Chemical Programming to Exploit Chemical Reaction Systems for Computation

Dissertation
zur Erlangung des akademischen Grades
doctor rerum naturalium (Dr. rer. nat.)
vorgelegt dem Rat der Fakultät für Mathematik und Informatik
der Friedrich-Schiller-Universität Jena

von

Naoki Matsumaru

geboren am Aichi, Japan
in 28. Aug. 1974

Gutachter

1. :	PD Dr. habil. Peter Dittrich
2. :	Prof. Dr.-Ing. Dietmar Fey
3. :	Prof. Andy Adamatzky

Tag der letzten Prüfung des Rigorosums: 20. Aug. 2009
Tag der öffentlichen Verteidigung: 27. Aug. 2009

Chemical Programming to Exploit Chemical Reaction Systems for Computation

Dissertation
to receive the degree Dr. rer. nat. from the
Department of Mathematics and Computer Science
Friedrich-Schiller-University Jena
Germany

submitted by

Naoki Matsumaru

born in Aichi, Japan
on 28. Aug. 1974

Committee members

1. :	PD Dr. habil. Peter Dittrich
2. :	Prof. Dr.-Ing. Dietmar Fey
3. :	Prof. Andy Adamatzky

Date of the last examination: 20. Aug. 2009
Date of the public defense: 27. Aug. 2009

Dedication

To my family.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I wish to express my gratitude to all those who supported to complete this thesis. Grateful acknowledgement goes to: Florian Centler, Michael Conrad, Gerd Grünert. Thomas Hinze, Bashar Ibrahim. Christoph Kaleta, Thorsten Lenser, Jeffrey O. Pfaffmann. Pietro Speroni di Fenizio, Ralf-Peter Weiss, and Klaus-Peter Zauner.

For financial support, I thank: Deutsche Forschungsgemeinschaft (German Research Foundation) and Bundesministerium für Bildung und Forschung (Federal Ministry of Education and Research).

Abstract

All known life forms process information on molecular level. Biochemical information processing found in nature is known to be robust, self-organizing, adaptive, decentralized, asynchronous, fault-tolerant, and evolvable. In order to further exploit these properties new programming techniques are required. This thesis is on programming approaches to exploit the computational capabilities of chemical systems, consisting of two parts.

In the first part, constructive design, research activities on theoretical development of chemical programming are reported. As results of the investigations, general programming principles, named organization-oriented programming, are derived. The idea is to design reaction networks such that the desired computational outputs correspond to the organizational structures within the networks. Due to a relation between organizations and fixed points, supported by a theorem, the constructed reaction networks can be utilized to develop dynamical reaction systems with desired output behaviors.

The second part, autonomous design, discusses on programming strategies without human interactions, namely evolution and exploration. Motivations for this programming approach include possibilities to discover novelty without rationalization. Regarding first the evolutionary strategies, we rather focused on how to track the evolutionary processes. Our approach is to analyze these dynamical processes on a higher level of abstraction, and usefulness of distinguishing organizational evolution in space of organizations from actual evolution in state space is emphasized. As second strategy of autonomous chemical programming, we suggest an explorative approach, in which an automated system is utilized to explore the behavior of the chemical reaction system as a preliminary step. A specific aspect of the system's behavior becomes ready for a programmer to be chosen for a particular computational purpose. In this thesis, developments of autonomous exploration techniques are reported.

Finally, we discuss combining those two approaches, constructive design and autonomous design, titled as a hybrid approach. From our perspective, hybrid approaches are ideal, and cooperation of constructive design and autonomous design is fruitful. In order to design chemical computing systems, those approaches can be employed alternatively from developing modular parts of chemical programs to developing whole programs. Furthermore, constructing chemical reaction systems produce useful experiences to improve autonomous designs. Autonomously designed systems are proper sources of inspiration for chemical programming in a constructive way.

Constructive Design

Introduction

When exploiting chemical reaction systems for computational purposes, users are demanded to program or control in a different manner from a conventional algorithmic manner due to the highly parallelized nature of the computational processes. In this thesis, we discuss two different approaches to program or design chemical computing systems: constructive design approach and autonomous design approach. Those two are not completely independent of each other, and a hybrid approach, combining multiple programming approaches, will be ideal. Nevertheless, those two approaches are studied in two separate parts. In Part I, chemical computing systems are designed in a constructive way.

0.1 Introduction

As chemical computing systems, we concentrate on well-stirred reaction vessels, populated with multiple molecules, so that spatial configurations are neglected. The molecular concentration profiles represent the state of the computational systems, and that state is changed through physicochemical interactions amongst those molecules. The output of the chemical computation is derived from the outcomes of the state changes. To program such a chemical system in a constructive manner is to define and manipulate molecules and their interactions. Since controlling and programming the physicochemical interactions in real chemistry is currently impractical, our targets are "artificial chemistries" [Dittrich, 2001]. An advantage of this limitation to the artifacts is that no restrictions from practical implementation on real chemistry need to be considered. Thus, our study can be focused on the theoretical aspects of programming chemical computing systems.

Investigating the theoretical aspects of chemical programming is beneficial. In chemical computing, chemical reaction rules are the fundamental components of the programs, specified at the microscopic levels. A global systems' state emerges from local interactions between molecules according to the reaction rules specified [Banzhaf et al., 1996]. The relation between those two levels is highly non-linear. The reaction rules are likely to be contingent on each other even though physi-electrical influences can be neglected in artificial simulation settings. For instance, two reactions can share the common species as products or reactants. Self-reproduction reaction rule should also be considered. To make the dependency between reactions more complex, reactants can be a product of multiple pathways, consisting of a series of reactions, operating together in a complex manner. These tangled reactions are operated in parallel, moreover. It seems scarcely possible to predict the global behavior from those local interactions. The main reasons of this difficulty are the requirement to analyze those systems as a whole. Studying only the local interactions does not lead to the understanding of the global systems' behavior. This gap is not specific for chemical systems but also applicable to technical systems employing a large number of entities. Establishing a theoretical basis of this emergence is much appreciated in general [Müller-Schloer, 2004]. Moreover, the ability to predict how a chemical program behaves is a prerequisite for programming by construction [Zauner, 2005]. Although being rather abstract, the study of artificial chemistries provides useful information with respect to the relationship between the local and global levels [Fontana and Buss, 1994; Furusawa and Kaneko, 1998; Jain and Krishna, 2001; Mayer and Rasmussen, 2000].

Chemical organization theory has been proposed by Dittrich and Speroni di Fenizio [2007] to provide a new perspective to study complex dynamical reaction networks. We suggest that theory as a tool helping to construct (i.e., program) and analyze (i.e., describe and understand) chemical computing systems [Matsumaru et al., 2007]. Applying chemical organization theory, the constructed reaction networks are decomposed into overlapping sub-networks called "(chemical) organizations" according to qualitative steadiness estimated from the network structure. A organization is defined as a set of molecular species that is closed and self-maintaining, and it has been proven that only an organizational set of species can constitute a stationary state, given an ordinary differential equation system describing the dynamics of the constructed reaction network system [Dittrich and Speroni di Fenizio, 2007]. Employing this analysis interactively, a chemical reaction network is programmed to behave in the system level as desired. This "organization-oriented chemical programming" is envisioned as a design methodology for programming chemical reaction systems (Chapter 5, Section 5.1). The next two chapters are dedicated to an introduction of the theory of chemical organizations with the theoretical definitions (Chapter 1) following the description in Dittrich and Speroni di Fenizio [2007], and a simple case study of how the theory is applied to understand the dynamical behavior of chemical systems (Chapter 2).

0.2 Motivations to Chemical Programming

0.2.1 Bio-inspired Computing

In order to implement a technical system to be more reliable, adaptable, and fault tolerant, inspirations have been sought from biological systems because natural systems have been exhibiting those characteristics to us for many years. For instance, no image processing is, so far, more reliable than human face recognition mechanisms. Biological organisms have successfully adapted to ever-changing environments. Insects continue to walk even if a few legs malfunction, termed as graceful degradation [Kitano, 2002].

Although the inception of influence from nature may go back to the acceptance of decimal numerical systems, in accordance with the number of fingers [Buchholz, 1959], biologically inspired computing still has a rather long history, beginning in 1950s according to a "Hitchhiker's Guide" by Lodding [2004]. Inter- and intra-cellular signal transduction processes are compared with a computer network system to address network security problems [Krüger and Dressler, 2004], for example. Immune system mechanisms have been employed to implement autonomous intrusion detection systems [D'haeseleer et al., 1996]. Self-organizing collective behavior observed in ant colonies has led to a new method [Dorigo et al., 1999; Bonabeau et al., 2000] to balance the network load in telecommunication networks [Schoonderwoerd et al., 1996] or in a distributed peer-to-peer system [Montresor and Babaoglu, 2003]. Other biologically inspired techniques for distributed computing can be found in [Babaoglu et al., 2006].

These recent applications of biologically inspired methods are preferably focused on decentralized, asynchronous distributed computing systems though they are not restricted to that (*cf.* [Hjelmfelt et al., 1991; Bäck et al., 1997]). This tendency of distributed systems can be rationalized by general principles of biological systems. Natural systems perform complex, life-critical operations by employing an enormous number of simple, cooperating entities. The group of those entities behave in a self-organizing manner without central controlling mechanisms. Out of an orchestrated, decentralized interplay of those entities, system behavior emerges on a global level. The entities are diverse in scale. Governments and companies are considered to be an entity of a global political or economical system. Human beings form a social system, and so do ants, bees, and bacteria. On a smaller scale, proteins, chemical compounds, or molecules are the target agents to investigate biological behaviors of natural organisms.

Focusing on the biochemical scale, interactions between molecular entities are physiochemical interactions, and the graph theory has been successfully applied. Promoted as "Network Biology" by Barabási and Oltvai [2004], it is necessary to analyze biological systems not only by practical but also by theoretical means, in order to elucidate mechanisms of natural phenomena and design principles of the systems. For example, it has been found that metabolic networks from different organisms share common architectural designs to achieve simultaneously two conflicting features: scale-free toplogy and high connectivity. This network structure is presumed to be responsible for a robust and fault-tolerant system as it was discovered by evolution [Jeong et al., 2000]. The empirical network of genetic transcription, signal transduction, and the nervous system have a particular topological feature to perform signal processing resilient to noises [Klemm and Bornholdt, 2005]. Mechanisms such as modularity, feedback loops, sharing of common protocols, and hierarchical organization of subcomponents are helpful to explain robustness properties of living systems as shown by Csete and Doyle [2002] and Stelling et al. [2004]. To analyze the robustness or structural fragility of reaction networks, the minimum number of reactions whose absence causes deficiency of a certain functionality of the network can be computed [Klamt and Gilles, 2004]. Schuster et al. [2000] have considered other constraint based approaches, such as elementary modes that is the minimum number of reactions to sustain a certain functionality (e.g, a specific flux) in a complex metabolic network. To get an appropriate inspiration from biological systems, these approaches yield valuable perspectives.

With respect to computation, Conrad [1989] studied dissimilarity between conventional computers and brains in the light of programmability and evolvability. Digital computers are realizations of a highly programmable computational model, while the brain is less programmable and not programmable in a conventional imperative way. High programmability is achieved by conventional computer languages that imply constraints on system's operations [Conrad, 1995]. However, evolvability is remarkably impaired such that a small random change of a program causes the system to behave entirely differently or, in most cases, to stop working due to invalid operations. Looking at nature or natural organisms, on the other hand, evolvability is inevitable to cope with the ever-changing environment. Evolutionary processes are the primary forces for biological organisms of long-term adaptation to the environmental change. The achievement of evolvability at a high level has contributed to the successful existence of biological systems. The brain has been modeled with highly evolvable neural networks, which is featured by the alternation of connectivity between its neurons [Beale and Jackson, 1990; Ellacott and Bose, 1996].

Efficiency in computation is another aspect to consider. Processing time of perceived information, for example, is extremely significant for natural systems. In engineering, systems are carefully designed for fast computation and low energy consumption. In [Conrad, 1990], these three properties of programmability, evolvability or adaptability, and computational efficiency, are linked for the purpose of distinguishing any computational systems. His argument, then, leads to a trade-off principle stating that it is impossible to achieve programmability, evolvability or adaptability, and computational efficiency at the same time at high level. Outstanding achievement of computational efficiency on a basis of programmable components results in little evolutionary adaptable systems. In other words, programmability is the cost for the efficiency and for the adaptability, which are found to be important principles of biological information processing [Conrad, 1988]. It should be noted that the discussion about the trade-off is rather informal and conceptual.

Sipper et al. [1997] proposed another approach to classify bio-inspired computing systems along three axes: Phylogeny, Ontogeny, and Epigenesis. Phylogeny means, inheriting genetic information from parents through evolutionary process. Ontogeny describes the context sensitive development or growth of a creature. Epigenesis includes learning processes [Teuscher et al., 2003]. Their argument is that bio-inspired computing systems should achieve all of these properties at high level because biological systems do. Analyzing the computing systems in terms of those criteria would also be fruitful.

0.2.2 Chemical Computing

The chemical reaction metaphor has been proposed as a source of inspiration for a novel computation paradigm [Banâtre and Métayer, 1986; Dittrich, 2005] since all known life forms process information using chemical processes [Küppers, 1990]. Using chemical reactions for formal computations, described in a formal language, has initially been suggested by Banâtre and Métayer [1986]. In their GAMMA system [Banâtre and Métayer, 1990], a chemical reaction is defined as a rewriting operation on a multiset, mimicking a well-stirred reaction vessel. Chemical rewriting systems have been extended to chemical abstract machines, abbreviated as CHAM, by Berry and Boudol [1992] in order to capture the spatial context of chemical systems. Membrane computing, also known as P systems by Păun [2000], falls into a similar discipline. In this approach, the importance of membrane is stressed, by which the reaction vessel is compartmentalized. Further extension has been developed by Giavitto and Michel [2001]. Their system, named MGS, is capable of dealing with an arbitrary topological structure in the reaction vessel [Banâtre et al., 2004].

Another form of chemical computing is derived from DNA assembly processes, and a noticeable contribution by Adleman [1994] presented that combinatorial problems are aimed to be solved through the self-assembly processes of DNA molecules. Ehrenfeucht et al. [2003] have investigated a formal system to model that computational process, consisting of three operations on strings over the nucleotide alphabets. A significant distinction of this DNA-based computing includes the explicit intention to *in vivo* application [Weiss et al., 1999]. A successful implementation of a logic circuit, a toggle switch, using a gene regulatory network in *Escherichia coli* is reported by Gardner et al. [1999].

Advantages of chemical computing over conventional computing with respect to computational performance are still controversial. Chemical computing based on chemical reactions has another aspect, however. Biology, specifically Systems Biology [Kitano, 2002] employed chemical reaction networks to describe biological behaviors [Hucka et al., 2003], which is the same language as chemical computing. This interoperability is an advantage such that transfer knowledge between two fields is straightforward. We can adopt chemical reaction network models for chemical computing systems. Reversing the direction, we can apply concepts of chemical computing to biochemical systems so that biological organisms can be interpreted as systems that compute. Following a philosophy of Structural Sciences [Artmann, 2003], this interdisciplinary study of computer science and biology may reveal a common structure within life forms in terms of computation.

In chemical computing, the outcome of computation appears as an emergent global behavior based on a manifold of local interactions [Banzhaf et al., 1996]. There exists a leap over the gap between local interactions and global systems behavior for general cases of biological phenomena. Based on published experimental data, biochemical reaction networks have been constructed by combining single signaling pathways, and their emergent behavior has been investigated by Bhalla and Iyengar [1999]. The difficulties of developing biologically inspired systems are originated from this gap because looking at only isolated local parts of the system will not immediately lead to the understanding of the global behavior. Some tools to fill this gap are available. The dynamical systems theory provides tools to determine the steady states and other attractor states of a system including their stability [Tu, 1994] The bifurcation theory elucidates the dependency of the systems' behavior on the model parameters [Tyson et al., 2001]. However, theoretical analysis to reveal the relations between local entities' behavior and global emergent behavior is still lacking [Müller-Schloer, 2004]. In a non-trivial situation, it is impossible to predict the behavior by methods that are more efficient than simulations. Analysis of such behavior is hard because of the extensive influences between entities. Decomposition of chemical computing systems into subsystems becomes impractical, unlike conventional hierarchical systems compared in [Steels, 1990]. In other words, a composition of subcomponents does not necessarily preserve the well-defined sub-functionalities of each subsystem. “The wholeness is not a collection of the parts” [Anderson, 1972] in terms of functionalities. Thus, chemical computing systems demand users to program in a different manner from conventional algorithmic, imperative manner.

Chapter 1

Theory of Chemical Organizations

Contents

1.1 Motivations

Biochemical reaction networks in living cells are highly interconnected. In order to tackle their complexity, network theory has been successfully applied in their analysis [Barabási and Oltvai, 2004]. A systematic analysis of many biological organisms revealed design principles in metabolic networks that enhance robustness and fault-tolerance of these systems [Jeong et al., 2000]. Structural features contributing towards robustness in biological systems are for example feedback loops, redundancy, and modularity [Stelling et al., 2004].

Precise description of the dynamical behavior of these systems requires knowledge of the kinetics and the parameters for each reaction. However, several aspects of the dynamical behavior can already be inferred from the static structural information of the reaction network [Bailey, 2001]. Correlations between the stability of steady states and the stoichiometric matrix have been studied by Clarke [1975, 1980]. Under steady state assumptions, feasible flux distributions of metabolic networks are also obtained from the stoichiometry information [Schilling et al., 1999; Schuster et al., 2000]; and conclusions about equilibrium states and their uniqueness can then be drawn using methods developed by Feinberg and Horn [1974]. Further assuming a maximum bacterial growth rate, a metabolic network reconstructed from genome sequence data has been tested with experimental data [Edwards and Palsson, 2000]. When modelling biochemical reaction networks with petri nets [Petri, 1962; Reddy et al., 1993], the concepts of liveness, reachability, t-invariants, and p-invariants imply potential dynamical behaviors [Lautenbach, 1973].

An advantage of these approaches is that kinetic parameters, which are scarce in biological data, are not required. The analysis method described in the chapter operates on the same level of abstraction. That is, an algebraic analysis of the reaction network explains the dynamical behavior of the system. It is important to note that in our analysis, we abstract details like concentration levels or the spatial distribution of a chemical species. On this relatively high level of abstraction, a system state is characterized only by the molecular species present and we can describe the dynamics of a system qualitatively, namely, as a movement between sets of species, instead of a movement in a more complex state space [Speroni di Fenizio and Dittrich, 2002].

Inspired by Fontana and Buss [1994], Dittrich and Speroni di Fenizio [2007] have introduced chemical organization theory, aiming at an understanding of dynamical complex biochemical processes only taking stoichiometry into account. An *organization* is defined as a set of molecular species that is (algebraically) closed and (dynamically) self-maintaining. The first property, *closure*, ensures that applying any reaction rule to members of an organization generates its members only; the second property, *self-maintenance*, is a theoretical capability of an organization to maintain all of its members. Those two properties, independent of the type of reaction dynamics assumed, stabilize qualitative states of a reaction vessel: Neither new molecular species appear, nor does any existing molecular species disappear. When the theory is applied, a reaction network is decomposed into overlapping sub-networks, forming a partial hierarchy of organizations. The hierarchy is used to describe the potential dynamical behavior of the reaction system as a movement between organizations. Only stoichiometric information is required to identify all organizations, making the method well suited for biological networks where kinetic data is often scarce. In contrast to other methods, no steady state assumptions are made so that dynamical behaviors of accumulating mass are also considered. Note also that using the closure property alone can already provide a powerful tool to get insight into the structure and function of a large network consisting of several thousands of compounds [Ebenhöh et al., 2004].

1.2 Terminology and Notation

Before going further, we would like to specify terms in this section. A *reaction network* is given as a tuple $\langle \mathcal{M}, \mathcal{R} \rangle$, consisting of a set of molecular species $\mathcal{M}$ and a set of reaction rules $\mathcal{R}$. A *molecular species* $a \in \mathcal{M}$ is an object in $\mathcal{M}$. To write a *reaction rule* $\rho \in \mathcal{R}$, we adopt a notion from chemistry:

$$\begin{aligned} & l_{a_1,\rho}\, a_1 + l_{a_2,\rho}\, a_2 + \cdots + l_{a_{|\mathcal{M}|},\rho}\, a_{|\mathcal{M}|} \\ \to \quad & r_{a_1,\rho}\, a_1 + r_{a_2,\rho}\, a_2 + \cdots + r_{a_{|\mathcal{M}|},\rho}\, a_{|\mathcal{M}|}. \end{aligned} \tag{1.1}$$

Note that "+" is not an operator here but a separator of elements. The *stoichiometric coefficients* $l_{a,\rho}$ on the left-hand side and $r_{a,\rho}$ on right-hand side describe the amount of molecular species $a \in \mathcal{M}$ in reaction $\rho \in \mathcal{R}$ to be consumed and to be produced, respectively.

This formalization of the reaction network is so general that reaction rules that are not balanced, such as $a \to a + b$, $2a \to b$, or $b \to a$, are also allowed.

In chemistry, we usually demand that, within a chemical reaction, mass is conserved, i.e.the mass on the left-hand side of a reaction rule is equal to the mass on its right-hand side. Chemical organization theory has also been intended to handle systems that are not balanced and where mass is not necessarily conserved. A simple biological model such as HIV model investigated in Chapter 2 is an example of a non mass-conserving reaction network.

We also define two mappings: $\mathrm{LHS}(\rho) \equiv \{a \in \mathcal{M} : l_{a,\rho} > 0\}$ and $\mathrm{RHS}(\rho) \equiv \{a \in \mathcal{M} : r_{a,\rho} > 0\}$, returning the set of species with a positive coefficient on the left-hand and right-hand side, respectively. The left-hand side species of reaction ρ given by $\mathrm{LHS}(\rho)$ are also called *reactants*, and the right-hand side species by $\mathrm{RHS}(\rho)$ *products*. These mappings omit the species irrelevant to the reaction, namely the coefficients on both sides are zero. Besides, the stoichiometric coefficients are also disregarded in the set returned by the mappings. Applicability of a reaction is considered only whether the combination of species, neglecting the stoichiometry, is satisfied. Formally, for reaction ρ to occur in $A \subseteq \mathcal{M}$, the necessary condition is $\mathrm{LHS}(\rho) \subseteq A$. A reaction is *applicable* to a set of species only when this condition is fulfilled.

Given a reaction network, organizational structure embedded within the network is scrutinized. Analyzing the reaction network using chemical organization theory, the network is decomposed into overlapping sub-networks, consisting of molecular species, called organizations, whose specification is given later in Definition 3. Prescribing the size of set as the number of species contained, organizations are arranged vertically with respect to their size. The largest organizations are placed at the top. A hierarchy of organizations is formed and is denoted as *organizational structure* of the reaction network.

A central element to describe chemical reaction systems is the stoichiometric matrix, which can be used to derive an ordinary differential equation model for the dynamics of the system based on, e.g., mass action kinetics. Writing a reaction rule as in Equation 1.1, the *stoichiometric matrix* $\mathbf{S}$ can be defined as:

$$\mathbf{S} = (s_{a,\rho}) = (r_{a,\rho} - l_{a,\rho}). \tag{1.2}$$

An entry $s_{a,\rho}$ of the stoichiometric matrix denotes the net amount of species a to be produced in reaction ρ. A negative entry indicates the species to be consumed in the reaction. A zero entry signifies no amount change of the species in the reaction, but it does not always mean that the species is irrelevant. In case $r_{a,\rho} = l_{a,\rho} > 0$, molecular species a does participate in reaction ρ, especially as the reactant, such that the reaction can take place. Only when $r_{a,\rho} = l_{a,\rho} = 0$, molecular species a can be neglected with respect to reaction ρ. Both cases are symbolized as the zero entry in the matrix $\mathbf{S}$ and are indistinguishable in the stoichiometric matrix representation of the reaction networks.

The dynamics of chemical reaction systems can be defined by an ordinary differential equation system in terms of the dynamics of concentration vector $\mathbf{x} = (x_a) \in \mathbb{R}^{|\mathcal{M}|}$. Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, a *reaction system* is an instance of the reaction network. A reaction system can be realized by putting a multiple copies of some molecular species from set $\mathcal{M}$ into a reaction vessel, sometimes called a *reactor*. The reaction vessel populated with *molecules*, instances of a species type in $\mathcal{M}$, changes dynamically the contents based on reaction rules given in $\mathcal{R}$. The concentration profile of the contents corresponds to the state of the reaction system, and dynamical change of that profile forms a trajectory in the concentration vector space. This trajectory is the dynamics an ordinary differential equation describes.

In passing, we use the terms "molecular species" and "species" synonymously, and the term "molecule" addresses a concrete object of a certain species. Species form a reaction network, and molecules reside in a reaction vessel to form a reaction system. The difference between species and molecule is similar to the difference between class and instance. A species represents a template or class of molecules, and molecules are multiple copies or instances of a species.

Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, the dynamics of the chemical reaction systems instantiated from the network can be described by an ordinary differential equation (ODE) system as follows:

$$\dot{\mathbf{x}} = \mathbf{S}\mathbf{v}(\mathbf{x}) \tag{1.3}$$

where $\mathbf{x} = (x_1, \ldots, x_{|\mathcal{M}|})^T$ is a concentration vector and $\mathbf{v}(\mathbf{x}) = (v_1(\mathbf{x}), \ldots, v_{|\mathcal{R}|}(\mathbf{x}))^T$ is a flux vector describing the reaction speed. When the mass action kinetics is assumed, an entry v_ρ of the flux vector for reaction $\rho \in \mathcal{R}$ is simply a product of concentrations of the reactants:

$$v_\rho = k_\rho \left(\prod_{a \in \mathcal{M}} x_a^{l_{a,\rho}} \right) \tag{1.4}$$

where $k_\rho \geq 0$ is a kinetic parameter. $\dot{x}_a = dx_a/dt$ denotes the element of $\dot{\mathbf{x}}$ describing the speed of concentration change of species a, referred as *production rate*. We may simply write the production rate of molecule a as $(\mathbf{S}\mathbf{v})_a$.

The concentration profile of the reactor contents corresponds to the *quantitative state* of the reaction system. We also refer the *qualitative state* as a set of molecular species present in the reaction system, whose concentration values are higher than a certain threshold. Typically, the threshold is set to zero or very small value. Using abstraction mapping defined later in Definition 4, the qualitative state is an abstraction of quantitative states with a threshold of zero or small value.

1.3 Core Theory

We now define the central concept of the theory, namely organization as a closed and self-maintaining set of molecular species. Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, properties of closure and self-maintaining are scrutinized for each set of molecular species in order to derive the organizational structure embedded within the network. A set of molecular species $A \subseteq \mathcal{M}$ is *closed* if, for all reactions ρ applicable on A, the products are contained in A, that is, $\mathrm{RHS}(\rho) \subseteq A$. In other words, there exists no reaction rule producing new molecular species not yet present in the organization using only species of that organization. This is similar to the algebraic closure of an operation in set theory.

Definition 1 (closure). *Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, a set of molecular species $A \subseteq \mathcal{M}$ is closed if, for every reaction $\rho \in \mathcal{R}$ with $\mathrm{LHS}(\rho) \subseteq A$, $\mathrm{RHS}(\rho) \subseteq A$ also holds.*

The closure property is conceived to assure that no new molecular species can appear, which would change the qualitative state of the reaction systems. Another force of the qualitative change is the disappearance of molecular species from the reaction vessels. The second property has been thought to address that change. Self-maintenance property is a theoretical capability of an organization to maintain all of its members. Loosely speaking, all molecular species that are used up within the set can also be reproduced by a reaction or some reaction pathways among species of that set. The general definition of self-maintenance becomes more complicated than that of closure because the reproduction aspect can be associated with many reaction processes. The maintenance of one molecular species can be supported by dynamical productions and consumptions of other molecular species, and many molecular species may operate and contribute as a whole in a complex pathway. The stoichiometry of the whole reaction network must be taken into consideration.

Formally, the definition of *self-maintenance* of a set of molecular species is given as follows:

Definition 2 (self-maintenance)**.** *Given an reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, a set of molecules species $B \subseteq \mathcal{M}$ is self-maintaining if there exists a flux vector $\mathbf{v} \in \mathbb{R}^{|\mathcal{R}|}$ satisfying the following three conditions:*

1. $v_\rho > 0$ *if* $\mathrm{LHS}(\rho) \subseteq B$
2. $v_\rho = 0$ *if* $\mathrm{LHS}(\rho) \nsubseteq B$
3. $(\mathbf{Sv})_a \geq 0$*,* $a \in B$*.*

These three conditions can be read as follows: The flux v_ρ must be positive when reaction ρ is applicable to the set B (Condition 1). All other remaining fluxes are set to zero (Condition 2). Finally, the production rate $(\mathbf{Sv})_a$ for all molecular species $a \in B$ must be nonnegative (Condition 3). Note that we have to find only one such flux vector in order to show whether a set is self-maintaining.

Closure and self-maintenance together define an organization:

Definition 3 (organization)**.** *A set of molecular species $O \subseteq \mathcal{M}$ that is closed and self-maintaining is called an organization.*

The organizational analysis decomposes the reaction network given into overlapping sub-networks, organizations. We visualize the set of all organizations by a Hasse-diagram (e.g., Figure 2.1 lower right corner, Figure 3.1). In that diagram, organizations are arranged vertically according to their size, that is, the number of species contained. The largest organizations, which are formed by a maximum number of species, are placed at the top of the Hasse-diagram. Two organizations are connected by a line if the lower organization is contained in the organization above and there is no other organization in between. The Hasse-diagram represents the hierarchical organizational structure of the reaction network under study.

1.4 Dynamical Part

For deriving the Hasse-diagram of organizations no detailed knowledge concerning the dynamics is required. Only stoichiometric information, i.e., the set of reaction rules, is sufficient. Therfore that part of chemical organization theory is referred as the static part. In the "dynamical part", the set of organizations is used to describe the dynamics of a reaction system as a movement between organizations. For further details see the original paper by Dittrich and Speroni di Fenizio [2007].

1.4.1 Fixed Points are Instances of Organizations

An important relation between organizations and dynamical behavior is stated by a theorem saying that every fixed point must be an instance of an organization [Dittrich and Speroni di Fenizio, 2007]. In other words, in a continuous dynamical reaction system given by an ODE, we can only obtain a steady state, or a fixed point, with a combination of molecular species that is an organization. This theorem links the static analysis of a network structure with a dynamical aspect of reaction systems. For presenting the theorem formally, the abstraction function, the opposite concept of instance, is introduced.

Abstraction is a mapping $\phi : X \mapsto \mathcal{M}$, which maps a state of the system to the set of molecular species that are present in the system. When the threshold Θ is 0 or very small, the abstraction of a state $\phi(x)$ corresponds to the qualitative state of the reaction system.

Definition 4 (abstraction)**.** *Given a dynamical system $\dot{\mathbf{x}} = f(\mathbf{x})$ and let $\mathbf{x}$ be a state in X, then the abstraction $\phi(\mathbf{x})$ is defined by*

$$\phi(\mathbf{x}) = \{i | x_i > \Theta, i \in \mathcal{M}\}, \quad \phi : X \to \mathcal{P}(\mathcal{M}), \quad \Theta \geq 0 \tag{1.5}$$

where $x_i \geq 0$ is the concentration of molecular species i in state $\mathbf{x}$, and Θ is a threshold chosen such that it is smaller than any positive coordinate of any fixed point of the dynamical system.

While $\phi(\mathbf{x})$ denotes the molecular species represented by the state $\mathbf{x}$, an *instance* of a set A is a state where exactly the molecules from A are present. We say that a state $\mathbf{x} \in X$ is an instance of $A \subseteq \mathcal{M}$ iff $\phi(\mathbf{x}) = A$.

Theorem 1. ***H**ypothesis: Let us consider a general reaction system whose reaction network is given by the reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ and whose dynamics is given by a differential equation $\dot{\mathbf{x}} = \mathbf{S}\mathbf{v}(\mathbf{x}) = f(\mathbf{x})$ as defined before. Let $\mathbf{x}' \in X$ be a fixed point, that is, $f(\mathbf{x}') = \mathbf{0}$, and let us consider a mapping ϕ as given by Definition 4, which assigns each state $\mathbf{x}$ to a set of molecular species. **T**hesis: $\phi(\mathbf{x}')$ is an organization. (Proof see Ref. Dittrich and Speroni di Fenizio [2007].)*

1.4.2 Consistent Reaction Networks

When analyzing the organizational structure of the reaction network, a functional procedure called *generate* may be utilized (specifically, in Chapter 7). As a part of the chemical organization theory by Dittrich and Speroni di Fenizio [2007], there are three kinds of such functions: $G_{CL}(S)$, $G_{SM}(S)$, and $G_{org}(S)$. Every function accepts an arbitrary set S of molecular species as an input, and the set is converted, in accordance with the given reaction network, to a set of molecular species with a desired property of closure, self-maintenance, or both, respectively. Classifying reaction systems depending on the behavior of those generate functions is speculated. One of distinctive behaviors is the uniqueness of the generated self-maintaining set and establishes a classification of *consistent* reaction networks.

Definition 5 (consistent)**.** *A reaction network is called* consistent*, if the closure and self-maintaining set generated by a set can uniquely be defined, i.e. given any set $S \subseteq \mathcal{M}$, the smallest closed set that contains S and the largest self-maintaining set contained in S are unique, respectively.*

Definitions of the three generate functions are following. with generate closure function. To generate the closure of a set, we expand it by interacting the molecules of the set and adding to the set any newly generated molecule. When no new molecule is generated, the set is closed.

Definition 6 (generate closed set)**.** *Given a set of molecules $S \subseteq \mathcal{M}$, we define $G_{CL}(S)$ as the smallest closed set C containing S. We say that S generates the closed set $C = G_{CL}(S)$ and we call C the closure of S.*

We can now define a generate operator for self-maintaining sets in the same way as for closed sets by saying that the self-maintaining set generated by a set S is the biggest self-maintaining set D contained in S.

Definition 7 (generate self-maintaining set)**.** *Given a consistent reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ and a set of species $S \subseteq \mathcal{M}$, we define $G_{SM}(S)$ as the biggest self-maintaining set D contained in S. We say that S generates the self-maintaining set $D = G_{SM}(S)$.*

Given a set S, $S \subseteq \mathcal{M}$, it is always possible to generate its closure [Speroni di Fenizio et al., 2000]. In a consistent reaction network, by definition, we can always generate uniquely for any given set $S \subseteq \mathcal{M}$ a self-maintaining set D by taking the biggest self-maintaining set that contains S. Since in consistent reaction systems both closure and self-maintaining set can be generated uniquely, we can also uniquely define the organization generated by a set S:

Definition 8 (generate organization)**.** *Given a set of molecules $S \subseteq \mathcal{M}$, the organization $O = G_{org}(S)$ generated by S is defined as $G_{org}(S) \equiv G_{SM}(G_{CL}(S))$.*

If O is an organization $G_{org}(O) = O$. The organizations are the fixed points of the "generate organization operator" G_{org}.

1.5 Summary and Discussion

This section summarizes how the theory of chemical organization is typically applied to analyze a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$: First, all organizations within the given reaction network are determined. The set of organizations decomposes the network into overlapping sub-networks. The hierarchy of the organizations is referred to as the organizational structure of the reaction network. The hierarchy or the organizational structure is visualized by a Hasse-diagram.

The derived organizational structure is interpreted using Theorem 1 in relation to the dynamical behavior of the reaction system. That theorem states that the abstraction of a fixed point fulfills the conditions to be an organization.

Reversing the argument, the set of organizations indicates potential combinations of species that may constitute fixed points. Analyzing the organizational structure suggests which set of molecular species may be contained in a steady state. Since dynamical reaction systems tend to be in a steady state after a sufficiently long time, it is likely that the set of species, when observed, is an organization (if the vessel is sufficiently large). Assuming dynamical systems' behaviors are characterized by the set of species present in the reaction vessel, the hierarchy of organizations will give, therefore, an overview of the potential behaviors of the reaction system. It is this characteristics that we utilize for computation in chemical computing.

The set of organizations and the set of fixed points are not in one-to-one correspondence, however. Distinctive multiple fixed points may be mapped by the abstraction function to the same organization. Because of the qualitative nature of the analysis, quantitatively different fixed points can belong to the same set of species. There may also exists an organization that is not an abstraction of any fixed point. The reverse of the thesis is not true in general, that is, an organization is not necessarily an abstraction of a fixed point. A derived organization serves only as a candidate of the abstraction of a fixed point.

Since the core part of the theory is developed based only on the algebraic structure of the reaction network, some aspects of dynamical behaviors such as stability of fixed points, oscillatory behaviors, or periodic attractors, are not addressed directly using the notion of organizations at the moment. Moreover, a quantitative value of, for instance, concentration is neglected because of the qualitative nature of the analysis method. The quantity of a molecular species is, depending on the threshold parameter Θ, abstracted to be qualitatively present (or high concentrations) or absent (or low concentrations). These dynamical aspects are mainly dependent on kinetic laws or parameters of reactions. Such information is not always available and mostly scare in biology, however. To compensate the unavailability of data and applicability of the theory, dynamical aspects of systems behaviors are initially regarded less. This highly abstracted view still provides us with meaningful perspectives of biological organisms [Barabási and Oltvai, 2004]. Relations between the organizations and those aspects are still under study.

Source :

Matsumaru, N., Speroni di Fenizio, P., Centler, F., and Dittrich, P. (2005b). A case study of chemical organization theory applied to virus dynamics. In Kim, J. T., editor, *Systems Biology Workshop at ECAL 2005, Workshop Proceedings CD-ROM*, Kent, UK.

Matsumaru, N., Centler, F., Speroni di Fenizio, P., and Dittrich, P. (2006a). Chemical organization theory applied to virus dynamics. *it - Information Technology*, 48(3):154–160.

Chapter 2

Case Study: Analysis of Virus Dynamics Model

Contents

2.1 HIV Immunological Population Dynamics Model

In order to evaluate the usefulness of chemical organization theory, we apply it to a model by Wodarz and Nowak [1999] describing the interaction of a virus (HIV) with immune system cells [Matsumaru et al., 2005]. The model has been developed in order to explain the efficacy of various drug treatment strategies. Especially it shows, why a drug treatment strategy does not try to remove the virus, but aims at stimulating the immune defense, such that the immune system controls the virus at low but positive quantities. The aim of this chapter is to show that chemical organization theory can reveal, even in such relatively small models, a structure (lattice of organizations), which can be used to describe the dynamics of the model and to explain the strategy of a drug treatment from a different perspective.

2.1.1 Model Description

In the model, there are four molecular species: uninfected CD4$^+$ T cells x, infected CD4$^+$ T cells y, cytotoxic T Lymphocyte (CTL) precursors w, and

CTL effectors z:

$$\mathcal{M} = \{\mathsf{x}, \mathsf{y}, \mathsf{w}, \mathsf{z}\}. \tag{2.1}$$

The concentration of each species is specified by x, y, w, and z, respectively. Wodarz and Nowak [1999] define the dynamics as an ordinary differential equation (ODE) system with kinetic parameters $a, b, c, d, h, p, q, \beta$, and λ:

$$\begin{array}{rcl} \dot{x} & = & \lambda - dx - \beta xy \\ \dot{y} & = & \beta xy - ay - pyz \\ \dot{w} & = & cxyw - cqyw - bw \\ \dot{z} & = & cqyw - hz \end{array} \tag{2.2}$$

From the given deterministic ODE model we derive chemical reaction rules, which form a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, so that the theory can be applied: The ODE model includes a decay term for each species. Thus, for each species we have a reaction rule transforming each molecular species into the empty set: $\mathsf{x} \to \emptyset$, $\mathsf{y} \to \emptyset$, $\mathsf{w} \to \emptyset$, and $\mathsf{z} \to \emptyset$. Since the concentration of uninfected CD4$^+$ T cells x increases with a constant rate λ, molecular species x is considered as an input (or inflow) species, resulting in a reaction rule $\emptyset \to \mathsf{x}$.

The infection by HIV viruses transforms a T cell x into an infected T cell y, which is denoted by the term βxy in the ODE model and the reaction rule $\mathsf{x}+\mathsf{y} \to 2\mathsf{y}$. The destruction of the infected CD4$^+$ T cells y by the CTL effectors z as specified by the term pyz is transformed into $\mathsf{y}+\mathsf{z} \to \mathsf{z}$. Note that this is a catalytic reaction with respect to the effector z because the concentration of z is not effected by this reaction. CTL precursor w multiplies with the catalytic help of both the infected and uninfected CD4$^+$ T cell in accordance with the term $cxyw$, which results in the reaction rule $\mathsf{x}+\mathsf{y}+\mathsf{w} \to \mathsf{x}+\mathsf{y}+2\mathsf{w}$. When the infected CD4$^+$ T cell y is detected by the CTL precursor w, the precursors differentiate into effectors z as captured by the rule $\mathsf{y}+\mathsf{w} \to \mathsf{y}+\mathsf{z}$. The corresponding term in the ODE model is $cqyw$. The whole set of reaction rules is listed in Table 2.1 The network derived from the HIV model investigated in this chapter is "fortunately" consistent so that we can always generate uniquely a self-maintaining set D for any given set $S \subseteq \mathcal{M}$.

A graphical representation of the network is shown in Figure 2.1, upper right corner. Since the number of molecular species $|\mathcal{M}|$ is four and the size of the reaction rule set $|\mathcal{R}|$ is nine, the stoichiometric matrix $\mathbf{S}$ is the following:

$$\mathbf{S} = \begin{array}{c} \mathsf{x} \\ \mathsf{y} \\ \mathsf{w} \\ \mathsf{z} \end{array} \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

where each row corresponds to a molecular species (x, y, w, z from the top) and each column corresponds to a reaction. As can be seen, the stoichiometric matrix does not contain all information of the reaction network. For example, the reaction rule $\mathsf{y}+\mathsf{z} \to \mathsf{z}$ appears only as the column vector $(0, -1, 0, 0)^T$. Molecular species y catalyzes the decay of z with no change of stoichiometric coefficients. Catalysts are neglected in the stoichiometric matrix representation.

