on average creates only 93 unique particles over the run. The huge spread in activity is the result of precisely when specific particles emerge in each run. In the top two systems this is obscured since unique particles are more abundant.

The two top reactors show a difference in behaviour over time. Since the reactors are mass conserving, the later in the run the more likely it is that larger particles will be involved in reactions. For example the activity of reactor 13 is similar between different points in the run and averages around 0.5. This suggests that there is a balance between reactions that change binding property and those that do not, and that this balance is not affected by the contents of the reactor. Reactor 18 however shows an increase in activity value later in the run. This suggests that successful reactions later in the run are more likely to result in changes to binding property.

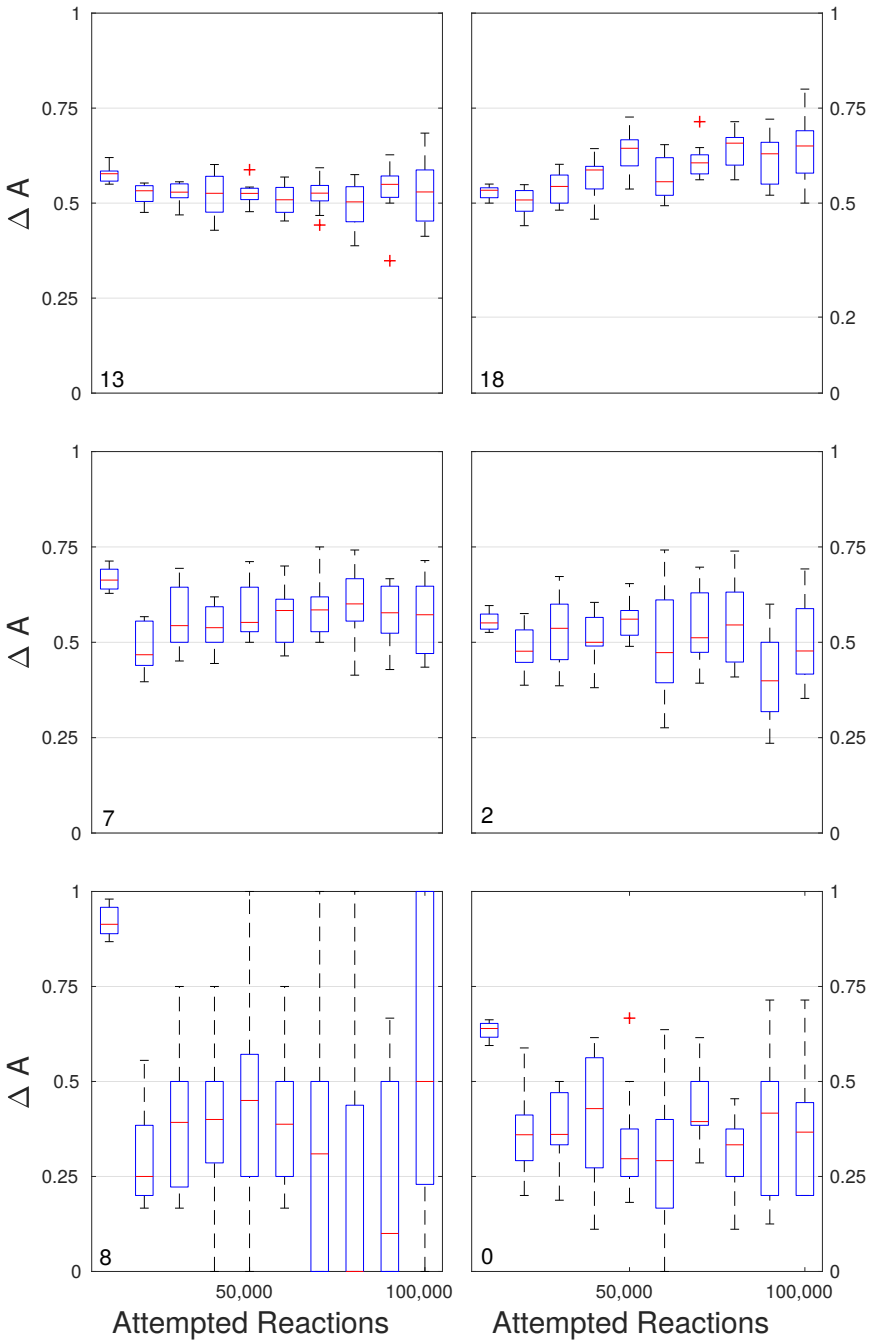


Figure 5.2: Mass Conserving: Plot shows the change in activity every 10,000 reactions attempts. The boxplots show the spread in behaviour over the 10 individual runs.

Since the only difference between earlier and later in the run is the size of particles involved we can say that composite particles have different behaviours from the atomic particles in this reactor.

5.2.4.2 Flow Food

Figure 5.3 shows results for a flow reactor which inputs atomic particles that were part of previous observed reactions.

This reactor type results in substantially more unique particles being found in each reactor instance. Over the run the concentration of inert particles in the system decreases as they are replaced by reactive ones, so the system finds more reactions overall.

As with the mass conserving example we see spread between runs increase as the reactors become more inert. Comparing Figure 5.2 and Figure 5.3 shows that the flow food reactor in general has a lower spread between runs. This is likely for the same reason, since food flow reactors are more reactive, the system is less sensitive to exactly which particles are observed.

Flow food reactors also show a decrease in activity measure compared to the mass conserving counterparts (Figure 5.2). A decreased activity measure indicates that more reactions are found which do not change the binding property. This behaviour is likely caused by reactions that progressively add the same particle to a structure without changing its binding property. Because these reactions are successful the flow reactor keeps feeding them, increasing the concentration of reactant particles and making the reactions more likely.

5.2.4.3 Flow Random

Figure 5.4 shows results for a flow reactor which inputs random particles from the reactor atomic set.

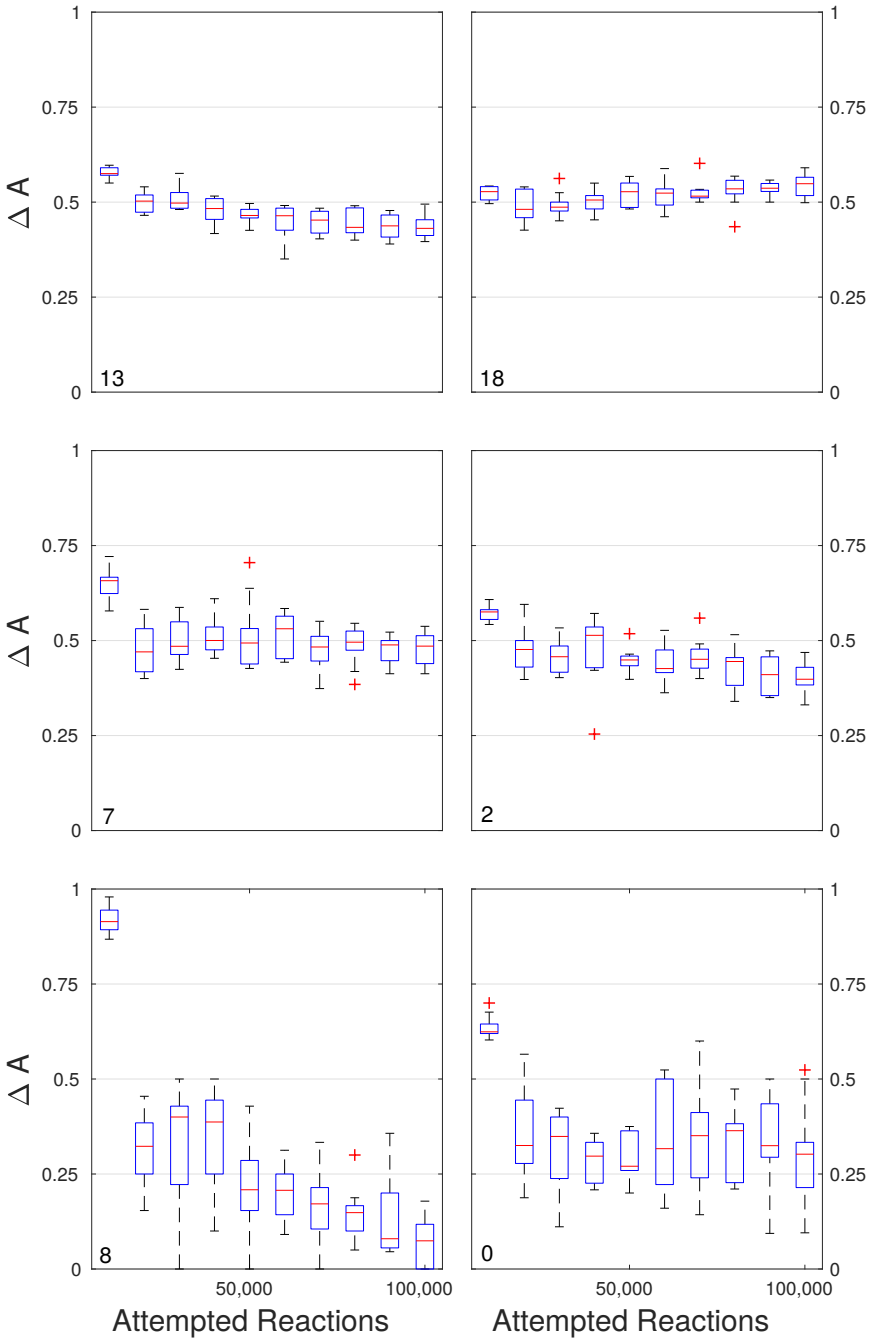


Figure 5.3: Flow Food: Plots showing the change in activity for each flow food reactor every 10,000 reactions attempts.

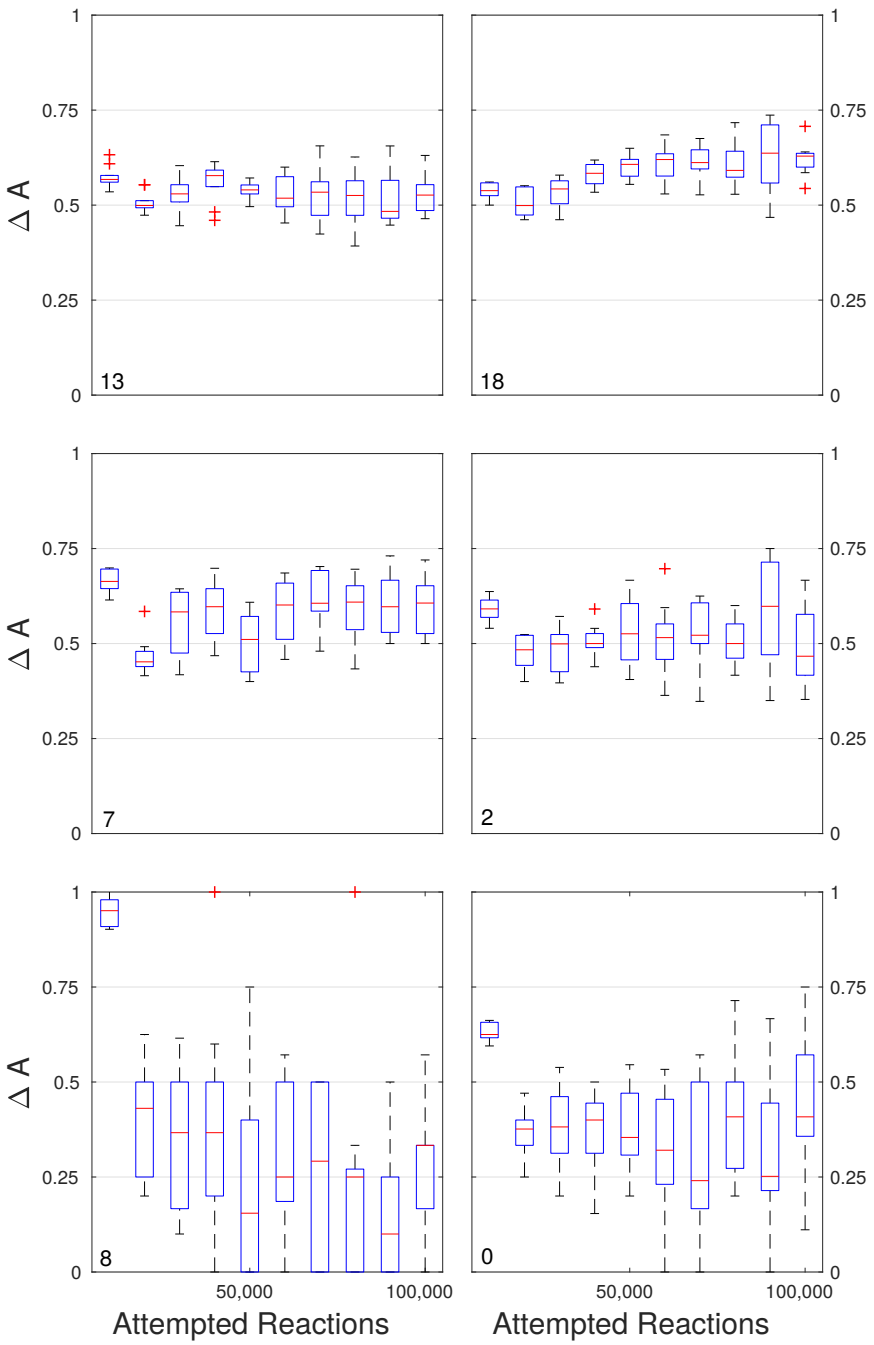


Figure 5.4: Flow Random: Plots showing the change in activity every 10,000 reactions attempts.

Overall this reactor type shows extremely similar behaviour to the mass conserving reactor, both in terms of activity measure and number of unique particles produced. This is not surprising since random inflow of particles maintains the concentrations of atomic particles in the system, making it equivalent to no flow. The similarity indicates that the concentration of composite particles in the mass conserving reactor is relatively low. If randomly replacing particles does not affect behaviour that means that mostly atomic particles are being replaced by other atomic particles. This suggests that the mass conserving reactor never reaches a point where its behaviour is dictated by the reactivity of composite particles.

5.2.4.4 Decanted Reactor

Figure 5.5 shows results for the decanted reactors. Each decanted reactor is populated with 100 copies of the 40 largest particles discovered in the Mass Conserving run. The decanted reactors are then run under mass conserving conditions for 100,000 reaction attempts. While each reactor instance is initialised with the same total number of composite particles, the total mass of each reactor instance varies depending on the total mass of the 40 largest particles discovered.

Table 5.1 shows that the decanted reactors are even more reactive then the atomic ones. In all cases substantially more reactions find unique particles. This trend is not necessary uniform ,with reactor 18 showing a larger increase then reactor 13, the same is true of reactors 8 and 0. This suggests that the behaviour of the reactor differs based on concentration of composites.

This change in behaviour is also observed when comparing Figure 5.2 and Figure 5.5. For example reactor 8 sees a very large decrease in activity measure. The majority of reactions in the decanted reactor now result in no change to the binding property. This explains the substantial increase in unique particles found since reactions of the form X-X + X $\rightarrow$ X-X-X are more likely not to change the binding property, meaning more X particles can be added to find new longer sequences.

