(World Congress on Computational Intelligence (WCCI'94))
Orlando, FL, USA, 1994
IEEE Press, Piscataway
Vol. II, pp. 651 — 655

Artificial Selection in a System of Self-Replicating Strings

Wolfgang Banzhaf

Department of Computer Science, Dortmund University

Baroper Str. 301, 44221 Dortmund, GERMANY

banzhaf@tarantoga.informatik.uni-dortmund.de

Increasingly, Artificial Life (AL) models are introduced to study various aspects of the formal foundation of the phenomenon of life [1]. Metabolism and the ability to self-replicate have long been considered as the most prominent of these aspects [2, 3, 4, 5]. It is natural to ask, how AL-systems might be used in application problems of today. Since AL-systems are based on competition between entities, which brings about selection of some entities that are more adapted than others (provided there are differences at all among entities), an evident idea is to artificially select for those entities that are useful from an outside perspective.

We implement this idea in the context of a recently proposed a simple model of self-replicating strings [6]. These strings form an ensemble called the string soup, in which they can encounter each other and react according to predefined reaction pathways to produce other strings. Details may be found in [7, 8].

The system is self-organizing in that every string comes in two alternative forms, its one-dimensional information holding string form, and in a second form, capable of operating on the first form of itself and of other strings. Thus, we exploit the physical identity between what is being organized (operands) and what organizes (operators) in order to ensure self-organization.

Figure 1 depicts the situation in a general setting. Each string is allowed to encounter every other string, much like chemical reactions occur in a well-stirred vessel. Strings are of different sorts, represented by different sequences of symbols, with some types coming in larger numbers, others in smaller. Regardless of the initial distribution of string sorts, those string types which are produced more often by reactions tend to dominate the string soup in the course of time, whereas string types which are produced seldomly tend to die out.

As in the past, we apply the following simplifications:
(i) Our set of symbols is restricted to the binary numbers $s_i \in 0, 1$
(ii) The length N of strings $\vec{s}$ in number of components $s_i, i = 1, ..., N$ is fixed at some square number
(iii) Our operators $\mathcal{P}$ are two-dimensional quadratic matrices, formed from binary sequences

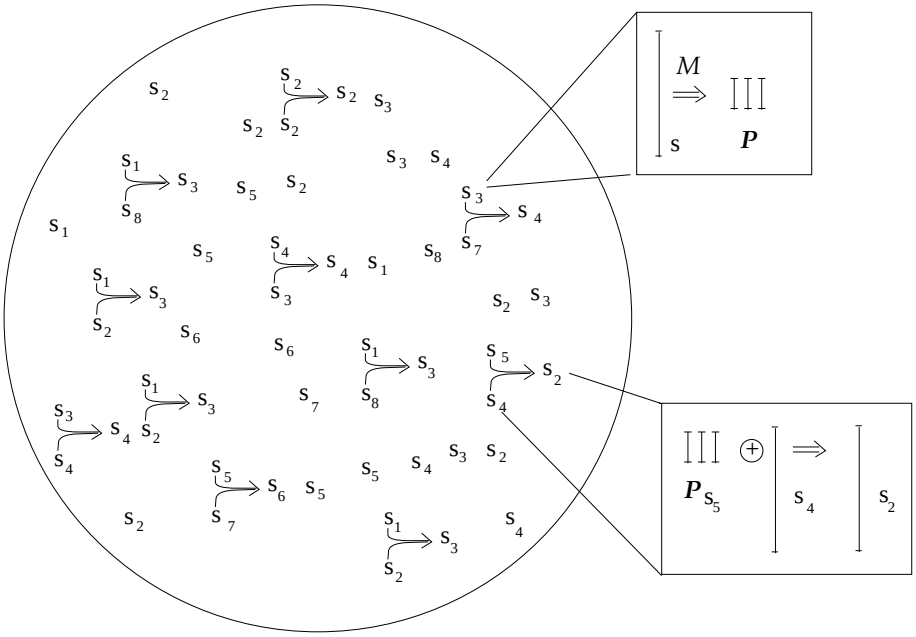

Figure 1: Strings encounter each other and react to produce new strings. Details are shown in the inset. Production of new strings and destruction of old strings are balanced in a competitive system.

according to a certain canonical folding rule $\mathcal{M}$.
(iv) Our interaction is a piecewise scalar product followed by taking a non-linearity.

