Optimization as Side-Effect of Evolving Allelopathic Diversity

Ludo Pagie[1,2] and Paulien Hogeweg[1]

[1] Bioinformatics Group, Utrecht University,
Padualaan 8, 3584 CH Utrecht, the Netherlands
{l.pagie, p.hogeweg}@bio.uu.nl
[2] Current address: Santa Fe Institute,
1399 Hyde Park Road,
Santa Fe, NM 87501, USA
ludo@santafe.edu

Abstract. Many bacteria carry gene complexes that code for a toxin-antidote pair, e.g. colicin systems. Such gene complexes can be advantageous for its host by killing competitor bacteria while the antidote protects the host. However, in order to evolve a novel and useful toxin first a proper antidote must be evolved.
We present a model of bacteria that can express and evolve such allelopathic systems. Although in the model novel types must evolve from existing types we find that nevertheless in general a high diversity of toxins evolves and, as a side-effect thereof, generalized immunity mechanisms.
We interpret the allelopathic systems in terms of an optimization problem: fitness cases are toxins and solutions present (potential) antidotes. As a side-effect of the evolution of allelopathic systems generalized solutions of the optimization task are evolved as well.

1 Introduction

Many bacteria, such as *Escherichia Coli* and related bacteria, carry colicin systems [7, 17]. Colicin systems are gene complexes that code for a toxic protein, i.e. a bacteriocin, and an antidote protein. The bacteriocin kills competitor bacteria, the immunity protein protects the bearer of the colicin system against the toxic protein. Different types of colicin systems exist. A type is defined as a unique combination of a toxic and an antidote pair. In general, different types of colicin systems are cross-sensitive; antidotes are specific for the corresponding toxin.

In natural circumstances colicin systems are often found in bacterium communities [4, 19] in a high diversity. Experimental data [1, 19] show that bacteria are often insensitive to many toxin types; through expression of the corresponding antidote or by more general means, for instance by lacking specific membrane receptors through which the toxin enters the cell. Many bacteria produce at least one toxin, but often bacteria are insensitive to many more toxin types than the number of toxins that they produce.

We have studied a spatial model of the evolution of colicin systems in a bacterium population in which the toxin gene and the antidote gene mutate independently [14]. In the model the number of different types of colicin easily increases, resulting in a high

diversity of colicins. The different colicin types that are present in the population occurs in two modes which we call the *individual-based mode* and the *population-based mode*, dependent on the cost per colicin system. In both modes high numbers of colicin types can evolve and be maintained at the level of the population.

The characteristics of the first mode, the individual-based mode which occurs when costs per colicin are relatively low, agree very well with the experimental data that are obtained from natural bacterium populations [1, 19]. All bacteria are insensitive to all toxins that are present in the population[1], but produces only a very small number of toxins. In this mode the bacteria can be said to evolve general 'immunity' capabilities, i.e. expression of antidotes against all possible colicin types. Moreover, this general behavior evolves although every individual bacterium sees only a few types of toxin in its lifetime. We have used the term *information integration* to denote the evolution of general strategies under such *sparse fitness evaluation* regimes [12].

The population-based mode, occurring when colicins impose high costs on their host, is characterized by a heterogeneous bacterium population, in which many bacterium groups exist that carry different sets of *complete* colicin systems. In the population-based mode each bacterium carries much less colicin types than the bacteria in the individual-based mode. The heterogeneity of the bacterium population, however, results in an equal number of colicin types being present in the bacterium populations in the individual-based mode and in the population-based mode. In the latter mode the individual bacteria cannot evolve general immunity, due to growth rate limitations. That means that they are sensitive to many of the toxins produced by other bacteria. Rather than maximizing the number of antidotes the individual bacteria optimize their ability to kill other bacteria, i.e. maximize the number of toxins that they produce. The effect is that the different groups of bacteria maintain a standoff based on *mutual killing*: bacteria of each group are sensitive to toxins that are produced by bacteria of other groups, but they also express toxins to which other bacteria are sensitive.