Conversely reactor 0 shows an increase in activity measure. This means the system

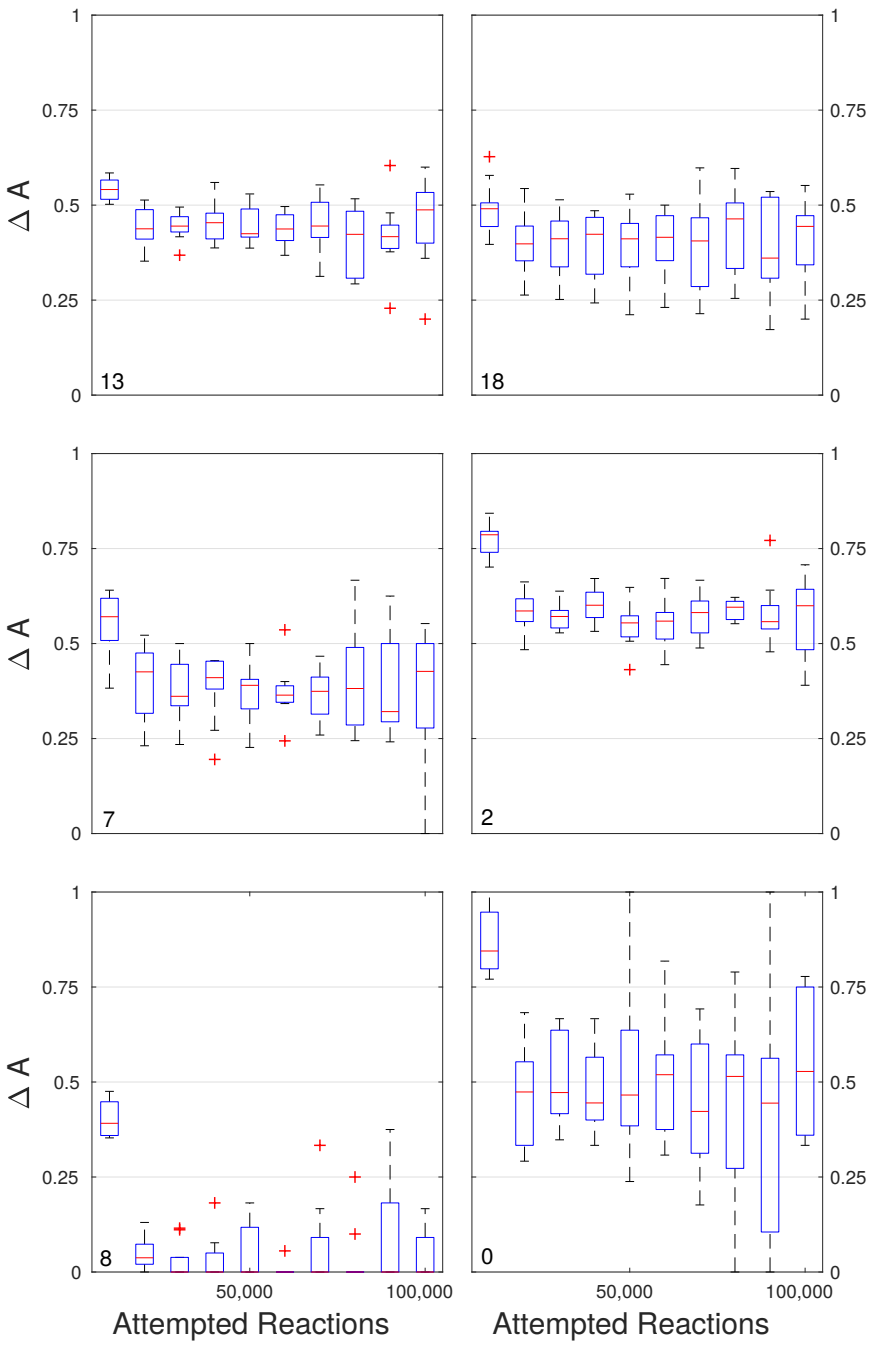


Figure 5.5: Decant: Plots showing the change in activity every 10,000 reactions attempts.

will find it harder to simply combine particles to create new larger composites since each reaction is more likely to change the binding property.

A similar trend can be seen between reactors 18 and 13. The decanted reactor 18 shows a much larger decrease in activity measure corresponding with a much larger increase in unique particles found. Meanwhile reactor 13 shows a smaller decrease in activity measure and a smaller increase in unique particles found.

Overall this suggests that the main driving force behind new composites being discovered are polymer-like reactions, which are less likely to change the binding property of the product, allowing for further addition. However, this observation cannot be generalised. Reactors 2 and 7 show very similar increases in unique particles, yet the activity measure of reactor 7 decreases, while increasing for reactor 2.

Some reactors also show a difference in activity measure spread between Figure 5.5 and Figure 5.2. An increase in spread between decanted runs (such as in reactor 18) suggests that the system is more sensitive to the specific composite particles used to populate the run. Alternatively decreased spread between runs (observed in reactor 2) implies that most composites have very similar behaviour, so the system is less sensitive to which ones are picked to populate the reactor.

5.2.5 Conclusion

In this chapter we present a metric for analysing the activity of our ssAchem. Instead of using a generic metric we tailor one, starting from general principles of information theory and equating them to the underlying dynamics of our system.

We then present a range of 4 reactor schemes designed to restrict system dynamics in ways we consider more realistic.

The results presented here illustrate some of the behavioural richness that the Spiky-RBN system can exhibit.

We see that different particle sets exhibit varying behaviours over system runs and

react in different ways to changes in reactor rules. This shows that the system is highly sensitive to instantiation and that behaviour is not just a product of particle properties but also of environmental properties. The variation between runs under the same conditions implies that the system can be sensitive even to the order in which reactions occur. It is possible that this variation can be removed by increasing the run lengths. However because the systems are mass conserving the order of reactions still matters, since many can end up being mutually exclusive if they rely on a common particle which has been used up.

Our measure for activity highlights possible behavioural differences in a way that can be related back to underlying mechanisms in the system. It allows us to formulate hypotheses about what specific behaviour is occurring 'under the hood'. This is important if we hope to reason about the behaviours and the system as a whole.

The analysis methods presented here do obscure many details about system behaviour, for example reaction dynamics and particle composition. However, the functional grouping on which the activity measure is based preserves some of that information. Further analysis on the functional groupings can highlight other aspects of system behaviour.

Our reactor results for Spiky-RBN illustrate how the configuration and initial state of a given AChem (here, varying flow schemes under the same reaction rules) have a profound effect on the resulting reactions.

This is a double edged sword. On the one hand it shows the system is expressive and capable of a range of behaviours. This also means that instantiation is possible method of controlling system behaviour. On the other hand the range of dynamics run to run suggest that the systems are very sensitive, which means generating specific behaviour even by controlling instantiation would be rather difficult.

It is interesting to relate this back to our natural inspiration. Laboratory chemistry concerns itself mostly with setting up the right types of particles under the carefully constructed conditions in order to produce expected behaviour. Much like in our system, 'instantiation' is key and small differences can have very undesirable results.

Chemists are very good at reasoning about instantiations and how best to set up systems, but this knowledge is derived from decades of experimentation.

Of course the natural system is substantially richer then our Spiky one. Other mechanisms exist which allow chemists to affect the reactivity of particles. At the most basic level energetics gives us a way to further restrict behaviour by introducing another controlling parameter.

In the next chapter we look at how to introduce an artificial analogue of energy into our system in a way that is consistent with our design requirements.

Chapter 6

Energetics

The model as described so far has already exhibited a number of interesting behaviours. and variations depending on initialisation and environmental constraints.

The richness in behaviours is substantially more limited as compared to what nature is capable of. One reason may be that the system as described so far is missing a large number of mechanisms that the natural system has access to.

Energetics is one such mechanism. Because our interest is not in modelling realistic reactions, but rather to generate a complex system, we can talk about energetics in very abstract terms.

Functionally, energetics refines the set of all conceivable reactions into the subset of reactions that are possible under the given environmental constraint (the temperature). As the temperature varies, the set of possible reactions varies. This temperature parameter is also internal to the system: the effects of a reaction often cause a change in temperature.

Overall, we can think of temperature as a way of changing component behaviour as well as an indirect method through which two system components can affect each other's behaviour.

This is as far as our analogue goes. We do not make any requirement on our analogue

of temperature maintaining other associated properties such as the relationship between temperature and reactivity. For this reason we shy away from using the term temperature and instead just refer to it as an 'environmental constraint'.

6.1 Energy and Environmental Constraint in Spiky RBN

In order to add a concept of energy to our system we need to augment our current linking algorithm. Whereas before the collision and stability criterion were binary functions, we would now like them to be probabilistic.

In addition, we need some global parameter for our environmental constraint which will have an effect on the reactivity of particles and their stability.

We can define what energetics should do in the context of our system.

- Reactions should be probabilistic with respect to the reactant particle's binding property.
- Reactions should be probabilistic with respect to a global parameter which we refer to as the environmental constraint.
- Reactions which produce a new product should affect the environmental constraint of the system.

In RBN-World, energetics was added by using the transient length of the reactant RBNs in order to calculate an activation energy. The Boltzmann factor was then used to calculate a probability of reaction.

The same method could be applied here; however it no longer fits the conceptual design of the system. The Boltzmann distribution is an observation on how the natural system behaves and the relationships between energy and temperature within that natural system. We should be aiming to find our own way of deriving a probability

that fits the conceptual frame of our particular system and is intrinsically linked to it.

Firstly, we need a property to represent particle energy. An obvious choice for this is the value of our spike. The energy of reactant particles determines their bonding behaviour and in our system the spike is the thing which determines this. Our spikes have a magnitude and a sign; when talking about the energy of a spike we can talk about its magnitude.

6.1.1 Energy-Environment relationship

We also require a way of coupling the environmental constraint of our system to the energy of the particles within it. As the constraint varies, the energy of our particles should vary. Within the context of our system we need to change spikes using some parameter which represents the environmental constraint.

A recurring problem when creating the system is that there are many possible ways of meeting our requirements. For example, we could simply add the environmental constraint as some arbitrary scaling function on spikes. We aim to restrict ourselves by attempting to maintain a consistent set of design principles. Instead of introducing an arbitrary mechanism by which the environment affects behaviour, we should aim to integrate the environmental constraint in such a way that it affects the sub-symbolic dynamics. In this way we do not prescribe how the environment affects our system, but instead observe how behaviour changes as a result of the environmental constraint.

To put this in a clearer manner we can classify the properties of our system so far into four categories.

- Sub-symbolic Structure (S_{SS})
  - The structural properties of the underlying representation. For example, topology of our RBN, number of nodes, number of inputs to each node,

boolean functions etc.

- Sub-symbolic Dynamics (D_{SS})
  - The dynamics of our underlying representation. For example, cycle and transient length, node states, etc.
- Particle Structure (S_P)
  - The properties representing the structure of our particle. In our case these are the interaction lists and links.
- Particle Dynamics (D_P)
  - The properties representing the dynamics of our particle. In our case the spikes defined by sign and magnitude.

The system has an inter dependency between these components.

$S_P = f(S_{SS})$: Particle structure is a function of the sub-symbolic structure.

$D_{SS} = f(S_{SS}, Init)$: sub-symbolic dynamics are a function of the sub-symbolic structure and the RBN initialisation.

$D_P = f(D_{SS}, S_P)$: particle dynamics are a function of the sub symbolic dynamics and the particle structure since they depend on the IL organisation.

We can now reason about how our environmental constraint should fit into this structure.

Our aim is for the environment to change the particle dynamics D_P. Therefore, the environmental constraint should affect either the sub-symbolic dynamics (D_{SS}) or the particle structure (S_P).

Conceptually, because this constraint is inspired by temperature we would prefer it to affect the dynamics rather than the structure of a particle. Since $D_{SS} = f(S_{SS}, Init)$ the natural place into which we can fit our constraint is the initialisation

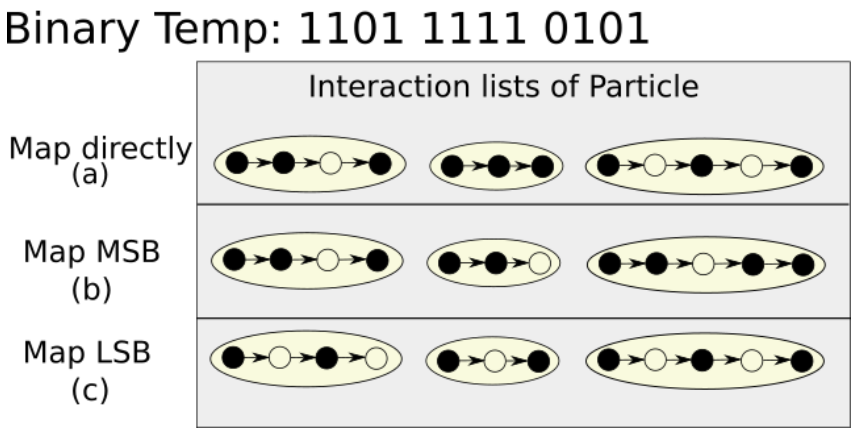


Figure 6.1: Different methods of mapping the binary environmental constraint to RBN. (a) directly maps whole pattern to nodes in order (b) maps MSB first into each interaction list separately and (c) maps LSB first into each interaction list separately

of the RBN. This fits well: different initialisations do not change the structure of the RBN but do change its dynamics.

Additionally, using initialisation to express the effect of the environment allows us to remove something we did not like about the system. In the system so far, the initial state of the RBN is an all True state. This choice is an arbitrary one, since there are no criteria by which to choose an initial state. This kind of arbitrary design choice is something we are attempting to avoid. Behaviour should be traceable and, if possible, we would like the system to set its own parameters.

6.1.2 Updated bonding Algorithm

Since the RBN initialisation can be represented as a binary string we can implement the constraint as a positive integer and initialise our RBN with the binary representation of that positive integer. Figure 6.1 shows a number of ways of mapping the binary pattern into an RBN.

The choice of how to map binary values to nodes should also not be arbitrary or random. Two identical particles should have identical behaviour under the same

environmental constraint. So the order by which we assign values to nodes should be consistent.

An easy approach would be to order the nodes in the RBN and assign values starting with the most significant bit of the constraint value such as Figure 6.1(a). When generating interaction lists we already have a mechanism of ordering nodes. Within each interaction list there is a well-defined linear order. We can order the interaction lists themselves based on some topological property which would give us a consistent total ordering of nodes.

One issue with this approach is that interaction lists would have variable sensitivity to the value of the constraint. The ILs initialised with the least significant bits of the constraint value (the right most IL in Figure. 6.1) would find their initial values changed much more often. In effect, our atoms would be more sensitive to the environment at one end and less sensitive at the other. Conceptually this seems wrong; the particle should not have such a heavily structurally dependant sensitivity to the environment.

