On the Emergence and Evolution of Artificial Cell Signaling Networks

James Decraene

Artificial Life Laboratory,
Research Institute for Networks and Communication Engineering,
School of Electronic Engineering, Dublin City University, Dublin, Ireland
james.decraene@eeng.dcu.ie

Abstract. This PhD project is concerned with the evolution of Cell Signaling Networks (CSNs) in silico. CSNs are complex biochemical networks responsible for the coordination of cellular activities. We are investigating the possibility to build an evolutionary simulation platform that would allow the spontaneous emergence and evolution of Artificial Cell Signaling Networks (ACSNs). From a practical point of view, realizing and evolving ACSNs may provide novel computational paradigms for a variety of application areas. This work may also contribute to the biological understanding of the origins and evolution of real CSNs.

1 Introduction

Biological cells may respond to their environment in multiple ways, examples are: cellular differentiation, growth, division, death etc. To respond appropriately to environmental conditions, cells have to integrate multiple internal and external signals. Cell Signaling Networks (CSNs) are responsible for relaying and integrating these signals within the cell [19, 16, 21]. Through evolution, CSNs have become highly efficient at governing critical cellular activities which ensure the adaptivity and survival of the organism.

The purpose of modeling and evolving these natural networks is manifold, a variety of applications may be distinguished in the following areas:

1. *Synthetic Biology*: From a theoretical point of view it allows the exploration of network structures and dynamics, to find emergent properties [4] or to explain the organization and evolution of networks. From a practical point of view, studying CSNs in silico allows one to carry out experiments that would not be possible due to financial or technical constraints. Pharmaceutical applications can also be identified such as in drug design [23, 14].
2. *Computer Science/Engineering*: As signal processing systems, CSNs can be regarded as special purpose computers [5]. In contrast to conventional silicon-based computers, the computation in CSNs is not realized by electronic circuits, but by chemically reacting molecules in the cell. Realizing and evolving *Artificial* Cell Signaling Networks (ACSNs) may provide new computational

paradigms for a variety of application areas (e.g. signaling processing, computation or control systems). These new computational approaches may benefit from desirable properties found in biological processes such as robustness, self-organization or adaptation.

This PhD project adheres to the second area of interest presented above: we are interested in building novel computational paradigms based on Artificial Cell Signaling Networks. Through the use of evolutionary computing techniques, we allow ACSNs to spontaneously emerge and adapt to the environment. The current biological understanding provided guiding points that directed the design of our Artificial Chemistry: the Molecular Classifier Systems (MCS). Preliminary studies demonstrated that real and artificial CSNs could be considered for computational and engineering purposes [22, 26, 6].

Our PhD project is part of the ESIGNET project [1], an European funded project that aims to investigate the possibility to computationally evolve and simulate ACSNs by means of Evolutionary Computation techniques. One requirement of this EU project is to maintain biological plausibilities, in the sense that the interactions between the simulated artificial molecules are to be realistic with respect to the chemical interactions found in real CSNs.

2 Research

As an abstraction of real CSNs, ACSNs are differentiated and simplified by some key properties. The selection of these particular characteristics is motivated by the will to employ ACSNs for computational purposes. Four research issues are distinguished and presented:

1. *Computation*: CSNs are usually treated in an aggregate manner, where the information is carried by molecular concentration. We may also consider the finer grained behaviors of individual molecules that are computational in nature. An enzyme can be regarded as carrying out pattern matching to identify and bind target substrates, and then executing a discrete computational operation in transforming these into the product molecule(s). This approach differs from traditional rewriting systems: operation is stochastic rather than deterministic, secondly, operation is reflexive in the sense that molecules can function as both rules (enzymes) and as messages (substrates/products).
2. *Evolution*: Due to the intricate and unpredictable nature of molecular interactions occurring in CSNs, designing ACSNs by hand may result in a challenging task. Artificial evolution may suggest that within suitable conditions, effective ACSNs (meeting some given performance objectives) may be designed through evolutionary processes [6, 18].
3. *Crosstalk*: This designates the phenomenon where signals from different pathways become mixed together. In traditional engineering, crosstalk is regarded as a defect that has the potential to cause system malfunction.

