Searching for Glider Guns in Cellular Automata: Exploring Evolutionary and other Techniques

E. Sapin and L. Bull

Faculty of Computing, Engineering and Mathematical Sciences,
University of the West of England,
Bristol BS16 1QY, UK
emmanuelsapin@hotmail.com, Larry.Bull@uwe.ac.uk

Abstract. We aim to construct an automatic system for the discovery of collision-based universal cellular automata that simulate Turing machines in their space-time dynamics using gliders and glider guns.
In this paper, an evolutionary search for glider guns with different parameters is described and other search techniques are also presented as benchmark. We demonstrate the sponstaneous emergence of an important number of novel glider guns discovered by genetic algorithms.

1 Introduction

The emergence of computation in complex systems with simple components is a hot topic in the science of complexity [1]. A uniform framework to study emergent computation in complex systems are cellular automata [2]. They are discrete systems in which an array of cells evolves from generation to generation on the basis of local transition rules [3].

The well-established problems of emergent computation and universality in cellular automata has been tackled by a number of people in the last thirty years [4], [5], [6], [7], [8] and remains an area where amazing phenomena at the edge of theoretical computer science and non-linear science can be discovered.

The most known universal automaton is the Game of Life [9]. It was shown to be universal by Conway [10] who employed *gliders* and *glider guns*. Gliders are mobile self-localized patterns of non-resting states, and glider guns are patterns which, when evolving alone, periodically recover their original shape after emitting some gliders.

The search for gliders was notably explored by Adamatzky et al. with a phenomenological search [11], Wuensche who used his Z-parameter and entropy [12] and Eppstein [13]. Sapin et al. have considered the emergence of gliders-based universality by use of a genetic algorithm [14].

We aim to construct an automatic system for the discovery of computationally universal cellular automata, spatially. Inspired by the link between universality and the presence of gliders and glider guns in cellular automata, we are here interested in the emergence of glider guns. In this paper, three search methods for glider guns are compared.

The paper is arranged as follows: Section 2 describes previous related work. Section 3 sets out the characteristics of the search methods. Then the result of the best search method are described in Section 4. The last section summarizes the presented results and discusses directions for future research.

2 Previous work

In this section, some previous work about cellular automata are presented. Brief descriptions of some search methods are given. Then some previous work about using an evolutionary approach to search for automata are presented.

2.1 Cellular automata

In [15], Wolfram studies the space $\mathcal{I}$ of 2D isotropic CA, with rectangular 8-cell neighbourhoods: if two cells have the same neighbourhood states by rotations and symmetries, then these two cells take the same state at the next generation. There are 512 different rectangular 8-cell neighbourhood states. An automaton of I can be described as shown figure 1 by telling what will become of a cell in the next generation, depending on its subset of isotropic neighbourhood states.

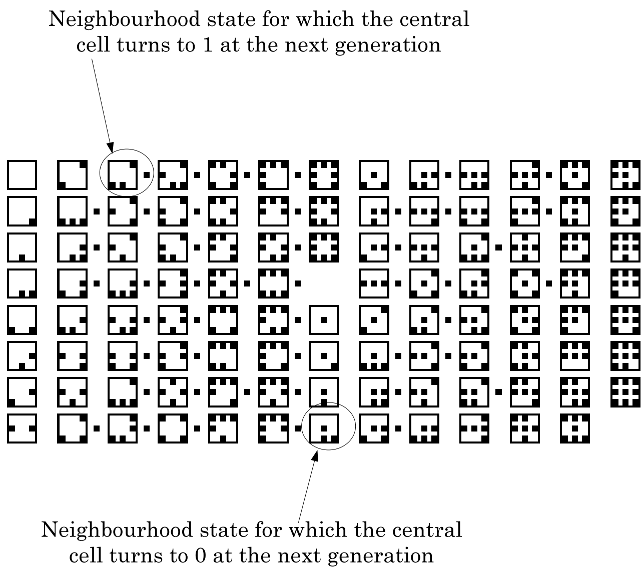

Fig. 1. The squares are the 102 neighbourhood states describing an automaton of $\mathcal{I}$. A black cell on the right of the neighbourhood state indicates a future central cell.

There are 102 subsets of isotropic neighbourhood states, meaning that there are 2^{102} different automata in $\mathcal{I}$.

2.2 Search methods

In order to search for universal automata, we have examined the search methods such as monte carlo, taboo search [16] and an evolutionary algorithm [17], as briefly described here.

