Artificial Chemistry Approach to Exploring Search Spaces Using Artificial Reaction Network Agents

Claire E. Gerrard, John McCall, Christopher Macleod,
IDEAS Research Institute,
Robert Gordon University,
Aberdeen, Scotland.
c.e.gerrard@rgu.ac.uk, j.mccall@rgu.ac.uk,
chris.macleod@rgu.ac.uk

George M. Coghill,
Department of Computing Science,
University of Aberdeen,
Aberdeen, Scotland.
g.coghill@abdn.ac.uk

***Abstract*— The Artificial Reaction Network (ARN) is a cell signaling network inspired representation belonging to the branch of A-Life known as Artificial Chemistry. It has properties in common with both AI and Systems Biology techniques including Artificial Neural Networks, Petri Nets, Random Boolean Networks and S-Systems. The ARN has been previously applied to control of limbed robots and simulation of biological signaling pathways. In this paper, multiple instances of independent distributed ARN controlled agents function to find the global minima within a set of simulated environments characterized by benchmark problems. The search behavior results from the internal ARN network, but is enhanced by collective activities and stigmergic interaction of the agents. The results show that the agents are able to find best fitness solutions in all problems, and compare well with results of cell inspired optimization algorithms. Such a system may have practical application in distributed or swarm robotics.**

Keywords*— *Artificial Reaction Networks; Artificial Chemistry; Swarm Robotics

I. INTRODUCTION

Single celled organisms display an astonishing array of complex behaviors. Some can avoid light with photo-sensitive spots; some actively hunt prey; while others can build protective shelters [1]. Such behaviors improved these organisms' chances of survival through the process of natural selection. In recent years a growing body of research has illuminated the remarkable capabilities of single cells to store and process information [2, 3]. The mechanisms involved are quite different from those of a digital computer. Within a cell, the current state is represented as a set of spatially distributed concentrations of chemical species. This data is processed by vast networks of chemical reactions termed cell signaling networks (CSNs). In this way, cells are able to respond to current environmental conditions, communicate with other cells, and perform internal self maintenance operations.

Several researchers have highlighted the processing capabilities of these networks [2, 4, 5] and similarities to Artificial Neural Networks (ANNs) [2, 5]. For example, it has been demonstrated that such networks can perform Boolean and fuzzy logic functions and are equivalent to a Turing machine [2, 4]. Furthermore, CSNs contain topological features such as feedback loops and interconnectivity, thus forming highly complex systems [2, 5, 6].

It is possible to abstract the computational properties of such chemical processing to create a type of model called an Artificial Chemistry. Artificial Chemistry is a subfield of A-Life, and in its broadest sense, it describes man-made systems which are similar to real chemical systems [7]. In previous work, a new Artificial Chemistry representation of CSNs- the Artificial Reaction Network (ARN) was introduced and investigated as a means to control limbed robots [8, 9, 10].

Our first aim is to show that an ARN network can be instantiated and used as the internal control system for multiple instances of cell-like autonomous distributed agents. Like biological cells these agents react to their environment, and stigmergically communicate to facilitate collective emergent behavior. Our second aim is not to present a new optimization algorithm; rather it is to show that these agents can perform a range of useful search behaviors in a variety of situations, and that their search strategy can compare to that of established optimization algorithms using similar cell inspired strategies. The agents are placed within a simulated environment with the task of finding the global minima of a set of well-known benchmark problems. The search spaces chosen are not high dimensional, but chosen to reflect problems which situated robotic agents could perform in real world environments.

The paper is structured as follows: section 2 provides an overview of the ARN representation; this is followed by an overview of the ARN agents in section 3. The experimental details are discussed in section 4 followed by results in section 5. Finally section 6 presents the conclusions.

II. ARTIFICIAL REACTION NETWORKS

A full account and verification of the ARN representation can be found in our work [8, 9, 10]; thus to preserve space only a brief summary is provided here.

The ARN comprises a set of networked reaction nodes (circles), pools (squares), and inputs (triangles) as shown in Fig. 1. Each pool stores the current available chemical species concentration (avail); this concentration represents data within

the system. Thus, the complete set of pool concentrations at time t, corresponds to the current state of the system. Inputs are a special type of pool, the only difference being that they are not updated by flux at each time step, and are used to represent continuous concentrations, for example, environmental inputs or enzymes. Each circle corresponds to a reaction unit, representing a reaction between a number of chemicals. Data is processed by reaction nodes transforming incoming pool values to connected outgoing pool values. Connections symbolize the flow of chemical into and out of reaction units and their weight (w) corresponds to reaction order. Connections provide the facility to create complex control structures using combinations of inhibitory and excitatory connections. Fig. 1 shows the reaction between species A and B to produce species C. At time interval Δt, each reaction unit's temporal flux value is calculated by applying Euler's approximation to the differential rate equation as shown in (1).