[1] http://www.esignet.net

Crosstalk occurs very naturally in CSNs due to the fact that common molecules from different pathways may share the same physical reaction space (the cell). However, in the case of CSNs, crosstalk also has additional potential functionalities, which may actually be constructive [2, 25].

4. *Robustness*: In order to ensure the correct functioning of biochemical networks, it is argued that key properties of these networks are to be robust [3]. Alon et al. demonstrated from studying E. coli chemotaxis that molecular interactions can exhibit robustness [1, 22]. A robust biochemical network is able to reach a steady state that is equivalent to the state observed before a perturbation occurs. Such properties are highly desirable in dynamic engineered systems when subjected to internal and external uncertainty and perturbation.

2.1 Goals

Given the above research issues of interest, we distinguished for the present the following goals to be achieved in the course of this PhD project. Moreover, as this PhD project is tied to the ESIGNET project, some of these goals also contribute to different workpackages of the ESIGNET project:

1. *Modeling CSNs*: A good understanding of available modeling techniques of CSNs is necessary. This will guide and assist the design of our evolutionary simulation platform (in respect to biological plausibilities).
2. *Evolving CSNs*: Similarly, it is required to be acquainted with the current state of the art regarding the evolution of CSNs in silico. This includes an investigation on the possibility to evolve ACSNs for computational purposes.
3. *Design of our Artificial Chemistry*: We propose the Molecular Classifier System (MCS), an evolutionary simulation platform that will be employed to investigate the emergence and evolution of ACSNs from a bottom-up approach.
4. *Evolving ACSNs*:
   (a) We intend to evolve ACSNs with pre-specified constraints using the MCS, these constraints would be defined as minimal as possible in order to minimize the number of engineered components. Also these constraints would be defined so as to allow the emergence of ACSNs equivalent (based on some properties) to CSNs occurring in nature. A first experiment could be to simulate the bacterial chemotaxis phenomenom and then to study and compare the resulting ACSN with a real chemotaxis signaling pathway.
   (b) Further experiments would involve the investigation of other natural signaling pathways. Then, to employ this approach to solve computational problems. A study on the computational complexity of our novel paradigm will follow.
5. *Crosstalk*: To obtain a better understanding of the crosstalk phenomenon and more specifically about the positive and negative effects of crosstalk. We would like to see if it is possible to specify a network topology that allows optimal control of crosstalk effects.

6. *Robustness*: We will investigate the ability of ACSNs to create and sustain specific internal conditions such as homeostasis. We would like to exhibit such robust behavior in simulated ACSNs, and how through evolutionary changes, robustness can be refined. Another consequent issue is to quantify the robustness of such systems to external shocks and changes of conditions.
7. *Artificial Cognitive Systems*: We are also interested in examining the construction of Artificial Cognitive Systems based on our ACSNs, this approach would then adhere to the emergent system paradigm of cognition [24].
8. *Insights in Theoretical Biology*: Ultimately, contributions in the biological understanding of the origins and evolutionary dynamics of real CSNs will be proposed.

2.2 Current status

At this stage of the project, we have conducted a comprehensive literature review on the modeling and evolving of CSNs. The design of the Molecular Classifier System was proposed, finally some experimental studies were performed:

1. *Modeling CSNs*:
   - We contributed to the realization of the state of the art report on Cell Signaling Networks [13]. This report provides a thematic bibliography involving studies from different scientific fields: Biology, Mathematics, Computer Science and Engineering.
   - In [8] we described a comprehensive survey on the different philosophies (Mathematics, Statistics and Computer Science) to model biochemical networks. This work was carried out in collaboration with the bio-analysis research group (an ESIGNET partner) from the Friedrich-Schiller-Universitat in Jena-Germany.
   - A technical report *Introducing Computational Modeling of Cell Signaling Networks* is in preparation and intends to thoroughly present a selection of CSN modeling techniques.
   - A journal paper *Towards a unified approach for the modeling, analysis and simulation of cell signaling networks* is also in preparation, this is again performed in collaboration with the Jena group. This paper presents a comprehensive evaluation of modeling techniques and bridges between the differing approaches.
2. *Evolving CSNs*:
   We presented several posters presenting our preliminary work on the evolution and applications of ACSNs [11, 10, 20].
3. *Design of our Artificial Chemistry*:
   - In [12] we presented our concept of ACSNs and of MCS. An extended version of this paper is in preparation and will be included in *Advances in Biologically inspired information systems: models, methods and tools*: the best paper issue of the Bionectics'06 conference.