Table 2.1: List of reaction rules modeling the response of the immune system to HIV infection, derived from ODE model developed by Wodarz and Nowak [1999]. See text for details.

ID	Reaction rules			Dynamics	Description
1	$\emptyset$	$\rightarrow$	x	λ	inflow
2	x	$\rightarrow$	$\emptyset$	dx	decay
3	y	$\rightarrow$	$\emptyset$	ay	decay
4	w	$\rightarrow$	$\emptyset$	bw	decay
5	z	$\rightarrow$	$\emptyset$	hz	decay
6	$\mathsf{x} + \mathsf{y}$	$\rightarrow$	$2\mathsf{y}$	βxy	HIV infection
7	$\mathsf{y} + \mathsf{z}$	$\rightarrow$	z	pyz	immune counteraction
8	$\mathsf{x} + \mathsf{y} + \mathsf{w}$	$\rightarrow$	$\mathsf{x} + \mathsf{y} + 2\mathsf{w}$	$cxyw$	effector production
9	$\mathsf{y} + \mathsf{w}$	$\rightarrow$	$\mathsf{y} + \mathsf{z}$	$cqyw$	activation of CTL

2.1.2 Lattice of Organizations

For applying the theory we check every possible set of species (i.e., 16 sets) whether it is closed and self-maintaining. As a result we found three organizations. The Hasse-diagram is depicted in Figure 2.1, lower right corner. The smallest organization consists only of the "healthy cells" x (uninfected CD4$^+$ T cells). There cannot be a smaller organization (i.e. the empty set) because x is an input species and therefore the empty set is not closed.

Looking at the reaction rules, we can see that x alone cannot produce anything else, and thus the set $\{\mathsf{x}\}$ is closed. Since x is an input species, the set $\{\mathsf{x}\}$ is obviously self-maintaining. We can also show formally that $\{\mathsf{x}\}$ is self-maintaining. A flux vector, $\mathbf{v} = (1,1,0,0,0,0,0,0,0)^T$, is chosen such that all conditions given in Definition 2 are fulfilled: all flows within the organization are positive, all flows outside are zero, and the production rates $\mathbf{Sv} = (0,0,0,0,0,0,0,0,0)^T$ are not negative in any component of the set $\{\mathsf{x}\}$.

The second organization, $\{\mathsf{x},\mathsf{y}\}$, contains "healthy cells" x together with "ill cells" (infected CD4$^+$ T cells). Looking at the reaction network, we can see that $\{\mathsf{x},\mathsf{y}\}$ is closed, because there is no reaction rule that allows to produce w or z just using x and y alone. We can show that $\{\mathsf{x},\mathsf{y}\}$ is self-maintaining with the flux vector $\mathbf{v} = (10,1,1,0,0,1,0,0,0)^T$ and the production rate $\mathbf{Sv} = (8,0,0,0)^T$.

The largest organization contains all species and is thus obviously closed. From the reaction rules, we can see that because x can be produced at an arbitrarily high rate, we can also produce y, z, and w at arbitrarily high rates. For example, choosing $\mathbf{v} = (100,1,1,1,50,1,50,10)^T$, $\mathbf{Sv}$ is positive in all components and therefore $\{\mathsf{x},\mathsf{y},\mathsf{w},\mathsf{z}\}$ is self-maintaining.

2.1.3 Discussion

From a mathematical analysis conducted by Wodarz and Nowak [1999] and simulation studies by Matsumaru et al. [2004] it is known that the model has two modes of behavior belonging to two asymptotically stable fixed points: One

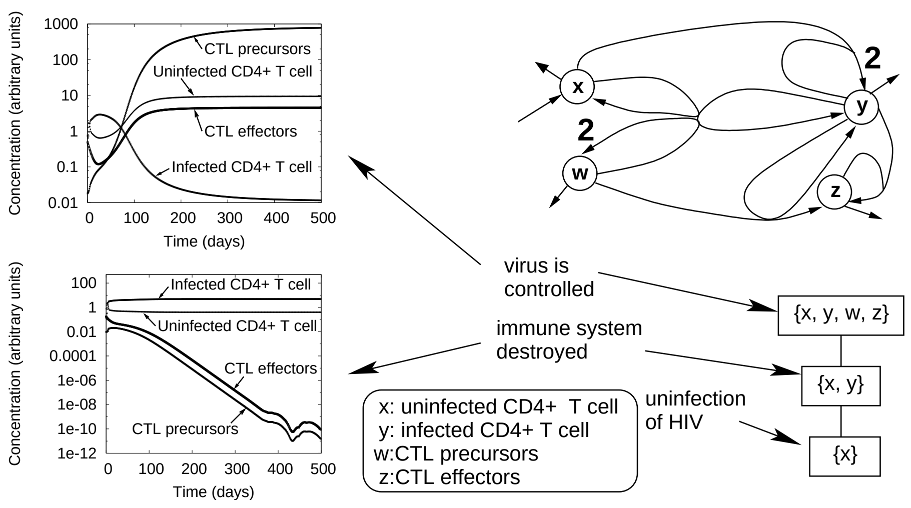


Figure 2.1: Illustration of the analysis of the HIV immunological response model by Wodarz and Nowak [1999]. The ODE model given in Eq. 2.2 is transformed to a chemical reaction network (right top). The resulting hierarchy of organizations is shown as a Hasse-diagram (right bottom). Two of the organizations represent the attractors: virus under control (top) and immune system destruction (middle). Dynamic simulations leading to both attractors are shown on the left. Parameters were taken from $\lambda = 1$; $d = 0.1$; $\beta = 0.5$; $a = 0.2$; $p = 1$; $c = 0.1$; $b = 0.01$; $q = 0.5$; $h = 0.1$. Initial concentrations for left, top plot: $x = 0.74$; $y = 0.75$; $w = 0.018$; $z = 0.49$. Initial concentrations for left, bottom plot: $x = 0.75$; $y = 0.14$; $w = 0.0095$; $z = 0.17$.

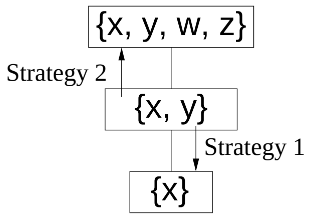


Figure 2.2: Illustration of two treatment strategies. Strategy 1 tries to remove HIV entirely from the system. Strategy 2 aims at establishing a long-term CTL-mediated control of viral load.

of the attractors is characterized by high virus load and no CTL precursors and effectors present. This state is interpreted as the complete destruction of the immune defense. The organization $\{\mathsf{x}, \mathsf{y}\}$ represents this attractor. When the HIV virus is controlled by the immune defense, all four molecular species are present in the system, constituting the other attractor. This state is reflected in the largest organization $\{\mathsf{x}, \mathsf{y}, \mathsf{w}, \mathsf{z}\}$. The smallest organization $\{\mathsf{x}\}$ can be interpreted as the condition where neither HIV virus nor the infected CD4$^+$ T cell is present.

After identifying the lattice of organizations, we would like to show that the strategy of a drug treatment can be explained on a relatively high level of abstraction, omitting details, using the lattice of organizations. The aim of the treatment is the movement from an organization representing an ill state to an organization representing a healthy state. Looking at the lattice of organizations of the model by Wodarz and Nowak [1999], we can envisage two strategies for a drug therapy illustrated in Figure 2.2. The first one tries to move the system into the smallest organizations $\{\mathsf{x}\}$, where no virus is present at all. An alternative strategy may move the system into the largest organization, where the virus is present but also an immune system takes control over the virus infection.

There are drugs available that can bring down the virus load by several orders of magnitude.If by this procedure the virus could be completely removed, the system would move into the smallest organization because the set $\{\mathsf{x}, \mathsf{w}, \mathsf{z}\}$ is not self-maintaining so that the system moves down into organization $\{\mathsf{x}\}$. Using a term defined in Definition 8, the set $\{\mathsf{x}, \mathsf{w}, \mathsf{z}\}$ generates organization $\{\mathsf{x}\}$. It has been observed, however, that the virus cannot be fully removed even though the virus load can be decreased below detection limit. After stopping the treatment, the virus appears again. Therefore, the actual strategy of a drug therapy is not to move the system into the lowest organization but into the highest organization. In practice, this is achieved by applying the drug periodically allowing the immune defense to increase [Wodarz and Nowak, 1999].

It is important to note that choosing the right level of abstraction depends on what should be explained. The lattice of organizations is a suitable level of abstraction for describing the overall strategy. However, *how* an actual drug treatment should look like in order to move the system into the largest organization cannot be answered by chemical organization theory. For this we have to chose a lower, more detailed, level of abstraction, e.g., as in the ODE model, which provides information on how the system can move from one organization to another.

2.2 Comparison of HIV Models

The theory of chemical organization can be used to compare models [Matsumaru et al., 2006a]. The results of analyzing five viral dynamics models with chemical organization theory are summarized in Table 2.2. The level of abstraction increases from left to right. Moving from right to left requires additional information, e.g., reaction kinetics to construct an ODE model from the network model. Depending on the purpose of the model, the appropriate level must be chosen carefully. Exact quantitative analysis of the model behavior is possible with ODE models, but estimating kinetic parameters is critical as pointed out by Wu et al. [1999].

2.2.1 Basic Model

The HIV infection process involves mainly three molecular species: uninfected T cells x, infected T cell y, and free virus particles v. Provided that the concentration of each species is specified by x, y, and v, respectively, the infection dynamics can be modeled as follows [Nowak and Bangham, 1996]:

$$\begin{aligned}\dot{x} &= \lambda - dx - \beta xv \\ \dot{y} &= \beta xv - ay \\ \dot{v} &= ky - uv.\end{aligned}$$

Since the deterministic ODE model is contrived on a basis of interactions between molecular species, the infection process can be represented as a form of chemical reaction network. Since uninfected T cells are produced at a constant rate λ, molecular species x is considered as an input species, resulting in the reaction rule: $\emptyset \rightarrow \mathsf{x}$. Each species is assumed to decay in the ODE model. As a reaction rule, each species is transformed into the empty set: $\mathsf{x} \rightarrow \emptyset$, $\mathsf{y} \rightarrow \emptyset$, and $\mathsf{v} \rightarrow \emptyset$.

The infection with HIV transforms an uninfected T cell x into an infected cell y, which is denoted by the term βxv. This interaction can be represented by the reaction rule: $\mathsf{x} + \mathsf{v} \rightarrow \mathsf{y} + \mathsf{v}$. Since variable v is not changed by that term, virus species v is also included in the right-hand side of the reaction rule. The last term to consider is ky representing the virus replication in the infected cell: $\mathsf{y} \rightarrow \mathsf{y} + \mathsf{v}$. Alternatively we can simply write $\mathsf{y} \rightarrow \mathsf{v}$ instead of two reactions because infected T cell y decays and, at the same time, produces virus v. With respect to the theory, it does not change the result.

Computing the closed and self-maintaining sets of molecular species in the HIV infection networks reveals the existence of two organizations. The smaller organization consisting of only uninfected T cell x represents as the states without virus infection. The larger organization contains all three molecular species and corresponds to the infected state. From mathematical analysis [Nowak and Bangham, 1996] it is known that the ODE model has two equilibrium states. These two states correspond to the two organizations of the network. This demonstrates that chemical organization theory delivers a proper analysis of the reaction network regarding its dynamical behavior.

2.2.2 CTL Response

The HIV infection model is extended to include immune responses by Nowak and Bangham [1996] by adding a new decay term for the infected T cell y:

$$\dot{y} = \beta xy - ay - pyz$$

where variable z represents the concentration of the cytotoxic T Lymphocyte (CTL) species z. The dynamics of CTL is given as:

$$\dot{z} = cyz - bz.$$

Upon detection of infected T cells y, CTL proliferates at rate cyz.

In addition to the reaction rules from the previous model, a decay reaction for CTL z and two further reactions are derived from the ODE model. The collision of infected T cell y and CTL z can have two outcomes: annihilation of infected cells ($\mathsf{y} + \mathsf{z} \rightarrow \mathsf{z}$) or proliferation of the CTL cells ($\mathsf{y} + \mathsf{z} \rightarrow \mathsf{y} + 2\mathsf{z}$). Analyzing the reaction network of nine reactions, the hierarchy of organizations contains three levels. The new species z is only involved in the largest organization. The two lower organizations are identical with the organizations of the previous model. The top organization corresponds to the equilibrium of the ODE model in which CTL immune response is continuously activated. According to a mathematical analysis [Nowak and Bangham, 1996], the activation of the immune system may be temporal if the concentration of the infected cell is smaller than a threshold value. This equilibrium with infected cells but without immune response is contained in the middle organization. The smallest organization at the bottom of the hierarchy represents the state with no infection.

2.2.3 Memory CTL

An ODE model with four molecular species is constructed in [Wodarz and Nowak, 1999]: uninfected CD4$^+$ T cells x, infected CD4$^+$ T cells y, CTL precursors w, and CTL effectors z. The concentration of each species is specified by x, y, w, and z, respectively. The dynamics is as follows:

$$\begin{aligned}\dot{x} &= \lambda - dx - \beta xy\\ \dot{y} &= \beta xy - ay - pyz\\ \dot{w} &= cxyw - cqyw - bw\\ \dot{z} &= cqyw - hz.\end{aligned}$$

A set of chemical reaction rules is derived from the ODE model. Since the virus species is omitted in the model, the virus infection occurs when infected cells attach to the uninfected: $\mathsf{x} + \mathsf{y} \rightarrow 2\mathsf{y}$. The CTL precursor differentiates to CTL effector on contact with virus infected T cells: $\mathsf{y} + \mathsf{w} \rightarrow \mathsf{y} + \mathsf{z}$, and the CTL effector kills infected T cells: $\mathsf{y} + \mathsf{z} \rightarrow \mathsf{z}$. In accordance with the term $cxyw$, proliferation of the CTL precursor is also dependent on both infected and uninfected T cells: $\mathsf{x} + \mathsf{y} + \mathsf{w} \rightarrow \mathsf{x} + \mathsf{y} + 2\mathsf{w}$. Despite the changes in the model, we found no major differences in the reaction network with respect to the organizational structure.

2.2.4 Quiescent Cell

Since the target T cell must be activated to be susceptible to infection, a model including the resting cell has been analyzed in [Callaway et al., 1999] and was simplified as follows [Callaway and Perelson, 2002] :

$$\begin{aligned}\dot{Q} &= \lambda - d_Q Q - \theta(v + B)Q\\ \dot{x} &= s\theta(v + B)Q - dx\\ &\quad -(1-\kappa)kvx\\ \dot{y} &= (1-\kappa)kvx - ay\\ \dot{v} &= ky - uv.\end{aligned}$$

Variable Q represents the concentration of quiescent cell species Q and variable B represents the concentration of any other antigen B than HIV v. The quiescent cell is activated by both HIV and other antigens into a T cell uninfected at rate $\theta(v + B)$, and the activation is written in form of a reaction rule as follows: $\mathsf{Q} + \mathsf{v} \rightarrow \mathsf{x} + \mathsf{v}, \mathsf{Q} + \mathsf{B} \rightarrow \mathsf{x} + \mathsf{B}$. Additionally, the chemical reaction network derived from the ODE model is composed of the infection process by HIV ($\mathsf{x} + \mathsf{v} \rightarrow \mathsf{y} + \mathsf{v}$), virus proliferation ($\mathsf{y} \rightarrow \mathsf{y} + \mathsf{v}$), decay reactions, and an influx of Q. In passing, this ODE model considers also drug therapy with a reverse transcriptase inhibitor, and the efficacy of the drug is represented by $0 \leq \kappa \leq 1$.

Analyzing the reaction network with the theory of chemical organizations reveals four organizations. Since the quiescent cell has an influx, the smallest organization is the set $\{\mathsf{Q}\}$. Directly above it, there are two distinct organizations. The one with four molecular species corresponds to the activation of the quiescent cell by HIV v and the infection of activated T cells x. Once the quiescent cell is transformed into the uninfected T cell, the HIV infects the T cell. At the same time, the infected T cell is necessary for the virus to reproduce. Thus, the infected T cell y is also part of the organization so that the species set becomes closed and self-maintaining. The other organization indicates the activation of the quiescent cell by the other antigen B. Both organizations contain the activated form of the T cell x. They only differ in the antigen responsible for the infection. The analysis using organization theory allows to distinguish between the two infection scenarios.

2.2.5 Drug Effect

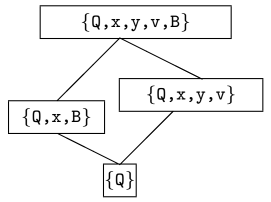
Perelson et al. [1996] developed a viral dynamics model to analyze the effects of two antiretroviral drug treatments. The reverse transcriptase inhibitor, blocking the infection of HIV, is represented in the model as coefficient $1 - \kappa$ ($0 \leq \kappa \leq 1$). High efficacy of the drug corresponds to $\kappa \approx 1$. We should note that the perfect inhibitor is represented by $\kappa = 1$, and the set $\{\mathsf{x}\}$ is the only organization although it is impractical to assume perfect inhibitions. The second antiviral drug, the protease inhibitor, impairs the protein synthesis process in the cell with efficacy η so that infected T cell y produces non-infectious virus v_{NI}: $\mathsf{y} \rightarrow \mathsf{y} + \mathsf{v}_{NI}$. When the inhibition failed with probability $1 - \eta$, the HIV reproduction reaction becomes as follows: $\mathsf{y} \rightarrow \mathsf{y} + \mathsf{v}_I$ where v_I represents normal infectious free virus.

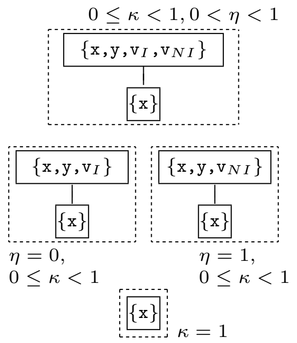
Considering also the extreme values of drug efficacy η, there are three different networks with respect to the proliferation of HIV giving rise to three different organizational structures. In case the drug is not applied to the patient or does not have any effect ($\eta = 0$), only the infectious virus is generated. This is identical case with the HIV infection model discussed before. The smallest organization is the set containing only uninfected T cell x, and above it, free virus particle v and infected T cell y are joined to form the organization. By setting $\eta = 1$, perfect inhibition of the infectious virus proliferation is modeled and only the non-infectious virus is produced from the infected T cell. The organization corresponding to the virus infected state is, in this case, composed of non-infectious virus v_{NI} instead of the infectious type. Statistically speaking, however, reproduction processes of both infectious and non-infectious virus are present in the dynamical reaction system, and the efficacy parameter is set to a value within $0 < \eta < 1$ to model the practical situation. If both of the reactions are included in the network simultaneously, the set containing both infectious and non-infections virus is found to be an organization.

Table 2.2: Three levels of model abstraction. The level of abstraction increases from left to right, and additional information is required to lower the abstraction level. At the highest abstraction level, we take *organizations* (sets of molecular species that are closed and self-maintaining) to understand and describe the dynamical behavior. See text for details.

ODE Model	Reaction Network Model	Organizational Structure
A: (M. A. Nowak, C. R. M. Bangham: *Science 272*, 5258 (1996), 74–79.) $\begin{array}{rcl} \dot{x} & = & \lambda - dx - \beta xv \\ \dot{y} & = & \beta xv - ay \\ \dot{v} & = & ky - uv \end{array}$	$\begin{array}{rclcrcl} \emptyset & \to & \mathtt{x}, & & \mathtt{x} & \to & \emptyset \\ \mathtt{y} & \to & \emptyset, & & \mathtt{v} & \to & \emptyset \\ \mathtt{x + v} & \to & \mathtt{y + v} & & & & \\ \mathtt{y} & \to & \mathtt{y + v} & & & & \end{array}$	$\begin{tikzcd} \boxed{\{x,y,v\}} \arrow[d, dash] \\ \boxed{\{x\}} \end{tikzcd}$
B: (M. A. Nowak, C. R. M. Bangham: *Science 272*, 5258 (1996), 74–79.) $\begin{array}{rcl} \dot{x} & = & \lambda - dx - \beta xv \\ \dot{y} & = & \beta xv - ay - pyz \\ \dot{v} & = & ky - uv \\ \dot{z} & = & cyz - bz \end{array}$	$\begin{array}{rclcrcl} \emptyset & \to & \mathtt{x}, & & \mathtt{x} & \to & \emptyset \\ \mathtt{y} & \to & \emptyset, & & \mathtt{v} & \to & \emptyset \\ \mathtt{z} & \to & \emptyset, & & \mathtt{x + v} & \to & \mathtt{y + v} \\ \mathtt{y + z} & \to & \mathtt{z} & & & & \\ \mathtt{y + z} & \to & \mathtt{y + 2\ z} & & & & \\ \mathtt{y} & \to & \mathtt{y + v} & & & & \end{array}$	$\begin{tikzcd} \boxed{\{x,y,v,z\}} \arrow[d, dash] \\ \boxed{\{x,y,v\}} \arrow[d, dash] \\ \boxed{\{x\}} \end{tikzcd}$
C: (D. Wodarz, M.A. Nowak: *PNAS 96*, 25 (1999), 14464–14469.) $\begin{array}{rcl} \dot{x} & = & \lambda - dx - \beta xy \\ \dot{y} & = & \beta xy - ay - pyz \\ \dot{w} & = & cxyw - cqyw - bw \\ \dot{z} & = & cqyw - hz \end{array}$	$\begin{array}{rclcrcl} \emptyset & \to & \mathtt{x}, & & \mathtt{x} & \to & \emptyset \\ \mathtt{y} & \to & \emptyset, & & \mathtt{w} & \to & \emptyset \\ \mathtt{z} & \to & \emptyset, & & \mathtt{x + y} & \to & \mathtt{2\ y} \\ \mathtt{y + z} & \to & \mathtt{z} & & & & \\ \mathtt{y + w} & \to & \mathtt{y + z} & & & & \\ \mathtt{x + y + w} & \to & \mathtt{x + y + 2\ w} & & & & \end{array}$	$\begin{tikzcd} \boxed{\{x,y,w,z\}} \arrow[d, dash] \\ \boxed{\{x,y\}} \arrow[d, dash] \\ \boxed{\{x\}} \end{tikzcd}$

Table 2.2: (continued)

ODE Model	Reaction Network Model	Organizational Structure
D: (D. S. Callaway, A. S. Perelson: *Bull. Math. Biol. 64* (2002), 29–64.) $\dot{Q} = \xi - fQ - \theta(v+B)Q$ $\dot{x} = s\theta(v+B)Q - dx - (1-\kappa)\beta xv$ $\dot{y} = (1-\kappa)\beta xv - ay$ $\dot{v} = N_T\delta y - uv$	$\emptyset \rightarrow \mathtt{Q}$, $\mathtt{Q} \rightarrow \emptyset$ $\mathtt{x} \rightarrow \emptyset$, $\mathtt{y} \rightarrow \emptyset$ $\mathtt{v} \rightarrow \emptyset$, $\mathtt{x} + \mathtt{v} \rightarrow \mathtt{y} + \mathtt{v}$ $\mathtt{y} \rightarrow \mathtt{y} + N_T\mathtt{v}$ $\mathtt{Q} + \mathtt{v} \rightarrow s\mathtt{x} + \mathtt{v}$ $\mathtt{Q} + \mathtt{B} \rightarrow s\mathtt{x} + \mathtt{B}$	

E: (A.S. Perelson, *et* al.: *Science 271*, 5255 (1996), 1582–1586.)
$\dot{x} = \lambda - dx - (1-\kappa)kv_I x$
$\dot{y} = (1-\kappa)kv_I x - \delta y$
$\dot{v}_I = (1-\eta)N_T\delta x - cv_I$
$\dot{v}_{NI} = \eta N_T\delta y - cv_{NI}$
	$P = 1 - \eta$
	$\emptyset \rightarrow \mathtt{x}$
	$\mathtt{x} \rightarrow \emptyset$
	$\mathtt{v}_I \rightarrow \emptyset$
	$\mathtt{v}_{NI} \rightarrow \emptyset$
	$\mathtt{x} + \mathtt{v}_I \rightarrow \mathtt{y} + \mathtt{v}_I$
	$\mathtt{y} \rightarrow \mathtt{y} + \mathtt{v}_I$
	$P = \eta$
	$\emptyset \rightarrow \mathtt{x}$
	$\mathtt{x} \rightarrow \emptyset$
	$\mathtt{v}_I \rightarrow \emptyset$
	$\mathtt{v}_{NI} \rightarrow \emptyset$
	$\mathtt{x} + \mathtt{v}_I \rightarrow \mathtt{y} + \mathtt{v}_I$
	$\mathtt{y} \rightarrow \mathtt{y} + \mathtt{v}_{NI}$
		 $0 \le \kappa < 1, 0 < \eta < 1$
		{x,y,v$_I$,v$_{NI}$}
		{x}
		{x,y,v$_I$}
		{x}
		{x,y,v$_{NI}$}
		{x}
		$\eta = 0$, $0 \le \kappa < 1$
		$\eta = 1$, $0 \le \kappa < 1$
		{x}
		$\kappa = 1$

2.2.6 Summary and Discussion

In this study we have shown that different models of immune response to HIV infection possess different lattices of organizations. As we can see in Table 2.2, a lattice provides a quick overview of the model's structure and its potential dynamics. We can see which kind of species together can constitute a steady state, namely exactly those forming an organization.

The difference in organizational structure (naturally) reflects the way the model has been extended. For example, changing the basic model by adding the immune response (Table 2.2, Row 1 and 2) results in a new organization to appear, which represents the infection antagonized by the immune response z. Extending the model does not necessarily change the lattice structure, as shown by the CTL memory model (Table 2.2, Row 3). The intention of the modelers is to emphasize effects of CTL memory precursor w for long-term viral load control mediated by CTL. The ODE model is designed for a steady state to contain both the CTL precursor and CTL effector z. This design principle is visible in the organizational structure as the largest organization to contain both the precursor and the effector.

The fourth model is an example for extending Model A such that the organizations are not arranged in a chain in the Hasse-diagram. The main concern of the model developers is to include quiescent cells Q, but the reason of the lattice structure not being in a chain is the general antigen B in addition to HIV virus particle v. From the organizational structure, both antigens B and v appear with activated T cell x. Only v of the two antigens is associated with infected T cell y, as intended by the model design.

Through the last model (Table 2.2, Row 5), we demonstrate how parameters could be handled in the static reaction network analysis. The quantity of some parameters determines the reaction network structure leading to different results of the static analysis. The efficacy of protease inhibitors represented by η is our example. Infected T cell y probabilistically produces infectious virus v_I or non-infectious virus v_{NI} as shown in the reaction network model of ODE Model E. Seeing the reaction as a stochastic process, the network structure alternates between them. When analyzing such a network with chemical organization theory, three cases are considered depending on the value of η. Two of them are described as the success and the failure of the protease inhibitions represented by $\eta = 0$ and $\eta = 1$, respectively. The other case $0 < \eta < 1$ takes a probabilistic view such that both reactions (infections and non-infections virus proliferation) occur in the whole system. We obtained lattices that differ only in their species composition. The other important parameter in this model κ, the efficacy of reverse transcriptase inhibitors, affects the results of our analysis in a trivial way.

From this perspective, the theory of chemical organization appears as a useful tool, which creates a first, rough map of the structure and potential dynamical behavior of a reaction system. The scaffold obtained as the set of organizations can guide further more detailed analysis, which may study the dynamics within or in-between organizations using classical tools from dynamical systems theory. The results of more detailed studies can in turn be explained and summarized with respect to the lattice of organizations.

Source :

Matsumaru, N., Centler, F., and Dittrich, P. (2005a). Chemical organization theory as a theoretical base for chemical computing. In Teuscher, C. and Adamatzky, A., editors, *Proceedings of the 2005 Workshop on Unconventional Computing: From Cellular Automata to Wetware*, pages 75–88. Luniver Press, Beckington, UK.

Matsumaru, N., Centler, F., Speroni di Fenizio, P., and Dittrich, P. (2007a). Chemical organization theory as a theoretical base for chemical computing. *International Journal of Unconventional Computing*, 3(4):285–309.

Chapter 3

Chemical Boolean Devices

Contents

In this chapter, we apply chemical organization theory as a design principle for chemical computing systems. Various boolean functions are exercised, and the formal procedure to implement the boolean functions with chemical reaction systems was devised. In short, the procedure starts with assigning two molecular species to each of the boolean variable. Those two pair-wise species are defined to vanish upon collision due to the contradictory states they are representing. The mapping from the input to the output variables is simply the reactions of transforming input species into proper output species. To initiate the computation, input species are injected, and topological structure of the reaction network is changed by influxes of the species. Outcomes of the computation are interpreted from the output species present in the reaction vessel.

When analyzing the organizational structure in the resulting reaction network including the influxes, there is only one organization of the species representing the desired input-output pair. We safely conclude according to the theorem that the output species desired only is persistent in the reaction vessel, and an ordinary differential equation system is simulated to validate that conclusion in a dynamical situation.

The outline of this chapter is following: A general procedure of converting a logic circuit into a chemical reaction network is described in Section 3.1. Like others (cf., Adamatzky and De Lacy Costello [2002]; Tsuda et al. [2004]; Zauner and Conrad [2001]), a simple non-linear logical operation XOR is implemented first (Section 3.2). Then, in Section 3.3, multiple NAND gates are combined, demonstrating the scalability of the method. Recurrent circuit like a flip-flop logic circuit (Section 3.4) and a controllable oscillator (Section 3.5) are also constructed. Both circuits contain a simple feedback loop, which is an important building block in biological signaling networks [Bhalla and Iyengar, 1999] to achieve robustness [Stelling et al., 2004] or multi-stationarity.

3.1 A Recipe For A Chemical Logic Circuit

In this section we present a procedure for designing chemical reaction networks implementing a logic circuit (see Table 3.1 for a non-formal recipe). A logic circuit is a composition of logic gates. As such it can be fully described by a set of boolean functions and boolean variables, forming a boolean network [Kauffman, 1969]. Let the boolean network be defined by a set of M boolean functions and a set of N ($\geq M$) boolean variables:

$$\{\mathsf{b}_1, \ldots, \mathsf{b}_M, \ldots, \mathsf{b}_N\} \tag{3.1}$$

where $\{\mathsf{b}_j | 1 \leq j \leq M\}$ are determined by the boolean functions (*internal variables*) and the remaining variables $\{\mathsf{b}_j | M < j \leq N\}$ are the input variables of the boolean network. The set of boolean functions is

$$\{\mathsf{b}_i = F_i(\mathsf{b}_{q(i,1)}, \ldots, \mathsf{b}_{q(i,n_i)}) \mid i = 1, \ldots, M\} \tag{3.2}$$

where $\mathsf{b}_{q(i,k)}$ indicates the boolean variable listed as the k-th argument of the i-th function. Since the i-th boolean function F_i takes n_i boolean variables as arguments, there are 2^{n_i} possible inputs. Thus the truth table T_i for function F_i has 2^{n_i} rows and $n_i + 1$ columns:

$$T_i : \begin{bmatrix} t^i_{1,1} & \cdots & t^i_{1,n_i} & t^i_{1,n_i+1} \\ \vdots & \ddots & \vdots & \vdots \\ t^i_{2^{n_i},1} & \cdots, & t^i_{2^{n_i},n_i} & t^i_{2^{n_i},n_i+1} \end{bmatrix} \tag{3.3}$$

where $t^i_{h,k} \in \{0, 1\}$ is the boolean value of the k-th argument in the h-th input case for the i-th boolean function. The $(n_i + 1)$-th column contains the output of F_i.

Given the boolean network, a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ is designed as follows. For each boolean variable b_j we assign two molecular species s_{2j-1} and s_{2j} representing the value 0 and the value 1 in it, respectively. Thus the set of molecular species $\mathcal{M}$ contains $2N$ molecular species as follows:

$$\mathcal{M} = \{s_{2j-1}, s_{2j} \mid j = 1, \ldots, N\} \tag{3.4}$$

The set of reaction rules can be decomposed into two sets of reactions:

$$\mathcal{R} = \mathcal{L} \cup \mathcal{D}. \tag{3.5}$$

Set of reactions $\mathcal{L}$ is derived from the logical operations of the boolean functions with $\mathcal{L} = \bigcup_{i=1}^{M} \mathcal{L}^i$ where $\mathcal{L}^i$ is a set of *logical reactions* associated with the truth table T_i of boolean function F_i. For each input case h (each row of the truth table), one reaction rule is created:

$$\mathcal{L}^i = \{A_{i,h} \to B_{i,h} \mid h = 1, \ldots, 2^{n_i}\}. \tag{3.6}$$

The left-hand side is a set of *reactants* $A_{i,h} = \{a_{i,1,h} + \cdots + a_{i,k,h} + \cdots + a_{i,n_i,h}\}$ where $a_{i,k,h}$ is a molecular species representing the boolean variable that is taken as the k-th argument of function F_i and thus $\mathsf{b}_{q(i,k)}$. Since two molecular species $s_{2q(i,k)-1}$ and $s_{2q(i,k)}$ are assigned to boolean variable $\mathsf{b}_{q(i,k)}$ depending on its content, the truth table T_i is used to select from the two. If the entry $t^i_{h,k}$ of the truth table is equal to 0, $\mathsf{b}_{q(i,k)}$ must be set to 0 in the h-th input case, and thus $s_{2q(i,k)-1}$ is chosen as the reactant. Otherwise, $a_{i,k,h}$ is $s_{2q(i,k)}$:

$$a_{i,k,h} = \begin{cases} s_{2q(i,k)-1} & if\ t^i_{h,k} = 0 \ , \\ s_{2q(i,k)} & if\ t^i_{h,k} = 1 \ . \end{cases} \tag{3.7}$$

Similarly, the right-hand side is a set of *products* $B_{i,h} = \{b_{i,h}\}$, and

$$b_{i,h} = \begin{cases} s_{2i-1} & if\ t^i_{h,n_i+1} = 0 \ , \\ s_{2i} & if\ t^i_{h,n_i+1} = 1 \ , \end{cases} \tag{3.8}$$

since the $(n_i + 1)$-th column of truth table T_i contains the output.

The other component of the set $\mathcal{R}$ is the set of *destructive reactions* $\mathcal{D}$. Since binary states of a boolean variable b_j are coded with two molecular species s_{2j-1} and s_{2j}, the state becomes undefined when both or neither of the species are present. In order to avoid such a case, the two opposite molecular species are defined to vanish upon collision:

$$\mathcal{D} = \{s_{2j-1} + s_{2j} \to \emptyset | j = 1, \ldots, N\}. \tag{3.9}$$

The resulting reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ implements the logic circuit without any input specified. The input variables of the boolean network $\{\mathsf{b}_j | M < j \leq N\}$ must be initialized externally because they are not set by the boolean functions. The initialization of the input variables is encoded by an inflow reaction, which is a zero-order reaction producing substances from the empty set. If an input variable b_j is initialized to 0, for example, the reaction network is changed to $\langle \mathcal{M}, (\mathcal{R} \cup \{\emptyset \to s_{2j-1}\}) \rangle$. It is possible for more than one variable to be initialized in this manner as it is possible for more than one molecular species to be injected by the influx.

Implementing logic circuits with periodic attractors While converting boolean networks into chemical reaction networks, feedback loops need special treatment. When considering boolean networks in general, the network can form feedback loops by connecting an output to an input so that the input is dependent on the output. This configuration can give rise to attractors having a period greater than one so that the system starts to oscillate between two (or more) states. An example of such a periodic attractor is an oscillator. When an oscillator is implemented with a reaction network, the complementary molecular species are alternating and thus decay instantaneously. To delay the

Table 3.1: Recipe for mapping a boolean circuit to a chemical reaction network.

Input: Boolean network given by two sets: a set of M boolean functions $\{F_1, \ldots, F_M\}$ and a set of N boolean variables $\{\mathsf{b}_1, \ldots, \mathsf{b}_M, \ldots, \mathsf{b}_N\}$. Variables $\{\mathsf{b}_1, \ldots, \mathsf{b}_M\}$ are determined by the boolean functions (*internal variables*); the remaining variables $\{\mathsf{b}_{M+1}, \ldots, \mathsf{b}_N\}$ are input variables of the boolean network.
Output: Reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ (a set of molecular species $\mathcal{M}$ and a set of reaction rules $\mathcal{R}$) representing the boolean network without any input variable specified.[a]
Algorithm:

```
1. For each boolean variable $\mathsf{b}_j$:
  (a) Add two molecular species, $b_j$ and $B_j$, to $\mathcal{M}$;[b]
  (b) Add one destructive reaction of the form $b_j + B_j \to \emptyset$ to $\mathcal{R}$;
2. For each boolean function $F_i$:
  (a) Create the truth table of $F_i$ with $2^{n_i}$ input cases
      (where $n_i$ is the arity of $F_i$);
  (b) For each input case, create a logical reaction.[c]
      i Left-hand side (reactants) corresponds to the input of $F_i$.
      ii Right-hand side (products) consists of one molecular species representing the respective boolean output of $F_i$.
```

[a]Specifying an input variable of the boolean network is coded by an inflow reaction.
[b]As a naming convention of molecular species in this paper, the lowercase species represents value 0 in the boolean variable, and the uppercase stands for 1.
[c]For example, the XOR-function is converted into reactions as follows:

b_2	b_3	$\mathsf{b}_1 = F_1(\mathsf{b}_2, \mathsf{b}_3)$		Reactants	$\to$	Products
0	0	0		$b_2 + b_3$	$\to$	b_1
0	1	1	$\Rightarrow$	$b_2 + B_3$	$\to$	B_1
1	0	1		$B_2 + b_3$	$\to$	B_1
1	1	0		$B_2 + B_3$	$\to$	b_1

complete destruction of the two species, an amplification process has to be introduced for variables that change in the periodic attractors. A detailed description of the implementation can be found in Section 3.5.

3.2 Case Study I: A Chemical XOR

To demonstrate how chemical organization theory can be used for chemical computing, an (artificial) chemical reaction network is designed to implement an XOR logic gate.

3.2.1 Reaction Network and Organizational Analysis

The XOR logic gate is defined as a set of three boolean variables $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ and a set of one boolean function $\{F_c\}$ where the function is: $\mathsf{c} = F_{\mathsf{c}}(\mathsf{a}, \mathsf{b})$, and the truth table is:

$$T_{\mathsf{c}} : \begin{array}{ccc} \mathsf{a} & \mathsf{b} & \mathsf{c} \\ \hline 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{array} \tag{3.10}$$

Since boolean variable c is the internal variable which is determined by a boolean function, the set of boolean variable should be listed as $\{\mathsf{c}, \mathsf{a}, \mathsf{b}\}$ according to the recipe in Section 3.1. Boolean variables and molecular species are ordered alphabetically for readability, however. Furthermore, the variable name is adopted as an index of functions.

Given the definition of the XOR boolean network, an algebraic chemistry $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \rangle$ is generated to implement the logic gate. Since there are $N = 3$ boolean variables, the set of molecular species consists of six molecular species:

$$\mathcal{M}_{\text{XOR}} = \{a, A, b, B, c, C\} \tag{3.11}$$

where the lower- and uppercase version of the variable name are assigned to the boolean variable of that name. For example, molecular species a represents boolean variable $\mathsf{a} = 0$, and A stands for $\mathsf{a} = 1$.

The set of reaction rules $\mathcal{R}_{\text{XOR}}$ is decomposed into two parts:

$$\mathcal{R}_{\text{XOR}} = \mathcal{L}_{\text{XOR}} \cup \mathcal{D}_{\text{XOR}} \tag{3.12}$$

where $\mathcal{L}_{\text{XOR}}$ is a set of reactions for the logical operation and $\mathcal{D}_{\text{XOR}}$ is a set of destructive reactions. Since there is only one function in the boolean network, $\mathcal{L}_{\text{XOR}} = \mathcal{L}^{\mathsf{c}}_{\text{XOR}}$ where $\mathcal{L}^{\mathsf{c}}_{\text{XOR}}$ is a set of logical reactions constructed from the boolean function F_c. From the truth table T_{c}, four logical reactions are derived:

$$\mathcal{L}_{\text{XOR}} = \mathcal{L}^{\mathsf{c}}_{\text{XOR}} = \{a + b \rightarrow c,\ a + B \rightarrow C,\ A + b \rightarrow C,\ A + B \rightarrow c\}. \tag{3.13}$$

The Hasse diagram in Figure 3.1 (A) shows the hierarchy of organizations of the reaction network that includes only the logical reactions $\mathcal{L}_{\text{XOR}}$. Twenty-eight sets of molecular species are found to be organizations. The remaining 36 sets do not satisfy either the closure or the self-maintenance property.

