Soft Comput (2014) 18:1899–1912
DOI 10.1007/s00500-013-1174-8

METHODOLOGIES AND APPLICATION

Exploring aspects of cell intelligence with artificial reaction networks

Claire E. Gerrard · John McCall · George M. Coghill · Christopher Macleod

Published online: 20 November 2013

Abstract The Artificial Reaction Network (ARN) is a Cell Signalling Network inspired connectionist representation belonging to the branch of A-Life known as Artificial Chemistry. Its purpose is to represent chemical circuitry and to explore computational properties responsible for generating emergent high-level behaviour associated with cells. In this paper, the computational mechanisms involved in pattern recognition and spatio-temporal pattern generation are examined in robotic control tasks. The results show that the ARN has application in limbed robotic control and computational functionality in common with Artificial Neural Networks. Like spiking neural models, the ARN can combine pattern recognition and complex temporal control functionality in a single network, however it offers increased flexibility. Furthermore, the results illustrate parallels between emergent neural and cell intelligence.

Keywords Artificial biochemical network (ABN) · Artificial chemistry · Artificial neural network (ANN) · Artificial reaction network (ARN) · Cell signalling network (CSN) · Robotics

Communicated by F. Marcelloni.

C. E. Gerrard (✉) · J. McCall
IDEAS Research Institute, Robert Gordon University, Aberdeen AB10 7GJ, UK
e-mail: c.e.gerrard@rgu.ac.uk

J. McCall
e-mail: j.mccall@rgu.ac.uk

G. M. Coghill
School of Computing Science, University of Aberdeen, Aberdeen AB24 3FX, UK
e-mail: g.coghill@abdn.ac.uk

C. Macleod
School of Engineering, Robert Gordon University, Aberdeen AB10 7GJ, UK
e-mail: chris.macleod@rgu.ac.uk

1 Introduction

In recent years, researchers have become increasingly interested in the complex behaviours displayed by individual cells. Such behaviours are exemplified by simple eukaryotic organisms called protists. These show an astonishingly varied repertoire of seemingly intelligent behaviours. For example, some have simple eye-spots to help avoid high light levels; others have locomotory appendages and stinging arrows to actively hunt and subdue their prey (Ford 2009). All this is accomplished without recourse to a neural network, the foundation-stone of intelligence in higher animals. The behaviour of such simple organisms may be labelled as "Cell Intelligence".

In order to generate this emergent high-level behaviour, a cell must be able to store and process information. Data is represented internally by a set of spatially distributed molecular concentrations. Cell Signalling Networks (CSNs) process this information within elaborate hierarchical network control structures which connect species together in productive or inhibitory unions. In this way, cells are able to respond to changes within their environment, communicate with other cells, and perform internal self maintenance operations (Bray 1995). Several researchers have highlighted the processing capabilities of these networks (Bray 1995; Arkin and Ross 1994; Bhalla 2003) and similarities to Artificial Neural Networks (ANNs) (Bray 1995; Bhalla 2003). For example, Bray (1995) claims that individual network units can perform Boolean and Fuzzy Logic and act as a Turing machine. In other work, Stadtman and Chock (1997) demonstrated that such a network can act as a flexible computational unit. Similar results were documented by Arkin and Ross (1994), and

recently a number of researchers have developed these ideas (Hild et al. 2010; Wang et al. 2011).

In this paper the properties and applications of a connectionist model inspired by CSNs termed the Artificial Reaction Network (ARN) are discussed. This representation was introduced previously (Gerrard et al. 2011) where it was used to create a simulation of the chemotaxis pathway of *Escherichia coli*. In later work biochemical network motifs were investigated as a means to perform computational processing in a single ARN based system (Gerrard et al. 2012a, b).

The ARN belongs to the branch of Artificial Life known as Artificial Chemistry Computing (ACC). This field of study utilizes principles of the Chemical Metaphor to construct novel software or hardware architectures in silico (Dittrich et al. 2001). In the Chemical Metaphor, data is stored in the form of molecular species and information processing occurs through interactions (reactions) between these molecules. The result of this computation appears as emergent global behaviour (Dittrich et al. 2001). ACC has been previously used in two main applications: simulating complex systems (biological, social or ecological) and in developing novel solutions to engineering or computational problems. Its approach can be broadly categorised into microscopic or macroscopic methods (Dittrich 2005). Microscopic methods treat each molecule explicitly, while in macroscopic methods, all the molecules of one type are represented by a value signifying, for example, concentration. Microscopic ACCs tend to model dynamics as stochastic molecular collisions, while macroscopic models tend to use continuous differential or discrete difference equations. The ARN is a macroscopic ACC and has a networked representation similar to other ACC models (Ziegler and Banzhaf 2000; Eikelder et al. 2009). For instance, in the Artificial Biochemical Neuron, concentrations of reactants form weighted links between reactions and their dynamics are modelled using Ordinary Differential Equations (ODEs) (Eikelder et al. 2009). Other related chemically inspired approaches can be found in the literature, for instance, the Gene Regulatory Network algorithm (Guo et al. 2009), the Digital Hormone System (Shen et al. 2004), the Artificial Homeostatic Hormone System (Hamann et al. 2010) and idiotypic Farmer based Artificial Immune Systems (Krautmacher and Dilger 2004). Like these models, the ARN represents molecular species as continuous concentrations where dynamics are modelled using ODEs. Its networked representation is specifically designed to represent "biological circuitry" and allows temporal and spatial dynamics to unfold real-time. As discussed later, it has properties in common with other models from both Artificial Intelligence and Systems Biology fields, including: Artificial Neural Networks, Random Boolean Networks, Petri Nets, and S-Systems.

The specific objectives of the results presented here are as follows. Firstly, to explore the mechanisms and computational properties that leads to emergent high-level behaviour in cells. Secondly, to further investigate applications of the ARN technique- specifically the control of motion in limbed robots. In this paper the following novel work is presented: (1) A complete overview of the ARN including its development, computational properties, advantages and disadvantages; and (2) the production of a complete ARN based control system for a limbed robot which combines pattern recognition and generation of time varying waveforms in a single network.

The paper is structured as follows: Sect. 2 discusses the ARNs development, representation, advantages and disadvantages. The ARN is then used to explore several computational aspects of Cellular Intelligence. The first of these is its pattern recognition capability (Sect. 3.1). In these experiments ARN parameters are set using a Genetic Algorithm (GA) and input patterns representing external environmental chemicals are mapped to output patterns. In Sect. 3.2, further processing capabilities of the ARN are investigated by determining its ability to regulate complex temporal dynamics. Its application in robotic control is then explored by using the resulting system to create waveforms which control the gaits of limbed robots. This network is then extended into a complete control system by combining it with the previous pattern recognition network (Sect. 3.3) in a single ARN. The results show that the ARN can function in both sensory input and motor output tasks which usually only more complex models can fulfil. Moreover the ARN offers increased flexibility over existing methods in robotic control tasks. The report concludes that the ARN is a versatile and powerful technique which has application in both simulation of chemical systems, and in robotic control, where it can offer a higher degree of flexibility and computational efficiency than benchmark alternatives. Furthermore, it provides a tool which may possibly throw further light on the origins and limitations of the primitive intelligence associated with cells and its parallels with neural intelligence.

2 The artificial reaction network

The ARN was briefly introduced in our previous work (Gerrard et al. 2011). This section provides a complete overview of the ARN starting with its basic formulation, and followed by its networked representation and computational properties.