In the model described above we used a simple genotype-phenotype mapping [14]; a colicin system was defined by two genes which were either active or inactive. New colicin systems arose through single mutations; existing colicin systems decayed, where each of the two genes could decay independently. Actually, the two genes of a colicin system are independent and thus are expected to evolve independently as well. But then, when a novel toxin evolves within a host the host should express the antidote corresponding to that toxin before actually producing it. A mechanism for this process was proposed by Riley [18] and is based on a diversification of immune functions of colicin systems. Once a colicin system has acquired an immune function that extends immunity that is required for the corresponding toxin the latter can change such that the original antidote no longer deactivates the new toxin. Some experimental evidence for this mechanism was presented by Tan & Riley [20].

In this paper we report on a study of the evolution of allelopathy using a more realistic genotype-phenotype mapping (GP-mapping) than in our previous studies. Novel toxins and antidotes must evolve on the basis of existing ones, rather then being created anew in a single mutation event. In conjunction to studying the evolution of novel

[1] Here, large scale insensitivity is brought about by colicin systems with "defective' toxin genes, rather than by means such as lacking certain membrane receptors.

allelopathic systems from existing ones using a more realistic GP-mapping we want to investigate if the evolution of allelopathy can be used as a search algorithm to find general solutions of optimization problems. In the individual-based mode described above the bacteria evolved general immunity repertoires. From the point of view of search algorithms we can interpret the immunity capability as a solution to the problem of protection against toxins in the environment.

The GP-mapping that we use in this model is based on a well studied optimization problem, the *density classification task* for cellular automata [2, 6, 9, 10, 15, 16]. The genes that code for toxins and antidotes are represented by bit strings. To determine whether an antidote gene confers immunity against a given toxin we map the antidote-toxin pair to a cellular automaton lookup table (CA) and an initial condition (IC). We iterate the CA starting with the IC for a fixed number of iterations. The final state of the CA is interpreted as a classification of the IC. Only if the classification is correct, given an a priori defined criterion, does the antidote confer immunity.

Thus, the evolutionary search algorithm in our model is as follows. Bacteria carry a 'potential solution' to confer immunity against fitness cases, or toxins, which it expresses itself or which are expressed by neighboring bacteria. Only if the potential solution is correct for a given fitness case is the host of the solution 'insensitive' to the toxin. Note that in this model there is not an explicit notion of fitness; bacteria can only 'use' the solution to defend themselves against toxins, and they can 'use' toxins to kill neighboring bacteria. New solutions are searched for (and found) *before* they are presented with challenging fitness cases. Search is not only blind it is also unguided.

We find nevertheless that the bacteria in the model evolve cellular automata that show general density classification behavior as a side-effect of being exposed to a diverse toxic environment. The bacteria experience a pressure to maintain immunity against the toxins that are present in their neighborhood, and at the same time explore novel antidotes and novel toxins. The exploration in an already diverse, and hazardous environment apparently is sufficient to evolve general density classifiers.

Whereas in evolutionary search algorithms the population size is generally constant in our model the population size is variable. Thus, the bacteria can die out if they do not form a viable population. In addition, the population in our model is embedded in space and individuals interact only with other individuals which are in their direct neighborhood.

2 The Model

Firstly, we will describe the model at the level of the bacteria and their interactions. After that we will describe the genotype-phenotype mapping, i.e. the density classification task, in more detail.

2.1 A Model of Bacteria and their Allelopathic Interactions

The model that we study here is closely related to the model that we studied in [14]. The bacteria and their interactions are modeled in an individual-based, discrete space, discrete time model, using synchronous updating. The bacteria live in a square grid of

50 by 50 cells, with periodic boundary conditions. Each cell contains a single bacterium or it is empty. Bacteria can colonize empty cells, they produce 'toxins', and they can die. The interactions between bacteria, and the colonization of empty cells by neighboring bacteria are defined in the Moore neighborhood, i.e. a central cell plus the eight cells that are its direct neighbors. Finally, the allelopathic systems carried by the bacteria undergo mutations with rates μ_{ca} and μ_{ic} (see below).