Ordering the interaction lists in such a way also forces a very rigid structure to the particle. At the time of developing this there was the intention of extending the model further with a more complex geometry based on attraction/repulsion between spikes and links. With that in mind we did not want to impose a rigid ordering between interaction lists, since that should be something to emerge due to the geometry extension.

Another approach is to map the constraint representation separately into each IL (Figure6.1 (b) and (c)). With this approach an IL still has a variability in environmental constraint sensitivity but it is substantially less pronounced. Now constraint sensitivity is a function of IL size, which seems more intuitive. Also, there is no requirement for ordering of interaction lists.

However, the system overall becomes less sensitive to the environment. While maximum size ILs are possible and have been observed, generally we find three or four ILs per atom. This means that the particle is likely to be sensitive to only a smaller

constraint range (again as a function of the IL size). Depending on how we map the constraint into the ILs we can either have cyclic sensitivity (if we map from least significant bit, Figure. 6.1(c)) or coarse-grained sensitivity (if we map from most significant, Figure. 6.1(b)). Conceptually it makes sense for particle behaviour to change as the constraint varies and not to be cyclic.

Each node in the RBN is assigned a binary value from the constraint representation. This representation implicitly imposes some "physical constants" on our system. Our constraint range for example is bound by the size of our atomic RBN. Our system is capable of distinguishing 2^N unique constraint values (where N is the number of nodes in an atomic RBN), because that is the maximum binary pattern we can assign to an atomic particle. This means our system has an absolute maximum and minimum value of the constraint. This is a departure from our temperature inspiration which does not have a concept of maximum temperature.

Despite having conceptual advantages, this approach has a significant drawback. While we can represent the environmental constraint as a value, this type of mapping means that we cannot guarantee one basic dynamic observed in the natural chemical system. Generally, at a lower temperature, natural particles are less reactive and as the temperature increases the reactivity increases. In our system this is not guaranteed, and is indeed unlikely to be the case. The binary pattern that represents our constraint value does not guarantee a more reactive particle as its value increases. In the simplest terms we cannot guarantee that an all False initialisation (corresponding to absolute zero) will be the least reactive state of our system or that it will be less reactive then an all True initialisation (representing maximum temperature value possible).

This is a significant departure from our inspirational model. However, we specifically attempted to find a way of introducing the concept which we consider most important in temperature in a way that is internal to the system design. It is no surprise that our RBN particles do not follow the behavioural patterns of physical particles. If we wish to more closely resemble the natural system we can implement some external energy function like in previous work and use it to guarantee that reactivity

will scale with 'temperature'. However, at this point our system design is naturally providing us with options for introducing the mechanism we would like. Our goal is not to simulate natural systems but rather to use them as inspiration, so when the opportunity presents itself to derive dynamics from underlying mechanisms rather than impose the dynamics, we must take it. This approach naturally provides us with a relationship between energy and our environmental constraint and limits to our constraint representation values. It also introduces dynamics which make the system more expressive since IL size dictates sensitivity to the constraint.

Overall this is how we want to approach system design. In software it is relatively simple to implement any behaviour we desire, especially if we have an idea of how the mechanics behind the behaviour work. It is tempting to shoehorn in new mechanisms to make the system fit our intuition of how it should work. It is our aim to avoid this temptation. The system should be set up in such a way that natural solutions present themselves and we can implement (for example temperature) by finding components of the system that fit the type of interactions we would like to have.

6.1.3 Energy Loop

We now have an internalised method of converting an environmental constraint value into new particle behaviour. We also require a method of translating that behaviour back into a change to the constraint representation after a reaction has completed.

As discussed above we consider the spike magnitude to be our analogue for energy. A reaction results in a change in energy by changing the spike magnitudes involved in the bond. This is again an internalised mechanism. The change in spike is governed by the underlying change in dynamics of the new composite RBN rather than by some explicitly defined function. So, by finding the difference in spike magnitudes between the reactants and products, we get a ΔE that describes the energy change.

$$\Delta E = \sum_{i=1}^{N_S^{AB}} |S_i^{AB}| - \left(\sum_{i=1}^{N_S^{A}} |S_i^{A}| + \sum_{i=1}^{N_S^{B}} |S_i^{B}| \right) \tag{6.1}$$

Here S_i^A is the value of the ith spike of particle A which in total has N_S^A spikes. Note that we calculate this change in energy over all spikes, in the reactants and products, not just the spikes involved in the bonding.

We now require a method of converting this value into a ΔT our environmental constraint. Our constraint range is naturally bounded between a minimum and maximum, so a constraint change function will also need to be bounded between these values. Unlike previously there is no intuitive place in the system from which we can derive the ΔT function. We have some idea about how it should look: we know it should be a function of ΔE and that the change cannot exceed our $Max(T)$. This vagueness means we again have a wealth of options in terms of implementation. Even something as simple as a direct scaling of ΔE would suffice.

It is interesting to note where this vagueness comes from. When defining our energy and environmental constraint we had a good intuition of exactly where to fit it into our system. Even though our energy definition is a departure from the natural systems we use as inspiration, we are none-the-less confident that it is a good design decision. This clarity comes from having a well-defined particle with clear analogues between our implementation and the natural system. However, now that we come to define the link between energy and the environment we find no intuitive approach.

This lack of intuition comes from the simplistic definition of the environment overall. In the natural system, temperature is an expression of the kinetic energy of particles; as such the underlying dynamics that determine temperature are complex and dependant on a number of factors. Our environmental definition lacks these dynamics, and as such we have no intuitive method from which to derive our environmental constrain and its relationship with energy. Our constraint is simply a value and therefore there is no information from which to derive intuition. We can rectify this by adding to our environmental definition. Introducing particle motion and vibration would allow us to derive a constraint from those underlying dynamics. However, this would be a substantial undertaking which would make the system more complicated (and therefore computationally slower) and would require a substantial amount of time.

Instead we chose a function that defines the relationship based on how we would like the terms to relate.

$$\Delta T = -2^N + \frac{2^{N+1}}{1 + \exp\left(-\frac{S_r}{N_r}\Delta E\right)} \tag{6.2}$$

Constraint change is bound by the range $0 \leq T < 2^N$, which is bound by the number of nodes N in our atomic RBNs, $-2^N < \Delta T < 2^N$. So, we centre our function around 0 and introduce these limits.

ΔT is a function of ΔE (Equation 6.1) and the proportion of the environment involved in the reaction $\frac{S_r}{N_r}$ where S_r is the number of atomic particles involved in the reaction and N_r the total number of atomic particles in the reactor. This ratio is added because our "environment" consists purely of the particles in the reactor. An exo- or endo-thermic reaction in a small reactor will have a much larger effect on the reactor temperature then the same reaction happening in a much larger reactor. This is analogous to adding a small quantity of hot water to a small container of cool water or a large container of cool water. When adding to the large container you see a small change in temperature, when adding to the small container you see a larger change in temperature. We would like our environmental constraint to also have this behaviour.

The ratio acts to dynamically define the "steepness" of our function; if we did not use it we would have to add a parameter for this and attempt to set it by performing a parameter sweep.

6.1.4 Updated bonding Algorithm

We can introduce what we defined above into the current bonding mechanism and leave it as a deterministic system. However we would like to include a probabilistic element. The last part of the design is to update our bonding algorithm to be probabilistic. The updated reaction algorithm is shown in Figure 6.2.

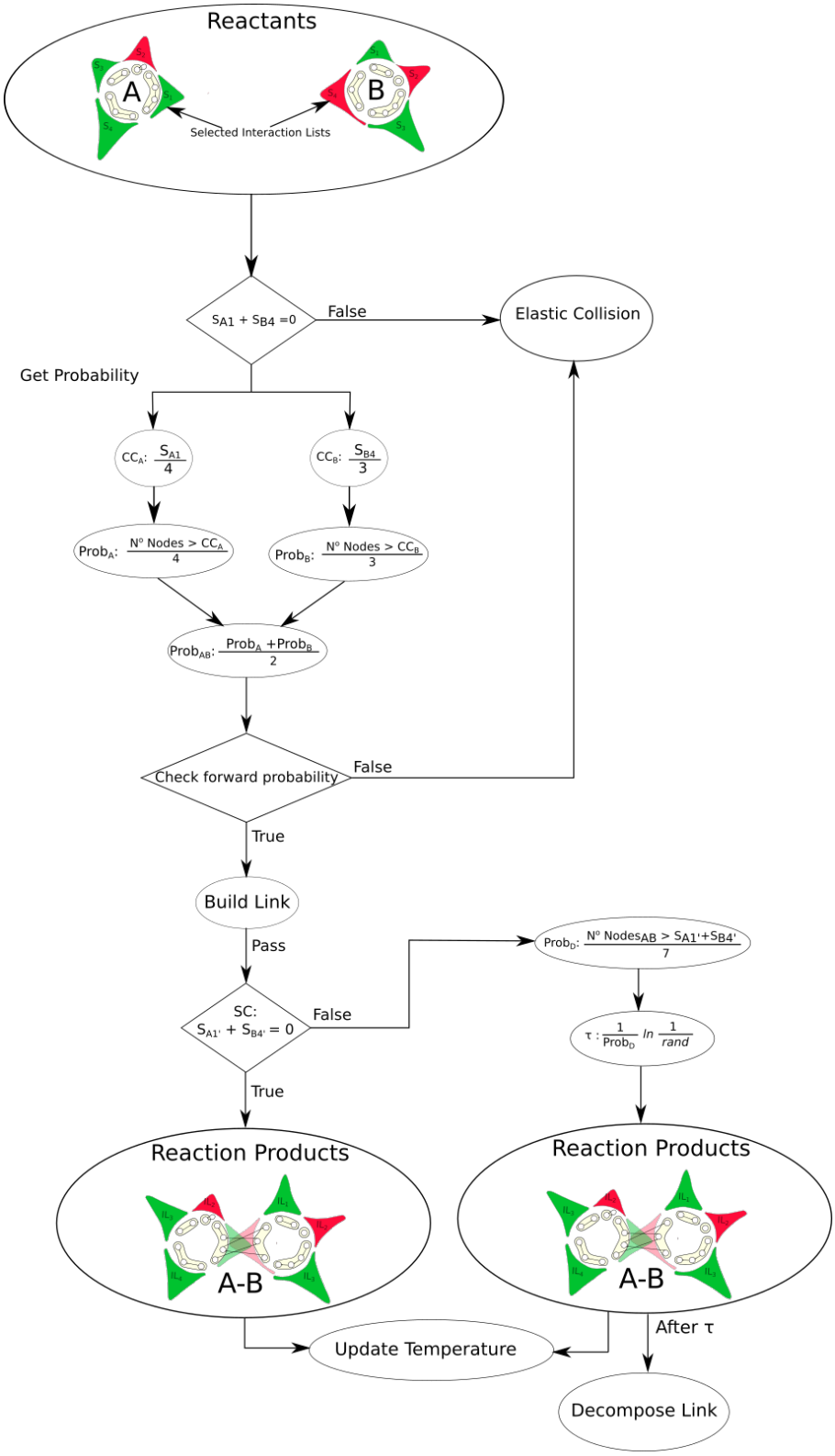


Figure 6.2: Reaction algorithm between two atomic particles with an environmental constraint. The spikes have already been calculated based on the constraint. They are selected as before and collision criterion is checked. If it passes then probabilistically based on CC a bond is formed. If the bond is stable, product is returned to reactor. If the bond is unstable τ is calculated and the product is returned for a decomposition at a later time. In either case environmental constraint is update.

Since the environment already affects particle dynamics we should derive the probability from those particle dynamics.

As discussed previously we consider the spike to be our analogue for energy. The spike is a summed quantity over the node states and cycle length. We can think of the spike as having a number of individual states represented by summing each node in the spike over the cycle length. When a reaction occurs the collision condition gives us a target value for the particles (CC). If we consider that the particles collide when the spike is in one of its possible states, we can think about the probability of the spike having a value greater or equal to the one required by the collision condition.

Currently however the ranges are mismatched. The spike and therefore our CC is bound by the interaction list size and cycle length:

$$-IL_{A1_{size}}c \leq S_{A1} \leq IL_{A1_{size}}c \tag{6.3}$$

whereas the individual spike node states are bound only to cycle length:

$$-c \leq N_{A1}c \tag{6.4}$$

To rectify this we divide the Spike by the number of nodes in the interaction List IL_{size}.

$$CC = \frac{S_{A1}}{IL_{A1_{size}}} \tag{6.5}$$

which puts our CC in the same range as the individual spike node states.

We can now get a probability by looking at the number of spike node states that are greater then CC.

$$Prob_A = \frac{N^o values > CC}{IL_{A1_{size}}} \tag{6.6}$$

Because the CC can be positive or negative there is imbalance. Negative CC values are likely to produce larger probabilities because all positive values will also count as greater than the CC. We can deal with this by only including values which have

the same sign when calculating probability. Because the collision condition will be negative for one reactant and positive for the other this imbalance is naturally corrected.

The overall probability of bonding is the average of the reactant probabilities. We can then sample from a uniform random distribution to determine if a reaction occurs or not. We could instead use the product of the probabilities for overall bonding probability. However at the time this seemed like it would produce a very restrictive algorithm by reducing bonding probability.

If the reaction is successful we update our constraint value according to Equations 6.1 and 6.2 and bond our particle based on the same bonding mechanism as before.

After bonding we again calculate a probability of decomposition in the same way but now based on the new spike values involved in the bond which take into account the new constraint value.

We again sample from a random distribution and calculate the time by which the particle will decompose using:

$$\tau = \frac{1}{Prob_D} ln \frac{1}{rand} \tag{6.7}$$

where $Prob_D$ is the decomposition probability and $rand$ the random number.

Time within our system is described in terms of reaction attempts so decomposition will occur after τ reaction attempts.

6.2 Experiments: Environmental scheme variants

We now explore how the addition of this new mechanism influences our reactor dynamics. The aim is to show that the introduction of this mechanism contracts or expands the range of possible reactions we can observe.

The energetics mechanism can be 'self contained' where reactions can vary the en-

vironmental constraint parameter themselves, but we can also set a constant value to run a reactor in something like a 'heat bath'. This is a useful exercise since our mechanisms do not afford us any intuition about which constraint values will give us the most/least reactive systems. It is highly likely that we will see high reactivity at different values depending on the particles in the reactor.

Here we present 3 reactor variants.

- Constant constraint value
- Varying constraint value
- Environmental feedback

As in the previous chapter, we run each reactor scheme for each of the 20 atomic particle sets. Every reactor begins with 40,000 atomic particles (200 copies of each atomic particle in the set). Each run consists of 100,000 reaction attempts and is repeated 10 times.

6.2.1 Constant constraint value

The first reactor variant maintains a constant value for the constraint and does not feedback the energy change due to reactions. As in the previous chapter, a reaction selects two particles at random, attempts the reaction under the new bonding algorithm (at the set constraint value) and returns all products to the reactor. If the bond is unstable we record the time at which it will decompose based on Equation 6.7.

Any bonds due for decomposition at this time are then decomposed and the stability of products is reassessed and recorded for further decomposition if required.

