relatively constant as N is increased, indicating a marked improvement in generalization.

Why does ϕ_{1d}, for example, perform relatively well on the $\rho_c = 1/2$ task? In Figure 1(d) it can be seen that, although the patterns eventually converge to fixed points, there is a transient phase during which a spatial and temporal transfer of information about the density in local regions takes place. This local information interacts with other local information to produce the desired final state. Roughly ϕ_{1d} successively classifies "local" densities with a locality range that increases with time. In regions where there is some ambiguity, a "signal" is propagated. This is seen either as a checkerboard pattern propagated in both spatial directions or as a vertical white-to-black boundary. These signals indicate that the classification is to be made at a larger scale. Note that regions centered about each signal locally have $\rho = \rho_c$. The consequence is that the signal patterns can propagate, since the density of patterns with $\rho = \rho_c$ is neither increased nor decreased under the rule.

The discovery of these rules was first reported in [5]. Methods for understanding their emergent logic were sketched there and were given in more detail in [1]. In the following we first briefly review the computational mechanics approach to CAs. Using these methods, we then analyze the strategy of ϕ_{1d}—the rule with highest $\mathcal{P}^N_{10^4}$ found in the 300 runs—and describe the generational progression by which it was evolved under the GA.

4.3 Computational Mechanics of Cellular Automata

Like many natural processes that are spatially-extended, cellular-automata configurations often organize over time into spatial regions that are dynamically homogeneous. Sometimes in space-time diagrams these regions are obvious to the eye as "domains": regions in which the same "pattern" appears. In order to understand this phenomenon, and also to deal with cases in which human observation is inadequate or undesired, the notion of "domain" was formalized in [8] by adapting computation theory to CA dynamics. There, a domain's "pattern" is described using the minimal deterministic finite automaton (DFA) that accepts all and only those configurations that appear in the domain. Such domains are called "regular" since their configurations are members of the regular language recognized by the DFA. More precisely, a regular domain Λ is a set of configurations that is temporally-invariant, $(\Lambda = \phi(\Lambda))$, and whose DFA has a single recurrent set of states that is strongly connected.

Regular domains play a key role in organizing both the dynamical behavior and the information-processing properties of CA. Once a CA's regular domains have been detected, for example, nonlinear filters can be constructed to filter them out, leaving just the deviations from those regularities. The resulting filtered space-time diagram reveals the propagation of domain "walls". If these walls remain spatially-localized over time, then they are called "particles" [4]. Particles are one of the main mechanisms for carrying information over long space-time distances. This information might indicate, for example, the result of some local processing which has occurred at an early time. In this way particles can serve as signals. Logical operations on the information they contain are performed when the particles interact. The collection of domains, domain walls, particles, and particle interactions for a CA represents the basic information-processing elements embedded in the CA's behavior—the CA's "intrinsic" computation.

