A Stochastic Model of Self-Stabilizing Cellular Automata for Consensus Formation

Stefania Monica
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma
43124 Parma, Italy
Email: stefania.monica@studenti.unipr.it

Federico Bergenti
Dipartimento di Matematica e Informatica
Università degli Studi di Parma
43124 Parma, Italy
Email: federico.bergenti@unipr.it

***Abstract*—In this paper we present a model of the dynamics of an interesting class of stochastic cellular automata. Such automata are variants of automata used for *density classification* and they are chosen because they can be effectively used to address consensus problems. After introducing the topic and the basic notation, we study the dynamics of such automata by means of simulations with varying periods and neighborhood structures. We use the results of simulations to extrapolate a stochastic model of the dynamics of such automata that can be used to estimate stabilization time.**

I. Introduction

In this seminal work dated 1966 [1], John von Neumann first introduced *Cellular Automata* (*CA*) as computing systems capable of self-reproduction and self-organization. Informally, we can think of a CA as a system made of a grid of cells each of which performs a (simple) computation. Each cell is connected to a set of neighbors in order to facilitate information exchange across the grid. All cells are equipped with a single rule that drives all their computations: they all compute the same function synchronously and the complex behavior of the system emerges from *(i)* the synchronous application of the same rule to different data, and *(ii)* the flow of information across the grid.

The homogeneous behavior that characterizes CA makes them ideal for the simulation of the dynamics of complex physical systems and they find in such an application domain their most common use. CA have also been used in many other applications where fast and parallel computation is required, e.g., low-level, real-time vision. From a theoretical point of view, CA are an interesting model for massively parallel and synchronous computation.

Today we recognize in CA the possibility of modeling complex and decentralized systems that can exhibit interesting adaptation properties, as discussed later in this paper, and we witness a recent renewal of interest in the subject. In particular, we are mainly interested in CA as a conceptual agent-based model that captures essential features of consensus dynamics. We are not interested in CA as an implementation technology, and the results presented in this paper are meant to be used in implementations that adopt industrial-strength agent technology—JADE [2], WADE [3], and AMUSE [4]—to target mobile scenarios (see, e.g., [5]–[8]).

Moreover, we are interested in modeling the emergent behaviors of *wireless sensor networks* used to support accurate localization of static (see, e.g., [9]–[11]) and moving (see, e.g., [12], [13]) targets to enhance their adaptability to dynamic environments, and to improve their robustness.

Finally, we recognize that the conceptual framework presented in this paper can be effectively used to model agent-based cooperation (see, e.g., [14], [15]), and it can be employed to enhance the dynamism and the flexibility of *workforce management systems* that deal with large workforces in complex situations (see, e.g., [16]).

This paper is organized as follows. Section II sets the basic notation about CA and self-stabilizing CA. It extends the ordinary notation with the introduction of a stochastic extension of CA that allows adopting stochastic functions to drive CA. Section III presents the simulations that were performed to study the dynamics of a particular class of CA, and that were used to extrapolate a stochastic model that formalizes the dynamics of the studied CA. Finally, Section IV concludes the paper with a short summary of presented results.

II. Cellular Automata

In this section we select one of the available formal definitions of CA and we cite some classic results. The selected definition is reasonably the most general available and we opt for such a definition because it does not restrict neither the size nor the neighborhood structure of automata, which is crucial for the simulations described in Section III.

Definition II.1 *A* Cellular Automaton A *is a structure:*

$$A =< S,\ d,\ V,\ f >$$

where $S \neq \emptyset$ is a finite set of state symbols, $d \in \mathbb{N}^+$ is the dimension *of the automata, $V \subseteq L$ is a finite* neighborhood structure *over the lattice $L = \mathbb{Z}^d$, and $f : S^V \to S$ is a transition function known as* rule*. A* cell *is a point x in the lattice L.*

