The Holland Broadcast Language and the Modeling of Biochemical Networks

James Decraene, George G. Mitchell, Barry McMullin, and Ciaran Kelly

Artificial Life Laboratory,
Research Institute for Networks and Communication Engineering,
School of Electronic Engineering, Dublin City University, Ireland
`{james.decraene,george.mitchell,mcmullin,ciaran.kelly}@eeng.dcu.ie`
`www.eeng.dcu.ie/~alife/`

Abstract. The Broadcast Language is a programming formalism devised by Holland in 1975, which aims at improving the efficiency of Genetic Algorithms (GAs) during long-term evolution. The key mechanism of the Broadcast Language is to allow GAs to employ an adaptable problem representation. Fixed problem encoding is commonly used by GAs but may limit their performance in particular cases. This paper describes an implementation of the Broadcast Language and its application to modeling biochemical networks. Holland presented the Broadcast Language in his book "Adaptation in Natural and Artificial Systems" where only a description of the language was provided, without any implementation. Our primary motivation for this work was the fact that there is currently no published implementation of the Broadcast Language available. Secondly, no additional examination of the Broadcast Language and its applications can be found in the literature. Holland proposed that the Broadcast Language would be suitable for the modeling of biochemical models. However, he did not support this belief with any experimental work. In this paper, we propose an implementation of the Broadcast Language which is then applied to the modeling of a signal transduction network. We conclude the paper by proposing that with some refinements it will be possible to use the Broadcast Language to evolve biochemical networks *in silico*.

Key words: Broadcast Language, adaptable representation, biochemical networks modeling

1 Introduction

Holland proposed the Broadcast Language so as to address some potential limitations in the application or performance of Genetic Algorithms (GAs) [6, 3]. Holland argued that GAs provide an efficient method of adaptation; however in the case of long-term adaptation, the efficiency of GAs could be limited by the representation used to encode the problem. In traditional GAs, this representation is fixed and may significantly influence the complexity of the fitness landscape. During long-term evolution this may limit the performance of the GA.

To overcome this limitation, Holland proposed to dynamically adapt the problem representation. Adapting the representation may then generate correlations between the problem representation and the GA performance.

Another feature discussed by Holland was the conjecture that the Broadcast Language is a Turing Complete programming language. If this is so, it would imply that the language would not dictate any long-term limits to its evolution. However although this issue clearly has intrinsic interest, it will not be consided further in the current paper.

Following this, Holland argued that the Broadcast Language would provide a straightforward representation for a variety of natural models such as Genetic Regulatory Networks or Neural Networks. This would show the computational power of the Broadcast Language and its capacity to adapt.

However, while recognising some of the potential merits of the Broadcast Language, we need to consider the fact that Holland did not support this approach with experimental evaluation; nor have we been able to identify any body of subsequently published work on the Broadcast Language in the literature.

We believe that there is a need for further investigations on the Broadcast Language because:

- The Broadcast Language may provide a useful framework for investigating a range of interesting problems in Evolutionary Computation and Theoretical Biology.
- The potentially interesting applications of the Broadcast Language were only outlined, not actually formally demonstrated, by Holland.
- Since Holland's early presentations [6], no further work on Broadcast Systems (Broadcast Language-based systems) can be found in the literature.

To initiate these further investigations we have implemented an execution platform for the Broadcast Language. We applied this to the study of the modeling of biochemical networks. This paper is organized as follows: we first introduce in more detail the Broadcast Language and then describe our implementation of the Broadcast System. We then demonstrate how to model a signal transduction network with the Broadcast Language. This is finally followed by a discussion of possible refinements toward the modeling of a specific problem instance: the evolution of biochemical networks *in silico*.

2 The Broadcast System

We use the formalism given by Holland in the original text [6]. We initially provide an overview of the Broadcast System and then present our implementation.

2.1 An overview

The Broadcast Language basic components are called *broadcast units* which are strings formed from the set $\varLambda = \{0,\ 1,\ *,\ :,\ \Diamond,\ \triangledown,\ \blacktriangledown,\ \vartriangle,\ p,\ '\}$. Broadcast units 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 one or more specific signal(s), possibly including the broadcast units themselves, then the broadcast unit would broadcast an output signal.

As an example, we may consider a given broadcast unit that upon detecting signals I_1 and I_2 would broadcast an output signal I_3. This is analogous to a biological phenomenon where an enzyme would form a product upon the binding of specific substrate(s) to its binding region(s). In this example an enzyme can be thought of as a broadcast unit, substrate(s) would be detected signal(s), the enzyme binding region(s) would refer to the broadcast unit condition part, the product is the output signal and finally the environment would be the reaction space (e.g., the cell).