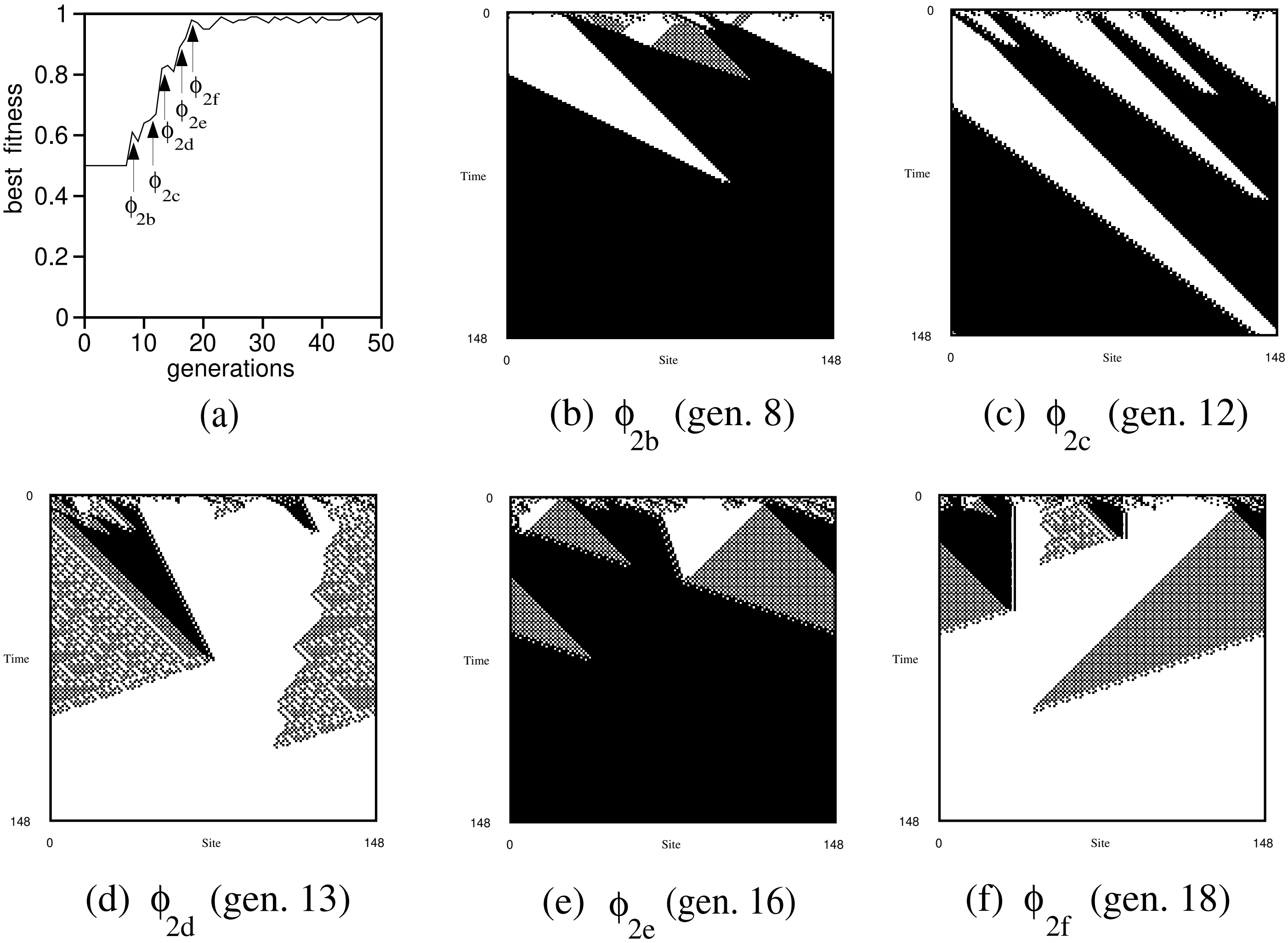

Figure 2: (a) Plot of the fitness of the most fit rule in the population versus generation in the run producing ϕ_{1d}. The arrows in the plot indicate the generations in which the GA discovered each new significantly improved strategy. (b)–(f) Space-time diagrams illustrating the behavior of the best rule at each of the five generations marked in (a).

4.4 The GA Discovery of Particle-Based Computation

In this section we apply the computational mechanics framework sketched above to describe in detail the route by which the GA discovered ϕ_{1d}.

Figure 2(a) plots best fitness (F_{100}) in the population versus generation for the first 50 generations of the run in which ϕ_{1d} was discovered. It is evident from the figure that, before the GA discovers high fitness rules, the fitness of the best CA rule increases in rapid jumps after periods of relative stasis. Qualitatively, the rise in performance can be divided into at several "epochs", each corresponding to the discovery of a new, significantly improved strategy for performing the $\rho_c = 1/2$ task. In Figure 2(a), the generations of these epochs are marked with a label for the best rule in the population at that generation.

