

## Two-state, $r = 1$ Cellular Automaton that Classifies Density

Mathieu S. Capcarrere,\* Moshe Sipper,\* and Marco Tomassini\*<sup>†</sup>

*Logic Systems Laboratory, Swiss Federal Institute of Technology,  
IN-Ecublens, CH-1015 Lausanne, Switzerland*

(Received 26 August 1996)

It has recently been shown that no one-dimensional, two-state cellular automaton can classify binary strings according to whether their density of 1s exceeds 0.5 or not. We show that by changing the output specification, namely, the final pattern toward which the system should converge, without increasing its computational complexity, a two-state,  $r = 1$  cellular automaton exists that can perfectly solve the density problem. [S0031-9007(96)01847-9]

PACS numbers: 89.80.+h, 02.70.Rw, 07.05.Bx

Cellular automata (CA) are discrete, dynamical systems that perform computations in a distributed fashion on a spatially extended grid. A cellular automaton consists of an array of cells, each of which can be in one of a finite number of possible states, updated synchronously in discrete time steps according to a local, identical interaction rule.

The dynamical behavior of a CA may give rise to emergent computation, referring to the appearance of global information processing capabilities that are not explicitly represented in the system's elementary components or in their local interconnections [1]. As such, they offer an austere yet versatile model for studying natural phenomena, as well as a powerful paradigm for attaining fine-grained, massively parallel computation.

An example of such emergent computation is to use a CA to determine the global density of bits in an initial state configuration. Known as the density classification problem, the one-dimensional, two-state CA is presented with an arbitrary initial configuration, and should converge in time to a state of all 1s if the initial configuration contains a density of 1s  $>0.5$ , and to all 0s if this density  $<0.5$ ; for an initial density of 0.5, the CA's behavior is undefined. Spatially periodic boundary conditions are used, resulting in a circular grid.

It has been shown that for a one-dimensional grid of fixed size  $N$ , and for a fixed radius  $r \geq 1$  [2], there exists no two-state CA rule which correctly classifies all possible initial configurations [3]; this says nothing, however, about how well an imperfect CA might perform. One such CA, known as the GKL rule [4], can correctly classify approximately 82% out of a random sample of initial configurations, for a grid of size  $N = 149$  [5]. Recently, researchers have focused on the use of artificial evolution techniques, demonstrating that high-performance (though imperfect) CAs can be evolved to solve this problem [5–8].

The density classification problem studied to date specifies convergence to one of two fixed-point configurations, which are considered as the output of the computation. The main result of this paper is that a perfect CA density

classifier exists, upon defining a different output specification. Consider the two-state,  $r = 1$  rule 184 CA [9], defined as follows:

$$s_i(t + 1) = \begin{cases} s_{i-1}(t), & \text{if } s_i(t) = 0, \\ s_{i+1}(t), & \text{if } s_i(t) = 1, \end{cases}$$

where  $s_i(t)$  is the state of cell  $i$  at time  $t$ . Upon presentation of an arbitrary initial configuration, the grid relaxes to a limit cycle, within  $[N/2]$  time steps, that provides a classification of the initial configuration's density of 1s: If this density  $>0.5$  (respectively,  $<0.5$ ), then the final configuration consists of one or more blocks of at least two consecutive 1s (0s), interspersed by an alternation of 0s and 1s; for an initial density of exactly 0.5, the final configuration consists of an alternation of 0s and 1s. The computation's output is given by the state of the consecutive block (or blocks) of same-state cells (Fig. 1); as proved in this paper, this rule performs perfect density classification (including the density = 0.5 case). We note in passing that the reflection-symmetric rule 226 holds the same properties of rule 184 studied below.

As the input configuration is random, this entails a high Kolmogorov complexity; intuitively, for a given finite string, this measure concerns the size of the shortest program that computes the string [10]. Both the fixed-point output of the original problem, as well as our own "blocks" output, involve a notable reduction with respect to this complexity measure. It has been noted by [8] that the computational complexity of the input is that of a nonregular language since a counter register is needed whose size is proportional to  $\ln(N)$ , whereas the fixed-point output of the original problem involves a simple regular language (all 0s or all 1s) [11]; we note that our novel output specification also involves a regular language (a block of two state-0 or state-1 cells). We thus conclude that our newly proposed density classifier is as viable as the original one with respect to these complexity measures, while surpassing the latter in terms of performance.