Following the above analogy, a substrate can be degraded during catalysis. We implement this phenomenon through the signal processing ability of broadcast units. Indeed general signal processing can also be performed with broadcast units: e.g., a broadcast unit may detect a signal I and broadcast a signal I', so that I' is some modification of the signal I.

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. 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, is designated a *broadcast device*. A broadcast device can be viewed as analogous to a protein complex in which interactions between the several proteins result in complex functional behavior of the molecule.

Holland also described in detail how he distinguishes between four key *types* of broadcast unit, designated types 1, 2, 3 and 4. See [6, 2] for a detailed description of this and also the more general syntax and semantics of the Broadcast Language.

2.2 The system

In this section we present our implementation of the Holland Broadcast System. We have implemented the Broadcast System using an Object Oriented paradigm, in which we may 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.

Figure 2.2 presents an overview of the system from its initialization to its termination.

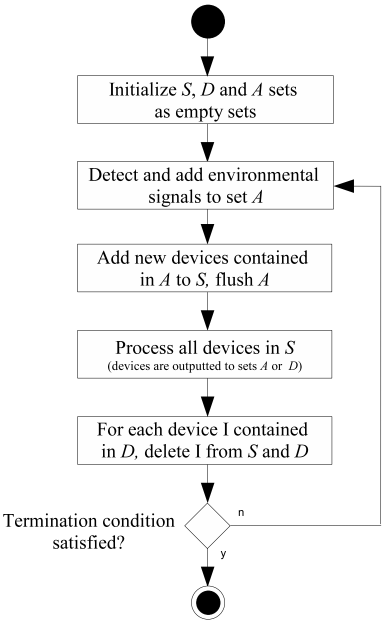

Fig. 1. Broadcast System flowchart

Following this, we discuss in detail each step presented in this diagram:

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 "external" 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 (see [6]) that are able to broadcast at the same time t. If those broadcast units can be satisfied by other signals (including themselves) then they broadcast their output signals. The latter output signals are then directly inserted into S. As these newly inserted signals may satisfy other similar type 4 broadcast units, it is necessary to repeat this whole process until no new signal gets inserted into S. This is the first subprocess to be performed because type 4 broadcast units may output signals that may contribute to other broadcast units contained in S at time t.
   (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 issued by type $1, 2$ and 4 broadcast units are stored in set 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 uniformly at random). D is then flushed.
6. *Termination condition*: this condition is set by the user, for example it may be an integer T indicating the maximum number of steps to be completed. If this user-defined termination condition is not satisfied then the system returns to step 1.

The above implementation addresses or clarifies a number of ambiguities that had been left open by Holland. We now show how the Broadcast Language is capable of modeling biochemical networks, which was one of Holland identified application areas.

3 Modeling a biochemical network

In this section we present a case study where we model a biochemical network with the Broadcast Language. We successfully model a signal transduction network, which was previously modeled with the aid of a Boolean network [9]. Note that this example given by Genoud only addresses the regulatory aspects of the signaling network.

One way to represent the regulatory aspects of a biochemical network is to use the Boolean formalism. With the Boolean abstraction, a (protein) molecule is considered as a logical expression having two different possible states. One possible state is the *on* state meaning that the molecule is present in the environment. To the contrary, when a molecule state is *off*, this indicates that this particular molecule is not present in the environment (cell). Figure 2 provides an example of a graphical boolean representation of a signal transduction network.

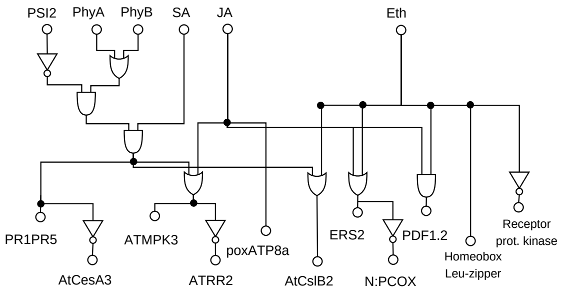

Fig. 2. Boolean representation of the signal transduction network controlling the plants defense response against pathogens.

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 Broadcast System 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. In this case, the broadcast devices stand for the boolean functions shown in Fig.2.

$$(PR1PR5) = (\neg PSI2 \wedge (PhyA \vee PhyB)) \wedge SA$$

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 the Broadcast Language.

In order to represent an `OR` gate that takes for input signals $PhyA$ and $PhyB$ we generate the following broadcast device:

$$I_1 = *p000000\Diamond : 1000000$$

This broadcast device 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 purpose of using signaling molecules will be shown in the description of the third broadcast device I_3.