A CA associates a *state* $s \in S$ to each cell $x \in L$. A *global configuration* of states c is defined in:

$$S^L = \{c \mid c : L \to S\}.$$

The neighborhood structure $V \subseteq L$, if not empty, is a set of $m \in \mathbb{N}^+$ vectors used to build the local neighborhood of each cell:

$$V = \{v_i \in L \mid i = 1,\ 2,\ \ldots,\ m\}.$$

Given V, the *neighborhood* $V_x \subseteq L$ of each cell x is created by means of the Abelian group of translations T of L into itself, which is defined in the usual way as $< L, + >$ where $+ : L \to L$ is the vector sum in L:

$$\forall x \in L \quad V_x = \{x + v \mid v \in V\}.$$

The *local configuration* of states c_{V_x} of a cell $x \in L$ can be defined using the global configuration as:

$$\begin{aligned} c_{V_x} : V_x &\to S \\ v &\mapsto c(x + v). \end{aligned}$$

The local configuration of states allows completing the definition of the rule of a CA as:

$$\begin{aligned} f : S^V &\to S \\ c_V &\mapsto s. \end{aligned}$$

In other words, for each cell $x \in L$, the rule f associates to the given local configuration of states c_{V_x} the future state $s \in S$ of x. This is better captured if we rewrite f as:

$$\forall x \in L \quad f(c_{V_x}) = f(s_1,\ s_2,\ \ldots,\ s_m)$$

where $s_i \in S$ are the states of the cells in the neighborhood of x: $s_i = c_{V_x}(v_i),\ v_i \in V$.

Given a global state c_0, a CA computes the following global states in terms of repeated and synchronous applications of f to all local configurations of states. Each repeated application of f is known as *step*.

We can finally define the function that a CA globally computes at each step as:

$$\begin{gathered} G_f : S^L \to S^L \\ \forall x \in L \quad [G_f(c)]\,(x) = f(c_{V_x}). \end{gathered}$$

Given an initial global configuration of states c_0, we can now compute the final global configuration of states after n steps as a repeated composition:

$$c_n = G_f^n(c_0).$$

CA as briefly formalized in this section are characterized by the following major features:

- Synchrony: computation is performed synchronously at each cell.
- Locality: computation is performed locally at each cell and global computation emerges from independent local computations.
- Homogeneity: all cells compute according to the same local function.
- Lack of memory: cells compute only on the basis of the current states of the cells in their respective neighborhood and no memory of past states is preserved.

An outstanding result of this formalization of CA is that CA are equivalent to Universal Turing Machines [17].

In practical applications of CA we are not interested in automata that span the entire lattice L; rather, we are often interested in automata defined over finite subsets of L. We introduce the restriction of CA over finite sets of L by means of so called *periodic* CA, as follows.

Definition II.2 *A cellular automaton A is* periodic *with period $l \in L$ if and only if for any initial configuration $c_0 \in S^L$:*

$$\forall n \geq 0,\ \forall x \in L \quad c_n(x + l) = c_n(x).$$

Such a definition is significant because of a classic result that states that a CA is periodic with period $l \in L$ if and only if its initial configuration c_0 is periodic with period l, i.e.:

$$\forall x \in L \quad c_0(x + l) = c_0(x).$$

In our work we are mainly interested in using CA to model complex systems intended to adapt to varying situations and capable of performing well under diverse conditions, especially for tasks that require decentralized coordination. This is the reason why we propose the following new class of CA:

Definition II.3 *A CA $A =< S,\ d,\ V,\ f >$ is called* self-stabilizing *if and only if:*

$$\begin{gathered} \forall c_0 \in S^L, \exists n \geq 0, \exists k \in S \quad \text{s.t.} \\ \forall m \geq n,\ \forall x \in L \quad c_m(x) = [G_f^m(c_0)](x) = k. \end{gathered}$$

Such a definition is very restrictive because it requires self-stabilizing CA to converge to *stable global configurations* characterized by all cells having the same state, which is a behavior that is known to be difficult to obtain. For example, it is well known that no periodic CA can solve the *majority problem* for all initial configurations [18]: no periodic CA can converge to stable global configurations that correctly discriminate if the majority of cells in the initial configuration were in state 0 or in state 1.

We need to extend CA is some way to ensure that the definition of self-stabilizing CA have some practical interest. One possibility is to have different rules across the lattice, which is a possibility that we have already explored and that we adopted to use CA for complex classification tasks [19], [20]. In this work we are interested in exploring another possibility: the use of *stochastic rules*. Such rules have already been used to solve the majority problem with arbitrary precision [21] and we intend to use them to support the development of a theory of self-stabilizing CA.