The probability that an empty cell is colonized is equal to the sum of the growth rates of the bacteria in its Moore neighborhood. For the simulations that we will discuss below the growth rate of bacteria is fixed at $g = 0.125$. Bacteria die with a rate $d_w = 0.1$. The death rate is increased if a bacterium is sensitive to toxins that are produced by itself or by neighboring bacteria. Per toxin for which a bacterium is sensitive its death rate is increased with $d_i = 0.3$.

The allelopathic systems that the bacteria carry are represented by a single CA, the antidote repertoire, and two ICs, the toxins. A CA can implement immunity to a few specific ICs, or toxins, but also implement a general immunity against many toxins. The latter strategy concurs with a CA that shows a high performance in the density classification task.

We found that we needed at least two ICs per bacterium in order to prevent a premature convergence of all ICs in all bacteria to the same density class. This state leads to an evolutionary dead-lock where all bacteria carry a CA that classifies ICs always as belonging to the same density class, irrespective of its density. In order to maintain diversity of ICs bacteria have to carry at least two ICs of different density classes. We fixed the number of ICs per bacterium to two in order to simplify the analysis of the results. In simulations in which the number of ICs was variable we found qualitatively the same results as we report here.

2.2 The Density Classification Task

The toxicity of ICs and the immunity given by CAs is based on the density classification task. The CAs are 1-dimensional, binary state cellular automata with a neighborhood size 3, the ICs are initial conditions of the cellular automata and are of length 149. Both CAs and ICs are represented as bit strings. The density of an IC is defined as the ratio of 1's to 0's in its bit string.

In the density classification task the CAs must classify ICs on the basis of the density of the IC. If the IC has a density less than 0.5 it belongs to class 0, otherwise it belongs to class 1. The CA is allowed to iterate for maximally 320 time steps, starting with the IC as initial condition. If the CA settles into a homogeneous state of all 0's it classifies the IC as being of class 0. If the CA settles into a homogeneous state of all 1's it classifies the IC as being of class 1. If the CA does not settle into a homogeneous state it does not classify the IC at all.

The *performance* of a CA is defined as the ratio of correct classifications it makes on a set of $10,000$ randomly created ICs [3]. We use the performance as an objective measure of the classification behavior of CAs and to compare CAs of different populations. 'Good' cellular automata typically have performance values of approximately 0.8 (e.g. the GKL rule; 0.81). However, a complete solution, i.e. a cellular automaton with 100% classification accuracy, does not exist [8]. In previous studies of (co-)evolutionary

optimization models that used this optimization problem it appeared difficult to evolve CAs with high performance values [9, 10, 16]. Recently, however, cellular automata have been found with performance values of up to 0.86 [6].

An important property of the density classification task is that most ICs, i.e. ICs with a density around 0.5, are most difficult to classify and easily evolve from one density class to the other one (i.e. by flipping as little as a single bit). In fact, the quality of classification of a 'good' cellular automaton, like for instance the GKL rule, decreases rapidly if it is evaluated on the basis of ICs whose density approaches 0.5 [6, 10].

parameter	value
field size S	50×50
bacterium growth rate g	0.125
bacterium death rate d_w	0.1
death rate per toxic IC d_i	+0.3
CA mutation rate μ_{ca}	0.1
IC mutation rate μ_{ic}	0.2

Table 1. Parameters and their default values.

The traditional point mutation operator, i.e. flipping a bit at a random position in the string, gives a strong bias towards initial conditions with a density value of 0.5 (see also [10]). Thus, the mutation operator is biased with respect to the property of ICs on the basis of which they are to be classified. We altered the mutation operator in order to make it neutral with respect to the density value of the IC it operates on. This 'density-neutral' mutation randomly increases or decreases the density of the IC by one. When the density is increased a randomly chosen 0-bit is flipped, for a decrease in density a randomly chosen 1-bit is flipped. ICs with densities of 0.0 or 1.0 mutate only if the density increases or decreases, respectively. Only at the beginning of a simulation such ICs with extreme density values exist, therefore, this does not significantly influence the results.