In the remainder of this paper we prove rule 184's ability to perfectly classify density. Throughout, we

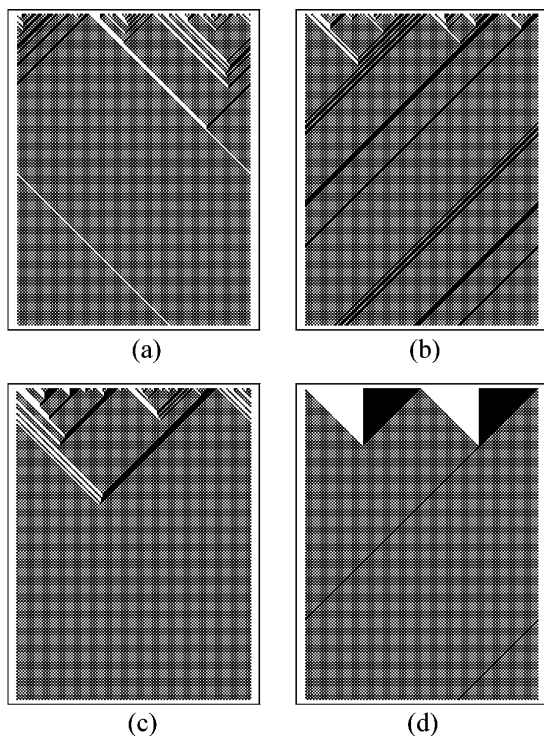


FIG. 1. Density classification: Demonstration of rule 184 on four initial configurations. White squares represent cells in state 0, black squares represent cells in state 1. The pattern of configurations is shown for the first 200 time steps, with time increasing down the page. Initial configurations in (a)–(c) were randomly generated. (a) Grid size is  $N = 149$ .  $D(S(0)) = 0.497$ , i.e., 75 cells are in state 0, and 74 are in state 1. The final configuration consists of an alternation of 0s and 1s with a single block of two cells in state 0. (b)  $N = 149$ .  $D(S(0)) = 0.537$ . The final configuration consists of an alternation of 0s and 1s with several blocks of two or more cells in state 1. (c)  $N = 150$ .  $D(S(0)) = 0.5$ . The final configuration consists of an alternation of 0s and 1s. (d)  $N = 149$ . Initial configuration consists of a block of 37 zeros, followed by 37 ones, followed by 37 zeros, ending with 38 ones. The final configuration consists of an alternation of 0s and 1s with a single block of two cells in state 1. In all cases the CA correctly classifies the initial configuration.

assume that cellular indices are computed modulus the grid size  $N$  (grid is circular), and that they are in the range  $\{0, \dots, N - 1\}$ ; for brevity we omit this range hereafter.

**Theorem.**—For a finite-size CA of size  $N$ , let  $S(t) = \{s_0(t), \dots, s_{N-1}(t)\}$  be the grid configuration at time step  $t$ , let  $D(\{s_i(t), \dots, s_{i+k-1}(t)\})$  be the density of 1s at time  $t$  over a block of  $k$  cells at positions  $\{i, \dots, i + k - 1\}$ , and let  $T = \lceil N/2 \rceil$ . Then note the following.

- (1) If  $D(S(0)) > 0.5$ , then (a) there exists a pair of adjacent cells  $i, i + 1$  such that  $s_i(T) = 1$  and  $s_{i+1}(T) = 1$ , and (b) for all  $i, s_i(T) = 0 \Rightarrow s_{i+1}(T) = 1$ .
- (2) If  $D(S(0)) < 0.5$ , then (a) there exists a pair of adjacent cells  $i, i + 1$  such that  $s_i(T) = 0$  and  $s_{i+1}(T) = 0$ , and (b) for all  $i, s_i(T) = 1 \Rightarrow s_{i+1}(T) = 0$ .
- (3) If  $D(S(0)) = 0.5$ , then for all  $i, s_i(T) \neq s_{i+1}(T)$ .

The proof of this theorem involves four lemmas, proved below.

**Lemma 1.**—For a finite-size CA of size  $N$ ,  $D(S(t + 1)) = D(S(t))$ ,  $t \in \{0, 1, \dots\}$ .

*Proof.*—Application of the transition rule to  $S(t) = 0^N$  or  $S(t) = 1^N$  yields  $S(t + 1) = 0^N$  or  $S(t + 1) = 1^N$ , respectively. Taking note of the grid’s circularity, any other configuration  $S(t)$  can be expressed as  $1^{a_1}0^{b_1}, \dots, 1^{a_n}0^{b_n}$ , where  $n \geq 1$ ,  $a_i, b_i > 0$ ,  $i \in \{1, \dots, n\}$ , and  $\sum_{i=1}^n a_i + b_i = N$ . It follows directly from the rule’s definition that a block  $1^{a_i}0^{b_i}$  at time  $t$  is transformed into  $1^{a_i-1}010^{b_i-1}$  at time  $t + 1$ . Thus, each such block conserves its density over one time step and the lemma is proven [12].

