

Evaluation of Update Rules for Parity Checking by Asynchronous Cellular Automata

Gerhard Hasslinger¹, Ulrich Schachner¹, Rolf Hoffmann¹

¹ Technical University of Darmstadt, Darmstadt, Germany

Abstract We investigate novel aspects of asynchronous Cellular Automata in a broad scope of rule variants for solving the parity problem. Given a one-dimensional binary vector, the task is to determine whether the parity over all bits is even or odd. Therefore, the convergence of an update process is studied with local updates on 2-, 3- or 4-bit neighborhood pattern following a uniform, parity preserving rule. We assume asynchronous random updates according to a Markov process, which is evaluated via simulation and analytical means. We compare and optimize rule variants regarding their convergence to two different final states, which indicate even versus odd parity. Beyond the all-one and all-zero configurations, we widen the scope to other final states and we include acyclic rule applications, which lead to promising convergence properties.

Keywords Asynchronous cellular automata, parity check, update rules, classification, convergence process, Markov state model

1 Introduction

Our objective is to find and analyze new update rules for an asynchronous Cellular Automaton (CA) that can solve the parity problem. It answers the question of whether a binary sequence (configuration, state) contains an even or odd number of ones, using only local interactions between cells. It is considered a classical benchmark for evaluating the computational abilities of a CA. In its classical form, the parity problem is solved for arbitrary configurations in a cell ring of odd size with synchronous updating [2][3][8][11] until a final fixed point state of all-zeros is reached to indicate even parity versus all-ones for odd parity.

1.1 Related Work

Parallel substitution algorithms [1] provide a general and powerful synchronous computational model based on substitution rules with decentralized control. They enable modifications of a set of arbitrary target cells using a basis and a context. The CA-write model [7] allows a cell to write information to its neighbors. Thus, a cell can change not only its own state/activation but also that of its neighbors. Therefore, this model is another option to describe the parity problem.

The classical parity problem with synchronous CA updating is addressed in [2][3][8] among other work, where the identification of a unique rule involving less than 9 neighbors still is open for future research [11]. This work stream assumes synchronous updates of the entire configuration by the same decision rule per cell, which involves the neighborhood of radius r . A different approach by Fatès [5] assumes stochastic asynchronous updates of a single local pattern in each step. Then a proposed 4-bit rule is sufficient to perform the parity check, which keeps or inverts both middle bits depending on the 4-bit neighborhood.

The convergence properties of an elementary asynchronous CA are classified in [6] for all 256 rules, which modify the center cell depending on a 3-bit neighborhood. Recent work [9] partly extends those convergence results to rule variants that modify a pair of bits depending on a 3-bit neighborhood. (A-)synchronous CA variants are compared in another recent study [4], where update rounds involving all cells in a random permutation order are considered as one option for a compromise of (a)synchronous modes.

1.2 Our approach and contributions

We investigate a stochastic and fully asynchronous CA approach as also studied by Fatès [5] in a broader scope of rule variants. We consider arbitrary final states or state classes to indicate even or odd parity, which are not restricted to odd ring sizes. Acyclic systems are included without rule application in positions requiring a cyclic wrap-around. We first focus on 2-bit rules involving only a pair of cells in each update step, which already enable a parity check with favorable convergence properties especially for acyclic systems. The results for the convergence time towards the final states are partly extended and optimized for cases of 3- and 4-bit rules. The mean convergence time is evaluated via simulation and transient analysis of the underlying Markov process for active as well as passive neighborhood pattern.

Paper structure Section 2 defines notations for the considered asynchronous CA environment with update rules and the Markov process for the system development. In Section 3, the convergence behavior of basic rules is evaluated via simulation and the convergence properties are summarized. Section 4 confirms and extends the simulation results using a transient analysis approach, whose scalability is limited by exponential complexity. The work finishes with a conclusion and options for future work.

2 Update Rules for Asynchronous Cellular Automata

2.1 System model and notations

We consider a cyclic ring of n binary cells $X = (x_1, \dots, x_n)$; $\forall k: x_k \in \{0, 1\}$. The asynchronous system performs local updates on the cells, where each m -bit update step depends on m neighboring cells $x_k, x_{k+1}, \dots, x_{k+m-1}$, whose binary pattern is transformed.

We assume that m -bit updates follow the same rule R , which overwrites a current m -bit pattern by a transformation function $f_R: \{0, 1\}^m \rightarrow \{0, 1\}^m$. When we denote $f_R(a_1, \dots, a_m) = (\tilde{a}_1, \dots, \tilde{a}_m)$ then an update step in position k transforms a configuration $(x_1, \dots, x_n)$ into $(x_1, \dots, x_{k-1}, \tilde{x}_k, \dots, \tilde{x}_{k+m-1}, x_{k+m}, \dots, x_n)$. This class of block transformations includes rules with radius r [2][3][9][11] as $(2r+1)$ -bit rules, but allows for changing the entire neighborhood of $2r+1$ cells beyond a cell in the middle.