The set $\{a, b\}$, for example, is not an organization because it is not closed. The reaction $a + b \rightarrow c$ is applicable and produces a new molecular species c that is not a member of the set $\{a, b\}$. The set $\{a, b, c\}$ is closed but not an organization because it is not self-maintaining. A production rate vector $\mathbf{f}$ is calculated as follows:

$$\mathbf{f} = \begin{pmatrix} f_a \\ f_A \\ f_b \\ f_B \\ f_c \\ f_C \end{pmatrix} = \mathbf{M}\mathbf{v} = \begin{pmatrix} -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & -1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -v_1 \\ 0 \\ -v_1 \\ 0 \\ v_1 \\ 0 \end{pmatrix} \tag{3.14}$$

where a stoichiometric matrix $\mathbf{M}$ is multiplied by the flux vector $\mathbf{v}$ with $v_1 > 0$ satisfying the condition 1 and condition 2 from the definition of self-maintenance. The third condition cannot be satisfied because the production rates f_a for

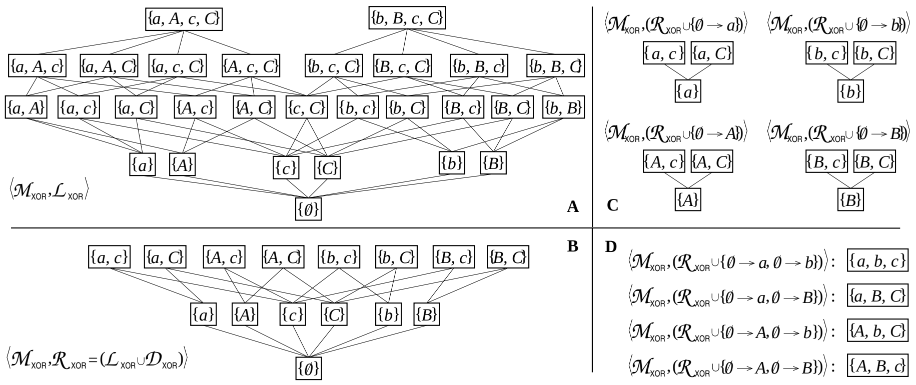

{a, A, c, C}
{b, B, c, C}
{a, A, c}
{a, A, C}
{a, c, C}
{A, c, C}
{b, c, C}
{B, c, C}
{b, B, c}
{b, B, C}
{a, A}
{a, c}
{a, C}
{A, c}
{A, C}
{c, C}
{b, c}
{b, C}
{B, c}
{B, C}
{b, B}
{a}
{A}
{c}
{C}
{b}
{B}
{∅}
⟨M_XOR, L_XOR⟩
A
⟨M_XOR, (R_XOR ∪ {∅ → a})⟩
⟨M_XOR, (R_XOR ∪ {∅ → b})⟩
⟨M_XOR, (R_XOR ∪ {∅ → A})⟩
⟨M_XOR, (R_XOR ∪ {∅ → B})⟩
C
B
⟨M_XOR, R_XOR = (L_XOR ∪ D_XOR)⟩
D
⟨M_XOR, (R_XOR ∪ {∅ → a, ∅ → b})⟩: {a, b, c}
⟨M_XOR, (R_XOR ∪ {∅ → a, ∅ → B})⟩: {a, B, C}
⟨M_XOR, (R_XOR ∪ {∅ → A, ∅ → b})⟩: {A, b, C}
⟨M_XOR, (R_XOR ∪ {∅ → A, ∅ → B})⟩: {A, B, c}

Figure 3.1: Hierarchy of organizations for the chemical reaction network implementing an XOR logic gate. (A) The network consists only of the logical reactions $\mathcal{L}_{\text{XOR}}$. (B) Destructive reactions $\mathcal{D}_{\text{XOR}}$ are added to exclude contradictions. The resulting reaction network $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \rangle$ implements the XOR logic gate without any input specified. (C) One input is defined by adding one influx reaction. (D) Adding the second input. The hierarchy of organizations collapses from (A) to (D), with the desired output as the only organization left in (D).

molecular species a and f_b for molecular species b cannot be greater or equal than 0 at the same time.

In this particular case of the reaction network, all organizations consist of combinations of molecular species that do not react with each other. A set of molecular species where no reaction can take place is obviously closed and self-maintaining. Provided that a set contains molecular species with no reactions among them, Condition 1 of Definition 2 is automatically fulfilled. According to Condition 2 of Definition 2, a zero flux vector $\mathbf{v} = \mathbf{0}$ is multiplied by the stoichiometric matrix M. The result is a zero production rate vector $\mathbf{f} = \mathbf{0}$. The zero vector fulfills Condition 3 of Definition 2, and thus all conditions for self-maintenance are satisfied.

With the species set of an organization being closed and self-maintaining, it is more likely to observe the presence of molecular species of an organization than of any other species combination in the reaction vessel. If the dynamics of the reaction network is modelled using ordinary differential equations, there exists a related organization for every fixed point of the system Dittrich and Speroni di Fenizio [2007].

The second component of the set $\mathcal{R}_{\text{XOR}}$ is a set of destructive reactions:

$$\mathcal{D}_{\text{XOR}} = \{a + A \to \emptyset, b + B \to \emptyset, c + C \to \emptyset\}. \tag{3.15}$$

Combining $\mathcal{D}_{\text{XOR}}$ and $\mathcal{L}_{\text{XOR}}$ the reaction network $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \rangle$ implements the XOR logic gate without any input specified. Its Hasse diagram of organizations is shown in Figure 3.1 (B). The number of organizations is reduced from 28 to 15.

Now we set the input variables of the boolean network a and b to initiate the computational process. For the initialization, an inflow reaction is added to

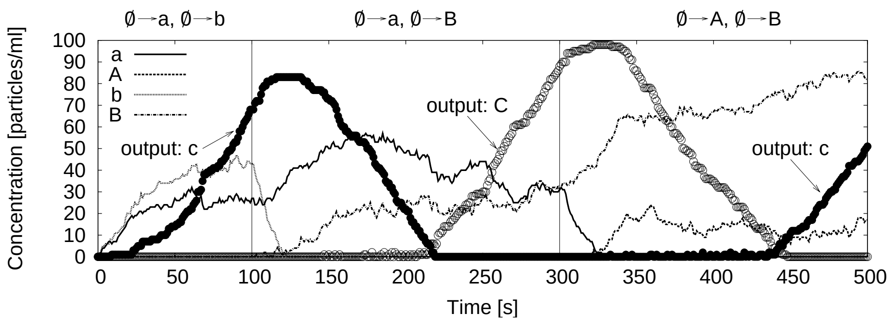


Figure 3.2: Dynamic behavior of the chemical reaction network implementing an XOR logic gate. The time course of all 6 molecular species is shown. Irreversible mass action kinetics are assumed for all reactions. Reaction rates are set to $k = 0.001$ for logical reactions. Reaction rates of destruction reactions are set to $k = 0.1$. For all irreversible constant influxes (*e.g.*, $\emptyset \rightarrow A$), the rates are set to $k = 1$. The reaction system is stochastically simulated with the biochemical network simulator *Copasi* using a compartment size of 1 ml. See text for details.

the reaction network. We start with providing one input only, leaving the other input variable undefined. Figure 3.1 (C) shows the results for the four resulting algebraic chemistries $\langle \mathcal{M}_{\text{XOR}}, (\mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow a\})\rangle$, $\langle \mathcal{M}_{\text{XOR}}, (\mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow A\})\rangle$, $\langle \mathcal{M}_{\text{XOR}}, (\mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow b\})\rangle$, and $\langle \mathcal{M}_{\text{XOR}}, (\mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow B\})\rangle$, respectively. We can see that providing one input signal has further reduced the behavioral freedom of the reaction system. Only three combinations of molecular species are left, which may be encountered in the reaction vessel as a stationary state. Furthermore we can see that – in this special case – the output is not determined from a stoichiometric point of view since, in all four Hasse diagrams, sets containing c and C are found to be closed and self-maintaining.

When we finally provide both inputs, the Hasse diagram of organizations collapses so that only one organization remains for every input condition (Figure 3.1 (D)). This implies that, no matter how we chose the dynamics, no other molecular species than those of the organization can be sustained in the reaction vessel regardless of the initial state. We can see that the remaining organization contains the desired output molecular species c or C, respectively. The analyzed algebraic chemistries are $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow a, \emptyset \rightarrow b\}\rangle$, $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow a, \emptyset \rightarrow B\}\rangle$, $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow A, \emptyset \rightarrow b\}\rangle$, and $\langle \mathcal{M}_{\text{XOR}}, \mathcal{R}_{\text{XOR}} \cup \{\emptyset \rightarrow A, \emptyset \rightarrow B\}\rangle$.

3.2.2 Dynamical Simulation

To validate the results from applying organization theory to the XOR reaction network, stochastic simulations are performed using the simulator packages *MGS* [Giavitto and Michel, 2001] and *Copasi* [Sahle et al., 2006].

Figure 3.2 shows a typical simulation run. The influx is defined as an irreversible constant flux with kinetic parameter set to 1. For all other reactions, we chose irreversible mass action kinetics. The parameters for the destructive reactions $\mathcal{D}_{\text{XOR}}$ are set to $k = 0.1$, and those for the logical reactions $\mathcal{L}_{\text{XOR}}$ are

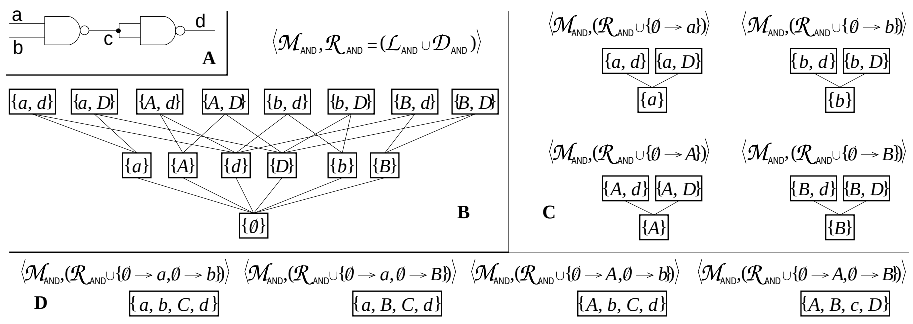

a
b
c
d
A
$\langle \mathcal{M}_{AND}, \mathcal{R}_{AND} = (\mathcal{L}_{AND} \cup \mathcal{D}_{AND}) \rangle$
{a, d} {a, D} {A, d} {A, D} {b, d} {b, D} {B, d} {B, D}
{a} {A} {d} {D} {b} {B}
{∅}
B
C
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to a\}) \rangle$
{a, d} {a, D} {a}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to b\}) \rangle$
{b, d} {b, D} {b}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to A\}) \rangle$
{A, d} {A, D} {A}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to B\}) \rangle$
{B, d} {B, D} {B}
D
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to a, \emptyset \to b\}) \rangle$
{a, b, C, d}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to a, \emptyset \to B\}) \rangle$
{a, B, C, d}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to A, \emptyset \to b\}) \rangle$
{A, b, C, d}
$\langle \mathcal{M}_{AND}, (\mathcal{R}_{AND} \cup \{\emptyset \to A, \emptyset \to B\}) \rangle$
{A, B, c, D}

Figure 3.3: Results of the theoretical analysis of the chemical reaction network implementing the logic circuit consisting of multiple gates. (A) Circuit diagram of an AND gate with two NAND gates. (B) The network consists of six logical reactions $\mathcal{L}_{\text{AND}}$ and four destructive reactions $\mathcal{D}_{\text{AND}}$. (C) An influx is added to define one input. Note that a single input like $\emptyset \to B$ does not determine the output. (D) Another inflow is added so that both inputs are defined. Despite the combination of two chemical logic gates, only the organization containing the desired output species is left in (D).

set to $k = 0.001$. At several simulation times, the input is changed in order to observe the switching of the XOR gate. Initially, there exist no molecular particles in the reactor, and two influxes of a and b are present. This corresponds to the case in which both the input variables a and b are set to 0. Since molecular species c is generated, the output is computed to c = 0.

At simulation time 100 s, the content of input variable b is switched to 1 by replacing the influx of molecular species b with the influx $\emptyset \to B$. The molecular particles b and c, whose concentrations are still high from the previous computation, deteriorate and finally vanish. The desired output C does not appear until the time point of approximately 200 s. Then, instead of a, the molecular species A is applied as an input starting from simulation time 300 s. The remaining molecules of species a and C from the previous computation decay first and the desired answer c appears in the end.

As seen from the dynamical simulation, the computational result represented by the qualitative final state of the reaction vessel is independent of the initial state. The applied continuous input is the only factor deciding on the final state. The output molecules are generated continuously while undesired species are removed from the reaction vessel by collisions with their anti-particles. When applying two inputs, the analysis of the reaction network revealed that only one organization exists, predicting only one species composition (the species of that organization) to be closed and self-maintaining, and thus likely to be observed in the reactor. The stochastic simulation confirms the result.

3.3 Case Study II: Multiple Logic Gates

Extensibility and scalability is an advantage of conventional logic gates. Multiple logic gates can be easily connected to realize different forms of computation. In this section, we demonstrate the connectivity of chemical logic gates and scalability of the theoretical analysis. As an example, we implement an AND and an OR gate by combining NAND gates.

3.3.1 AND Gate by Connecting Two NAND Gates

An AND gate can be constructed by sequentially connecting two NAND gates (Figure 3.3 (A)). The single logic NAND gates are chemically implemented in the same way as the XOR gate in the previous example.

The boolean network is defined by a set of four boolean variables $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$ and a set of two boolean functions $\{\mathsf{c} = F_{\mathsf{c}}(\mathsf{a}, \mathsf{b}), \mathsf{d} = F_{\mathsf{d}}(\mathsf{c})\}$. The first NAND is associated with F_{C} and the second is with F_{D}. The truth table T_{C} for the first NAND gate has four rows. On the other hand, the truth table T_{D} of the second NAND gate has only two rows, since the function F_{D} requires only one argument. The reaction network $\langle \mathcal{M}_{\text{AND}}, \mathcal{R}_{\text{AND}} \rangle$ is constructed as follows:

$$\mathcal{M}_{\text{AND}} = \{a, A, b, B, c, C, d, D\} \tag{3.16}$$

and

$$\mathcal{R}_{\text{AND}} = \mathcal{L}_{\text{AND}} \cup \mathcal{D}_{\text{AND}} = (\mathcal{L}^{\mathsf{c}}_{\text{AND}} \cup \mathcal{L}^{\mathsf{d}}_{\text{AND}}) \cup \mathcal{D}_{\text{AND}} \tag{3.17}$$

where

$$\begin{aligned} \mathcal{D}_{\text{AND}} &= \{a + A \to \emptyset,\ b + B \to \emptyset,\ c + C \to \emptyset,\ d + D \to \emptyset\}, \\ \mathcal{L}^{\mathsf{c}}_{\text{AND}} &= \{a + b \to C,\ a + B \to C,\ A + b \to C,\ A + B \to c\}, \\ \mathcal{L}^{\mathsf{d}}_{\text{AND}} &= \{2c \to D,\ 2C \to d\}. \end{aligned}$$

The two reaction rules in $\mathcal{L}^{\mathsf{d}}_{\text{AND}}$ are equivalent to a NOT operation.

The reaction network $\langle \mathcal{M}_{\text{AND}}, \mathcal{R}_{\text{AND}} \rangle$ with six reactions and four destructive outflows is analyzed for organizations (closed and self-maintaining sets of molecular species), and the result is shown as the Hasse diagram in Figure 3.3 (B) depicting a hierarchy of organizations in the reaction network. The reaction network implements the AND gate without any input specified. Initialization of input variables a and b is represented by adding inflows to the set of reactions. In Figure 3.3 (C), hierarchies of organizations in the reaction network are shown when one inflow is provided. Hasse diagrams in Figure 3.3 (D) show the hierarchy of organizations in the reaction network with two input fluxes in which both input variables are defined. The same discussion as in the previous XOR logic gate example can be applied. When both inputs are provided, only one organization remains for every input condition and the organization contains the desired output molecular species d or D, respectively. The theoretical analysis suggests that AND behavior emerges regardless of an initial state and regardless of the dynamics chosen (cf. Section 1.4).

3.3.2 OR Gate by Connecting Three NAND Gates

Another example of connecting chemical logic gates is an OR circuit with three NAND gates (Figure 3.4 (A)). The logic circuit can be defined by five boolean variables $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{e}\}$ and three boolean functions $\{\mathsf{c} = F_{\mathsf{c}}(\mathsf{a}), \mathsf{d} = F_{\mathsf{d}}(\mathsf{b}), \mathsf{e} = F_{\mathsf{e}}(\mathsf{c}, \mathsf{d})\}$. The reaction network $\langle \mathcal{M}_{\text{OR}}, \mathcal{R}_{\text{OR}} \rangle$ implementing the logic circuit

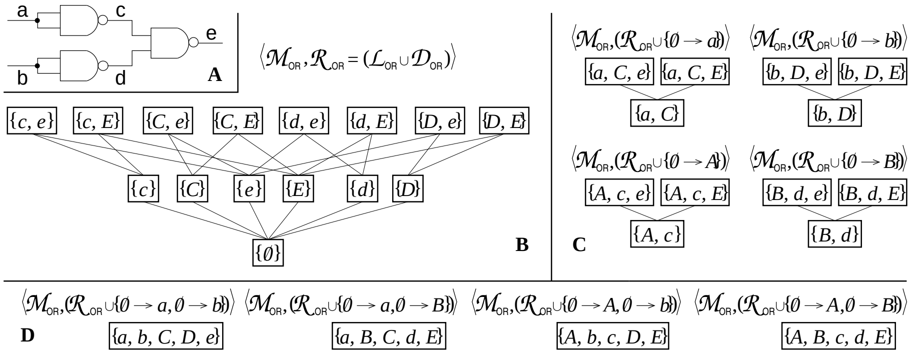

a
b
c
d
e
A
⟨$\mathcal{M}_{OR}$, $\mathcal{R}_{OR}$ = ($\mathcal{L}_{OR}$ ∪ $\mathcal{D}_{OR}$)⟩
{c, e} {c, E} {C, e} {C, E} {d, e} {d, E} {D, e} {D, E}
{c} {C} {e} {E} {d} {D}
{∅}
B
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → a})⟩
{a, C, e} {a, C, E}
{a, C}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → b})⟩
{b, D, e} {b, D, E}
{b, D}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → A})⟩
{A, c, e} {A, c, E}
{A, c}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → B})⟩
{B, d, e} {B, d, E}
{B, d}
C
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → a, ∅ → b})⟩
{a, b, C, D, e}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → a, ∅ → B})⟩
{a, B, C, d, E}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → A, ∅ → b})⟩
{A, b, c, D, E}
⟨$\mathcal{M}_{OR}$,($\mathcal{R}_{OR}$∪{∅ → A, ∅ → B})⟩
{A, B, c, d, E}
D

Figure 3.4: Results of the theoretical analysis of a chemical reaction network implementing the logic circuit consisting of multiple gates. (A) Circuit diagram of an OR gate with three NAND gates. (B) The network consists of eight logical reactions $\mathcal{L}_{\text{OR}}$ and five destructive reactions $\mathcal{D}_{\text{OR}}$. (C) An influx is added. (D) Two inflows are added, specifying two input values. Despite the combination of three chemical logic gates, only the organization including the desired output species is left in (D).

(without any input specified) consists of ten molecular species:

$$\mathcal{M}_{\text{OR}} = \{a, A, b, B, c, C, d, D, e, E\}. \tag{3.18}$$

The set of reaction rules is

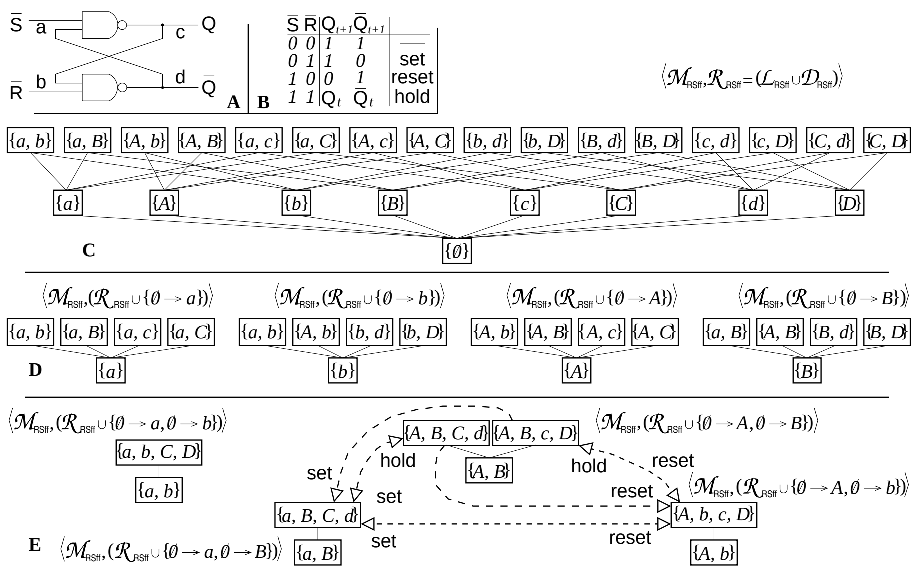
$$\mathcal{R}_{\text{OR}} = \mathcal{L}_{\text{OR}} \cup \mathcal{D}_{\text{OR}} = (\mathcal{L}^c_{\text{OR}} \cup \mathcal{L}^d_{\text{OR}} \cup \mathcal{L}^e_{\text{OR}}) \cup \mathcal{D} \tag{3.19}$$

where

$$\begin{aligned}
\mathcal{L}^{\mathsf{c}}_{\text{OR}} &= \{2a \rightarrow C,\ 2A \rightarrow c\},\\
\mathcal{L}^{\mathsf{d}}_{\text{OR}} &= \{2b \rightarrow D,\ 2B \rightarrow d\},\\
\mathcal{L}^{\mathsf{e}}_{\text{OR}} &= \{c + d \rightarrow E,\ c + D \rightarrow E,\ C + d \rightarrow E,\ C + D \rightarrow e\},\\
\mathcal{D}_{\text{OR}} &= \{a + A \rightarrow \emptyset,\ b + B \rightarrow \emptyset,\ c + C \rightarrow \emptyset,\ d + D \rightarrow \emptyset,\ e + E \rightarrow \emptyset\}.
\end{aligned}$$

The reaction network given is analyzed with chemical organization theory and the result is shown in Figure 3.4 (B). In Figure 3.4 (C) and (D), Hasse diagrams depicting the hierarchy of organizations in the chemical reaction network including influxes are shown. As the other cases, one inflow is not enough to determine the output since output molecular species e and E are both found to be a member of the organizations. Defining a value for both input variables, by adding two influxes to the reaction network, reduces the number of organizations in the network to one, and the only organization consists of the desired combination of molecular species.

It is interesting to note that in our current implementation of a chemical OR gate, the output is not determined by a single input flux like $\emptyset \rightarrow B$ ($\mathsf{b} = 1$), while input a is unspecified (Figure 3.4 (C), right). Theoretically, for $\mathsf{b} = 1$ the output should be 1, independently of a. We can now use chemical organization theory to search for chemical networks that are also able to cope with unspecified inputs (not shown here).



A
B
C
D
E
S̄ R̄ Q_{t+1} Q̄_{t+1}
0 0 1 1
0 1 1 0 set
1 0 0 1 reset
1 1 Q_t Q̄_t hold
⟨M_RSff, R_RSff = (L_RSff ∪ D_RSff)⟩
⟨M_RSff, (R_RSff ∪ {∅ → a})⟩
⟨M_RSff, (R_RSff ∪ {∅ → b})⟩
⟨M_RSff, (R_RSff ∪ {∅ → A})⟩
⟨M_RSff, (R_RSff ∪ {∅ → B})⟩
⟨M_RSff, (R_RSff ∪ {∅ → a, ∅ → b})⟩
⟨M_RSff, (R_RSff ∪ {∅ → A, ∅ → B})⟩
⟨M_RSff, (R_RSff ∪ {∅ → A, ∅ → b})⟩
⟨M_RSff, (R_RSff ∪ {∅ → a, ∅ → B})⟩
set
hold
reset

Figure 3.5: Analysis of a chemical reaction network implementing an RS flip-flop circuit with respect to its emergent behavior at the systems level. (A) Circuit diagram of the RS flip-flop. (B) Truth table describing its behavior. (C) Hierarchy of organizations of the reaction network. (D) An influx is added to define one input. (E) Two inflows are added, specifying two input values. The analysis using chemical organization theory reveals that we can expect a dynamical behavior corresponding to the operation of a flip-flop circuit. See text for details.

3.4 Case Study III: A Chemical Flip-Flop

In this section, we apply our approach to a more complicated example: the flip-flop logic circuit. As opposed to the previous example, a flip-flop circuit is bistable, which is achieved by two feedback connections. When we analyze the organizations of our chemical instantiation of the flip-flop, the bistability of the circuit will also become apparent. This allows us to explain the dynamical behavior of the chemical flip-flop in terms of chemical organization theory on an abstract level, which does not need to refer to concentration levels.

3.4.1 Reaction Network and Organizational Analysis

The RS (Reset and Set) flip-flop circuit consists of two NAND gates connected in parallel as shown in Figure 3.5 (A). The behavior can be described by the truth table as shown in Figure 3.5 (B). The output of one logic gate is connected to one of the two inputs of the other gate, forming a feedback loop. The “set” operation $(\bar{\mathsf{S}}, \bar{\mathsf{R}}) = (0, 1)$ changes the output Q to 1, and the “reset” operation $(\bar{\mathsf{S}}, \bar{\mathsf{R}}) = (1, 0)$ sets Q to 0. When both inputs are set to 1, the output is kept as in the previous state. The one-bit information whether the output Q has been 0 or 1 is stored by the “hold” operation, i.e. $(\bar{\mathsf{S}}, \bar{\mathsf{R}}) = (1, 1)$. Normally, the input $(\bar{\mathsf{S}}, \bar{\mathsf{R}}) = (0, 0)$ is prohibited because the circuit will go into a state where $\mathsf{Q} = 1$ and $\bar{\mathsf{Q}} = 1$. Application examples for the flip-flop are memory and counter circuits.

The flip-flop logic circuit can be defined by the set of four boolean variables $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$ and the set of two boolean functions $\{\mathsf{c} = F_{\mathsf{c}}(\mathsf{a}, \mathsf{d}), \mathsf{d} = F_{\mathsf{d}}(\mathsf{b}, \mathsf{c}\}$. Variables a and b are input variables for the boolean network and the internal variables are c and d. According to the recipe described in Section 3.1, the algebraic chemistry $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \rangle$ is constructed. The set of molecular species consists of eight molecular species

$$\mathcal{M}_{RSff} = \{a, A, b, B, c, C, d, D\}. \tag{3.20}$$

The set of reaction rules is composed of three sets

$$\mathcal{R}_{RSff} = \mathcal{L}_{RSff} \cup \mathcal{D}_{RSff} = (\mathcal{L}^{c}_{RSff} \cup \mathcal{L}^{d}_{RSff}) \cup \mathcal{D}_{RSff} \tag{3.21}$$

where

$$\begin{aligned}
\mathcal{L}^{c}_{RSff} &= \{a + d \to C, a + D \to C, A + d \to C, A + D \to c\},\\
\mathcal{L}^{d}_{RSff} &= \{b + c \to D, b + C \to D, B + c \to D, B + C \to d\},\\
\mathcal{D}_{RSff} &= \{a + A \to \emptyset, b + B \to \emptyset, c + C \to \emptyset, d + D \to \emptyset\}.
\end{aligned}$$

When we apply our analysis to the reaction network $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \rangle$ implementing the RS flip-flop without any input specified, we found 25 organizations consisting of up to two molecular species, which do not react (Figure 3.5 (C)). If values of the two input variables are defined, two influxes are added to the set of reaction rules $\mathcal{R}_{RSff}$ so that four algebraic chemistries are analyzed $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \cup \{\emptyset \to a, \emptyset \to b\} \rangle$, $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \cup \{\emptyset \to a, \emptyset \to B\} \rangle$, $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \cup \{\emptyset \to A, \emptyset \to b\} \rangle$, and $\langle \mathcal{M}_{RSff}, \mathcal{R}_{RSff} \cup \{\emptyset \to A, \emptyset \to B\} \rangle$. As seen in Figure 3.5 (E), the number of organizations found in the network is reduced to two or three for each input case. Since the output species c, C, d, and D are in the set of the reactants, no reaction occurs when those species are not present in the reaction vessel. Thus, the smallest organization contains only the two inflow species. Above it, the designated output species are included in the organizations. This implies that the presence of the output species c, C, d, or D in the reaction vessel is necessary for the flip-flop operation. In other words, the input molecular species alone cannot generate the organization representing an operational mode of the flip-flop.

The operation of the flip-flop can be described by transitions between organizations containing output species: The set and reset operation move the reaction system to the states corresponding to organization $\{a, B, C, d\}$ (set) and $\{A, b, c, D\}$ (reset). Recall that for the set and reset operation we add $\{\emptyset \to a, \emptyset \to B\}$ and $\{\emptyset \to A, \emptyset \to b\}$ to the set of reaction rules, respectively.

For the hold operation (including $\emptyset \to A, \emptyset \to B$), the flip-flop has two stable states represented by the organizations $\{A, B, C, d\}$ and $\{A, B, c, D\}$. If the reaction vessel had been in organization $\{a, B, C, d\}$ previously, it will move into organization $\{A, B, C, d\}$; and if it had been in organization $\{A, b, c, D\}$ before, it will move into organization $\{A, B, c, D\}$. Symbolically speaking, the lowercase input species is replaced by its uppercase due to the input change, but the output state remains unchanged.

For the sake of completeness, the cases in which only one influx is added to the network are shown in Figure 3.5 (D). A set of molecular species that no

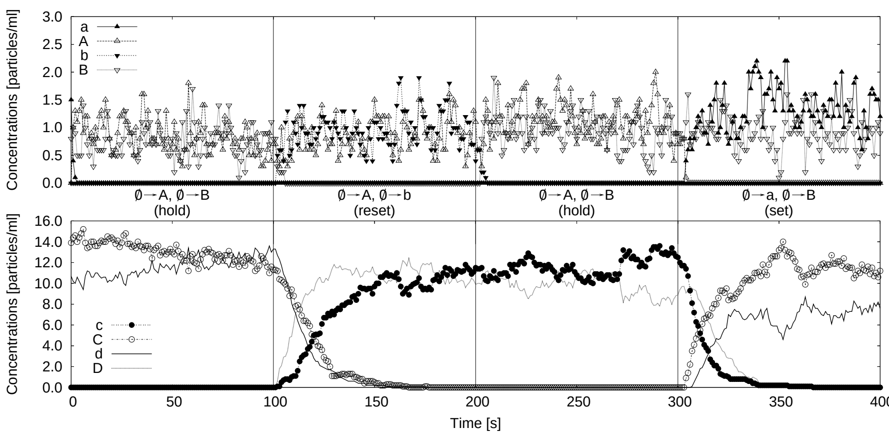


Figure 3.6: Dynamic behavior of the chemical reaction network implementing a RS flip-flop logic circuit. The top figure shows the time course of the input species a, A, b, and B. The bottom figure shows the concentrations of the output species. Irreversible mass action kinetics is assumed for all reactions. The kinetic parameters are set to $k = 1$ for all zero-order reactions (*e.g.*, $\emptyset \rightarrow A$). The kinetic parameter is set to $k = 0.001$ for destructive reactions. For the other second-order reactions producing output species c, C, d, or D, the kinetic parameter is $k = 0.1$. The reaction system is stochastically simulated with the biochemical network simulator *Copasi* of 10 ml.

reaction rule (including decay reaction) is applicable is an organization because no molecular species is produced (closed) or consumed (self-maintaining). The smallest organizations contain only the input species with the influx. Adding one species that does not interact with the input species forms another organization. Since adding another species makes a reaction rule applicable and molecular species are used up with no reproduction, there exists no organization of size greater than two.

3.4.2 Dynamical Simulation

In order to validate the discussion of the previous section we performed stochastic simulations (using *MGS* Giavitto and Michel [2001] and *Copasi* [Sahle et al., 2006]) of reaction systems implementing the chemical flip-flop. Figure 3.6 shows a typical simulation run. The influx is defined as an irreversible constant flux with kinetic parameter set to 1. For all other reactions we chose irreversible mass action kinetics. The kinetic parameters are set to 0.1 for the second-order reactions that produce output species c, C, d, or D. For destructive reactions, the kinetic parameters are set to 0.001. During the first "hold" phase (0 - 100 s), the concentration of C and d remain high. In the following "reset" phase (100 - 200 s), the input reactions $\emptyset \rightarrow A$ and $\emptyset \rightarrow b$ are added to "reset" the system so that the output variable c is set to 0. The concentration of C and d decreases gradually and species c and D accumulate in the reaction vessel. The system eventually reaches a state in which only members of the organization $\{A, b, c, D\}$ are present as expected from the algebraic analysis. In the next phase (200 - 300 s), the input flow of b is replaced by that of B, $\emptyset \rightarrow B$, to "hold" the output of the previous phase. Although the input species have changed, no qualitative change is detected in the bottom graph. Finally, in the last phase (300 - 400 s), the "set" operation is executed by changing the influx $\emptyset \rightarrow A$ to $\emptyset \rightarrow a$. The transition to the state represented by the organization $\{a, B, C, d\}$ is observed.

Although the same input species are injected in the two "hold" phases, the states of the reaction vessel in terms of molecular species present are different depending on the initial conditions. The bistable behavior of the flip-flop circuit is implemented dynamically by the chemical reaction system, which we have expected from our theoretical analysis in the previous section. The reaction network with the two influxes $\emptyset \rightarrow A$ and $\emptyset \rightarrow B$ has two organizations with four species: The system state in the first "hold" phase corresponds to the organization $\{A, B, C, d\}$, and the members of the organization $\{A, B, c, D\}$ are present during the second "hold" phase.

3.5 Case Study IV: An Oscillator

The final case study should elucidate how our method behaves when applied to boolean circuits exhibiting periodic attractors. With a direct feedback, a NAND logic gate can be configured as a controllable oscillator. Analyzing this system shows that the two alternating states are represented by only one organization in the corresponding chemical system. The organization is the union of the sets of molecular species representing each of these states. We will show that, compared to organizations representing fixed points, the organization representing an oscillation contains "contradicting" molecular pairs like a and A.

Figure 3.7 (A) shows a circuit diagram of the oscillator with a NAND gate (decomposed into an AND and a NOT gate) and a truth table describing the oscillatory behavior. A feedback loop is formed by feeding the output from the NOT gate to one of the inputs. The dynamical behavior has two operational modes depending on the value of the input variable a, which is the open input of the circuit. When $\mathsf{a} = 0$, output variable d and the linked input variable b become 1, independently of an initial value of the other variable b.

The stationary state with $\mathsf{b} = \mathsf{d} = 1$ is one operational mode of the circuit, while the other is an oscillation between two states. Setting $\mathsf{a} = 1$ causes the output variable d and linked variable b to alternate between 0 and 1. Provided that b contained 0 at time t, output variable d becomes 1. Since variable d is connected to variable b, the contents of variable b at time $t + 1$ is switched to 1 which is $\bar{\mathsf{b}}$ at time t. When the value of b becomes 1 at time t, variable d will get a value of 0, and so does variable b at time $t + 1$. Repeating the process successively, the value of boolean variables b and d will oscillate between 0 and 1 for each time step.

3.5.1 Chemical Implementation without Amplified Feedback

The oscillator can be defined by a set of three boolean variables $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ and a set of two boolean functions $\{\mathsf{c} = F_{\mathsf{c}}(\mathsf{a}, \mathsf{b}), \mathsf{b} = F_{\mathsf{b}}(\mathsf{c})\}$. According to the recipe given in Section 3.1, a reaction network $\langle \mathcal{M}_{osc1}, \mathcal{R}_{osc1} \rangle$ is designed as follows:

$$\mathcal{M}_{osc1} = \{a, A, b, B, c, C\} \tag{3.22}$$

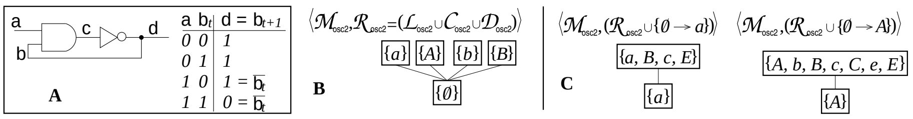


Figure 3.7: Result of analyzing a chemical reaction network implementing a controllable oscillator using chemical organization theory. (A) Circuit diagram and the dynamical oscillatory behavior described as a truth table. (B) Catalytic reactions $\mathcal{C}_{osc2}$ for amplifying feedback signals are introduced into the set of reactions $\mathcal{R}_{osc2}$. (C) When an influx $\emptyset \rightarrow A$ is added, a set $\{A, b, B, c, C, e, E\}$ is also found to be an organization which could be interpreted as the oscillatory behavior since pair-wise molecular species (*e.g.*, b and B) are both in the organization. The destructive reactions $\mathcal{D}_{osc2}$ allow alternative dis-/appearance of the two species.

and

$$\mathcal{R}_{osc1} = \mathcal{L}_{osc1} \cup \mathcal{D}_{osc1} \tag{3.23}$$

where

$$\mathcal{L}_{osc1} = \{a + b \rightarrow c, a + B \rightarrow c, A + b \rightarrow c, A + B \rightarrow C, c \rightarrow B, C \rightarrow b\},$$
$$\mathcal{D}_{osc1} = \{a + A \rightarrow \emptyset, b + B \rightarrow \emptyset, c + C \rightarrow \emptyset\}.$$

The reaction network $\langle \mathcal{M}_{osc1}, \mathcal{R}_{osc1} \rangle$ implements the oscillator circuit without input a specified so far. Without input, there are five organizations each containing not more than one molecular species: $\{\emptyset\}$, $\{a\}$, $\{A\}$, $\{b\}$, and $\{B\}$ As a result, there cannot be any oscillation.

The same is true, as expected, when the input variable a is initialized to 0. In that case the reaction network is modified to $\langle \mathcal{M}_{osc1}, (\mathcal{R}_{osc1} \cup \{\emptyset \rightarrow a\}) \rangle$. There are two organizations $\{a\}$ and $\{a, B, c\}$. The latter set of molecules corresponds to the expected (stationary) behavior of the boolean circuit.

For a $= 1$ the boolean circuit oscillates. When considering the corresponding reaction network $\langle \mathcal{M}_{osc1}, (\mathcal{R}_{osc1} \cup \{\emptyset \rightarrow A\}) \rangle$, the set $\{A\}$ is found to be the only organization. Hence, given dynamics, no matter how we initialize the reaction system, only molecules of species A and nothing else will inevitably remain after some transient, and there is obviously no oscillation possible. The reason for this behavior is that, apart from $\{A\}$, there is no set of molecular species that is self-maintaining. The lack of self-maintenance is due to the destruction of molecules through the reactions $b + B \rightarrow \emptyset$and $c + C \rightarrow \emptyset$, as long as there are molecules of type b, B, c, and C left.

3.5.2 Chemical Implementation with Amplified Feedback

The preceding investigation showed that the naively derived chemical system cannot oscillate like the boolean circuit because the necessary molecular species are not self-maintaining. A solution to this problem is to counteract the consumption of molecules by introducing an amplification mechanism for each periodically changing variables, as already noted in the last paragraph of Section 3.1

We chose variable b to be amplified, which is realized by a catalytic reaction.

The new reaction network $\langle \mathcal{M}_{osc2}, \mathcal{R}_{osc2} \rangle$ contains two additional molecular species e and E, which are produced instead of b and B, respectively (*i.e.*, they replace b and B in the previous reaction rules). Molecular species b and B are now produced by catalytic reactions of the form $e \rightarrow e+b$ and $E \rightarrow E+B$. We can see that b and B can now be consumed by other reactions without causing a drain of the output of the NAND gate.

The resulting chemistry is defined as follows: the set of molecular species is

$$\mathcal{M}_{osc2} = \{a, A, b, B, c, C, e, E\} \tag{3.24}$$

and the set of reaction rules is

$$\mathcal{R}_{osc2} = \mathcal{L}_{osc2} \cup \mathcal{C}_{osc2} \cup \mathcal{D}_{osc2} \tag{3.25}$$

where $\mathcal{C}_{osc2}$ is a set of catalytic reactions. The set of logical reactions becomes

$$\mathcal{L}_{osc2} = \{a+b \rightarrow c, a+B \rightarrow c, A+b \rightarrow c, A+B \rightarrow C, c \rightarrow E, C \rightarrow e\},$$

and the set of catalytic reactions is

$$\mathcal{C}_{osc2} = \{e \rightarrow e+b, E \rightarrow E+B\}.$$

Since there are four pairs of molecular species in the algebraic chemistry, the set of destructive reactions is now

$$\mathcal{D}_{osc2} = \{a+A \rightarrow \emptyset, b+B \rightarrow \emptyset, c+C \rightarrow \emptyset, e+E \rightarrow \emptyset\}.$$

Given the reaction network $\langle \mathcal{M}_{osc2}, \mathcal{R}_{osc2} \rangle$ implementing the controllable oscillator, chemical organization theory is applied to find organizations in the reaction network. The result of the analysis is shown in Figure 3.7 (B) as a hierarchy of organizations. The reaction network is extended by an influx to analyze the case in which variable a is initialized to 0 or 1. Figure 3.7 (C) shows the hierarchies of organizations found in the extended reaction networks. The smallest organizations for both input cases are composed of the single molecular species with influx because no reaction occurs without a feedback signal. For each input case, the biggest organizations correspond to the operational modes of the oscillator.

When influx $\emptyset \rightarrow A$ is added to the reaction network, the biggest organization is the set $\{A, b, B, c, C, e, E\}$. This implies that the pair-wise molecular species like b and B or c and C are sustained in a reaction vessel even though the two pair-wise molecules decay instantly upon collision due to the destructive reactions such as $b+B \rightarrow \emptyset$. An interpretation of the situation is the oscillating operational mode. Due to the amplified feedback reaction, coexistence of the pair-wise species is now possible.

3.5.3 Dynamical Simulation

To confirm the interpretation of the persistence of the pair-wise species in an organization, we stochastically simulated the reaction vessel using *C*opasi [Sahle et al., 2006]. As an initial state of the reaction vessel, it is necessary to have a positive concentration of non-input molecular species because input molecules

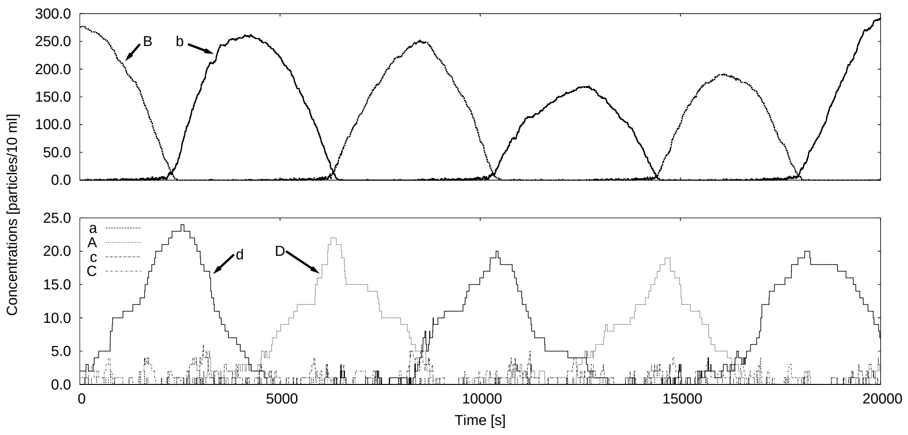


Figure 3.8: Dynamical oscillatory behavior of the chemical reaction network involving a feedback loop as shown in Figure 3.7. The loop is implemented with a catalytic reaction producing an input species using an output species as an catalyst. The upper figure shows the dynamical concentration changes of the species B and b, and those of the other species are shown in the lower figure. For all first- and second-order reactions, irreversible mass action kinetics is assumed, and the kinetic parameter is set to 0.01. An influx is assumed as an irreversible constant flux with a kinetic parameter of 0.001. The compartment volume is set to 10 ml.

a and A cannot produce anything else without other molecular species (*cf.* Figure 3.7 (D)). We chose the concentration of molecular species B as approximately 25 molecules per ml. The dynamical concentration profile in the reaction compartment is shown in Figure 3.8. Alternative appearance of the pair-wise molecular species b and B in the upper graph or d and D in the lower graph is apparent.