The introduction of stochastic rules in CA leads to the following definition¿

Definition II.4 *A* stochastic *CA A is a structure:*

$$A =< S,\ d,\ V,\ \mathbf{f} >$$

where $S \neq \emptyset$ is a finite set of state symbols, $d \in \mathbb{N}^+$ is the dimension of the automaton, $V \subseteq L$ is a finite neighborhood structure over the lattice $L = \mathbb{Z}^d$, and $\mathbf{f} : S^V \to S$ is a stochastic transition function known as stochastic rule.

The only difference with a traditional CA is that the local computation of a stochastic CA is now based on a

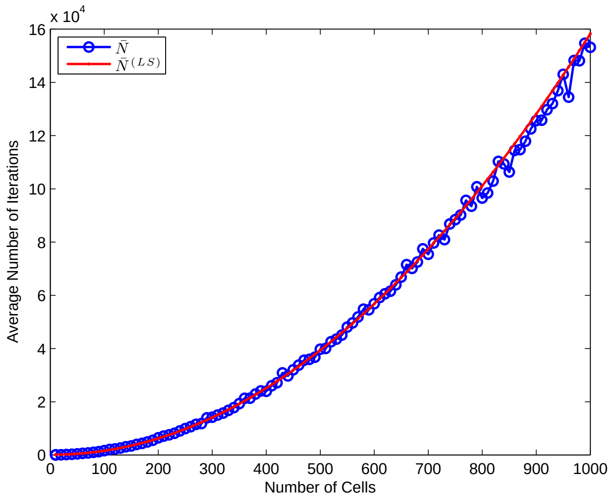

Fig. 1. Average number of iterations needed to reach a stable configuration (blue circles) and its quadratic LS approximation (red line) as a function of the number of cells for $m = 3$.

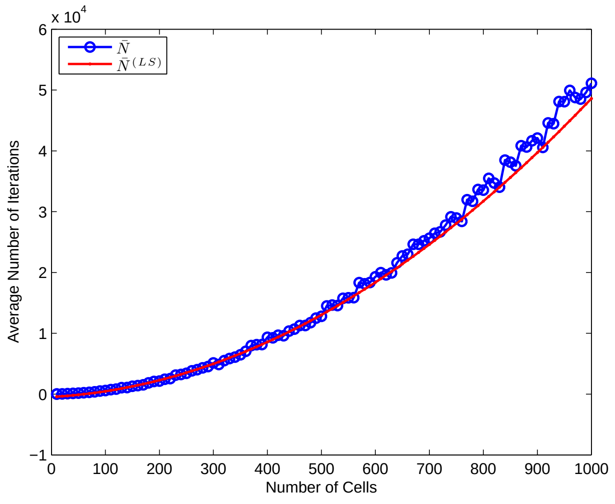

Fig. 2. Average number of iterations needed to reach a stable configuration (blue circles) and its quadratic LS approximation (red line) as a function of the number of cells for $m = 5$.

stochastic function and the global configuration of the CA evolves as a *discrete stochastic process*. Such a process is *Markovian* because the local computation that $\mathbf{f}$ performs depends only on current local configurations and no memory of past configurations is used. In the practice of stochastic CA, the stochastic rules are normally expressed as non-stochastic functions that depend on a random variable $\mathbf{v}$.

The definition of stochastic CA allows extending self-stabilizing CA as follows.

Definition II.5 *A stochastic CA* $A =< S,\ d,\ V,\ \mathbf{f} >$ *is called* self stabilizing *if and only if:*

$$\forall c_0 \in S^L, \exists n \geq 0, \exists k \in S \quad \text{s.t.} \quad \forall \mathrm{m} \geq \mathrm{n}$$
$$P\{\forall x \in L, c_m(x) = [G_f^m(c_0)](x) = k\} = 1.$$

III. Proposed Stochastic Model

The stochastic model of the dynamics of a particular class of self-stabilizing CA that we develop in this section is based on extrapolations from simulations. Such simulations are performed under common assumptions: *(i)* only 1-dimensional CA are considered ($d = 1$); *(ii)* only binary CA are considered ($S = \{0, 1\}$); and only periodic CA with period $l \in \mathbb{Z}^+$ are considered.

Under these assumptions, CA are defined over the lattice $L = \mathbb{Z}_l$, the set of integers modulo l, and the period l is the actual number of cells in considered CA. We do not assume a specific neighborhood structure and therefore the proposed model is not limited to so called *(stochastic) elementary CA* [21].