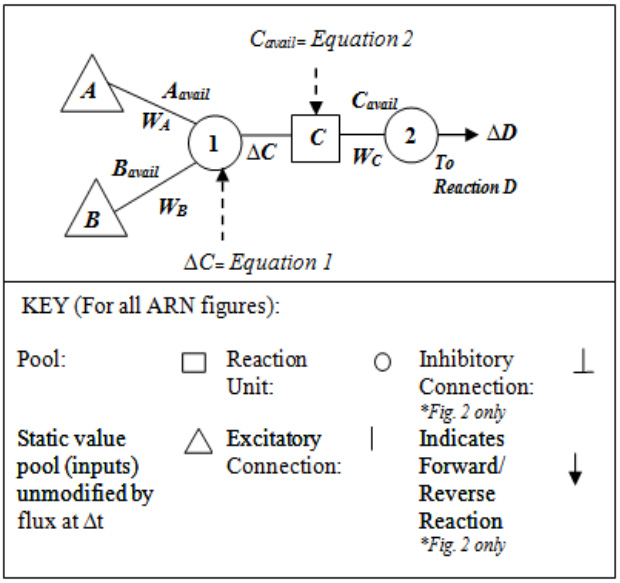

Fig. 1. The Artificial Reaction Network representation.

$$\Delta C = \left(K_{f(C)} \left(A_{avail}^{W_A} B_{avail}^{W_B} \right) - K_{r(C)} \left(C_{avail}^{W_C} \right) \right) \Delta t \quad (1)$$

$$C_{avail} = C_{avail} + \left(\Delta C - \frac{W_C}{W_C + \alpha} \Delta D \right) \quad (2)$$

Where:
A, B, C, D = Species Concentrations
W = Reaction order (weight)
avail = Available species concentration
K_f = Forward rate constant
ΔC = Change in species concentration C
K_r = Reverse rate constant
α=sum of other incoming weights

This value is then used to update the current concentration of each reaction's connecting pools as shown in (2). Pools may asymptotically approach 0, and thus below a particular threshold a pool is considered empty and its value set to zero. A reaction step may proceed if it meets its preconditions. Preconditions are met if incoming inhibitory pools are inactive, and incoming excitatory pools are active. In a similar way the completion of a reaction step will fulfill a number of post conditions, which depend on the parameters and connections of the reaction step.

III. ARN Controlled Agents

In the following experiments a number of autonomous ARN controlled software agents termed "cytobots" ("cyto" from Greek for cell, and "bot" from robot) are created and initialized within an artificial environment containing a nutrient landscape. The cytobots task is to find the maximum food level by moving around within this simulated environment in as few evaluations (reading the value of food at the current position) as possible. Similar to the way in which a CSN acts as the control system to a cell, the behavior of each cytobot is controlled by its own instance of an ARN network. In this way, the ARN directs the agent's movement, enables the agent to react to situated environmental patterns, and allows it to stigmergically communicate with other cytobots to contribute to higher level function. The cytobot ARN network was designed to produce two simple behavioral modes: foraging and starvation, both are based on the movement patterns of unicellular organisms. Cytobots forage by performing a biased random walk behavior while consuming food at each passing location. This pattern of movement is exemplified by the bacteria *Escherichia coli* (*E. coli*), where foraging cells alternate periods of runs (forward motion) and random redirections known as tumbles. By comparing concentrations of attractants and repellants in a temporal fashion, the organism is able to reduce the frequency of tumbles up concentration gradients of attractants, and down gradients of repellants, thus providing the bias. This behavior implements a type of optimization where biased periods of movement in the direction of attractants lead to overall travel toward more favorable conditions [11].

The starvation behavior is based on the pattern of motion displayed by starving cells of the cellular slime mould *Dictyostelium discoideum* (*D. discoideum*). During the vegetative stage of *D. discoideum*, cells move up gradients of folic acid secreted by its bacterial prey. When the food resource has been depleted, the amoebae begin to starve and enter the aggregation phase of their life cycle. During aggregation, starving cells secrete cAMP (cyclic adenosine monophosphate) which serves as a signal to attract surrounding amoebae towards a central location [12]. In this simulation, when the cytobots enter starvation mode, the level of food surrounding the agent represents corresponding levels of attractant cAMP. When a cytobot travels over areas of low or zero food it enters the starvation mode. Instead of turning in a random direction, the new direction is weighted toward higher concentrations of food within its surrounding area. This behavior forces exploration of unexplored search space because previously visited positions have a food level of 0.