- In [9] , the broadcast language is examined: this is a programing formalism devised by Holland which shares some key properties with the MCS [17]. This system was investigated to provide complementary insights for the design of the MCS. Moreover, this work provided an evaluation of the broadcast language to modeling biochemical networks. A derivation of this formalism is proposed and includes the MCS main concepts.
- A technical report [7] was produced and presents our implementation of the Holland broadcast language, this was necessary as no published detailed specification of the language can be found in the literature.
- A presentation *Towards the design of Molecular Classifier Systems* was given at the bio-analysis group in Jena. This talk presented a summary of our work regarding the MCS and the broadcast language and how it is possible to combine both approaches.

2.3 Future planning and study

From January 2007, we intend to carry out experimental studies using our MCS. We will first investigate the emergence of ACSNs that would be equivalent (based on some defined properties) to real signaling networks. At a later stage (Fall 2007), this work will then focus on the use of ACSNs to solve a variety of computational problems. Throughout this study, we are interested in the evolution/growth of complexity of ACSNs when applied to different biological/computational problems. Following this, we will explore the possibility of building Artificial Cognitive Systems based on Artificial CSNs. Ultimately, this understanding on ACSNs will then be re-applied in Theoretical Biology, contributions may be given on the origins and evolutionary dynamics of real CSNs. Our PhD project started on December 2005 and is funded by the European Union for 3 years. Thus our current plan is to complete this PhD within 3 years. However completing a PhD in 3 years is quite challenging, an alternative would be to perform most of the experimental/analytical work within 3 years, the fourth year would then be dedicated to the writing up of the thesis.

3 Results

In this section we introduce the concept of Molecular Classifier Systems. Then we present the broadcast language which was proposed by Holland in 1975 and is similar to the MCS on many aspects. Evaluating the broadcast language provided us with valuable insights for the design of the MCS in order to implement ACSNs.

3.1 The Molecular Classifier System

We define the Molecular Classifier System (MCS) as a class of string-rewriting based Artificial Chemistries. This approach is inspired by Hollands Learning

Classifier Systems (LCS). In Hollands LCS, a demarcation is distinguished between *rules* and *messages*, however as mentioned earlier operations in a biochemical networks are intrinsically *reflexive*. The MCS addresses this issues by removing this rules/messages demarcation found in the LCS.

The behavior of the condition/binding properties and action/enzymatic functions is specified by a "chemical" language defined in the MCS. The chemical language defines and constrains the complexity of the chemical reactions that may be represented and simulated with the MCS. For example, a MCS model using a limited number of computational functions may only fatefully represent very simplistic chemical reactions.

In the MCS approach, a reaction between molecules may only occur if the informational string of a first molecule satisfies/binds with the conditional part of a second molecule. The second molecule may be the same as the first molecule leading to self-binding. The condition part refers to the binding properties of a molecule whereas action refers to the computational ("enzymatic") function. This pattern matching occurring implies a notion of *specificity* or "binding strength". A molecule having a high specificity would have less chance to react with another one. Whereas a molecule having a low specificity is likely to bind to another more often ($\approx$ chemical kinetics).

When two molecules can bind and consequently react to each other, the action part of one of the molecules is used to carry out the enzymatic operations upon the binding molecule (substrate). This operation results in producing another offspring (product). The symbols contained in the MCS action part are processed in a sequential order (parsed from left to right). The outcome (product) of the reaction depends on the nature of the symbols' functionality.