The monte carlo method consists solely of generating random solutions and testing them.

Tabu search traverses the solution space by testing mutations of an individual solution. Tabu search generates many mutated solutions and moves to the best solution of those generated. In order to prevent cycling and encourage greater movement through the solution space, a tabu list is maintained of partial or complete solutions. It is forbidden to move to a solution that contains elements of the tabu list, which is updated as the solution traverses the solution space.

Evolutionary algorithms have been used with cellular automata in a number of ways, after [18].

2.3 Evolving Cellular Automata

Previously, several good results from the evolution of cellular automaton rules to perform some useful tasks have been published. Mitchell et al. [19–22] have investigated the use of evolutionary computing to learn the rules of uniform one-dimensional, binary cellular automata. Here a Genetic Algorithm produces the entries in the update table used by each cell, candidate solutions being evaluated with regard to their degree of success for the given task — density and synchronization.

Sipper [23] has presented a related approach, which produces non-uniform solutio¡ ns. Each cell of a one or two-dimensional cellular automata is viewed as a genetic algorithm population member, mating only with its lattice neighbours and receiving an individual fitness. He shows an increase in performance over Mitchell et al.'s work, exploiting the potential for spatial heterogeneity in the tasks. Koza et al. [24] have also repeated Mitchell et al.'s work, using Genetic Programming [25] to evolve update rules. They report similar results.

3 Search for Glider Guns

This section describes the used search for glider guns. To compare the parameters of the search methods, glider guns that emitting the glider in figure 2 are searched for. The first search method is an evolutionary algorithm. Monte carlo algorithm and tabu search are also used as benchmark.

3.1 Evolutionary Algorithm

The parameters of the evolutionary algorithm are described here.

Fitness Function
 Two fitness functions have been tried.

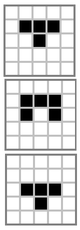

Fig. 2. Glider.

- First fitness function
  The computation of the fitness function is based on the one used in [26]. A random configuration of cells is evolved by the tested automaton. After this evolution, the presence of gliders G is checked by scanning the result of the configuration of the cells. The value of the fitness function is the number of gliders that appeared divided by the total number of cells.
- Second fitness function
  The computation of the second fitness function is based on the first one. A random configuration of cells is evolved by the tested automaton. After this evolution, the presence of gliders G is checked by scanning the result of the configuration of the cells. The size of the biggest square S without cells in the evolved configuration of cells is computed. The value of the fitness function is the multiplication of the number of gliders that appeared by the size of the square S.

Initialization
The search space is the set $\mathcal{I}$ described in Section 2. Cell-state transition table can describe an automaton of this space. An individual is an automaton coded as a bit string of 102 Booleans representing the values of a cell at the next generation for each neighbourhood state. The 102 bits of an automaton are divided into two subsets. The first subset, called invarious subset, is the neighbourhood states used by the glider G and their values are determined by the evolution of G. The process that determines these neighbourhood states is detailed for the glider of figure 2 in the figure 3. The other neighbourhood states are in the second subset, called unused subset, and are initialised at random. The population size is 100 individuals.

Genetic Operators
The mutation function simply consists of mutating one bit among the second subset of the 102 bits. The rates 1,5 and 10 percent are tried, together with three crossover operators.

- No crossover
  The genetic algorithm is tried without crossover.
- Central point
  A single point crossover with a locus situated exactly on the middle of the genotype is tried.
- Random point