2.1 Basic formulation

Rate equation models can be used to represent many different physical systems and so are very general and flexible in their applications. In the domain of chemistry, they can directly represent (or be slightly modified to represent) all

the common reaction types. They form the basis of S-systems (Savageau and Voit 1987) and are well characterised in biochemical simulations. The basic rate equation is described by two terms as given by Eq. (1):

$$\frac{d[P_j]}{dt} = k_f \prod_{n=1}^{N} [R_n]^{\alpha_n} - k_r \prod_{i=1}^{M} [P_i]^{\beta_i} \quad (1)$$

the first half corresponds to the rate of generation of product j (P_j) and is equal to the forward reaction rate (k_f), multiplied by the product of the concentrations of the N reactants ($[R_n]$), each raised to the power of its reaction order α_n. The second term represents the rate of decomposition of product back into its original reactants. This depends on the reverse reaction coefficient (k_r) multiplied by the product of the concentrations of the M products $[P_i]$, each raised to the power of its reaction order β_i. For example, consider the simple reaction between two reactants labelled A and B with reaction orders of q and s respectively. These produce a single product P. In this case, Eq. (1) is reduced to Eq. (2):

$$\frac{d[P]}{dt} = k_f[A]^q[B]^s - k_r[P] \quad (2)$$

when used in S-systems, a group of rate equations are normally set up — one for each reaction. The left hand of each equation is then set to zero and they are solved simultaneously to yield the steady-state response. If the dynamic responses are required, then numerical solution methods like Runge-Kutta are normally applied.

2.2 A networked representation

Clearly a large set of simultaneous Ordinary Differential Equations (ODEs), in their basic mathematical form, limit the conceptualisation, visualisation and communication of complex topologies. Furthermore, in this form, each ODE term is tightly coupled, and is difficult to isolate and manage. Therefore, in order to create a connectionist representation with distinct biological processing units, capable of constructing complex biological circuits, the method needs to be modified. This may be done by isolating each reaction in the network to form a discrete node which may then be modified independently of the other reactions. Such a node can be viewed as analogous to a neuron in an ANN and has been named an Artificial Reaction Node; by analogy networks of such nodes may be termed ARNs. Similarly to an ANN, each ARN node is a processing unit, transforming a number of inputs into an output. In an interconnected network of such units, global behaviour is determined by the complete set of connections, and unit parameters. Furthermore, by isolating each reaction like this, the individual pathways or units which make up the system can be changed, reconnected or evolved by (for instance) a genetic algorithm. This also allows an individual part of the network to be independently modified and its effects studied. Such a feature is useful in simulating disease pathways. Isolating the reactions in this way facilitates two other important practical advantages. Firstly, visual "drag and drop" interfaces can be developed. These allow researchers to quickly change network or reaction parameters in order to study their effect. This, in turn, allows simple visualisation of the system in a graphical form which makes its conceptualisation easier. Secondly, it makes the application of object-orientated programming techniques very simple, as each node can be coded as an instance of an object.

In developing the system described, Euler's method was chosen in order to solve the rate equations. This offers some advantages, in that it is simple and computationally cheap, but more importantly, it allows the whole network to run quickly in simulated real time—so that its temporal dynamics can be seen to unfold during a run. This gives the option of changing parameters in real time so that a user can observe any dynamic resulting behaviour. Furthermore, the temporal output of the network could potentially be used as a control system for an "artificial cell" robot—a cytobot.

Using the simple two input system shown in Eq. (2), multiplying through by dt and changing to a discrete finite time-step Δt, the Euler approximation is described by Eq. (3):

$$\Delta[P] = (k_f[A]^q[B]^s - k_r[P])\Delta t \quad (3)$$

this reaction needs to be isolated from the others, so that it can form a discrete "unit". This can be done most easily by borrowing the concept of "pools" from Petri-nets (Murata 1989). Petri-nets pass tokens between such pools as part of their operation. In the system discussed here, the pools may hold the number of available molecules, the concentration of the reacting chemicals (for example in moles per litre) or the mass of reactants. As the reaction proceeds, the reacting species pass from the input pools (depleting them) to the output pools (enriching them). So, in the previous example, to generate one molecule of product requires q molecules of reactant A and s molecules of reactant B. In this case, the pool containing A would get depleted by an amount ΔA as described by Eq. (4):

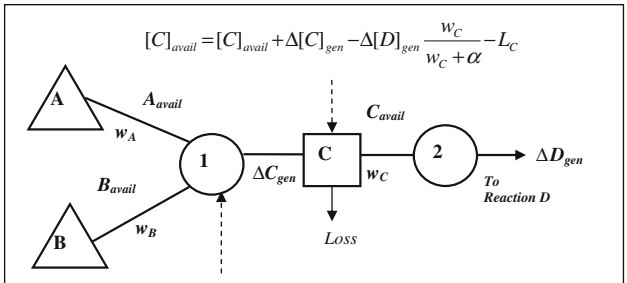
$$\Delta[A] = \Delta[P]q \quad (4)$$

where ΔP is the amount of product generated (which would be added to pool P). This equation works if the units used are number of molecules or moles per litre (which are not conserved quantities). However if mass or a similarly conserved quantity is used then ΔA is given by Eq. (5):

$$\Delta[A] = \Delta[P]\frac{q}{q+s} \quad (5)$$

the whole system using more general symbols is shown diagrammatically in Fig. 1 (for a conserved quantity).

It comprises a set of connected reaction nodes (circles), pools (squares), and inputs (triangles). Each pool represents



α equals sum of other weights connected to inputs of unit D

KEY (and for all other ARN diagrams):		w:	Reactant species order
Straight Line:	Excitatory connection	Triangle:	Input
T shaped line:	Inhibitory connection	Square:	Pool
A, B, C:	Reactant species	Circle:	Reaction unit
avail:	Available	gen:	Generated

Fig. 1 Schematic diagram of a simplified Artificial Reaction Network (ARN). Reactant chemicals A and B react at unit 1. The rate of the reaction at unit 1 at time t is given by Eq. (3). The current concentration in pool C is updated using Eq. (5)

the current available protein species concentration (*avail*) in a compartment and each circle corresponds to a reaction unit, representing an interaction (reaction) between a number of chemicals.

The use of pools allows current concentration of species and their dynamics to be simply viewed. As a biological modelling tool, chains of pools could be used to represent gradients and translocation of species across membranes. Similarly, a loss component can also be added to the pools to represent the destruction of reactants or products by specific or general proteases or other degradation routes as shown in Fig. 1.

Connections symbolize the flow of species into and out of reaction units and their weight (w) corresponds to reaction order. The connections can be either excitatory, or inhibitory. A reaction with both excitatory and inhibitory connections will proceed if all connected inhibitory pools are empty and its excitatory connected pools have the required concentrations. Thus the input pools serve as pre-conditions which must be met before the reaction can proceed. The inhibitory connections act as discrete on/off switches to either the forward or reverse reaction.

The ARN has been extensively verified against standard methods of representing chemical systems (Gerrard et al. 2011), where it was shown to provide the same degree of accuracy as other ODE models.

2.3 Computational properties

The overall structure may be compared to a perceptron, where the pools correspond to inputs, the reaction units to the weighted sum function, and these are joined together by weighted connections.

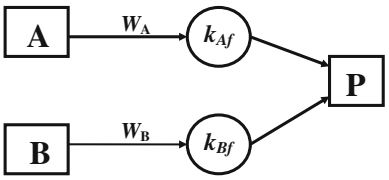


Fig. 2 A simple ARN network

It is fairly easy to see that the computational properties of the ARN are similar to those of the ANN. For example, consider the simple network shown in Fig. 2.