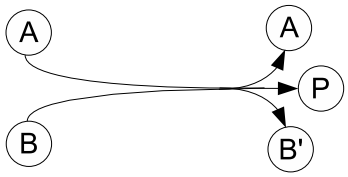

Fig. 1. Schematic of a reaction in the MCS: When a molecule A can react with a molecule B, the action statement of molecule A is "executed" upon the informational string of the binding molecule B. A is viewed as an enzyme and B as a substrate, thus A's structure is not affected by the reaction whereas B's structure is degraded and a product P is generated. A's action statement operators take as inputs the symbols of B's string. An offspring molecule P is generated as a result of these operations

The definitive set of operations is still under investigation as we are trying to understand what are the minimal operational requirements to allow a primitive ACSN to spontaneously emerge. However in the remainder of this section, we present a candidate solution based on a variant of the Holland broadcast language.

3.2 The Broadcast Language

The broadcast language is a programing formalism introduced by Holland in 1975 [17, 7], which can be thought of as the precursor for the LCS. A key property shared between the MCS and the broadcast language is the removal of any demarcation between messages and rules. A second beneficial property is the ability of the broadcast language to provide a straightforward representation to a variety of natural models such as Genetic Regulatory Network models.

The broadcast language basic components are called *broadcast units* which can be viewed as condition/action rules. Whenever a broadcast unit conditional statement is satisfied, the action statement is executed. This means that whenever a broadcast unit detects in the environment the presence of (a) specific signal(s), including themselves, then the broadcast unit would broadcast an output signal.

Some broadcast units may broadcast a signal that may constitute a new broadcast unit. Similarly, a broadcast unit can be interpreted as a signal detected by another broadcast unit. Broadcast units may also process a given signal, in the sense that, a broadcast unit may output a signal that is some modification of the detected/input signal. As a result, a broadcast unit may create new broadcast units or detect and modify an existing broadcast unit. A set of broadcast units, combined as a string, designates a *broadcast device*.

Table 1. Comparison of biological and broadcast language terminology

Biology	Broadcast Language
sequence of amino acids from $\{A, R, N, D, C, E, \ldots\}$	string of symbols from $\Lambda = \{0,\ 1,\ *,\ :,\ \Diamond,\ \nabla,\ \blacktriangledown,\ \triangle,\ p,\ '\}$
substrate	input signal
product	output signal
protein with no enzymatic function	null unit
enzyme	broadcast unit
protein complex	broadcast device
cellular milieu	list of strings from Λ

As a summary, the above table presents a comparison between the biological and the broadcast language terminology.

3.3 Methodology

In this section we present our implementation of the broadcast language system. This work was also motivated by the fact that, although described by Holland, no implementation / further studies on the broadcast language was available in the literature.

The Broadcast Language: syntax and semantics. The broadcast language alphabet Λ is finite and contains ten *symbols*, Λ^* is the set of strings over Λ. The symbols constitute the atomic elements of the language.

$$\Lambda = \{0,\ 1,\ *,\ :,\ \Diamond,\ \triangledown,\ \blacktriangledown,\ \triangle,\ p,\ '\}$$

Let I be an arbitrary string from Λ^*, in I, a symbol is said to be quoted if it is preceded by a symbol $'$. The finite collection of broadcast devices can be described by its *state* S at each timestep t. Four types of broadcast unit can be distinguished, any broadcast units that do not follow one of the four schemes (see below) are null units. Broadcast units may engage in the following interactions based on discrete timesteps:

1. $*I_1 : I_2$ If a signal of type I_1 is detected at time t then the signal I_2 is broadcast at time $t + 1$.
2. $* : I_1 : I_2$ If there is no signal of type I_1 present at time t then the signal I_2 is broadcast at time $t + 1$.
3. $*I_1 :: I_2$ If a signal of type I_1 is detected at time t then a *persistent* string of type I_2 (if any) is removed from the environment at the end of time t.
4. $*I_1 : I_2 : I_3$ If a signal of type I_1 and a signal of type I_2 are both present at time t then the signal S_3 is broadcast *at the same time* t unless the string I_3 contains unquoted symbols $\{\triangledown, \blacktriangledown, \triangle\}$ or singly quoted occurrence of $*$, in which case the string I_3 is broadcast a time $t + 1$.

The interpretation of each symbol in Λ is now presented:

$\{0, 1\}$ 0 and 1 are the basic elements to specify a signal. A string such as 010110 can be regarded as the signature of a particular signal. This signature can be employed by a broadcast unit to detect and identify a signal.