In our simulations, we consider different values of the period l: from 100 to 1000 with step 10. We also consider neighborhood structures of different sizes: $m = 3$, $m = 5$, $m = 7$, and $m = 9$.

In each case, we assume that the neighborhood structure is $V = \{-(m-1)/2, \ldots, (m-1)/2\}$, so that the neighbors of a cell $x \in \mathbb{Z}_l$ are the previous $(m-1)/2$ cells, the following $(m-1)/2$ cells, and the cell x itself.

Let us denote as $n_{0,x}$ the number of cells in state 0 in the neighborhood V_x of a generic cell x, and as $n_{1,x} = m - n_{0,x}$ the number of cells in state 1 in the same neighborhood. We define the stochastic rule of the class of CA that we study as:

$$\mathbf{f}(s_{x+v_1}, \ldots, s_{x+v_m}) = \begin{cases} 1 & w.p. \ \dfrac{n_{1,x}}{m} \\ 0 & w.p. \ \dfrac{n_{0,x}}{m} \end{cases} \tag{1}$$

where $\{s_{x+v_i}\}_{i=1}^m$ is the states of cells $\{x + v_i\}_{i=1}^m$ in the neighborhood V_x of cell x.

The studied rule says that the probability for a cell to be in state 0 (resp. 1) in the next step is directly proportional to the number of cells in state 1 (resp. 0) in its neighborhood in the current step. The rule ensures that when all cells in the neighborhood of a cell x are in state 0 (resp. 1), next state for x will be 0 (resp. 1) with probability 1.

Due to the randomness in the rule defined in (1), the number of steps needed to reach a stable global configuration, denoted as $\mathbf{N}$, is a random variable and we denote its average value as $\bar{N}$. We are interested in analyzing the behavior of $\bar{N}$ as a function of the number of cells l and of the neighborhood size m. We show that $\bar{N}$ can be accurately approximated as a quadratic function of l, regardless of m.

We consider values of the period l from 100 to 1000, with step 10. For each of these values and for each of the considered values of m, we perform 1000 (independent) simulation runs, each of which is randomly initialized. We derive a quadratic approximation $\bar{N}_{\mathrm{m}}^{(LS)}$ of $\bar{N}$ by applying the Least Squares (LS) technique to the results of simulations.

Let us start by considering the results for $m = 3$. The local stochastic rule defined in (1) corresponds to the rule of a

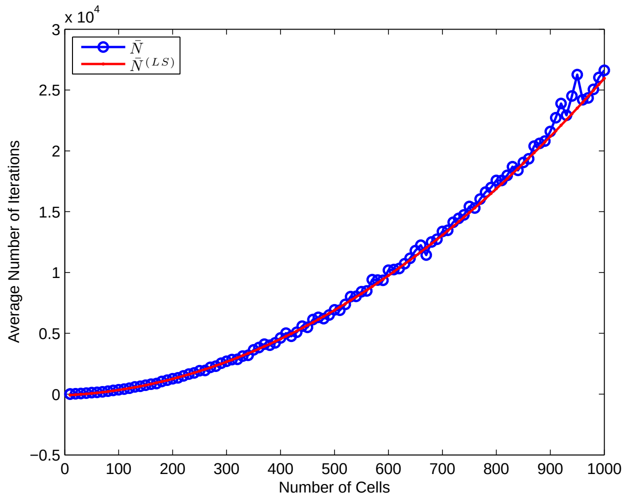

Fig. 3. Average number of iterations needed to reach a stable configuration (blue circles) and its quadratic LS approximation (red line) as a function of the number of cells for $m = 7$.

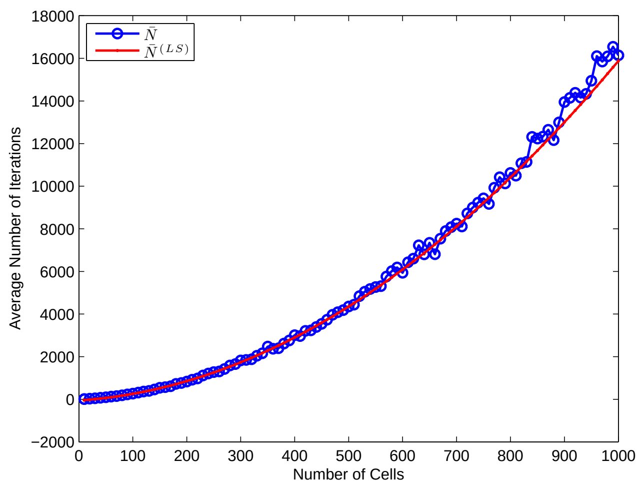

Fig. 4. Average number of iterations needed to reach a stable configuration (blue circles) and its quadratic LS approximation (red line) as a function of the number of cells for $m = 9$.

well known CA called *Fukś density classifier* for the specific case of $p = \frac{1}{3}$ (see [22]).