We repeat each reactor run 10 times for each of the 6 constraint values show in Table. 6.1. The constraint values are chosen based on the binary patterns they place in the interaction lists. Because of the average size of the ILs mostly the least

Temp	Hex	Binary
0	0x000	000000000000
2730	0xAAA	101010101010
3276	0xCCC	110011001100
3822	0xEEE	111011101110
4032	0xFC0	111111000000
4095	0xFFF	111111111111

Table 6.1: Environmental constraint values used in experiment represented as Dec, Hex and Binary

significant bits will be present. We expect that constraint values of 0 and 4032 for example to produce very similar results.

6.2.2 Varying Constraint Value

The second reactor variant externally changes the environmental constraint over the reactor run. The constraint changes every 8,333 reaction attempts, starting at zero and progressing in order through the values of Table. 6.1. This allows us two cycles through the value range in the 100,000 reactions.

The decanting reactor from the previous chapter shows that composites have different behaviour from atomic. The aim here is to see if the effects of the environmental constraint also vary depending on what has happened in the reactor before.

In this reactor, every time we transition the value of the constraint we re-evaluate bonds to see if the environmental variation has changed the bond stability. Changes in τ as a result of this new environmental constraint are added or subtracted from the time to decomposition. If this results in a negative time to decomposition then the bond is immediately decomposed. Bonds that were previously stable might decompose and bonds that were previously scheduled to decompose might become stable.

6.2.3 Environmental Feedback

The final reactor includes the environmental feed back loop as described in the preceding sections. Initial the environmental constraint is set at 0 and successful reactions change this value. Like the varying reactor, every change in constraint causes a re-evaluation of bonds and subsequently an update to the decomposition times.

In addition to reactions, products and particles in the system, we also record environmental constraint values over time.

6.3 Results and Discussion

In this section we explore the results from the above experiments. As with the previous chapter we focus on 6 reactors (13, 18, 7, 2, 8, 0) so that we can compare the effects of energetics on the system. Again we draw distributions of the activity measure ΔA (Equation 5.2) across 10 runs of each reactor under each experimental set up.

A reaction now occurs by sampling a random number against the reaction probability. Decomposition is also now recorded as a separate reaction since it occurs some time later. Activity is calculated in the same way as before and linking and decomposition are treated as distinct events.

6.3.1 Constant Constraint Value

Figures 6.3–6.8 show in blue the activity for the selected reactors at the constraint value in Table 6.1. In red are the results of mass conserving experiment without energetics from the previous chapter for comparison.

In all cases we see a substantial increase in ΔA compared to the results from the previous chapter. The least active reactors (8 and 0) show activity similar to that of the

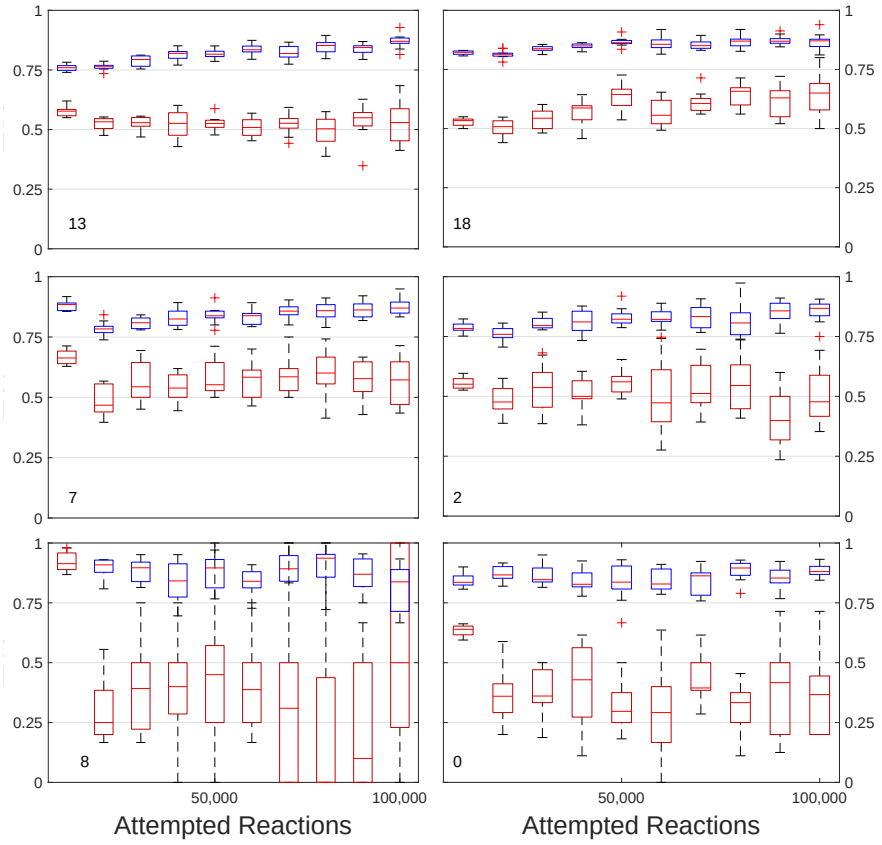


Figure 6.3: Constant environmental constraint = 0: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

most active. This is to be expected since reactions that were previously impossible can now be observed. Overall, all instances will be more likely to observe successful reactions. This implies that reactions that do not meet our stability constraint are also more likely to result in changes to spikes. Unstable reactions contribute to the activity measure twice first when formed and then when decomposing, while the overall effect over time may be neutral it will still have an effect on activity. However reactions are still only counted once. A consistently high activity implies that these temporary particles do give rise to new reactions themselves. If they were not able of further linking we would see activity start high and drop off as the unstable particles do not lead to any new unique reactions.

We also see that the variance in activity between runs decreases in all cases, most substantially for the less active reactors. This is consistent with our observations

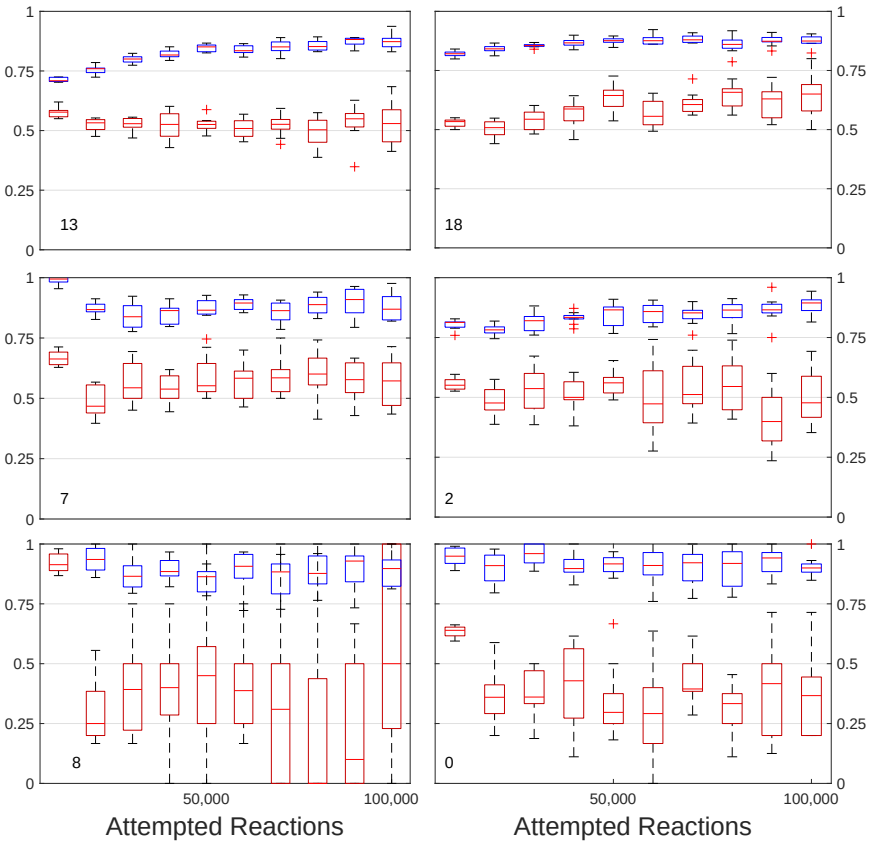


Figure 6.4: Constant environmental constraint = 2730: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

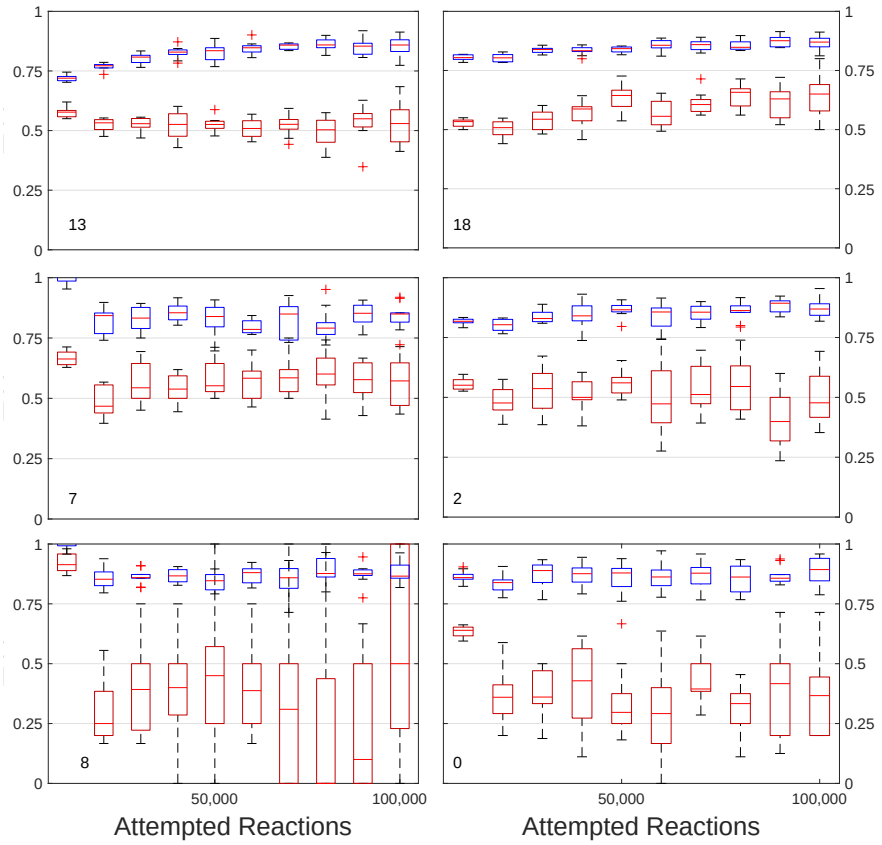


Figure 6.5: Constant environmental constraint = 3276: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

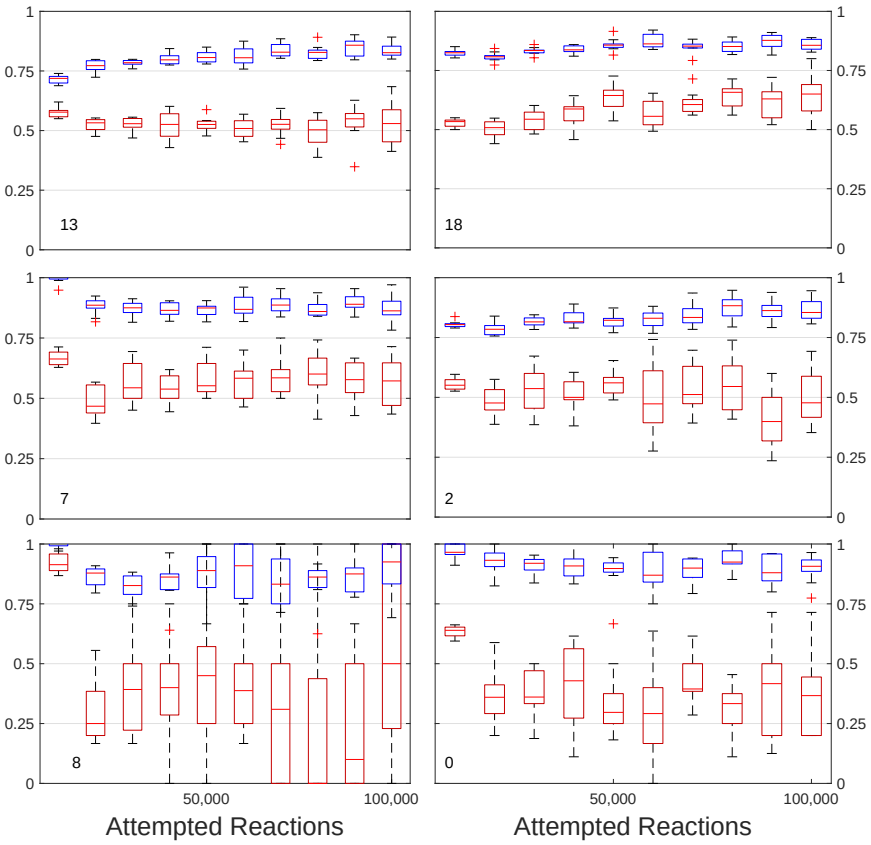


Figure 6.6: Constant environmental constraint = 3882: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

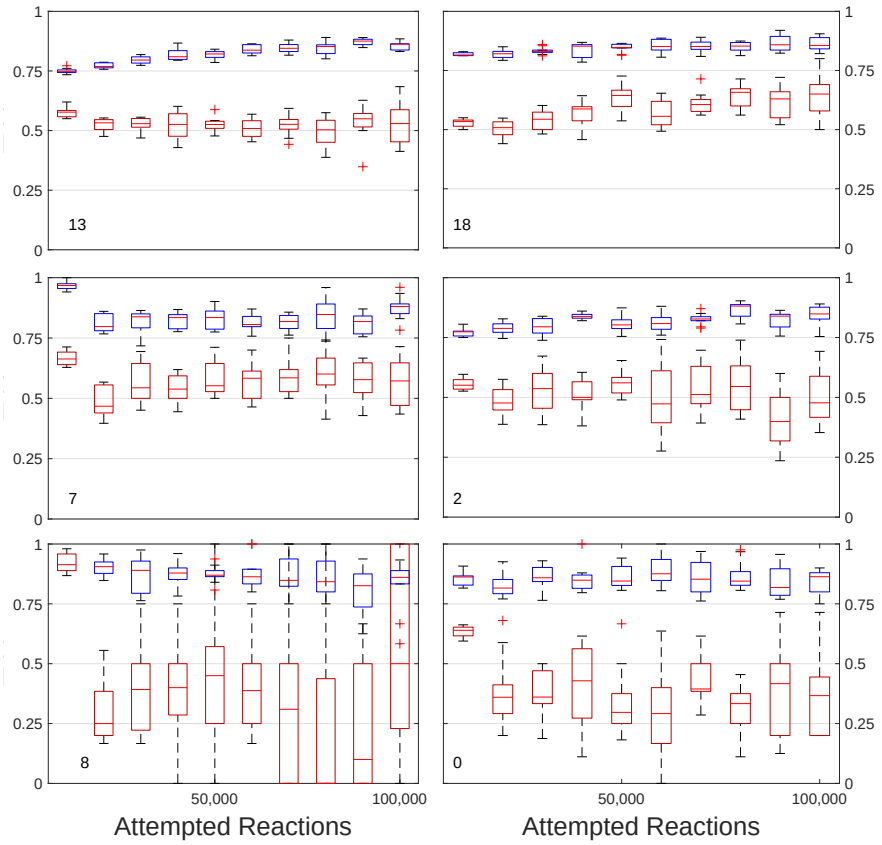


Figure 6.7: Constant environmental constraint = 4032: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