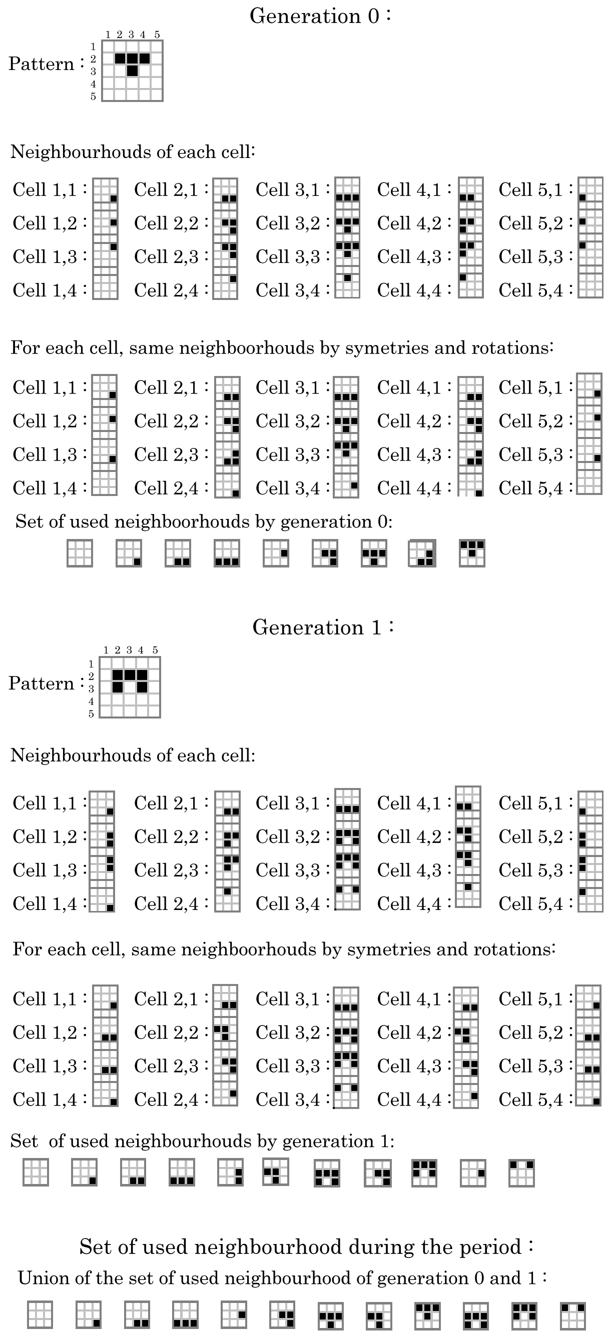

Fig. 3. Detail of the construction of set of neighbourhood states that are used by a glider.

A last kind of recombination is tried with a single point crossover with a locus randomly situated.

A linear ranking selection and a binary tournament selection of size 2 are tried.

Evolution Engine

An elitist strategy in which the best half of population is kept and a non-elitist strategy in which the new population is made of only children are tried.

Stopping Criterion

The presence of a glider gun is continuously checked. The test is inspired by Bays' test [27] and also used in [28]. After the evolution of the random configuration of cells, the pattern is isolated and tested in an empty universe. If a pattern P reappears at the same place with gliders around then the pattern P is a glider gun. When a glider gun is found the algorithm stops.

The value of the fitness function and the generation of the best rule are memorized. If after ten new generations the algorithm has not found a better rule the algorithm stops.

Thanks to these stopping criteria, an execution of the algorithm stops after an average of 38 generations.

3.2 Monte Carlo method

In one million randomly generated automata, the presence of glider guns after the evolution of a random configuration of cells was tested. The test is the one used for the stopping criterion of the algorithm described in Section 3.1. There are not any guns found by this method.

3.3 Tabu Search

A random automaton A is generated, two fitness functions are tried to measure the performance of this automaton:

All the automata obtained by mutating one bit among the unused subset, as described section 2.5, are tested by the fitness function. The best one who is not in a list L of the last chosen automaton is chosen to become the new automaton A. The sizes of 10, 100 and 1000 are tried for the list L. The presence of glider guns is checked in all the tested automata.

The algorithm stops when the best automaton, among the automata obtained by mutation, who is not in a list L is not better than the current automaton. With this stopping criterion, an execution of the algorithm stop after an average of 49, 42 and 35 generations depending on the size of the list L.

3.4 Discussion

For each of the values of the parameters, the number of executions which find a gun are shown in table 1.

Table 1. Number of executions from a total of 100 per experiment that find a gun under a given combination of parameters or operators. The three numbers correspond to the 1,5, and 10 mutation rates. No guns were found with the monte carlo algorithm.