In Figure 1, the values of $\bar{N}$ (averaged over the 1000 runs) are shown (blue circles) as a function of the period l which varies from 100 to 1000. As intuitively expected, $\bar{N}$ is an increasing function of the period l, i.e., more iterations are needed (on average) to reach a stable configuration if a greater number of cells is considered. More precisely, Figure 1 shows that the values of $\bar{N}$ can be accurately approximated as a quadratic function of l, in agreement with [21]. In particular, the LS technique applied to the considered samples leads to the following quadratic approximation

$$\bar{N} \simeq \bar{N}_3^{(LS)}(l) = 0.16l^2 - 1.10l + 99.$$

In Figure 1, the values of $\bar{N}_3^{(LS)}(l)$ (red line) are shown and it can be noticed that they fit well the measured values of $\bar{N}$ for all the values of l. If we define the relative error on a sample as:

$$\varepsilon(l) = \frac{|\bar{N}(l) - \bar{N}_3^{(LS)}(l)|}{\bar{N}(l)}$$

we obtain that the average relative error (averaged over the considered values of l) is 2.14%.

We now consider a larger neighborhood structure with $m = 5$. As in the case with $m = 3$, we perform 1000 simulation runs for values of the period l from 100 to 1000 in order to investigate the average number of steps $\bar{N}$ needed to reach a stable configuration. Figure 2 shows the values of $\bar{N}$ (blue circles) as a function of the period l and the quadratic LS approximation $\bar{N}_5^{(LS)}(l)$ (red line), which is:

$$\bar{N}_5^{(LS)}(l) = 0.04l^2 + 4.93l - 459.66.$$

The quadratic approximation is good also in this case, and the average relative error (averaged over the considered values of l) is 4.6%.

Similarly, in Figure 3 the values of $\bar{N}$ (blue circles) obtained considering the neighborhood with $m = 7$ are shown. The quadratic approximation $\bar{N}_7^{(LS)}(l)$ given by

$$\bar{N}_7^{(LS)}(l) = 0.02l^2 + 1.94l - 96.03$$

is also shown (red line). Once again, the LS approximation obtained on the values of l is accurate as the average relative error is 2.4%.

Finally, Figure 4 shows the values of $\bar{N}$ (blue circles) obtained considering the neighborhood of size $m = 9$ and their quadratic approximation $\bar{N}_9^{(LS)}(l)$ (red line), given by

$$\bar{N}_9^{(LS)}(l) = 0.01l^2 + 1.70l - 53.94.$$

Also in this last case the quadratic approximation is good and it leads to an average relative error of 2.84%.

From the presented result, we can conclude that the larger is the neighborhood, the faster is the convergence to a stable global configuration, which is by far not surprising. Moreover, even if in all cases $\bar{N}$ is an increasing function of l which can be accurately approximated as a quadratic function, a comparison between the obtained results shows that larger neighborhoods have functions that increase slower.

We now focus on the results obtained with $l = 100$, for all the values of $m \in \{3, 5, 7, 9\}$ and we are interested in analyzing the *Probability Mass Function* (*PMF*) of the random variable $\mathbf{N}$.

Keeping m fixed, we consider the number of steps N needed to reach a stable global configuration in a simulation run, and we call $N_{\max}$ the maximum value of N for all runs. We then divide the interval $[0, N_{\max}]$ into 50 subintervals I_j, $j \in \{1, \dots, 50\}$. We then count, for each interval I_j, the number of times that N falls into I_j. Finally, counts are normalized to obtain a PMF. The results are shown in Figure 5 for $m = 3$, Figure 6 for $m = 5$, Figure 7 for $m = 7$, and Figure 8 for $m = 9$.