**Corollary 1.1.**—For a finite-size CA of size  $N$ ,  $D(S(t + 1)) = D(S(0))$ ,  $t \in \{0, 1, \dots\}$ .

*Proof.*—Follows directly from lemma 1 by recursive application.

**Lemma 2.**—Given a block  $x0^\alpha 1^\beta y$ ,  $x, y \in \{0, 1\}$ ,  $2 \leq \alpha, \beta \leq N - 2$ , at time  $t$ , beginning at cell  $i$  (i.e., cell  $i$  is the block’s leftmost cell), then at time  $t + v$ ,  $v = \min(\alpha, \beta) - 1$ , and beginning at cell  $i + v$ : (1) If  $\alpha > \beta$  there is a block  $x0^{\alpha-\beta+1}1y$ . (2) If  $\beta > \alpha$  there is a block  $x01^{\beta-\alpha+1}y$ . (3) If  $\alpha = \beta$  there is a block  $x01y$ .

*Proof.*—Applying the transition rule to a block  $x0^\alpha 1^\beta y$  at time  $t$ , beginning at cell  $i$ , results at time  $t + 1$  in a block  $x0^{\alpha-1}1^{\beta-1}y$  beginning at cell  $i + 1$ . By simple recursion, at time  $t + u$ ,  $u \leq \min(\alpha, \beta) - 2$ , there is a block  $x0^{\alpha-u}1^{\beta-u}y$ , beginning at cell  $i + u$ .

Consider the case  $\alpha > \beta$ , and let  $u = \beta - 2$ . At time  $t + u$ , there is a block  $x0^{\alpha-\beta+2}1^2y$ , beginning at cell  $i + u$ . Applying the transition rule results at the next time step,  $t + v$ ,  $v = \beta - 1$ , in a block  $x0^{\alpha-\beta+1}1y$ , beginning at cell  $i + v$ . The case  $\beta > \alpha$  follows analogously.

For  $\alpha = \beta$ , at time  $t + \beta - 2$  there is a block  $x0^21^2y$ , beginning at cell  $i + \beta - 2$ , which yields a block  $x01y$  at the next time step, beginning at cell  $i + \beta - 1$  ( $= i + v$ ).

**Lemma 3.**—Given a block  $0^\alpha$  (respectively,  $1^\alpha$ ),  $1 \leq \alpha \leq N$ , at time  $t$  beginning at cell  $i$ , then at time  $t - u$ ,  $u \leq t$ , there was a block  $0^\alpha$  ( $1^\alpha$ ) beginning at cell  $i - u$  ( $i + u$ ).

*Proof.*—By contradiction (we prove the  $0^\alpha$  case, with the  $1^\alpha$  case following analogously). Suppose at time  $t - 1$  there exists  $j \in \{i - 1, \dots, i + \alpha - 2\}$ , such that  $s_j(t - 1) = 1$ . Given that  $s_{j+1}(t) = 0$ , this implies that  $s_{j+1}(t - 1) = 1$  and  $s_{j+2}(t - 1) = 0$ ; however, this results in  $s_j(t) = 1$  and  $s_{j+2}(t) = 1$ , at least one of which must be within the  $0^\alpha$  block beginning at cell  $i$ , thus contradicting the assumption. Therefore, at time  $t - 1$  there is a block  $0^\alpha$  beginning at cell  $i - 1$ ; the lemma is proven by recursively applying this argument.

**Lemma 4.**—For a finite-size CA of size  $N$ , no two blocks  $0^\alpha$  and  $1^\beta$ ,  $2 \leq \alpha, \beta \leq N - 2$ , can coexist at time  $\lceil N/2 \rceil$ .

*Proof.*—By contradiction. If two blocks  $0^\alpha$  and  $1^\beta$  coexist at time  $[N/2]$ , then by lemma 3 they coexisted at time 0 (each shifted by  $[N/2]$  cells). This means that over the  $[N/2]$  time steps the blocks had been displaced by one cell per time step, the  $0^\alpha$  block “moving” to the right, the  $1^\beta$  block moving to the left. Thus, both blocks would “meet” after  $[(N - (\alpha + \beta))/2]$  time steps at the latest, satisfying the conditions of lemma 2. This implies that, at most, one block would remain after  $[(N - (\alpha + \beta))/2 + \min(\alpha, \beta) - 1]$  time steps. This latter expression  $< [N/2]$ , thus proving the lemma.