For the first seven generations, the highest F_{100} value is 0.5, achieved by a number

Wall Type	Particle	Velocity (Gen. = 8)	Velocity (Gen. = 18)
WB	P_{WB}	1.0	0.0
BW	P_{BW}	2.0	0.0
$\#W$	$P_{\#W}$	-1.0	-3.0
$W\#$	$P_{W\#}$	-1.0	-1.0
$B\#$	$P_{B\#}$	3.0	3.0
$\#B$	$P_{\#B}$	1.0	1.0

Table 2: The six particles generated by the best rule in generation 8 (cf. Figure 2(b)) and in generation 18 (cf. Figure 2(f)).

of rules. In these rules most neighborhoods map either to 1 or to 0. Rules in which most neighborhoods map to 1 quickly settle to the all 1s configuration irrespective of ρ_0. Similarly, rules in which most neighborhoods map to 0 quickly converge on the all 0s pattern. In either case, the rules are able to correctly classify exactly half of the ICs and thus have a fitness of 0.5. (In [10] this behavior was termed "epoch 1".)

In generation 8, a new strategy (ϕ_{2b}) is discovered resulting in significantly better performance ($F_{100} \approx 0.61$). Its behavior is illustrated by the space-time diagram in Figure 2 (b). ϕ_{2b} was created by a crossover between rule that maps most neighborhoods to 0 and a rule that maps most neighborhoods to 1. A closer inspection of the behavior of this rule reveals that it always defaults to the all 1s configuration except when ρ_0 is close to zero, in which case it relaxes to the all 0s configuration, yielding $F_{100} > 0.5$. (In [10] this behavior was termed "epoch 2".) An analysis of particle interactions explains ϕ_{2b}'s behavior. After a very short transient phase, ϕ_{2b} essentially creates three regular domains: the all 1s domain (B), the all 0s domain (W) and the checkerboard domain $(10)^* \bigcup (01)^*$ ($\#$). This behavior can be understood readily from the ϕ_{2b}'s rule table: out of the 28 neighborhood patterns with five or more 1s (0s), 27 (22) result in an output bit of 1 (0). Therefore in any IC, small "islands" of length 1, 2 or 3 containing only 1s or 0s in are quickly eliminated, resulting in locally uniform domains. To maintain a # domain, the neighborhood patterns (1010101) and (0101010) must produce a 1 and 0 respectively, as is done in ϕ_{2b}.

After the B, W, and # domains are created, the subsequent behavior of ϕ_{2b} is primarily determined by the interaction of the particles representing the domain walls. Given that there are three domains, for this CA there are six particles, one for each domain wall type: $P_{BW}, P_{WB}, P_{B\#}, P_{\#B}, P_{W\#}$ and $P_{\#W}$. The velocity of these particles (i.e., the slope of the domain wall corresponding to the particle) are given in Table 2. There are two annihilative interactions: $P_{BW} + P_{WB} \rightarrow \emptyset_B$ and $P_{B\#} + P_{\#B} \rightarrow \emptyset_B$ (where $\emptyset_B$ indicates a B domain with no particles). Two interactions are reactive: $P_{BW} + P_{W\#} \rightarrow P_{B\#}$ and $P_{B\#} + P_{\#W} \rightarrow P_{BW}$. These interactions can all be seen in Figure 2(b). Several particle interactions never occur as a result of the direction and magnitude of the particle velocities. For example, the particles $P_{\#W}$ and $P_{W\#}$ have the same velocity and never interact. On the other hand, although $P_{\#B}$ moves in the same direction as P_{BW}, the former has only half the speed of the latter, and the two particles never have the chance to interact. For similar reasons, the # domain and its four associated particles $P_{B\#}, P_{\#B}, P_{W\#}$, and $P_{\#W}$ play no appreciable role in the determination of the final configuration of the CA. This is easily verified by complementing the output bit of one of the two neighborhood patterns necessary to maintain the # domain and observing that very little change occurs in the rule's fitness.