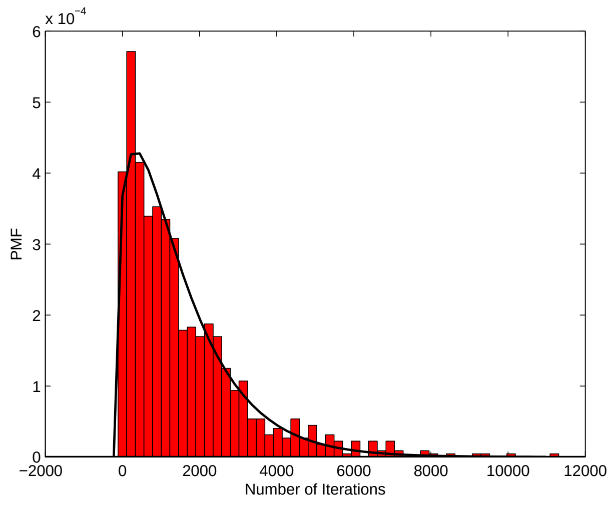

Fig. 5. PMF of the random variable $\bar{N}$ for $l = 100$ and $m = 3$.

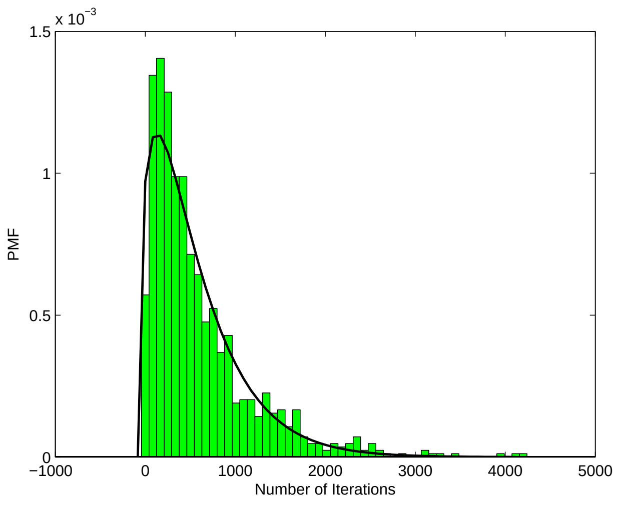

Fig. 6. PMF of the random variable $\bar{N}$ for $l = 100$ and $m = 5$.

In each case, we can argue that the PMF can be approximated as a *gamma distribution* $p(t) = at^{1/2}e^{-bt}$, defined for $t \geq 0$, where a and b are constants that need to be properly set in order to make sure that the integral of $p(t)$ on its domain is equal to 1 and that the average value is equal to $\bar{N}$. These two conditions lead to the following expressions for the coefficients a and b:

$$\int_0^{+\infty} p(t)\mathrm{d}t = 1 \qquad \int_0^{+\infty} tp(t)\mathrm{d}t = \bar{N}. \tag{2}$$

Introducing the change of variable $\tau = bt$, the first integral in (2) can be evaluated as follows:

$$\frac{a}{b^{3/2}}\int_0^{+\infty} \tau^{1/2}e^{(-\tau)}\mathrm{d}\tau = \frac{a}{b^{3/2}}\Gamma\left(\frac{3}{2}\right) = \frac{a\sqrt{\pi}}{2b^{3/2}}.$$

The same change of variable leads to the following formula

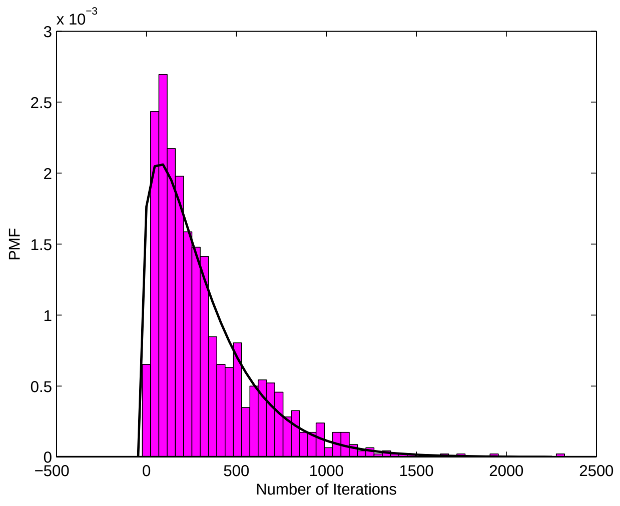

Fig. 7. PMF of the random variable $\bar{N}$ for $l = 100$ and $m = 7$.

