KIMIKA Volume 19, Number 2, pp. 77–82 (2003)

ISSN 0115–2130

Artificial chemistry: Basic concepts and application to combinatorial problems

Jaderick P. Pabico*, Elmer-Rico E. Mojica, and Jose Rene L. Micor

Institute of Computer Science
College of Arts & Sciences
University of the Philippines-Los Baños
College, Laguna 4031, PHILIPPINES

In artificial chemistry (ACHEM), the objects (molecules) are data and the interactions (reactions) among them are driven by an algorithm. An object expresses its duality as it can appear as a machine (operator) or as a data (operand). Thus an object can process other objects or it can be processed. This dualism of objects enables us to implicitly define a constructive computational procedure using chemistry as metaphor to solve complex real-world problems. In this paper we introduce ACHEM as a distributed stochastic algorithm that simulates reaction systems of algorithmic objects inspired by natural chemical systems. Then we apply ACHEM to find solutions to the traveling salesman problem. Results show that ACHEM is an example of the successful use of a natural metaphor to design an optimization algorithm.

Keywords: artificial chemistry; combinatorial optimization; Traveling Salesman Problem; TSP

INTRODUCTION

Combinatorial optimization problems such as the traveling salesman (TSP), shop floor control and scheduling (i.e., job-shop scheduling), distribution of goods and services (i.e., vehicle routing), aircraft landing and scheduling, product design (i.e., VLSI layout), DNA sequence analysis, and many others are problems whose solutions are of real-world importance. Exact algorithms have been proposed to these problems but were proven inefficient for large problem instances. Because it has been proven that these problems belong to the class of NP-hard problems [1], heuristics and metaheuristics methodologies for computation provide practical solutions for large problem instances.

Solutions to combinatorial optimization, using instances of TSP as representative problems, have been studied extensively. Graph-based heuristics such as branch and bound [2], as well as multi-agent-based and nature-inspired algorithms such as genetic algorithms [3], memetic algorithms [4–6], tabu search [7], simulated annealing [8], simulated jumping [9], neural networks [10], and ant colonies [11–13] have been used and shown to find optimal and near optimal solutions.

In recent years, the chemical metaphor, called artificial chemistry (ACHEM), has emerged as a computation paradigm [14–19]. Chemical and biochemical systems of living organisms have been shown to possess computational properties [20–22]. In an algorithmic chemistry, the objects (atoms or molecules) are data or solutions and the interactions (collisions or reactions) among objects are defined by an algorithm. The objects and their interactions to one another were used to solve several *toy* problems such as the generation of prime numbers, robot control [19], and number division [23].

In the current effort, we explored ACHEM to solve combinatorial problems using instances of TSP. We mapped Hamiltonian tours to artificial molecules, defined the cost of traversing the tours as molecular mass, and designed chemical reactions as functions for creating solutions to TSPs from

**To whom correspondence should be addressed.*

initially randomly generated molecules in a occasionally-stirred reaction tank. With these metaphors, we were able to find optimal and near-optimal solutions to TSP with the same efficiency as the known multi-agent-based heuristics.

This paper is centered on the introduction of the basic concepts of ACHEM and on its application to combinatorial optimization problems represented by instances of the TSP. Here, we will present ACHEM as a distributed approach to combinatorial optimization based on the natural chemical systems by discussing ways of how information can be processed and created by a collection of artificial molecules floating in a simulated reactor tank. We will show, via mapping of molecules to Hamiltonian tours, relationship of molecular mass to molecule's rate of reaction, and the reaction algorithm, that ACHEM can also be used to solve combinatorial optimization problems.

DEVELOPMENT OF ARTIFICIAL CHEMICAL SYSTEM

In this section, we start with a brief definition of the TSP and a discussion on the basic concepts of ACHEM with a focus on solving the TSP. We then proceed to the development of algorithms that mimic chemical reactions resulting in information processing. The processing of information happens in artificial reactor guided by reaction rules. As this simulation is inspired by the concepts of chemistry, we use the nomenclature of this science while cautioning the readers that they are analogical in nature only.

The traveling salesman problem. TSP is formally defined as the problem of finding the shortest Hamiltonian cycle of a graph $G(V, E)$. A graph is composed of a set of cities $V = \{v_1, v_2, \ldots, v_n\}$, a path set $E = \{(v_i, v_j): v_i, v_j \in V\}$, and a cost measure matrix C, where each element c_{ij} is the cost measure associated with path $(v_i, v_j) \in E$. A Hamiltonian cycle is a closed tour that visit each city once. We have a symmetric TSP if $c_{ij} = c_{ji}$. If $c_{ij} \neq c_{ji}$ for at least one $c \in C$, then we have an asymmetric TSP.