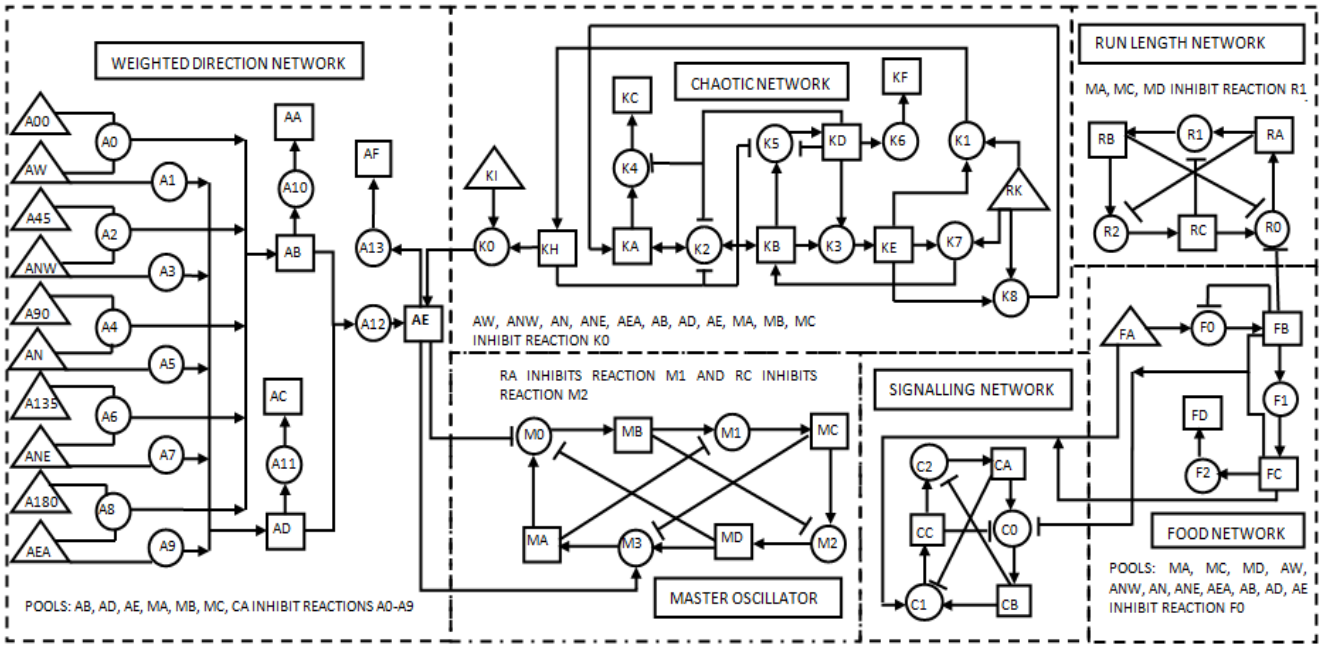

Fig. 2. The cytobot ARN network. Each cytobot is controlled by an instance of this network. The network is composed of 6 subnetworks

Consumption of environmental food therefore serves as a stigmergic signal, where agents are inclined to move up the nutrient gradient created by their foraging activities. The cytobot ARN was designed to perform the starvation and foraging behaviors described above and is composed of 6 subnetworks as shown in Fig. 2. Each subnetwork contributes a functional aspect to either or both starvation and foraging behaviors. The subnetworks are discussed below.

A. The Master Oscillator

The master oscillator functions to synchronize all the outputs from all the other subnetworks together and is what each agent references at each time step to ascertain its current behavior. It is a simple closed loop, with a token unit of chemical cycling around it. It consists of 4 reaction units: M0, M1, M2, and M3 (all with reaction rate of 1) and 4 pools MA, MB, MC and MD. Each pool activates one of three behaviors, and for every time step that a particular pool contains the token unit, its corresponding behavior is performed. Pool MA activates turn, MC activates run and pools MB and MD activate stop. If these pools were switches to motor actuators on a simple wheeled robot, pool MC would switch on all wheel motors, while pool MA would switch on wheel motors on the left side only, thus turning the robot. The remaining pools would act as off switches. The other subnetworks inhibit or excite the reaction units of the master oscillator to allow or prevent chemical flow. The number of time steps that a chemical is present in a particular pool indicates the length of time that a particular behavior is performed. Thus if pool MC contains a chemical for 10 time steps, then the agent will move forward for 10 time steps; similarly if this were pool MA, the agent would turn for 10 time steps.

B. The Food Network and The Run Length Network

The food network senses the level of food within the environment and connects to the run length network to modify the number of steps forward according to the level sensed. The value of food at the cytobots' current position is stored at input pool FA. The forward rate of reaction node F0 is 1, thus the content of FA is transferred to pool FB in a single time step. The presence of chemical in pool FB inhibits the run network reaction R0 for a number of time steps according to the level of food (by setting forward rate of unit F1 to 1 and weight to 0 this can be an exact correlation). This in turn stops pool RC in the run length network from emptying. Pool RC inhibits reaction M2 of the master oscillator thus preventing pool MC from emptying for the same number of time steps. As discussed previously, the number of time steps which pool MC contains the token unit represents the number of time steps to move forward.

C. The Signaling Network

The signaling network functions as a switch between starvation and foraging mode. Low food levels trigger the starvation response and allow the weighted direction network to control each new angle. Sufficient food will switch off the weighted direction network and allow the chaotic network to control each new angle. It is a simple closed loop with a token unit of chemical flowing around it. Pool CA acts as a switch between foraging and starvation behavior, where the presence of chemical in CA inhibits the weighted direction network-while its absence switches on the weighted direction network; this in turn inhibits the chaotic network, as shown in Fig. 2. In this component, all reaction units have a forward flux of 0.5; which ensures a minimum number of time steps for each behavior.