If we assume that the orders W_A and W_B are unity and the reverse reaction rates are zero, then the rate of change of the product pool P is given by Eq. (6):

$$[\dot{P}] = k_{Af}[A] + k_{Bf}[B] \tag{6}$$

which is the same expression as for the activity of a perceptron if A and B are the inputs and the k terms the weights. So a network of such nodes has at least the same computational capabilities as MLPs (Rumelhart and McClelland 1986). In fact the addition of non-unity orders means that effectively the node can produce non-linear separators in a similar way to polynomial neurons (Woo and Khor 2004) and are rather similar to so-called "sigma-pi" units (Gurney 1992)—although with the added dimension of dynamic behaviour which will be discussed in Sect. 3.2.

2.4 Disadvantages

There are some potential disadvantages associated with the ARN. Firstly, the Euler approximation has an associated cumulative error. This is because it is an iterative linear approximation to a complex function. It may be thought of as the first-order term of a Taylor expansion of the function. For example, if we say that the rate equation is a function of a set of reactants and products **R**, we could write an abbreviated version of Eq. (1) as given by Eq. (7):

$$\frac{d[P]}{dt} = f(\mathbf{R}) \tag{7}$$

the full Taylor series for Euler approximation to the second order would then be described by Eq. (8):

$$R_0 + f(\mathbf{R})\Delta t + \frac{\Delta t^2}{2!}\frac{d^2 f(\mathbf{R})}{dt^2} + \ldots \text{ and so on} \tag{8}$$

because the series is truncated to the linear term the error of the approximation is the sum of the missing terms. In reality the error contribution from successive terms is usually negligible providing that the step-size is small. The error may be of consequence if the user is trying to simulate a complex

biochemical system very accurately. However, as previously discussed, this is not the main purpose of the ARN.

Other difficulties can arise using hybrid models and detailed discussions are provided in the literature (Kowalewski 2002). Two such issues can occur in this representation where both produce an unnatural result due to the problem of trying to represent a discrete system (of individual molecules) by a continuous mathematical expression. In a real biological system there are a finite number of molecules and the chemistry acts the same way on all of these until they are exhausted. This however is not always the case when applying the governing equation (Eq. 3). When the current reactant concentration is above or equal to one the result is representative of the chemistry. However this is not the case if it is less than one. For example, consider a hypothetical non-reversible reaction where species A, with an order of 2, reacts to produce species B. In the case where the initial concentration of A equals 2, k_f equals 1 and Δt equals 1, then the result of Eq. (3) is 4. However, if the initial concentration of A is 0.5 the result of Eq. (3) is 0.25 rather than the expected value of 1. In practice this is easily sorted by restricting the range of the concentrations or using different units.

A similar issue occurs where a pool, for example S, inhibits a reaction unit by an inhibitory connection. This reaction will always be inhibited while there remains any amount of chemical in S. Meanwhile S is involved in another reaction where the resultant flux is depleting S at each time step. As the concentration of S decreases so too does the flux. This leads to an infinite sequence of decreasing concentrations of S which asymptotically approaches zero. Therefore, S will always contain a smaller but positive value and as a result the inhibited reaction can never occur. In reality this would not occur since individual molecules would react in an individual manner. This problem is solved by simply setting a threshold—if a pool concentration is less than the threshold its concentration is set to 0.

3 Experiments and results

In the following sections the methodology and results for the following experiments are presented: (1) An ARN based pattern recognition system; (2) the use of an ARN based system to regulate time varying waveforms and its application in control of limbed robotic gaits; and (3) an ARN which combines the previous networks into a complete quadrupedal robotic control system capable of recognising input patterns and generating the required gait response.

3.1 Pattern recognition

A key mechanism of cell intelligence is the ability of a cell to recognise and respond to specific patterns of chemical signals within its environment. Receptors recognise and bind to particular environmental chemicals. These are transduced and cell response is determined by a chain of signalling events.

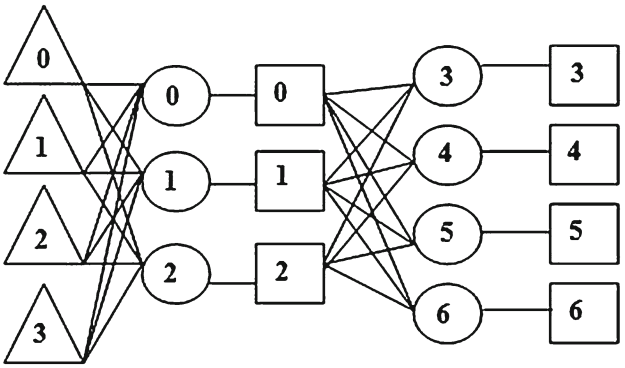


Fig. 3 The structure of the ARN used for pattern recognition experiments. The network consists of four inputs (*triangles*), seven reaction units (*circles*) and seven pools (*squares*). Each index of the input pattern array is fed into the corresponding input number. Output patterns are output at pools (*squares*) 3–6

The ARNs pattern recognition capability was tested in both the context of a general pattern recognition device and in an abstract biological setting. In each case, four separate patterns composed of four input and four associated output mass values were applied to the ARN. Each pattern comprised values of either 0.1, representing low concentration, or 1 corresponding to high concentration. The ARN was set up as shown in Fig. 3 and consists of seven pools, four inputs and seven reaction units organised into two layers. The network structure was based on that of standard (fully connected feedforward) Multilayer Perceptrons (MLPs) (Rumelhart and McClelland 1986). This is because MLPs are a benchmark connectionist method well known to facilitate pattern recognition.

In biological CSNs, network parameters are determined by genetic factors which are subject to evolution. To achieve a related effect within this artificial setting a genetic algorithm (GA) was adopted to train the network to produce the correct output. The initial value of all pools was 0.01. Each input value of a pattern was fed into its corresponding input unit. For example, the first, second and third input value of pattern 1 is 0.1 and the fourth is 1 (see Table 1)—thus input unit 0–3 (see Fig. 3) were initialised to 0.1 and input unit 3 to 1. The output values were generated by the final layer of pools (3–6). The target output values for each pattern are given under the heading "Output" of Table 1, and the actual values associated after training are given under "Actual Output". A population of 100 solutions were randomly initialised. Each solution comprised a complete set of network parameters including the forward and reverse rates for each unit and the weights for each connection between pools (or inputs) and units. Due to its temporal properties the network was run for 100 cycles

Table 1 Patterns and results for both general and abstract biological setting experiments

General pattern setting				Abstract biological setting			
Pattern	Input	Output	Actual output	Pattern	Input	Output	Actual output
1	0.1	0.1	0.1	1	1 (WR)	1 (IS)	1
	0.1	0.1	0.1		1 (SR)	0.1 (F)	0.1
	0.1	0.1	0.1		0.1 (SA)	1 (O)	1
	1	0.1	0.1		0.1 (WA)	0.1 (DS)	0.1
2	1	1	1	2	0.1 (WR)	0.1 (IS)	0.1
	0.1	1	1		0.1 (SR)	1 (F)	1
	1	1	1		0.1 (SA)	0.1 (O)	0.1
	0.1	0.1	0.1		1 (WA)	0.1 (DS)	0.1
3	1	1	1	3	0.1 (WR)	1 (IS)	1
	1	0.1	0.1		1 (SR)	0.1 (F)	0.1
	1	1	1		1 (SA)	1 (O)	1
	1	0.1	0.1		0.1 (WA)	0.1 (DS)	0.1
4	1	1	1	4	1 (WR)	0.1 (IS)	0.1
	0.1	1	1		0.1 (SR)	0.1 (F)	0.1
	1	1	1		0.1 (SA)	1 (O)	1
	1	0.1	0.1		1 (WA)	0.1 (DS)	0.1

Key inputs: *WR* weak repel, *SR* strong repel, *SA* strong attract, *WA* weak attract, key outputs: *IS* increase speed, *F* reorientation (up chemical gradient), *O* reorientation (down gradient), *DS* decrease speed

(a cycle ends when the complete set of pools in the network are updated once using Eq. 3 where $\Delta t = 1$) in order to obtain steady-state output values. The solution fitness was then calculated where fitness was the inverse of the error on output and the target error was 0.01. The least fit half of the population was discarded and the remaining solutions were subject to mutation and crossover in order to create the new population. To minimise the number of generations, the mutation and crossover rates were adjusted to final settings of 0.4 single point crossover and 10 % uniform mutation. The average number of generations required to reach the target error was 387. The parameters of one solution are given in Table 2. The results from this general pattern recognition experiment are shown in Table 1. As can be seen the ARN was able to recognize all four patterns correctly.