Table 1. Instances of large TSP used in this research and the approximate time each would take if solved with a deterministic polynomial time algorithm.

Number of Cities	*Cost of Best Tour*	*Time*
10	500	4.03×10^{-5} s
20	1000	312.87 days
30	1500	9.35×10^{13} years
40	2000	2.87×10^{28} years
50	2500	1.07×10^{46} years
100	5000	Can not be computed anymore

In this research, we used ACHEM to solve the TSP instances listed in Table 1. These problems have known optimal solutions and each has computational complexity of $n!$, where n is the number of cities. Exact solutions to these problems become inefficient to run when n becomes very large. For example, it would take about 1.07×10^{43} millenniums for a deterministic polynomial time algorithm running on a 90 GHz Pentium 4 PC to find all solutions to a 50-city TSP. This means that there is no deterministic polynomial time algorithm that so far exists that can solve TSP efficiently. Thus, we can only use nondeterministic polynomial (NP) time solutions such as ACHEM to solve the TSP efficiently. This is the reason why combinatorial optimization problems such as the TSP is said to belong to NP class of problems.

Basic concepts of artificial chemistry. In the physical world, chemical reactions happen under specific physical and structural conditions. Molecules carry some information specific to their composition (e.g., molecular weight and molecular structure) while the reaction between molecules causes changes to the composition of the reacting molecules. With this idea in mind, we see the composition of molecules as a kind of information storage while the reaction between them as a kind of information processing. The more molecules involved in the reaction and the faster the reaction, the more information is processed. Therefore, we can create an abstract system, similar to chemical systems, which is capable of information storage and processing.

Formally, ACHEM is defined by a triple (M, R, A) where M is a set of artificial molecules, R is a set of reaction rules describing the interaction among molecules, and A is an algorithm driving the ACHEM system. The molecules in M may be abstract symbols [24], strings of characters [25–27], λ-expression [15], binary strings [19, 28], numbers [18], or proofs [29]. In this paper, we introduce Hamiltonian tours as molecules that store solutions to TSP.

The rules in R can be defined explicitly [24] or implicitly by using string matching and string concatenation [25, 27, 30], λ-calculus [15, 31], Turing machines [20], finite state machines or machine language [19], proof theory [31], matrix multiplication [28], or simple arithmetic operations [18]. In this paper, we present our reaction rule as a reordering algorithm that creates new molecules when other molecules collide.

The algorithm A describes how the rules are applied to a *soup* of artificial molecules. The algorithm may simulate a well-stirred abstract reaction tank (no topology) [15, 19, 25], an Euclidean discrete reaction vessel [24, 30], a continuous 3D space [32], or a self-organizing topology [33]. In this effort, our algorithm A simulates a topology-less reaction tank that partitions the *soup* into levels of reaction activities as a function of molecular mass.

Application of ACHEM to TSP. The vertices $v_i \in V$, $\forall\ i = 1, \ldots, n$ are considered as the set of atoms in the n-city TSP abstract world. These atoms exist in stable molecular forms that can be considered as Hamiltonian cycles. The set of artificial molecules M is the set of Hamiltonian cycles. Each of the molecules $m \in M$ is a fixed-length n-ary string $m = \{0|1|\ \ldots|n\}^n$ with the constraint that m contains only the n permutation of cities taken n. This constraint assures us that m encodes a valid Hamiltonian cycle. The cost f of traversing the Hamiltonian cycle (Equation 1) is a function of the cost matrix C and can be regarded as the molecular mass of m. The molecular mass is directly proportional to the excitation energy of the molecule.

$$f = c_{n,1} + \sum_{i=1}^{n-1} c_{i,i+1} \qquad (1)$$

The reaction rules are all zero-order reversible (i.e., non-catalytic) reactions of the form $m_1 + m_2 \rightarrow m_3 + m_4$. All collisions of two molecules m_1 and m_2 have unique outcomes, m_3 and m_4. Each collision can be represented as a function R: $M \times M \rightarrow M \times M$. However, if the products of the reaction are the same as the reactants (i.e., $m_1 + m_2 \rightarrow m_1 + m_2$) then we have an elastic collision.