The `NOT` gate is expressed through the use of type 2 broadcast unit. To represent `NOT` $p0000010$ ($PSI2$), the following broadcast device is defined:

$$I_2 = * : p0000010 : 1000001$$

The above broadcast device stipulates that when no persistent $PSI2$ molecule is present then the signaling molecule 1000001 is broadcast at time $t + 1$.

Following the given example, we want to express an `AND` gate. The expression (($p0000000$ `OR` $p0000001$) `AND` (`NOT` $p0000010$)) can be translated into the following broadcast device:

$$I_3 = *1000000 : 1000001 : 1000010$$

I_3 would broadcast 1000010 only if 1000000 *and* 1000001 are detected. The detection of 1000000 indicates that either $p0000000$ ($PhyA$) or $p0000001$ ($PhyB$) is on. Secondly, detecting 1000001 implies that $p0000010$ ($PSI2$) has not been detected.

$$I_4 = *p0000011 : 1000011$$

The broadcast device I_4 is used to broadcast a signaling molecule 1000011 if $p0000011$ (SA) is detected.

$$I_5 = *1000010 : 1000011 : 1000100$$

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 follows:

$$I_6 = *1000100 : p0000101$$

```
Case:    1                   2                   3                   4                   5
         [PhyA] off          [PhyA] on           [PhyA] on           [PhyA] off          [PhyA] on
         [PhyB] on           [PhyB] off          [PhyB] off          [PhyB] off          [PhyB] on
Inputs   [PSI2] on           [PSI2] on           [PSI2] on           [PSI2] off          [PSI2] on
         [SA] off            [SA] on             [SA] off            [SA] off            [SA] on
         [JA] on             [JA] off            [JA] off            [JA] off            [JA] on
         [ETH] on            [ETH] off           [ETH] on            [ETH] off           [ETH] on

         [PR1PR5] off        [PR1PR5] off        [PR1PR5] off        [PR1PR5] off        [PR1PR5] off
         [ATMPK3] on         [ATMPK3] off        [ATMPK3] off        [ATMPK3] off        [ATMPK3] on
         [AtCesa3] on        [AtCesa3] on        [AtCesa3] on        [AtCesa3] on        [AtCesa3] on
         [ERS2] on           [ERS2] off          [ERS2] on           [ERS2] off          [ERS2] on
         [N:PCOX] off        [N:PCOX] on         [N:PCOX] off        [N:PCOX] on         [N:PCOX] off
Outputs  [poxATP8a] on       [poxATP8a] off      [poxATP8a] off      [poxATP8a] off      [poxATP8a] on
         [ATRR2] off         [ATRR2] on          [ATRR2] on          [ATRR2] on          [ATRR2] off
         [AtCslB2] on        [AtCslB2] off       [AtCslB2] on        [AtCslB2] off       [AtCslB2] on
         [PDF1.2] on         [PDF1.2] off        [PDF1.2] off        [PDF1.2] off        [PDF1.2] on
         [HomLeuZip] on      [HomLeuZip] off     [HomLeuZip] on      [HomLeuZip] off     [HomLeuZip] on
         [RePrkinase] off    [RePrkinase] on     [RePrkinase] off    [RePrkinase] on     [RePrkinase] off
```

Fig. 3. A series of results obtained with our implementation of the Broadcast System. The Boolean network representation of the signal transduction network (Fig.2) was implemented with our system. A molecule is *on* when at least one occurrence of the corresponding broadcast device is found after time $t = 4$. These results present the states of the output molecules $PR1PR5$, $AtCesA3$, etc according to the differing states of the input molecules $PhyA$, $PhyB$, etc.

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, see Fig. 3. We may note that because some broadcast units broadcast at time $t+1$, a cascade of similar reactions may then take a certain amount of time steps to process the whole network. This is indeed necessary so that every boolean functions described in the model are processed. In the current example, 4 time steps are necessary to obtain the output states accounting for every boolean gates.

This example showed that the Broadcast Language is a straightforward method to model a biochemical network when the latter is described with a Boolean formalism. The same method could also be applied to represent other genetic regulatory networks as they can be modeled with Boolean networks [7].

4 Discussion

In the case study above, we demonstrated that the Broadcast Language can model Genetic Regulatory Networks (GRNs). The ability of the Broadcast Language to mirror Boolean networks illustrates the wide ranging processing power that Broadcast Systems are capable of.

A key advantage to using the Broadcast System, as mentioned by Holland, is the ability of the system to work in conjunction with GAs. By allowing the coupling of GAs with the Broadcast System, a variety of evolutionary operators (mutation, crossover etc) are accessible. With these operators it would be possible to design Broadcast Systems that model the evolution of GRNs.