D. The Weighted Direction Network

The weighted direction network senses food within the agents' immediate environment and calculates a tumble angle which is weighted toward higher food levels. This network interfaces with the environment via a number of receptor pools (AW, ANW, AN, ANE, AEA) which sense the level of food around the cytobot. These pools represent receptors positioned at points around the front of its perimeter, allowing the agent to travel in a similar way to that of a polarized biological cell. For example, during the aggregation phase of their life cycle, *D. discoideum* cells are polarized, and one side becomes the leading edge which always faces in the direction of travel [12]. For each receptor input pool, there is a static pool containing a fixed level of chemical in correspondence to its direction. Directions start from AW (west) with a corresponding numeric value of 0 (A00) and progress in 45 degree steps through each direction to east. As the receptor positions around the agent are fixed, directions are always relative to that in which the agent is facing. All connections have a weight of 1 with the exception of the connection between pool AD and reaction A12 which has a weight of -1. This negative connection raises the sum of food detected in pool AD to -1, which multiplied by AB, allows an average angle to be calculated.

The calculated angle interfaces with the remaining subnetworks at pool AE. In an actual organism receptors are set around the cell perimeter and direct movement appropriately. In this simulation, for simplicity, a count of the number of time steps that MA contains the token unit is processed to gain the turn angle relative to the agents' current heading using (3). Thus if the number time steps is 120 and the agent is facing north, then the current heading would equal 0 and the new heading would equal 30.

$$h \equiv ((n-90)+c) \bmod 360 \tag{3}$$

Where:
h= new heading
n = count of time steps pool MA contained chemical
c = current heading

E. The Chaotic Network

The chaotic network, as shown in Fig. 2, is responsible for generating pseudo random angles which agents use to perform the foraging tumble behavior. It is a networked implementation of a Logistic Map, see (4). Without prior knowledge of the initial conditions the output of the logistic map is unpredictable, whereas with prior knowledge it is deterministic- therefore the series cannot be described as truly random but as pseudo random. Its output has long been proposed as a pseudo-random number generator. Ulam and von Neumann [13] were the first to examine this and it has been successfully used in this capacity by several researchers [14]. The probability density distribution of the Logistic Map is non-uniform and is described by (5) [14]. When $\lambda=4$ the distribution is "U" shaped with higher probability of values closer to the minima and maxima of X and symmetric distribution at the midpoint. The general shape of the distribution is invariant for the complete range of state variables from 0 to 1.

$$X_{n+1} = \lambda X_n (1 - X_n) \tag{4}$$

Where:
X_n= state variable of value $0 \le X_n \le 1$
λ= system parameter of value $1 \le \lambda \le 4$

$$P(X) = \frac{1}{\pi\sqrt{X(1-X)}} \tag{5}$$

Where:
P(X) = probability of X occurring

The chaotic network component, as shown in Fig. 2, operates in the following manner. At the start of the simulation, the pools KA and KB of each cytobots' chaotic network are initialized to the same random value between 0 and 1 (to 5 decimal places). This represents the first value of X where X is the state variable of (4). All the other pools are initialized to 0 with the exception of the static pools KI and RK whose initial values are 360 and 1 respectively. Reaction K2 is responsible for generating each new value of X and has a forward and reverse rate of 4 (the logistic map exhibits chaotic behavior when λ is 4). The connection between KA and K2 has a weight of 1 and the connection between K2 and KB has a weight of 2. The remaining series of reactions function to copy the value of X 3 times, where 2 copies serve as the new initial values of KA and KB and the remaining copy participates in the final output of the network at KH. Static pool KI has a fixed value of 360 which in reaction K0, allows the network to convert the pseudo random number at KH to an angle value between 0 and 360. However, reaction K0 cannot proceed until all 11 pools that inhibit it are empty. These inhibitory connections ensure that random angles are not output while the agent is in starvation mode, and that pool AE is empty before adding more chemical.

The ARN implementation of the Logistic Map was tested against the recursive relation shown in (4). The results generated for (4), were obtained using Matlab, where $\lambda=4$, initial X = 0.927725, and iterated 100000 steps. The complete range of state variables between 0 and 1 were divided into 100 equal subintervals and the frequency of occurrence of each subinterval interval was plotted. Similarly, the chaotic network component of the ARN was run for 100000 cycles, using the same parameters of X and λ. These results were processed in the same way and are shown in Fig. 3. The frequency distribution gained from the ARN is identical to that obtained using matlab and by other researchers using the same parameters [14]. The same comparison was repeated 100 times at different values of X, and the ARN consistently produced the same values as (4).