Similar to the cycle crossover in genetic algorithms [34], the reaction rule R performs reordering under the constraint that each city comes from one reactant or the other. The reaction rule is described in Algorithm 1.

Algorithm 1. Reaction Rule

1. Let an integer $l \in [1,n]$ be the index of the city encoded in any molecule m. The indexing order does not matter (i.e. whether the index goes from left to right or vice-versa) as long as we are consistent throughout this algorithm.
2. Take a random integer between 1 and n and assign it to l. Let $l^0 = l$.
3. Taking the reactant m_1, locate the lth atom in m_1 and move it as the lth atom for m_3.
4. Take note of the lth atom in m_2 and locate it in m_1. Replace the value of l with the index of the atom found in m_1.
5. Repeat steps 3 to 4 until the lth atom in m_2 is the same as the l^0th atom in m_1.
6. For all indeces l with no atoms yet in m_3, move the lth atom from reactant m_1 as the lth atom in product m_3.
7. Repeat steps 2 to 6 for reactant m_2 and product m_4.

In the above reaction rule, an elastic collision of the form $m_1 + m_2 \rightarrow m_1 + m_2$ happens when the stopping criterion described in step 5 is reached during the first iteration.

The reactor algorithm A operates on a *soup* of molecules $S = \{m_1, \ldots, m_{\|S\|}\}$, $\| S \| << \| M \|$. The development of S is realized by iteratively applying steps 2 to 4 of the following algorithm:

Table 2. Parallelism of real chemistry and ACHEM and the respective notations used.

Real Chemistry	*ACHEM*	*Notation*
Set of atoms	Set of cities	M
An atom	A city	V
A molecule	A Hamiltonian cycle	M
Molecular mass	Cost of tour	F
Reaction	Algorithm 1	R
Universe	Soup	S
Chemical system	Algorithm 2	A

Algorithm 2. Reactor Algorithm

1. Initialize the soup with $\| S \|$ molecules selected randomly from M.
2. Using stochastic sampling with replacement, select two molecules m_1 and m_2 from S without removing them.
3. Apply the reaction rule in Algorithm 1 for the two reactants m_1 and m_2 to produce the products m_3 and m_4.
4. Decay the heavier molecules by removing them out of S and replace them with randomly selected molecules from M.
5. Repeat steps 2 to 4 until the soup is nearly saturated with molecules of lower molecular mass.
6. What remains is a soup of molecules that encode optimal or near-optimal solutions to TSP.

One iteration of steps 2 to 4 will constitute one epoch in our ACHEM simulation time. The sampling procedure of step 2 of Algorithm 2 gives molecules with low molecular mass a higher probability to react or collide with other molecules. This mimics the level of excitation energy the molecule needs to overcome for it to react with another molecule. This means that the lighter the molecule, the higher the chance that it will collide with other molecules. In regards to TSP, the lower the cost of the Hamiltonian tour, the higher the chance that it will interact with another tour to create a new pair of tours. The parallelism between ACHEM and real chemistry and the list of notations used in this paper are summarized in Table 2.

RESULTS AND DISCUSSION

ACHEM efficiency and quality of solutions found. Table 3 shows the minimum cost of Hamiltonian cycle and the actual computing time it took to solve each instance of the TSP. ACHEM found the exact best tour in 86 sec for the 10-city TSP. At small problem instances such as the 10-city TSP, ACHEM is inefficient (the deterministic polynomial time algorithm can solve this problem in only 4.03×10^{-5} second on the same PC). However, for large problem instances, ACHEM was able to find near-optimal solutions under reasonable amount of time.

Table 3. The quality of solutions found by ACHEM and ACHEM's efficiency.

Number of Cities	*Cost of Best Tour*	*Cost of Tour Found by ACHEM*	*ACHEM Running Time (s)*
10	500	500	86
20	1,000	1,013	520
30	1,500	1,533	613
40	2,000	2,053	730
50	2,500	2,567	815
100	5,000	5,180	1,360