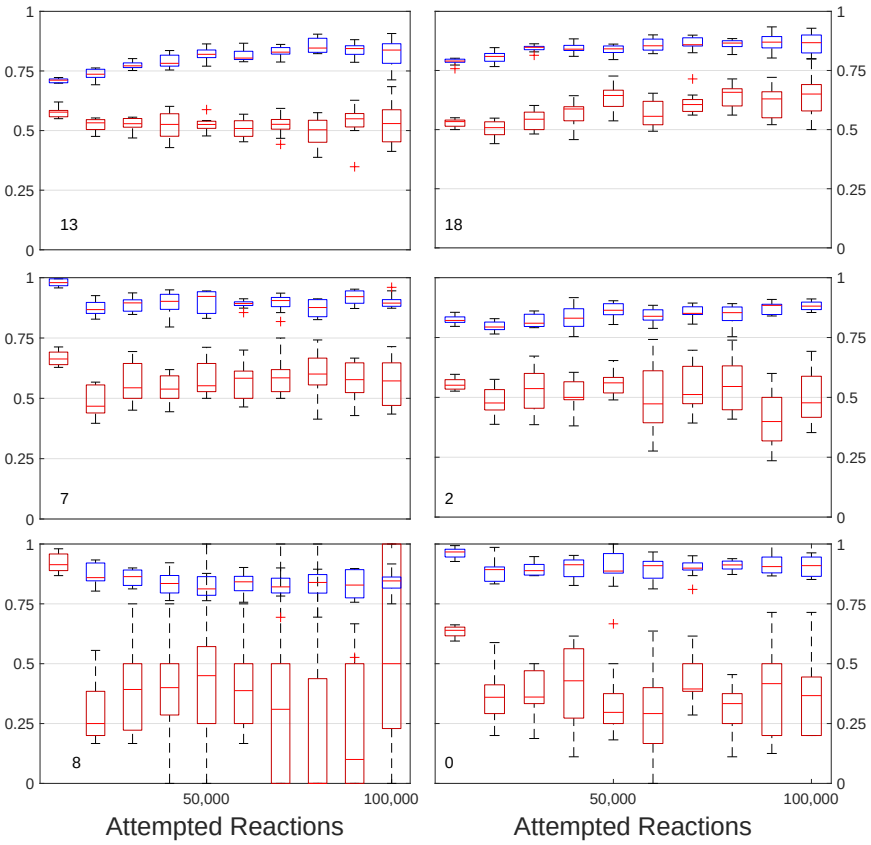


Figure 6.8: Constant environmental constraint = 4095: blue boxplots represent activity at this value, red represents mass conserving reactor without energetics algorithm

so far. Higher reactivity results in less variation between runs because the reactors are less reliant on generating a specific particle that opens up a larger number of possible reactions.

Most strikingly the graphs show that there is very little difference between different constraint values. This is unexpected. One possibility is that the constraint values we select happen to all be very reactive. However the mass conserving experiments are run with initial all false state, which is the same as zero value for the environmental constraint, Figure 6.3 has a zero value and we still observe behaviour similar to the other environmental constraint values. This implies that the new bonding algorithm is very permissive.

Reactor 7 shows some interesting behaviour. At all constraint values except 0 (Figures 6.4–6.8) we see that some runs reach an activity of 1 in the first 10,000 reactions. This means that every observed reaction results in a change in spike. It follows on from the mass conserving experiment where we see the initial 10,000 reactions being more random. So the reactor behaviour is similar but amplified by the addition of energetics. The fact that at constraint value zero we do not observe this (Figure 6.3) implies that the environmental constraint is restricting the possible paths.

6.3.2 Environmental feedback

Figure 6.9 shows results for reactors under varying constraint value. The reactors attempt 8,333 reaction at each constraint value according to Table 6.1, going from 0 and passing through the table twice.

As the constant constraint value experiment shows, we see high activity at all values. We therefore expect that varying the environmental constraint within a run will not have a very large effect on activity measures. In general this is the case. All reactors again show higher activity then without the environmental constraint mechanism. Varying the constraint value does not have a very large effect on the activity measure in general. Reactors 18 and 2 behave in a vary similar manner to the constant value

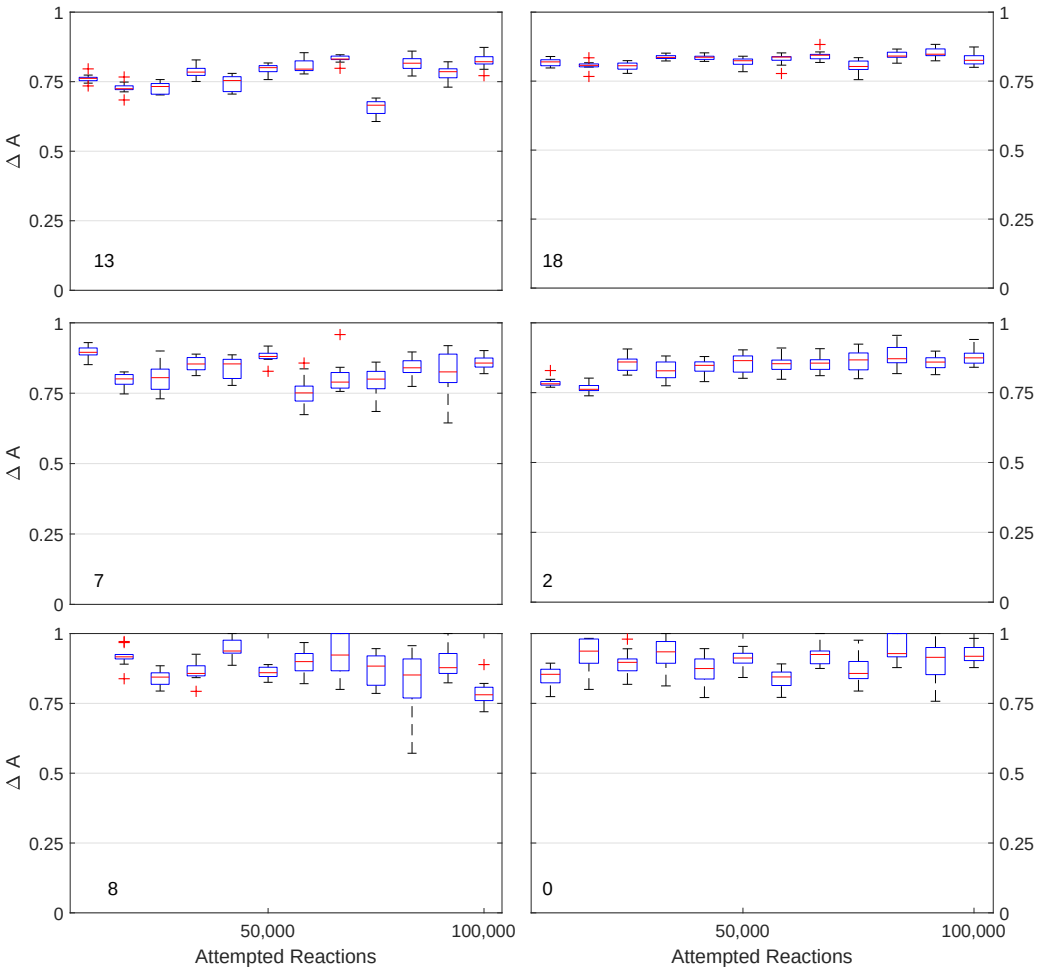


Figure 6.9: Reactors under varying environmental constraint value

counter-parts, and it seems that changing the constraint value does not have much of an effect on the median or variance of activity.

There are, however, some interesting specific behaviours. Reactor 13 shows a very substantial drop in activity at constraint value 3276 the second time round (9th boxplot). This is not present the first time the reactor encounters this value (3rd boxplot). Drops in activity suggest a tendency towards reactions that do not change spike properties. Over that time period the reactor is nevertheless active, averaging 249 reactions, compared to 236 the period before and 224 the period after. So it is not the case that fewer reactions are driving the activity down. Overall it appears that constraint value 3276 is much less conducive to reactions that change particle behaviour. The activity measure is still greater than 0.5, so reactions are still more likely to be change spike properties than not, but the probability is substantially lower.

The variance across the 10 runs is still low for that value. This implies that the reactor finds itself in this state every run. That means that the tendency towards lower activity at the specific environmental constraint value is a property of the reactor as a whole, not just of specific instances.

Reactor 7 also shows a noticeable change in activity the second time it reaches constraint value 0 (6th box plot). As the reactor goes through the that value, average activity increases; however, when it cycles back to 0 the activity drops. This shows that the environmental constraint by itself is not the determining factor in reactor behaviour. As reactor contents changes due to reactions, the behaviour of the reactor changes as well, even under the same environmental factors. This is related to observation from the previous constant constraint value experiment. We saw there that all constraint values except 0 had reactors reaching maximum activity. Based on these two experiments it seems that, for reactor 7, a 0 value is less likely to produce changes in spike property then other values.

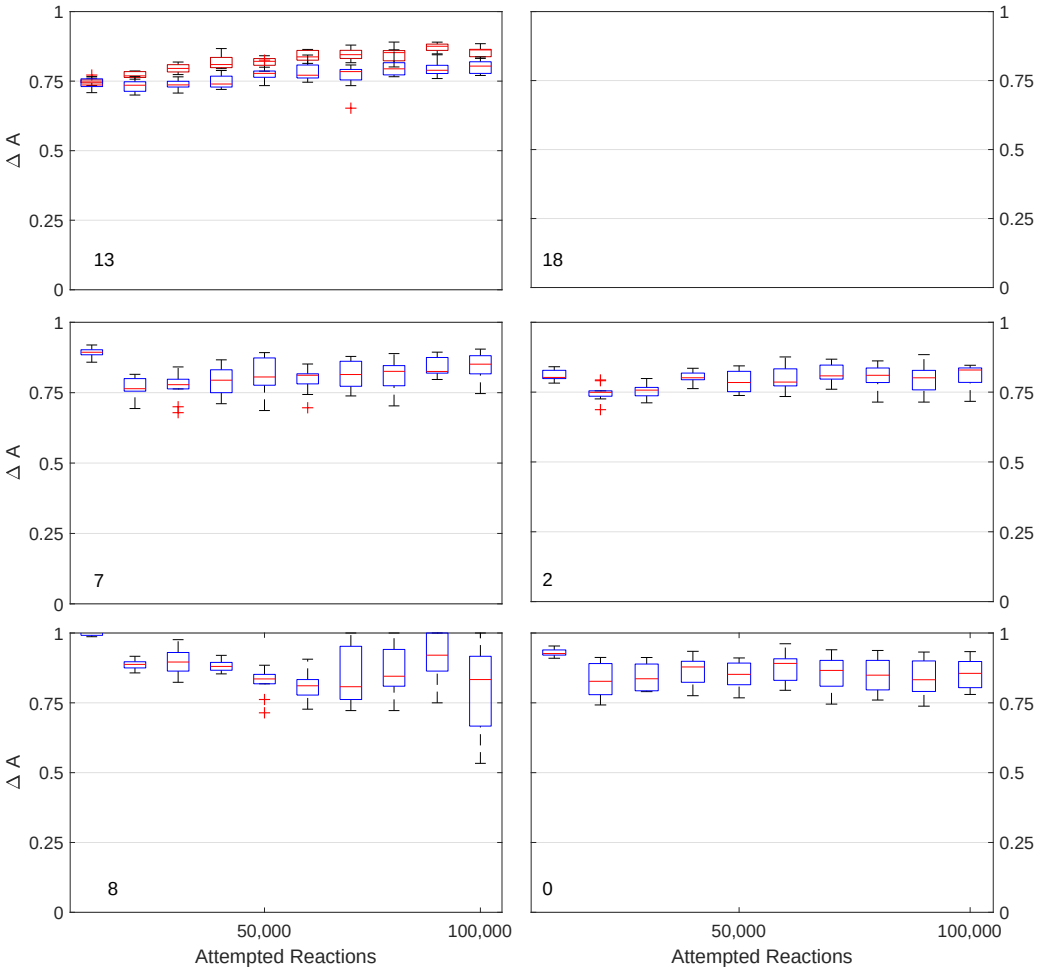


Figure 6.10: Reactors with environmental feedback. Reactor 18 is missing since activity couldn't be calculated. Reactor 13 includes results for constant constraint value(in red) for comparison.

6.3.3 Environmental Feedback

Figure 6.10 shows the results for the reactors with environmental feedback. Reactor 13 is shown alongside results from the constant constraint experiment for comparison.

Reactor 18 is missing since activity was not calculated. The activity calculation involves generating a reaction network graph and modifying it as described in the previous chapter. This can be computationally costly, especially when there is a large number of particles, since equality checks require graph matching. Since all particle graphs need to also be stored there is a substantial memory footprint. Reactor 18 had the largest number of unique particles and the memory footprint exceeded 16GB, causing a crash on the system. Due to time constraints we could not re-run the calculations.

This experiment confirms that environmental feedback does not have a large impact on activity measure. In general the inclusion of the environmental mechanic pushes activity higher. Most likely this is because the constraint mechanism allows mismatches in linking spikes which the activity measure would record.

All reactors have a very similar activity and distribution when compared to the previous experiments. Reactor 13 is somewhat unusual with a noticeable drop in activity measure when environmental feedback is included.

Figure 6.11 shows how the environmental constraint value varies within the reactor runs. The most striking observation is that the constraint value in all cases increases. This implies that most reactions are ‘exothermic‘ and increase the value. Most likely this is an indication of what constitutes a ‘reactive’ particle: in general reactive particles are ones which increase their spike size as they become involved in reactions.

We also see that the less reactive reactors have a substantially smaller total constraint values. This is to be expected since less reactions means less constraint value change opportunities.