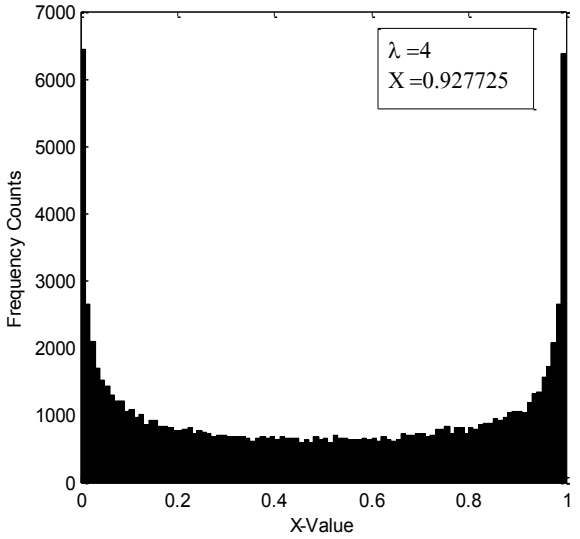

Fig. 3. Frequency distribution for each value of X resulting from the chaotic network when the first value of X is 0.927725 and λ=4

IV. METHODOLOGY

In the following experiments, cytobot agents are applied to the task of finding the minima in a number of benchmark 2D optimization problems. These are the following functions: Rosenbrock, Peaks, Inverted sinc, and Bowl (see Table 1 for formulae, domains, and minima). Three experiments were performed for each function, where each uses either 1, 3 or 6 cytobots and is performed 100 times. The task of the agents is to find the minima of the functions within as few evaluations (reading value of food at current x, y coordinate) as possible. The range of output values for each of the functions represents the concentration of food (also the fitness of an agent at that point) within a simulated environment. Values approaching the minima represent higher food levels, and values approaching the maxima represent lower food levels. The simulated environment consists of a 2D area of 400 x 400 pixels. A scaling factor is used to map the domain to the actual dimensions of the simulation, e.g. Rosenbrock domain of [-2, 2] mapped to a simulation space of [-200, 200] by a scaling factor of 100. For display purposes, a corresponding grayscale color is used to show the distribution of food within the environment as displayed in the screenshot of the simulation in Fig. 4. Each agent consists of a token to mark its current position and an instance of an ARN network, as discussed in section 3. At the start of each experimental run, each agent's ARN network is initialized as described in section 3, and each is positioned at random x,y coordinates within the search space. The agents undergo alternating phases of "searching" and "repositioning," for a number of cycles until one reaches a position of within 0.04 of the global minima of the function. This value was chosen as it is within 1% of the global minima for all the functions used. The high level pseudocode describing the searching and repositioning phases is provided in Fig. 5. Searching is characterized by the 2 ARN controlled behaviors- foraging and starvation, as described in section 3. In each search phase each agent performs a total of 3 moves (3 evaluations of the environment). The length of a run corresponds to the number of pixels a cytobot moves forward and is subject to the output from the run length network. After each tumble, and before moving forward, the food level at the current position is input into the ARN network as described previously. The agents travel at a speed of 1 pixel per time step, thus the number of time steps produced by the run length network corresponds directly to the number of pixels the agent moves forward. As a cytobot travels, the food at each passing position is consumed and its path within the simulation is represented in black (as shown in Fig. 4). During the search phase, a central control unit, external to all cytobot agents, keeps track of each cytobots best fitness and the coordinates of that value.

Let $A = \{a_0, \cdots, a_N\}$ equal the set of all agents

$$f_{tot} = \sum_{a=0}^{N} f_a \tag{6}$$

Then $\forall a \in A \setminus \{a^*\}$

$$f_{ra} = \frac{f_a}{f_{tot}} \tag{7}$$

$$d_{ax} = x_a - x_{a^*} \tag{8}$$

$$d_{ay} = y_a - y_{a^*} \tag{9}$$

$$p_x = x_{a^*} + \left(\sum_{a=0}^{N} d_{ax} \times f_{ra} \right) \tag{10}$$

$$p_y = y_{a^*} + \left(\sum_{a=0}^{N} d_{ay} \times f_{ra} \right) \tag{11}$$

$$x_a = rand(r) + p_x \tag{12}$$

$$x_a = rand(r) + p_y \tag{13}$$

Where:

a*= agent with highest fitness
f_a= fitness of agent a_n
f_{tot}= total fitness of all agents
f_{ra}= ratio of agent a_n fitness to f_{tot}
x_{a^*}= the x coordinate of a*
y_{a^*}= the y coordinate of a*
x_a= agent a_n x coordinate
y_a = agent a_n y coordinate
d_{ax}=difference between x_a and x_{a^*}
d_{ay}= difference between y_a and y_{a^*}
p_x=total of all d_{ax}
p_y= total of all d_{ay}
rand(r)= a random value within a defined radius r