$*$ This symbol indicates that the subsequent *symbols* until the next unquoted $*$ (if any) are to be interpreted as a broadcast unit. If a broadcast device I does not contain any unquoted $*$ then I is a null unit.

: This symbol is used as a punctuation mark to differentiate the arguments of a broadcast unit. The symbol : (position and frequency) determines the type of the broadcast unit as presented earlier.

$\Diamond$ When this symbol is met in the input argument of a broadcast unit, it indicates that a signal detected by the broadcast unit may present any symbol at this position without affecting its acceptation or rejection by the broadcast unit. Also if $\Diamond$ occurs at the rightmost position of an input argument, then $\Diamond$ acts as a multiple character *wildcard*.

$\triangledown$ When this symbol occurs in the input and output arguments of a broadcast unit, it designates any arbitrary initial (prefix) or terminal (suffix) strings of symbols. This allows one to pass a string of symbols from the input signal to the broadcast signal ($\approx$ unit processing).

$\blacktriangledown$ This symbol is similar to $\triangledown$ but can also concatenate different inputs signals.

$\triangle$ This symbol is employed in the same manner as $\triangledown$ and $\blacktriangledown$ but designates an arbitrary *single* symbol whose position can be anywhere in the argument of a given broadcast unit.

p When this symbol occurs at the first position of a string, it designates a *persistent* string. This string would then persist over time until it is deleted, even if the string is not an active broadcast unit. A null device occurring at time t which is not persistent exists only for one timestep and is removed at the end of time t.

$'$ This symbol is used to *quote* a symbol in the arguments of a broadcast unit. When a symbol is said to be quoted, it acts as a simple literal, i.e. a $'\triangle$ would only match $\triangle$.

The Broadcast Language: implementation. In our implementation of the Holland broadcast language, we distinguish three main classes: `Env` represents the environment, this object holds a list of all current existing devices. The class `BDevice` designates a broadcast device, an instantiation of `BDevice` may hold from 0 to n `BUnit` objects. The `BUnit` class refers to a broadcast unit, it may contain one or two argument(s) and an output signal, all represented by strings of characters. In this system based on discrete timesteps, the sequential operation is as follows. At timestep t, all broadcast devices including null devices are stored in a vector of devices S. This vector is held by an instance of `Env`. A vector of character strings A is used to hold signals (strings) to be added to S at the beginning of t. At time $t = 0$, S is empty and A represents the initial set of broadcast devices. D is a vector of strings holding signals to be removed from S at the end of timestep t. An overview of the system from its initialization to its termination is given:

1. *Initialization*: an `Env` object is instantiated, vectors S, A and D are created and are empty by default.
2. *Environmental signals*: at this step, input signals (strings of character) given by the environment are added to set A. At time $t = 0$, the input signals correspond to the initial set of signals. A *detector* may be built to probe the environment and insert new signals into set A.
3. *Transferring signals from set A to S*: signals contained in set A are inserted in set S. Set A is then flushed. Each signal inserted in S is processed into broadcast devices (`BDevice` objects); if a signal generates an active broadcast device then this broadcast device is parsed into broadcast units (`BUnit` objects).
4. *Processing signals in S*: this step is broken up into two sequential subprocesses:
   (a) we first look for broadcast units of type 4 that are able to broadcast at the same time t. If those broadcast units can be satisfied by other signals then they broadcast their output signals. The latter output signals are then inserted into S. As these newly inserted signals may satisfy other similar broadcast units, it is necessary to repeat this process until no new signal gets inserted into S.
   (b) Then each broadcast device in S is processed in a sequential order: if a broadcast device I is active then each broadcast unit I_i contained

in I may broadcast its output signal upon detecting adequate signals. A broadcast unit which has already been activated at time t may not broadcast again within that timestep, under any circumstances. Output signals broadcast by type $1, 2$ and 4 broadcast units are stored in A. If a type 2 broadcast unit is activated then its output signal is inserted into set D. Finally, if a broadcast device I is a null device and is not a persistent signal, then this device signal is added to set D.