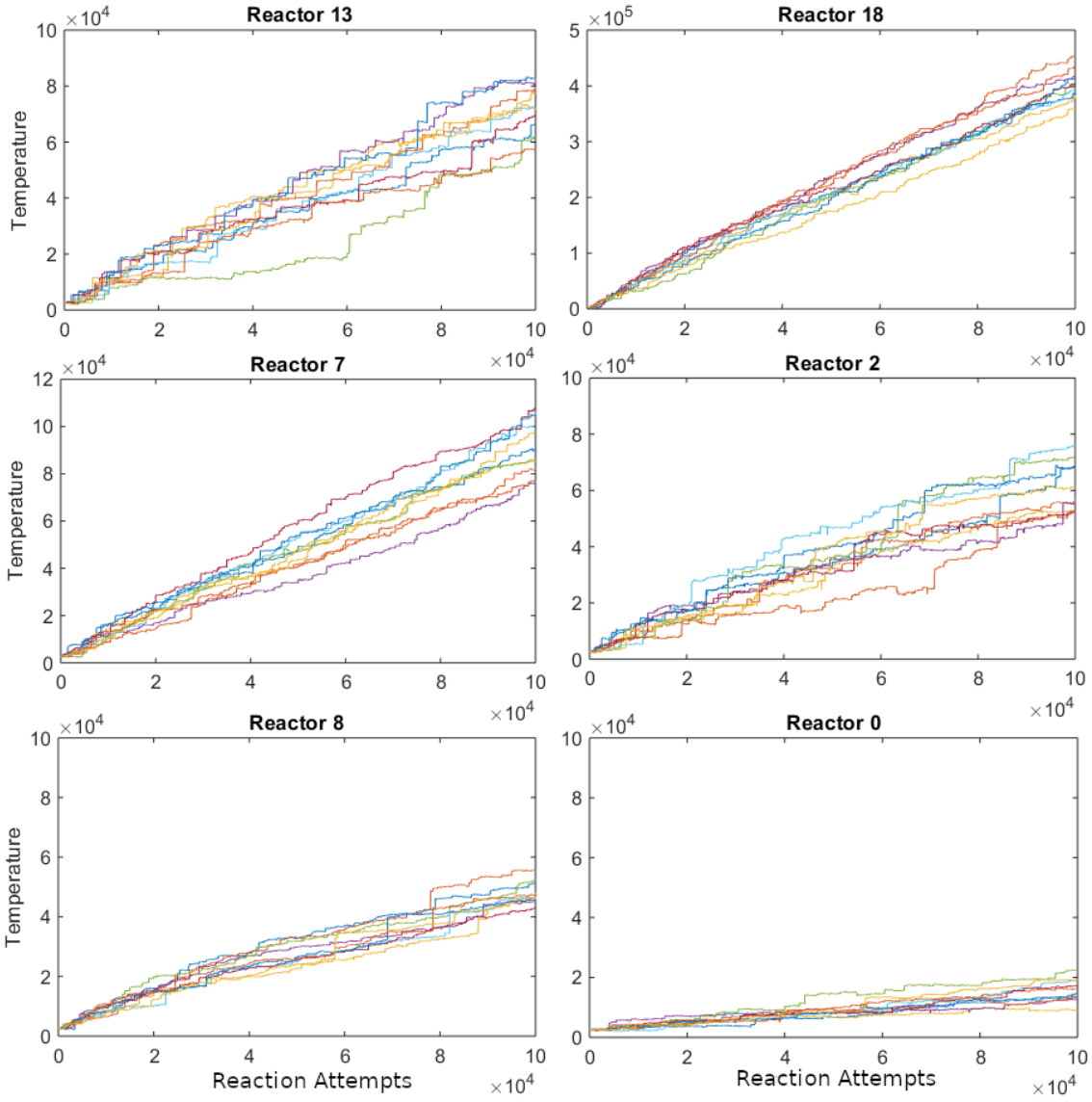


Figure 6.11: Changes in environmental constraint over the runs

Reactor 18 shows a much steeper increase in constraint value than the others. As the most reactive reactor this is expected: substantially more reactions increase the constraint value faster.

The constraint value figure does expose more differences between the reactors than the activity measure. We see that reactors 0, 8, 2 and 13 all show large jumps in constraint value corresponding to reactions that are exceptionally exothermic and have a substantial effect. Reactor 8 does not show the same behaviour. Here reactions have relatively similar, small constraint value changes, producing a 'smoother' line overall.

We also observe some variation between runs. Particularly reactor 13 shows one instance with a very slow change to the constraint value until a particular reaction around 60,000 attempts pushes the value higher. After that reaction the constraint value changes at a faster pace. This suggests that after that reaction the reactor start behaving differently and reaches a constraint value that enables a different, more 'exothermic' set of reactions. Reactor 2 also shows similar behaviour, but again only in one run.

6.3.4 Conclusion

In this chapter we introduced a mechanism inspired by energetics. We allowed particles to bond probabilistically and to vary their behaviour based on an environmental parameter. We then allowed the system to vary this environmental parameter as a function of its dynamics.

Overall we found that this new mechanisms pushed our activity measure substantially higher, implying a more random system. We found that all reactors become more likely to form links. This is expected since the probabilistic change to linking (Figure 6.2) made the algorithm more permissive by relaxing the stability condition. It is worth considering how to scale the activity measure by excluding certain things. This would allow us to focus the measure on major system events and have it be

less ‘sensitive’ to small perturbations.

The increase in activity is partially due to the new linking algorithm. We now probabilistically allow links with mismatched spike values. That means we are more likely to observe reactions where one or both spikes in the link changed and therefore contribute to the activity measure. We could recalculate activity under different conditions. Currently if at least one spike changes it contributes to activity. Instead we could require that at least two or three spikes change, and ignore spikes involved in links. This would decrease the activity value by excluding small changes to the overall system. It would be interesting to plot activity at a range of spike change thresholds in order to get a sense of how disruptive reactions are to system state however, time did now allow us to perform this.

Another likely contributing factor to the high activity is very unstable short-lived particles. If a reaction results in an unstable particle that will decompose quickly, then the composition and decomposition count towards activity, however they do not have any long lasting effect on the system. This is another area where we could perform some pruning of the activity measure.

The most unexpected observation was that varying our environmental constraint had only a small effect on activity. The activity measure is a good indicator of overall dynamics, however similarly high activity under different environmental constraint does not imply that the reactors are producing the same types of particles, simply that the particle dynamics (under linking) are similar. This is shown by the constraint value plots in the final experiment. While activity was high for all reactors we saw that there was differences in how the constraint value changed between some reactors, as well as between specific runs.

The overall high activity does suggest that the probabilistic linking algorithm is too permissive and eclipses any effects due to to the constraint. It would be interesting to include the environmental constraint mechanism (varying initialisation based on constraint value) with the original linking algorithm (ignoring the probabilistic additions) to see if the effects of our environmental constraint are more prominent but

again time did not permit this.

The final experiment exposed another interesting characteristic. In general we do not find a balance between ‘endo-‘ and ‘exothermic‘ reactions. Most reactions result in a positive energy change. This implies that particles capable of linking have a specific property under linking. It is hard to be certain if this is generally true, or an artifact of how atomic particles were selected.

In this work we chose to encode the environmental constraint from the lowest bit, making the particles more sensitive to small changes (Figure 6.1). However, the experiments show that the constraint value changes rapidly in the reactor, so it might be better to encode it from the highest bit. This would make the system less responsive to small changes when we allow environmental feedback.

Overall the mechanisms added in this chapter seem to make the system too reactive, mostly due to the new linking algorithm. We could constrain this by adding a scaling factor of some kind to the probability, but this would go against our design principles as it would be an arbitrary factor external to the system. Instead we could go back to the non-probabilistic bonding algorithm. This would decrease the overall reactivity and give us a better view of the effects of the environmental constraint. Another option would be to add a different mechanism that decreases reactivity. One option for this is to allow particles to ‘fold’ based on their spikes in the same way that biological molecules fold due to attraction and repulsion. By folding particles we can effectively block off possible linking spikes, decreasing the particle reactivity. Related to this is the addition of a spatial reactor. Again based on spike attraction and repulsion, particles moving in a 2D environment would constrain their possible reaction options. These are examples of other naturally inspired mechanisms that would provide environmental based constraints to the syste. The system here was designed with the possibility of adding these mechanism; it might be premature to suggest that it is too reactive until the effects of such new mechanism can be considered.

Chapter 7

Influencing System behaviour

7.1 Introduction

We have so far shown that our system has a varied range of behaviours. These behaviours depend on the specific particle instances in the reactor as well as on the environmental constraints we place on the reactor (such as mass conservation or our environmental constraint analogy to temperature).

If we provide the system with a feedback mechanism so that it can augment its own constraints, such as temperature, the behaviour changes yet again.

This behavioural range is something that we aimed to show in the system and as such is welcome. However, it does pose problems when attempting to define what behaviour is possible and which specific behaviours will be observed.

The range of interactions possible within the system varies as the system progress as a result of particles changing due to bonding or changes in the environment. This means that it is very difficult to predict the dynamics of a reactor by just looking at the atomic particles within it.

Again, this is very much by design since for a system to be complex it must be more than just the sum of its parts. However, we should none the less attempt to reason

about our system.

This chapter covers a conceptual model we use to reason about complex systems. While we use our particular system to talk about the model it is more generally applicable.

In order to discuss why such systems are hard to define we have to consider what about their instantiation (be it initialisation or specific constraints) causes such wide-ranging behaviour. If we wish to use complex systems towards specific tasks we have to consider how to construct them in a way that guarantees certain behaviours or properties.

To aid us in this discussion as well as to clarify where this behavioural ambiguity arises from, we construct a model of an instance of our system.

7.2 Instance model

We consider an instance of a system to contain four components, shown diagrammatically in Figure 7.1. We can think of this as a representation of one instance of a reactor. The diagram does not represent the specific Spiky model described in this work; it includes further mechanisms such as motion and non bonding interaction which are not developed in this work, they are there to show how a more expressive system fits into the conceptual model.

Below we discuss each component in detail but here we give a short overview of the model. An instance of our system contains specific particles and has an instantiated definition of the environment, collectively called the *Core Components*. The particles and environment interact with each other based on the dynamics we have programmed, which we refer to as *Core Interaction Channels*. Because we consider the system complex the Core Interaction Channels do not fully describe the dynamics, so there must also be other emergent dynamics described by the *Emergent Interaction Channels*. Finally, the tooling and metrics used to observe and record the reactor instance are described in the *Observation and Analysis* block.

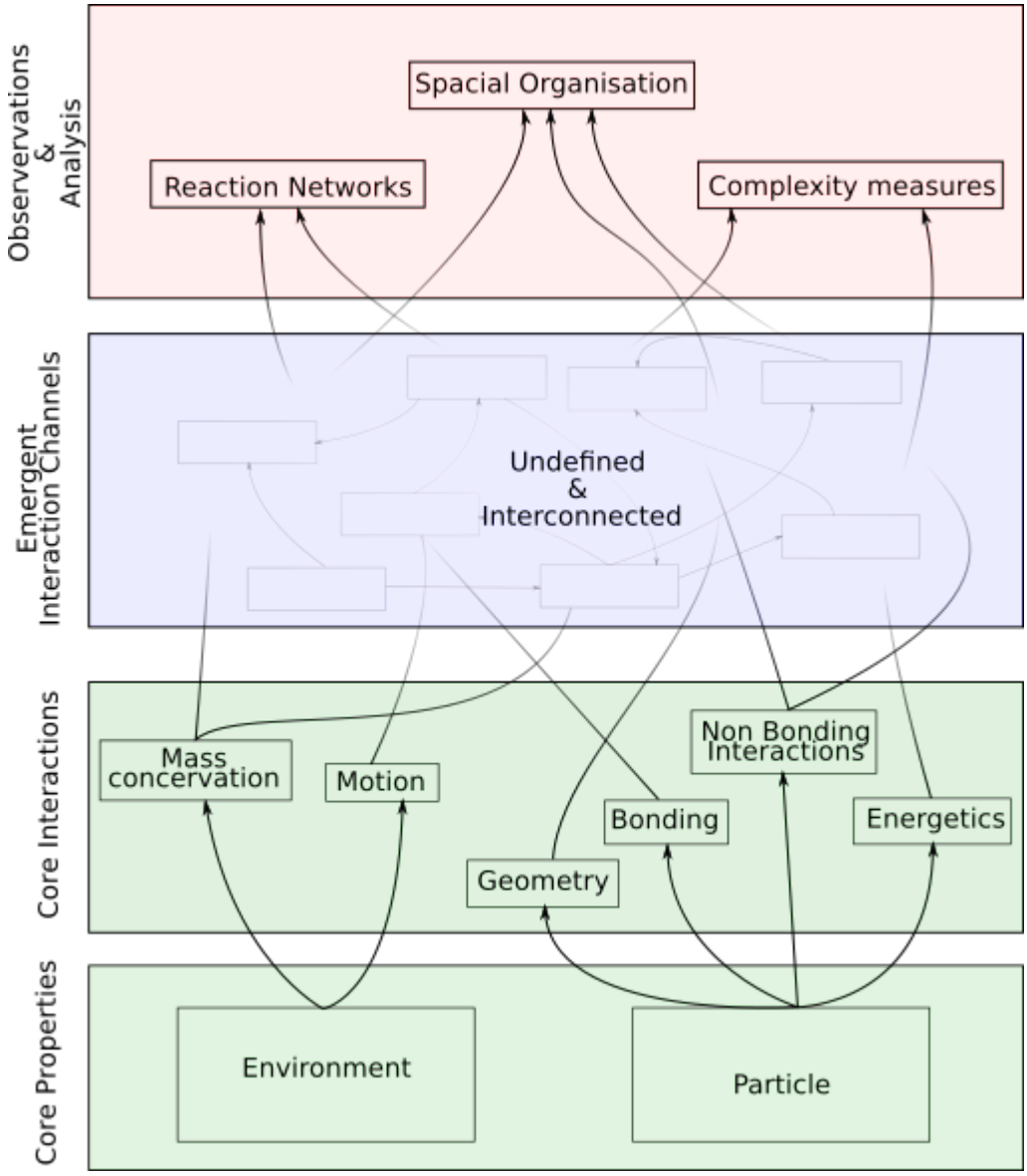


Figure 7.1: A representation of one instance of a system like ours. It contains the core properties describing the system state , the core interactions describing dynamics, the emergent components and the observation and analysis we apply to the instance.

7.2.1 Core Components

Our reactor contains the Core Components, instantiated components that define our system. In our case Core Components include the particles and their associated properties, as well as our specific definition of the environment such as the environmental constraint and reactor rules for particle flow etc. If the environment were spatial it would include the position of our particles within it. This layer of the model fully describes the system state at a given time.

In our instance model the environment and the particles are considered at the same level. Within the context of system behaviour there is no distinction between properties of the environment and properties of the particles within that environment. This is because behaviour depends on both. For example, the likelihood of observing a specific reaction in a reactor depends not only on the properties of the reactants but also on the concentration of the reactants within the reactor and the constraints placed by the environment of the reactor. Conversely the likelihood of a reactor having a given environmental constraint value depends on the reactions which are likely to occur. After running the reactor it is very difficult to separate the effects of environment and particles.

7.2.2 Core Interaction Channels

The dynamics of the reactor are defined in terms of Interaction Channels. An Interaction Channel is a path through the system that allows core components to influence each other's properties and therefore behaviours.

The Core Interaction Channels are the dynamics which we program into our system, such as bonding and 'temperature' effects. Bonding is a very direct Interaction Channel. A bond directly influences the properties and dynamics of the reactant particles. However, an Interaction Channel does not have to be direct. For example, the environmental constraint allows for particles to indirectly influence each other.

A reaction which results in a change to environmental constraints could influence the behaviour of all particles, not just those involved in the reaction itself.