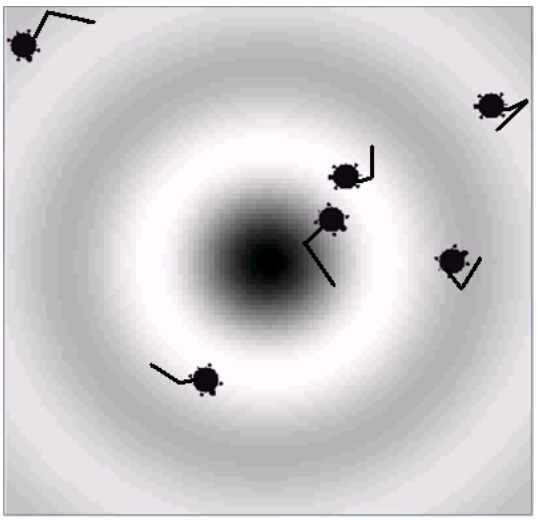

Fig. 4. A screen shot of the simulation showing 6 cytobots in the inverted sinc search space. The greyscale color represents the food distribution.

```
INITIALISE cytobots
WHILE (best fitness outwith 0.04 of global minima)

  START search phase
    WHILE (more searching phase moves)
      FOR each agent start searching phase
        Turn agent
        Set receptor pools of food network
        Set receptor pools of weighted direction network
        Move agent forward
        IF (new food level > previous food level)
            Record fitness
            Record current position
        END IF
      END FOR
    ENDWHILE
  END search phase

  START reposition phase
  CALCULATE new central point P to reposition
  INITIALISE agents at new position
 END reposition phase
END WHILE
```

Fig. 5. High level pseudocode for each experiment

After completing the searching phase, agents switch to the repositioning phase. This phase is used to focus searching toward areas containing higher food levels and is inspired by stages of the life cycle of *D. discoideum*. Having depleted the level of nutrients within the immediate environment, *D. discoideum* cells begin to starve, and aggregate to form a slug. The slug travels in the direction of more favorable conditions by moving toward attractants such as light, warmth, and humidity. On finding a suitable location, it eventually forms a fruiting body which disperses spores within its immediate surroundings. The spores mature into cells, and begin foraging within the new environment [12]. When the cytobots enter the repositioning phase, the central control unit processes each agent's best fitness position to compute a new central point P, weighted in favor of higher fitness, as described by equations (6-13). Agents are then repositioned randomly within an area of radius r from point P to begin the next search phase. For the purposes of this simulation travelling to the new position was not modeled, as this does not affect overall behavior and would only occur if the cytobots were applied to real world environments.

V. RESULTS

The experimental results are displayed in Table 1. For each experiment, the average best, f_a, and best solution, f_b, for 100 independent runs are presented. The average number of evaluations and the standard deviation for all agents is displayed as "Avg Eval for all agents" and "Std Dev" respectively. The average number of relocations for each agent is presented in the final column as "Avg Reloc per agent".

In all experiments the cytobots were able to find the global minima. Cytobots performed best in Bowl and Rosenbrock functions, where, using 6 cytobots, the average number of total evaluations and relocations per agent respectively for Bowl was 56.4 and 2.1 and for Rosenbrock was 79.8 and 3.4. The cytobots performed least well in the Inverted Sinc search space, where the lowest number of total evaluations was 94.8 using 6 cytobots. In all the experiments, a slight increase in the number of cytobots generally results in a significant reduction in the total number of evaluations performed. This is most significant for Peaks where using 3 and 6 cytobots results in approximately 30% and 60% respective reductions in the total number of evaluations when compared to the results for 1 cytobot. The Mann Whitney U test was used to determine any significant (95% confidence) statistical difference in the total number of evaluations between experiments using 1 and 3 and 3 and 6 cytobots. In all experiments there was a significant difference between 1 and 3 agents, with the exception of the Inverted Sinc function. In Peaks there was a significant difference in all experiments, while in the Inverted Sinc there was no significant difference found. Thus increasing the number of cytobots from 1 to 3 both reduces the time to find the global minima and the number of evaluations, but this effect can be quickly reversed if too many cytobots are added.

The paths of agents through the search space indicate reasons for variation in results. In simple landscapes such as Bowl, agents descend steadily toward the minima, as shown in Fig. 6. Similarly in Rosenbrock, agents quickly descend to the narrow valley and are forced to steadily move along it by moving up the nutrient gradient created by the consumption of food, until finding the global minima. In Peaks, agents move from their initial positions and search many parts of the domain. Fig. 7 shows the agents' trajectories, one can see that peaks are avoided and troughs are pursued. However, if fewer agents are used they may quickly become trapped in local minima causing a significant rise in the number of evaluations. Increasing the number of agents by a small amount expands the amount of search space explored per cycle, and increases the chance of finding better solutions and/or leaving local minima. Another possibility is to increase the number of moves for each searching phase, thus allowing an agent to travel a sufficient distance to escape local minima. Similar solutions could be adopted in the Inverted Sinc function.