Multilayer Perceptrons have similar properties. For instance, each neuron can be approximated as either active or inactive and is comparable to the ARN whose concentration is either high or low. However, MLPs lack an explicit time dimension whereas the ARN processes inputs over a time period. In this case the ARN was subject to a continuous flux of inputs over 100 cycles causing the pool concentrations to enter a transient phase and stabilise at steady-state. The implications are that, unlike the MLP where processing is discrete-time, stored patterns are recalled only if inputs are applied for a length of time greater than that required to reach steady-state. Thus, this experiment demonstrates that the ARN is an appropriate pattern recognition technique when the requirement is to establish if a set of conditions have held true over a time period. This functionality is not so easily generated in other neural models. Discrete-time neural models provide a direct mapping from input to output and in their basic form they are unsuited for temporal pattern recognition. Continuous time models can provide this functionality but are generally more computationally complex. One such model is the Artificial Biochemical Network (ABN) (MacLeod and Capanni 2010). It is a connectionist representation which, like the ARN, can be used to recognise continuous data streams. It has a weighted sum activation function combined with leaky integrator and generates a pulse width modulated output. In other work MacLeod and Capanni (2010) setup an ABN network using 11 ABN units. This network was trained using a GA to map identical sized patterns to those used here. Like the ARN, the ABN recognised all patterns, however the reported training time was longer (average of 496 generations) and the ABN network used four additional units (1 ABN unit is approximately as complex as a single ARN unit).

In a further experiment, using the previously described network structure and set-up, the ARN was trained to recognize an additional four patterns, where the inputs were chosen to correspond to chemical signatures (for example, attractants or repellents) that trigger specific movement responses. These patterns are given in Table 1. Here, the ARN network represents a highly abstracted CSN that controls chemotactic motion of a generalised single celled organism. This artificial amoeba is assumed to have a default slow swim behaviour and in the presence of chemoeffectors the behaviour is

Table 2 Resulting network parameters for one solution after training using the genetic algorithm

General pattern setting parameters					
Pool	Initial concentration	Weight of connection	Reaction unit	Forward rate	Reverse rate
0	1st pattern value (e.g. if pattern is no.1 input is 0.1)	2.999	0	0.723	2.816
1	2nd pattern value	−2.915	1	5.411	0.837
2	3rd pattern value	0.424	2	0.969	0.643
3	4th pattern value	−0.278	3	0.120	4.310
4	0.01	−1.714	4	1.003	1.455
5	0.01	0.750	5	0.093	0.006
6	0.01	0.435	6	1.081	0.580
7	0.01	1.319	Note that in this case to simplify the program		
8	0.01	−0.104	the hidden layer pool concentrations were		
9	0.01	0.501	updated using the unweighted flux of		
10	0.01	1.492	the product		

updated accordingly. Each input signifies an environmental chemical, where input: 0 is a weak repellent (WR), 1 a strong repellent (SR), 2 a strong attractant (SA) and 3 a weak attractant (WA). Specific combinations of environmental chemicals generate specific output response, where repellents have precedence over attractants. The presence of chemical concentration to a value approximate to 1, in an output pool, corresponds to a particular behavioural response, where output pool: 0 increases speed (IS), 1 reorientation to face up chemical gradient (F), 2 reorientation down chemical gradient (O) and output 3 decreases speed (DS). Therefore, as an example, on detecting both a strong repellent and strong attractant the cell re-orientates to face down the chemical gradient and increases speed. As can be seen in Table 1, the network generated the correct response for all the abstract biological patterns.

One property of a CSN is robustness, where correct response may be generated in the presence of noise or loss of connections. In order to test this property within the ARN, random noise was introduced to the trained general pattern recognition network. Each pattern was subjected to 10 % increments of uniformly distributed random noise to a total level of 60 % of the input range. At each noise level outputs were obtained for all four patterns. It can be seen in the graph in Fig. 4 that the performance of the network gently degrades as noise is added. Error levels within 5 % are reported for both the ABN and MLP models (MacLeod and Capanni 2010) at levels of up to 50 % noise in pattern recognition tasks of the same complexity.

Similarly to an ANN, the ARN pattern recognition system, is a robust connectionist network and thus provides an intuitive bridge between biology and AI. Furthermore, this experiment illustrates that that such pattern recognition mechanisms are plausible in single celled organisms.

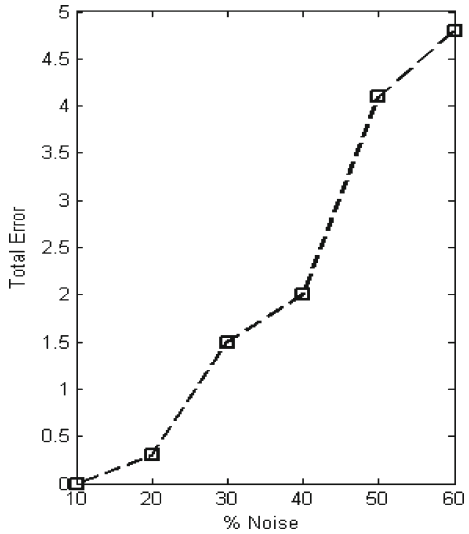


Fig. 4 Total error (y-axis) for all four patterns after introduction of random noise (x-axis) to patterns at 10 % level increments

3.2 Regulation of temporal dynamics and control of limbed robots

A common motif of CSNs is periodic oscillatory patterns of protein concentrations. Such patterns relate to particular cellular behaviours (Bray 1995; Ankers et al. 2008; Kholodenko 2006). Many illustrations of these oscillatory patterns can be found within the literature (Ankers et al. 2008; Ferrell 2004). Such temporal dynamics are explored within the ARN in order to validate its ability to represent such patterns, to explore potential application, and to gain deeper understanding of the regulatory mechanisms involved within CSNs and their role in cell intelligence.