This mutation operator is not neutral with respect to all possible bit strings. Many strings have density values around 0.5; the density neutral mutation operator under samples these bit strings. In the simulations, however, we see that the ICs evolve toward density values near 0.5; in the evolutionary process these ICs are preferentially sampled. We use the density-neutral point mutation in the model with a rate $\mu_{ca} = 0.1$ for the CAs and a rate $\mu_{ic} = 0.2$ for the ICs. These rates amount to probabilities per bit of approximately 0.0013 for the ICs and 0.0008 for the CAs. In table 1 we have given a table of the parameter and the values that we used in the simulations that are described in the next section.

3 Results

In this paper we foremost want to establish the possibility of evolving novel allelopathic types on the basis of existing ones. Secondly, we want to show that the evolutionary

dynamics that result in this system can lead to the evolution of general strategies. This characteristic of the evolution of allelopathic types can be interpreted, and possibly used, as a search process for general solutions to optimization problems.

We did several runs with the parameter settings given in table 1. Although we did a few additional simulations with different parameter settings we have not performed a rigorous analysis of parameter sensitivity. The results that we report here are, however, also typical for the additional simulations that we studied.

3.1 A Typical Simulation

In this model we have a variable bacterium population size. All bacteria have equal growth and death rates, thus we would expect a stable population size in the absence of allelopathic systems. Because the effectiveness of the toxins and the antidote repertoires in the bacterium population evolve independently the effective bacterium death rate changes over time. Thus, also the bacterium population size is expected to change over time. The fluctuations only indicate changes in the *relative* effectiveness of the toxins and antidotes, it does not indicate the absolute effectiveness of, say, the antidote repertoires.

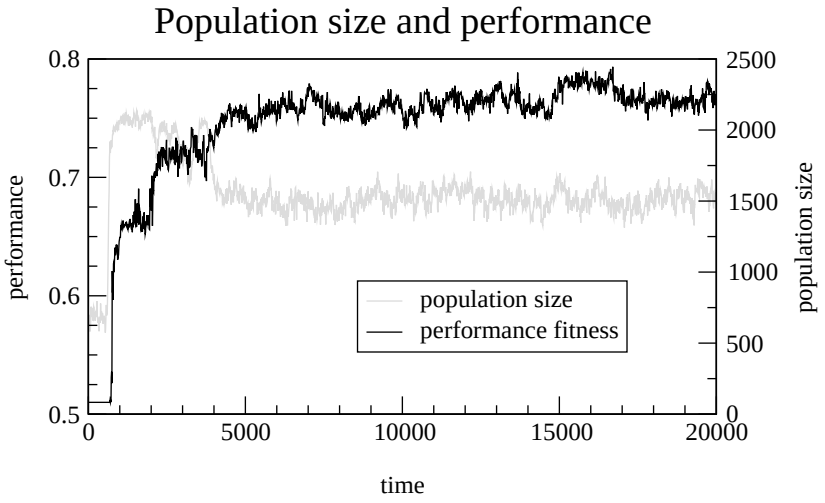

Fig. 1. Evolution of the performance of the best individual, plus the bacterium population dynamics.

In fig. 1 we plot the population dynamics together with the performance of the best CA in the population. The bacterium population starts out small; apparently many bacteria are sensitive to the toxins that are produced in their neighborhood. At $t \approx 700$ the bacterium population size increases sharply. At this point most bacteria carry antidote repertoires that neutralize all toxins that are present in their neighborhood. Around

$t \approx 4000$ the bacterium population size drops again, indicating an increase in the effective death rate of the bacteria; apparently the bacteria have evolved more effective toxin repertoires. This latter situation remains stable until the simulation ends at $t = 20,000$.

The sharp increase in the population size at $t \approx 700$ is accompanied by a sharp increase in the performance of the best CA (fig. 1). The performance of the best CA at $t = 0$, i.e. $p \approx 0.5$, is typical for CAs that randomly classify IC to class 0 or to class 1, and for CAs that classify all ICs to one and the same density class. Between $t \approx 700$ and $t \approx 2000$ the performance of the best CA is approximately $p \approx 0.65$. This performance is typical for CAs that use *block-expanding* strategies to classify ICs [3, 5].