An example is different mass conservation rules. Consider a reactor which contains only A and B atomic particles, which are capable only of producing the composite particle ABA. Reactants are randomly picked from the reactor. If there are always equal concentrations of all 3 possible particles then any successful reaction between A and B does not change the possible dynamics of the system: A and B have only the direct interaction channel defined by bonding. However, when mass conservation is introduced this changes. Every successful ABA particle reduces the concentration of A particles, which changes the likelihood of bonding. There is now an indirect interaction channel between the particles in the reactor. Over time the dynamics change as a result of this. By introducing a constraint on the system we have increased the interaction channels in the system and therefore promoted more complex dynamics.

These Core Interaction Channels describe all paths that we code into our model. They describe the microscopic dynamics of our system.

7.2.3 Emergent Interaction Channels

If our system is complex that means over the run time new dynamics will emerge on the macro scale that were not present on the micro scale. We describe these emergent dynamics as Emergent Interaction Channels. For example, catalysts within the system would be an emergent interaction channel since they augment particle behaviour and were not specifically encoded into the core dynamics. In a system with a concept of molecular geometry, large particles could fold in on themselves blocking off possible bonds, which results in an emergent Interaction Channel.

The Emergent Interaction Channels describe the macroscopic behaviour of the system. As our system is written statically and the program does not change over the run time, the only way an Emergent Interaction Channel can be created is by

combining the pre-existing interaction channels. For example, the emergence of a catalyst is possible because of the core mechanisms of energy (and its influence on bonding) and temperature (and its influence on particle behaviour). Catalysis is an example of a specific instance of bonding interactions that give rise to a specific possible energy interaction. The example of large particles changing their behaviour through folding is similar: the bonding characteristic of a large particle is directly affected by how it interacts with itself.

It is difficult to describe fully the specific emergent channels we will observe without running the system. An emergent channel is the result of a combination of Core Interaction Channels and properties. It will likely require specific interdependence between them that is subject to the instantiation. For example, a catalyst C which catalyses a reaction $A + B \rightarrow A\text{–}B$ will be specific to that instance of our reactor. If we place the same catalyst with different particles we cannot guarantee it will behave like a catalyst again. This is true not only of the particles in our system but also of the environment; at different environmental constraint values or with different dynamics it is possible that we will not observe the catalysis. We can clearly state that all emergent interaction channels are a result of core interaction channels and properties because that describes the complete system: there is no other place that the emergence can come from.

Emergent Interaction Channels can themselves be components to other emergent channels. For example, the existence of catalysts could allow for even more complex dynamics like catalytic hyper cycles. This combinatoric explosion makes it difficult to define the emergent behaviours without experimentation.

7.2.4 Observation and Analysis

The final component of our instance model is ourselves as observers. The evaluation tooling and how we record data from the system must be taken into account since it can lead to obscurification. Even though we are a passive observer of our system and do not interfere with its running, our understanding of the system depends

heavily on the data we collect. Our choice of what metrics to gather will augment our understanding of the dynamics. This is very important when attempting to explore emergent phenomena. For example, if we are recording only particle size and composition it is difficult for us to differentiate reaction dynamics. Two reactors can produce similarly sized particles consisting of a similar range of atomic particles; this does not mean that the systems produce the composites in the same way. One could involve catalytic reaction with a large number of intermediate steps, the other direct reactions which happen to be highly likely no matter the environmental constraints. If we attempt to draw conclusions about what is required for a reactor to form large particles from this limited observation we could end up missing some specific dynamic. We could end up attributing the phenomenon (in this case long chain composites) to one reactor property when in fact it resulted from two very different emergent properties that were obscured by our analysis.

In our system our observation can be relatively high resolution, outside of the probabilistic elements introduced by energy the rest is deterministic. Even the probabilistic components are derived from the system and we can reproduce the probabilities. Practically we are constrained by the volume of data. While we can record every event in the system and reproduce doing so is only worth while if we find something specific we wish to understand better.

The observation component of the instance model is included so that we can be aware of the effects it can have on our understanding of the system.

7.3 Directing emergent behaviour

Our instance model highlights some of the issues around working with emergent systems. Inherently they are very powerful, able to show growing complexity and the ability to find novel solutions to problems without requiring us to specifically encode the solutions. However, leveraging this power is difficult.

If we aim for an emergent system to be used for a specific purpose then we must

be able to describe and direct its behaviour in some way. We need to be confident that the emergent behaviours that are possible will lead to a solution and that there are no emergent behaviours that negatively impact our system (no bugs, as software developers refer to them).

7.3.1 Exploring behaviour in isolation

One method of exploring and describing behaviour would be to take a "classical" scientific approach. We can attempt to describe all emergent interaction channels and drill down on each one in order to expose what specific core properties and interaction channels gave rise to them. With this information we can then set up instances of our model that show only the emergent interactions we consider positive. Normally we attempt to do this by extracting the behaviour and abstracting out all components not related to it in order to study it in isolation. However due to the highly interconnected nature of the emergent interaction channels this can be difficult. An Emergent Interaction Channel can be the result of almost all Core Interaction Channels to a greater or lesser degree. Any simplification would result in a change to behaviour of the emergent channel, meaning the information we gather in isolation is not necessarily representative. Moreover there is no guarantee that the desired Emergent Interaction Channels will not inhibit each other or produce new unexpected channels that are detrimental to our solution. If our system aims to be in some way open ended in regards to its behaviour this problem only becomes more serious since we are sure we cannot define all emergent behaviours over time.

Even if we can identify and correctly define all favourable Emergent Channels and then create an instance that supports only those, we would in effect be programming a specific solution to the problem. This exhaustive exploration has produced a system that behaves in a specific and defined way. This misses the point. If we require a direct solution and know what that consists of, it is easier to just program that in a standard way. If we are not leveraging the system's ability to produce novelty there is no need to go to all the trouble of using it.

We can make a similar argument for the software approach to constructing the system. In general, we iteratively construct system components, test to ensure they behave as expected, integrate them into the whole and test that. However, the same issues appear. Because of the interconnected nature of the emergent components introducing any new core component is likely not only to produce new emergent channels but also augment the existing ones. This means that testing components individually is unable to guarantee how they will behave and that integration testing is not representative until we have the whole system in place.

Of course, there are practical considerations. This thesis describes work undertaken using such an iterative software development approach: we first built a core model and explored its behaviour, then extended it. This approach was taken because it is not viable to construct a full system and then attempt to explore it within the time limitations. Even so, the system described here is missing a number of probably essential behaviours, such as nonbonding interactions and molecular geometry.

7.3.2 Constructing emergent systems

The instance model points out a number of challenges that are faced when attempting to create an emergent, open ended system. It does however provide a useful insight into how we can go about constructing systems that show emergent behaviour.

As discussed, emergent interaction channels can be created only by composing core interaction channels. This composition is possible because our core interaction channels are functions over our core properties, which themselves depend on our underlying dynamics (the RBNs). For two Core Interaction channels to be composable they must have a method of influencing the properties on which they depend. This does not require that all core interaction channels are functions over a single core property. In fact, that would be detrimental since behaviour would only scale with one parameter.

In our system we derive our core properties from the sub-symbolic RBN dynamics. Any core interaction that affects the particle will affect not just one of the particles properties but all of them. This is also the case for composites; a change to one particle in a composite can affect all properties of all particles in that composite.

We can say that in general for a system to show a large number of emergent interaction channels it must have a number of core interaction channels which can be composed. These core interaction channels must be functions over more than one core property. However, the core properties should derive from a small set of underlying dynamics. This ensures they are interdependent meaning that the core interaction channels are composable.

In our system everything is derived from and affects the behaviour of the RBNs. This is true even of the environment if we allow an environmental feedback mechanism depending on reactions.

7.3.3 Controlling emergence through constraint

The instance model highlights one possible way of controlling emergent systems. As we have said before the core properties and interaction channels are static over the run time. This means that any constraints placed upon them are maintained over the run time of the system. A simple example would be mass conservation. If no core interaction channel can create or destroy atomic particles then the system guarantees mass conservation.

Any emergent dynamics also maintains the constraints imposed, because the emergent interaction channels can only be composed of core interaction channels.

This suggests an avenue of control over the system. Instead of attempting to find specific behaviour that meet our requirements we can constrain the core interactions so that no matter what emergent behaviour we observe we can guarantee some property of it, based on the constraints.

We can think about encoding problems as constraints over our Core Interaction

Channels. Any emergent behaviour we observe can be viewed as the system exploring the possible dynamics within the defined constraints.

This seems like an interesting way of encoding problems into emergent systems. It frees us from attempting to control and restrict the emergent dynamics and instead allows us to leverage them in order to find novel solutions.

The approach does require a different way of thinking about the tool. Systems constructed in this way are less likely to have a final system state that provides a solution. Instead we should look at the system dynamics and interpret them as the actions the system can take under specific constraints. These actions can be seen as the algorithmic steps that allow the system to maintain its constraints.

7.4 Conclusion

We have presented an instance model designed to allow us to reason about complex systems.

The model highlights the difficulty of attempting to build specific behaviours into emergent systems. The opaque and interconnected nature of the mechanisms leading to emergence make it difficult to apply standard methods to constructing or analysing these systems.

On the other hand the model leads us to a possible method of design. By designing with constraints written into the system we have access to properties which will be true no matter the behaviour. This gives us a way of controlling emergent systems without having to fight with the emergence. Under this design methodology emergent systems are not tools for finding solutions by looking at final system state. They are tools for exploring dynamics which we can interpret as steps the system can take while maintaining its constraints.

Chapter 8

Conclusion

8.1 Contributions

We have created a sub symbolic artificial chemistry using a consistent design approach:

- We design interactions between particles and only explore behaviour
- The interactions act exclusively on properties of the particles and the environment
- Properties are only derived from the sub symbolic representation
- Properties and interactions are not the result of arbitrary design decisions

We have shown that RBNs are a suitable underlying representation that has provided sufficient richness to allow us to implement the interactions and properties described in this work.

The system has shown a wide range of behaviours and an emergent richness.

The design approach has proved useful in guiding us in building this system. We have managed to derive properties from the underlying representation in a consistent

manner. This was achieved by finding a distinction between structural and dynamic properties and using it to guide our design.

The interactions are all determined by the underlying properties. We have attempted to minimise the amount of arbitrary decisions by always relating back to RBN properties.

However this has pushed us to designs that diverge from the natural systems we took as initial inspiration. An example is the implementation of energetics. While consistent with our representation of a particle, it does not conform to the 'expected' correlation between temperature and energetics.

We could have implemented energetics in a different manner that would have met these 'expectations' by adding a probability function and not changing the bonding method. However, that probability would then have been external to the system and would have been an arbitrary choice that enforces the 'expected' behaviour. Instead we focused on where the energetics naturally fit in the system as it stood.

Finally we reason about complex systems in general using an instance model of our system. With this we reason that a good approach towards using complex systems is to program them through constraints as oppose to attempt to get specific consistent emergent behaviour. Using the instance model we show that to reason about a complex system requires full understanding of its instantiation, the environment and initial state are just as important as the system components or the rules they obey.

8.2 Future Work

The system has a number of additional mechanism that could be added and a deeper search of the dynamics was not possible within the time constraints of this work.

Optimisation The system naturally lends itself to parallelisation. Reactions are self contained meaning we can perform multiple attempts at the same time

provided they are between different reactant instances. Energetics requires all particles to be recalculated every time the temperature changes. This calculations seems like it would parallelise well onto GPU solutions.

Interactions A spatial environment and non bonding interactions, such as attraction and repulsion between particles, would allow particle geometry to have a greater emergent impact. Folding of particles would allow geometry to have a greater influence on particle behaviour.

Tooling Tools for exploring instances, such as the ability to trace through reaction paths, automatically detect reaction cycles etc, would greatly increase our ability to analyse and understand the dynamics.

Exploration of Instances Due to the time constraints we were not able to fully explore the dynamics of the reactors we produced. A more in depth study of instances (using the tooling above) is important in describing what specific emergent dynamics the system can produce.

Controlling emergence The instance model described in the previous chapter suggests a path towards defining complex systems which can be programmed through careful construction, not by defining behaviour but by constraining core interactions.

Bibliography

[1] Bachrach, J., Beal, J., and McLurkin, J. (2010). Composable continuous-space programs for robotic swarms. *Neural Computing and Applications*, 19(6):825–847.

[2] Banks, E. R. (1971). Information processing and transmission in cellular automata.

[3] Banzhaf, W., Dittrich, P., and Eller, B. (1999). Self-organization in a system of binary strings with spatial interactions. *Physica D: Nonlinear Phenomena*, 125(1):85–104.

[4] Bedau, M. A., McCaskill, J. S., Packard, N. H., Rasmussen, S., Adami, C., Green, D. G., Ikegami, T., Kaneko, K., and Ray, T. S. (2000). Open problems in artificial life. *Artificial life*, 6(4):363–376.

[5] Benkö, G., Flamm, C., and Stadler, P. F. (2003a). Generic properties of chemical networks: Artificial chemistry based on graph rewriting. In *Advances in artificial life*, pages 10–19. Springer.

[6] Benkö, G., Flamm, C., and Stadler, P. F. (2003b). A graph-based toy model of chemistry. *Journal of Chemical Information and Computer Sciences*, 43(4):1085–1093.

[7] Bennett, C. H. (1995). Logical depth and physical complexity. *The Universal Turing Machine A Half-Century Survey*, pages 207–235.

[8] Berry, G. and Boudol, G. (1989). The chemical abstract machine. In *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 81–94. ACM.

[9] Bertz, S. H. (1981). The first general index of molecular complexity. *Journal of the American Chemical Society*, 103(12):3599–3601.

[10] Bialek, W., Nemenman, I., and Tishby, N. (2001). Complexity through nonextensivity. *Physica A: Statistical Mechanics and its Applications*, 302(1):89–99.

[11] Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K., et al. (2002). A racing algorithm for configuring metaheuristics. In *GECCO*, volume 2, pages 11–18.

[12] Bonchev, D. and Trinajstić, N. (1977). Information theory, distance matrix, and molecular branching. *The Journal of Chemical Physics*, 67(10):4517–4533.

[13] Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41.

[14] Bullnheimer, B., Hartl, R. F., and Strauss, C. (1999). A new rank based version of the ant system – a computational study. *Central European Journal of Operations Research*, 7(1):25–38.

[15] Chaitin, G. J. (1966). On the length of programs for computing finite binary sequences. *Journal of the ACM*, 13(4):547–569.

[16] Ciencias del Espacio, I. (2000). Measuring mutual information in random boolean networks. *Complex systems*, 12:241–252.

[17] Colorni, A., Dorigo, M., Maniezzo, V., and Trubian, M. (1994). Ant system for job-shop scheduling. *Belgian Journal of Operations Research, Statistics and Computer Science*, 34(1):39–53.

[18] Crutchfield, J. P. (1994). The calculi of emergence: computation, dynamics and induction. *Physica D: Nonlinear Phenomena*, 75(1):11–54.

[19] Crutchfield, J. P. and Feldman, D. P. (2003). Regularities unseen, randomness observed: Levels of entropy convergence. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 13(1):25–54.