Thus, for this epoch we can focus exclusively on the boundaries between the B and the W regions, since initially they play the major role. The space-time behavior shows that the particle P_{WB} moves to the right with a velocity of 1.0, while the particle P_{BW} moves to the right with a velocity of 2.0. As a result, P_{BW} soon catches up with P_{WB}. The interaction of these two walls results in a B domain, eliminating the W domain. Therefore when an island of W is surrounded by B domains, the size of the former shrinks until it vanishes from the lattice. Conversely, when an island of B is surrounded by W domains, the B domain grows until it occupies the whole lattice. However, for an island of B to expand, it must contain at least five cells. Thus if any initial configuration contains a block of five 1s *or* results in the creation of such a block when the rule is applied over subsequent time steps, then the CA inevitably relaxes to the all 1s configuration.

The explanation of ϕ_{2b}'s behavior in terms of particles and particle interactions may seem a complicated way to describe simple behavior. But the usefulness of the computational mechanics framework for explicating computation will become clearer as the space-time behavior becomes more complex. In the succeeding generations, the velocity of the particles and their interactions play the most crucial role in determining the fitness of the CA rule.

The next four generations produce no new epochs (i.e., no significantly new strategies) but a modest improvement in the best fitness. During this period three innovations were observed, all of which appear in the rule ϕ_{2c} created in generation 12. ϕ_{2c}'s behavior is illustrated in Figure 2(c). First, for an island of B to expand and take over the whole lattice, it now must contain at least seven cells. Since it is less likely that an island of B of length seven is going to form (when compared to an island of length five) from ICs with low densities, more ICs with low densities are correctly classified. Second, the velocity of the particle P_{BW} is decreased to 1.5. Since the velocity of the particle P_{WB} remains constant at 1.0, it now takes longer to eliminate W domains. Third, the velocity of the particle $P_{\#W}$ is modified from -1.0 to -3.0, allowing the annihilative reaction: $P_{W\#} + P_{\#W} \rightarrow \emptyset_W$. Unlike the innovation that resulted in $\phi_2 b$, where crossover played the major role, we find that these modifications (and the ones that follow) are primarily due to the mutation operator.

In generation 13, a new epoch begins with the discovery of ϕ_{2d}, with a sharp jump in F_{100} to 0.82 corresponding to a significant innovation. ϕ_{2d}'s behavior is illustrated in Figure 2(d). Here we use # to refer to an interesting variation of the checkerboard ($(10)^* \bigcup (01)^*$) domain. While the velocity of P_{WB} is held constant at 1.0, P_{BW} now moves with a velocity of 0.5 in the same direction. Since P_{BW} cannot catch up with P_{WB}, an island of W can now expand when surrounded by B, with the condition that W has to be at least six cells in length. With the above exception, the rule still defaults to the all 1s configuration. On the plus side, the best rule now correctly classifies most of the initial configurations with $\rho_0 < 0.5$. However, this strategy does cause a few (new) misclassifications when ρ is slightly greater than 0.5 due to the chance formation of an island of W of length six. In addition to the above changes, a new interaction is allowed to take place for the first time. The decrease in velocity of the particle P_{BW} to 0.5 not only results in the removal of small islands of B but it also allows $P_{\#B}$ to interact with P_{BW}. The interaction results in the particle $P_{\#W}$ with a velocity of -3.0. From this juncture the # domain and its four associated particles play a major role in determining the fitness of a CA rule.

After a brief stasis over three generations, two developments result in the improved performance seen in ϕ_{2e}, the best rule in generation 16 ($F_{100} = 0.89$). The velocity of P_{BW}

is further reduced to 1/3. More importantly, the particle P_{WB} spontaneously creates a new # domain with two boundaries, one on each side: $P_{W\#}$ moving to the left with a velocity of -1.0, and $P_{\#B}$ moving to the right with a velocity of 1.0 (Figure 2(e)). Since $P_{W\#}$ is moving to the left and can interact with right moving P_{BW} to create $P_{B\#}$, the rule can stop the growth of W domains of length more than 6 cells. Therefore this rule can not only eliminate isolated islands of B, but also remove islands of W. However, because P_{BW} is moving to the right with a velocity of 1/3, it takes longer to remove islands of B than to remove islands of W. This problem is rectified by the discovery of ϕ_{2f} in generation 18, in which the velocity of P_{BW} is set to zero (Figure 2(f)), yielding $F_{100} = 0.98$. From generation 19 till the end of the GA run, little change is seen in either the fitness of the best rule in the population or its behavior (see Figure 1(d)).