5. *Delete signals from sets S and D*: for each signal I_d contained in set D, if there is a signal of the form I_d present in S then this signal is deleted from S. If there are n signals in S that are of the form I_d then only one of those signals is deleted (selected at random). D is then flushed.
6. *Termination condition*: If this user-defined termination condition is not satisfied then the system returns to step 1.

The above implementation addresses and clarifies a number of ambiguities that had been left open by Holland.

3.4 Experiments

In this section we present a case study where we use the broadcast language to model a signal transduction network which was previously modeled with the aid of a Boolean network [15], see Fig. 2. With the Boolean abstraction, a molecule is considered as a logical expression having two different possible states: *on* or *off*, meaning that the molecule is present in the environment or not.

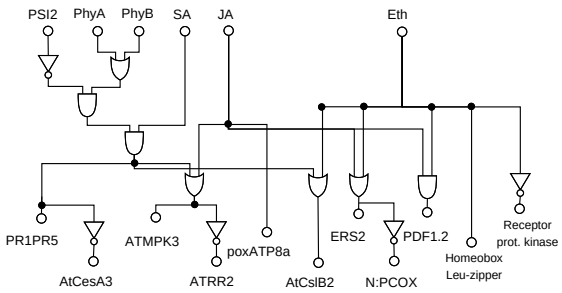

Fig. 2. Boolean representation of the signal transduction network controlling the plants defense response against pathogens.

We use the broadcast language to mirror the Boolean network of the biochemical network presented in Figure 2. To accomplish this, we proceed to a direct mapping of each Boolean function to broadcast devices. Using this model, one may determine the states of the output molecules according to the states of the input molecules.

We first represent each molecule (substrate) $PhyA$, $PhyB$, Eth, etc., with a string (signal) such as $p0000000$, $p0000001$, $p0000010$, etc. We then define the broadcast devices (enzymes) which enable the reactions to occur in this network.

$$(PR1PR5) = (\neg PSI2 \wedge (PhyA \vee PhyB)) \wedge SA \tag{1}$$

The above equation describes the state of $PR1PR5$ according to the states of $PSI2$, $PhyA$, $PhyB$ and SA. We now present how to express this Boolean expression using broadcast devices, see Table 2.

Table 2. Broadcast devices employed to express Eq. 1

Broadcast device
I_1 $*p000000\Diamond : 1000000$
I_2 $* : p0000010 : 1000001$
I_3 $*1000000 : 1000001 : 1000010$
I_4 $*p0000011 : 1000011$
I_5 $*1000010 : 1000011 : 1000100$
I_6 $*1000100 : p0000101$

- In order to represent an `OR` gate that takes for input signals $PhyA$ and $PhyB$ we generate I_1, which indicates that whenever persistent signals $p0000000$ or $p0000001$ ($PhyA$ or $PhyB$) are detected, the signaling molecule 1000000 is broadcast. This example also demonstrates how to represent *crosstalk* phenomena in the broadcast language.
- The `NOT` gate is expressed through the use of a type 2 broadcast unit. To represent `NOT` $p0000010$ ($PSI2$), we define I_2 which stipulates that when no persistent $PSI2$ molecule is present then the signaling molecule 1000001 is broadcast at time $t + 1$.
- The expression (($p0000000$ `OR` $p0000001$) `AND` (`NOT` $p0000010$)) is designated by I_3 which would broadcast 1000010 only if 1000000 *and* 1000001 are detected.
- The detection of 1000000 indicates that either $p0000000$ ($PhyA$) or $p0000001$ ($PhyB$) is present. Secondly, detecting 1000001 implies that $p0000010$ ($PSI2$) has not been detected.
- The broadcast device I_4 is used to broadcast a signaling molecule 1000011 if $p0000011$ (SA) is detected. I_5 is similar to I_3 and represents an `AND` gate taking into account the results of I_3 and I_4.
- This broadcast device, if satisfied, broadcasts a signaling molecule that is employed to activate $PR1PR5$ ($p0000101$), as shown in I_6.

The whole Boolean network may be built following the above described method. This case study was implemented with our system and tested against a selection of inputs, and the outputs reacted precisely in accordance with the boolean functions specified by the network.