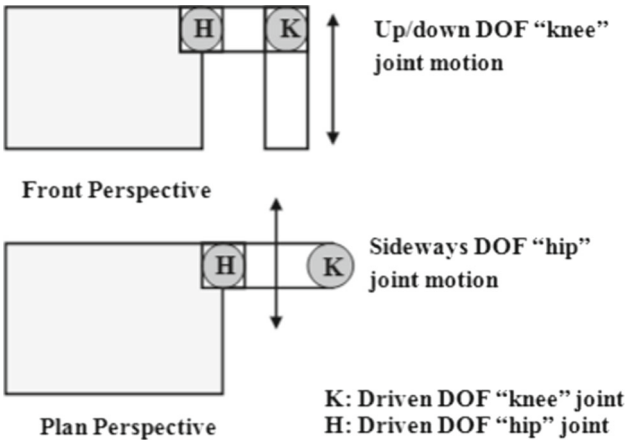


Fig. 5 The structure of a Lynxsmotion quadrupedal robot leg. Each leg has two DOFs and each DOF is controlled by a seperate motor

One method of exploring the ARNs ability to reproduce such temporal patterns, while investigating its potential AI applications, is by creating an ARN based controller to reproduce the patterns associated with robotic gaits. Terrestrial locomotion of limbed animals is achieved by multiple phase locked patterns of limb movements known as gaits. For example, depending on speed of locomotion and terrain, quadrupeds commonly walk, trot and gallop (Dagg 1973). The gait phase is a value that ranges from 0 to 1 as the gait cycle proceeds. Therefore, the motion of each limb can be described relative to the gait phase. The ideal quadrupedal gaits are described by Dagg (1973) and others (Hildebrand 1997) and are used as a standard for comparison here and similarly in other studies (Collins and Richmond 1994). In the walk gait the legs move a quarter cycle out of phase; in the trot gait each pair of diagonal limbs move half a cycle out of phase.

In these experiments, an ARN controller was implemented to generate gaits of a Lynxsmotion dual-servo quadruped 2 (Q2) robot. Each robotic leg is controlled by two servo motors, one for each degree of freedom (DOF). One motor raises the leg and the other turns it. The structure of the robotic legs is shown in Fig. 5, further details of which are given by Toth and Parker (2003).

Signals are sent by the ARN to each motor and control the angle of the rotor for each DOF, using a simple position to pulse width modulator interface circuit to control the servo. The structure of the ARN based controller is shown in Fig. 6 and was designed to include abstractions of functional motifs found in CSNs including inhibitory/excitatory reactions, cyclic loops, and feedback structures. An excellent overview of these motifs can be found in the literature (Tyson and Novák 2010). The controller comprises a network of four repeating structural units or modules, where a module is separated by a dashed line. Each module controls the two motors of a separate leg and comprises three reaction units and three pools: A, B and C. Pool A controls the up/down (U/D) motor, Pool B the back/forward (B/F) motor and Pool C controls the off period for both motors. Pool activity is regulated by a series of excitatory and inhibitory connections between reaction units. The type of connection represents the inhibitory and excitatory properties of specialized regulatory proteins common to CSNs such as enzymes. The overall network structure is organized as a closed loop allowing protein species to be recycled to the first module and thus generate a temporal oscillatory pattern. The structure of the ARN controller is capable of producing all the common gaits. The type of gait is easily modified by a simple adjustment of the initial pool values. For example, by initializing a C pool a walk gait will be generated, where the C pool chosen will determine the starting leg, and the value determines the angle to which the leg is raised. Similarly, a trot gait is achieved by initializing two C pools within alternate modules. In this particular design, the value to which the C pool(s) are initialized determines the DOF angle and were set specifically for the physicality of the particular robot although it can be freely varied.

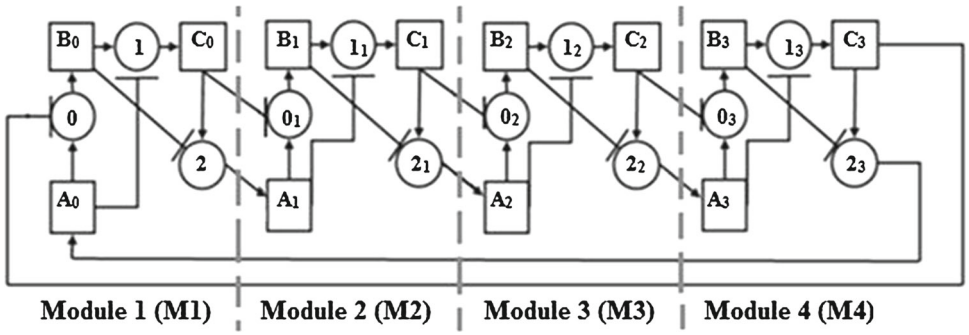


Fig. 6 The ARN based controller. Each module (shown seperated by a *dashed line*) is mapped to a single leg and is responsible for controlling the two motors which generate its motion. Pool A of each module controls the up/down motor. Pool B the back/forward motor and Pool C the stop period for each of these motors

The network architecture remains fixed throughout these experiments and the network parameters are manually set. This method was employed so that the outputs could be directly compared with other published work on similar Central Pattern Generators (CPGs) (Billard and Ijspeert 2000; Collins and Richmond 1994; Liu et al. 2009). In cases like these where the search landscape is unknown, one effective way to set the network parameters would be to use Evolutionary Algorithms or a similar pseudorandom search technique, and the current authors have employed this in other examples (for instance in the pattern recognition experiments). Due to the feedback connections, the use of gradient decent algorithms such as Backpropagation would be difficult in this application. The ARN controller was considered to generate a specific gait if the relative phases of the respective oscillatory signals were within 2 % of the standard gait cycle described previously.

Higher values of 10 % were used in other studies (Collins and Richmond 1994), and this was considered reasonable due to the variation found in real animal gaits (Afelt et al. 1983). In each case, the controller first generates the U/D motor oscillation and on reaching the maximum value the B/F motor is initiated.

As can be seen in Fig. 7, the walk gait results show that the legs are a quarter cycle out of turn, with phases of 0.0, 0.25, 0.5, 0.75 between limbs in clockwise order from FL (front left) leg. Similarly, the trot gait results in Fig. 8 show that the opposite legs are half a cycle out of turn with phases respectively of 0.0, 0.5, 0.0, 0.5. The frequency of oscillations and therefore the gait speed is easily adjusted by applying uniform increase or decrease to k_f of each unit.

Both phase locked limb patterns produced by the ARN match the standard, and compare well with other connectionist models. For example, Billard and Ijspeert present a Central Pattern Generator (CPG) based neural controller for a quadrupedal AIBO robot, similarly with two DOFs for each leg (Billard and Ijspeert 2000). The limb phases generated by this network correspond to the standard and to those produced by the ARN. Here, the network is composed of eight coupled non-linear oscillators and each oscillator consists of six leaky integrator neurons (a total of 96 neurons). Each neuron implements an activation function which is approximately as complex as the reaction unit function of the ARN, and therefore the complexity of the network is equivalent to approximately 96 ARN reaction units. Similar correspondence is found in other sources. For instance, Collins explores a CPG based neural controller for a quadrupedal robot with one DOF per limb, and compares three types of activation function models: Stein, Van der Pol, and FitzHugh-Nagumo. The controller is composed of a network of four coupled non-linear oscillators (Collins and Richmond 1994), where each oscillator controls a separate limb. The Stein model consist of three first order differential equations, the Van der Paul model consists of a second order differential equation and the FitzHugh–Nagumo model consists of two first order differential equations. All these models have approximately twice the complexity as the output produced by the ARN unit. In this case all three models require a pulsing signal to drive the network. Generally speaking the structure of these models is less flexible then either the Billard and Ijspeert (2000) model or the ARN due to their rigidly fixed internal parameters. All these models produced the gait patterns within 10 % of the standard, whereas the ARN matched the standard for both trot and walk.

Overall the ARN has a very similar capacity to generate both walk and trot gaits as the compared controllers. However, in general, it affords a higher degree of flexibility and is less computationally complex. Although robotic gaits might seem unconnected with cellular intelligence, the ARNs ability to produce them illustrates how cellular networks can generate the complex temporal patterns necessary in emergent behaviour.