In general, we can say that a boolean circuit that has periodic attractors (*e.g.*, a circuit that can oscillate) will lead to chemical organizations that contain "contradicting" molecular pairs such as b and B. Thus we can take those organizations as indicators for oscillatory behavior. However, our theory does not allow to say more about the nature of that oscillation. Actually, it is possible that for specific rate laws chosen, we might obtain a stationary state in the chemical system, whereas the corresponding attractor of the boolean network is periodic. Under which circumstance this is the case and how periodic attractors appear in the light of chemical organization theory has to be studied theoretically in more detail in the future.

3.6 Conclusion

In this chapter, using various boolean functions as case studies, we demonstrated how chemical organization theory can be useful for constructing chemical reaction networks. When analyzing organizational structure in a reaction network implementing a logic gate, only the desired output form an organization within the network with input flows. This analysis concludes that the dynamical reaction systems based on the constructed reaction networks are certainly utilized for the chemical logic gates intended, and the conclusion was validated through simulations. The scalability of this analysis method is also demonstrated.

Significant results are illustrated in the cases of flip-flop and oscillator. The bistability of flip-flop is reflected by two organizations within the reaction network. Using chemical organization theory, we were able to explain the properties of the chemical flip-flop in a new, comprehensible way by referring to the Hasse diagram of organizations as a movement between organizations (Figure 3.5 (E)). This description is more compact than a classical description referring to the 8-dimensional concentration state space, as demonstrated in Section 3.4.2. The oscillatory behavior, on the other hand, causes chemical organizations containing “contradictory” molecular pairs such as a and A. In this light it should be noted that the chemical system is more complex than the original boolean circuit because an on- and off-signal can be present at the same time. Furthermore, variables can be unspecified, e.g. representing an unspecifed “open” input. Even in that case, the dynamics of the chemical system is well defined, as opposed to the boolean network.

source :

Matsumaru, N. and Dittrich, P. (2006). Organization-oriented chemical programming for the organic design of distributed computing systems. In *1st international conference on bio inspired models of network, information and computing systems (BIONETICS)*, volume 275 of *ACM International Conference Proceeding*, Cavalese, Italy. IEEE. also available at `http://www.x-cd.com/bionetics06cd/`.

Matsumaru, N., Lenser, T., Hinze, T., and Dittrich, P. (2007b). Designing a chemical program using chemical organization theory. *BMC Systems Biology*, 1(Suppl 1):P26. from BioSysBio 2007: Systems Biology, Bioinformatics, and Synthetic Biology, Manchester, UK, 11-13 January 2007.

Matsumaru, N., Lenser, T., Hinze, T., and Dittrich, P. (2007c). Toward organization-oriented chemical programming: A case study with the maximal independent set problem. In Dressler, F. and Carreras, I., editors, *Advances in Biologically Inspired Information Systems*, volume 69 of *Studies in Computational Intelligence*, pages 147–163. Springer, Berlin.

Chapter 4

Maximal Independent Set Problem

Contents

Our next target is the maximal independent set problem. The problem is defined in an undirected graph, and a solution is a set of vertexes such that there exists no direct edge between any of those vertexes. It is maximal when no more vertexes can be added in the set without violating that property. See Figure 4.1 for a simple example. A simple algorithm to find a solution can be described as follows: The set is initialized to contain all of vertexes. A vertex in the set is randomly selected, and neighboring vertexes are excluded from the set. Neighboring vertexes are a set of vertexes that are connected to the target vertex by an edge. Continuing these steps, the resulting set is a maximal independent set. Conventional algorithms to solve the maximal independent set problem were theoretically studied (e.g., [Luby, 1986]). The problem becomes NP-hard only when the largest maximal independent set in the given graph is to find, referred as maximum independent set problem.

What this maximal independent set problem makes interesting becomes apparent when discussed in the context of distributed computing. Assuming each vertex is a computing entity, and each edge is a communication link between entities so that it corresponds to distributed computing environments. Maximal independence is a global emerging property from local properties (namely, whether two vertexes are connected). Algorithms for a distributed computing environment to solve this problem are investigated for self-stabilizing properties [Shukla et al., 1995; Herman, 2003; Ikeda et al., 2002] because maximal independent set is stable in a global view. Since our motivation to chemical computing, or more generally bio-inspired computing, includes possible advantages on distributed computing, this problem is discussed in this chapter.

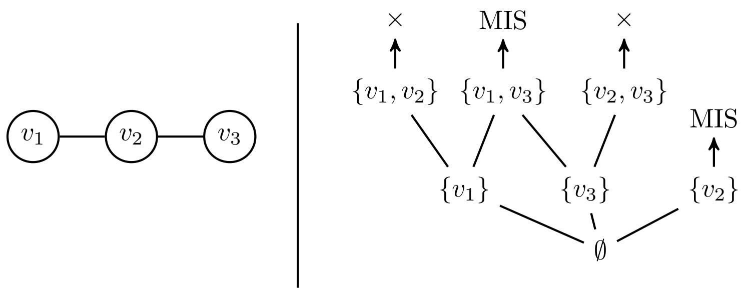


Figure 4.1: Illustration of the Maximal Independent Set (MIS) problem. (Left) The base undirected graph given consists of three vertexes linearly connected. (Right) Schematic representation of solving the MIS problem by adding a vertex. On the contrary to the simple algorithm described in the text, we start with the empty set for the explanatory purpose. The empty set is clearly the independent set. Adding vertex v_1 or v_3 keeps the property of independence in the set. These sets are not maximal yet since v_3 or v_1, respectively, can be added without violating the independence property. The set $\{v_1, v_3\}$ is a maximal independent set. Another solution to the MIS problem in this example is the set $\{v_2\}$. The solution to the *maximum* independent set problem is $\{v_1, v_3\}$ because of the largest size 2 while the other independent sets has size of 1 for $\{v_1\}$, $\{v_2\}$, or $\{v_3\}$; or 0 for $\emptyset$.

An application of this problem lies in wireless sensor networks to determine cluster-heads, which manage logical clustering structures in the networks. Sensor nodes that can communicate directly with the cluster-head form a cluster, and no cluster-heads are neighbors. Assigning a role to each sensor node benefits to increase lifetime of the whole network [Reichenbach et al., 2006].

In this chapter, we show how chemical organization theory helps programming distributed processes of chemical computing, taking the maximal independent set (MIS) problem as an example. Section 4.1 presents a chemical programming for that problem, converted and adapted from the algorithm proposed by Shukla et al. [1995]. In Section 4.2, the correspondence between MIS and chemical organizations is given as a proof[1]. Specific problem instances and the corresponding chemical programs are followed in Section 4.3. For each instance, the chemical program is analyzed using chemical organization theory, investigating the organizational structures embedded in the constructed reaction network. From these examples, the correspondence between MIS and the organizations is observed.

4.1 Chemical Programming for the MIS problem

Chemical reaction systems for MIS problem can be described shortly at first, then, formal description follows. The procedure starts with assigning two molecular species to each vertex, specifying the positive membership or negative membership in the set of vertexes. Those two pair-wise species are defined to vanish upon collision due to the contradictory states they are representing.

[1]with the great help of Thorsten Lenser

For each vertex, there are two sorts of reactions: one to promote adding the vertex to the set and the other to exclude the vertex from the set. The promoting reactions are defined for each vertex to react from the negative membership species of the neighboring vertex. The existence of positive membership species for all neighboring vertexes collaboratively reacts and produces negative membership species to exclude the vertex from the set.

Maximal independent set (MIS) is formally defined: Let an undirected graph $G = \langle V, E\rangle$ be defined by a set of N vertexes $V = \{v_1, \ldots, v_N\}$ and a set of edges E. When two vertexes v_p and v_q are connected, the pair of the vertexes is in the set of edges: $(v_p, v_q) \in E$. Note that the order of the pair is insignificant, that is, $(v_p, v_q) = (v_q, v_p)$. A set of vertexes $I \subset V$ is independent if no two vertexes in the set are adjacent: $(\forall v_p, v_q \in I : (v_p, v_q) \notin E)$. An independent set is maximal if no vertex can be added to the set while keeping the property of independence. Including another vertex in the MIS should violate the independence property.

We now present formally a procedure for designing chemical reaction networks solving MIS problem (see Table 4.1 for a short recipe). Given the undirected graph G, a reaction network $\langle \mathcal{M}, \mathcal{R}\rangle$ is designed as follows: For each vertex v_j, we assign two molecular species s_j^0 and s_j^1 representing the membership of the vertex in the MIS. The subscript of the species name corresponds to the index number of the vertex. High concentration of species s_j^1, higher than a threshold chosen to be smaller than any positive coordinate of any fixed point, means that the vertex v_j is included in the MIS. High concentration of species s_j^0 expresses that the vertex v_j is *not* included in the MIS. Thus the set of molecular species $\mathcal{M}$ contains $2N$ molecular species:

$$\mathcal{M} = \{s_j^0, s_j^1 \mid j = 1, \ldots, N\} \tag{4.1}$$

The set of reaction rules $\mathcal{R}$ is constructed by assembling reactions for each vertex:

$$\mathcal{R} = \bigcup_{i=1}^{N} \mathcal{R}^i = \bigcup_{i=1}^{N} (\mathcal{V}^i \cup \mathcal{N}^i \cup \mathcal{D}^i). \tag{4.2}$$

For each reaction set $\mathcal{R}^i$, there are three sorts of reactions. The first two sorts are adapted from two predicates constituting a program for any distributed processor to solve the MIS problem under a central scheduler [Shukla et al., 1995]. A reaction rule to produce species s_i^1 is the first:

$$\mathcal{V}^i = (\overbrace{s_j^0 + s_k^0 + \cdots + s_l^0}^{n_i} \rightarrow n_i s_i^1) \tag{4.3}$$

where n_i is the number of vertexes connected to vertex v_i and $v_j, v_k, \ldots, v_l$ are its neighboring vertexes, that is, $(v_i, v_j), (v_i, v_k), \ldots, (v_i, v_l) \in E$. The left-hand side of the reaction contains n_i terms, and this reaction is interpreted as follows: When no neighboring vertex is included in the MIS, the target vertex v_i should be included in the set.

The negation of this predicate is considered by a set of n_i reactions:

$$\mathcal{N}^i = \{s_j^1 \rightarrow s_i^0 | (v_i, v_j) \in E\}. \tag{4.4}$$

Table 4.1: Recipe for mapping an undirected graph to a chemical reaction network to solve maximal independent set problem.

Input: Undirected graph $G = \langle V, E\rangle$ where V is a set of N vertexes $V = \{v_1, \dots, v_N\}$ and E is a set of edges. When two vertexes v_p and v_q are connected, $(v_p, v_q) \in E$.
Output: Reaction network $\langle \mathcal{M}, \mathcal{R}\rangle$ (a set of molecular species $\mathcal{M}$ and a set of reaction rules $\mathcal{R}$) representing the chemical program to solve the maximal independent set problem.
Algorithm:

1. For each vertex v_j:

(a) Add two molecular species, s_j^0 and s_j^1, to $\mathcal{M}$;[2]

(b) Add one *destructive reaction* of the form $s_j^0 + s_j^1 \rightarrow \emptyset$ to $\mathcal{R}$;

(c) Add one reaction to $\mathcal{R}$ of the form:

$$(\dots + s_i^0 + \dots \rightarrow n_j s_j^1)$$

where n_j is the number of edges connected to vertex v_j and $(v_j, v_i) \in E$.

(d) Add a set of n_j reactions to $\mathcal{R}$:

$$\{s_i^1 \rightarrow s_j^0 | (v_i, v_j) \in E\}.$$

[2]As a naming convention of molecular species in this paper, the superscript indicates the membership for the maximal independent set.

This is the second type of reactions, which produce species s_i^0 from any species corresponding to the neighboring vertexes with superscript 1. This rule can be interpreted as follows: If there exists at least one neighboring vertex included in the MIS, then the target vertex v_i should be excluded from the maximal independent set (otherwise the definition of the MIS would be violated). Generating species s_i^0 forces vertex v_i not to be included in the set.

The last component of set $\mathcal{R}^i$ is a *destructive reaction*. Since the membership of the MIS is a binary state, the state becomes undefined when neither or both of the species are present. In order to avoid the latter case, the two opposite molecular species are defined to vanish upon collision:

$$\mathcal{D}^i = s_i^0 + s_i^1 \rightarrow \emptyset. \tag{4.5}$$

Note that the reaction network is defined such that molecules react only if they are located on the same vertex or are neighbors. Thus, the resulting (artificial) chemical system can be interpreted as a spatially distributed compartmentalized reaction system, where a compartment j holds only the two chemical species representing a vertex v_j, namely s_j^0 and s_j^1 and where the topological structure of the compartments is equivalent to the undirected graph.

4.2 Proof of exact correspondence between MISs and organizations of size N

Given an undirected graph $G = \langle V, E\rangle$ where $V = \{v_1, \ldots, v_N\}$ is a set of N vertexes and E is a set of edges, a reaction network $\langle \mathcal{M}, \mathcal{R}\rangle$ can be constructed as described in Section 4.1, where $\mathcal{M} = \{a_1, \ldots, a_{2N}\} = \{s_i^0, s_i^1 | i = 1, \ldots, N\}$ is a set of $2N$ molecular species and $\mathcal{R} = \{(A_j \to B_j)\}$ is a set of reaction rules. Here, we show with a proof that the constructed reaction network contains organizations with N species forming the largest and those organizations correspond to MIS in the given graph. To prove this, we first introduce the following lemma stating the maximum number of species each organization in the constructed reaction network can contain.

Lemma 1. *In the reaction network $\langle \mathcal{M}, \mathcal{R}\rangle$ constructed as described in Section 4.1, no organization can contain species s_k^0 and s_k^1 together. Therefore, no organization with a size (number of species) greater than N can exist.*

Proof. Let $O \subset \mathcal{M}$ be an organization, and suppose the organization contains s_k^0 and s_k^1 simultaneously ($s_k^0, s_k^1 \in O$) regarding vertex v_k. From the definition of the organization to be self-maintaining, there exists a flux vector $\mathbf{v} = (v_1, \ldots, v_{|\mathcal{R}|})^T$ satisfying the three conditions listed in Definition 2. Due to the third condition, the production rates f_i with respect to species belonging to the organization is greater than or equal to zero. Sum of those production rates should also be greater than or equal to zero.

$$\sum_{\{i|a_i \in O\}} f_i = \sum_{\{i|a_i \in O\}} \sum_{j=1}^{|\mathcal{R}|} m_{ij} v_{r_j} \geq 0.$$

Since organization O is also closed, the production rates for the species excluded from O should be zero: $f_i = 0$ if $a_i \notin O$. Therefore, this equation can be extended to:

$$\sum_{i=1}^{|\mathcal{M}|} f_i = \sum_{\{i|a_i \in O\}} f_i + \sum_{\{i|a_i \notin O\}} f_i = \sum_{i=1}^{|\mathcal{M}|} \sum_{j=1}^{|\mathcal{R}|} m_{ij} v_{r_j} \geq 0 \tag{4.6}$$

Here, j-th reaction $A_j \to B_j$ is denoted as r_j.

The set $\mathcal{R}$ of reaction rules can be divided into the three sets $\mathcal{V}, \mathcal{N}, \mathcal{D}$ (Section 4.1):

$$\begin{aligned}
&\textstyle\sum_{i=1}^{|\mathcal{M}|} f_i \\
&= \textstyle\sum_{i=1}^{|\mathcal{M}|} \left[\sum_{\{j|r_j \in \mathcal{V}\}} m_{ij} v_{r_j} + \sum_{\{j|r_j \in \mathcal{N}\}} m_{ij} v_{r_j} + \sum_{\{j|r_j \in \mathcal{D}\}} m_{ij} v_{r_j}\right] \\
&= \textstyle\sum_{\{j|r_j \in \mathcal{V}\}} \sum_{i=1}^{|\mathcal{M}|} m_{ij} v_{r_j} + \sum_{\{j|r_j \in \mathcal{N}\}} \sum_{i=1}^{|\mathcal{M}|} m_{ij} v_{r_j} \\
&\qquad\qquad + \textstyle\sum_{i=1}^{|\mathcal{M}|} \sum_{\{j|r_j \in \mathcal{D}\}} m_{ij} v_{r_j} \\
&= \textstyle\sum_{\{j|r_j \in (\mathcal{V} \cup \mathcal{N})\}} v_{r_j} (\sum_{i=1}^{|\mathcal{M}|} m_{ij}) + \sum_{i=1}^{|\mathcal{M}|} \sum_{\{j|r_j \in \mathcal{D}\}} m_{ij} v_{r_j}
\end{aligned} \tag{4.7}$$

For the reactions of type $\mathcal{V}$ and $\mathcal{N}$, sum of the stoichiometric coefficients is arranged to be zero:

$$\forall j | r_j \in (\mathcal{V} \cup \mathcal{N}) : \sum_{i=1}^{|\mathcal{M}|} m_{ij} v_{r_j} = 0 \tag{4.8}$$

Thus, the last term of Equation (4.7) must be non-negative:

$$\sum_{i=1}^{|\mathcal{M}|} \sum_{\{j | r_j \in \mathcal{D}\}} m_{ij} v_{r_j} \geq 0 \tag{4.9}$$

To keep this inequality, fluxes for the reactions of type $\mathcal{D}$ should be zero because all of the stoichiometric coefficients for the type $\mathcal{D}$ reactions are negative. The flux for reaction $(s_k^0 + s_k^1 \rightarrow \emptyset) \in \mathcal{D}$ must be set to a positive value, however, if both s_k^0 and s_k^1 are contained in organization O at the same time (Condition 1). The sum of the production rates cannot be positive, and at least one production rate has to be negative. This contradicts the definition of the organization, and hence, two species s_k^0, s_k^1 cannot coexist in the organization. □ □

Next we define the set of species induced by a set of vertexes:

Definition 9. *Let $I \subset V$ be a set of vertexes. We call $S_I = \{s_1^{b_1}, \ldots, s_{|V|}^{b_{|V|}}\}$ the set of species that is induced by I if $b_i = 1$ when $v_i \in I$ and $b_i = 0$ when $v_i \notin I$.*

Given a subset of vertexes $I \subset V$ in an undirected graph $G = \langle V, E \rangle$, a set of species S_I can be "induced" such that the subscript of the species name specifies the vertex identification number and the superscript is the binary state whether the vertex v_i is a member of set I ($b_i = 1$) or not ($b_i = 0$). The constructed set of species consists of $|V|$ species so that all of the vertexes in the graph are considered.

Using this notation, the exact correspondence between maximal independent sets and organizations of size $N = |V|$ can be stated as follows:

Theorem 2. *Given an undirected graph $G = \langle V, E \rangle$, a set $I \subset V$ of vertexes is a maximal independent set iff the induced set S_I of species is an organization.*

Proof. The first part of the proof is to show the necessary condition, namely a set of species induced by a MIS is an organization. Let $I \subset V$ be a MIS and S_I be the set of species induced by I. To show that S_I is an organization, two criteria will be tested: closure and self-maintenance.

Closure: Assume that S_I is not closed, i.e. there exists a reaction $(A_j \rightarrow B_j) \in \mathcal{R}$ that is applicable to S_I and that produces a species that is not in the set S_I. If such a reaction has the form of $(s_j^1 \rightarrow s_k^0) \in \mathcal{N}$ for an edge $(v_j, v_k) \in E$, then we know $s_j^1 \in S_I$ and thus $v_j \in I$. Since that reaction is assumed to violate the closure property, $s_k^0 \notin S_I$. The induced set of species S_I is defined to include either s_k^0 or s_k^1, so $s_k^1 \in S_I$. As a result, set S_I contains s_j^1 and s_k^1, meaning that set I includes both v_j and v_k. This leads to a contradiction that set I containing v_j and v_k is a MIS even though there is an edge $(v_j, v_k) \in E$.

The reaction to violate the closure property can have the form $(s_h^0 + s_l^0 + \cdots + s_m^0 \rightarrow n_k s_k^1) \in \mathcal{V}$. In that case no neighboring vertexes v_p with respect to v_k are included in set I because $s_p^0 \in S_I$ for any p such that $(v_p, v_k) \in E$. To violate the closure, s_k^1 should be excluded from S_I so that vertex v_k is not in I. Since none of neighboring vertexes are in I, however, v_k can be added to I with keeping the independence. This contradicts the fact that the independent set I is maximal.

The third type of reactions $(\mathcal{D})$ can be neglected because there are no products. From these arguments, no reaction can produce new species that is not in S_I. It follows that S_I is closed.

Self-maintenance: To satisfy the conditions of self-maintenance, the flux vector $\mathbf{v}$ is set as: $v_{r_j} = 1$ if $A_j \in \mathcal{P}_M(S_I)$ and $v_{r_j} = 0$ if $A_j \notin \mathcal{P}_M(S_I)$. Given this $\mathbf{v}$, we show that production rates for all species in S_I are non-negative. From the definition of set S_I induced by a set of vertexes, either s_m^0 or s_m^1, not both, is included in S_I. Therefore, The fluxes for reaction type $\mathcal{D}$ are set to zero.

Assume a species with the superscript of one, s_m^1, is in S_I. This implies $s_p^0 \in S_I$ for any neighboring vertexes v_p of vertex v_m since v_m is in MIS but none of neighboring vertexes v_p are included. There is only one reaction in $\mathcal{V}$ producing s_m^1 with the stoichiometry of n_m, where n_m is the number of vertexes connected to vertex v_m, and containing reactants $A_j \in \mathcal{P}_M(S_I)$. Production rate of species s_m^1 caused by reaction type $\mathcal{V}$ is calculated to n_m because flux for the reaction is set to 1. In the type of $\mathcal{N}$, there are n_m reactions with s_m^1 as the reactant. Setting the fluxes to 1 for these reactions, the production rate caused by this type of reactions is $-n_m$. Combining those, the production rate of species s_m^1 is 0.

Next, a species with the superscript of zero, s_m^0, is assumed to be in S_I. Since $v_m \notin I$ from the definition, at least one or maximum n_m neighboring vertexes are in the MIS I. Let $1 \leq g \leq n_m$ be the number of neighboring vertexes in the MIS I. There are g reactions of type $\mathcal{N}$ applicable to S_I depending on I. Stoichiometric coefficients of these reactions for s_m^0 are all 1. In the reactions of type $\mathcal{V}$, the same number of reactions as type $\mathcal{N}$ are applicable. For a vertex in the MIS, no neighbors are included in the MIS according to the definition. It follows that a reaction in $\mathcal{V}$ should be applicable for each vertex included in the MIS. If v_m has g neighboring vertexes included in the MIS I, there have to be g reactions of type $\mathcal{V}$ with the coefficients -1. Hence, production rate of the species s_m^0 is equal to zero.

The second part shows the sufficient condition. Namely, if a set of species induced by a set of vertexes is an organization, then this set of vertexes is a MIS. Given a set of vertexes I and its induced set of species S_I, which is an organization, we need to show that I is a MIS. Taken any of two vertexes v_p and v_q from the set I, we know that $s_p^1 \in S_I$ and $s_q^1 \in S_I$ from the definition of the induced set of species. Suppose there exists an edge $(v_p, v_q) \in E$ between those two vertexes. The reaction $s_p^1 \rightarrow s_q^0$ defined for that edge would produce s_q^0 inside the organization S_I. In order to keep S_I as an organization, s_q^1 and s_q^0 should coexist, which is impossible according to Lemma 1 given above. Therefore, we conclude that $(v_p, v_q) \notin E$ and I is an independent set.

If I does not represent a "maximal" independent set, we could add a vertex $v_{p'} \in V \backslash I$ to I, and $I \cup \{v_{p'}\}$ remains an independent set. To be more general, there exists a non-empty set of vertexes $I' \subseteq V \backslash I$ such that the union of these sets $I \cup I'$ becomes the MIS. Since we showed in the first part of this proof that a MIS induces an organization, $S_{I \cup I'}$ induced by set $I \cup I'$ is an organization whereas S_I is also an organization from the assumption. Set $S_{I \cup I'}$ differs from S_I with respect to any vertex $v_{p'}$ in I': $s^0_{p'} \in S_I$, $s^1_{p'} \in S_{I \cup I'}$. Those indexes p' have to be chosen such that vertexes $v_{p'}$ are not in I and no neighboring vertexes $v_{q'}$ are in set I. If vertex $v_{q'}$ is already contained in I and vertex $v_{p'}$ is added, then set $I \cup I'$ cannot be an independent set. However, the absence of those neighboring vertexes $v_{q'}$ in S_I would produce $s^1_{p'}$ by the reactions in $\mathcal{V}$, which violates the closure property of the organization S_I because the set only contains $s^0_{p'}$. Thus, there are no such indexes p', and set I' is an empty set. □ □

4.3 Examples of Chemical Programming to Solve the MIS Problem

4.3.1 Linear graph with three vertexes

Provided that an undirected graph $G = \langle V, E \rangle$ consists of three vertexes and those vertexes are connected linearly as shown in Fig. 4.2 (A):

$$G = \langle \{v_1, v_2, v_3\}, \{(v_1, v_2), (v_2, v_3)\} \rangle. \tag{4.10}$$

Following the recipe, a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ is constructed. The set of molecular species $\mathcal{M}$ consists of six species because the graph contains $N = 3$ vertexes:

$$\mathcal{M} = \{s^0_1, s^1_1, s^0_2, s^1_2, s^0_3, s^1_3\}. \tag{4.11}$$

Our naming convention for the species is that the subscript of the species name is associated with the index of the graph vertex and that the superscript stands for the membership of the MIS. For example, species s^1_2 stands for vertex v_2 being included in the MIS, and s^0_2 represents the opposite for the same vertex.

For each vertex v_1, v_2, and v_3, reaction rules are constructed. The destructive reactions are:

$$\mathcal{D} = \bigcup_{i=1}^{3} \mathcal{D}^i = \{s^0_1 + s^1_1 \to \emptyset, s^0_2 + s^1_2 \to \emptyset, s^0_3 + s^1_3 \to \emptyset\}.$$

The reaction rules to produce positive membership species are composed of three reactions:

$$\mathcal{V} = \bigcup_{i=1}^{3} \mathcal{V}^i = \{s^0_2 \to s^1_1, s^0_1 + s^0_3 \to 2s^1_2, s^0_2 \to s^1_3\}$$

Finally, the non-membership species are also produced:

$$\mathcal{N} = \bigcup_{i=1}^{3} \mathcal{N}^i = \{s^1_2 \to s^0_1, s^1_1 \to s^0_2, s^1_3 \to s^0_2, s^1_2 \to s^0_3\}$$

The whole set of reactions $\mathcal{R}$ results in:

$$\begin{aligned}\mathcal{R} &= \mathcal{V} \cup \mathcal{N} \cup \mathcal{D} \\ &= \{s_2^0 \to s_1^1, s_2^1 \to s_1^0, s_1^0 + s_3^0 \to 2s_2^1, \\ &\quad s_1^1 \to s_2^0, s_3^1 \to s_2^0, s_2^0 \to s_3^1, s_2^1 \to s_3^0, \\ &\quad s_1^0 + s_1^1 \to \emptyset, s_2^0 + s_2^1 \to \emptyset, s_3^0 + s_3^1 \to \emptyset\} \qquad (4.12)\end{aligned}$$

The reaction network is analyzed for its hierarchical organizational structure within the reaction network. When applying chemical organization theory (Chapter 1), the chemical reaction network is decomposed into a hierarchy of overlapping sub-networks, called organizations. These organizations provide an overview of the potential (emergent) behavior of the system because only a set of molecular species forming an organization can be stable [Dittrich and Speroni di Fenizio, 2007]. Furthermore, the dynamics of the system can be explained as a transition between organizations instead of a movement in the potentially more complex state space.

In our example, the reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ possesses five organizations:

$$O = \{\emptyset, \{s_1^0\}, \{s_3^0\}, \{s_1^0, s_2^1, s_3^0\}, \{s_1^1, s_2^0, s_3^1\}\} \qquad (4.13)$$

Figure 4.2 (B) visualizes these organizations as a Hasse diagram. In passing we note that the organizations do not form a lattice, because there is not a unique largest organizations. The two largest organizations represent the two desired solutions to the MIS problem, namely "010" and "101". This explains that in a dynamical reaction system implementing the designed reaction network, the species combinations representing desired solutions are more likely to stay in the dynamical system and the other solutions consisting of species that are not an organization cannot stably exist (cf. [Dittrich and Speroni di Fenizio, 2007]). Interestingly the analysis has also uncovered three smaller organizations. These organizations represent uncompleted computations due to a lack of molecules. For example, the empty organization trivially implies: if there are no molecules in the system, no molecule will enter the system and there will be no computation. If we setup our chemical computing system such that these small organizations are avoided, the system must produce a solution.

We can now ask whether these solutions, organizations $\{s_1^0, s_2^1, s_3^0\}$ and $\{s_1^1, s_2^0, s_3^1\}$, are stable or whether the system, once they have been found, might move spontaneously down to a smaller organization below them. In general, this type of question requires to investigate the dynamics, such as, rate constants, in detail. Here, however, we can see already by looking at the reaction rules that organization $\{s_1^1, s_2^0, s_3^1\}$ must be stable, because all reactions are mass-conserving so that the empty organization (the only organization below) can never be reached. The deficiency value [Feinberg and Horn, 1974] for the organization is calculated to be zero so that the asymptotically stable state is contained, assuming mass action kinetics. The situation with organization $\{s_1^1, s_2^0, s_3^1\}$ is more complicated. It contains also the small organizations $\{s_1^0\}$ and $\{s_3^0\}$, and the deficiency value for the organization is one. Hence, we cannot use the same argument as before. The stability of that organization depends on the kinetics applied (not shown here).

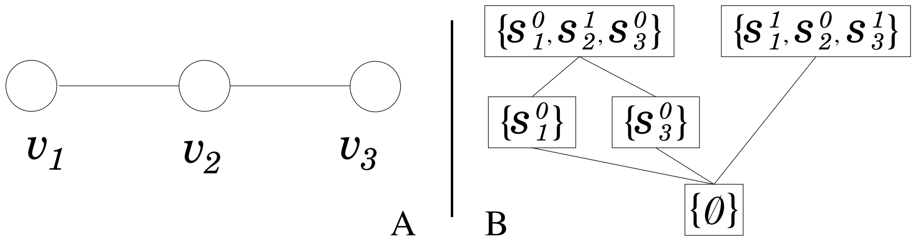


Figure 4.2: Analysis of a chemical program with organization theory. (A) Graph structure and (B) hierarchy of organizations within the chemical reaction network for the maximal independent set problem for the linear 3-vertex graph.

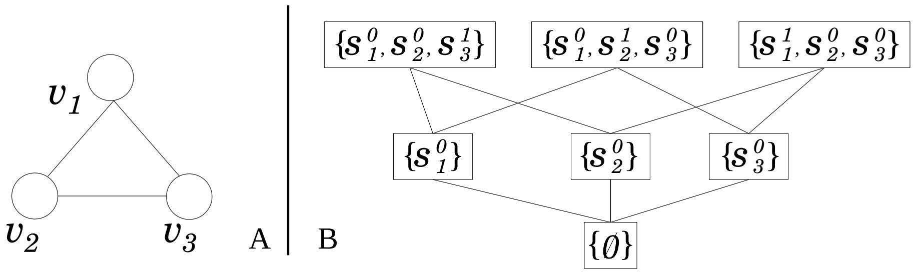

v_1
v_2
v_3
A
B
$\{s_1^0, s_2^0, s_3^1\}$
$\{s_1^0, s_2^1, s_3^0\}$
$\{s_1^1, s_2^0, s_3^0\}$
$\{s_1^0\}$
$\{s_2^0\}$
$\{s_3^0\}$
$\{\emptyset\}$

Figure 4.3: Analysis of a chemical program with organization theory. (A) Graph structure and (B) hierarchy of organizations within the chemical reaction network for the maximal independent set problem for the circular 3-vertex graph.

4.3.2 Circular graph with three vertexes

The similar discussion is applicable to the circular graph structure. For instance, three vertexes are connected as depicted in Fig. 4.3 (A) to form a circular structure. The undirected graph can be defined as follows:

$$G = \langle V = \{v_1, v_2, v_3\}, E = \{(v_1, v_2), (v_2, v_3), (v_1, v_3)\}\rangle. \tag{4.14}$$

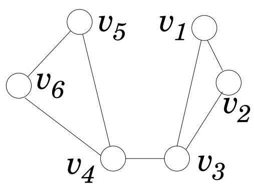
According to the recipe, a reaction network is constructed, and the resulting network is following:

$$\mathcal{M} = \{s_1^0, s_1^1, s_2^0, s_2^1, s_3^0, s_3^1\} \tag{4.15}$$

and

$$\begin{aligned} \mathcal{R} \quad = \quad & \{s_2^0 + s_3^0 \to 2s_1^1, s_2^1 \to s_1^0, s_3^1 \to s_1^0, \\ & s_1^0 + s_3^0 \to 2s_2^1, s_1^1 \to s_2^0, s_3^1 \to s_2^0, \\ & s_1^0 + s_2^0 n \to s_3^1, s_1^1 \to s_3^0, s_2^1 \to s_3^0, \\ & s_1^0 + s_1^1 \to \emptyset, s_2^0 + s_2^1 \to \emptyset, s_3^0 + s_3^1 \to \emptyset, \}. \end{aligned} \tag{4.16}$$

Analyzing this reaction network reveals seven overlapping organizations as shown in Fig. 4.3 (B). The largest organizations are composed of three species,



A | B

$\{s_1^1, s_2^0, s_3^0, s_4^0, s_5^0, s_6^1\}$ $\{s_1^0, s_2^1, s_3^0, s_4^1, s_5^0, s_6^0\}$

$\{s_1^1, s_2^0, s_3^0, s_4^1, s_5^0, s_6^0\}$ $\{s_1^0, s_2^1, s_3^0, s_4^0, s_5^1, s_6^0\}$

$\{s_1^1, s_2^0, s_3^0, s_4^0, s_5^1, s_6^0\}$ $\{s_1^0, s_2^0, s_3^1, s_4^0, s_5^0, s_6^1\}$

$\{s_1^0, s_2^1, s_3^0, s_4^0, s_5^0, s_6^1\}$ $\{s_1^0, s_2^0, s_3^1, s_4^0, s_5^1, s_6^0\}$

Figure 4.4: Analysis of a chemical program with organization theory. (A) Graph structure with six vertexes and seven edges. (B) The largest organizations within the chemical reaction network for the maximal independent set problem with respect to the graph. Each organization with the size of six corresponds to a solution to the maximal independent set problem.

and each species specifies the different vertex state.

$$\{s_1^1, s_2^0, s_3^0\}, \{s_1^0, s_2^1, s_3^0\}, \{s_1^0, s_2^0, s_3^1\} \tag{4.17}$$

Apparently, each organization corresponds to a solution to the maximal independent set problem on this graph structure. When vertex v_1 is included in the maximal independent set, the other two vertexes should not be in the independent set.

4.3.3 Graph with 6 vertexes

Two circular graphs with three vertexes are connected as shown in Fig. 4.4 (A) so that both circles and lines are contained. Since the graph consists of six vertexes, the reaction network holds 12 molecular species. Twenty-six reactions among those species constitute the reaction network. Within that reaction network, there are 49 organizations in total. In Fig. 4.5, a whole hierarchy of the organizations is shown, and only the largest organizations with six species are listed in Fig. 4.4 (B). Focusing on the largest organizations within the reaction network, only the sets of species representing solutions to the maximal independent set problem are found to be the organization.

4.4 Conclusion

We have shown that chemical organization theory can serve as a tool to predict the potential behavior of a chemical program given its "microscopic" reaction rules, without need to know the kinetics in detail. The desired solutions to the MIS problem appeared as organizations. Furthermore, the organizational analysis uncovers organizations representing incomplete computations due to a lack of molecules. Chemical organization theory can now guide further improvements of the chemical program, which aim at reducing or even removing completely these "undesired" organizations.

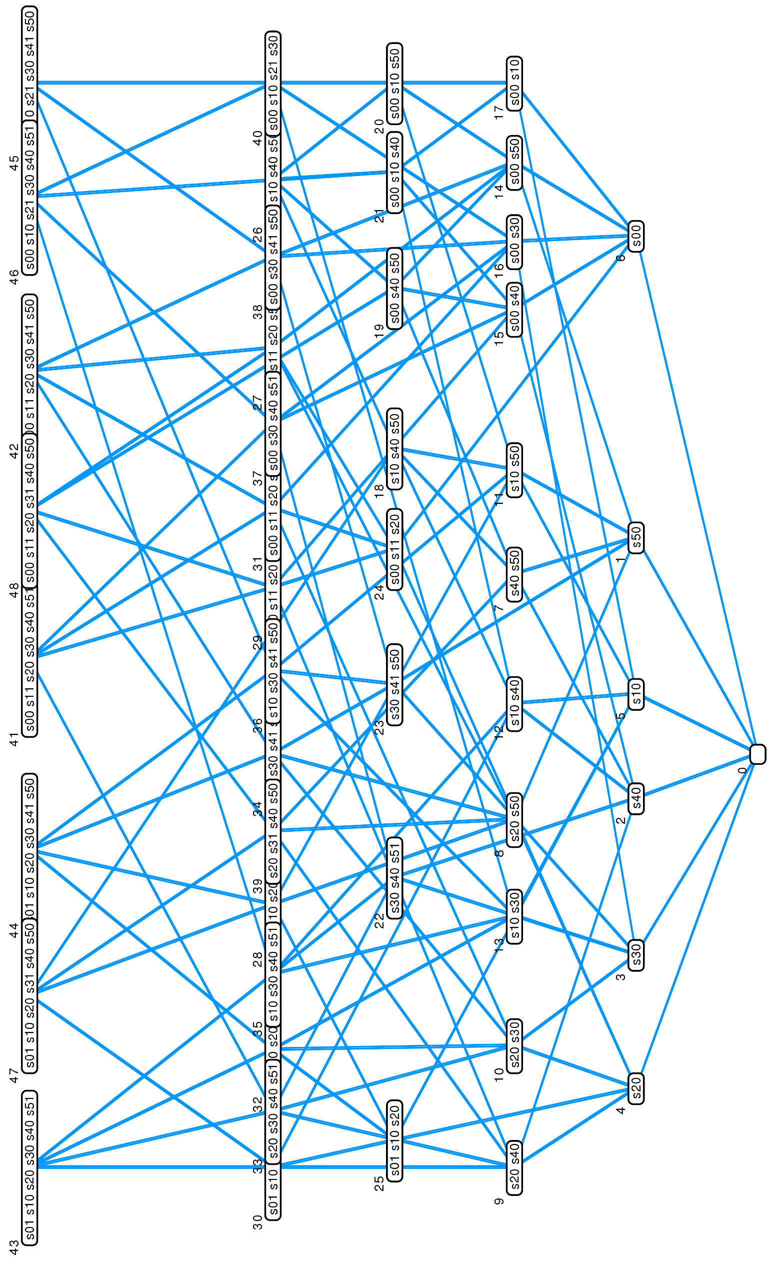


Figure 4.5: Hierarchy of chemical organizations within the reaction network programmed to solve the maximal independent set problem in the graph structure depicted in Fig. 4.4 (A). There are 49 organizations in total, and eight organizations with six species are the largest on top of the diagram. The potential dynamical behaviors of the reaction network to solve the maximal independent set problem appear as the largest organizations.

Source :

Dittrich, P. and Matsumaru, N. (2007). Organization-oriented chemical programming. In *7th International Conference on Hybrid Intelligent Systems (HIS)*, IEEE Conference Proceedings, pages 18–23. IEEE.

Chapter 5

Conclusion and Outlook

Contents

In the previous chapters, a constructive approach as a design method for chemical computing has been exemplified. In these examples, a general constructive design principle of chemical computing systems are conformed. That design principle is summarized and discussed here.

The term “constructive” refers to a *bottom-up* approach such that the reaction rules are manually and rationally chosen or designed to build the whole network. A great deal of this design procedure depends on the programmers' intuition about how the constructed program would behave, similar to conventional programming paradigms. Especially for the chemical programming paradigms, however, perceiving that intuitive image of the system behavior becomes quickly overwhelming because of the highly parallelized computation processes. Furthermore, counting fully on the programmers' intuition is not an entirely adequate way to develop reliable systems. Hence, an analytical tool to help understanding dynamical behaviors of the chemical programs is highly beneficial.

We programmed a chemical computing system at the level of reaction networks, a pair of a set of molecular species and a list of reaction rules among those. By restricting ourselves to modify those lists for programming processes, we were able to employ the notion of chemical organizations for that purpose. Chemical reaction networks were constructed so that, when analyzing the organizational structure within, the desired output forms an organization. Through case studies in Chapter 3 with boolean functions, we observed a noticeable association between output behaviors and organizational structure. When implementing flip-flop with two steady states, the bistability reflected two different organizations in the constructed network. It is also important to note that there was an undesired organization below those two, which could not be removed. Similarly, examples of organizational structures with respect to MIS problem in Chapter 4 are composed of both desired organizations and undesired. Furthermore, the organizational analysis uncovers organizations representing incomplete computations due to a lack of molecules. Chemical organization theory can now guide further improvements of the chemical program, which aim at reducing or even removing completely these undesired organizations. This idea has been outlined by Dittrich and Matsumaru [2007] as general principles of "organization-oriented chemical programming", presented also in Section 5.1. As the outlook, in Section 5.2.1, we present a benchmark problem so that a quantitative evaluation of the chemical system properties is possible. Moreover, in Section 5.2.2, our vision of a technical environment is sketched, providing an architecture for easy implementation of a chemical program.