3.5 Fusing MCS and the Broadcast Language

We demonstrated that the broadcast language can model Genetic Regulatory Networks (GRNs). This was due to the ability of the broadcast language to mirror Boolean networks which illustrates its wide ranging processing power. Nevertheless, it was also highlighted that the broadcast language is limited regarding the representation and simulation of CSNs [9]. To address this issue, we propose to combine the MCS concept with the broadcast language in a new system termed "MCS.b". The MCS.b complements the broadcast language (syntax and semantics) and extends it by including the following refinements:

- Instead of processing all broadcast devices sequentially and deterministically during a time step, the MCS.b processes as follows: at each time step t, we pick n pairs of broadcast devices at random. For each pair of devices, one of the broadcast devices is designated (at random) as the *catalyst device* and the second one as the *substrate device*. If the conditional statement of the catalyst device is satisfied by the signal of the substrate device, then the action statement of the catalyst device is executed upon the substrate device.
- n designates the number of pairs of broadcast devices that will interact during a timestep. It is also plausible to consider n as the temperature in real chemistry. Temperature has an important role in chemical reactions, indeed molecules at higher temperature have a greater probability to collide with one another.
- In the broadcast language specification given by Holland, additional rules were required to resolve some ambiguities raised by the interpretation of broadcast devices. To facilitate this, the MCS.b simplifies the interpretation of broadcast units by preserving broadcast units of type 1 only.
- Similarly the notion of non-persistent devices is removed: by default all devices are considered as persistent molecules.
- As type 3 broadcast units and non-persistent devices no longer exist in this proposal, no molecule can be deleted from the population. However the deletion of molecules is needed to obtain evolutionary pressure. Our suggestion is as follows: each time two molecules react together, we pick a molecule at random and delete it from the population.

By combining the strength of both the MCS and broadcast language, we expect the MCS.b to be capable of modeling, simulating and evolving ACSNs in a more fateful manner. At present, we have conducted a number of preliminary experiments examining the spontaneous emergence of collective autocatalytic sets among others. This was expected to be trivial as this phenomenon was already demonstrated with other relateds Artificial Chemistries (Tierra, Alchemy, etc.). Initial results suggest that the MCS.b performs as expected, however before these results can be presented to the research community, validation against empirical biological data is required.

4 Achievements

During this first year of PhD, we carried out a comprehensive literature review on the representation and simulation of CSNs. This review allowed us to obtain a global understanding on the area of CSNs, moreover this work provided us with guiding points for the design of the MCS. We then examined an approach similar to the MCS: the Holland broadcast language. We demonstrated the modeling of a simple signaling pathway using the broadcast language, but this work highlighted the limitations of this formalism. As a result, we presented some refinements of the broadcast language that would result in a platform combining some of the MCS concepts and of the broadcast language. Although our resulting MCS.b will require further evaluation to precisely represent real biochemical networks, this system combining the MCS original concept and the broadcast language approach allow for the implementation of an evolutionary simulation platform to study *artificial* biochemical networks *in silico*.

5 Feedback

We would be very grateful for any suggestions and criticisms regarding our work, we are continuously looking for novel computational techniques and concepts that may contribute to our project. Future issues include the topic of Crosstalk and Robustness, any comments or suggestions concerning these research topics would be very welcomed. Since Crosstalk and Robustness imply notions from the networking and engineering fields, insights from these disciplines would be of great interest.

Acknowledgement: This work was funded the European Integrated Project ESIGNET in the EU FP6 NEST Initiative (contract no. 12789).

References

1. U. Alon, M.G. Surette, N. Barkai, and S. Leibler. Robustness in bacterial chemotaxis. *Nature*, 397(6715):168–171, January 1999.
2. A.M. Arias and P. Hayward. Filtering transcriptional noise during development: concepts and mechanisms. *Nature Reviews Genetics*, 7(1):34–44.
3. N. Barkai and S. Leibler. Robustness in simple biochemical networks. *Nature*, 387(6636):913–917, June 1997.
4. U.S. Bhalla and R. Iyengar. Emergent properties of networks of biological signaling pathways. *Science*, 283(5400):381–387, Jan 1999.
5. D. Bray. Protein molecules as computational elements in living cells. *Nature*, 376(6538):307–312, Jul 1995.
6. A. Deckard and H.M. Sauro. Preliminary studies on the in silico evolution of biochemical networks. *Chembiochem*, 5(10):1423–1431, October 2004.
7. J. Decraene. The Holland Broadcast Language. Technical Report ALL-06-01, Artificial Life Lab, RINCE, School of Electronic Engineering, Dublin City University, 2006.