3.3 Complete robotic control system

It was demonstrated in Sect. 3.1 that an ARN can recognize patterns. Furthermore it was demonstrated in Sect. 3.2 that such a system can generate temporal output patterns which can be used in control tasks. Of course in the natural world these two behaviors are linked together.

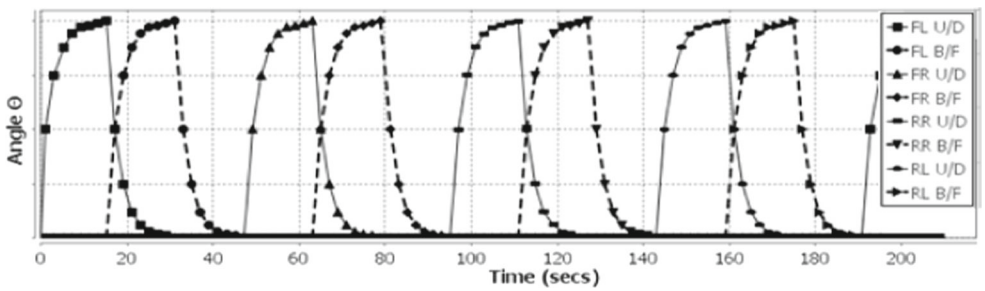


Fig. 7 Output generated for the walk gait. Legs are front left (FL), front right (FR), rear right (RR) and rear left (RL). The up/down (U/D) motor is displayed as a *solid line* and the back/forward (B/F) motor is displayed as a *dashed line*

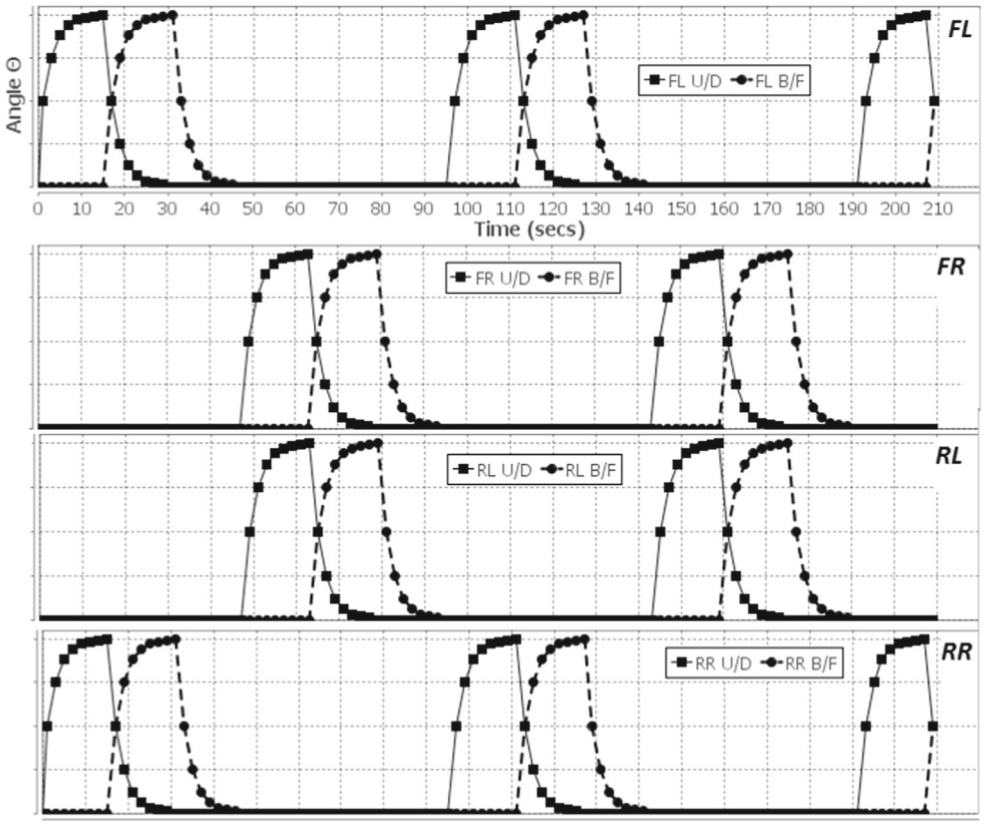


Fig. 8 Output generated for the trot gait. Legs and motors are labelled as before—see Fig. 7

In the following experiment it is illustrated that both pattern recognition and control function can be combined within a single ARN based system. Here, a more complex ARN was created to recognize specific patterns and in response automatically generate the associated temporal gait. The ARN in this experiment reuses the pattern recognition and gait network previously described in Sects. 3.1 and 3.2 respectively. The complete ARN system is shown in Fig. 9, and is functionally divided into three smaller ARN components: pattern recognition, control, and a connecting network.

The structure of the pattern recognition network, its implementation, and training methods are identical to those described in Sect. 3.1. In this case the network was trained to recognize three separate patterns (as shown in Table 3) composed of four input and four associated output mass values. The output pools of the pattern recognition network are equal to the input pools 0, 1, 2, and 3 of the connecting network. The connecting module comprises six pools (4 inputs and 2 outputs) and two reaction units and its structure is based on logic gait motifs found in biochemical networks (Tyson and Novák 2010). Essentially this component operates like two parallel Boolean AND gates, where a value of 1 at pools 0 and pool 1 will output a value of 1 at pool 4, as will a value of 1 at pools 2 and 3 output a 1 at pool 5. Two negative feedback connections between the interface network and both ARN control system subunits (shown as dashed line connections) are responsible for switching between the gaits. Therefore, if a value of 1 is output at the interface network pool 4, it will inhibit all the reaction 2's of the ARN walk subunit, thus stopping the walk gait pattern from being generated. Conversely, if a value of 0 is output at pool 4 the walk will be generated. In the same way, pool 5 of the interface controls the switching on/off of the trot control subunit. Table 3 shows the range of required behaviors in response to particular outputs generated by the connecting network.

The control system comprises two separate ARN subunits, both identical in structure and implementation to the ARN described in Sect. 3.2. Each of these subunits is responsible for generating a specific temporal gait pattern: one generates walk the other trot. The two ARN subunits provide distinct gait patterns due to the differences in initialization of the concentration values of C pools.

The complete system was tested to confirm its ability to both generate the correct behavior and automatically transition between the behaviors in response to firing input patterns 0–3. The time periods in which patterns were applied, and the expected output states are shown in Table 4.

The results for this experiment are displayed in Fig. 10. The phases produced for each gait are exactly as described previously in Sect. 3.2. The on/off periods of both trot and