Evolutionary algortihm:

	First fitness function:				Second fitness function:			
	Elitist		Non Elitist		Elitist		Non Elitist	
	Tournament	Ranking	Tournament	Ranking	Tournament	Ranking	Tournament	Ranking
No Crossover	16, 32, 8	14, 14, 13	42, 11, 12	8, 1, 4	15, 43, 12	19, 16, 14	50, 13, 15	8, 3, 5
Middle Crossover	21, 18, 17	21, 14, 10	64, 20, 19	54, 23, 19	31, 23, 16	18, 14, 16	84, 22, 25	64, 33, 23
Random Crossover	3,2,2	3, 3, 2	4, 2, 0	1, 2, 1	1,1,4	0, 2, 3	3, 4, 1	0, 3, 0

Tabu search:

	First fitness function	Second fitness function
List of size 10	12	20
List of size 100	15	25
List of size 1000	16	28

The best parameters for the evolutionary algorithm, among the tested ones, are a mutation rate of 1, a non elitist strategy, a tournament selection and a central crossover and second fitness function. The evolutionary algorithm with these parameters have been chosen to obtain the glider guns described in the next Sections.

The good results of the central crossover can be explained by the fact that the first 51 neighbourhood states determine the birth of cells, while the other 51 determine how they survive or die. The elitist strategy that kept half of the population is worse than the non-elitist strategy. An elitist strategy that just kept one parent could be tried, however.

The results of a monte carlo algorithm and tabu search,presented as benchmarks, are not as good as the evolutionary approach. The results of the evolutionary algorithms without crossover are about the same of the tabu search.

4 Results of the algorithm

The results of the genetic algorithm with the best parameters for the glider in figure 2 are described here.

4.1 Number of guns

In order to determine how many different glider guns were found, an automatic system that determines if a gun is new is required. So, in order to determine if a gun is new, the set of neighbourhood states used by the given gun are compared to the ones of the other guns. For each gun and each neighbourhood state, three cases exist:

- The neighbourhood state is not used by this gun.
- The neighbourhood state is used by this gun and the value of the central cell at the next generation is 0.
- The neighbourhood state is used by this gun and the value of the central cell at the next generation is 1.

Two guns are different iff at least one neighbourhood state is not in the same case for the two guns.

Thank to this qualification of different guns leads, through the experimentations, 10008 different glider guns were discovered. All these guns have emerged spontaneously from random configurations of cells. The 10008 guns can be found in [29] in Life format.

The total number of different guns findable by this algorithm is unknown but the evolution of the percentage of new guns among the last 1000 different found guns is given by the figure 4.

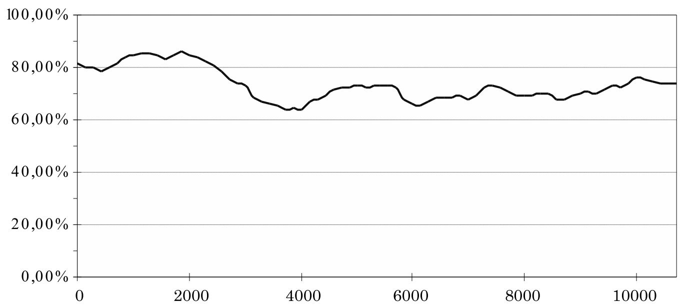

Fig. 4. Evolution of the percentage of new guns among 1000 different found guns.

Suppose each gun has the same probability to be found.

Let N be the total number of guns findable by this algorithm. The probability of a gun found by the algorithm to be new would be $1 - 10008/N$. The number of new guns among the last 1000 different found guns is 755. So the total number of guns findable by the algorithm could be estimated by $N = 10008 * 1000/245$ about 40849.

5 Synthesis and perspectives

This paper deals with the emergence of computation in complex systems with local interactions. A search for glider guns has been presented, building on previous work in [26, 28].

In particular, monte carlo method, tabu search and evolutionnary algorithms are explored with different parameters. The best results are found for an evolutionnary algorithm. The experimentation showed that cross over in the evolutionary algorithm plays a key role in the search process. Future work will consider other search techniques.

The algorithm succeeded in finding 10008 glider guns [29] for the glider of figure 2. The discovery of the emergence and existence of so many different glider guns for the same glider represent a significant contribution to the area of complex systems that considers computational theory.

Further goals can be to find all the glider guns possible and to calculate how many automata exhibit these guns. All these automata may be potential candidates for being shown universal automata thanks to an automatic system for the demonstration of universal automata that can be developed. Then, another domain that seems worth exploring is how this approach could be extended to automata with more than 2 states.

Future work could also calculate for each automata some rule-based parameters, e.g., Langton's lamda [30]. All automata exhibing glider guns may have similar values for these parameters that could lead to a better understanding of the link between the rule transition and the emergence of computation in cellular automata and therefore the emergence of computation in complex systems with simple components.