Interestingly, it turns out that the behavior of the best rule discovered by the GA is very similar to the behavior of a well-known CA, the Gacs-Kurdyumov-Levin (GKL) CA [7]. The GKL rule (ϕ_{GKL}) was invented to study reliable computation and phase transitions in one-dimensional spatially-extended systems. As shown in Table 1, ϕ_{GKL} has higher $\mathcal{P}^N_{10^4}$ than any rule found by the GA, although ϕ_{GKL} and ϕ_{1d} produce the same three domains $B, W, \#$ and also result in identical particles, particle velocities, and particle interactions. The difference in performance results, for example, from asymmetries in ϕ_{1d} not present in ϕ_{GKL} (see [10]); the detailed differences will be described in future work.

5. Conclusion

The results presented in this paper demonstrate how evolution can engender emergent computation in a spatially-extended system. The GA found CAs that used particle-based computation to achieve relatively high performance on a density classification task. These discoveries are encouraging, both for using GAs to model the evolution of emergent computation in nature and for the automatic engineering of emergent computation in decentralized multicomponent systems. The version of the GA used in our experiments discovered sophisticated CAs only rarely. A number of factors have been identified that limit its performance; most notable among them are the noisy fitness function F_100 and the breaking of the inherent symmetries of the $\rho_c = 1/2$ task. Mitchell, Crutchfield, and Hraber [10] suggest a number of approaches for overcoming these impediments, and we expect that these approaches will result in evolutionary-computation methods that discover sophisticated particle-based computation for a wide variety of applications.

Acknowledgments

This research was supported by the Santa Fe Institute, under the Adaptive Computation and External Faculty Programs, and by the University of California, Berkeley, under contract AFOSR 91-0293.

References

[1] J. P. Crutchfield. The emergent logic of the Gacs-Kurdyumov-Levin cellular automaton. In preparation, 1993.

[2] J. P. Crutchfield. The calculi of emergence: Computation, dynamics, and induction. *Physica D*, in press, 1994.

[3] J. P. Crutchfield. Is anything ever new? Considering emergence. In G. Cowan, D. Pines, and D. Melzner, editors, *Integrative Themes*, volume XIX of *Santa Fe Institute Studies in the Sciences of Complexity*, Reading, MA, 1994. Addison-Wesley. In press.

[4] J. P. Crutchfield and J. E. Hanson. Turbulent pattern bases for cellular automata. *Physica D*, 69:279–301, 1993.

[5] J. P. Crutchfield and M. Mitchell. The evolution of emergent computation. *Science*, 1994. submitted.

[6] S. Forrest. Emergent computation: Self-organizing, collective, and cooperative behavior in natural and artificial computing networks: Introduction to the Proceedings of the Ninth Annual CNLS Conference. *Physica D*, 42:1–11, 1990.

[7] P. Gacs, G. L. Kurdyumov, and L. A. Levin. One-dimensional uniform arrays that wash out finite islands. *Probl. Peredachi. Inform.*, 14:92–98, 1978.

[8] J. E. Hanson and J. P. Crutchfield. The attractor-basin portrait of a cellular automaton. *J. Stat. Phys.*, 66:1415, 1992.

[9] C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors. *Artificial Life II.* Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley, Reading, MA, 1992.

[10] M. Mitchell, J. P. Crutchfield, and P. T. Hraber. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, in press, 1994.

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

[12] C. Moore. Personal communication.

[13] N. H. Packard. Adaptation toward the edge of chaos. In J. A. S. Kelso, A. J. Mandell, and M. F. Shlesinger, editors, *Dynamic Patterns in Complex Systems*, pages 293–301, Singapore, 1988. World Scientific.

[14] T. Toffoli and N. Margolus. *Cellular Automata Machines: A new environment for modeling.* MIT Press, Cambridge, MA, 1987.

[15] S. Wolfram, editor. *Theory and applications of cellular automata.* World Scientific, Singapore, 1986.