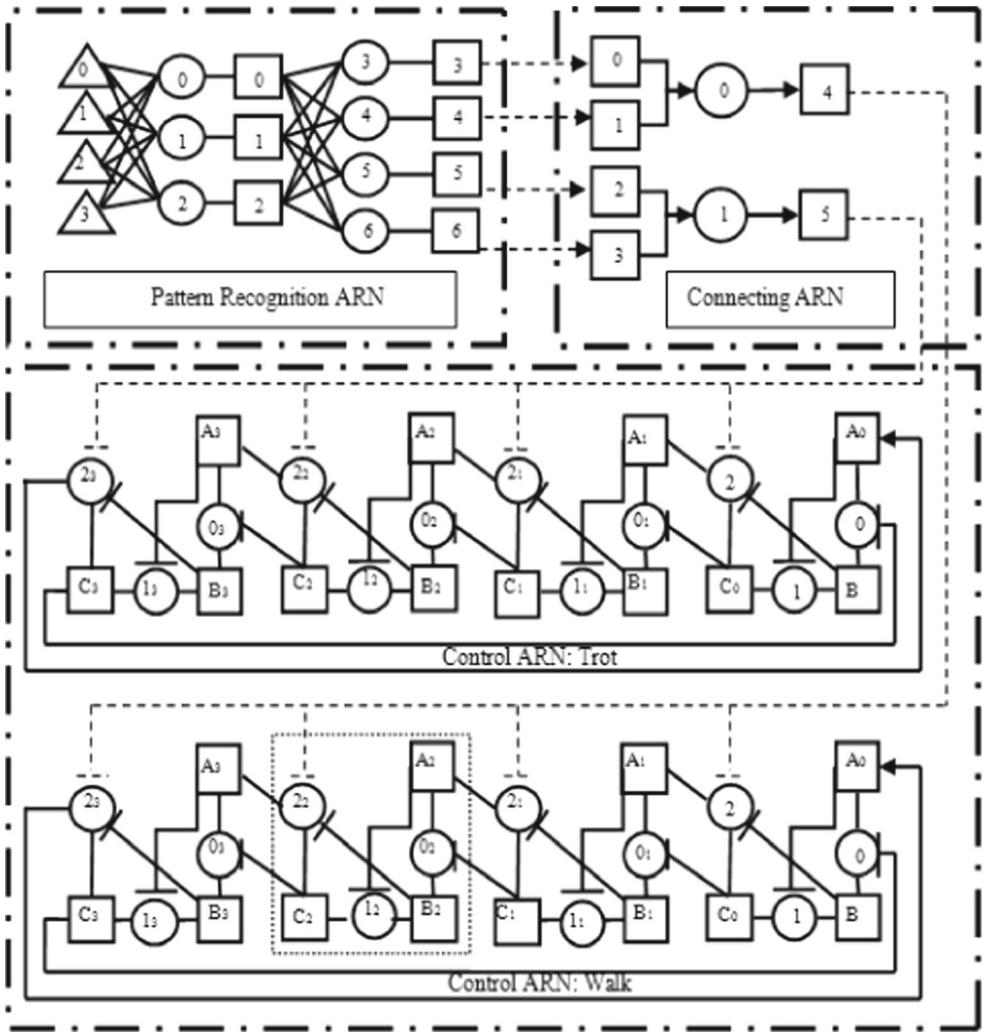


Fig. 9 A complete control system for a quadrupedal robot. On recognition of particular patterns the pattern recognition ARN generates the associated output pattern. The connecting network implements two parallel Boolean AND gates which act as switches turning the walk or trot components of the control ARN off/on. The control ARN generates the required waveform which controls the robotic gait

walk gaits are in agreement with the expected durations displayed in Table 4 with a slight transitional delay. The ARN controller, and gait phases produced have previously been compared with CPG models in Sect. 3.2. The transitions between gaits generated by these models may now be compared with those of the ARN. The results given for the Billard and Ijspeert model, show transitions from walk to gallop in approximately four leg cycles, whereas the ARN transitions from walk to trot within two leg cycles. In both cases the transitions are very smooth. There are three models described by Collins, and although gait graphs are provided for all these, gait transitions are only given for the Stein model. Here gaits transition quickly within approximately two leg cycles. However, in contrast to the ARN and Billard and Ijspeert model, the leg movements during transition are very irregular.

This complete control system demonstrates that the ARN, like a CSN, is capable of both recognizing patterns and controlling overall behavior in a single network. With the exception of spiking models, few ANNs can achieve this functionality. However, spiking models are often less flexible. For example, in the Integrate and Fire model information is rate coded and all the spikes generated are uniform (Maass 1997). Thus, unlike the ARN the Integrate and Fire model (Maass 1997) lacks the flexibility to produce pulse-width and pulse-amplitude coded information. The gait phases and transitions compared well with CPG neural controllers and showed that the ARN has application in similar robotic control tasks and can offer lower computationally complexity. These experiments illustrate how a CSN might perform the complex processing associated with the high-level behav-

Table 3 Patterns applied to the complete control system and expected gait generated

Pattern	PR network input pool no.	PR network input value	CN network input pool no.	CN input value (and output of the PR network)	CN output pool no.	CN output value	Gait
1	0	1	0	1	4	1	Inhibit walk
	1	0.1	1	1			
	2	1	2	0	5	0	Trot
	3	0.1	3	0			
2	0	0.1	0	0	4	0	Walk
	1	1	1	0			
	2	0.1	2	1	5	1	Inhibit trot
	3	1	3	1			
3	0	1	0	1	4	1	Inhibit walk
	1	0.1	1	1			
	2	0.1	2	1	5	1	Inhibit trot
	3	1	3	1			

Key: *PR* pattern recognition, *CN* connecting network

Table 4 Pattern applied to the network and expected durations of gaits

Pattern	Walk ARN network	Trot ARN network	Start time	End time	Duration
2	On	Off	0	210	210
1	Off	On	210	440	230
2	On	Off	440	560	120
1	Off	On	560	700	140
3	Off	Off	700	800	100

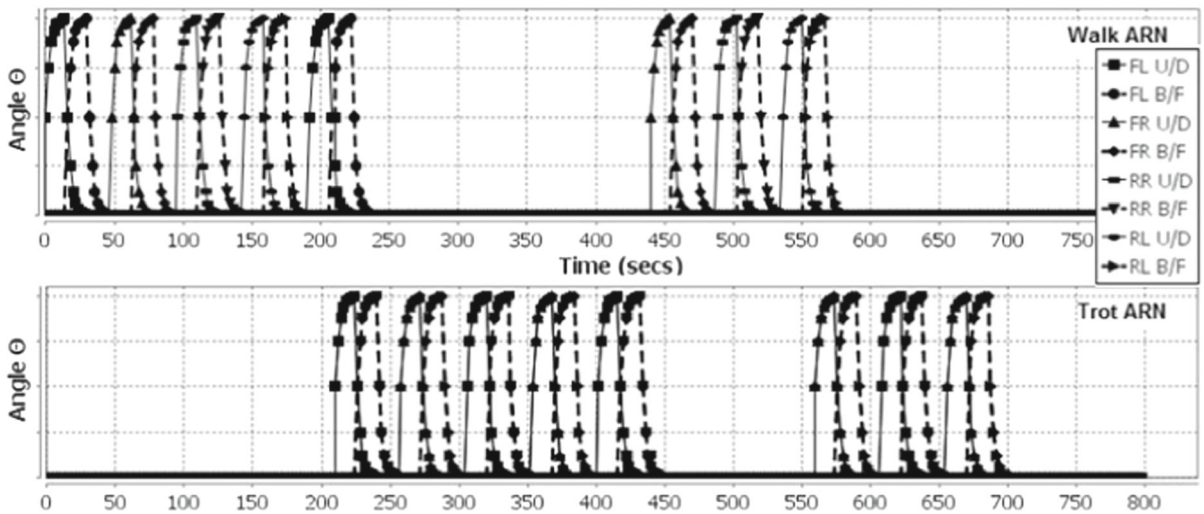


Fig. 10 The output of the complete ARN control system over 800 s. Legs and motors are labelled as before See Fig. 9

iors displayed by single celled organisms. Furthermore, it shows that abstractions of both neural networks and CSNs operate in similar ways, and have comparable functionality. This illustrates a close relationship between neural and cell intelligence.

4 Conclusions

The ARN is a new connectionist model, based on the dynamics of CSNs. It is accurate enough to represent actual chemical concentrations in the cytoplasm of a cell, but simple

enough to construct biological circuitry with applications in AI. Perhaps most importantly it is a useful tool for investigating the surprising emergent behaviours of single cells. It may help to elucidate the mechanisms involved in these, and their similarities and differences from neural based intelligence (and intelligence in its widest philosophical sense). Although other researchers have used techniques such as S-systems and Petri Nets to do related work, the ARN is unique in that it was conceived as a much more connectionist, unit-based representation, designed specifically to investigate cell intelligence.