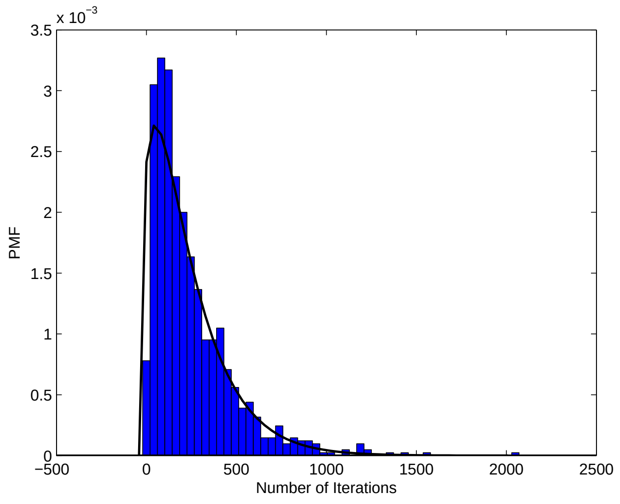

Fig. 8. PMF of the random variable $\bar{N}$ for $l = 100$ and $m = 9$.

for the second integral in (2):

$$\frac{a}{b^{5/2}}\int_0^{+\infty} \tau^{3/2}e^{(-\tau)}\mathrm{d}\tau = \frac{a}{b^{5/2}}\Gamma\left(\frac{5}{2}\right) = \frac{3a\sqrt{\pi}}{4b^{5/2}}.$$

Still the same change of variable can be applied to the second integral in (2) and, given that the conditions in (2) link a and b with the average value $\bar{N}$, we can obtain the following formulas for the coefficients a and b as a function of $\bar{N}$

$$\frac{a\sqrt{\pi}}{2b^{3/2}} = 1 \qquad \frac{3a\sqrt{\pi}}{4b^{5/2}} = \bar{N} \tag{3}$$

which is equivalent to

$$a = \frac{2}{\sqrt{\pi}}\left(\frac{3}{2\bar{N}}\right)^{3/2} \qquad b = \frac{3}{2\bar{N}}. \tag{4}$$

In Figure 9 the *Cumulative Distribution Function* (*CDF*) of the random variable $\mathbf{N}$ is shown, for $l = 100$ and for the considered values of m. As expected, greater values of the neighborhood size fasten the convergence of the CDF to 1.

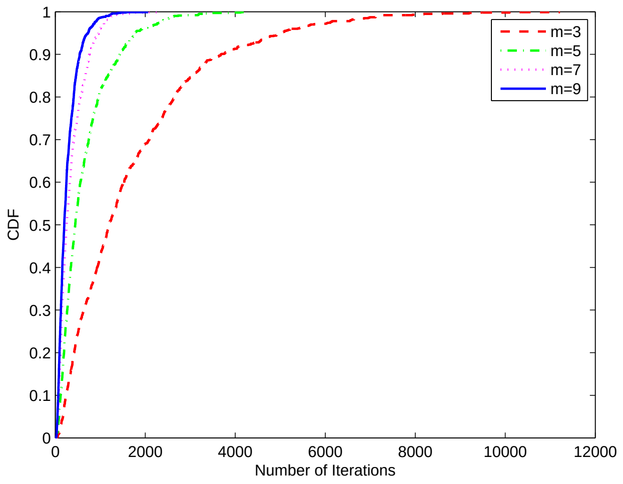

Fig. 9. CDF of the random variable N for $l = 100$ and: $m = 3$ (dashed red line), $m = 5$ (dash-dotted green line), $m = 7$ (dotted magenta line), $m = 9$ (solid blue line).

IV. Conclusions

The stochastic model of the dynamics of some self-stabilizing CA introduced in this paper is an important tool both for the theoretical study of the properties of such automata, and for their practical applications to solve consensus problems. The model provides a good approximation of the dynamics of such automata and it also provides a means to estimate the number of steps needed to converge to a stable configuration. This possibility is particularly important for the practical application of studied CA to solve consensus problems in real-world situations where an estimation of the expected stabilization time is always demanded.

References

[1] J. von Neumann, *Theory of Self-Reproducing Automata*. University of Illinois Press, 1966.