For cyclic updates, the positions of the cells are evaluated modulo n ($n + k \equiv k$). We can distinguish *active* update patterns $f_R(a_1, \dots, a_m) \neq (a_1, \dots, a_m)$ and *passive* patterns, which remain unchanged under a considered rule.

We consider asynchronous update steps, each of which is performed in a random and uniformly chosen position $k = 1, \dots, n$ with probability $1/n$ according to the rule. Then the updates follow a discrete Markov process, where the Markov state corresponds to the current configuration $(x_1, \dots, x_n)$. Thus, a configuration is also denoted as a (Markov) state of the CA. We study the convergence behavior of the update process

from a random and uniformly chosen start configuration towards an absorbing state of the Markov chain, i.e., a configuration without any active update pattern. Regarding a parity check, 2 absorbing states are expected, which indicate even ($n_1 \bmod 2 = 0$) versus odd parity ($n_1 \bmod 2 = 1$) regarding to the number n_1 of 1-bits of the configuration.

2.2 Basic 2-bit update rules for a parity check

We first focus on update rules for preserving parity on a pair (x_k, x_{k+1}) of cells in a position $k = 1, \dots, n$, which invert a subset of the bit pattern of the pair:

$(x_k, x_{k+1}) \rightarrow (1 - x_k, 1 - x_{k+1})$, or, as transformations P, M, L, R on each bit pattern:

P (Plus two 1-bits): $00 \rightarrow 11$, M (Minus two 1-bits): $11 \rightarrow 00$,

L (shift to the left): $01 \rightarrow 10$, and R (shift to the right): $10 \rightarrow 01$.

There are 16 rule variants, which partially invert the 2-bit pattern. We denote a rule variant for pairwise updates by the subset of active pattern among $\{P, M, L, R\}$, e.g. the rule ML executes the transformations M and L, but omits P and R, such that the pattern 00 and 10 remain unchanged.

On a closer look, it is sufficient to focus on a subset of those 16 variants, because the others have equivalent behavior due to configuration symmetries of the CA [6][9]. Moreover, we assume the configurations $0 \dots 0$ and $10 \dots 0$ as the attracting final states for indicating even versus odd parity and we exclude rule variants, if those states cannot be entered from all arbitrary start configurations.

Consequently, we exclude the rules LR, L, R and $\{\}$ because they leave the number of 0- and 1-bits in the cells unchanged. The rules PM, P and M are excluded because they have no active pattern to change configurations of the type $0101 \dots 01$.

Instead, we focus on a set of 4 relevant update rules for pairs of cells in Table 1:

Table 1: Relevant 2-bit update rules

Rule	Pattern Updates	Comment
PMLR	$11 \leftrightarrow 00, 01 \leftrightarrow 10$	$0 \leftrightarrow 1$ -balanced rule with bidirectional shifts
PML	$11 \leftrightarrow 00, 01 \rightarrow 10$	$0 \leftrightarrow 1$ -balanced rule with shifts of 1-bits to the left
MLR	$11 \rightarrow 00, 01 \leftrightarrow 10$	Rule for reducing 1s with bidirectional shifts
ML	$11 \rightarrow 00, 01 \rightarrow 10$	Rule for reducing 1s with shifts of 1-bits to the left

All other rules that have not been excluded, are equivalent to one of those rules, i.e.,

$PMR \equiv PML$, $PR \equiv PL$ and $MR \equiv ML$ via left $\leftrightarrow$ right rotation of the configuration and

$PLR \equiv MLR$, $PL \equiv ML$ and $PR \equiv MR$ via bitwise exchange of 0- and 1-bits.

Both symmetries are also exploited in classifications and analysis of 3-bit rules [6][9].

3 Computational Evaluation of the Classification Time

3.1 Classification by 2-bit update rules

We use simulation and analytical arguments in order to study the classification time of the four remaining 2-bit update rules PMLR, PML, MLR and ML from a randomly chosen start configuration towards the final states $0 \dots 0$ and $10 \dots 0$.

The convergence behavior of the $0 \leftrightarrow 1$ -balanced rules differs from rules for reducing 1s. The updates for both rules PMLR and PML imply an ergodic Markov process on all 2^{n-1} states with the same even or odd parity. This process visits each of the 2^{n-1} configurations with the same steady state probability $1/2^{n-1}$ and does not converge.

Those Markov chains include a state $0 \dots 0$ or $10 \dots 0$ as candidate to indicate even versus odd parity, both as recurrent states. In order to establish $0 \dots 0$ and $10 \dots 0$ as absorbing states, they would have to be detected, such that further transitions from both states could be blocked by a modified rule. We measure the classification time (CT) until reaching $0 \dots 0$ or $10 \dots 0$ from a random, uniformly distributed initial configuration as the number of required (active) update steps. Then the mean CT for both rules PMLR and PML is exponentially increasing with the ring size n , because about half of the space of 2^{n-1} configurations is visited beforehand, see Table 2.