5.1 Organization-Oriented Programming

Chemical computing can be distinguished whether a computation takes places within one organization or whether the computation can be explained as a movement between organizations. Computation within one organization exploits quantitative changes of species concentration in a chemical system. For an example, chemical reaction system developed by [Deckard and Sauro, 2004] computes the square-root so that the final concentration x_T of a molecular species is the square root of the initial concentration x_0, $x_T = \sqrt{x_0}$. Enzymatic computation investigated by Zauner and Conrad [2001] is another example of chemical computing within one organization in general because the concentration value of the enzymatic reaction products is mapped to outcomes of the computation.

Computation between organizations, on the other hand, is characterized by the qualitative state changes of the reaction system, such as appearance of new species and/or disappearance of existing species. The classical DNA computing [Adleman, 1994] is a case of such computation. Through the computation processes, the combination of molecular species present in the reaction system changes over time. The fundamental assumption of organization-oriented programming is that the change should be understood as a movement between chemical organizations.

In passing, the qualitative state changes is not the necessary condition of the organization-oriented approach. In other words, not all of the qualitative state changes can be understood as a movement between chemical organizations because not all of the species combinations are organizations. The reaction system may be in a state where the consisting species combination is not an organization although strongly depending on dynamical implementations. Dynamical experimental setup of the reaction system to implement chemical programs should be arranged so as to end up in such an organization. Alternatively, theory of chemical organization provides a view such that a chemical organization covers not only the exact species combination but also the other combinations from which the organization is "generated" [Dittrich and Speroni di Fenizio, 2007].

5.1.1 Design Principles

When following an organization-oriented approach, we first concentrate on the reaction network neglecting kinetic laws. The reaction network is designed with respect to its organizational structure, considering the following principles P1-P5. Then, in the second step, the kinetics including kinetic parameters is specified for fine tuning of the computation as stated in P6. the kinetic laws determine the dynamics between and inside organizations.

P1: There should be one organization for each output behavior class Assume that computation appears as a movement between organizations, and the output behavior can be categorized in different discrete behavior classes. That is, species combinations in the computational reaction system uniquely identify the output behavior. The reaction network should be designed so that there should exist at least one organization corresponding to each output behavior class categorized. For an instance, the output behavior of RS flip-flop logic circuit on the hold operation are categorized into two: reset state and set state. The chemical flip-flop described in Section 3.4 are designed to contain two organizations in the base reaction network, which correspond to two different states of the flip-flop.

P2: The set of molecular species (and the organization) representing a result should be in the closure of the species representing the initial input The closure denotes a set of molecular species that is generated by adding all possible reaction products until no more new species can be produced. This principle assures that there is a reaction path from the initial input configuration to the desired output species. Otherwise, the desired output will not appear as a result of the computation. The chemical logic gates described in Section 3.2 or Section 3.3 are designed with strictly following this principle.

Furthermore, it is expected that the desired output set is contained in a self-maintaining set within that closure. The self-maintenance property of the set of molecular species indicates theoretical possibilities to sustain all the species in the dynamical reaction systems, so the desired output species may be sustained in the reaction system until the outcomes of the computation is observed. The ideal case is that the desired output is represented by a largest self-maintaining set within that closure. In case that there exists a larger self-maintaining set than the desired output set, the dynamics may settle above the desired one. This argument leads to the next principle.

P3: The set of molecular species representing an input should generate the organization representing the desired output To generate the organization from a set of species, by definition given in Section 8, the closure of the given set is taken at first. Then we remove species until we reach a largest self-maintaining set contained in the closure. This principle will be fulfilled on the following two conditions: the desired output is contained within the closure of the input (P2 is fulfilled), and the largest self-maintaining set contained in the closure corresponds to the desired output. For the chemical logic gate examples in Section 3.2 and Section 3.3, this case was also fulfilled. When inflow reactions are added, the closure of the input species turns out to be also self-maintaining.

The largest self-maintaining set within a closure is not always unique in general although it is uniquely generated in a specific class of reaction networks, called semi-consistent [Dittrich and Speroni di Fenizio, 2007]. In chemical computing, the uniqueness is not required. It can be even beneficial, on the contrary, as in the chemical flip-flop in Section 3.4. The two distinctive organizations are largest self-maintaining sets contained in the closure of all species.

P4: Eliminate organizations not representing a desired output Since each organization potentially includes fixed points, the reaction system's dynamics may converge to one of the organizations. Hence, it makes sense to eliminate organizations not representing an output in order to avoid false computational outputs. This can be achieved by destroying either its closure property or its self-maintenance.

P5: An output organization should have no organization below The dynamics of the reaction system that moves from one organization O_1 to another O_2 below (*i.e.*, $O_2 \subset O_1$) is called a downward movement. This dynamical move can be theoretically prevented by the self-maintenance property with the right kinetics. Practically speaking, this move may occur spontaneously due to, *e.g.*, stochastic effects because the self-maintenance property only ensures the possibilities to sustain all species. Following this principle, a downward movement can be restricted.

P6: Assure, if possible, stoichiometrically the stability of an output organization Instead of eliminating organizations below the desired output as in the previous principle P5, the downward movement can be ruled out by purely stoichiometric argument. It may be possible to design the reaction network such that the organization representing the desired output is stable for any kinetic law. As a simple example consider the system $\mathcal{R} = \{a \to b, b \to a\}$, which has two organizations: $\{\emptyset\}$ and $\{a, b\}$. Doe to mass-conservation, the system can never move spontaneously from the organization with two species to the empty one.

P7: Use kinetic laws for fine tuning The kinetic laws determines the systems' behavior within an organization and the transition dynamics between organizations. One of rationals for the right kinetics is to assure that the dynamical reaction systems are stable in the output organizations, restricting mainly the downward movement. Finding the right kinetic laws is in general a non-trivial task. However, the existence of such laws is ensured by chemical organization theory to a certain extend, and we have seen in the examples (Chapter 3 and 4) that following principles P1-P6 simplify this tasks significantly. Classical dynamical systems theory is certainly reliable for this task, and it is even possible to derive at lease in some cases rigorously dynamical stability from network structure [Clarke, 1980; Feinberg and Horn, 1974]. Another point of consideration is a trade-off between that stability and the speed of computation since chemical reaction systems may compute by moving amongst organizations.

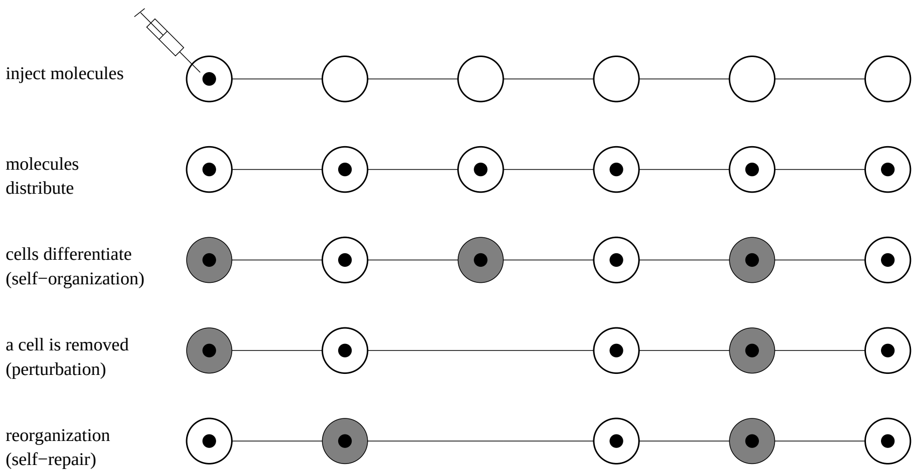

inject molecules
molecules distribute
cells differentiate (self−organization)
a cell is removed (perturbation)
reorganization (self−repair)

Figure 5.1: Scenario of a benchmark problem, where a set of sensor nodes have to differentiate such that pairwise neighbors are in different states.

5.2 Outlook

5.2.1 Benchmark problem scenario

To allow not only qualitative but also quantitative evaluation of our approach, a benchmark problem is desirable. We envision a variant of the maximal independent set problem addressed in Chapter 4 as a simple benchmark. The scenario, shown in Figure 5.1, takes place on a distributed computational system, expressing self-organizing and self-repair (or self-adaptive) behaviors. Computing entities are linked, and the communications between them are established via the link. An actual application area of this scenario is sensor networks Culler et al. [2004] because of the following reasons: Since the network consists of a large number of small sensor particles spread over a large area, self-organizing properties are practical to take control over such systems. In fact, this idea is the motivations of the research field *Organic Computing* [Müller-Schloer, 2004]. Self-adaptability is demanded because of unpredictable change of network structure and environments.

A simple benchmark problem is sketched in Figure 5.1: Assume that sensor nodes are arranged linearly. Specific molecules are distributed over the network. Then the network should self-organize such that pairwise neighboring nodes are in different states, for example, one class should perform a measurement at night and the other at daytime. When nodes are removed or added dynamically, spontaneous reconfiguration should occur (self-repair). The recovery time or number of acceptable perturbations can serve as a quantitative measure of the systems performance.

5.2.2 Potential technical environment

For a concrete application a chemical programming environment and a runtime system as sketched in Fig. 5.2 could be implemented. It consists of a compiler that takes a high level description of a chemical program as input. A chemi-

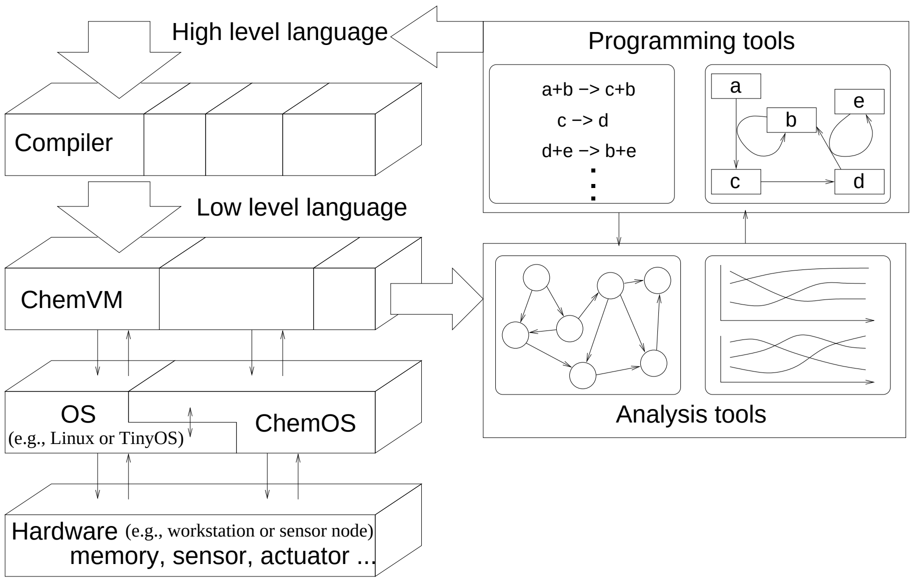


Figure 5.2: Schematic representation of architecture of the chemical programming workbench. ChemVM: a virtual machine that is able to run a (low-level) chemical program. Compiler: compiles a high-level chemical language to a lower-level language that can be run on a ChemVM. ChemOS: its main task is to handle input-output to other (conventional) software processes running on the same system, or to hardware sensors and actuators. The architecture should allow to "plug in" different compilers, which may compile the same program to different virtual machines, e.g., a deterministic or a stochastic machine.

cal program consists of a list of molecules and reaction rules including kinetic laws. The compiler generates "chemical byte code", which can be processed by the chemical virtual machine. The advantage of the compilation step is that different chemical languages can be run on the same virtual machine. For example, a simple language where molecules are just symbols and reaction rules are explicit transformation rules, or a more complex language where molecules posses a structure and reaction rules are defines implicitly by referring to that structure (*e.g.*, prime number chemistry). The virtual machine requires some input-output functionality, which is partly taken from the underlying operating system. Special communication between the chemical program and other hardware, such as sensors or actuators, is handled by the ChemOS (chemical operating system).

The sketched architecture and the theoretical approach exemplified throughout this Part should lead to a practical framework for "chemical programming". By doing so, we expect to make available a technology that allows to create computational systems with the properties of their biological counterpart.

Autonomous Design

Introduction

In the previous part, we followed a constructive programming strategy where modifications of the reaction networks are made only by a rational human programmer. For that approach, it is inevitable that the target problem should be divided into sub-problems, or at least the programmer should be aware of the structure of the problem intuitively or logically. This is similar to conventional programming. While that constructive approach for designing the chemical reaction network *in vivo* has been also pursued by [Guido et al., 2006], our focus here is to design chemical systems in an autonomous manner. The main difference is that human interactions during the programming process are minimized. As a result, the programmer is now free of the structural analysis of the problems. In other words, it is not necessarily clear for the programmer *how* the target problem is solved. The most popular comparison of the these two approaches may be the systems engineered and evolved.

Comparing the two approaches of constructing and evolving, the engineered system tends to be simpler and more effective because irrational components are usually omitted. These simplicity and effectiveness lead to the stiffness so that the system is intolerant to modifications. Tiny modifications cause unpredictable effects on the system's, and the effects are normally negative such as function failure. There may be also a case such that major system changes cause no effects at all. On the other hand, evolved systems embraces more components than necessary, and that extra complexity may lead to distinctive characteristics such as robustness and adaptability. The evolved systems may be robust against a functional failure of components, for example, by assigning an identical sub-function to different components in order to sustain the functionality as a whole. This mechanisms also influenced engineers so that a module to keep a complete duplicate may be explicitly implemented in the system although this is out of favor because it just doubles the necessary space and resources without definitive payoffs. Güdemann et al. [2007] modeled self-adaptive behaviors of systems with three kinds of robotic tools such that a malfunctioning robots are substituted by the other robots. The adaptability, in principle, is another aspect of evolvability for biological organisms to adapt to the ever-changing environments. Making use of this evolvability is believed to distinguish the chemical computing systems from conventional computers because conventional computing systems exhibit severe difficulties on this point due to the intolerance to modifications. Hence, one main theme in this part is to program chemical reaction systems by evolution, and we focus on the analysis of the evolutionary process. This investigation may lead to the deeper understanding of evolvability.

As autonomous design techniques, two approaches constitute this Part, and the first is to employ the principles of evolution. The next Chapter 6 discusses programming chemical reaction network using evolutionary algorithms, based on a work together with Lenser et al. [2008], in comparison with constructed reaction systems. In Chapter 7, the idea to understand evolutionary processes using chemical organization theory is further elaborated. Chapter 8 and 9 present the other approach of autonomous design, cooperating with principles of exploration. This explorative approach is fundamentally different because programming is not associated with modification of chemical computing systems. Instead, systems are explored and searched for interesting behaviors. The basic idea is that an autonomous system is used, as a preliminary step, to explore the behavior of the chemical reaction system. Then, a specific aspect of the system's behavior will be utilized for a particular computational purpose.

Those two approaches, evolution and exploration, share the common assumption: the effects of changing a reaction system are hard to predict in advance. The evolutionary design approach modifies reaction networks at random. Whether the modifications are fit or not is evaluated after those have been made. A prediction process is not involved in this way of programming. For the explorative approach, the intention to alter reaction systems are even disregarded. When dealing with natural systems, however, this assumption is believed to be appropriate since factor interactions established within the systems are complex and tangled. A substantial amount of efforts must have been spent to prevent undesired side-effects in order to implement a theoretical blueprint of chemical computing systems. Moreover, that undesired behaviors can be novel. Extensive interactions between components should not be hindered so that unpredictable behaviors occur. That way, there would be possibilities to utilize abundant complexities embraced within those natural systems. Controlling or programming such systems is difficult, but price of programmability may be received as other systems' feature of computational efficiency and adaptability [Conrad, 1988].

Source :

Lenser, T., Matsumaru, N., Hinze, T., and Dittrich, P. (2008). Tracking the evolution of chemical computing networks. In Bullock, S., Noble, J., Watson, R. A., and Bedau, M. A., editors, *Proceedings of the Eleventh International Conference on Artificial Life*. MIT Press, Cambridge, MA.

Chapter 6

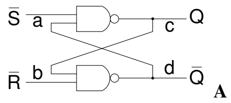
Comparing Evolved Reaction Networks with Constructed Reaction Networks

Contents

In a previous work, Lenser et al. [2007] have developed a software designed to evolve biological networks (called SBMLevolver) and measured the performance impact of certain design decisions for that algorithm. Adopting that software package, we evolved a reaction network capable of flip-flop operation [Lenser et al., 2008] [1], on which this chapter is based. There are mainly two focuses in this study. One is the final product of the evolution, that is, the network evolved to function as flip-flop. We compared the evolved networks with a manually constructed one presented in Chapter 3, Section 3.4. This comparison was possible because the identical coding scheme was chosen to represent boolean values with molecular species. Additionally, the evolutionary process itself was our focus, as pointed out by Bedau and Brown [1999], in order to distinguish relevant evolutionary activities. We used chemical organization theory [Dittrich and Speroni di Fenizio, 2007], described also in Chapter 1, for that purpose to trace the trajectory of evolving chemical reaction networks. Similarly, Matsumaru et al. [2006b] used that theory to study the evolutionary dynamics of artificial chemical systems, which is presented in the next chapter.

We concluded that evolution selects for an organizational structure that is related to function. That is, the resulting computation can be explained as a transition between organizations. Furthermore, an evolutionary process can be successfully tracked as a change of the organizational structure, which provides a fundamentally different view than looking at the structural changes of the

[1] the author is listed as the second author



B

$\bar{S}$	$\bar{R}$	Q_{t+1}	$\bar{Q}_{t+1}$	
0	*0*	*1*	*1*	—
0	*1*	*1*	*0*	set
1	*0*	*0*	*1*	reset
1	*1*	Q_t	$\bar{Q}_t$	hold

Figure 6.1: Circuit diagram and operation mode of flip-flop.

reaction networks. In our experiments, 90% of evolutionary improvements coincide with a change in the organizational structure.

This chapter consists of four sections: In the following section (Section 6.1), the experimental setting to evolve a reaction network is presented. As results in Section 6.2, three aspects of the evolutionary process are given. In addition to the traditional aspect of the dynamical behavior of the evolution, we analyse the dynamical change in terms of the chemical organization within the reaction networks. We also show a reaction network evolved for the flip-flop function in Section 6.3.1, and the organizational structure within the network is investigated. The flip-flop operation is described on the level of organizations and dynamically simulated. In Section 6.4, we discuss the evolved network in comparison with the hand-designed chemical flip-flop.

6.1 Method of Evolutionary Design

We employ an evolutionary algorithm that instantiates a natural selection process on chemical reaction networks [Fernando and Rowe, 2007]. The algorithm can mutate the reaction rules $\mathcal{R}$ of a reaction network with a fixed predefined set of molecular species $\mathcal{M}$. There are three mutation operators: to add a reaction , to delete a reaction, or to replace a reaction with a different one. To keep things simple, we employ a (1+1)-EA. That is, one parent generates one offspring, while the better of the two survives. If both have the same fitness, the offspring is kept so that neutral mutations and thus search space exploration is enabled. No parameter fitting is done, and each reaction is always associated with randomly chosen reaction parameters. A change in that parameters can only be realized through a replacement of a reaction with the same reaction, which has a different value. Only mass-action kinetics of first and second order are used in the evolution.

We employ the identical coding scheme to represent boolean values with molecular species as in [Matsumaru et al., 2007] and in Chapter 3, Section 3.4. In Figure 6.1, an (RS - Reset and Set) flip-flop is, once again, shown for the structure and the behavior in form of a truth table. To represent the four binary variables `a`, `b`, `c` and `d`, two opposing species x^0 and x^1 for each binary variable `x` are assigned. The presence of x^0 denotes the value `x = 0`, and x^1 denotes `x = 1`. To help maintain a valid state inside the system, we fix four destructive reactions $x^0 + x^1 \to \emptyset$ for all four species pairs $x^i = a^i, b^i, c^i, d^i$. These reactions cannot be changed or deleted by the evolutionary algorithm.

The ideal flip-flop that is the target of the artificial evolution works in the following way: The set operation $(\bar{S}, \bar{R}) = (0, 1)$ changes the state Q to 1, while the reset $(\bar{S}, \bar{Q}) = (1, 0)$ changes Q to 0. To hold the previous state, both inputs are set to 1. The forbidden input $(\bar{S}, \bar{Q}) = (0, 0)$ is not considered in the

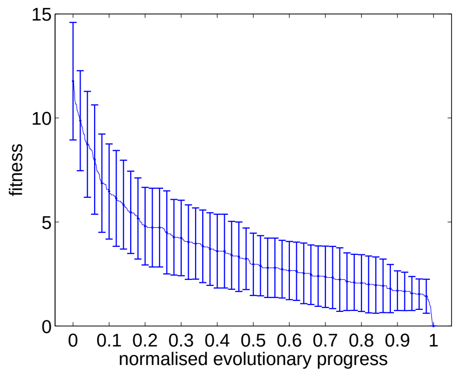


Figure 6.2: Average fitness value from beginning to end of evolutionary runs, from 30 independent repetions. The x-axis denotes the normalised evolutionary progress from the random initial solution ($x = 0$) to finding a solution with fitness 0 ($x = 1$). For this, the different runs were resampled to 1000 samples, as described in the text. Errorbars indicate standard deviation.

fitness evaluation. In chemical form, the input $(\bar{S}, \bar{R}) = (0, 1)$ is represented by defining an inflow for a^0 and b^1, that is, $\{\emptyset \to a^0, \emptyset \to b^1\} \subseteq \mathcal{R}$; and the other two cases are treated similarly. The initial concentrations of c^i and d^i are set according to the previous state Q_t. Taking this together, we get six different test cases, coming from three different operations with two initial conditions each.

For each case, we specify either the presence or the absence of each species as desired, measured in steady state after simulating the reaction system for 1000 seconds. Numerical integration is done using the SBML ODE Solver Library [Machné et al., 2006]. The classification as present or absent is decided by a concentration threshold of 10^{-9} (arbitrary units). For example, in the reset case, the following steady state concentrations are considered as correct: $a^1 = 1, a^0 = 0, b^1 = 0, b^0 = 1, c^1 = 0, c^0 = 1, d^1 = 1, d^0 = 0$ where 1 and 0 are the classification of presence or absence, respectively, not the concentration. The fitness value is then calculated by counting the number of wrong presence / absence measurements, with 0 being the best possible fitness value. Once a fitness of 0 is reached, the evolution stops.

6.2 Evolutionary Process

To analyse the evolution of reaction networks acting as flip-flops, we performed 30 indepent runs in order to evaluate properties of a "typical" run. A statistical analysis of those 30 runs is shown in this section.

The average fitness development (Figure 6.2) shows a stronger gain in fitness at the beginning, while the convergence towards zero is slower later on. Eventually, all runs reached a fitness of zero, i.e. the networks behaved as specified in the fitness function. Since a run stops exactly when the fitness of the current individual is 0, the number of generations usually differ between runs. In order to be able to average over these runs, we had to resample the

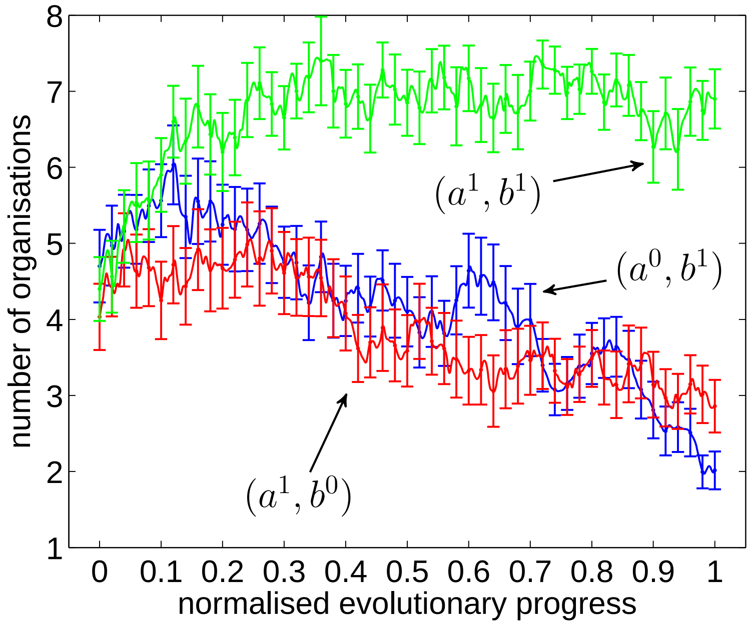


Figure 6.3: Average number of organizations from 30 independent runs of the evolution. Colors denote the (a^0, b^1) input (blue), the (a^1, b^0) input (red), and the (a^1, b^1) input (green). Errorbars indicate the standard error. Unit of x-axis as in figure 6.2.

data on fitness and number of organizations, such that a common number of measurements for each run is achieved. To this end, we constructed a timescale of "normalised evolutionary progress", defined by its endpoints 0.0 at the beginning of the evolution and 1.0 at the end when the final solution is found. The MATLAB function `resample`, which applies an anti-aliasing lowpass FIR filter during the resampling process, was used to create new data points at 1001 equally spaced points between 0.0 and 1.0.

Figure 6.3 shows the average number of organizations from 30 independent runs. The graph is a combination of three lines corresponding to three flip-flop operations. Since the three input settings are realized by enabling or disabling inflow reactions for input species a^1, a^0, b^1 and b^0, the reaction network structure for each operation is different, and so is the organizational structure. We need to compute three organizational structures for each candidate solution evaluated, one for each input settings. Looking at the number of organizations for the three different input cases, we can see that starting from around four to five organizations on average, the numbers diverge between the set/reset operations and the hold operation. While the number of organizations for the set/reset organization converges between two and three, the hold operation yields around seven organizations on average.

By comparing the organizational structures between successive candidate solutions, we calculated that 90% of all fitness improvements are accompanied by a change in the organizational structure for at least one input case. In contrast, only 18% of organizational changes also come with a fitness improvement.

When looking at the lineage of networks that led to the final solution, disregarding unsuccessful candidates, we find that 35% of all mutations changed the organizational structure for at least one input.

6.3 Analysis of Evolved Network

6.3.1 A Chemical Flip-Flop Evolved

An outcome of the evolutionary process described above is analyzed. The reaction network considered here has a fitness value of 0, *i.e.* it solves the given task. Comparing the fitness development of this evolution (not shown here) with the average of the 30 runs shown in Figure 6.2, we can conclude that this single run progressed in a fairly standard way. This is especially true given that the behavior of the 30 runs is quite diverse, as indicated by the large standard deviations. The length of this run (162 generations) is also in the usual region, with an average run taking 221 generations. Moreover, with respect to the number of organizations, the product analyzed here is in agreement with the average shown in Figure 6.3, even though the number of organizations for the set/reset operations are at the outer limits of the typical range (five and one, respectively). The product chosen here is a “typical” solution the evolutionary process finds.

The chemical flip-flop evolved by the evolutionary process is shown in Figure 6.4 as a reaction network. There are seven reactions, labeled as *rea1* to *rea7* in the figure, in addition to four reactions of cooperative decay (not shown in the figure), $a^1 + a^0 \rightarrow \emptyset$, $b^1 + b^0 \rightarrow \emptyset$, $c^1 + c^0 \rightarrow \emptyset$, $d^1 + d^0 \rightarrow \emptyset$. This base reaction network is extended to include inflow reactions, representing the inputs to the flip-flop circuit, depending on the operations. Organizational structures of the reaction system for each operational mode are shown in Figure 6.5.

Analysing the organizational structure of the reaction network, it becomes evident that the reaction system based on this reaction network is surely usable for the flip-flop computation. Including the two inflows $\emptyset \rightarrow a^1$ and $\emptyset \rightarrow b^0$ in the reaction network, as shown in Figure 6.5 A, only one set of species $\{a^1, b^0, c^0, d^1\}$ satisfies the conditions to be an organization. It implies that only this species combination can be found in the dynamical reaction system in equilibrium states. Therefore, the reset operation can be realized in the evolved reaction system. The network with the inflows of $\emptyset \rightarrow a^0$ and $\emptyset \rightarrow b^1$ contains five organizations as shown Figure 6.5 B, and one of those $\{a^0, b^1, c^1, d^0\}$ corresponds to the set operation.

Changing inflow reactions to $\emptyset \rightarrow a^1$ and $\emptyset \rightarrow b^1$ achieves the hold operation. In terms of the organizations, as shown in Figure 6.5 C, the two organizations *orgHR*= $\{a^1, b^1, c^0, d^1\}$ and *orgHS*= $\{a^1, b^1, c^1, d^0\}$ in the reaction network with those inflows reflect the bistability of the flip-flop circuit. Depending on the state at the previous time step, the hold operation results in a different state, namely the previous one. When the reaction system has been in the state after the set operation, (*i.e.*, *orgS*), the hold operation brings the system to the state of *orgHS*, keeping the output species unchanged as c^1 and d^0. Holding the information that the system has been reseted can be achieved by moving the system state from *orgR* to *orgHR*.

The last operation of setting both inputs to be zero ($\texttt{a} = \texttt{b} = 0$) is forbidden for the flip-flop circuit. If adding two inflows of $\emptyset \rightarrow a^0$ and $\emptyset \rightarrow b^0$

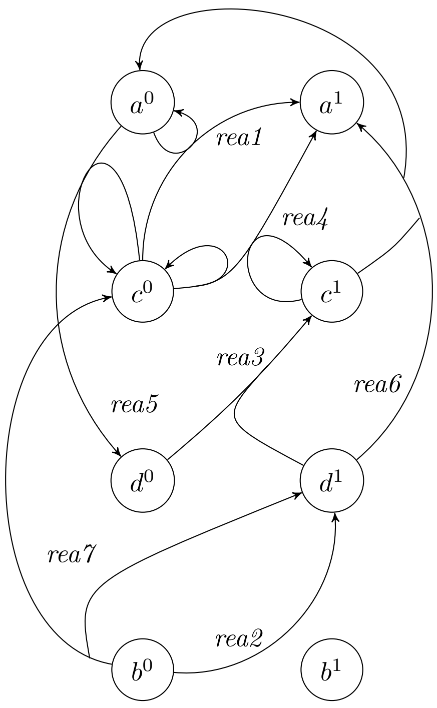


Figure 6.4: Chemical reaction network implementing flip-flop circuits, designed through an evolutionary process. Cooperative decay reactions ($a^1 + a^0 \to \emptyset$, $b^1 + b^0 \to \emptyset$, $c^1 + c^0 \to \emptyset$, $d^1 + d^0 \to \emptyset$) are omitted.

to the base reaction network, one set of species becomes the organization: $\{a^1, a^0, b^0, c^1, c^0, d^1, d^0\}$. Only b^1 is not involved to form the organizational structure.

If no inflow reaction is presented, there are 42 organizations in the base reaction network. The smallest organization is the empty set $\emptyset$. The sets containing four species forms the largest organizations, and there are four organizations of that size. The organizations with the size of four in Figure 6.5 are also found to be the organization without inflows, except the organization labeled as *orgR*. In fact, all organizations in Figure 6.5 except *orgR* are also organizations without inflows.

6.3.2 Dynamical Behavior

To validate the organizational analysis of the reaction network, a dynamical reaction system is constructed and simulated with *Copasi* [Sahle et al., 2006], a biochemical reaction system simulator. Agreeing to the fitness calculation of the evolutionary design process, mass action kinetics is assumed for every reaction, if applicable. The ordinary differential equations (ODEs) for the input

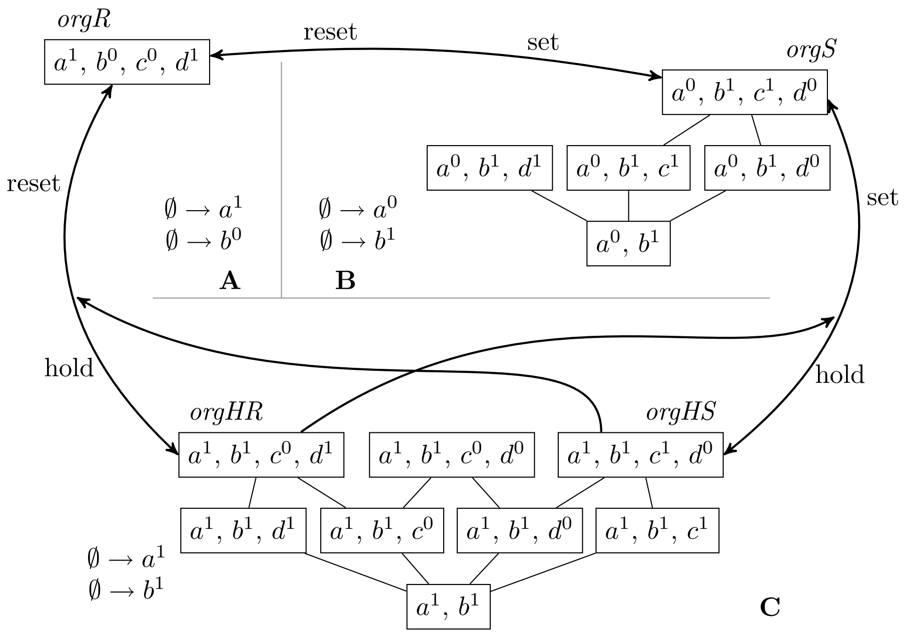


Figure 6.5: Organizational structure in the reaction network shown in Figure 6.4.

species read:

$$\begin{aligned}
\dot{[a^1]} &= k_1[a^0][c^0] \\
&\quad +k_4[c^1][c^0] + k_6[c^1][d^1] - d_a[a^1][a^0] + I_{a^1}(1-[a^1]) && (6.1)\\
\dot{[a^0]} &= -k_5[a^0][c^0] + k_6[c^1][d^1] - d_a[a^1][a^0] + I_{a^0}(1-[a^0]) && (6.2)\\
\dot{[b^1]} &= -d_b[b^1][b^0] + I_{b^1}(1-[b^1]) && (6.3)\\
\dot{[b^0]} &= -k_2[b^0] - k_7[b^0] - d_b[b^1][b^0] + I_{b^0}(1-[b^0]) && (6.4)
\end{aligned}$$

where a kinetic parameter for a reaction *rea_id* is denoted as k_{rea_id}. Kinetic parameters for the cooperative decay reactions are represented by d, and the subscript specifies the pair. For example, the decay rate of the cooperative decay reaction $a^0 + a^1 \to \emptyset$ is denoted as d_a.

Inflow reactions representing the operation of reset, set, and hold are controlled by the four parameters: I_{a^1}, I_{a^0}, I_{b^1}, and I_{b^0}. These parameters are binary variables, accepting only 0 or 1. For example, when the chemical flip-flop is set, I_{a^0} and I_{b^1} are set to one and the other pair of parameters $I_{a^1} = I_{b^0} = 0$ is set to zero. Inflows are assumed to be constant fluxes. Furthermore, the inflows are linked to normal decay reactions such as $a^1 \to \emptyset$ in order to avoid endless increase of the input species concentration. The resulting term of the ODE is $I_{a^1}(1 - [a^1])$, for example.

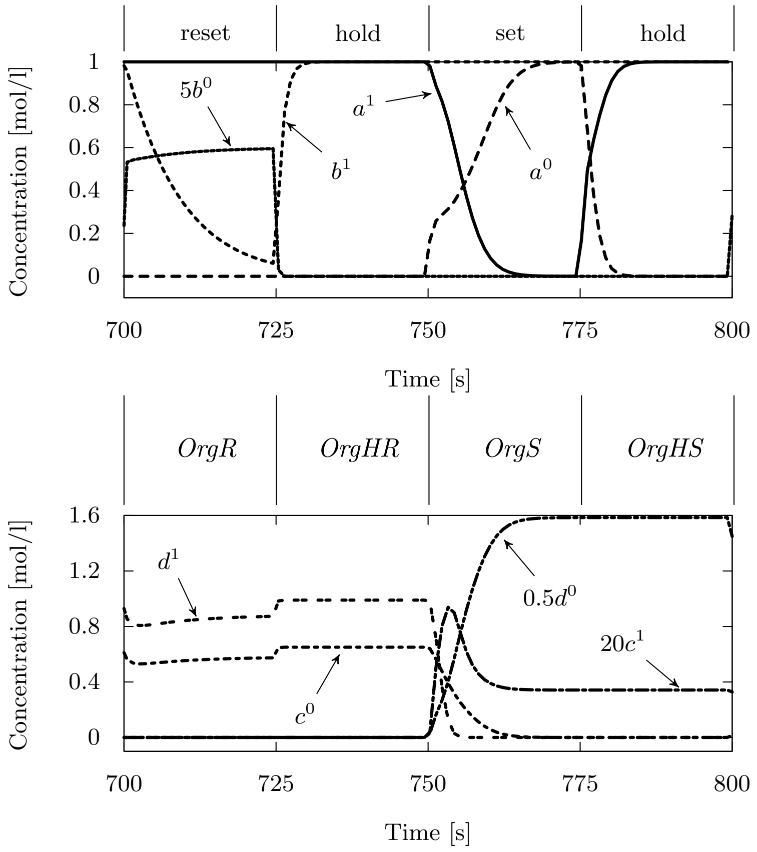


Figure 6.6: Dynamical simulation of chemical flip-flop designed by evolution. Parameters are set as follows: $d_a = d_b = 0.1$, $k_4 = 2.33941$, $k_6 = 2.83745$, $k_1 = 4.44231$, $k_5 = 3.62963$, $k_7 = 4.82838$, $k_3 = 1.0$, $k_2 = 0.1$, $d_c = 0.001$, $d_d = 1.0$. Additionally, for each operation of reset, set, and hold, inflow reactions are activated. For the set operation, the parameters are set such that $I_{a^0} = I_{b^1} = 0$ and $I_{a^1} = I_{b^0} = 1$ to activate inflows of a^1 and b^0 species and to deactivate the others. The reset operation is initiated by setting $I_{a^0} = I_{b^1} = 1$ and $I_{a^1} = I_{b^0} = 0$. The hold operation is achieved with the parameter settings of $I_{a^1} = I_{b^1} = 1$ and $I_{a^0} = I_{b^0} = 0$.

The ODEs for the output species read:

$$\dot{[c^1]} = k_3[d^1][d^0] - k_6[d^1][c^1] - d_c[c^1][c^0] - I_{b^0}[c^1] \quad (6.5)$$

$$\dot{[c^0]} = -k_1[a^0][c^0] + k_7[b^0] - d_c[c^1][c^0] - I_{b^0}[c^0] \quad (6.6)$$

$$\dot{[d^1]} = k_2[b^0] -k_3[d^1][d^0] - k_6[d^1][c^1] + k_7[b^0] - d_d[d^1][d^0] - I_{b^0}[d^1] \quad (6.7)$$

$$\dot{[d^0]} = -k_3[d^1][d^0] + k_5[a^0][c^0] - d_d[d^1][d^0] - I_{b^0}[d^0] \quad (6.8)$$

Kinetic parameter values are also provided as the outcome of the evolutionary design, but we manually adjusted the values so that the operations can be continuously repeated. When the fitness of the reaction system was calculated during the evolution process, three of the operations were evaluated separately and the reaction system was reinitialized for each case. This re-initialization step between operations is prevented so that the end state of the previous operation becomes the initial state of the next operation. For that purpose, the outflows of the input species are added as described above in order to restrict the increase of the concentration. For the output species, the outflows are also added as shown above, activated only when the inflow of b^0 is present. This modification is also to restrict the increase of the concentrations of the output species, specially, when the system is reseted.

The last modification is the kinetic parameter of the reaction *rea1*, k_1, from 4.44231 to 0.5. The rational of this adjustment is: under the input condition "set", the system is observed to converge to the organization of $\{a^0, b^1, d^1\}$, instead of *orgS*. This behavior is results from the fast extinction of c^0 species so that the generation of d^0 by *rea5* is insufficient. Slowing down the reaction speed of *rea1*, species c^0 stays in the system longer and produces d^0 enough to neutralize d^1.

6.4 Discussion

We found that most fitness improvements come together with change in organizational structure (90%), showing that organization analysis indeed yields insight into the evolutionary process. On the other hand, most organizational changes are fitness-neutral (82%), indicating that a lot of the information given in the set of organizations does not directly relate to the measured function of the networks. We have also observed a fitness improvement caused purely by the change of a kinetic parameter, as well as by changes of network structure not reflected in the organizations.

Another observation is that the number of organizations for the set and reset operations is substantially smaller than that for the hold operation, in analogy to the hand-constructed flip-flop by Matsumaru et al. [2007] and Chapter 3, Section 3.4. In comparison to that solution, the evolved networks show a larger number of organizations for each input case. To realize the flip-flop behavior in the reaction system, the minimum number of organizations in the reaction network is one for the set and reset operation and three for the hold operation. The hand-designed flip-flop implementation shown in Section 3.4 has two organizations for set and reset and three for hold. The evolved networks, on the other hand, have more organizations on average between two and three each for set and reset, and seven for hold. This implies that even though the function of the flip-flop networks is reflected in their organizational structure, this structure contains more information than only the operational modes specified in the fitness function.

Evolving organizations For this work, the notion of chemical organizations is used only passively such that the evolutionary processes are analyzed with it. As an interesting extension of this work, one could use organizational analysis actively in order to direct the evolution of reaction networks. By first designing the perfect organizational structure and then evolving networks with this structure, it would be possible to study whether these network have the desired functionality. This idea can be rationalized by the findings in this work such that the change of organizational structures contributed largely to the improvements of the fitness values. In addition, the original fitness evaluation involves a step of random assignments of kinetic parameters, and these parameter values are influential to the behaviors of reaction systems. It is possible that proper reaction networks are discarded because of the wrong choice of kinetic parameters. If including the organizational analysis in the fitness evaluation and giving an extra reward for the properness of the network structure, convergence of the fitness value will be accelerated. To make it even more rapid, the notion of organizations can be utilized to control mutation processes so that the evolution is guided to eliminate unnecessary organizations. Further investigation regarding the effect of different structural mutation operators on the organizational structures is beneficial.