6 ACKNOWLEDGEMENTS

The work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of the United Kingdom, Grant EP/E005241/1.

References

1. S. Wolfram. *A New Kind of Science*. Wolfram Media, Inc., Illinois, USA, 2002.
2. J. Von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, Ill., 1966.
3. S. Wolfram. Universality and complexity in cellular automata. *Physica D*, 10:1–35, 1984.
4. E. R. Banks. *Information and transmission in cellular automata*. PhD thesis, MIT, 1971.
5. N. Margolus. Physics-like models of computation. *Physica D*, 10:81–95, 1984.
6. K. Lindgren and M. Nordahl. Universal computation in simple one dimensional cellular automata. *Complex Systems*, 4:299–318, 1990.
7. K. Morita, Y. Tojima, I. Katsunobo, and T. Ogiro. Universal computing in reversible and number-conserving two-dimensional cellular spaces. *In A. Adamatzky (ed.), Collision-Based Computing, Springer Verlag.*, pages 161–199, 2002.
8. A. Adamatzky. Universal dymical computation in multi-dimensional excitable lattices. *International Journal of Theoretical Physics*, 37:3069–3108, 1998.
9. M. Gardner. The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223:120–123, 1970.

10. E. Berlekamp, J. H. Conway, and R.Guy. Winning ways for your mathematical plays. *Academic Press, New York*, 1982.
11. A. Adamatzky G. J. Martinez and H. V. McIntosh. Phenomenology of glider collisions in cellular automaton rule 54 and associated logical gates chaos. *Fractals and Solitons*, 28:100–111, 2006.
12. A. Wuensche. Discrete dinamics lab (ddlab),www.ddlab.org, 2005.
13. D. Eppstein. http://www.ics.uci.edu/ eppstein/ca/.
14. E. Sapin, O. Bailleux, J.J. Chabrier, and P. Collet. A new universel automata discovered by evolutionary algorithms. *Gecco2004.Lecture Notes in Computer Science*, 3102:175–187, 2004.
15. S. Wolfram and N.H. Packard. Two-dimensional cellular automata. *Journal of Statistical Physics*, 38:901–946, 1985.
16. F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:533–549, 1986.
17. J. H. Holland. Adaptation in natural and artificial systems. *University of Michigan*, 1975.
18. N. H. Packard. Adaptation toward the edge of chaos. *In J. A. S. Kelso, A. J. Mandell, M. F. Shlesinger, eds., Dynamic Patterns in Complex Systems,Syngapore: World Scientific*, pages 293–301, 1988.
19. M. Mitchell, J. P. Crutchfield, and P. T. Hraber. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, 75:361–391, 1994.
20. P. T. Hraber, M. Mitchell and J. P. Crutchfield. Revisiting the edge of chaos : Evolving cellular automate to perform computations. *Complex systems*, 7:89–130, 1993.
21. W. Hordijk, J. P. Crutchfield and M. Mitchell. Mechanisms of emergent computation in cellular automata. *Parallel Problem Solving from Nature-V, A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel (eds.), Springer-Verlag*, 866:344–353, 1998.
22. R. Das, J. P. Crutchfield, M. Mitchell and J. E. Hanson. Evolving globally synchronized cellular automata. *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 336–343, 1995.
23. M. Sipper. Evolution of parallel cellular machines. *D. Stauffer, editor, Annual Reviews of Computational Physics*, V. World Scientific:243–285, 1997.
24. D. Andre, J. R. Koza, F. H. Bennett III and M. A. Keane. Genetic programming III: Darwinian invention and problem solving. *San Francisco, CA: Morgan Kaufmann.*, 1999.
25. J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: MIT Press, 1992.
26. E. Sapin, O. Bailleux, and J.J. Chabrier. Research of complex forms in the cellular automata by evolutionary algorithms. *EA03. Lecture Notes in Computer Science*, 2936:373–400, 2004.
27. C. Bays. Candidates for the game of life in three dimensions. *Complex Systems*, 1:373–400, 1987.
28. E. Sapin, O. Bailleux, and J.J. Chabrier. Research of a cellular automaton simulating logic gates by evolutionary algorithms. *EuroGP03. Lecture Notes in Computer Science*, 2610:414–423, 2003.
29. E. Sapin. http://uncomp.uwe.ac.uk/sapin/ea/gun.
30. C. L. Langton. Computation at the edge of chaos. *Physica D*, 42, 1990.