*Theorem proof.*—According to corollary 1.1, the density of the initial configuration is preserved at each successive time step. According to lemma 4, after  $[N/2]$  steps only  $0^\alpha$  or  $1^\beta$  blocks exist,  $2 \leq \alpha, \beta \leq N - 2$ , but not both (except for density = 0.5, where no such block exists). This means that the “correct” block must exist, with no occurrence of the “incorrect” one, thereby proving the theorem. We also note that after  $[N/2]$  time steps the number of cells in state 1 short (respectively, in excess) of  $[0.5N]$  is given by  $(\sum_{i=1}^m a_i) - m$ , where  $m$  is the number of  $0^{a_i}$  ( $1^{a_i}$ ) blocks, and  $a_i$  are their respective sizes.

Note that in order to “read” the output one can either terminate the CA’s execution after  $[N/2]$  time steps, or, alternatively, let it continue running (for a maximum of  $N - 1$  additional time steps) until the (cycling) two-cell, same-state block arrives at two predetermined cells.

Are there any other density classifiers in the two-state,  $r = 1$  class of CAs? Our experimental results suggest that rules 184 and 226 are the only ones that perform perfect density classification with respect to the output specification discussed in this paper [13].

In summary, we have shown that a locally specified,  $r = 1$  CA of any finite size  $N$  can classify the global density of bits for an arbitrary initial state configuration. It has previously been determined that this problem cannot be resolved by two-state CAs of any radius, if one insists on a fixed-point output. By changing the output specification, without increasing its complexity, perfect density classification can be attained. It is interesting that the system giving rise to this (difficult) emergent computation exists in the simplest class of one-dimensional CAs, namely, two-state,  $r = 1$ . This raises the intriguing question of whether other such simple CAs exist, which, while not capable of universal computation, may nonetheless prove highly efficient in solving specific tasks.

We are grateful to Daniel Mange and Jacques Zahnd for helpful discussions and their careful reading of this manuscript.

---

\*Electronic address of authors: {name.surname}@di.epfl.ch

†Also at the Swiss Scientific Computing Center, Via Cantonale, CH-6928 Manno, Switzerland.

- [1] *Emergent Computation: Self-organizing, Collective, and Cooperative Phenomena in Natural and Artificial Computing Networks*, edited by S. Forrest (MIT Press, Cambridge, MA, 1991).
- [2] In a one-dimensional CA, each cell has  $2r + 1$  neighbors ( $r$  cells on either side, as well as itself), where  $r$  is referred to as the radius.
- [3] M. Land and R.K. Belew, Phys. Rev. Lett. **74**, 5148 (1995).
- [4] P. Gacs, G.L. Kurdyumov, and L. A. Levin, Probl. Pereda. Inf. **14**, 92 (1978).
- [5] D. Andre, F.H. Bennett III, and J.R. Koza, in *Genetic Programming 1996: Proceedings of the First Annual Conference*, edited by J.R. Koza, D.E. Goldberg, D.B. Fogel, and R.L. Riolo (MIT Press, Cambridge, MA, 1996), pp. 3–11.
- [6] M. Sipper and E. Ruppin, Physica D (Amsterdam) (to be published).
- [7] M. Sipper, Physica (Amsterdam) **92D**, 193 (1996).
- [8] M. Mitchell, J.P. Crutchfield, and P.T. Hraber, Physica (Amsterdam) **75D**, 361 (1994).
- [9] Rule numbers are given in accordance with Wolfram’s convention, see S. Wolfram, Rev. Mod. Phys. **55**, 601 (1983).
- [10] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications* (Springer-Verlag, New York, 1993).
- [11] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory Languages and Computation* (Addison-Wesley, Redwood City, CA, 1979).
- [12] A different proof of this lemma, studied in a more general framework, is given by T. Hattori and S. Takesue, Physica (Amsterdam) **49D**, 295 (1991).
- [13] Each of the 256 two-state,  $r = 1$  CA rules was tested on 1000 randomly generated initial configurations for a grid of size  $N = 149$ , where a correct output is considered to be that specified by the theorem. As expected, rules 184 and 226 yielded a success rate of 100%, followed by rules 57 and 99 trailing markedly behind at 60%. Interestingly, these latter two rules produce patterns that are visually similar to rules 184 and 226; however, the simple aforementioned test reveals their complete inadequacy.