The results presented above show that the ARN (and by extension, cellular networks) are capable of performing pattern recognition in a similar way to artificial neural models and also producing complex temporal dynamics reminiscent of spiking neural models. Additionally, it was shown that the ARN can model biological reactions and simulate real CSN pathways with an accuracy matching those of standard simulation methods (Gerrard et al. 2011). This combination of attributes makes it a unique and useful tool. The ARN systems presented above show clearly that biochemical networks are quite capable of producing many of the behaviours normally ascribed to neural networks. This helps to illuminate the many interesting results now emerging from the behavioural biology of single cells. Of course the neuron is itself a biochemical network, and one future application of the ARN may be to help unravel its more complex internal dynamics.

The simplicity of the ARN makes it a potentially useful model in more practical AI and engineering systems. As demonstrated in the case of robotics, its ability to function in both input (sensory) and output (efferent or motor) networks and in the interconnection between these, gives it applications which usually only much more complex models can fulfill. This is particularly useful in the field of robotics, where such flexibility has particular application in evolutionary control networks.

The authors intend to extend the work reported here by producing more complex cell based robots (cytobots). These will allow us to explore more aspects of cellular intelligence (for example the role of learning in these systems) as well as some practical applications such as vehicles to clear oil-spills/polution by moving along chemical gradient, rather like that in which chemotaxis operates. The ARN may also have useful applications in other areas of science—for example in modelling the complex interconnected chemical networks present in environmental and soil chemistry.

Supplementary information and code can be found at the following link: https://drive.google.com/folderview?id=0B-xGVfJFH9UmZlJTV1pROFFRb00&usp=sharing.

References

Afelt Z, Blaszczyk J, Dobrzecka C (1983) Speed control in animal locomotion: transitions between symmetrical and nonsymmetrical gaits in the dog. Acta Neurobiol Exp 43:235–250

Ankers J, Spiller D, White M, Harper C (2008) Spatio-temporal protein dynamics in single living cells. Curr Opin Biotechnol 19:1–6

Arkin A, Ross J (1994) Computational functions in biochemical reaction networks. Biophys J 67:560–578

Bhalla U (2003) Understanding complex signaling networks through models and metaphors. Prog Biophys Mol Biol 81:41–65

Billard A, Ijspeert A (2000) Biologically inspired neural controllers for motor control in quadruped robot. In: Proceedings of the IEEE–INNS–ENNS international joint conference on neural networks, Italy. IEEE, pp 637–641

Bray D (1995) Protein molecules as computational elements in living cells. Nature 376(6538):307–312

Collins J, Richmond S (1994) Hard-wired central pattern generators for quadrupedal robots. Biol Cybern 71:375–385

Dagg A (1973) Gaits in mammals. Mammal Rev 3:135–154

Dittrich P (2005) Chemical computing. Unconventional programming paradigms. Springer, Berlin/Heidelberg

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

Ferrell J (2004) Self perpetuating states in signal transduction: positive feedback, double negative feedback and bistability. Cur Opin Cell Biol 14(2):142–148

Ford B (2009) On intelligence in cells: the case for whole cell biology. Interdisipl Sci Rev 34(4):350–365

Gerrard C, McCall J, Coghill G, Macleod C (2011) Artificial reaction networks. In: Proceedings of the 11th UK workshop on computational intelligence, Manchester, pp 20–26

Gerrard C, McCall J, Coghill G, Macleod C (2012a) Temporal patterns in artificial reaction networks. In: Proceedings of the 22nd international conference on artificial neural networks, part 1, vol 7552, Lausanne, pp 1–8

Gerrard C, McCall J, Coghill G, Macleod C (2012b) Adaptive dynamic control of quadrupedal robotic gaits with artificial reaction networks. In: Proceedings of the 19th international conference on neural information processing, vol 7663, part 1, Doha, pp 280–287

Guo H, Meng Y, Jin Y (2009) A cellular mechanism for multi-robot construction via evolutionary multi-objective optimization of a gene regulatory network. BioSystems 98(3):193–203

Gurney K (1992) Training nets of hardware realizable sigma–pi units. Neural Netw 5:289–303

Hamann H, Stradner J, Schmickl T, Crailsheim K (2010) A hormone-based controller for evolutionary multi modular robotics: from single modules to gait learning. In: Evolutionary computation (CEC) IEEE, pp 1–8In, Evolutionary Computation (CEC) IEEE, pp 1–8

Hild W, Pollinger K, Caporale A, Cabrele C, Keller M, Pluym N, Buschauer A, Rachel R, Tessmar J, Breunig M et al (2010) G protein-coupled receptors function as logic gates for nanoparticle binding and cell uptake. Proc Natl Acad Sci USA 107:10667–10672

Hildebrand M (1997) Analysis of asymmetrical gaits. J Mamm 58:131–156

Kholodenko B (2006) Cell signaling dynamics in time and space. Nat Rev Mol Cell Biol 7(3):165–176

Kowalewski S (2002) Modeling, analysis and design of hybrid systems. Lect Notes Control Inf 279:153–171

Krautmacher M, Dilger W (2004) AIS based robot navigation in a rescue scenario. In: Springer lecture notes computer science, vol 3239, pp 106–118

Liu C, Chen Y, Zhang J, Chen Q (2009) CPG driven locomotion control of quadruped robot. IEEE International conference on systems, man and cybernetics, San Antonio

Maass W (1997) Networks of spiking neurons: the third generation of neural network models. Neural Netw 10(9):1659–1671

MacLeod C, Capanni N (2010) Artificial biochemical networks: a different connectionist paradigm. Artif Intell Rev 33(1–2):123–134

Murata T (1989) Petri nets: Properties analysis and applications. Proc IEEE 77(4):541–580.

Rumelhart D, McClelland J (1986) Parallel distributed processing: explorations in the microstructure of cognition. Volume I: Foundations.

Savageau M, Voit E (1987) Recasting nonlinear differential equations as S-systems: a canonical nonlinear form. Math Biosci 87:83–115

Shen W, Will P, Galstyan A, Chuong C (2004) Hormone-inspired self-organization and distributed control of robotic swarms. Auton Robots 17(1):93–105

Stadtman E, Chock P (1997) Superiority of interconvertible enzyme cascades in metabolic regulation: analysis of multicyclic systems. Proc Nat Acad Sci USA 74:2766–2770

ten Eikelder T, Crigins S, Steijaert M, Liekens A. Hilbers P (2009) Computing with feedforward networks of artificial biochemical neurons. In: Proceedings of the 2nd international workshop on natural computation, Japan, vol 1. Springer, pp 38–47

Toth D, Parker G (2003) Evolving gaits for the lynx motion hexapod II robot. In: Proceedings of the 7th World multiconference on systems, cybernetics, and informatics, Orlando, pp 229–234

Tyson J, Novák B (2010) Functional motifs in biochemical reaction networks. Annu Rev Phys Chem 61:219–240

Wang B, Kitney R, Joly N, Buck M (2011) Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology. Nat Commun 2:508

Woo W, Khor L (2004) Blind restoration of nonlinearly mixed signals using multilayer polynomial neural network. IEEE Proc Vis Image Signal Process 151(1):51–61

Ziegler J, Banzhaf W (2000). Evolving a" nose" for a robot. In: Proceedings of the genetic and evolutionary computation conference (GECCO-2000)