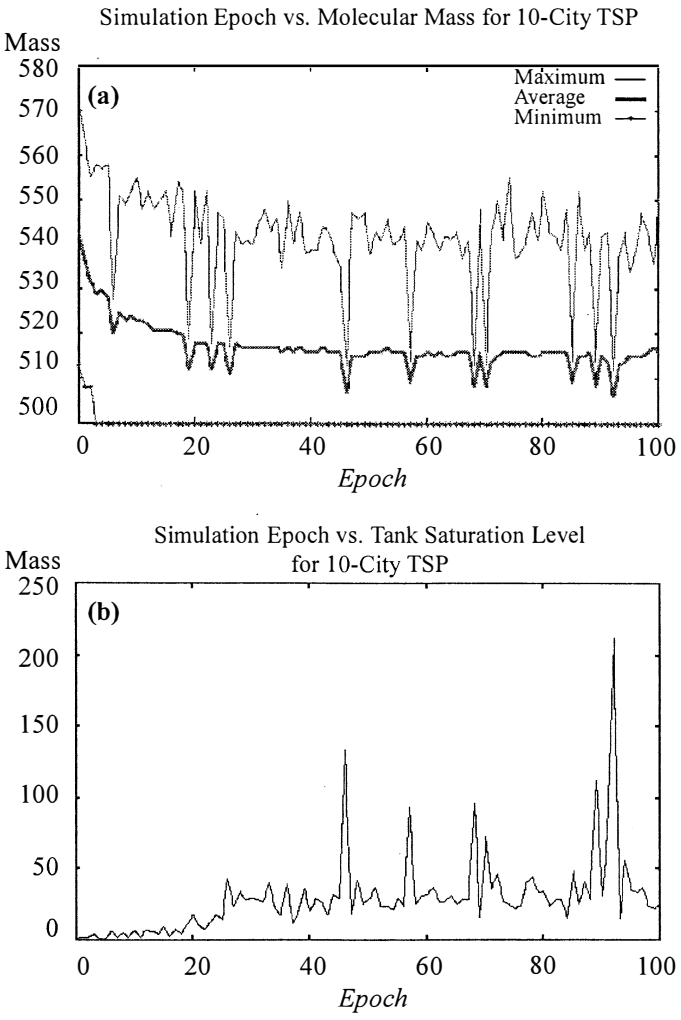

Fig. 1. The development of the soup in a reactor tank that solves the 10-City TSP.

Soup development. Figure 1a shows the development of the soup in a reactor tank that solves the 10-City TSP. The exact optimum Hamiltonian cycle was found at the fourth simulation epoch while the soup was developed into having molecules that store better Hamiltonian cycles. The downward spikes in the maximum line shows that during the simulation, the soup was almost saturated with molecules of lower molecular mass. This observation was collaborated by Figure 1b that shows the

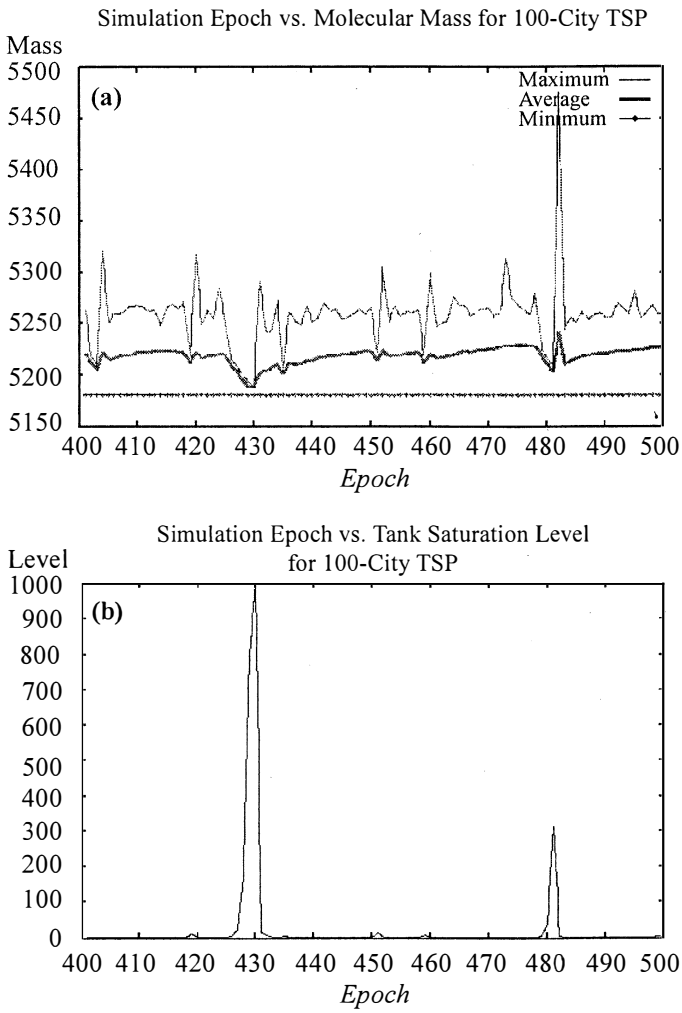

Fig. 2. The development of the soup in a reactor tank that solves the 100-City TSP.

tank's saturation level of molecules with lower molecular mass. The peaks in the graph coincide with the downward spikes in the maximum line of Figure 1a. This signifies that ACHEM was able to find other Hamiltonian cycles of almost optimal costs during the simulation epoch where the peaks and spikes occur.