Using this set of simplifications, we have the following formalism:
A mapping $\mathcal{M}$: $\vec{s} \to \mathcal{P}_{\vec{s}}$ and an interaction $\mathcal{P}_{\vec{s}} \oplus \vec{s}' \to \vec{s}''$ with

$$s''_{i+k\sqrt{N}} = \sigma \left[\sum_{j=1}^{j=\sqrt{N}} \mathcal{P}_{\vec{s}\ ij}\ s'_{j+k\sqrt{N}} - \Theta \right] \tag{1}$$

$$i = 1, ..., \sqrt{N} \qquad k = 0, ..., \sqrt{N} - 1$$

$$\sigma[x] = \begin{cases} 1 & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases} \tag{2}$$

and Θ is an adjustable threshold, here fixed at $\Theta = 1$.

It was noticed that we had to counter the excessive over-production of certain pathological string types. Specifically, the 0-string (destructor) consisting only of 0s, and the 1-string (exploitor) consisting only of 1s had to be counterbalanced. We did so by substituting all destructors and, somewhat more gently, by substituting on a probability basis strings with high content of 1s. In this way having reached a balanced system we examined and reported its autonomous behaviour, also studying the effect of a possible mutation rate that acts by hitting each binary number in a string with an equally low probability [9]. The overall algorithm is shown in Figure 2.

So far, events in the system are autonomous, with reactions going on indepent of an outside observer. As in other artificial life systems, we have established interactions between entities that follow their own rules and determine their own agenda. In order to make some use of the dynamics, we have to introduce a semantics giving meaning to the information that resides in each of the strings in the soup. Different strings would then mean different things, and newly produced string types could mean new things to the outside observer. This is the moment when artificial selection is coming in. Natural selection is already present in the system and

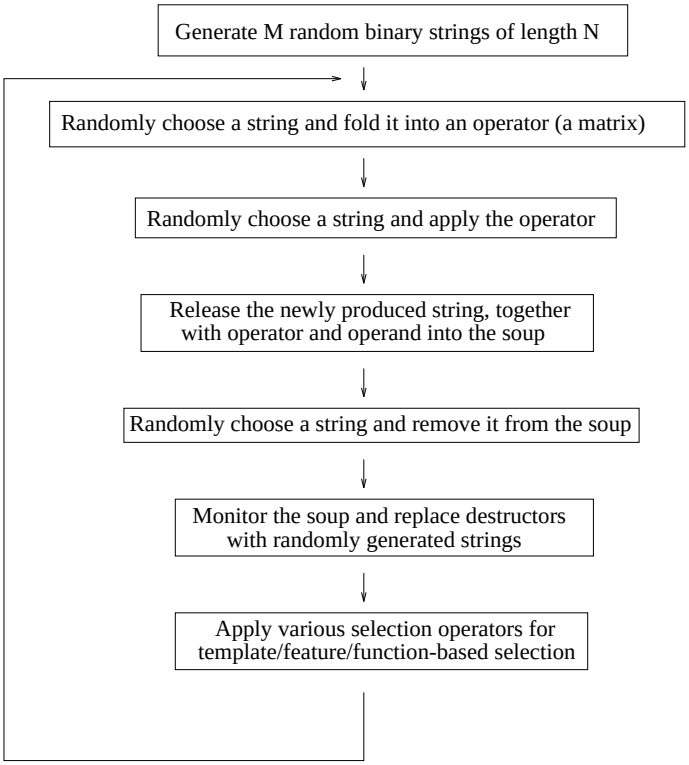

Figure 2: Flow diagram of one sweep through the algorithm. M sweeps (M: size of the string population) constitute one generation.

is brought about by the competition of string types for place in the soup (space is limited). Artificial selection can now be added to direct the evolution of the soup composition.