Source :

Matsumaru, N., Speroni di Fenizio, P., Centler, F., and Dittrich, P. (2006b). On the evolution of chemical organizations. In Artmann, S. and Dittrich, P., editors, *Explorations in the complexity of possible life: abstracting and synthesizing the principles of living systems, Proceedings of the 7th German Workshop of Artificial Life*, pages 135–146. Aka, Berlin.

Chapter 7

Tracking Chemical Evolution

Contents

As a conclusion of the previous chapter, transformation of organizational structures can be an important step in the evolution of a chemical computing system to improve the fitness [1]. In this chapter, we further elaborate how an evolution of chemical systems can be tracked. A method is to distinguish the organizational evolution in the set of organizations from the actual evolution. There are three types of movements in the level of organizational evolutions [Dittrich and Speroni di Fenizio, 2007], and those movements are illustrated as the method using a constructive artificial chemistry. It is shown that the two levels of evolutions interact with each other in a non-trivial manner. Employing also the representation of the organizational evolution to study chemical evolution is useful.

Two Evolutionary Processes Before proceeding further, we would like to emphasize the essential differences of two evolutionary processes presented in this chapter and the previous chapter. In the previous chapter, the target system of the evolutionary process is the reaction network. The network structure was given explicitly in the global domain external to the reaction systems, and it is the network structure that is mutated directly. In this case, the application of chemical organization theory was straightforward. When observing dynamical evolutionary processes of reaction systems in general, on the other hand, molecular concentration profiles of the systems are obtained but the reaction network structure is not immediately available. Analysis of evolutionary

[1]experimented and analyzed chiefly by Thorsten Lenser and presented in Chapter 6

processes targets the changes of concentration profiles in reaction systems. To simulate such a case, the molecules for the experiment in this chapter are defined to have a structure, and that determines the reaction rules. The reaction rules are, so to speak, implicit because they are encoded into the structure of the molecules. The extinction of one molecular species from the reaction vessel results in the removal of a reaction rule. The appearance of new species adds a rule to the reaction network. For the analysis of such systems using chemical organization theory, reconstruction of the reaction network is obligatory.

7.1 Two Levels of Chemical Evolution

In the theory of chemical organization, the dynamical change of the concentration profile in state space X can be mapped to the set of organizations L. This mapping arises two-level representation of the dynamical (evolutionary) behavior. We distinguish the movement in L as *organizational evolution* from *actual evolution* in X. Figure 7.1 schematically describes, including the two-level evolutions, the analysis of a reaction system using chemical organization theory. Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$, the organizational structure embedded within the network structure is derived (Figure 7.1, Left) in order to abstract behavioral modes of dynamical reaction systems. To realize a movement in the state space X, dynamics of the reaction systems are necessary to be given additionally. Regarding two states $\mathbf{x}_{t_1}, \mathbf{x}_{t_2} \in X$ at time points t_1 and t_2 ($t_1 < t_2$), the organizations $O_{t_1}, O_{t_2} \in L$ can be generated as follows: $O_{t_1} = G_{org}(\phi(\mathbf{x}_{t_1}))$, $O_{t_2} = G_{org}(\phi(\mathbf{x}_{t_2}))$. Applying the abstraction ϕ mapping, we can obtain the set of species $S_t = \phi(\mathbf{x}_t)$ at time t. The organizational evolution denotes the transition between two organizations, and the movement can be categorized into three directions [Dittrich and Speroni di Fenizio, 2007]: upward, downward, and sideward (Figure 7.1, Right). In case $O_{t_1} \supset O_{t_2}$, the movement in the state space is classified as a downward movement. The other way of inclusion, namely $O_{t_1} \subset O_{t_2}$, is an upward movement. The dynamical change ($O_{t_1} \neq O_{t_2}$) that is neither downwards nor upwards is called a sideward movement. The equality, $O_{t_1} = O_{t_2}$, is excluded since no movement is detected on the level of organizations.

Evolution and Organization Evolution has been recognized as a main process of developing natural organisms with complex behaviors from a simple form. Associating organizations in reaction networks with behavioral modes of the reaction systems, the high complexity of organisms' behaviors might be correlated to high complexities of the organizational structure. Extending this argument, the upward movement and the sideward movement becomes particularly significant in the field of evolution. Those two transitions effect the new sub-organization in the reaction network.

7.2 Experimental Setup

In this section, we demonstrate how the chemical organization theory gives an insight to chemical evolution. An artificial chemistry system called automata chemistry [Dittrich and Banzhaf, 1998] is used to generate chemical evolution.

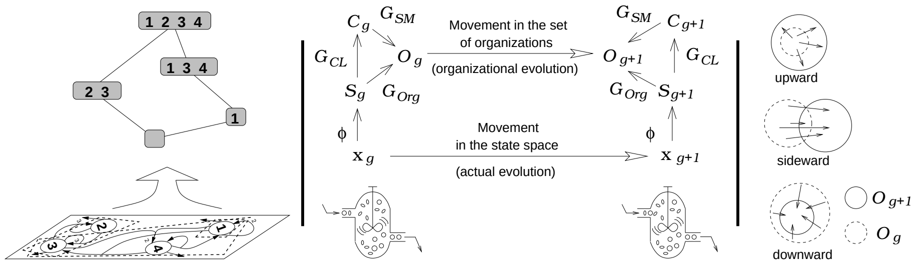


Figure 7.1: Illustration of the two levels of chemical evolution. **Left:** Based on the static network structure, the reaction network is decomposed into overlapping sub-networks called organizations. The hierarchical organizational structure of the network is visualized by a Hasse-diagram. **Middle:** To analyze the system's dynamics, a movement from state $\mathbf{x}_g$ to a state $\mathbf{x}_{g+1}$ in state space is mapped to a movement from organization O_g to organization O_{g+1}. The actual evolution of the state $\mathbf{x}$ from g to $g+1$ does not necessarily lead to a change of the organizations, that is why we distinguish the *actual evolution* of the set of molecules actually present in the reactor from the *organizational evolution* of the organizations reachable from that molecules. **Right:** The organizational evolution is categorized into three movements: upward, downward, and sideward. See text for detail.

7.2.1 Automata Chemistry

Molecular species are binary strings $s \in \{0,1\}^{32}$ with a constant length of 32 bit. Two strings can catalyze the production of a third string: $s_1 + s_2 \Rightarrow s_3$. One of the strings s_1 is mapped to an automaton A_{s_1} according to a well defined instruction table (we used code table II in [Dittrich and Banzhaf, 1998] allowing self-replication). The other s_2 serves as the input to A_{s_1}. The result of executing the program on the input string is the product $s_3 = A_{s_1}(s_2)$.

The dynamical system consists of the following three components:

i *A soup (population) of objects*
The soup contains a collection of objects, and these objects may be character sequences, lambda-expressions, or numbers. Here, we use as an object binary strings $s \in \{0,1\}^{32}$ with a constant length of 32 bit.

ii *A collision or reaction rule*
A collision rule defined the interaction among two objects which may lead to the generation of new objects. Here, *automata reaction* with code table II [Dittrich and Banzhaf, 1998] is adopted. To instantiate a deterministic reaction $s_1 + s_2 \rightarrow s_3$ where $s_1, s_2, s_3 \in \{0,1\}^{32}$, binary string s_1 is "folded" into the instruction for an automaton A_{s_1} and string s_2 is read by the automaton as the input. The instruction is a sequence of register commands, and each command is represented by four bits. Hence, the 32 bit string is converted into the sequence of eight commands. The product string s_3 is the result of the manipulation on the input string by the instruction. The detail explanation can be found in [Dittrich and Banzhaf, 1998].

iii *Dynamics specification*
This is the component to specify how the objects locates/moves in the

soup and how the reaction rule is applied. Here, the soup is well-stirred so that every object can interact with every other objects. The application of the reaction is defined as a stochastic process. The pseudo code specifying the dynamics is listed in Table 7.1

Preparing a reactor (or reaction vessel) containing N string objects, multiple copies of the species are placed in the reactor to simulate the dynamical behavior of the reaction system. In each time step, two string objects are randomly chosen to react, and the reactants are inserted back into the reactor without deleting the two reactands. One randomly chosen molecule in the reactor is replaced by the product in order to keep the total number of the objects in the reactor constant. In short, the system is a catalytic flow system in a well stirred reactor. In one *generation*, N steps are executed.

7.2.2 Reaction Dynamics: Pseudo Code

The algorithmic reaction dynamics is presented here as pseudo code. In line 1–4, the parameters to determine simulation settings are set. Reactor (or reaction vessel) R of size N is initialized in line 5 such that there exists N string objects in the reactor R. The reactor is filled with N copies of a string to realize a homogeneous state. A heterogeneous state can be achieved with one copy of N random strings. Within a generation repeated with the **for**-loop from line 7, there are two kinds of processes: reaction and mutation. The reaction process is executed in line 9–16, which specifies how reaction rules are applied and affect states of the reactor. Two string s_1 and s_2 are chosen randomly from the reactor (line 11 and 12) as an operator and an operand, respectively. The operator string of 32 bits is converted into a program consisting of eight register commands. The program gets the operand as the input and manipulates the input string. The output of the program is the result of the $+$ operation in the automata chemistry (line 13). The reactor state is altered by replacing a string object with the reaction product so that the total quantity of the objects in the reactor is unchanged. There are N reactions in a generation, repeated by the **for**-loop in line 9. As a result, it is probable that every string object in the reactor is updated via reactions in one generation.

The mutation process is within the **for**-loop of line 19–25 when the **if**-condition in line 18 is satisfied. Generations between two mutation events are counted with variable $t_{interval}$. A randomly chosen string object s_m is mutated to s'_m by negating the k-th bit of the original string.

7.2.3 Analysis Method

Theoretically speaking, the automata chemistry consists of $2^{32} = |\mathcal{M}|$ binary strings as molecular species and reactions among them, forming the reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$. Since it is impractical to consider the entire network, however, only the small part related to the reactor state $\mathbf{x}_g \in X$ at generation time g is considered as $\langle \mathcal{M}_g, \mathcal{R}_g \rangle$ where $\mathcal{M}_g = G_{CL}(\phi(\mathbf{x}_g))$ is the set of molecules that can be generated from $\mathbf{x}_g$.

Representing the reactor state as a multiset: $\mathbf{x}_g = \{m_1, m_2, \dots, m_N\}$ where N is the size of the reactor, the abstraction ϕ of the reactor state is the set S_g of molecular species present in the reactor, ignoring the multiplicity: $S_g = \phi(\mathbf{x}_g) = \{s \in \mathbf{x}_g | \#(s \in \mathbf{x}_g) > 0\}$ where $\#(s \in \mathbf{x}_g)$ denotes the number of

Table 7.1: Listing to define dynamics of automata chemistry

```
Algorithm: Automata chemistry

$g_{max}$      ← maximum_generation #
$t_{mut}$      ← mutation_interval   # interval between mutations
$n_{mut}$      ← mutation_bit        # number of bits to be mutated
$N$            ← reactor_size        #        (0 for no mutation)
$M$            ← InitReactor($N$)    # initialization
$t_{interval}$ ← $t_{mut}$           # init.  mutation interval counter
for $g \leftarrow 0$ to $g_{max}$ do
    $g$        ← $g + 1$                # start new generation
    for $i \leftarrow 0$ to $N$ do
        $i$    ← $i + 1$                # $N$ reactions in one gen.
        $s_1$  ← chooseRandom($M$)      # choose operator
        $s_2$  ← chooseRandom($M$)      # choose operand
        $p$    ← $s_1 + s_2$            # react
        $s_3$  ← chooseRandom($M$)      # an obj.  to be replaced
        $M$    ← $(M \setminus \{s_3\}) \cup \{p\}$   # new reactor state
    end
    $t_{interval} \leftarrow t_{interval} - 1$       # count down till next mutation
    if $t_{interval} == 0$ then
        for $j \leftarrow 0$ to $n_{mut}$ do
            $j$     ← $j + 1$                # $n_{mut}$ obj.  to be mutated
            $s_m$   ← chooseRandom($M$)      # obj.  to be mutated
            $k$     ← random[0,31]           # $k$-th bit to be mutated
            $s'_m$  ← negate($s_m$,$k$)      # negate $k$-th bit of $s_m$
            $M$     ← $(M \setminus \{s_m\}) \cup \{s'_m\}$   # new reactor state
        end
        $t_{interval} \leftarrow t_{mut}$            # re-init.  mutation counter
    end
end
```

Table 7.2: Listing of function to generate closed set.

```
Function : Generate closure $G_{CL}$
Input: Set of species ($S$)
Output: Closed set of species ($CL$)

begin
    $CL \leftarrow S$
    $A \leftarrow \emptyset$
    while $A \neq \emptyset$ do
        $A \leftarrow \emptyset$
        foreach $(s_i, s_j) : s_i, s_j \in CL$ do
            $p \leftarrow s_i + s_j$
            if $p \notin CL$ then $A \leftarrow A \cup \{p\}$
        end
        $CL \leftarrow CL \cup A$
    end
end
```

Table 7.3: Listing of function to generate self-maintaining set.

```
Function : Generate self-maintaining set $G_{SM}$
  Input: Set of species ($S$)
  Output: Self-maintaining set of species ($SM$)

  begin
      $SM \leftarrow S$
      $B \leftarrow SM$
      while $SM \neq B$ do
          $B \leftarrow \emptyset$
          foreach $(s_i, s_j) : s_i, s_j \in SM$ do
              $p \leftarrow s_i + s_j$
              $B \leftarrow B \cup \{p\}$
          end
          $SM \leftarrow SM \cap B$
      end
  end
```

occurrences of element s in multiset $\mathbf{x}_g$. Taking the closure of the set of species: $C_g = G_{CL}(S_g)$ listed as the pseudo code in Table 7.2, the reaction network is constructed by setting $\mathcal{M}_g = C_g$. The set of reaction rules $\mathcal{R}_g = (\mathcal{C}_g \cup \mathcal{D}_g)$ is composed of two kinds of reactions: catalytic reactions $\mathcal{C}_g$ and decay reactions $\mathcal{D}_g$. Decay reactions $\mathcal{D}_g = \{(m \to \emptyset) | m \in \mathcal{M}_g\}$ are included since every object is subject to be replaced by a reaction product. In this dynamical simulation, it is not possible for the reaction vessel to be empty even though every object species is defined to decay. Since two objects initiating a reaction are not altered by the reaction, the process is defined as a catalytic reaction $\mathcal{C}_g = \{(m_i + m_j \to m_i + m_j + m_k) | m_i, m_j \in \mathcal{M}_g, m_i + m_j \Rightarrow m_k\}$ where the rule $m_i + m_j \Rightarrow m_k$ is defined in the automata chemistry. Note that $m_k \in \mathcal{M}_g$ because of the closure property of the reaction network. As a result, there are $|\mathcal{R}_g| = |\mathcal{C}_g| + |\mathcal{D}_g| = |\mathcal{M}_g|^2 + |\mathcal{M}_g|$ reaction rules.

Considering the characteristics of the reaction network, the function G_{SM} to generate the self-maintaining set can be defined as listed in Table 7.3. The reaction network is designed so that every molecular species decays but the reactants of all catalytic reactions are conserved. Only the decay reactions contribute to the negative production rate of species. To be self-maintaining for the set of species, those negative production rates must be compensated. Therefore, the set is self-maintaining if all of the elements are produced by the catalytic reactions. In order to generate the biggest self-maintaining set contained in the original, species not produced by the reactions are excluded from the set until every species is produced.

Given the reaction network $\langle \mathcal{M}_g, \mathcal{R}_g \rangle$, we compute all organizations to extract the hierarchical organizational structure in the reaction network. The set of all organizations is denoted as L_g. It forms together with the union $\sqcup$ and intersection $\sqcap$ of organizations an algebraic structure $\langle L_g, \sqcup, \sqcap \rangle$ called a lattice[2]. The biggest organization $O_g \in L_g$ is generated from the whole set of the species present in the reaction vessel: $O_g = G_{org}(S_g)$ where $G_{org} \equiv$

[2]Since the algebraic chemistry is designed as a reactive flow system, the set of all or-

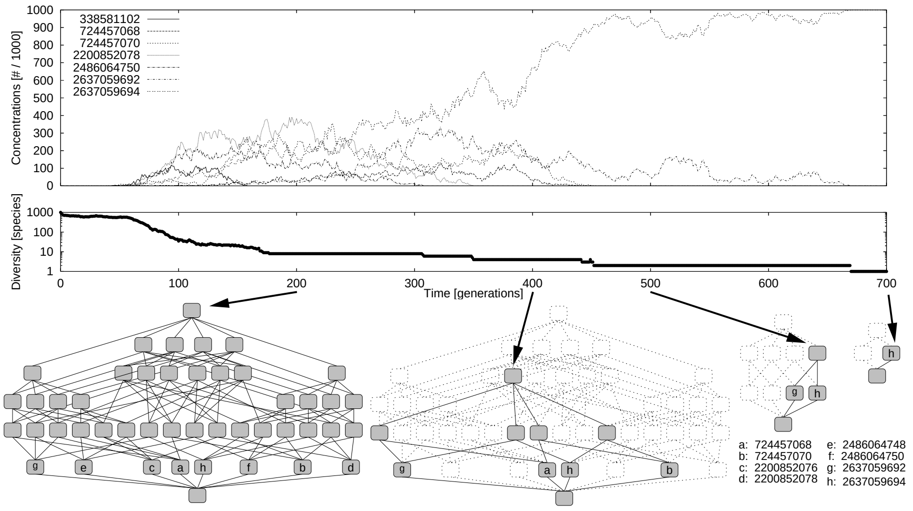


Figure 7.2: Dynamical behavior of automata chemistry showing several downward movements. The reactor of size $N = 1000$ is filled initially with one copy each of N species of random binary string with fixed length 32 bits. Top: concentration profile of the reactor with respect to some prominent species. Middle: diversity as the number of different species present in the reactor. Bottom: lattice of organizations at generation 200, 400, 500, and 700. Dotted boxes and lines are the organizations and links missing compared with the previous lattice structure.

$G_{SM} \circ G_{CL}$. The other sub-organizations are generated from any subset of the set: $O'_g = G_{org}(S'_g \subset S_g)$

7.3 Results

In order to study downward movements, we simulate our artificial chemistry without any external perturbations, that is, mutations, at first. Then, in Section 7.3.2, we will demonstrate upward and sideward movements by introducing moderate mutations, which cause constructive perturbations.

7.3.1 Dynamical behavior as downward movement

The reactor of size $N = 1000$ is heterogeneously initialized with N random objects. Figure 7.2 shows the typical dynamical behavior of the simulated chemical evolution in three forms. The concentration change of some species (with relatively high quantity) is plotted to view the dynamical change of the reactor state. As in the middle graph, the number of molecular species present in the reactor is plotted as diversity. A tendency to decrease diversity may describe this evolutionary behavior. This dynamical behavior is analyzed with the theory of chemical organization, and the results are given as a series of

ganizations in such a system is proved by Dittrich and Speroni di Fenizio [2007] to form a lattice.

Hasse-diagrams, visualizing the lattice of organizations L_{200}, L_{400}, L_{500}, and L_{700} (Figure 7.2, bottom). The labels in the box indicate species that are new in the corresponding organization and are not contained in any of the organizations below it. Since the organizational structure depends on the qualitative state of the reaction vessel, the result is not affected as long as the diversity stays the same.

At $g = 200$, there are eight species in the reactor, and those species form the biggest organizations: $O_{200} = G_{org}(S_{200}) = S_{200}$. Forty-six organizations are found as shown in the leftmost Hasse-diagram. In the next 200 generations, four species are drained so that the diversity value of the reactor becomes four. The lattice of organizations L_{400} consists of ten organizations including the biggest organizations formed by the remaining four species. Comparing two sets of the organizations L_{200} and L_{400}, we found that $L_{200} \subset L_{400}$ so that it is possible to impose the lattice L_{400} on L_{200} as shown in the figure. The solid lines represent the lattice at $g = 400$, and the organizations vanished during the 200 generations are drawn by the dotted lines

This dynamical change of the reactor state can be explained as a downward movement. The sets of species present in the reactor at generation $g = 200$ and 400 are the organizations: $O_{200} = G_{org}(S_{200}) = S_{200}$ and $O_{400} = G_{org}(S_{400}) = S_{400}$. The inclusion $O_{400} \subset O_{200}$ is true since species present in the reactor only disappear and no new species appears within that 200 generations. Similarly, this argument is applicable between S_{400} and S_{500} and between S_{500} and S_{700}. In this simulation settings, only the reactions can produce possibly new species, but applying the chemical reactions to the set of existing species cannot disrupt the closure property of the organization. Only the downward movement is thus feasible.

7.3.2 Upward and sideward movement

To demonstrate upward and sideward movement, a mutation process is introduced. Every 100 generation, ten objects are chosen randomly, and each binary string object is mutated by inverting one randomly chosen bit. The reactor is initialized homogeneously with $N = 1000$ copies of a certain species, so the diversity value is 1 in the beginning. Figure 7.3 (top) shows the dynamical behavior of the concentration profile with respect to the prominent species, and the number of the species existing in the reactor is plotted as diversity. The rapid increase of the diversity every 100 generation is caused by the ten new mutants. The organizational structure in the reaction network is computed every 10 generation. At the moment of the mutation event, the network is analyzed just before the mutation, and the effect of the mutation is observed only after ten generations. At the bottom, the dynamical change of the lattice structure from $g = 360$ is depicted. The organizations and links are drawn by bold lines if inherited from the previous structure, and the dotted lines are used if vanished.

Starting with two organizations (empty set and set of two species), the mutation at $g = 400$ introduces new species to the reaction system and the reaction network is expanded. After ten generations, the reaction system settles to the state with four species. The reaction network with the four species is composed of six organizations, and the biggest organization is the set of those four species. This lattice structure is sustained in the next 200 generations

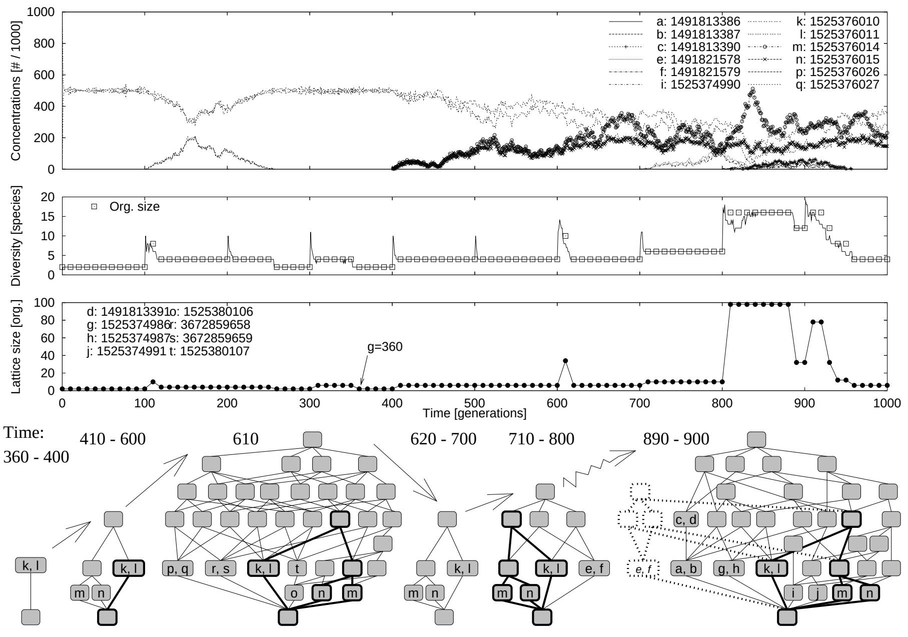


Figure 7.3: Dynamical behavior of automata chemistry exhibiting upward and sideward movement. The reactor of size $N = 1000$ is initialized heterogeneously. Every 100 generations, 10 string objects are chosen to be mutated by an one-bit negation. Top: dynamical change of concentration profile of the reactor with respect to prominent species. Second top: the number of unique species in the reactor as diversity and the size of the biggest organization generated, calculated every 10 generations. Third top: the number of the organizations in the reaction network. Bottom: Hasse-diagrams depicting organizational structure in the reaction network. Starting from $g = 360$, two upward movements are achieved until $g = 610$ although the lattice immediately shrinks (downward movement). From 800 to 900, a sideward movement is observed.

including one mutation at $g = 500$. Temporarily, the next mutation at $g = 600$ brings up the system to the organization of ten species and thirty-four organizations in the reaction network as observed at $g = 610$. After 20 generations, all of the new organizations are vanished, and the lattice structure comes back to that prior to the mutation at $g = 600$.

These are typical upward and downward movements. The mutation process produces new species outside of the closure and causes upward movement. The dilution flow removes species from the reaction vessel randomly, and the system goes to the organization below. Since the concentration of the new species is very low, the new organizations brought about by the upward movement has a disadvantage statistically. Thus, the upward movement is canceled mostly.

The sideward movement is observed between O_{800} and O_{900}. The mutation process at $g = 800$ introduces new species to the reactor system and pushes the system into the bigger organization consisting of sixteen species. Ninety-eight organizations are found in the reaction network. Each of the sixteen species is maintained for a relatively long period (90 generations), but four of the species are eventually depleted. In consequence, lattice structure L_{900} has 32 organizations. As illustrated in Figure 7.3 bottom, four organizations associated with species e and f are missing in L_{900} in comparison with L_{800}.

7.3.3 Diversity and Organization

In the previous examples, the generated organization from the reactor state is mostly the same as the abstraction (i.e., $G_{org}(\phi(\mathbf{x})) = \phi(\mathbf{x})$). In other words, the set of species present in the reactor is an organization. In that case the diversity (number of different species present) seems an adequate representation of the evolutionary behavior. However, the benefit to apply the generate function becomes evident in Figure 7.4, where we can see that a decrease in diversity does not necessarily imply a decrease of the generated organization.

For this simulation the reactor of size $N = 1000$ is initialized with sixteen species, which form a reaction network holding 146 organizations as shown in Figure 7.5 (left). Mutation is disabled so that the only downward movement can occur. The reactor is in the biggest organization at the top of the lattice structure, and there are four organizations directly below as shown in Figure 7.5 (right). The organizational structure is sustained for a long time ≈ 800 generations until a series of species destruction causes the reaction system to move downwards. Around generation 200, the diversity is reduced due to the disappearance of the species from the reactor, and the reduction is dynamically compensated by regenerating the disappeared species. When the set of species present in the reactor is not an organization anymore, violating the closure property in this case, the dynamical reaction system tends to move to the state that is the instance of an organization. The organizational set is a candidate of the steady state and the other species combinations are not stable [Dittrich and Speroni di Fenizio, 2007].

The organization generated from the reduced set of species is, however, unchanged during that period. Applying the generate function takes the structure of the reaction network into consideration. By representing the dynamical behavior on the level of the organization, the dynamical change of the underlying reaction network is focused. Furthermore, temporal stochastic effects can be separated from the permanent effects, causing downward movements.

7.4 Discussion and Conclusion

In this chapter, we have demonstrated that chemical organization theory can provide another level of explanation to understand chemical evolution. With the help of the theory, we can consider two levels of chemical evolution: (1) the actual evolution of the reaction vessel, that is, the arrival and disappearance of chemical species; and (2) the "organizational evolution", that is, the change of the organization generated by the current set of molecules present in the vessel (Figure 7.1, middle). Our results suggest that actual evolution of the reaction vessel does not trivially imply its organizational evolution and vice versa.

We have characterized, as usual in evolution theory, the actual evolution of the reactor by the change of its diversity, which reflects the arrival and disappearance of chemical species. The evolution on the organizational level was characterized as downward, upward, or sideward movement in the organization

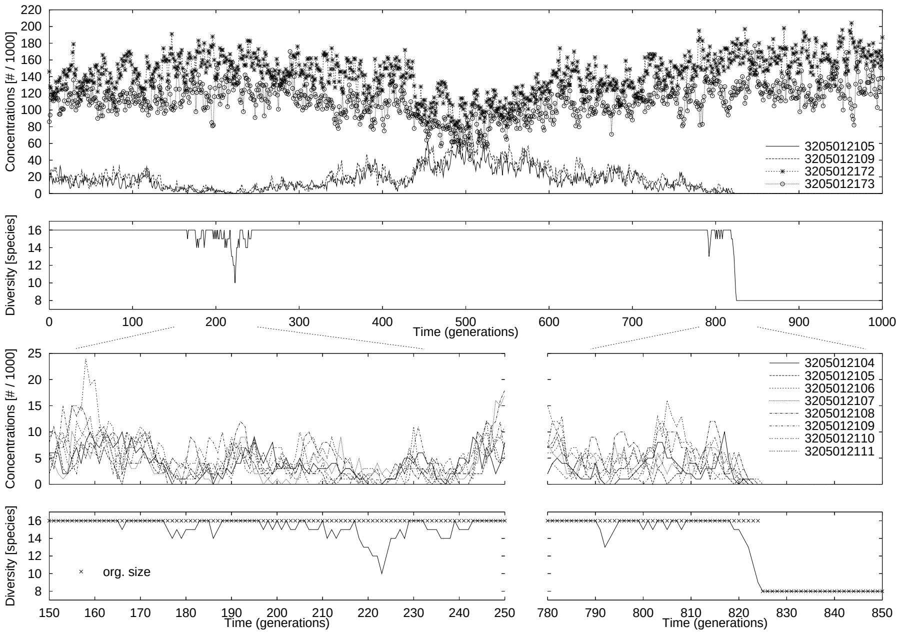


Figure 7.4: Dynamical behavior of automata chemistry exhibiting long-term preservation of an organization and then downward movement. The reactor of size $N = 1000$ is initialized with sixteen species, and the organizational structure in the reaction network among those species consists of 146 organizations as shown in Figure 7.5 (left). Top: concentration profile of the reactor with respect to the prominent species and diversity as the number of unique species present in the reactor. Bottom left: zoomed into [150:250] to show in detail the dynamical behavior compensating qualitative disturbance. Bottom right: zoomed into [780:850] where stochastic effects eventually caused downward movement.

space. As suggested by our experimental results (Figures 7.2- 7.4), downward movement correlates with decreasing diversity whereas an upward movement correlates with increasing diversity. However, in general, the relation between the actual level (actual state of the system) and the organizational level (organization the system is in) is not that simple. In fact, we have shown that there can be a decrease or increase in diversity without any change on the organizational level (i.e., the organization generated does not change). Even a process that appears like a creative evolutionary process on the actual level can in fact be just a downward movement on the organizational level (see e.g. Figures 6 and 7 in Ref. [Dittrich and Banzhaf, 1998]). In other words, an increase in diversity or the appearance of new molecular species (on the actual level) does not necessarily imply an upward or sideward movement but a downward movement (on the organizational level). Finally, it is even possible that an upward movement is accompanied by a decrease of diversity, e.g., in case some new molecular species take a large portion of the reaction vessel, although we have not experimentally demonstrated this case, yet.

An important aspect left for future research is to characterize the intrin-

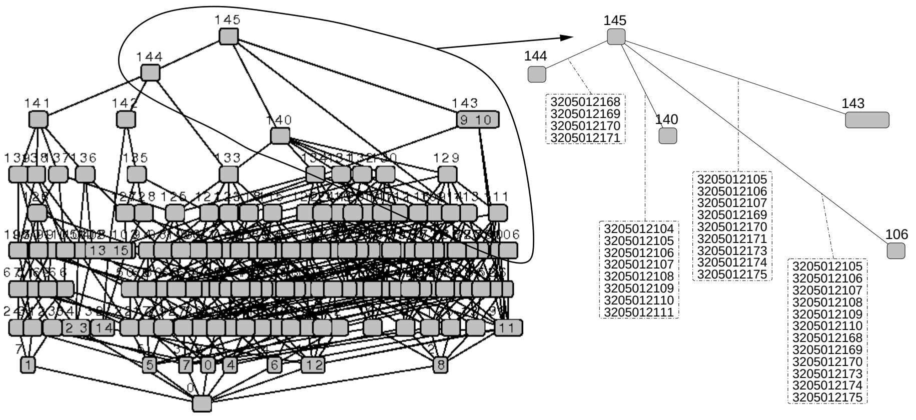


Figure 7.5: Organizational structure in the reaction network of the sixteen species with which the reactor for Figure 7.4 is initialized. **Left:** the whole lattice structure containing 146 organizations. **Right:** four organizations directly below the biggest organization (labeled as 145). For each link to below, the missing species are listed. Since twelve species constitute the organization labeled as 144, for instance, the link to that organization is associated with four species. The downward movement demonstrated in Figure 7.4 is from organization 145 to 140.

sic stability of organizations. As we observed, not all organizations show the same level of stability: some organizations are sustained over long periods while others are inherently unstable, or unstable under the small external perturbation. What exactly makes an organization stable or unstable is at the moment only a speculation, yet the topology of the reaction network [Stelling et al., 2004; Klemm and Bornholdt, 2005] and the existence of an attractor inside the organization could be important aspects to take into account.

When investigating evolutionary processes, the issue of complexity is inevitable and controversial. Previous studies suggest that evolution shows unlimited growth of complexity [Bedau and Brown, 1999]. According to a recent analysis by McMullin [2000], the research about machines that grow in complexity can be traced back to works by John von Neumann or even further to western philosophical and theological thinking. Fontana and Buss [1994] presented an artificial chemistry in which systems' complexity would increase by combining multiple non-complex systems. Another example is biochemical signaling pathways which are coupled and display emergent behaviors such as bistability [Bhalla and Iyengar, 1999]. We speculate that the organizational structure of the reaction network (the lattice structure of organizations) has a close relation to the complexity of the dynamical reaction system. The number of organizations in the network is a facet of the system complexity because of the possible association between sub-organizations and dynamical behaviors. The structural features of the lattice, not only the size of the organizational structure, must also be taken into consideration. The sub-organizations are connected linearly, or they form a hypercube or even more complex patterns. All these aspects may be relevant to the evolution of the system, yet the exact way in which they would affect it is still to be investigated and evaluated.

Source :

Matsumaru, N., Centler, F., Zauner, K.-P., and Dittrich, P. (2004). Self-adaptive scouting - autonomous experimentation for systems biology. In Raidl, G. R., Cagnoni, S., Branke, J., Corne, D. W., Drechsler, R., Jin, Y., Johnson, C., Machado, P., Marchiori, E., Rothlauf, F., Smith, G. D., and Squillero, G., editors, *Applications of Evolutionary Computing, EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC Coimbra, Portugal, April 2004, Proceedings*, volume 3005 of *LNCS*, pages 52–62. Springer, Berlin.

Chapter 8

Scouting: an Exploration algorithm

Contents

8.1 Exploration as Design Approach

This chapter and the following chapter are devoted to an explorative approach. Programming strategies discussed in Part I and in Chapter 6 are always associated with modifying the reaction network structure. The explorative approach is conceived to use chemical reaction systems as they are. The basic idea is to use an automated system to explore the behavior of the chemical reaction system, as a preliminary step. Then, a specific aspect of the system's behavior will be chosen to be utilized for a particular computational purpose. When dealing with real chemical systems, this approach has an apparent advantage because modifying the reaction network is very much restricted. Manipulating only one specific reaction link with no effect on other links is also impractical. Furthermore, it may be most reasonable to let biological or biochemical systems run as they are supposed by nature. Our knowledge about those natural systems is too limited to fully predict effects, including impacts on environments, of the system modifications.

The explorative approach is motivated by a preceded study of an exploration algorithm called *scouting* by Pfaffmann and Zauner [2001]. In this chapter, this algorithm is introduced together with two self-adaptive mechanisms we incor-

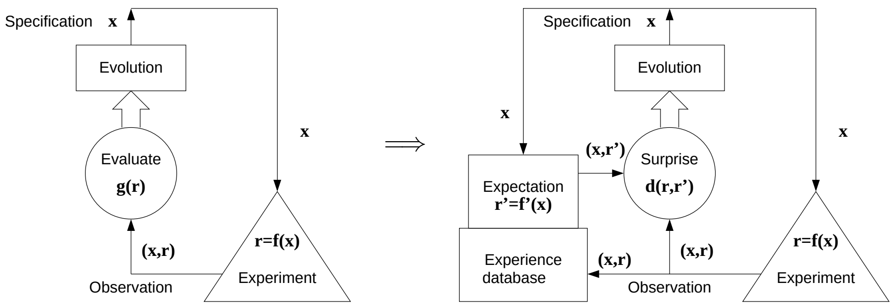


Figure 8.1: Scouting combines the notion of information being equivalent to surprise value with evolutionary computation for autonomous exploration. Conventional evolutionary experimentation (left) is compared with Scouting algorithms (right). See text for details.

porated into. The operational behaviors of the algorithm are demonstrated through examples of three applications: one in this chapter (Section 8.4) and two in the next chapter. Note that this programming approach is still suggestive, and only the behaviors of the scouting algorithms are studied in this thesis.

8.2 Scouting Algorithms

The scouting algorithm [Pfaffmann and Zauner, 2001] is an evolutionary experimentation method for autonomous experimentation. Experiments are dynamically scheduled to explore systems' behavior such that maximal information gain at each step is achieved. In accordance with communication theory, information is quantified as the surprise value of arriving data [Cherry, 1966]. Motivations of this algorithm are to obtain experimental data sufficient to build quantitative system-level models as intended in Systems Biology Kitano [2002]. Automated high-throughput methods and recent sensor technologies made it possible to produce a large quantity of data. To realize their potential, however, computational techniques have to be bear not only to discover regularities in existing data, but rather the experimental procedure itself has to be embedded in a closed-loop discovery process [Langley et al., 1987; King et al., 2004].

In the present context, the work by Kulkarni and Simon [1990] is of particular interest. They developed a program that attempts to generate experiments, in which unexplained phenomena are enhanced. Notably, the program does not start out with a pre-set goal as is common in optimization experiments but decides on its objectives dynamically. This work demonstrated that an algorithm can successfully navigate an immense search space by emulating the interplay of adjusting hypotheses and modifying experiments, which is characteristic of human experimenters [Gooding, 1990].

Figure 8.1 shows a general scheme of the scouting algorithm in a comparison with the conventional evolutionary experimentation. An evolution strategy breeds a genotype specifying experimental conditions such as parameter settings, annotated as specification **x**. The experiment specification is forwarded to experiment setup **f** to conduct the experiment as specified. Response **r** is measured, forming observation $(\mathbf{x}, \mathbf{r})$ together with the experiment specification. The empirically observed response, i.e., the phenotype, is evaluated to determine the fitness of the corresponding genotype. In a conventional algorithm (Figure 8.1 left), the phenotype is evaluated using a prescribed fitness function g. The conventional evolutionary algorithm is renowned as an optimization method to find optimal value of $\mathbf{x}$ to satisfy the given criteria. An example is parameter fitting, to determine parameter values $\mathbf{x}$ such that the resulting response $\mathbf{r}$ fits to the predetermined observations or pre-obtained data. The scouting algorithm (Figure 8.1 right) is not considered as a classical optimization method, however. The extension of the conventional evolutionary algorithm is rather intended to direct an empirical exploration of a complex system in order to gain as much information about the experiment as possible.

In the scouting algorithm as shown in the right-hand side of Figure 8.1, there are additional actions to determine the fitness of a genotype, specification $\mathbf{x}$. In addition to conducting the experiment for response $\mathbf{r}$, the experience obtained in the past experiments is used to make a prediction of what would happen if an experiment were performed as specified by $\mathbf{x}$. This is the expectation $\mathbf{r}'$ regarding the outcome of the specified experiment. These two phenotypical values $\mathbf{r}$ and $\mathbf{r}'$ are compared in order to appraise the corresponding genotype $\mathbf{x}$. When the deviation between expectation $(\mathbf{x}, \mathbf{r}')$ and observation $(\mathbf{x}, \mathbf{r})$ is large, high fitness value is assigned to that genotype $\mathbf{x}$. It follows that the more unexpected the outcome of an experiment is, the more likely similar parameter settings will be investigated in subsequent experiments. Hereto we borrow from communication theory the notation that the information contents of a message corresponds to its 'surprise value' [Cherry, 1966; Hartley, 1928; Shannon and Weaver, 1949; Lindley, 1956].

Simultaneously, the scouting algorithm contains another dynamics related to the experience database. After the fitness value is assigned, the new observation is added to the experience database, and every experiment performed contributes to the database during the scouting run. This database together with a prediction mechanics forms an empirical model $\mathbf{f}'$ of the experiment given, and this model is used to formulate an expected response $\mathbf{r}'$ of the experiment. As a consequence the expectations for experiments under similar conditions are likely to be more accurate to the measured responses. Thus a parameter setting that leads to a surprising response will initially attract further investigation through the descendants of the genotype that discovered the setting. Then, as more information on the conditions accumulates in the experience database, the response to such settings will become 'common knowledge' and therefore the fitness of genotypes specifying experiments in this region of the parameter space will be low. Parameter settings in the other regions will be selected for subsequent experiments. Incidentally this property prevents the scouting method from getting stuck on a local fitness peak—an important issue in the context of evolutionary experimentation. This aspect is noteworthy when the cost of experiments is high. Costly fitness evaluation, including getting empirical observations, necessitates the use of small populations, and this issue of trapped in a local peak becomes critical when the small size of population is used.

8.3 The Self-adaptive Scouting Algorithm

Combining the scouting algorithm with two parameter adaptation strategies, we advanced the algorithm to be self-adaptive. Mutation strength, how different from a parent an offspring is, and population size, how many offsprings are generated from one parent, are two parameters whose values are automatically adapted as scouting runs. Those two parameters are known to be critical for the behavior of even conventional evolutionary algorithms. In this section, that adaptation strategies for mutation strength and population size are described. Then, we present the pseudo code of *Self-adaptive Scouting* algorithm [Matsumaru et al., 2004].