The respective development of the soup for the 20- to 100-City TSPs, respectively, show the same behavior with that of Figure 1a indicating that ACHEM is robust and its efficiency is independent on the size of the problem being solved. Similarly, the tank's saturation level of molecules with lower molecular mass for 20- through 100-City TSPs suggest that ACHEM was able to find other solutions of similar minimal costs despite of the varying size of the problem (see Figure 2 showing only the last 100 epochs for the 500-epoch run for the 100-City TSP).

Effects of reversible reaction. The suddenness of peaks in saturation graphs presented above may be attributed to the nature of the reaction rule used in the ACHEM experiments. A

soup of molecules saturated with higher molecular mass, depending on the frequency of collisions among molecules, may suddenly develop into a soup saturated with molecules of lower molecular mass at one time and then revert into a soup of molecules with higher molecular mass at another time. Take for example a hypothetical universe consisting only of four atoms $\{x_1, x_2, x_3, x_4\}$ where only two atoms exist at a time following a reversible chemical reaction of the form $x_1 + x_2 \rightarrow x_3 + x_4$. Let the atoms x_1 and x_2 have low molecular mass and the atoms x_3 and x_4 have high molecular mass. If the frequency of collision between the atoms is high (i.e., the universe is small), the universe will be experiencing a series of alternating shifts between sudden low and sudden high masses. If the frequency of collision between atoms is low (i.e., the universe is big), the universe will be experiencing a series of alternating shifts between plateuing low and plateuing high masses.

To avoid the disrupting effects of the reaction rule on the solutions being solved by the artificial chemical system, it is recommended that the reaction rule be designed to keep the reactants to the product side of the reaction equation. For example, a second-order catalytic reaction rule of the form $x_1 + x_2 + X \rightarrow x_1 + x_2 + x_3 + x_4$ that can initiate a mass-action kinetics might be a better one than the rule that we used in this study. Here, the concentration of the implicit substrate X might be kept constant. The production of new atoms x_3 and x_4 might create an implicit competition for space which may lead to an evolutionary process. We will deal with this kind of reaction rule in our future research.

SUMMARY

In this paper, we have shown that artificial chemical objects can store information while the reactions among them can initiate information processing. We have designed an artificial chemical system that is capable of solving large instances of combinatorial optimization problems such as the TSP. By giving computational metaphor to molecular properties as solutions and to molecular reactions as ways to create new solutions, our ACHEM system was able to develop an artificial soup of molecules in a reactor tank that store optimal and near-optimal solutions to TSP. We have shown that ACHEM can find quality solutions to TSP within a reasonable amount of time.

REFERENCES

1. Garey, M. R. and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, 1979).

2. Tschoke, S., Luling, R., and Monien, B. *Proceedings of the 9th International Parallel Processing Symposium*, p. 182. (1995).

3. Pabico, J. P. and Albacea, E. A. *Proceedings of the Workshop and Conference on Modeling, Simulation, and Scientific Computing (Model 99).* (19–20 November 1999, Ateneo de Manila University, 1999).

4. Moscato, P. and Norman, M. G. *In* Valero, M., Onate, E., Jane, M., Larriba, L., and Suarez, B. *Parallel Computing and Transputer Applications*, p. 187. (IOS Press: Amsterdam, 1992).

5. Freisleben, B. and Merz, P. *In; Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, p. 616. (1996).

6. Freisleben, B. and Merz, P. *In* Voigt, H.M., Ebeling, W., Rechenberg, I. And Schwefel, H.P. *Proceedings of the 4th Conference on Parallel Problem Solving from Nature*, p. 890. (Springer, 1996).

7. Zachariasen, M. and Dam, M. *In Metaheuristics International Conference 95, Breckenridge, CO.* (1995).

8. Martin, O.C. and Otto, S.W. *In* Laporte, G. and Osman, I. *Metaheuristics in Combinatorial Optimization, Annals of Operations Research*, Vol. 63, p. 57. (Amsterdam, 1996).