This can be done most generally by applying independent selection operators to strings. These operators should be equipped with a different type mapping $\mathcal{S}$. The selection operators interpret each string according to this mapping, thereby bringing in relevant outside information. For instance, they could interpret $\vec{s}$ as a solution to an optimization problem one is interested in solving:

$$\mathcal{S} : \vec{s} \rightarrow \vec{x} = \{x_1, ..., x_L\}^T \tag{3}$$

with L being the dimension of the optimization problem and x_i a suitable component of a solution. $\mathcal{S}$ must not be the same for all of the selection operators. Simple examples are templates or features.

After application of this mapping which generates a solution to the problem that is to be solved, the selection operator computes a measure of quality $Q(\vec{x})$ of the resulting solution (sometimes these two steps are lumped together in a single mapping). In order to direct the evolution towards strings representing advantageous solutions, the selection operator then destroys a less advantageous solution. Less advantageous solutions are solutions which are inferior in quality (either on an average or best-of-so-far basis) to the solutions already encountered by the selection operator.

In order to apply this selection pressure in an arbitary direction, it is necessary to substitute the destroyed low-quality string by a variant of another string. Thus, in a third step, the selection operator picks a new string from the soup and replaces the destroyed one by a (slightly changed version of this string). In low-dimensional systems, a random string may be substituted as well. This procedure is necessary in systems with a fixed number of components, since there are no possibilities for strings with low production rate, yet favored by selection operators, to increase their concentration.

We have done many simulations with varying selection pressure. Here, we shall report only two experiments in a low-dimensional system that illustrates the key point: Indepedent action of selection operators. The system we have been using for both experiments is a population of $M = 10000$ binary strings of dimension $N = 4$ reacting according to equation (1). The initial configuration of the soup is generated by using random strings.

Figure 3 was a run with template-based selection, looking for 4-bit strings that carry a 1 in its inner components, $\vec{s} = (*, 1, 1, *)$, where $*$ means either 0 or 1 is allowed. Consequently, strings carrying this signature were amplified more and more strongly, depending on the selection pressure characterized by the number of independently acting selection operators n_{sel}.

Figure 4 was a run using two different selection operators. For simplicity, one was used to compute a scalar quality measure $Q(\vec{s})$ directly from the binary sequence:

$$Q(\vec{s}) = \sum_{i=1}^{4} s_i \cdot 2^{i-1} \tag{4}$$

Each quality selection operator stored the average quality $\bar{Q}$ of string encountered so far, and destroyed strings that exceeded $\bar{Q}$, effectively searching for the string with minimal Q. The other one was implementing a constrained into this optimization process by checking for the hypothetical feasibility of a solution. Only those strings should be feasible that carried a sequence (0,1,1,0) or (1,0,0,1). In this way, only two strings were allowed to pass the feasibility test.

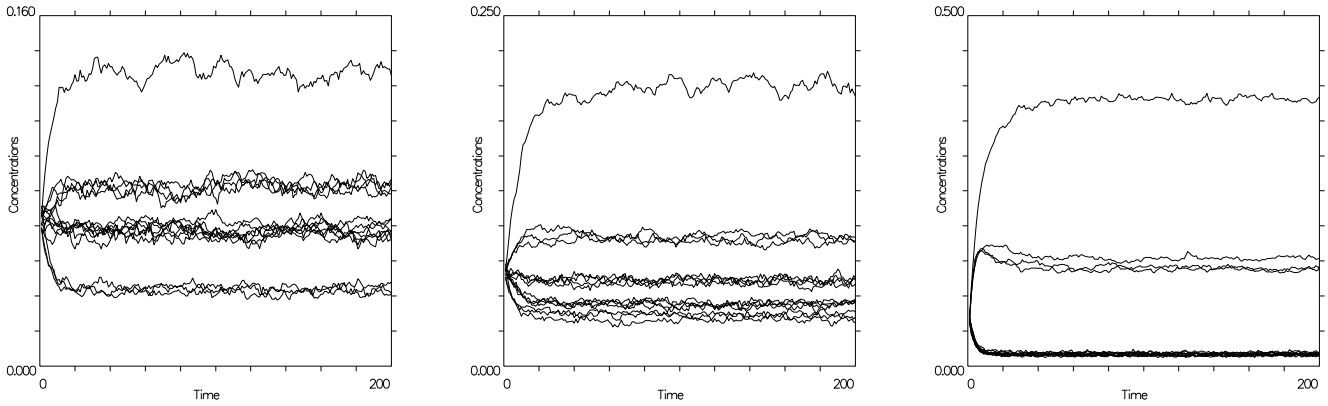

Figure 3: Increasing selection pressure with template selection for strings with (*,1,1,*). Left: $n_{sel} = 1$; Middle: $n_{sel} = 2$; Right: $n_{sel} = 10$. Four strings were amplified rather strongly: $s^{(15)}, s^{(14)}, s^{(7)}, s^{(6)}$, with $s^{(15)}$ dominating. (String names derive from converting the binary sequence into numbers to base 10.