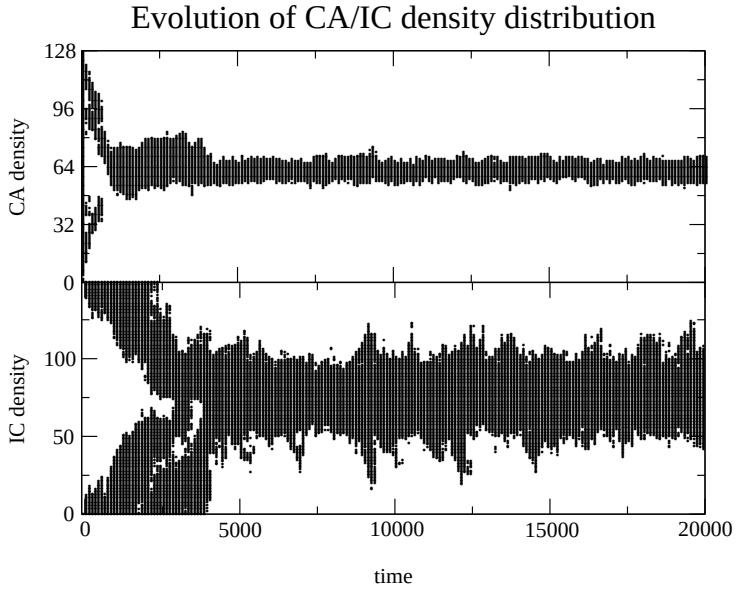

Fig. 2. Evolution the density distribution of the CAs (top panel), and of the ICs (lower panel). The length of the bit strings of the CAs is 128, the bit strings of ICs have length 149. Both ICs and CAs converge to medium density values.

In fig. 2 we plot the evolution of the distribution of the density values of the CAs (top panel) and the ICs (lower panel). At first CA have density values that are very low or very high. The behavior of CAs at this time, classifying all ICs to one density class, can be accomplished best with a CA that is encoded by a bit string consisting of all 0's or all 1's. The transition from low performance to intermediate performance ($t \approx 700$) coincides with a convergence of CA density values to a distribution broadly around 0.5 (i.e. 64 bits). A block-expanding strategy always settles in a homogeneous state of 1's (0's) unless there is a sufficiently large block of 0's (1's) present in the IC. This block then is expanded to over the whole state of the CA. This strategy requires both 0's and 1's in the CA bit string although in uneven numbers.

In the next 5000 time steps the performance of the best CA increases to values around $p \approx 0.75$. CAs with a performance in that range typically use *particle-based* strategies to classify the density of ICs [3, 5]. Particle-based strategies use mesoscale information processing rules on the basis of (interactions between) regular domains [3]. All of the known CAs with high performance values use particle-based strategies. The transition to particle-based CAs concurs with a strong convergence of the CA density distribution on values around 0.5 (fig. 2). In [11] it was argued that CAs with high performance values will have density values of 0.5. The performance of the best CA in the remaining 15.000 time steps fluctuates between $p \approx 0.75$ and $p \approx 0.80$. The best CA in the the simulation occurs at $t = 16,700$ and has a performance fitness of $p \approx 0.79$.