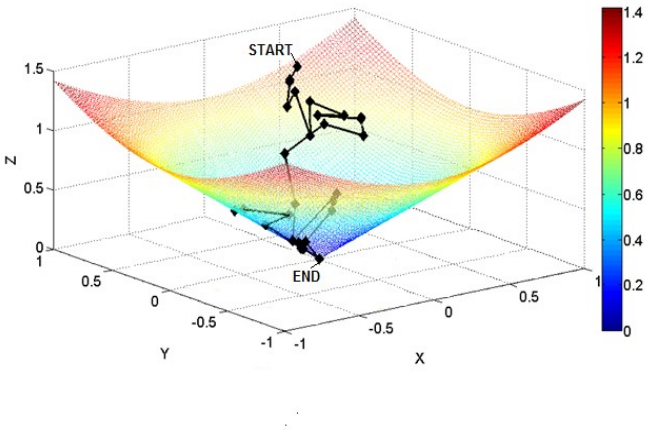

Fig. 6. Typical path of 1 cytobot in Bowl search space

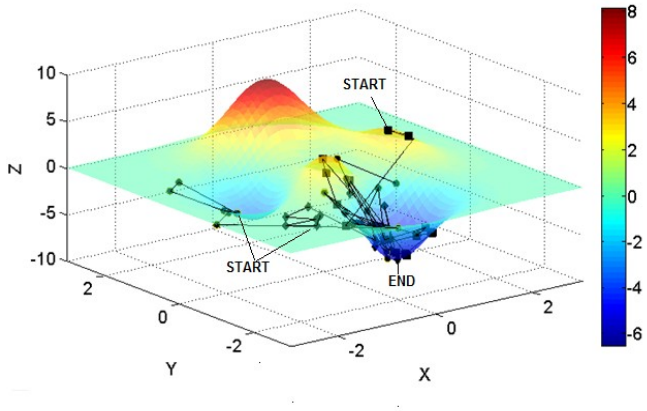

Fig. 7. Typical path of 3 cytobots in Peaks search space

TABLE I. CYTOBOT AGENTS 2D SEARCH SPACE RESULTS

Functions	**No. of Agents**	**Formulae**	**Avg (fa) Best and Best (fb) Results**	**Domain**	**f(x*) = f***	**Avg Eval for all agents (Std Dev)**	**Avg Reloc per agent**
Rosenbrock's Function	1	$f(x) = 100 \cdot (x_2 - x_1^2)^2 + (1 - x_1)^2$	f_a = 0.04 $f_b = 0$	$x_1, x_2 \in [-2,2]$	$f(1,1) = 0$	98.5 (16.3)	32
Rosenbrock's Function	3	As above	f_a = 0.03 $f_b = 0$	As above	As above	84.6 (9.5)	8.4
Rosenbrock's Function	6	As above	f_a = 0.01 $f_b = 0$	As above	As above	79.8 (4.8)	3.4
Bowl	1	$f(x) = \sqrt{x_1^2 + x_2^2}$	f_a = 0.03 $f_b = 0$	$x_1, x_2 \in [-1,1]$	$f(0,0) = 0$	81.9 (14.8)	26.3
Bowl	3	As above	f_a = 0.02 $f_b = 0$	As above	As above	64.2 (7.6)	6.1
Bowl	6	As above	$f_a = 0.02$ $f_b = 0$	As above	As above	56.4 (5.8)	2.1
Peaks	1	$f(x) = 3*(1-x_1)^2 \cdot \exp(-(x_1^2) - (x_2+1)^2)$ $-10 \cdot (x_1/5 - x_1^3 - x_2^5) \cdot \exp(-x_1^2 - x_2^2)$ $-1/3 \cdot \exp(-(x_1+1)^2 - x_2^2)$	f_a = -6.51 f_b = -6.55	$x_1, x_2 \in [-3,3]$	$f(0.23,-1.63)$ = -6.55	151.7 (59.2)	49.6
Peaks	3	As above	f_a = -6.51 f_b = -6.55	As above	As above	108.9 (17.4)	11.1
Peaks	6	As above	f_a = -6.52 f_b = -6.55	As above	As above	64.8 (5.7)	2.6
Inverted sinc function	1	$f(x) = -1 \cdot \left(\frac{\sin\left(\sqrt{x_1^2 + x_2^2}\right)}{\sqrt{x_1^2 + x_2^2}} \right)$	f_a = -1.04 f_b = -1	$x_1, x_2 \in [-10,$	$f(0,0) = -1$	163.5 (63.4)	53.5
Inverted sinc function	3	As above	f_a = -1.03 f_b = -1	As above	As above	109.5 (30.1)	11.2
Inverted sinc function	6	As above	f_a = -1.03 f_b = -1	As above	As above	94.8 (10.1)	4.2

These results are compared with other optimization algorithms inspired by behaviors of single celled organisms. For example, Passino developed the Bacterial Foraging Optimization Algorithm (BFOA), inspired by foraging behaviors, reproduction and dispersal events in the life cycle of *E. coli* [15]. Like the foraging behavior of the cytobots, movement is modeled as a biased random walk, where, after each random redirection, the cell moves forward a length according to current food levels. In a nutrient hill-climbing experiment (without swarming effects), 50 cells are initialized at random starting positions within a 2D search space. This search space is similar to Peaks but with 5 troughs and a domain of [30, 30]. Similarly to cytobots the cells tend toward valleys and avoid peaks. After 4 generations (4 reproductive steps), and moving 100 chemotactic steps (moves) between generations, the cells find the global minima.