9. Amin, S. *Annals of Operations Research.* 86, 23 (1999).

10. Miglino, O., Menczer, D., and Bovet, P. *Journal of Biological Systems.* 2 (3), 357 (1994).

11. Gambardella, L. M. and Dorigo, M. *In Proceedings of the Twelfth International Conference on Machine Learning*, p 252. (Tahoe City, CA, 1995).

12. Gambardella, L. M. and Dorigo, M. *In Proceedings of the IEEE International Conference on Evolutionary Computation,* p. 622. (Nagoya, Japan, 1996).

13. Dorigo, M. and Gambardella, L. M. *BioSystems.* 43, 73 (1997).

14. Berry, G. and Boudol, G. *J. Theoretical Computer Science.* 96, 217 (1992).

15. Fontana, W. *In* Langton, C.G., Taylor, C., Doyne-Farmer, J., and Rasmussen, S. *Proceedings of the Workshop on Artificial Life*, Vol. 88, p. 159. (Addison-Wesley, 1992).

16. Banzhaf, W. *In* Banzhaf, W. and Eeckman, F. H. *Evolution and Biocomputing*, Vol. 899, p. 69. (Springer, Berlin, 1995).

17. Ikegami, T. and Hashimoto, T. *In* Moran, F., Moreno, A., Merelo, J. J., and Chacon, P. *Advances in Artificial Life: Proceedings of Third European Conference on Artificial Life*, Vol. 929, p. 234. (Springer-Verlag: Berlin, 1995).

18. Banzhaf, W., Dittrich, P., and Rauhe, H. *Nanotechnology.* 7 (1), 307 (1996).

19. Dittrich, P., Banzhaf, W., Rauhe, H., and Ziegler, J. *In* Dittrich, P., Rauhe, H., and Banzhaf, W. *Proceedings of the Second German Workshops on Artificial Life*, p. 19. (University of Durtmond, 1998).

20. Hjemfelt, A., Weinberger, E. D., and Ross, J. *In: Proceedings of National Academy of Sciences of the United States of America.* 88 (24), 10983 (1991).

21. Adleman, L. M. *Science.* 26, 1021 (1994).

22. Arkin, A. and Ross, J. *J. Biophysics.* 67 (2), 560 (1994).

23. Dittrich, P. *In* Nehaniv, C. L. and Wagner, G. P. *The Right Stuff: Appropriate Mathematics for Evolutionary and Developmental Biology,* Technical Report No. 315, p. 27. (University of Hardfordshire School of Information Science, 1998).

24. Varela, F. J., Maturana, H. R., and Uribe, R. *BioSystems.* 5 (4), 187 (1974).

25. Bagley, R. J. and Farmer, J. D. *In* Langton, C.G., Taylor, C. Doyne-Farmer, J., and Rasmussen, S. *Proceedings of the Workshop on Artificial Life v. 5 of Santa Fe Institute Studies in the Sciences of Complexity*, p. 93. (Addison-Wesley, 1992).

26. Kauffman, S. A. *The Origins of Order* (Oxford University Press, 1993).

27. McCaskill, J. S., Chorongiewski, H., Mekelburg, D., and Tangen, U. *International J. of Physical Chemistry.* 98, 1114 (1994).

28. Banzhaf, W. *Biological Cybernetics.* 69, 269 (1993).

29. Fontana, W. and Buss, L. W. *In* Casti, J. L. and Karlqvist, A. *Boundaries and Barriers: On the Limits to Scientific Knowledge*, p 56. (Addison-Wesley, 1996).

30. Lugowski, M. W. *In* Langton, C. G. *Artificial Life.* (1989).

31. Fontana, W. and Buss, L. W. *In Proceedings of the National Academy of Science.* 91 (2), 757 (1994).

32. Zauner, K. P. and Conrad, M. *In* Hofestadt, R., Loffler, M., Lengauer, T., and Schomburg, D. *Computer Science and Biology - Proceedings of the German Conference on Bioinformatics*, p. 336. (Universitat Leipzig, Germany, 1996).

33. Dittrich, P. and Banzhaf, W. *In* Husbands, P. and Harvey, I. *Fourth European Conference on Artificial Life, University of Sussex, Brighton, UK.* (MIT Press, 1997).

34. Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning.* (Addison-Wesley, 1989).