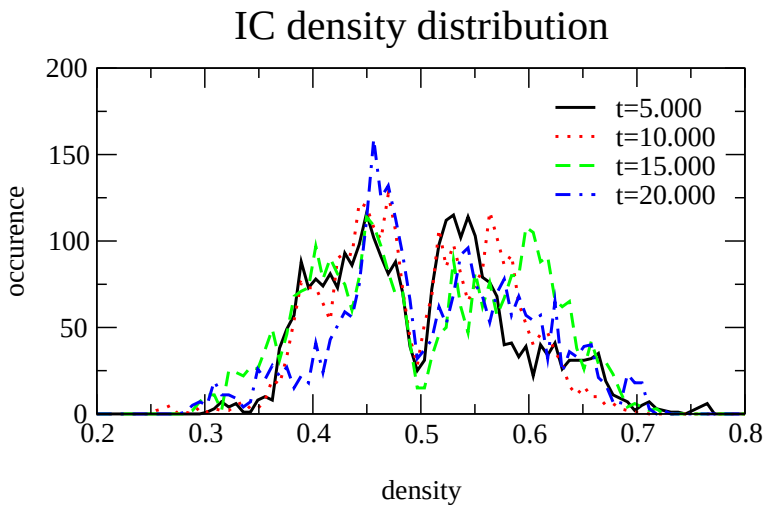

Fig. 3. Distribution of IC density values for all ICs at t=5,000, t=10,000, t=15,000, and t=20,000

In the last period when the bacteria have very general antidote repertoires; the performance of the CAs that they code for is relatively high. The bacterium population size is nevertheless significantly smaller than the carrying capacity; apparently the bacteria carry very effective toxins as well. Indeed, the ICs have density values close to 0.5 (fig. 2). Note that the density distribution of the IC-population shows two peaks, one around 0.45 and one peak around 0.55 (fig. 3). The diversity of the IC density classes is maintained rather than that it collapses to a monomorph population of ICs as in sect. 2.1. It seems that in this state a good general strategy is favorable over a specialized one.

4 Discussion

In the previous section we showed that novel allelopathic types can evolve also when they must evolve on the basis of already existing types. Rather then creating a complete

new allelopathic system in a single mutational event, as in earlier studies [14], here the genes that code for the toxins and antidotes evolve on the basis of already existing genes. As in the evolution of natural allelopathic systems immunity to a novel toxin should be evolved before actually evolving and producing the novel toxin. We find that this readily happens; in all simulations antidote repertoires evolve that show medium to high degrees of generality. In 8 out of 30 simulations we find CAs with performance values around 0.8, in the remaining 22 simulations we find CAs with performance values between 0.65 and 0.7.

As an evolutionary search algorithm this model seems inefficient; it requires the evolution of correct solutions prior to the use of fitness cases to evaluate the solutions. Indeed, we see that simulations run for large numbers of generations before good solutions are found (here at $t \approx 5000$). Note, however, that per time step CAs are evaluated on at most 18 ICs. This number even decreases if the local density of bacteria is less than 1. Thus, in terms of number of "fitness evaluations" this model is very efficient (see also [13]).

Also, compared to other (co-)evolutionary optimization models [21] we find CAs with reasonable high performance in a relatively large number of simulations. Also, the diversity is very high despite the low mutation rate and the absence of a diversity promoting selection mechanisms (e.g. sharing or niching). We find that the number of unique CA genotypes makes up 39% of the population, the number of unique IC genotypes makes up 70% of the population. These percentages are much higher than those reported in [15]. Actually, a large number of different good solutions is preserved in the population. For instance, in the simulation described in the previous section at t=20,000 25% of all bacteria have a performance within 10% of the best CA ($p = 0.77$). In these 363 CAs we find 100 unique genotypes.

Other than the selection mechanism we also use a variable population size in our model. Elsewhere, we reported on a study of a spatial coevolutionary optimization model, using the same optimization task [15]. As one of the possible evolutionary outcomes we observed red queen dynamics[2]. These red queen dynamics are characterized by large fluctuations in the individual fitness values, even with periods with fitness values equal to zero. In a model that includes a variable population size this can not occur; the population would simply die out. In the model described here the toxins (i.e the fitness cases, or in terms of coevolutionary optimization; the 'parasites') are part of a complex that includes the antidote (i.e. the solution, or 'host'). This limits the toxin to evolve only so that the corresponding antidote can 'solve' it, otherwise it would commit suicide. Our results show that this limitation does not a priori restrict the evolutionary process from finding good solutions.