Previous works on the modeling of the evolution of GRNs can be found in the literature [5, 1]. Nevertheless we believe that the study of the Broadcast Language would complement this understanding. As argued by Holland [6], One benefit is that with an adaptable representation the Broadcast Language would prevent evolutionary plateaus being encountered during the evaluation. In long term evolution, this may be of high significance as we commonly meet such plateaus in evolutionary systems [4, 8].

Although the modeling of the evolution of GRNs is valuable, we focus on a related, but currently not so well understood class of biochemical network, the Cell Signaling Networks (CSNs).

In the case study we provided above, we presented a CSN model where only the regulatory aspects of the CSN were covered. Although this qualitative approach is of interest, this significantly limits the power of Broadcast Systems to model biochemical networks. As currently defined, the Broadcast System cannot express concentration kinetics and it is well known that molecular concentrations play an important role in chemical reactions.

In order to refine the Broadcast Language we outline some refinements which focus on the following points:

- To incorporate chemical kinetics in Broadcast Systems.
- To strengthen the biological plausibilities in the modeling of CSNs with Broadcast Systems.

- To facilitate the evaluation of the Broadcast System.

Examining the first point we must consider collision theory: molecules must collide to react together. When the molecular concentration increases, the probability of collision increases as well. These collisions occur at random and are best described as Brownian motion. However simulating Brownian motion is computationally expensive. We approximate this phenomena in the Broadcast System by adjusting the way broadcast devices are processed:

- Instead of processing all broadcast devices sequentially during a time step, we propose the following: 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 number of pairs of broadcast devices is a constant and refers to 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 System, in order to increase the "temperature", one may increment the integer number n.

In order to improve the biological application of the system, and to facilitate its evaluation, the following refinements are proposed:

- 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, we suggest to simplify the nature 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 our 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.

The above suggestions simplify and strengthen the ability of Broadcast Systems to model biochemical networks. However to model precisely real biochemical networks, more attributes are needed to describe accurately these complex systems. A solution is to implement this derivation of the Broadcast System as an agent-based model, where the agents behavior and adaptation is determined by broadcast devices. This allows the definition of additional molecular properties (e.g., spatial location, state, etc) for each agent.

Although our proposed work will require further evaluation to precisely represent real biochemical networks, these refinements allow for the design of an evolutionary simulation platform to study *artificial* biochemical networks *in silico*.

5 Conclusion

In this paper we presented our implementation of the Holland Broadcast System and demonstrated the modeling of a signal transduction network with this approach. This work was motivated by the desire to implement the Holland system and also to apply it to biochemical networks modeling. We evaluated our implementation and showed that the Broadcast Language is suitable to model GRNs. We then discussed the benefits of Broadcast Systems to evolve GRNs through the use of GAs. Nevertheless it was shown later that Broadcast Systems are limited regarding the study of biochemical networks from a quantitative point of view. Following this, we proposed refinements that allow the Broadcast Language to model the evolution of biochemical networks accounting for the quantitative aspects. These refinements provide the following additional benefits: reinforcement of the biological applications of the system and facilitate its evaluation. This approach may contribute to the understanding of the evolutionary dynamics of biochemical networks.

Acknowledgement

This work was funded by ESIGNET (Evolving Cell Signaling Networks in Silico), an European Integrated Project in the EU FP6 NEST Initiative (contract no. 12789).

References

1. Stefan Bornholdt. Modeling Genetic Networks and Their Evolution: A Complex Dynamical Systems Perspective. *Biol. Chem*, 382(9):1289–1299, 2001.
2. James Decraene. The Holland Broadcast Language. Technical Report ALL-06-01, Artificial Life Lab, RINCE, School of Electronic Engineering, Dublin City University, 2006.
3. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.
4. Dominique Groß and Barry McMullin. The creation of novelty in artificial chemistries. In *ICAL 2003: Proceedings of the eighth international conference on Artificial life*, pages 400–409, Cambridge, MA, USA, 2003. MIT Press.
5. Jennifer Hallinan and Janet Wiles. Evolving genetic regulatory networks using an artificial genome. *Proceedings of the second conference on Asia-Pacific bioinformatics-Volume 29*, pages 291–296, 2004.
6. John H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, USA, 1992.
7. Stuart A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution.* Oxford University Press, May 1993.
8. Tim Taylor. Creativity in evolution: individuals, interactions, and environments. pages 79–108, 2002.
9. Marcela B. Trevino Santa Cruz Thierry Genoud and Jean-Pierre Mtraux. Numeric simulation of plant signaling networks. *Plant Physiology*, 126:1430–1437, August 2001.