8.3.1 Adaptive Mutation Strength

In former implementations of the scouting algorithm, mutation strength σ was a manually defined step-function dependent on the surprise value of the parent experiment (cf. [Centler et al., 2003] and Section 9.2). In the evolution step of the algorithm, an offspring is created from the parent by adding a normally distributed value with mean 0 and standard deviation σ. Varying σ controls the strength of the mutation. Given the current surprise value s and the current mutation strength σ, the mutation strength is adapted as follows:

$$\sigma \leftarrow \sigma \cdot e^{(\bar{s}-s)/\bar{s}}, \tag{8.1}$$

where $\bar{s}$ is the average surprise value over all past experiments. As a result of this adaptation, the region from which offspring are chosen shrinks if the current surprise is above the average surprise. A surprise value below the average, on the other hand, causes the region to expand—eventually leading to random search.

We also investigated standard step size adaption methods from evolution strategies (ES), e.g., [Rechenberg, 1994; Hansen and Ostermeier, 2001], and found that those adaptation methods fail to keep up with the rapid dynamics of our fitness landscape. In the scouting algorithm, the fitness is obtained as the deviation between an expectation computed from the experience database and an observation. Since every experiment is stored in the experience database, the expectation improves continuously and the fitness landscape changes rapidly. In a deterministic setting (e.g., where the experiment is a simulation without randomness) even the individual with the highest fitness will have zero fitness when it is evaluated a second time (cf. [Weicker and Weicker, 1999; Arnold and Beyer, 2002]).

8.3.2 Adaptive Population Size

Originally, the population size λ had to be given by the user and determines the number of offspring generated from one parent (i.e., a $(1, \lambda)$-strategy in ES-terminology). If the population size is constant, we found that some individuals with high fitness are discarded because there was one individual with even higher fitness selected as the parent for the next generation. Conversely, it also happened that individuals with low fitness are selected as a parent.

The second adaptation scheme is developed to avoid this situation. We introduced an adaptive generation change with employing a threshold value. Whenever the surprise value is higher than a threshold, the specification of the experiment is selected as a parent. The threshold at generation g is denoted as $\Theta^{(g)}$ and defined as the average surprise value of the second-best individuals in the past generations:

$$\Theta^{(g)} := \frac{1}{g-1} \sum_{k=1}^{g-1} s_{2;\lambda_k}^{(k)} \quad \text{for } g > 1, \qquad \Theta^{(1)} := 0. \tag{8.2}$$

Following Beyer and Schwefel [2002], we use the notation of *order statistics* (e.g., Arnold et al. [1992]) by identifying the surprise value of the second-best individual out of λ_k individuals of generation k by $s_{2;\lambda_k}^{(k)}$. The population size at generation k is λ_k. Generally speaking, the mth-best individual out of n individuals of generation k is denoted as $s_{m;n}^{(k)}$. For a generation with only one member, we define this individual to be the "second-best": $s_{2;1}^{(k)} := s_1^{(k)}$, where $s_i^{(k)}$ is the surprise value of the ith individual of generation k (in the order of experiments performed).

The threshold is calculated as described above because the second-best surprise value separates the best, which is selected as the parent for the next generation, from the other offspring. The second-best surprise value works implicitly as a threshold in each generation. Furthermore, this method guarantees the threshold to be above the average surprise value $\bar{s}$, so that the mutation strength σ can become both bigger and smaller using the scheme of Section 8.3.1.

8.3.3 Pseudo Code

The complete scouting algorithm is presented here in more detail as pseudo code. During initialization (line 1–5), the minimum mutation strength σ_{min} and the number of experiments to perform t_{max} is set by the user. The mutation strength σ is initialized with 1. The number of the current experiment t and the number of the current generation g is set to 0. In line 6–9, an initial experiment specification is randomly chosen, the experiment performed, and stored with response $\mathbf{r}$ in the experience database DB. $\mathbf{x}_i^{(g)}$ is the experiment specification of the ith individual of generation g. Within the **while**-loop, a new generation is started by choosing the parent $\mathbf{x}^{(g)}$ of generation g to be the last individual of the last generation (line 13). The new generation is then populated within the **repeat**-loop. In line 16, a new experiment specification is created as a mutated copy of the parent individual (see Section 8.3.1), and the expectation $\mathbf{r}'$ is computed from the experience database. This is done here, as in [Matsumaru et al., 2002], by averaging over the (up to) 5 closest experiment entries from the experience database with inverse cubic distance weighting. The response $\mathbf{r}$ is derived by performing the experiment, and the result is stored in the experience database. Finally, the surprise value is calculated in line 21, and the mutation strength is adapted in line 22 (see Section 8.3.1). In generation

Table 8.1: Listing of self-adaptive scouting. The scouting algorithm is incorporated with two self-adaptive mechanisms of mutation strength and population size.

Algorithm: Self-adaptive Scouting

```
1  $\sigma_{min}$ ← minimum_resolution
   $t_{max}$ ← maximum_experiments  # number of exp.  to perform
   $\sigma$ ← 1                     # mutation strength
   $t$ ← 0                          # time (experiments)
5  $g$ ← 0                          # time (generations)
   $\mathbf{x}^{(0)}_{\lambda_0}$ ← random_specification  # choose initial experiment
   $\mathbf{r}$ ← $f(\mathbf{x}^{(0)}_{\lambda_0})$     # conduct experiment
   $t$ ← $t+1$                      # increment time (experiments)
   DB ← InsertIntoDB(DB, ($\mathbf{x}^{(0)}_1$, $\mathbf{r}$)) # initialize experience DB
10 while $t < t_{max}$ do
       $g$ ← $g+1$                  # start a new generation
       $\lambda_g$ ← 0              # number of individuals
       $\mathbf{x}^{(g)}$ ← $\mathbf{x}^{(g-1)}_{\lambda_{(g-1)}}$   # choose the parent
       repeat
15         $\lambda_g$ ← $\lambda_g + 1$              # add a new individual by
           $\mathbf{x}^{(g)}_{\lambda_g}$ ← Mutate($\mathbf{x}^{(g)}, \sigma$)   #        mutating the parent
           $\mathbf{r}'$ ← Predict(DB, $\mathbf{x}^{(g)}_{\lambda_g}$)  # compute expectation
           $\mathbf{r}$ ← $f(\mathbf{x}^{(g)}_{\lambda_g})$             # conduct experiment
           $t$ ← $t+1$                # increment time (experiments)
20         DB ← InsertIntoDB(DB, ($\mathbf{x}^{(g)}_{\lambda_g}$, $\mathbf{r}$))  # update DB
           $s^{(g)}_{\lambda_g}$ ← d($\mathbf{r}, \mathbf{r}'$)        # evaluate fitness as surprise
           $\sigma$ ← Max($\sigma_{min}, \sigma \cdot \exp(1 - s^{(g)}_{\lambda_g}/\bar{s})$ )
                                      # adapt mutation strength
       until $s^{(g)}_{\lambda_g} > \Theta^{(g)}$ or $t \geq t_{max}$
   end
```

g, the mean surprise value over all experiments is calculated as follows:

$$\bar{s} := \frac{1}{g} \sum_{k=1}^{g} \frac{1}{\lambda_k} \sum_{i=1}^{\lambda_k} s_i^{(k)}. \tag{8.3}$$

The **repeat**-loop is left and a new generation started, once the surprise value of an experiment is above the threshold $\Theta^{(g)}$ (see Section 8.3.2).

8.4 Scouting an HIV-immune System Model

Now we demonstrate the behavior of our new algorithm by applying it to a concrete model of immunological control of HIV by Wodarz and Nowak [1999]. The model is a 4-dimensional ordinary differential equation (ODE) system. A mathematical analysis reveals that the model has two asymptotically stable fixed points. Using scouting, we will now explore how the model behaves depending on its initial state given a fixed parameter setting. In passing, this model is also analyzed using chemical organization theory in Chapter 2.

8.4.1 Experiment Setup

The experiment is a dynamic simulation of the immunological control model taken from [Wodarz and Nowak, 1999]. which contains 4 variables: uninfected CD4$^+$ T cells x, infected CD4$^+$ T cells y, cytotoxic T lymphocyte (CTL) precursors (CTLp) w, and CTL effectors z. The dynamics is given by the ODE system:

$$\begin{aligned}\dot{x} &= \lambda - dx - \beta xy,\\ \dot{y} &= \beta xy - ay - pyz,\\ \dot{w} &= cxyw - cqyw - bw,\\ \dot{z} &= cqyw - hz.\end{aligned} \tag{8.4}$$

Uninfected CD4$^+$ T cells are produced at a rate λ, decay at a rate dx, and become infected at a rate βxy. Infected cells decay at a rate ay and are killed by CTL effectors at a rate pyz. The production of CTLp at rate $cxyw$ requires uninfected CD4$^+$ cells, virus load represented by y, and CTLp themselves. CTLp decay at a rate bw and differentiate in CTL effectors at a rate $cqyw$. CTL effectors decay at a rate hz. Here we set the parameters as in [Wodarz and Nowak, 1999] $\lambda = 1, d = 0.1, \beta = 0.5, a = 0.2, p = 1, c = 0.1, b = 0.01, q = 0.5, h = 0.1$.

Given a specification $\mathbf{x} = (x_1, x_2, x_3, x_4)$, we perform the experiment and obtain the response $\mathbf{r} = (r_1, r_2, r_3, r_4)$ in the following way: (1) We set the initial state of the ODE system as follows: $x_{t=0} = x_1, y_{t=0} = x_2, w_{t=0} = x_3 \times 0.05, z_{t=0} = x_4$. (2) We integrate the ODE numerically (using *lsode* built in *octave* [Eaton, 2002]) for a duration of t_1 (here, $t_1 = 500$) and obtain the response as the final state: $r_1 = x_{t=t_1}, r_2 = y_{t=t_1}, r_3 = w_{t=t_1}, r_4 = z_{t=t_1}$.

8.4.2 Scouting the Model Behavior

For scouting, we set the range of specification $\mathbf{x}$ of the experiment to $[0,1]^4$ and the minimum mutation strength $\sigma_{min} = 0.01$. We allow a total number of $t_{max} = 2000$ experiments in a scouting run. Every experiment integrates numerically the ODE representing the HIV immunological dynamics.

Figure 8.2 shows the progress of the surprise value $s_j^{(g)}$ (top) and mutation strength σ (bottom) of a typical run of (self-adaptive) scouting. In Figure 8.3, experiments 50–100 are shown in detail. Surprise value and mutation strength are plotted together with the average surprise value $\bar{s}$ to illustrate the mutation strength adaptation. The difference between the current surprise value and the average surprise determines the adaptation of the mutation strength according to Equation. 8.1.

The time evolution of the population size λ_g is plotted on the left-hand side of Figure 8.4. For the purpose of explaining the population size adaptation, the right-hand side of the graph shows in detail the surprise value of every individual experiment from generation 10–17, the second-best surprise, and the threshold. Generation 17 consists of 6 individuals $\mathbf{x}_{40} \dots \mathbf{x}_{45}$ ($\mathbf{x}_i$ denotes

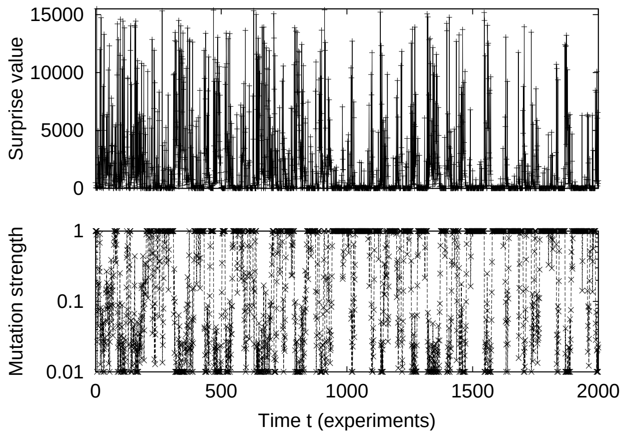


Figure 8.2: Surprise value $s_i^{(g)}$ and mutation strength σ while exploring the behavior of the HIV immunology model with the self-adaptive scouting algorithm. See text and Figure 8.3 for details.

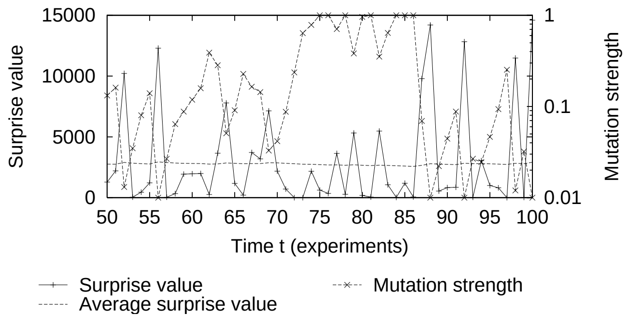


Figure 8.3: Surprise value $s_i^{(g)}$ and mutation strength σ while exploring the behavior of the HIV immunology model with self-adaptive scouting. A surprise value higher than the average decreases the mutation strength. The mutation strength is increased, when the surprise value is less than the average (see Equation 8.1). This mutation strength adjustment helps the scouting algorithm to concentrate the samples on surprising regions

the ith experiment conducted). Because the surprise of the 6th offspring is greater than the threshold, the individual $\mathbf{x}_{45}$ is selected to be the parent of generation 18. The surprise experienced by the experiment $\mathbf{x}_{43} = \mathbf{x}_4^{(17)}$ is the second-best surprise value in the generation. This value is used to calculate the threshold for the following generation. Since the first offspring of generation 15 ($\mathbf{x}_{35} = \mathbf{x}_1^{(15)}$) yields a higher surprise than the threshold, the population size of generation 15 is 1. The second-best surprise value for this generation is, in this case, the best one.

Figure 8.5 shows the 2000 specifications sampled by the scouting algorithm. The 4-dimensional data is projected on 6 graphs showing every possible com-

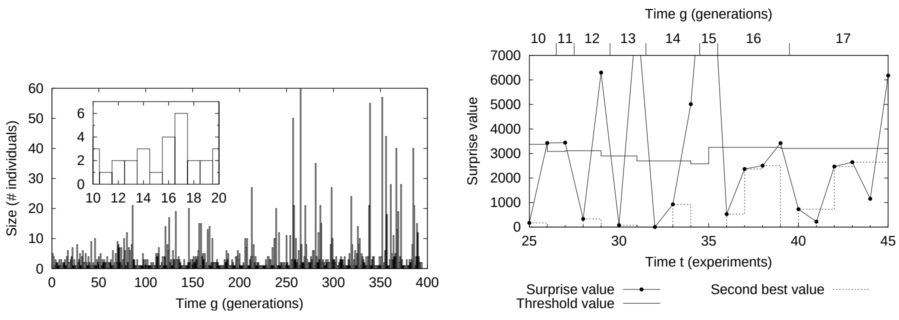


Figure 8.4: Population size λ_g and threshold $\Theta^{(g)}$. See text for details

bination of the four dimensions. Each point represents an initial state of the ODE model. The sampling points seem to spread equally except for the last graph with CTL precursors and effectors as axes. The pattern appearing in the last graph matches with the border of the two basins of attraction of the two asymptotically stable fixed points of the ODE model. The respective dynamical behavior of the model is shown in Figure 8.6. As seen in Figure 8.5, scouting has explored the borderline between the two modes of behavior more accurately, thus the borderline can be described much more precisely than for cases where systematic or random sampling would have been used. To illustrate this, a 2-dimensional projection of systematic sampling given by a full 7^4 factorial design is shown in Figure 8.7. In this design of experiments, each of the 4 factors $x_1, \ldots, x_4$ is explored equidistantly on 7 levels [Hinkelmann and Kempthorn, 1994]. With approximately the same number of experiments as in the scouting method, the borderline can only roughly be approximated.

8.5 Concluding Remarks

We introduced an algorithm capable of exploring unknown phenomena without the need for manual adjustments aimed at an application domain. We described how the two parameters crucial for the exploration, the mutation strength and the population size, can be adapted automatically, and why existing techniques for evolutionary optimization were not applicable. Our experience with the algorithm provides some evidence that it can be applied usefully for exploring complex systems. However, the next important step is to quantify the performance of scouting systematically. A suitable assessment measure may be the predicting strength of the experience database after a given number of samples. The process of evolution underlies the complexity observed throughout the realms of biology—it may also hold the key to tackle this complexity.

8.5.1 Scouting is not optimization

Finally, it is important to emphasize here that scouting algorithms are not classical optimization methods even though they are a variant of evolutionary algorithms. Generally, the aim of an optimization algorithm is finding the best solution among all possible solutions. More formally, given an objective function $F\colon Y \to Q$, which assigns a certain quality $\mathbf{q} \in Q$ to each solution from the search space Y, an optimization algorithm tries to find the solution $\mathbf{y} \in Y$

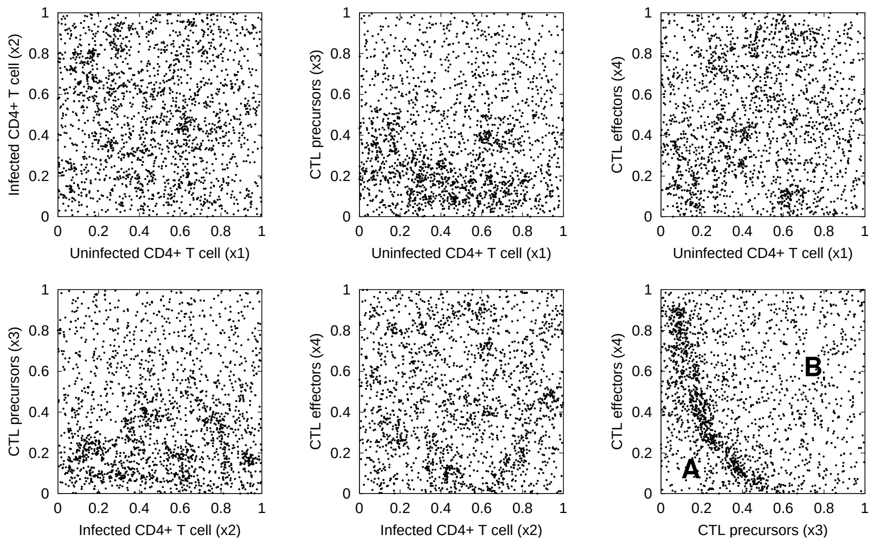


Figure 8.5: Six different 2-dimensional projections of 2000 sampled locations (specifications) of a typical scouting run on the HIV immunology model developed by Wodarz and Nowak [1999]. This model is described as an ODE with four variables. The scouting algorithm initializes the variables, which are plotted as dots (the range of CTL precursors is scaled to $[0, 0.05]$), and observed the states of the model after 500 time steps as the response. When the locations are plotted with CTL precursors and effectors as axes (lower rightmost plot), a pattern of dense area shows up. The pattern matches with the border of two modes of behavior of the model, which are shown in Figure 8.6

such that $F(\mathbf{y})$ gets maximized. The result of optimization is a single best-so-far solution $\mathbf{y}$. For scouting, an experiment $f\colon X \to R$ is given, where X is the search space consisting of all possible experiment specifications and R the set of all possible responses. In contrast to optimization, the objective here is not to find a single best experiment $\mathbf{x} \in X$ (or a pareto set), but to gain as much information about f as possible by conducting experiments. The result of scouting is an experience database, which embodies the complete knowledge acquired about f and can be considered as an empirical model. There is no objective function f given, and thus scouting is not optimization.

Trivially, every computational problem can be formulated as an optimization problem, e.g., by defining the objective function returning an optimum when its argument is the solution of the problem. For example, a sorted sequence $\mathbf{y} = (y_1, \ldots, y_n)$ optimizes the objective function $F(\mathbf{y}) = \sum_{i<j}(y_i < y_j)$. Most sorting algorithms, however, contradict the typical picture of optimization where a sequence of evaluations of the objective function leads to a solution. *Bubble sort*, for instance, might be better regarded as a greedy strategy that seeks a local optimum to achieve the global optimum. The same is true for scouting with respect to the (implicit) aim of maximizing the total information about the experiment f. Every step (or every generation) of the scouting algorithm can be viewed as a step of a greedy strategy that tries to

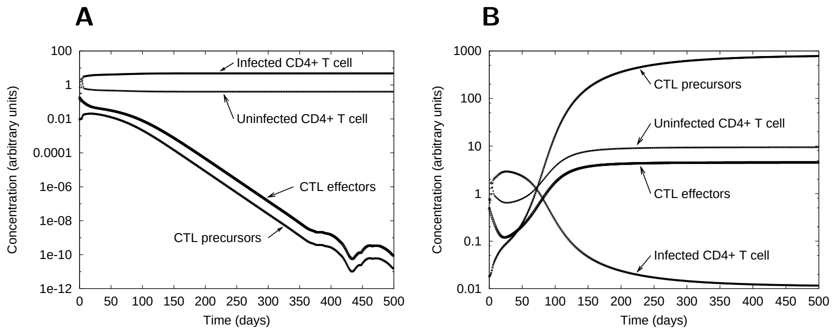


Figure 8.6: The HIV immunology model has two modes of behavior: (A) immune system damaged and (B) CTL response established. The labels correspond to those in Figure 8.5

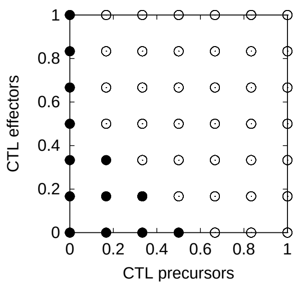


Figure 8.7: Two-dimensional projection of systematic sampling. Initial state leading to a damaged immune system are marked by •; initial states establishing CTL response are indicated by ∘. Experiments with no initial infected $CD4^+$ T cells are excluded. Comparing to the lower rightmost panel in Figure 8.5, the borderline of the two modes of behavior of the model can be approximated only roughly

maximize the local information gain in terms of maximal surprise in the next experiment.

Source :

Matsumaru, N., Colombano, S., and Zauner, K.-P. (2002). Scouting enzyme behavior. In Fogel, D. B., El-Sharkawi, M. A., Yao, X., Greenwood, G., Iba, H., Marrow, P., and Shackleton, M., editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 19–24. IEEE Press, Piscataway, NJ.

Centler, F., Dittrich, P., Ku, L., Matsumaru, N., Pfaffmann, J., and Zauner, K.-P. (2003). Artificial life as an aid to astrobiology: Testing life seeking techniques. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Advances in Artificial Life, ECAL 2003*, volume 2801 of *LNAI*, pages 31–40. Springer, Berlin.

Chapter 9

Further Applications of Scouting

Contents

We continue on studying explorative behaviors of scouting algorithms. In this chapter, two further examples are presented: The scouting algorithms are applied to explore enzyme behaviors in conjunction with automated wet-lab experiment apparatus, in order to demonstrate the possibilities for the algorithms to cope with real chemical reaction systems[1]. The other application is in the context of artificial life research, as a tool for the life-seeking endeavor[2]. These applications was investigated with original scouting algorithms, no self-adaptive mechanisms are incorporated. We only give a highlight here, so please consult the source publication for further details.

9.1 Scouting Enzyme Behavior

Enzymes are known to be context sensitive such that the catalytic activity is modulated by physicochemical signals from milieu components, which are selected and fused through the conformational dynamics [Conrad, 1992; Freire, 1998; Sols, 1981]. By changing milieu conditions, enzymes vary their activity level. This response has been suggested by Zauner and Conrad [2001] to be a computing principle using bio-macromolecules. In conjunction with the computer controlled fluidics setup shown in Figure 9.2, the scouting algorithms are employed to probe the enzyme malate dehydrogenase (MDH) for its response to chemical signals of $MgCl_2$ and citrate.

9.1.1 Experiment Setup

[1]This wok has been done as a part of the author's thesis for the degree of Master of Science from Wayne State University, Detroit, Michigan, USA.
[2]This work is reported in the publication by my colleagues, Centler et al. [2003]

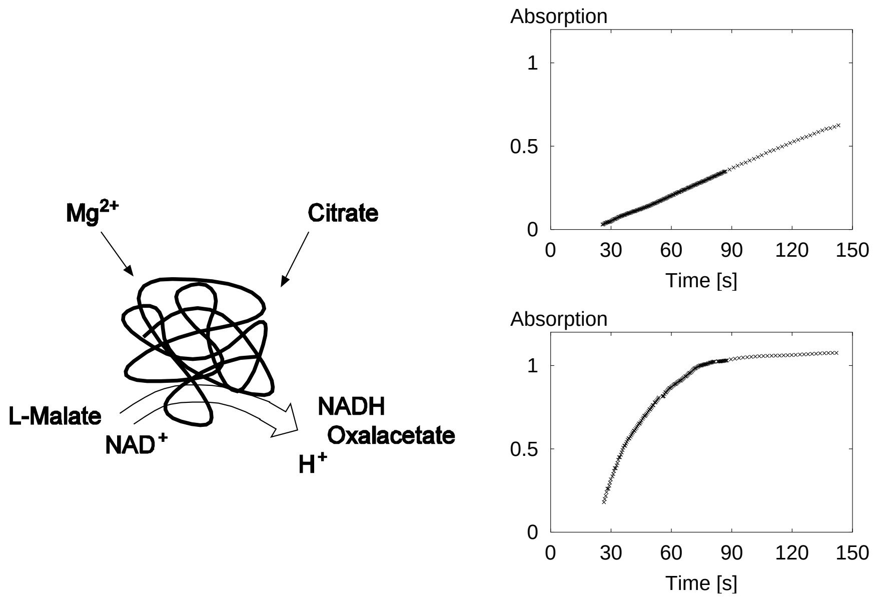


Figure 9.1: Schematic representation of the enzymatic reaction as in Eq. 9.1 (left) and empirical data of Mg^{2+} effects on the enzyme activity (right). Mg^{2+} and citrate in milieu do not participate in the reaction the enzyme MDH catalyzes, but the catalytic activity of the enzyme is influenced by those factors through the conformational dynamics. The activity corresponds to the production rate, which can be measured with spectrophotometer due to the different light absorption characteristics at the wave length of 339nm. The graphs on right-hand side show the progress of the reaction catalyzed. The enzyme is activated by a small amount of Mg^{2+} as shown in the bottom graph, compared with no Mg^{2+} in milieu as shown in the above graph.

The laboratory setup conducts an enzyme assay that measures enzyme activity with respect to milieu composition. An enzyme malate dehydrogenase (MDH) catalyzes the oxidation of malate to oxalacetate by reducing the oxdized form (NAD^+) of nicotinamide adenine dinucleotide to the reduced form NADH [Englard and Siegel, 1969]:

$$\mathrm{L-malate} + \mathrm{NAD}^+ \xleftrightarrow{MDH} \mathrm{oxalacetate} + \mathrm{NADH} + \mathrm{H}^+ \qquad (9.1)$$

NADH differs in its absorption of ultraviolet light significantly from NAD^+, a property that allows for the convenient monitoring of the reaction by measuring the increase in NADH concentration with a spectrophotometer [King, 1965]. At high pH the equilibrium for the reaction is on the right side. The progress of the reaction is affected by the composition of the reaction milieu. The two factors which we vary are citrate, a known regulator with context dependent activating or inhibiting effect on MDH [Gelpí et al., 1992] and $MgCl_2$, also reported to have both activating and inhibiting effects on MDH [Wong and Smith, 1976; Bracht and de P. Campello, 1979; Dér and Ramsden, 1998]. The factors that could be investigated are not limited to chemical compounds known

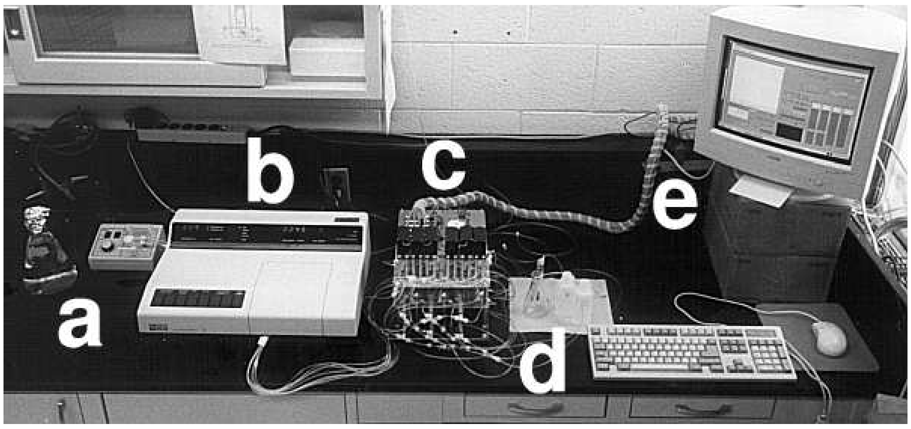


Figure 9.2: Laboratory setup used in the scouting experiments. A peristaltic pump and a reservoir of distilled water (a) serve to flush the cuvette that is installed in the spectrophotometer (b). Servo driven syringe pumps and valves (c) are controlled through a serial interface (e) by the computer that runs the evolutionary search program. The syringe pumps first mix chemical milieus from stock solutions (d) in the photometer cuvette. A reaction is then initiated by injecting enzyme solution into the cuvette and its progress is monitored with the photometer. Adopted from [Matsumaru et al., 2002]

to have a physiological function, however.

Every genotype evaluation causes a computer controlled experiment to be performed. The outcome of the experiment is a time series of (over 300) absorption measurements reflecting the activity of the enzyme. To facilitate the representation of this response in the experience database we fit the following expression to the measured data:

$$a(t) \;=\; c_1\left(1 - c_2 e^{-c_3 t}\right) + c_4 t + c_5 t^2 + c_6 t^3 \tag{9.2}$$

The entry in the experience database is a pair of the factor levels and the parameters representing the observation. To obtain the expectation corresponding to a particular factor setting, first $n = 5$ (or less if not enough entries exist) nearest neighbors in the parameter space and their distances to the factor setting under consideration are determined. Then the expectation $a_{ex}(t)$ is calculated from the parameter sets of these n entries averaging with inverse cubic distance weighting (w_i):

$$a_{ex}(t) \;=\; \sum_{i=1}^{n} w_i a_i(t), a_{ex}(0) = 0 \tag{9.3}$$

The surprise value (s) of an observation is calculated by numerically integrating the absolute difference between the expected absorption $a_{ex}(t)$ and the observed absorption $a_{ob}(t)$:

$$s \;=\; \frac{1}{3}\int_{t=0}^{t=300s} \text{abs}\left[a_{ex}(t) - a_{ob}(t)\right] \tag{9.4}$$

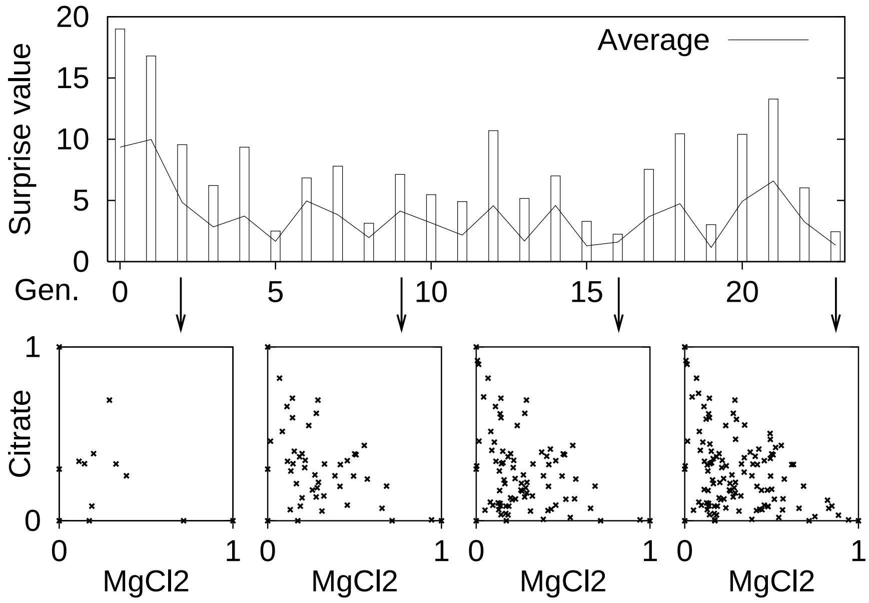


Figure 9.3: Temporal progress of scouting, see text for details.

Accordingly the fitness of a genotype is the surprise value.

9.1.2 Results

The scouting above described was used to conduct an experiment aimed at mapping out the sensitivity of the reaction catalyzed by MDH with respect to two effector substances, $MgCl_2$ and citrate. Both were varied over the range 0–300 mM. The system probed 120 locations in the factor space, evolving for 23 generations. Figure 9.3 depicts the progress of these sampling dynamics. In the lower graphs, the locations in the factor space at which experiments were performed are shown for four time slices. Note that the milieu component axes are scaled as relative concentration, therefore only the milieu compositions represented by the lower left half of the diagrams are possible. Bars in the upper graph show the development of the maximum surprise value found within the population; the curve shows its average surprise value in a population. The population size was 5 except in generation 0 which was initialized with three genotypes. As can be seen from the bar graph in the figure, peak surprise values have not yet leveled off and consequently further information about the behavior of the enzyme would be gained by taking more than the 120 samples.

The observations collected in the scouting process can be combined into an empirical description of the probed phenomenon. Two snapshots of the time development of the absorption during the reaction are shown in Figure 9.4. The contours allow us to visualize the interaction between the two milieu-factors.

9.2 Artificial life as an Aid to Astrobiology

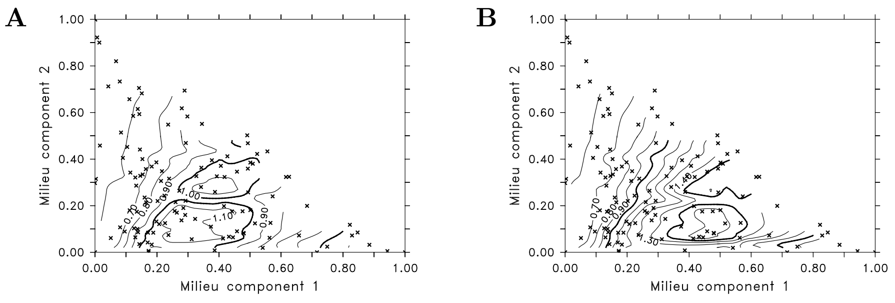


Figure 9.4: Contours show interpolated NADH absorption in arbitrary units at times 90 s (A) and at 200 s (B) after the reaction start. The axes show relative concentration of $MgCl_2$ as component 1 and citrate as component 2. The contours represent the information gained from 120 assays run with the milieu compositions marked by crosses.

The scouting algorithm is also studied in the context of seeking for signs of extraterrestrial biota. There are two special aspects to regard for this endeavor. One is a broader view about life form as outlined in NASA Astrobiology Roadmap [2002] since our biosphere's specific molecular machinery such as DNA and proteins may not work to identify examples of life elsewhere. The capability of recognizing life in whatever form it may take is crucial. This is already addressed in artificial life research from its very onset; in Langton's words:

> "[...] certainly, the dynamic processes that constitute life—in whatever material bases they might occur—must share certain universal features—features that will allow us to recognize life by its dynamic form, without reference to its matter." [Langton, 1989, p. 2]

The other aspect is the demand of highly autonomous devices because the search for signs of extraterrestrial biota is characterized by vast, hard to access areas and severe restrictions on communication bandwidth with limited assistance from earth. The scouting algorithm has addressed the autonomous guide of experiments. Combing these two, we reported on the application of artificial chemistry modeling [Dittrich et al., 2001] to evaluate an autonomous experimentation technique with regard to its capability to detect chemical signatures of life. The results presented in this section is based on the publication by Centler et al. [2003][3].

9.2.1 Experiment setup

To create a test scenario for the study of the behavior of the scouting algorithm, we simulated an abstract artificial planetary sphere with an inanimate chemistry. The assumption here is that life necessitates the synthesis of a larger

[3]the author is listed as the fourth author.

variety of substances than commonly produced by inorganic reactions. The chemical composition is locally perturbed when life forms are introduced. The scouting algorithm is then used to explore the planetary sphere.

For simulating spatially inhomogeneous chemistry a two-dimensional cellular automata implementation of the chemical dynamics is convenient [Adamatzky, 1994]. The field is arranged as a 200×200 cellular automaton, and the state of each cell represents a chemical component present at the location of the cell. All reaction rules take the form $\mathrm{A}+\mathrm{B} \xrightarrow{p} \mathrm{C}+\mathrm{D}$, two reactants and two products associated with the probability p of the reaction to occur. If component B is present within the set of four (von Neumann) neighbors of component A, then A may be substituted by C and B substituted by D. The probability for this event to take place is proportional to p. Because the order of components in the reaction rules matters, diffusion is expressed by the rules of the form $\mathrm{A}+\mathrm{B} \xrightarrow{p} \mathrm{B}+\mathrm{A}$.

In the cellular automaton field, randomly created constructive artificial chemistry described in [Speroni di Fenizio and Dittrich, 2002] is implemented. The chemical model is defined so that substances are always created by the co-operative action of two catalysts: $\mathrm{X} \xrightarrow{(\mathrm{E},\mathrm{F})} \mathrm{Y}$ where X is a substrate molecule, E and F are co-acting catalysts and Y is the reaction product. This was motivated by biopolymer synthesis process and entails the assumption of an inexhaustible pool of building blocks being available, but the present model does take substrates explicitly into account. To realize these reactions, in which a substrate, a product, and two catalysts participate, with the binary reaction scheme of the cellular automata, intermediate agents have been introduced for each reaction. Above reaction would be represented in the cellular automata by two rules:

$$\mathrm{E}+\mathrm{F} \xrightarrow{p_1} \mathrm{E}+\mathrm{I}, \tag{9.5}$$

$$\mathrm{X}+\mathrm{I} \xrightarrow{p_2} \mathrm{Y}+\mathrm{F}. \tag{9.6}$$

The intermediate agent I is assumed to be highly reactive and leads to a rapid transformation of substrate X to product Y, i.e., $p_1 \ll p_2$.

In Table 9.1, the reaction network is shown. Note that the reaction matrix is asymmetric. Each column represents reactions when the substance referred at the top row takes the place of E in Equation 9.5. Each row corresponds to reactions occurring when F in Equation 9.5 and Equation 9.6 is substituted with the substances referred in the leftmost column. Entries in the table represents the reaction product Y. Substance numbers marked with bold face are selected to serve as substrates X.

For the purpose of supplying the enough substrates, the reaction network includes decay reactions that transform any substance other than intermediates into the substrate substances through rules of the form $\mathrm{A}+\mathrm{B} \xrightarrow{p} \mathrm{X}+\mathrm{B}$, where $A, B \in \{0, 1, \ldots, 9\}, \mathrm{X} \in \{0, 6, 7\}$. Finally, an additional catalyst is introduced as *biota*. It consumes the substrate molecules {0,6,7} and excretes the substance 1. For all simulations in which the biota is present, exactly six cells are set to the biota state: 6 cells for one biota patch, 3 cells each of two patches, and 2 cells each of three patches. All substances, but not the biota, had the ability to diffuse.

Initialized with a pseudo-random distribution of components, the cellular

Table 9.1: The network of potential reactions. Numbers in the leftmost column refer to substances that take the place of E in equation 9.5, numbers in the top row correspond to F in equations 9.5 and 9.6, and entries in the table correspond to Y in equation 9.6. Substances that can serve as substrates are marked with bold face.

	0	1	2	3	4	5	**6**	**7**	8	9
0		0						6		
1	2			5						
2				1						0
3	5			0						1
4	8		6	6	4		8	4	8	5
5				1		9				8
6		8	0					7	2	
7		5				5		9		
8	1	2	9	1		6	3			2
9	0	8		0	3		6		0	0

automaton field is updated 10^8 times (2500 updates per cell). To update the simulation space, a cell and one of its von Neumann neighbors are pseudo-randomly selected. The subset of rules applicable to the two chemical components located in the two cells is determined and in accordance with their probabilities one of these rules may be applied to transform the content of the two cells. A snapshot of the updated component distribution in the simulation space is the target system to explore using the scouting algorithm.