References

[1] M. Achtman, A. Mercer, B. Kusecek, A. Pohl, M. Heuzenroeder, W. Aaronson, A. Sutton, and R. P. Silver. Six widespread bacterial clones among Escherichia coli K1 isolates. *Infect. Immun.*, 39:315–335, 1983.

[2] Red queen dynamics occurred when the population was globally mixed, whereas evolution of general solutions occurred when individuals remained localized in space.

[2] David Andre, Forrest H Bennett III, and John R. Koza. Evolution of intricate long-distance communication signals in cellular automata using genetic programming. In *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, volume 1, Nara, Japan, 16–18 May 1996. MIT Press.

[3] James P. Crutchfield and Melanie Mitchell. The evolution of emergent computation. *Proceedings of the National Academy of Sciences*, 92:10742–10746, 1995.

[4] D. L. Hartl and D. E. Dykhuizen. The population genetics of Escherichia coli. *Annu. Rev. Genet.*, 18:31–68, 1984.

[5] Wim Hordijk, James P. Crutchfield, and Melanie Mitchell. Lecture notes in computer science. In *PPSN-V*, volume 1498, pages 613–622, 1998.

[6] Hugues Juillé and Jordan B. Pollack. Coevolving the ideal trainer: Application to the discovery of cellular automata rules. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 519–527, University of Wisconsin, Madison, Wisconsin, USA, 22-25 July 1998. Morgan Kaufmann.

[7] J. Konisky. Colicins and other bacteriocins with established modes of action. *Annu. Rev. Microbiol.*, 36:125–144, 1982.

[8] Mark Land and Richard K. Belew. No perfect two-state cellular automata for density classification exists. *Physical Review Letters*, 74(25):5148–5150, June 19 1995.

[9] Melanie Mitchell, James P. Crutchfield, and Rajarshi Das. Evolving cellular automata with genetic algorithms: A review of recent work. In *Proceedings of the First International Conference on Evolutionary Computation and its Applications (EvCA'96)*, 1996.

[10] Melanie Mitchell, James P. Crutchfield, and Peter T. Hraber. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, 75:361–391, 1994. SFI Working Paper 93-11-071.

[11] Melanie Mitchell, Peter T. Hraber, and James P. Crutchfield. Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, 7:89–130, 1993.

[12] Ludo Pagie. *Information integration in evolutionary processes*. PhD thesis, Bioinformatics group, Utrecht University, 1999.

[13] Ludo Pagie and Paulien Hogeweg. Evolutionary consequences of coevolving targets. *Evolutionary Computation*, 5(4):401–418, 1997.

[14] Ludo Pagie and Paulien Hogeweg. Colicin Diversity: a Result of Eco-evolutionary Dynamics. *J. Theor. Biol.*, 196:251–261, 1999.

[15] Ludo Pagie and Paulien Hogeweg. Information integration and red queen dynamics in coevolutionary optimization. In *Proceedings of Congress on Evolutionary Computation*, 2000.

[16] Jan Paredis. Coevolving cellular automata: be aware of the red queen! In T. Baeck, editor, *Proceedings of the 7th Int. Conference on Genetic Algorithms (ICGA 97)*, pages 393–400, 1997.

[17] A. P. Pugsley. The ins and outs of colicins. Part II. Lethal action, immunity and ecological implications. *Microbiol. Sci.*, 1:203–205, 1984.

[18] M. A. Riley. Molecular mechanisms of colicin evolution. *Mol. Biol. Evol.*, 10:1380–1395, 1993.

[19] M. A. Riley and D. M. Gordon. A survey of Col plasmids in natural isolates of Escherichia coli and an investigation into the stability of Col-plasmid lineages. *J. Gen. Microbiol.*, 138:1345–1352, 1992.

[20] Y. Tan and M. A. Riley. Rapid invasion by colicinogenic Escherichia coli with novel immunity functions. *Microbiology.*, 142:2175–2180, 1996.

[21] Justin Werfel, Melanie Mitchell, and James P. Crutchfield. Resource sharing and coevolution in evolving cellular automata. Submitted to IEEE Trans. Evol. Comp., 1999.