Similarly in other work, Chen et al applied BFOA using 6 cells to the 2D Bowl function with domain [-5,5], and the global minima was found within 50 chemotactic steps [16]. In our experiments, 6 cytobots find the global minima after an average of 9.4 evaluations, which is the equivalent to 9.4 moves (or 9.4 chemotactic steps in the terminology of Chen et al). After adjusting for the difference in domain size, the numbers of moves are highly consistent for cytobots and the cells in BFOA. In other related work, Monismith et al created the slime mould optimization algorithm inspired by the life cycle of *D. discoideum* [17]. The state space is represented as a sparse mesh which cells populate and make modifications to, for example, deposit attractant. Using a combination of behavioral states inspired by the life cycle of *D. discoideum,* artificial cells perform local searches, and move to positions in favor of their personal best and the best fitness of their neighborhood. The slime mould optimization algorithm, like the cytobots, finds the global minima of the 2D Rosenbrock function.

VI. CONCLUSIONS

The results presented above show that the agents are able to find best fitness solutions in all problems, and match the performance of cell inspired optimization algorithms in similar search spaces. Increasing the number of agents by small increments (2 or 3), can half the number of function evaluations required to find the global minima. These experiments serve as a preliminary to implementing ARN systems to control real world distributed autonomous robotic agents. Such agents could be applied to similar search problems in real world environments, for example oil spill cleanup operations, where the objective is to travel to higher concentrations of oil, while consuming it at each passing location. The cytobots obviously do not compare directly with conventional optimization techniques like Genetic Algorithms, since they have a complex internal structure. However this is not their purpose and they may be much more effectively utilized as the control systems in autonomous agents. This application demands an internal control system which can function without reference to other agents within the environment which are operating in parallel. By modifying the environment, (in this case by consumption of food), the agents can stigmergically communicate and enhance and/or facilitate emergent behavior. The cytobots offer a unique range of abilities. Like cells, their internal network of spatially distributed dynamic chemical species allows them to autonomously coordinate and direct their movement, recognize and respond to patterns in the environment, and produce high-level behavior.

In future work, it is intended to further explore the AI applications of the cytobot agents, and later, to create swarms of cytobot robots with applications in real world environments.

REFERENCES

[1] B. J. Ford, “Are cells Ingenious?,” The Microscope. vol. 52, no. 3-4, pp. 135-144, 2004.

[2] D. Bray, “Protein molecules as computational elements in living cells,” Nature, vol. 376, no. 6538, pp. 307-12, July 1995.

[3] T. Nakagaki, H. Yamada, and A. Toth, “Maze-solving by an amoeboid organism,”. Nature, vol. 407, no. 6803, pp. 470-470, September 2000.

[4] A. Arkin, J. Ross, “Computational functions in biochemical reaction networks,” Biophys. J., vol. 67, pp. 560-578, August 1994.

[5] U. S. Bhalla, “Understanding complex signaling networks through models and metaphors,” Prog. Biophys . Mol. Biol. vol. 81, no. 1, pp. 45-65, January 2003.

[6] B. Kholodenko, “Cell signaling dynamics in time and space,” Nat. Rev. Mol. Cell Biol., vol. 7, no. 3, pp. 165-176, March 2006.

[7] P. Dittrich, J. Zeigler, and W. Banzhaf, W. Artificial Chemistries- a reivew. Artifi. Life vol. 7, no. 3, pp. 225-275, 2001.

[8] C. E. Gerrard, J. McCall, G. M, Coghill, and C. Macleod, “Artificial Reaction Networks,” Proceedings of the 11th UK Workshop on Computational Intelligence, Manchester, UK, pp. 20-26, September 2011.

[9] C. E. Gerrard, J. McCall, G. M, Coghill, and C. Macleod, “Temporal patterns in Artificial Reaction Networks,” Proceedings of The 22nd International Conference on Artificial Neural Networks Lausanne, part 1, vol. 7552, pp. 1-8, September 2012.

[10] C. E. Gerrard, J. McCall, G. M. Coghill, and C. Macleod. “Adaptive Dynamic Control of Quadrupedal Robotic gaits with Artificial Reaction Networks,” Proceedings of The 19th International Conference on Neural Information Processing Doha, vol. 7663, part 1, pp. 280-287, November 2012.

[11] N. Vladimirov, and V. Sourjik, “Chemotaxis: how bacteria use memory,” J. Biol. Chem., vol. 390, no. 11, pp. 1097-1104, November 2009.

[12] R. H. Kessin, “Making Streams,”Nature, vol. 422, pp. 481-482, April 2003.

[13] S. M. Ulam and J. von Neumann, “On combinations of stochastic and deterministic processes,” Bull. Amer. Math. Soc, vol. 53, pp. 1120, 1947.

[14] V. Patidar, K. K. Sud, and N. K Pareek, “A Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing,” Informatica, vol. 33, pp. 441-452, 2009.

[15] K. M. Passino, “Biomimicry of bacterial foraging for distributed optimization and control”, IEEE Control Systems, vol. 22, no. 3, pp.52–67, June, 2002.

[16] H. Chen, Y. Zhu, and K. Hu, “Cooperative bacterial foraging optimization,” Discrete Dyn. Nat. Soc., vol. 2, no. 1, pp. 501-517, August 2009.

[17] D. R. Monismith, and B. E. Mayfield, “Slime mold as a model for numerical optimization,” IEEE Swarm Intelligence Symposium, St Louis, USA, pp. 21-23, 2008.