Experience and expectation The scouting algorithm is applied to detect unusual chemical signatures in a complex background chemistry [Yung and DeMore, 1999]. It is assumed that the chemical composition can be sampled locally, but no a priori knowledge of the chemical effects of potentially present biota is available to the algorithm. For sampling the algorithm has to choose a location $\mathbf{x} = (x, y)$ in the simulation space. The measured data at this location is a concentration vector $\mathbf{r} = (r_1, \ldots, r_n)$ for each of the n substances in a ± 2 pixel vicinity. Correspondingly the entries $(\mathbf{x}, \mathbf{r})$ in the experience database and the expectations $(\mathbf{x}, \mathbf{r}')$ formed for a location prior to its sampling are such vectors. The surprise value d constitutes the fitness criteria, computed as follows:

$$d(\mathbf{r}, \mathbf{r}') = -\sum_{i=1}^{n} |r_i - r'_i| \ln |r_i - r'_i| \tag{9.7}$$

The expectation $\mathbf{r}'$ for a location $\mathbf{x} = (x, y)$ is computed as distance-weighted average over the (up to) 25 measurements nearest to $\mathbf{x}$ available in the experience database. A population size of 10 genomes, all offspring of the single best parent was used for evolution (i.e., a $(1, 10)$-strategy [Beyer and Schwefel, 2002]). We mutate an offspring by adding an equally distributed random number taken from within a radius of δ. The mutation strength δ depends on

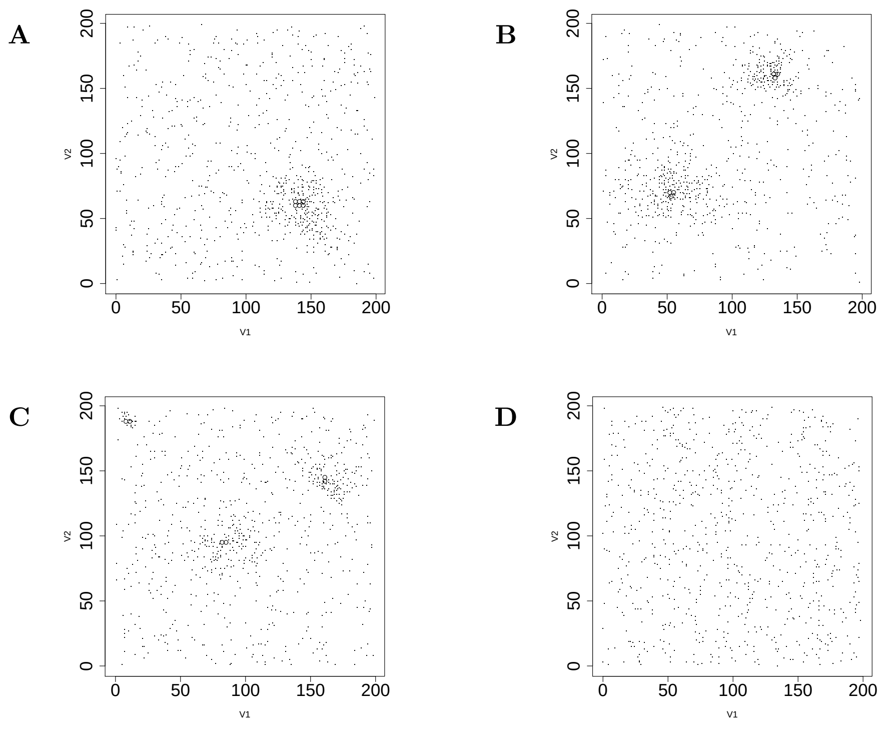


Figure 9.5: Detection of an unusual chemical composition. Actual location of biota (∘) and the 800 locations (·) probed by scouting, are shown for 1, 2, 3, and 0 biota patches, A, B, C, and D, respectively.

the previous surprise

$$\delta = \begin{cases} 0.03 & \text{if} \quad \sqrt{4.2} \le \mathrm{d}(\mathbf{r},\mathbf{r}'), \\ 0.04 & \text{if} \quad 2.0 \le \mathrm{d}(\mathbf{r},\mathbf{r}') < \sqrt{4.2}, \\ 0.1 & \text{if} \quad \sqrt{3.5} \le \mathrm{d}(\mathbf{r},\mathbf{r}') < 2.0, \\ 0.99 & \text{else} . \end{cases} \tag{9.8}$$

9.2.2 Results

As a result of applying scouting algorithm to the simulation space, Figure 9.5 shows the 800 locations probed. Their density indicates areas of high interest to the scouting algorithm. As seen in figures, sampling locations are successfully concentrated in the vicinity of the biota patches. To identify clusters in the sampling positions, hierarchical clustering method with subsequent expectation maximization (performed with *mclust* in *R* [Ihaka and Gentleman, 1996; Fraley and Raftery, 2002]) is employed. Taking the mean position of the samples allocated to one cluster, the location of biota is predicted fairly good, despite the complex chemical background.

Chapter 10

Hybrid Approach

In this thesis, we argued how to program chemical reaction systems in order to exploit their computational properties. Methods are categorized into two: constructive design and autonomous design. In Part I, constructive design, we showed mainly how chemical organization theory can serve as a tool to facilitate programming of chemical reaction systems at the level of the reaction network. This programming method is summarized under the organization-oriented programming principles in Section 5.1 as a generalized programming guideline using the notion of chemical organizations. In Part II, autonomous design, we considered methods to program chemical reaction systems autonomously, using two strategies of evolution and exploration.

These two design approaches are not necessarily separated, however. The combination, a hybrid approach, is rather fruitful for each other. When combining two programming methods, it is logical to structure the programs in a form of hierarchy [Dittrich et al., 2007]. Parts of the programs are designed with one method, while another method guides the design of the whole programs using also those parts as building blocks. An example of this hybrid approach was already presented in Chapter 6 where a chemical flip-flop is designed by evolution. The overall programming process is governed by evolutionary algorithms, an autonomous method, and the building blocks of the reaction system are manually predefined in order to improve the performance of that process. Particularly, the cooperative decay of the contradictory species, such as $a^0 + a^1 \to \emptyset$, is fixed as a building block. From the experiences of constructing a chemical flip-flop in Chapter 3, these rules are meaningful especially for that coding scheme in which two species are assigned to one boolean variable to represent contradicting states, respectively. Another example is outlined by our colleague[1]. Taking the chemical boolean functions and the chemical flip-flop constructed in Chapter 3 as building blocks, computational universality is established so that any computational program could be evolved, in principle. These building blocks could be arranged using evolutionary algorithms to build arbitrary chemical programs.

Reversing the order of the hierarchy, the constructive approach can be improved by autonomous programming methods. Specifically, parts are designed by evolution and combined manually to construct chemical programs. This includes the idea to employ models of biological organisms for our computa-

[1]by Thomas Hinze, unpublished and through personal communication

tional purpose as building blocks since the organisms in nature are evolved. The field of Systems Biology, especially, provides varieties of biology-oriented models for chemical computing, and the employment is straightforward thanks to the language compatibility of those fields: chemical reaction network. One may further extend to employ real biological systems as building blocks, not just the computational models. That way, rich complexity of physio-electro-chemical interactions, embedded within natural systems, becomes available.

Besides, there is another level of benefits received through analysis from autonomously designed systems for constructing computational systems. As mentioned in Chapter 5.2.2, an advantage of autonomous approaches is the possibilities to find novel, not rationalized (yet) solutions. Needless to say, this is the main motivations of evolutionary algorithms, an autonomous design method. Aspirin has once and still shown a noticeable example of un-rationalized novelties. The effects are known for a long time but without knowing the exact mechanisms [Flower, 2003]. By analyzing evolved systems, we can gain knowledge about those novelties, and that knowledge can be used as a source of inspiration for constructing computational systems.

Bibliography

Adamatzky, A. (1994). *Identification of Cellular Automata.* Taylor and Francis, London.

Adamatzky, A. and De Lacy Costello, B. (2002). Experimental logical gates in a reaction-diffusion medium: The XOR gate and beyond. *Phys. Rev. E*, 66(4):046112.

Adleman, L. M. (1994). Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024.

Anderson, P. W. (1972). More is different. *Science*, 177(4047):393–396.

Arnold, B., Balakrishnan, N., and Nagaraja, H. (1992). *A First Course in Order Statistics.* New York. Wiley.

Arnold, D. V. and Beyer, H.-G. (2002). Random dynamics optimum tracking with evolution strategies. In Guervós, J. J. M., Adamidis, P., Beyer, H.-G., nas, J.-L. F.-V., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature VII (PPSN-2002)*, volume 2439 of *LNCS*, pages 3–12, Granada, Spain. Springer Verlag.

Artmann, S. (2003). Artificial life as a structural science. *Philosophia naturalis*, 40(2):183–205.

Babaoglu, O., Canright, G., Deutsch, A., Caro, G. A. D., Ducatelle, F., Gambardella, L. M., Ganguly, N., Jelasity, M., Montemanni, R., Montresor, A., and Urnes, T. (2006). Design patterns from biology for distributed computing. *ACM Transactions on Autonomous and Adaptive Systems*, 1(1):26 – 66.

Bäck, T., Fogel, D. B., and Michalewicz, Z., editors (1997). *Handbook of Evolutionary Computation.* Oxford University Press, New York, NY and Institute of Physics Publishing, Bristol.

Bailey, J. E. (2001). Complex biology with no parameters. *Nat. Biotechnol.*, 19(6):503–504.

Banâtre, J.-P., Fradet, P., and Radenac, Y. (2004). Principles of chemical programming. In Abdennadher, S. and Ringeissen, C., editors, *RULE'04 Fifth International Workshop on Rule-Based Programming*, pages 98–108. Tech. Rep. AIB-2004-04, Dept. of Comp. Sci., RWTH Aachen, Germany.

Banâtre, J.-P. and Métayer, D. L. (1986). A new computational model and its discipline of programming. Tech. Rep. RR-0566, INRIA.

Banâtre, J.-P. and Métayer, D. L. (1990). The GAMMA model and its discipline of programming. *Sci. Comput. Program.*, 15(1):55–77.

Banzhaf, W., Dittrich, P., and Rauhe, H. (1996). Emergent computation by catalytic reactions. *Nanotechnology*, 7(1):307–314.

Barabási, A.-L. and Oltvai, Z. N. (2004). Network biology: Understanding the cell's functional organization. *Nat. Rev. Gen.*, 5:101–113.

Beale, R. and Jackson, T. (1990). *Neural Computing: An Introduction.* Institute of Physics Publishing.

Bedau, M. and Brown, C. T. (1999). Visualizing evolutionary activity of genotypes. *Artif. Life*, 5:17–35.

Berry, G. and Boudol, G. (1992). The chemical abstract machine. *Theor. Comput. Sci.*, 96(1):217–248.

Beyer, H.-G. and Schwefel, H.-P. (2002). Evolution strategies - a comprehensive introduction. *Natural Computing*, 1(1):3–52.

Bhalla, U. S. and Iyengar, R. (1999). Emergent properties of networks of biological signaling pathways. *Science*, 283:381–387.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (2000). Inspiration for optimization from social insect behaviour. *Nature*, 406:39–42.

Bracht, A. and de P. Campello, A. (1979). Effect of the ionic strength on the kinetic properties of the mitochondrial L-malate dehydrogenase. *Experientia*, 35:1559–1561.

Buchholz, W. (1959). Fingers or fists? (the choice of decimal or binary representation). *Commun. ACM*, 2(12):3–11.

Callaway, D. S. and Perelson, A. S. (2002). HIV-1 infection and low steady state viral loads. *Bull. Math. Biol.*, 64:29–64.

Callaway, D. S., Ribeiro, R. M., and Nowak, M. A. (1999). Virus phenotype switching and disease progression in HIV-1 infection. *Proc. Biol. Sci.*, 266(1437):2523–2530.

Centler, F., Dittrich, P., Ku, L., Matsumaru, N., Pfaffmann, J., and Zauner, K.-P. (2003). Artificial life as an aid to astrobiology: Testing life seeking techniques. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Advances in Artificial Life, ECAL 2003*, volume 2801 of *LNAI*, pages 31–40. Springer, Berlin.

Cherry, C. (1966). *On Human Communication: A Review, a Survey, and a Criticism*, chapter 5. MIT Press, Cambridge, MA, 2nd edition.

Clarke, B. L. (1975). Theorems on chemical network stability. *The Journal of Chemical Physics*, 62(3):773–775.

Clarke, B. L. (1980). Stability of complex reaction networks. *Advances in Chemical Physics*, 42:1–213.

Conrad, M. (1988). The price of programmability. In Herken, R., editor, *The Universal Turing Machine: A Fifty Year Survey*, pages 285–307. Oxford University Press, New York.

Conrad, M. (1989). The brain-machine disanalogy. *BioSystems*, 22:197–213.

Conrad, M. (1990). Molecular computing. In Yovits, M. C., editor, *Advances in Computers*, volume 31, pages 235–324. Academic Press, Boston.

Conrad, M. (1992). The seed germination model of enzyme catalysis. *BioSystems*, 27:223–233.

Conrad, M. (1995). Scaling of efficiency in programmable and non-programmable systems. *BioSystems*, 35:161–166.

Csete, M. E. and Doyle, J. C. (2002). Reverse engineering of biological complexity. *Science*, 295(5560):1664–1669.

Culler, D., Estrin, D., and Srivastava, M. (2004). Overview of sensor networks. *Computer*, 37(8):41–49.

Deckard, A. and Sauro, H. (2004). Preliminary studies on the in silico evolution of biochemical networks. *ChemBioChem*, 5:1423–1431.

Dér, A. and Ramsden, J. J. (1998). Evidence for loosening of a protein mechanism. *Naturwissenschaften*, 85:353–355.

D'haeseleer, P., Forrest, S., and Helman, H. (1996). An immunological approach to change detection: Algorithms, analysis and implications. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 110–119. IEEE Computer Society Press.

Dittrich, P. (2001). *On Artificial Chemistries*. PhD thesis, University of Dortmund, Department of Computer Science, D-44221 Dortmund, Germany.

Dittrich, P. (2005). The bio-chemical information processing metaphor as a programming paradigm for organic computing. In Brinkschulte, U., Becker, J., Hochberger, C., Martinetz, T., Müller-Schloer, C., Schmeck, H., Ungerer, T., and Würtz, R., editors, *ARCS '05 - 18th International Conference on Architecture of Computing Systems 2005*, pages 95–99. VDE Verlag, Berlin.

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

Dittrich, P., Hinze, T., Ibrahim, B., Lenser, T., and Matsumaru, N. (2007). Hierarchically evolvable components for complex systems: Biologically inspired algorithmic design. In Jost, J., Helbing, D., Kantz, H., and Deutsch, A., editors, *Proceedings of the European Conference on Complex Systems (ECCS2007)*, page 85. TU Dresden.

Dittrich, P. and Matsumaru, N. (2007). Organization-oriented chemical programming. In *7th International Conference on Hybrid Intelligent Systems (HIS)*, IEEE Conference Proceedings, pages 18–23. IEEE.

Dittrich, P. and Speroni di Fenizio, P. (2007). Chemical organisation theory. *Bull Math Biol*, 69(4):1199–1231.

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial chemistries - a review. *Artif. Life*, 7(3):225–275.

Dorigo, M., Caro, G. D., and Gambardella, L. (1999). Ant algorithms for discrete optimization. *Artif. Life*, 5(2):137–172.

Eaton, J. W. (2002). *GNU Octave Manual*. Network Theory Limited, Bristol, UK.

Ebenhöh, O., Handorf, T., and Heinrich, R. (2004). Structural analysis of expanding metabolic networks. *Genome Informatics*, 15(1):35–45.

Edwards, J. S. and Palsson, B. O. (2000). The escherichia coli mg1655 in silico metabolic genotype: Its definition, characteristics, and capabilities. *PNAS*, 97:5528–5533.

Ehrenfeucht, A., Harju, T., Petre, I., Prescott, D. M., and Rozenberg, G. (2003). Formal systems for gene assembly in ciliates. *Theoretical Computer Science*, 292(1):199–219.

Ellacott, S. and Bose, D. (1996). *Neural Networks: Deterministic Methods of Analysis*. International Thomson Computer Press.

Englard, S. and Siegel, L. (1969). Mitochondrial L-malate dehydrogenase of beef heart. In Lowenstein, J. M., editor, *Citric Acid Cycle*, volume XIII of *Methods in Enzymology*, pages 99–106. Academic Press, New York.

Feinberg, M. and Horn, F. J. M. (1974). Dynamics of open chemical systems and the algebraic structure of the underlying reaction network. *Chemical Engineering Science*, 29:775–787.

Fernando, C. and Rowe, J. (2007). Natural selection in chemical evolution. *J. Theor. Biol.*, 247(1):152–167.

Flower, R. J. (2003). The development of cox2 inhibitors. *Nat Rev Drug Discov*, 2(3):179–191.

Fontana, W. and Buss, L. W. (1994). 'The arrival of the fittest': Toward a theory of biological organization. *Bull Math Biol*, 56:1–64.

Fraley, C. and Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97:611–631.

Freire, E. (1998). Statistical thermodynamic linkage between conformational and binding equilibria. *Adv.in Prot. Chem.*, 51:255–279.

Furusawa, C. and Kaneko, K. (1998). Emergence of rules in cell society: Differentiation, hierarchy, and stability. *Bull. Math. Biol.*, 60:659–87.

Gardner, T. S., Cantor, C. R., and Collins, J. J. (1999). Construction of a genetic toggle switch in escherichia coli. *Nature*, 403:339–342.

Gelpí, G. L., Dordal, A., Montserrat, J., Mazo, A., and Cortés, A. (1992). Kinetic studies of the regulation of mitochondrial malate dehydrogenase by citrate. *Biochem. J.*, 283:289–297.

Giavitto, J.-L. and Michel, O. (2001). MGS: a rule-based programming language for complex objects and collections. In van den Brand, M. and Verma, R., editors, *Electr. Notes in Theor. Comput. Sci.*, volume 59. Elsevier Science Publishers.

Gooding, D. (1990). *Experiment and the Making of Meaning.* Kluwer Academic Publishers, Dordrecht.

Güdemann, M., Angerer, A., Ortmeier, F., and Reif, W. (2007). Modeling of self-adaptive systems with SCADE. In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pages 2922–2925. IEEE.

Guido, N. J., Wang, X., Adalsteinsson, D., McMillen, D., Hasty, J., Cantor, C. R., Elston, T. C., and Collins, J. J. (2006). A bottom-up approach to gene regulation. *Nature*, 439(7078):856–860.

Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.*, 9(2):159–195.

Hartley, R. V. L. (1928). Transmission of information. *Bell System Tech. J.*, 7:535–563.

Herman, T. (2003). Models of self-stabilization and sensor networks. In Das, S. R. and Das, S. K., editors, *IWDC*, volume 2918 of *LNCS*, pages 205–214. Springer, Berlin.

Hinkelmann, K. and Kempthorn, O. (1994). *Design and Analysis of Experiments, Volume 1, Introduction to Experimental Design.* Wiley, New York.

Hjelmfelt, A., Weinberger, E. D., and Ross, J. (1991). Chemical implementation of neural networks and turing machines. *Proc. Natl. Acad. Sci. USA*, 88:10983–10987.

Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., Goryanin, I. I., Hedley, W. J., Hodgman, T. C., Hofmeyr, J.-H., Hunter, P. J., Juty, N. S., Kasberger, J. L., Kremling, A., Kummer, U., Novère, N. L., Loew, L. M., Lucio, D., Mendes, P., Minch, E., Mjolsness, E. D., Nakayama, Y., Nelson, M. R., Nielsen, P. F., Sakurada, T., Schaff, J. C., Shapiro, B. E., Shimizu, T. S., Spence, H. D., Stelling, J., Takahashi, K., Tomita, M., Wagner, J., and Wang, J. (2003). The systems biology markup language (sbml): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531.

Ihaka, R. and Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314.

Ikeda, M., Kamei, S., and Kakugawa, H. (2002). A space-optimal self-stabilizing algorithm for the maximal independent set problem. In *Proceedings of the Third International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pages 70–74.

Jain, S. and Krishna, S. (2001). A model for the emergence of cooperation, interdependence, and structure in evolving networks. *Proc. Natl. Acad. Sci. U. S. A.*, 98(2):543–547.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabási, A.-L. (2000). The large-scale organization of metabolic networks. *Nature*, 407:651–654.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, 22:437–467.

King, J. (1965). *Practical Clinical Enzymology.* D. Van Nostrand, London.

King, R. D., Whelan, K. E., Jones, F. M., Reiser, P. G. K., Bryant, C. H., Muggleton, S. H., Kell, D. B., and Oliver, S. G. (2004). Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427:247–252.

Kitano, H. (2002). Systems biology: A brief overview. *Science*, 295:1662–1664.

Klamt, S. and Gilles, E. D. (2004). Minimal cut sets in biochemical reaction networks. *Bioinformatics*, 20(2):226–234.

Klemm, K. and Bornholdt, S. (2005). Topology of biological networks and reliability of information processing. *Proc. Natl. Acad. Sci. U.S.A.*, 102(51):18414–9.

Krüger, B. and Dressler, F. (2004). Molecular processes as a basis for autonomous networking. In *International IPSI-2004 Stockholm Conference: Symposium on Challenges in the Internet and Interdisciplinary Research (IPSI-2004 Stockholm).*

Kulkarni, D. and Simon, H. A. (1990). Experimentation in machine discovery. In Shrager, J. and Langley, P., editors, *Computational Models of Scientific Discovery and Theory Formation*, pages 255–273. Morgan Kaufmann Pubishers, San Mateo, CA.

Küppers, B.-O. (1990). *Information and the Origin of Life.* MIT Press, Cambridge, MA.

Langley, P., Simon, H. A., Bradshaw, G. L., and Zytkow, J. M. (1987). *Scientific discovery: Computational exploration of the creative processes.* MIT Press, Cambridge, MA.

Langton, C. G. (1989). Artificial life. In Langton, C. G., editor, *Artificial Life*, SFI Studies in the Science of Complexity, pages 1–45. Santa Fe Institute, Los Alamos, New Mexico, Addison-Wesley, Redwood City, CA.

Lautenbach, K. (1973). Exact liveness conditions of a petri net class (in german). GMD Report 82, GMD, Bonn, German.

Lenser, T., Hinze, T., Ibrahim, B., and Dittrich, P. (2007). Towards evolutionary network reconstruction tools for systems biology. In E. Marchiori, J.H. Moore, J. R. E., editor, *Proceedings of the Fifth European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBIO)*, volume 4447 of *LNCS*.

Lenser, T., Matsumaru, N., Hinze, T., and Dittrich, P. (2008). Tracking the evolution of chemical computing networks. In Bullock, S., Noble, J., Watson, R. A., and Bedau, M. A., editors, *Proceedings of the Eleventh International Conference on Artificial Life*. MIT Press, Cambridge, MA.

Lindley, D. V. (1956). On a measure of the information provided by an experiment. *Ann. Math. Statist.*, 27:986–1005.

Lodding, K. N. (2004). The hitchhiker's guide to biomorphic software. *Queue*, 2(4):66–75.

Luby, M. (1986). A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15:1036 – 1055.

Machné, R., Finney, A., Müller, S., Lu, J., Widder, S., and Flamm, C. (2006). The sbml ode solver library: a native api for symbolic and fast numerical analysis of reaction networks. *Bioinformatics*, 22(11):1406–1407.

Matsumaru, N., Centler, F., Speroni di Fenizio, P., and Dittrich, P. (2006a). Chemical organization theory applied to virus dynamics. *it - Information Technology*, 48(3):154–160.

Matsumaru, N., Centler, F., Speroni di Fenizio, P., and Dittrich, P. (2007). Chemical organization theory as a theoretical base for chemical computing. *International Journal of Unconventional Computing*, 3(4):285–309.

Matsumaru, N., Centler, F., Zauner, K.-P., and Dittrich, P. (2004). Self-adaptive scouting - autonomous experimentation for systems biology. In Raidl, G. R., Cagnoni, S., Branke, J., Corne, D. W., Drechsler, R., Jin, Y., Johnson, C., Machado, P., Marchiori, E., Rothlauf, F., Smith, G. D., and Squillero, G., editors, *Applications of Evolutionary Computing, EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC Coimbra, Portugal, April 2004, Proceedings*, volume 3005 of *LNCS*, pages 52–62. Springer, Berlin.

Matsumaru, N., Colombano, S., and Zauner, K.-P. (2002). Scouting enzyme behavior. In Fogel, D. B., El-Sharkawi, M. A., Yao, X., Greenwood, G., Iba, H., Marrow, P., and Shackleton, M., editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 19–24. IEEE Press, Piscataway, NJ.

Matsumaru, N., Speroni di Fenizio, P., Centler, F., and Dittrich, P. (2005). A case study of chemical organization theory applied to virus dynamics. In Kim, J. T., editor, *Systems Biology Workshop at ECAL 2005, Workshop Proceedings CD-ROM*, Kent, UK.

Matsumaru, N., Speroni di Fenizio, P., Centler, F., and Dittrich, P. (2006b). On the evolution of chemical organizations. In Artmann, S. and Dittrich, P., editors, *Explorations in the complexity of possible life: abstracting and synthesizing the principles of living systems, Proceedings of the 7th German Workshop of Artificial Life*, pages 135–146. Aka, Berlin.

Mayer, B. and Rasmussen, S. (2000). Dynamics and simulation of micellar self-reproduction. *Int. J. Mod. Phys. C*, 11(4):809–826.

McMullin, B. (2000). John von Neumann and the evolutionary growth of complexity: Looking backwards, looking forwards... In Bedau, M. A., McCaskill, J. S., Packard, N. H., and Rasmussen, S., editors, *Artificial Life VII*. MIT Press, Cambridge, MA.

Montresor, A. and Babaoglu, O. (2003). Biology-inspired approaches to peer-to-peer computing in bison. In *The Third International Conference on Intelligent System Design and Applications*.

Müller-Schloer, C. (2004). Organic computing: On the feasibility of controlled emergence. In *Proceedings of the 2nd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS2004*, pages 2–5. ACM Press, New York.

NASA Astrobiology Roadmap (November 2002). Astrobiology roadmap. U.S. National Aeronautics and Space Administration (NASA). Available at: `http://astrobiology.arc.nasa.gov/roadmap/roadmap.pdf`.

Nowak, M. A. and Bangham, C. R. M. (1996). Population dynamics of immune responses to persistent viruses. *Science*, 272(5258):74–79.

Perelson, A. S., Neumann, A. U., Markowitz, M., Leonard, J. M., and Ho, D. D. (1996). Hiv-1 dynamics in vivo: Virion clearance rate, infected cell life-span, and viral generation time. *Science*, 271(5255):1582–1586.

Petri, C. A. (1962). *Kommunikation mit Automaten*. PhD thesis, Universität Bonn.

Pfaffmann, J. O. and Zauner, K.-P. (2001). Scouting context-sensitive components. In Keymeulen, D., Stoica, A., Lohn, J., and Zebulum, R. S., editors, *The Third NASA/DoD workshop on Evolvable Hardware*, pages 14–20, Long Beach, California. Jet Propulsion Laboratory, California Institute of Technology, IEEE Computer Society.

Păun, G. (2000). Computing with membranes. *J. Comput. Syst. Sci.*, 61(1):108–143.

Rechenberg, I. (1994). *Evolutionsstrategie'94*. frommann-holzboog, Stuttgard.

Reddy, V. N., Mavrovouniotis, M. L., and Liebman, M. N. (1993). Petri net representations in metabolic pathways. In *Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology*, pages 328–336. AAAI Press.

Reichenbach, F., Bobek, A., Hagen, P., and Timmermann, D. (2006). Increasing lifetime of wireless sensor networks with energy-aware role-changing. In *Proceedings of the 2nd IEEE International Workshop on Self-Managed Networks, Systems & Services (SelfMan 2006)*, pages 157–170, Dublin, Ireland.

Sahle, S., Gauges, R., Pahle, J., Simus, N., Kummer, U., Hoops, S., Lee, C., Singhal, M., Xu, L., and Mendes, P. (2006). Simulation of biochemical networks using copasi - a complex pathway simulator. In *Winter Simulation Conference, 2006. WSC 06. Proceedings of the*, pages 1698–1706.

Schilling, C. H., Schuster, S., Palsson, B. O., and Heinrich, R. (1999). Metabolic pathway analysis: Basic concepts and scientific applications in the post-genomic era. *Biotechnol. Prog.*, 15:296–303.

Schoonderwoerd, R., Bruten, J. L., Holland, O. E., and Rothkrantz, L. J. M. (1996). Ant-based load balancing in telecommunications networks. *Adapt. Behav.*, 5(2):169–207.

Schuster, S., Fell, D. A., and Dandekar, T. (2000). A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks. *Nat. Biotechnol.*, 18:326–332.

Shannon, C. E. and Weaver, W. (1949). *The Mathematical Theory of Communication.* University of Illinois Press, Urbana. Reprinted 1967.

Shukla, S. K., Rosenkrantz, D. J., and Ravi, S. S. (1995). Observations on self-stabilizing graph algorithms for anonymous networks. In *Proceedings of the Second Workshop on Self-Stabilizing Systems*, pages 7.1–7.15.

Sipper, M., Sanchez, E., Mange, D., Tomassini, M., Pérez-Uribe, A., and Stauffer, A. (1997). A phylogenetic, ontogenetic, and epigenetic view of bio-inspired hardware systems evolutionary computation. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 1(1):83–97.

Sols, A. (1981). Multimodulation of enzyme activity. *Current Topics in Cellular Regulation*, 19:77–101.

Speroni di Fenizio, P. and Dittrich, P. (2002). Artificial chemistry's global dynamics. movement in the lattice of organisation. *The Journal of Three Dimensional Images*, 16(4):160–163.

Speroni di Fenizio, P., Dittrich, P., Ziegler, J., and Banzhaf, W. (2000). Towards a theory of organizations. In *German Workshop on Artificial Life (GWAL 2000), in print*, Bayreuth, 5.-7. April, 2000.

Steels, L. (1990). Towards a theory of emergent functionality. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 451–461, Cambridge, MA, USA. MIT Press.

Stelling, J., Sauer, U., Szallasi, Z., Doyle, F. J., and Doyle, J. (2004). Robustness of cellular functions. *Cell*, 118:675–685.

Teuscher, C., Mange, D., Stauffer, A., and Tempesti, G. (2003). Bio-inspired computing tissues: towards machines that evolve, grow, and learn. *Biosystems*, 68:235–244.

Tsuda, S., Aono, M., and Gunji, Y.-P. (2004). Robust and emergent *p*hysarum logical-computing. *Biosystems*, 73(1):45–55.

Tu, P. N. V. (1994). *Dynamical systems : an introduction with applications in economics and biology.* Springer, Berlin, 2 edition.

Tyson, J. J., Chen, K., and Novak, B. (2001). Network dynamics and cell physiology. *Nature Reviews Molecular cell biology*, 2:908–916.

Weicker, K. and Weicker, N. (1999). On evolution strategy optimization in dynamic environments. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress on Evolutionary Computation*, volume 3, pages 2039–2046, Mayflower Hotel, Washington D.C., USA. IEEE Press.

Weiss, R., Homsy, G., and Knight, T. (1999). Toward in vivo digital circuits. In *Proceedings of the Dimacs Workshop on Evolution as Computation*.

Wodarz, D. and Nowak, M. A. (1999). Specific therapy regimes could lead to long-term immunological control of HIV. *Proc. Natl. Acad. Sci. U.S.A.*, 96(25):14464–9.

Wong, P. C.-P. and Smith, A. F. (1976). Assay of serum NAD-dependent malate dehydrogenase using malate as substrate. *Clinica Chemica Acta*, 72:409–412.

Wu, H., Ruan, P., Ding, A. A., Sullivan, J. L., and Luzuriaga, K. (1999). Inappropriate model-fitting methods may lead to significant underestimates of viral decay rates in hiv dynamic studies. *J. Acquir. Immune Defic. Syndr.*, 21(5):426–428.

Yung, Y. L. and DeMore, W. B. (1999). *Photochemistry of Planetary Atmospheres*. Oxford University Press, New York.

Zauner, K.-P. (2005). From prescriptive programming of solid-state devices to orchestrated self-organisation of informed matter. In Banâtre, J.-P., Giavitto, J.-L., Fradet, P., and Michel, O., editors, *Unconventional Programming Paradigms: International Workshop UPP 2004*, volume 3566 of *LNCS*, pages 47–55. Springer, Berlin.

Zauner, K.-P. and Conrad, M. (2001). Enzymatic computing. *Biotechnology Progress*, 17(3):553–559.

About The Author

Curriculum Vitae

Name	Naoki Matsumaru
Citizenship	Japan
Date of birth	28. August, 1974 (Shinshiro, Aichi, Japan)

Contact:	Biosystemanalyse
	Institut für Informatik
	D-07743 Jena, Germany
	(+49) 3641-9-46461
	`E-Mail:naoki@minet.uni-jena.de`
	`URL:www.minet.uni-jena.de/~naoki/`

Home address:	Sophienstr. 27, D-07743 Jena, Germany (+49)-3641-597275

Education

- M.S. in Computer Science: *Automated Protein Exploration for a Biologically Inspired Molecular Information Processor*, Advisor: Prof. Klaus-Peter Zauner, Wayne State University, Detroit, MI, USA, May 2002.
- B.S. in Computer Science and Engineering, Advisor: Prof. Kshirasager Knaik, University of Aizu, Fukushima, Japan, March 1998.

Employment

- August 2000 – May 2001: Graduate Teaching Assistant, Department of Computer Science, Wayne State University
- August 2001 – December 20001: Graduate Teaching Assistant, Department of Computer Science, Wayne State University
- January 2002 – September 2002: Graduate Research Assistant (funded by NASA), Department of Computer Science, Wayne State University
- 2002 October – 2005 August : wissenschaftlicher Mitarbeiter (research associate) (funded by BMBF), Department of Mathematics and Computer Science, Friedrich-Schiller-Universität Jena
- 2005 September – present : wissenschaftlicher Mitarbeiter (research associate) (funded by DFG - SPP 1183 *Organic Computing*), Department of Mathematics and Computer Science, Friedrich-Schiller-Universität Jena

Research interests: Chemical computing, Molecular computing, Computational models of biological information processing The role of information in living matter Bio-inspired computing Biophysics

Autobiographical Statement

Naoki Matsumaru received his Bachelor of Science, supervised by Prof. Kshirasagar Naik, with a major in computer science and engineering from the University of Aizu, Fukushima, Japan, in March, 1998. His thesis for the degree was to compare wireless communication network strategies using object-oriented programming with Java. Subsequently, he went to the United States to pursue his interest in the interface of nature and technology by enrolling in chemical engineering at Oklahoma State University, Stillwater, Oklahoma. There he learned about Dr. Michael Conrad's research on biological information processing. He traveled to Wayne State University, Detroit, Michigan, in 1999 to learn from Dr. Michael Conrad. Numerous discussions with him and his colleagues were enjoyable and inspiring, and the author took all the classes Dr. Conrad offered until his death in December, 2000. The author continued his study with the help of Dr. Klaus-Peter Zauner, one of Dr. Conrad's colleagues, by joining a NASA funded project to develop a molecular controller for a robot. He achieved his Master of Science in Computer Science in May, 2002, (advisor: Prof. Dr. Klaus-Peter Zauner, Wayne State University, Detroit). The title of his thesis is "Automated Protein Exploration for a Biologically Inspired Molecular Information Processor" (successfully defended March 29th, 2002), and he developed a system to autonomously explore protein behaviors. When he had an opportunity to join the Bio Systems Analysis group led by PD Dr. Peter Dittrich, at Friedrich-Schiller University Jena, Jena, Germany (funded by BMBF - Federal Ministry of Education and Research), he was excited about the possibilities to pursue further his research interest in biological information processing and chemical computing. There, he extended his expertise in autonomous experimentations, and simultaneously, he became familiar with Dr. Dittrich's theory of chemical organizations. He was fascinated by the theory and its potential application to chemical programming. When Dr. Dittrich decided to apply for a project under the Organic Computing initiative funded by DFG - German Research Foundation, the author supported his decision and also contributed to the application. The acceptance of the project application motivated the author to investigate further in chemical computing.

The author is married since 2003, and in 2007, his first daughter was born.

Publications

——— 1998 ———

- N. Matsumaru, K. Naik, and D. S. L. Wei. Comparing three location management strategies for tracking mobile systems. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications 1998 (PDPTA 1998)*, 1998.

——— 2002 ———

- N. Matsumaru, S. Colombano, and K.-P. Zauner. Scouting enzyme behavior. In D. B. Fogel, M. A. El-Sharkawi, X. Yao, G. Greenwood, H. Iba, P. Marrow, and M. Shackleton, editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 19–24. IEEE Press, Piscataway, NJ, 2002.

—————— 2003 ——————

- N. Matsumaru, S. Ziagos, F. Centler, K.-P. Zauner, S. Wölfl, and P. Dittrich. Autonomous exploration of dynamic biological systems by scouting. In H.-W. Mewes, V. Heun, D. Frishman, and S. Kramer, editors, *Proceedings of German Conference on Bioinformatics, Volume II*, page 210. Belleville Verlag Michael Farin, München, oct 2003.

- F. Centler, P. Dittrich, L. Ku, N. Matsumaru, J. Pfaffmann, and K.-P. Zauner. Artificial life as an aid to astrobiology: Testing life seeking techniques. In W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, editors, *Advances in Artificial Life, ECAL 2003*, volume 2801 of *LNAI*, pages 31–40. Springer, Berlin, 2003.

—————— 2004 ——————

- N. Matsumaru, F. Centler, K.-P. Zauner, and P. Dittrich. Autonomous experimentation for systems biology. Poster abstract presented at BioPerspectives 2004, Wiesbaden, Germany, may 2004.

- N. Matsumaru, F. Centler, K.-P. Zauner, and P. Dittrich. Towards applied systems biology: application centered models from autonomous experimentation. 5th International Conference on Systems Biology (ICSB), Heidelberg, Germany, oct 2004. Poster abstract.

- N. Matsumaru, F. Centler, K.-P. Zauner, and P. Dittrich. Self-adaptive scouting - autonomous experimentation for systems biology. In G. R. Raidl, S. Cagnoni, J. Branke, D. W. Corne, R. Drechsler, Y. Jin, C. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G. D. Smith, and G. Squillero, editors, *Applications of Evolutionary Computing, EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC Coimbra, Portugal, April 2004, Proceedings*, volume 3005 of *LNCS*, pages 52–62. Springer, Berlin, 2004.

—————— 2005 ——————

- N. Matsumaru, F. Centler, and P. Dittrich. Chemical organization theory as a theoretical base for chemical computing. In C. Teuscher and A. Adamatzky, editors, *Proceedings of the 2005 Workshop on Unconventional Computing: From Cellular Automata to Wetware*, pages 75–88. Luniver Press, Beckington, UK, 2005.

- N. Matsumaru, P. Speroni di Fenizio, F. Centler, and P. Dittrich. A case study of chemical organization theory applied to virus dynamics. In J. T. Kim, editor, *Systems Biology Workshop at ECAL 2005, Workshop Proceedings CD-ROM*, Kent, UK, 5-9 September 2005.

—————— 2006 ——————

- N. Matsumaru, F. Centler, P. Speroni di Fenizio, and P. Dittrich. Chemical organization theory applied to virus dynamics. *it - Information Technology*, 48(3):154–160, 2006.

- N. Matsumaru, P. Speroni di Fenizio, F. Centler, and P. Dittrich. On the evolution of chemical organizations. In S. Artmann and P. Dittrich, editors, *Explorations in the complexity of possible life: abstracting and synthesizing the principles of living systems, Proceedings of the 7th German Workshop of Artificial Life*, pages 135–146. Aka, Berlin, 2006.

- N. Matsumaru and P. Dittrich. Organization-oriented chemical programming for the organic design of distributed computing systems. In *1st international conference on bio inspired models of network, information and computing systems (BIONETICS)*, volume 275 of *ACM International Conference Proceeding*, Cavalese, Italy, December 11-13 2006. IEEE. also available at `http://www.x-cd.com/bionetics06cd/`.

———— 2007 ————

- N. Matsumaru, F. Centler, P. Speroni di Fenizio, and P. Dittrich. Chemical organization theory as a theoretical base for chemical computing. *International Journal of Unconventional Computing*, 3(4):285–309, 2007.

- F. Centler, P. Speroni di Fenizio, N. Matsumaru, and P. Dittrich. Chemical organizations in the central sugar metabolism of escherichia coli. In *Mathematical Modeling of Biological Systems, Volume I.* A Birkhäuser book, 2007.

- N. Matsumaru, T. Lenser, T. Hinze, and P. Dittrich. Designing a chemical program using chemical organization theory. *BMC Systems Biology*, 1(Suppl 1):P26, 2007. from BioSysBio 2007: Systems Biology, Bioinformatics, and Synthetic Biology, Manchester, UK, 11-13 January 2007.

- N. Matsumaru, T. Lenser, T. Hinze, and P. Dittrich. Toward organization-oriented chemical programming: A case study with the maximal independent set problem. In F. Dressler and I. Carreras, editors, *Advances in Biologically Inspired Information Systems*, volume 69 of *Studies in Computational Intelligence*, pages 147–163. Springer, Berlin, 2007.

- P. Dittrich and N. Matsumaru. Organization-oriented chemical programming. In *7th International Conference on Hybrid Intelligent Systems (HIS)*, IEEE Conference Proceedings, pages 18–23. IEEE, 17-19 Sept. 2007 2007.

- T. Hinze, R. Faßler, T. Lenser, N. Matsumaru, and P. Dittrich. Effizient chemisch rechnen durch deterministische reaktionssysteme mit regelpriorisierung. In M. Droste and M. Lohrey, editors, *Proceedings of 17. Theorietag Automaten und Formale Sprachen*, pages 68–73. Universität Leipzig, 2007.

- T. Hinze, S. Hayat, T. Lenser, N. Matsumaru, and P. Dittrich. Hill kinetics meets p systems: A case study on gene regulatory networks as computing agents in silico and in vivo. In G. Eleftherakis, P. Kefalas, and G. Păun, editors, *Proceedings of the Eight Workshop on Membrane Computing (WMC8)*, pages 363–381. SEERC Publishers, 2007.

- T. Hinze, S. Hayat, T. Lenser, N. Matsumaru, and P. Dittrich. Hill kinetics meets p systems. In G. Eleftherakis, P. Kefalas, G. Păun, G. Rozenberg, and A. Salomaa, editors, *Membrane Computing*, volume 4860 of *LNCS*, pages 320–335. Springer Verlag, 2007.

- P. Dittrich, T. Hinze, B. Ibrahim, T. Lenser, and N. Matsumaru. Hierarchically evolvable components for complex systems: Biologically inspired algorithmic design. In J. Jost, D. Helbing, H. Kantz, and A. Deutsch, editors, *Proceedings of the European Conference on Complex Systems (ECCS2007)*, page 85. TU Dresden, 2007.

———— 2008 ————

- T. Hinze, S. Hayat, T. Lenser, N. Matsumaru, and P. Dittrich. Biosignal-based computing by ahl induced synthetic gene regulatory networks. In P. Encarnacao and A. Veloso, editors, *Proceedings of the First International Conference on Bio-Inspired Systems and Signal Processing (BIOSIGNALS2008)*, volume 1, pages 162–169. IEEE Engineering in Medicine and Biology Society, Institute for Systems and Technologies of Information Control and Communication (INSTICC) press, 2008.

- T. Lenser, N. Matsumaru, T. Hinze, and P. Dittrich. Tracking the evolution of chemical computing networks. In S. Bullock, J. Noble, R. A. Watson, and M. A. Bedau, editors, *Proceedings of the Eleventh International Conference on Artificial Life*. MIT Press, Cambridge, MA, 2008.

- T. Hinze, R. Faßler, T. Lenser, N. Matsumaru, and P. Dittrich. Event-driven metamorphoses of P systems. In *Proceedings of 9th Workshop on Membrane Computing, WMC9*, LNCS. Springer, Berlin, 2008. (accepted).