[2] JADE (Java Agent DEvelopment framework) web site. [Online]. Available: http://jade.tilab.com

[3] WADE (Workflows and Agents Development Environment) web site. [Online]. Available: http://jade.tilab.com/wade

[4] AMUSE (Agent-based Multi-User Social Environment) web site. [Online]. Available: http://jade.tilab.com/amuse

[5] F. Bergenti, G. Caire, and D. Gotta, "Agents on the move: JADE for Android devices," in *Procs. Workshop From Objects to Agents*, 2014.

[6] F. Bergenti, G. Caire, and D. Gotta, "Interactive workflows with WADE," in *Procs. IEEE Int'l Conf. Enabling Technologies: Infrastructures for Collaborative Enterprises*, 2012, pp. 10–15.

[7] F. Bergenti, G. Caire, and D. Gotta, "Agent-based social gaming with AMUSE," in *Procs.* 5^{th} *Int'l Conf. Ambient Systems, Networks and Technologies (ANT 2014) and* 4^{th} *Int'l Conf. Sustainable Energy Information Technology (SEIT 2014)*, ser. Procedia Computer Science, vol. 32, 2014, pp. 914–919.

[8] F. Bergenti, G. Caire, and D. Gotta, "An overview of the AMUSE social gaming platform," in *Procs. Workshop From Objects to Agents*, 2013, pp. 85–90.

[9] S. Monica and G. Ferrari, "Particle swarm optimization for auto-localization of nodes in wireless sensor networks," in 11^{th} *International Conference on Adaptive and Natural Computing Algorithms (ICANNGA 2013)*, 2013.

[10] S. Monica and G. Ferrari, "Impact of the number of beacons in PSO-based auto-localization in UWB networks," in *International Conference on the Applications of Evolutionary Computation (EvoApplications 2013), track on Nature-inspired Techniques for Communication Networks and other Parallel and Distributed Systems (EvoCOMNET 2013)*, 2013, pp. 42–51.

[11] S. Monica and G. Ferrari, "Swarm intelligent approaches to auto-localization of nodes in static UWB networks," *Applied Soft Computing*, in press.

[12] S. Monica and G. Ferrari, "Optimized anchors placement: an analytical approach in UWB-based TDOA localization," in *Proceedings of the* 9^{th} *International Wireless Communications and Mobile Computing Conference (IWCMC 2013)*, 2013, pp. 982–987.

[13] S. Monica and G. Ferrari, "UWB-based localization in large indoor scenarios: Optimized placement of anchor nodes," *IEEE Transactions on Aerospace and Electronic Systems*, in press.

[14] F. Bergenti and A. Poggi, "Agent-based approach to manage negotiation protocols in flexible CSCW systems," in *Procs.* 4^{th} *Int'l Conf. Autonomous Agents*, 2000, pp. 267–268.

[15] F. Bergenti, A. Poggi, and M. Somacher, "A collaborative platform for fixed and mobile networks," *Communications of the ACM*, vol. 45, no. 11, pp. 39–44, 2002.

[16] F. Bergenti, G. Caire, and D. Gotta, "Large-scale network and service management with WANTS," in *Industrial Agents: Emerging Applications of Software Agents in Industry*, P. Leitão and S. Karnouskos, Eds. Elsevier, 2014.

[17] E. F. Codd, *Cellular Automata*. New York: Academic Press, 1968.

[18] L. Mark and R. Belew, "No perfect two-state cellular automata for density classification exists," *Physical Review Letters*, vol. 74, no. 25, pp. 1548–1550, 1995.

[19] S. Cagnoni, F. Bergenti, M. Mordonini, and G. Adorni, "Evolving binary classifiers through parallel computation of multiple fitness cases," *IEEE Transactions on Systems, Man and Cybernetics Part B–Cybernetics*, vol. 3, no. 35, pp. 548–555, 2005.

[20] F. Bergenti, "An evolutionary approach to agent-based pattern classification," *Communications of SIWN*, vol. 5, pp. 23–27, 2008.

[21] N. Fates, "Stochastic cellular automata solutions to the density classification problem," *Theory of Computing Systems*, vol. 53, no. 2, pp. 223–242, 2013.

[22] H. Fukś, "Nondeterministic density classification with diffusive probabilistic cellular automata," *Physical Review E*, vol. 66, no. 6, 2002.