[20] De Castro, L. N. and Von Zuben, F. J. (2000). The clonal selection algorithm with engineering applications. In *Workshop Proceedings of GECCO 2000, Workshop on Artificial Immune Systems and their Applications*, pages 36–39.

[21] De Castro, L. N. and Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE transactions on evolutionary computation*, 6(3):239–251.

[22] Dehmer, M. and Mowshowitz, A. (2011). A history of graph entropy measures. *Information Sciences*, 181(1):57–78.

[23] di Fenizio, P. S. and Banzhaf, W. (2000). A less abstract artificial chemistry. *Artificial Life*, 7:49–53.

[24] Dittrich, P. and Banzhaf, W. (1998). Self-evolution in a constructive binary string system. *Artificial Life*, 4(2):203–220.

[25] Dittrich, P., Ziegler, J. C., and Banzhaf, W. (2001). Artificial chemistries—a review. *Artificial life*, 7(3):225–275.

[26] Dorigo, M. and Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66.

[27] Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(1):29–41.

[28] Droop, A. and Hickinbotham, S. (2012). A quantitative measure of non-neutral evolutionary activity for systems that exhibit intrinsic fitness. *Artificial life*, 13:45–52.

[29] Drossel, B., Mihaljev, T., and Greil, F. (2005). Number and length of attractors in a critical kauffman model with connectivity one. *Physical Review Letters*, 94(8):088701.

[30] Eberhart, R. C. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science*, volume 1, pages 39–43. IEEE.

[31] Eberhart, R. C. and Shi, Y. (1998). Comparison between genetic algorithms and particle swarm optimization. In *Evolutionary Programming VII*, pages 611–616. Springer.

[32] Esser, J. and Schreckenberg, M. (1997). Microscopic simulation of urban traffic based on cellular automata. *International Journal of Modern Physics C*, 8(05):1025–1036.

[33] Faulconbridge, A. (2011). *RBN-World: sub-symbolic artificial chemistry for artificial life*. PhD thesis, University of York.

[34] Faulconbridge, A., Stepney, S., Miller, J. F., and Caves, L. S. (2010). RBN-World: The hunt for a rich AChem. In *ALife XII, Odense*, pages 261–268. MIT Press.

[35] Faulconbridge, A., Stepney, S., Miller, J. F., and Caves, L. S. (2011). RBN-World: A sub-symbolic artificial chemistry. In *ECAL 2009, Budapest*, pages 377–384. Springer.

[36] Faulkner, P., Krastev, M., Sebald, A., and Stepney, S. (2018). Sub-symbolic artificial chemistries. In *Inspired by Nature*, pages 287–322. Springer.

[37] Fontana, W. and Buss, L. W. (1994). What would be conserved if "the tape were played twice"? *Proceedings of the National Academy of Sciences*, 91(2):757–761.

[38] Gardner, M. (1970). The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223(4):120–123.

[39] Gershenson, C. (2004). Introduction to random boolean networks. *arXiv preprint nlin/0408006*.

[40] Gilbert, W. (1986). Origin of life: The RNA world. *Nature*, 319(6055).

[41] Gillespie, R. J. and Hargittai, I. (2013). *The VSEPR model of molecular geometry*. Courier Corporation.

[42] Gillespie, R. J. and Nyholm, R. S. (1957). Inorganic stereochemistry. *Quarterly Reviews, Chemical Society*, 11(4):339–380.

[43] Gonçalves, W., Pinto, R., Sartorelli, J., and de Oliveira, M. (1998). Inferring statistical complexity in the dripping faucet experiment. *Physica A: Statistical Mechanics and its Applications*, 257(1):385–389.

[44] Grassberger, P. (1986). Toward a quantitative theory of self-generated complexity. *International Journal of Theoretical Physics*, 25(9):907–938.

[45] Harvey, I. and Bossomaier, T. (1997). Time out of joint: Attractors in asynchronous random boolean networks. In *Proceedings of the Fourth European Conference on Artificial Life*, pages 67–75. MIT Press.

[46] Higashi, N. and Iba, H. (2003). Particle swarm optimization with gaussian mutation. In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, SIS'03*, pages 72–79. IEEE.

[47] Hutton, T. J. (2002). Evolvable self-replicating molecules in an artificial chemistry. *Artificial life*, 8(4):341–356.

[48] Hutton, T. J. (2004). A functional self-reproducing cell in a two-dimensional artificial chemistry. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE IX)*, pages 444–449. MIT Press.

[49] Hutton, T. J. (2005). Replicators that make all their own rules. In *Workshop on Artificial Chemistry and Its Applications, 8th European Conference on Artificial Life*.

[50] Hutton, T. J. (2007). Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial life*, 13(1):11–30.

[51] Imran, M., Hashim, R., and Khalid, N. E. A. (2013). An overview of particle swarm optimization variants. *Procedia Engineering*, 53:491–496.

[52] Inverardi, P. and Wolf, A. L. (1995). Formal specification and analysis of software architectures using the chemical abstract machine model. *IEEE Transactions on Software Engineering*, 21(4):373–386.

[53] Karaboga, D. and Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, 214(1):108–132.

[54] Karreman, G. (1955). Topological information content and chemical reactions. *The bulletin of mathematical biophysics*, 17(4):279–285.

[55] Kauffman, S., Peterson, C., Samuelsson, B., and Troein, C. (2003). Random boolean network models and the yeast transcriptional network. *Proceedings of the National Academy of Sciences*, 100(25):14796–14799.

[56] Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467.

[57] Kazadi, S. T. (2000). *Swarm engineering*. PhD thesis, California Institute of Technology.

[58] Khare, A. and Rangnekar, S. (2013). A review of particle swarm optimization and its applications in solar photovoltaic system. *Applied Soft Computing*, 13(5):2997–3006.

[59] Kolmogorov, A. N. (1965). Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7.

[60] Krastev, M., Sebald, A., and Stepney, S. (2016). Emergent bonding properties in the Spiky RBN AChem. In *ALife 2016*, pages 600–607. MIT Press.

[61] Krastev, M., Sebald, A., and Stepney, S. (2017). Functional grouping analysis of varying reactor types in the Spiky-RBN AChem. In *ECAL 2017*, pages 247–254. MIT Press.

[62] Krishnanand, K. and Ghose, D. (2005). Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*, pages 84–91. IEEE.

[63] Lerman, K., Galstyan, A., Martinoli, A., and Ijspeert, A. (2001). A macroscopic analytical model of collaboration in distributed robotic systems. *Artificial Life*, 7(4):375–393.

[64] Lessing, L., Dumitrescu, I., and Stützle, T. (2004). A comparison between aco algorithms for the set covering problem. In *Ant Colony Optimization and Swarm Intelligence*, pages 1–12. Springer.

[65] Lewis, G. N. (1916). The atom and the molecule. *Journal of the American Chemical Society*, 38(4):762–785.

[66] Liu, J.-l. (2005). Rank-based ant colony optimization applied to dynamic traveling salesman problems. *Engineering Optimization*, 37(8):831–847.

[67] Lopez-Ruiz, R., Mancini, H., and Calbet, X. (2002). A statistical measure of complexity. *arXiv preprint nlin/0205033*.

[68] Lucht, M. W. (2012). Size selection and adaptive evolution in an artificial chemistry. *Artificial life*, 18(2):143–163.

[69] Madina, D., Ono, N., and Ikegami, T. (2003). Cellular evolution in a 3D lattice artificial chemistry. In *Advances in artificial life*, pages 59–68. Springer.

[70] Martinoli, A., Ijspeert, A. J., and Mondada, F. (1999). Understanding collective aggregation mechanisms: From probabilistic modelling to experiments with real robots. *Robotics and Autonomous Systems*, 29(1):51–63.

[71] Mayer, B. and Rasmussen, S. (1998). Self-reproduction of dynamical hierarchies in chemical systems. In *Artificial life VI*, pages 123–129. MIT Press.

[72] McMullin, B. (1997). SCL: An artificial chemistry in Swarm. Technical Report 1997-01-002, Santa Fe Institute, USA.

[73] Mitchell, M., Crutchfield, J. P., and Das, R. (1996). Evolving cellular automata with genetic algorithms: A review of recent work. In *Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA96)*. Moscow.

[74] Mitchell, M., Crutchfield, J. P., and Hraber, P. T. (1994). Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D: Nonlinear Phenomena*, 75(1):361–391.

[75] Mohan, B. C. and Baskaran, R. (2012). A survey: Ant colony optimization based recent research and implementation on several engineering domain. *Expert Systems with Applications*, 39(4):4618–4627.

[76] Montero, E., Riff, M.-C., Pérez-Caceres, L., and Coello, C. A. C. (2012). Are state-of-the-art fine-tuning algorithms able to detect a dummy parameter? In *Parallel Problem Solving from Nature-PPSN XII*, pages 306–315. Springer.

[77] Mowshowitz, A. (1968a). Entropy and the complexity of graphs: I. an index of the relative complexity of a graph. *The bulletin of mathematical biophysics*, 30(1):175–204.

[78] Mowshowitz, A. (1968b). Entropy and the complexity of graphs: II. the information content of digraphs and infinite graphs. *The Bulletin of mathematical biophysics*, 30(2):225–240.

[79] Mowshowitz, A. (1968c). Entropy and the complexity of graphs: III. graphs with prescribed information content. *The bulletin of mathematical biophysics*, 30(3):387–414.

[80] Mowshowitz, A. (1968d). Entropy and the complexity of graphs: IV. entropy measures and graphical structure. *The bulletin of mathematical biophysics*, 30(4):533–546.

[81] Nouyan, S., Campo, A., and Dorigo, M. (2008). Path formation in a robot swarm. *Swarm Intelligence*, 2(1):1–23.

[82] Ono, N. and Ikegami, T. (2001). Artificial chemistry: Computational studies on the emergence of self-reproducing units. In *European Conference on Artificial Life*, pages 186–195. Springer.

[83] Parpinelli, R. S. and Lopes, H. S. (2011). New inspirations in swarm intelligence: a survey. *International Journal of Bio-Inspired Computation*, 3(1):1–16.

[84] Păun, G. (2000). Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143.

[85] Peixoto, T. P. (2014). The graph-tool python library. *figshare*.

[86] Prokopenko, M., Boschetti, F., and Ryan, A. J. (2009). An information-theoretic primer on complexity, self-organization, and emergence. *Complexity*, 15(1):11–28.

[87] Rashevsky, N. (1955). Life, information theory, and topology. *The bulletin of mathematical biophysics*, 17(3):229–235.

[88] Ray, A. (2004). Symbolic dynamic analysis of complex systems for anomaly detection. *Signal Processing*, 84(7):1115–1130.

[89] Richards, M. and Ventura, D. A. (2004). Choosing a starting configuration for particle swarm optimization. In *2004 IEEE International Joint Conference on Neural Networks*, volume 3, pages 2309–2312. IEEE.

[90] Schönfisch, B. (1995). Propagation of fronts in cellular automata. *Physica D: Nonlinear Phenomena*, 80(4):433–450.

[91] Schrödinger, E. (1926). An undulatory theory of the mechanics of atoms and molecules. *Physical Review*, 28(6):1049.

[92] Shalizi, C. R. (2001). *Causal architecture, complexity and self-organization in the time series and cellular automata*. PhD thesis, University of Wisconsin–Madison.

[93] Shalizi, C. R., Shalizi, K. L., and Crutchfield, J. P. (2002). An algorithm for pattern discovery in time series. *arXiv preprint cs/0210025*.

[94] Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

[95] Shi, Y. and Eberhart, R. (1998). A modified particle swarm optimizer. In *IEEE International Conference on Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence*, pages 69–73. IEEE.

[96] Socolar, J. E. and Kauffman, S. A. (2003). Scaling in ordered and critical random boolean networks. *Physical review letters*, 90(6):068702.

[97] Soysal, O., Bahçeci, E., and Sahin, E. (2007). Aggregation in swarm robotic systems: Evolution and probabilistic control. *Turkish Journal Electrical Engineering*, 15(2):199–225.

[98] Soysal, O. and Sahin, E. (2005). Probabilistic aggregation strategies in swarm robotic systems. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*, pages 325–332. IEEE.

[99] Stützle, T. and Hoos, H. (1997). Max-min ant system and local search for the traveling salesman problem. In *IEEE International Conference on Evolutionary Computation 1997*, pages 309–314. IEEE.

[100] Stützle, T. and Hoos, H. (1998). Improvements on the ant-system: Introducing the max-min ant system. In *Artificial Neural Nets and Genetic Algorithms*, pages 245–249. Springer.

[101] Stützle, T. and Hoos, H. H. (2000). Max–min ant system. *Future generation computer systems*, 16(8):889–914.

[102] Suzuki, Y., Fujiwara, Y., Takabayashi, J., and Tanaka, H. (2001). Artificial life applications of a class of P systems: Abstract rewriting systems on multisets. In *Multiset Processing*, pages 299–346. Springer.

[103] Suzuki, Y. and Tanaka, H. (1997). Symbolic chemical system based on abstract rewriting system and its behavior pattern. *Artificial Life and Robotics*, 1(4):211–219.

[104] Trucco, E. (1956). A note on the information content of graphs. *Bulletin of Mathematical Biology*, 18(2):129–135.

[105] Varela, F. G., Maturana, H. R., and Uribe, R. (1974). Autopoiesis: the organization of living systems, its characterization and a model. *Biosystems*, 5(4):187–196.

[106] Vesterstrøm, J. and Thomsen, R. (2004). A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In *Congress on Evolutionary Computation, CEC2004*, volume 2, pages 1980–1987. IEEE.

[107] Von Neumann, J. and Burks, A. W. (1966). *Theory of self-reproducing automata*. University of Illinois Press.

[108] Wang, H., Li, C., Liu, Y., and Zeng, S. (2007). A hybrid particle swarm algorithm with Cauchy mutation. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 356–360. IEEE.

[109] Wang, X. R., Lizier, J. T., and Prokopenko, M. (2011). Fisher information at the edge of chaos in random boolean networks. *Artificial life*, 17(4):315–329.

[110] Whalen, S., Bishop, M., and Crutchfield, J. P. (2010). Hidden Markov Models for automated protocol learning. In *Security and Privacy in Communication Networks*, pages 415–428. Springer.

[111] Winfield, A. F. T., Harper, C. J., and Nembrini, J. (2005). Towards dependable swarms and a new discipline of swarm engineering. In *Swarm robotics 2004*, pages 126–142. Springer.

[112] Wongthanavasu, S. and Ponkaew, J. (2013). Cellular automata for pattern recognition. In Salcido, A., editor, *Emerging Applications of Cellular Automata*.

[113] Xing, L.-N., Chen, Y.-W., Wang, P., Zhao, Q.-S., and Xiong, J. (2010). A knowledge-based ant colony optimization for flexible job shop scheduling problems. *Applied Soft Computing*, 10(3):888–896.

[114] Yuan, Z., de Oca, M. A. M., Birattari, M., and Stützle, T. (2012). Continuous optimization algorithms for tuning real and integer parameters of swarm intelligence algorithms. *Swarm Intelligence*, 6(1):49–75.