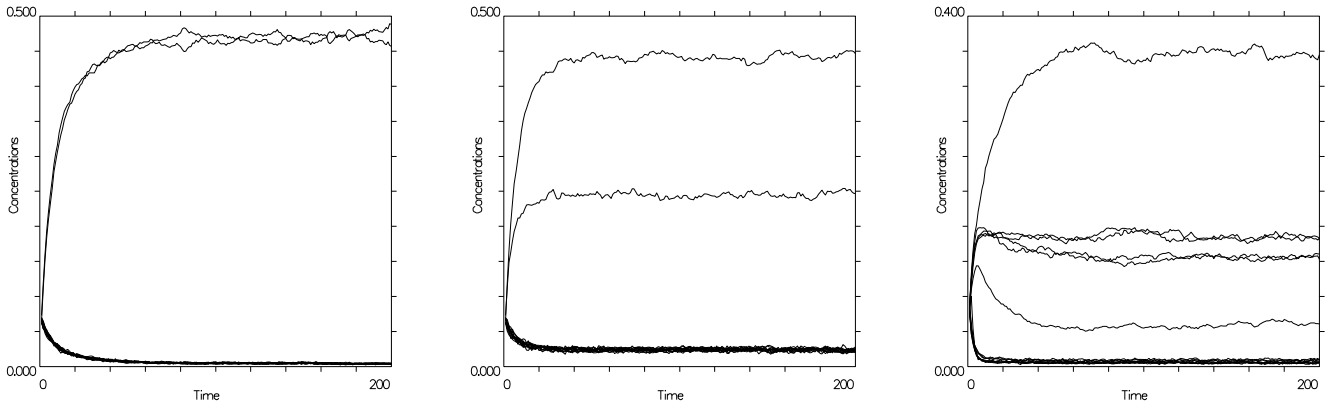

Figure 4: Varying selection pressure with two different types of selection operators: Feasibility test and quality test (see text). Left: $n_{sel}(F) = 10, n_{sel}(Q) = 0$, strings $s^{(6)}$ and $s^{(9)}$ dominate; Middle: $n_{sel}(F) = 10, n_{sel}(Q) = 1$, string $s^{(6)}$ dominates, followed by $s^{(9)}$; Right: $n_{sel}(F) = 0, n_{sel}(Q) = 10$, $s^{(1)}$ dominates,followed by $s^{(2)}$,$s^{(4)}$ and $s^{(3)}$,$s^{(5)}$

References

[1] C.G. Langton, C. Taylor, J.D.Farmer, S. Rasmussen (Eds.), *Artificial Life II.* Addison-Wesley, Redwood City, CA, 1991

[2] A.I. Oparin, *Genesis and Evolutionary Development of Life*, Academic Press, New York, 1968

[3] L. Orgel, *The Origins of Life*, John Wiley, New York, 1973

[4] F. Dyson, *Origins of Life*, Cambridge Univ. Press, Cambridge, 1985

[5] M. Eigen, *Steps toward Life: a perspective on evolution*, Oxford University Press, 1992

[6] W. Banzhaf, *Self-replicating sequences of binary numbers*, Computers and Mathematics, **26** (1993) 1

[7] W. Banzhaf, *Self-replicating sequences of binary numbers — Foundations I: General*, Biological Cybernetics, **69** (1993) 269

[8] W. Banzhaf, *Self-replicating sequences of binary numbers — Foundations II: Strings of Length $N = 4$*, Biological Cybernetics, **69** (1993) 275

[9] W. Banzhaf, *Self-replicating sequences of binary numbers — Foundations III: Larger Systems*, Biological Cybernetics, submitted