8. J. Decraene, P. Dittrich, T. Hinze, T. Lenser, B. McMullin, and G. G. Mitchell. A multidisciplinary survey of modeling techniques for biochemical networks, 2006. Integrative Post-Genomics IPG'06, Lyon (F), November-December.
9. J. Decraene, G. G. Mitchell, and B. McMullin. The Broadcast Language and the Modeling of Biochemical Networks. In *Genetic Programming, 10th European Conference, EuroGP2007, Valencia, Spain, April 11-13, 2007, Proceedings*, 2007. To appear.
10. J. Decraene, G.G. Mitchell, C. Kelly, and B. McMullin. An approach to evolving cell signaling networks in silico. In *Proceedings of the International Workshop on Systems Biology 2006 (IWSB 06)*, page 34, July 2006.
11. J. Decraene, G.G. Mitchell, C. Kelly, and B. McMullin. Evolving Artificial Cell Signaling Networks. In *Proceedings of the European Conference of Complex Systems ECCS '06, Oxford, England*, September 2006.
12. J. Decraene, G.G. Mitchell, and B. McMullin. Evolving Artificial Cell Signaling Networks using Molecular Classifier Systems. In *Proceedings of the first International Conference on Bio-Inspired Models of Network, Information and Computing Systems (BIONETICS 06)*, December 2006.
13. ESIGNET. Cell signalling networks: State of the art report. Technical report, EU ESIGNET FP6-NEST-2003-1-12789 report Deliverable D3.1, March 2006.
14. N.J. Eungdamrong and R. Iyengar. Modeling cell signaling networks. *Biol Cell*, 96(5):355–362, Jun 2004.
15. T. Genoud, B. Marcela, S.C. Trevino, and J-P. Mtraux. Numeric simulation of plant signaling networks. *Plant Physiology*, 126:1430–1437, August 2001.
16. E.J.M. Helmreich. *The Biochemistry of Cell Signalling*. Oxford University Press, USA, 2001.
17. J.H. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA, 1992.
18. J.H. Holland. Exploring the evolution of complexity in signaling networks. *Complexity*, 7(2):34–45, 2002.
19. T. Hunter. Signaling–2000 and beyond. *Cell*, 100(1):113–127, Jan 2000. Historical Article.
20. C. Kelly, J. Decraene, G.G. Mitchell, B. McMullin, and D. OBrien. Cellular computation using classifier systems. In *Proceedings of the International Workshop on Systems Biology 2006 (IWSB 06)*, page 34, July 2006.
21. G. Krauss. *Biochemistry of Signal Transduction and Regulation*. John Wiley & Sons, 2003.
22. D.A. Lauffenburger. Cell signaling pathways as control modules: complexity for simplicity? *Proc. Natl. Acad. Sci. USA*, 97(10):5031–3, 2000.
23. K. Vancompernolle and P. Ball. Synthetic biology: Applying engineering to biology. Nest high-level expert group report, Published by the European Commission 2005, 2003. available at http://www.eurosfaire.prd.fr/7pc/bibliotheque/consulter.php?id=157.
24. D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *IEEE Transactions on Evolutionary Computation*, 2006.
25. D. Volfson, J. Marciniak, W.J. Blake, N. Ostroff, L.S. Tsimring, and J.Hasty. Origins of extrinsic variability in eukaryotic gene expression. *Nature*, December 2005.
26. T.M. Yi, Y. Huang, M.I. Simon, and J. Doyle. Robust perfect adaptation in bacterial chemotaxis through integral feedback control. *Proc Natl Acad Sci U S A*, 97(9):4649–4653, April 2000.