Table 2: Mean number of active updates until a first entrance to $0 \dots 0$ or $10 \dots 0$ from a random start configuration for the balanced pairwise update rule PMLR

Ring Size n	5	7	9	11	13	15	17	19	21	23
Mean CT	16.8	73.4	293.4	1221	4448	$1.8 \cdot 10^4$	$6.8 \cdot 10^4$	$2.8 \cdot 10^5$	$1.1 \cdot 10^6$	$4.4 \cdot 10^6$

On the other hand, the rules MLR and ML imply a Markov update process with more favourable but still imperfect convergence properties. We observe that the number n_1 of cells of value $x_k = 1$ (1-bits) in a configuration is reducing down to ≤ 1 during the update process for both rules. Each update step of the MLR and ML rule either preserves n_1 for L or R pattern ($01 \leftrightarrow 10$) or reduces n_1 when M: $11 \rightarrow 00$ is applied.

Consequently, the state $0 \dots 0$ is an absorbing state for even parity, while all configurations with $n_1 = 1$, i.e., $10 \dots 0, 010 \dots 0, \dots, 0 \dots 01$ form an attracting class of states for odd parity. Once the update process has entered this class of states, it stays within this Markov subchain, which is recurrent for the rule MLR or periodic for ML.

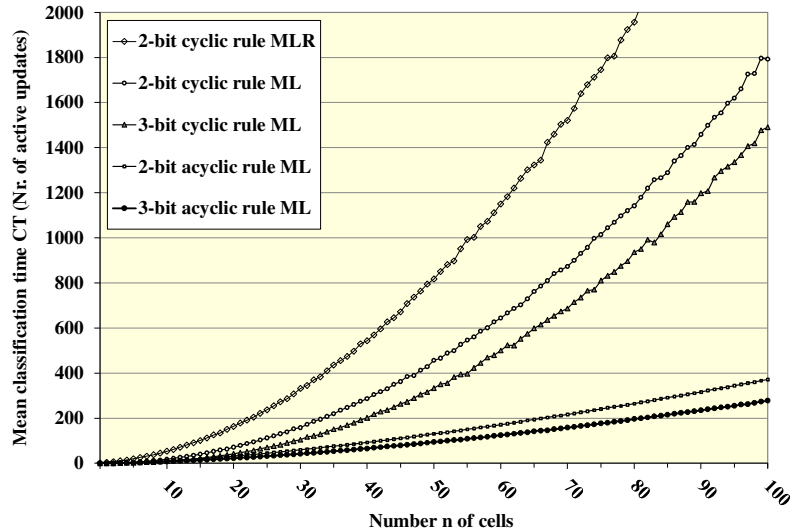


Figure 1: Mean time to reach a classifying state by 2-bit and 3-bit update rules

The two upper curves in Figure 1 show the mean time until the MLR and ML rule first enter the state $0 \dots 0$ for even parity or the state $10 \dots 0$ for odd parity. We start the simulations from a random, uniformly chosen configuration with probability 2^{-n} for each state. Then we perform transition steps according to each rule again at randomly chosen positions. We count the number of active update steps until one of both final states is entered. Passive update steps are considered later in the evaluations of Section 3.4. The results in Figure 1 are obtained as the mean values over 10 000 simulation runs, each of which is subject to a high standard deviation.

3.2 Acyclic rules for a unique and faster convergence

Acyclic CA rules omit m -bit updates with a cyclic wrap-around from the position(s), $n - m + 2, \dots, n$, e.g., there are no 2-bit updates on the pair (x_n, x_1) . This has major influence on the convergence behavior especially for the rules with only left (or only right) shifts of 1-bits. The acyclic ML rule shifts 1-bits towards the first positions via transition L, where they are concentrated and eliminated by transition M. The acyclic rule also implies an acyclic Markov process of transitions of active update patterns M and L, such that no configuration is revisited. Moreover, this Markov process converges to unique absorbing states $0 \dots 0$ for even parity and $10 \dots 0$ for odd parity.

Thus, the acyclic ML rule converges from arbitrary binary configurations to one of both intended final states, which indicate the parity in the first bit. The simulation results in Figure 1 show an essentially faster convergence of the acyclic ML rule as compared to the cyclic variants.

Moreover, the worst case classification time, i.e. the maximum number of active transitions of the acyclic ML rule is $n(n-1)/2$. For a proof we observe that a 1-bit in position k ($x_k = 1$) of a configuration can undergo a maximum number of $k-1$ shifts to the left via transition L $01 \rightarrow 10$, until it is in the first position. On the whole, the maximum number of transitions of type L is $\sum_{k=2}^n k-1 = n(n-1)/2$. In addition, there are transitions M $11 \rightarrow 00$ that reduce the number of 1-bits. For each transition of type M a 1-bit is eliminated in a position $k \geq 2$, which prevents at least one possible type L transitions. The maximum classification time $n(n-1)/2$ is encountered from the configuration $1 \dots 1$, if all transitions of type M are performed in the first position.

As a final note, a synchronized update process, which applies the 2-bit rule ML in decreasing order on the positions $k = n-1, n-2, \dots, 2, 1$, performs the parity check in $n-1$ update steps and delivers the result in the first bit.

3.3 Evaluation of 3-bit and 4-bit rules

Next, we compare the previous results for 2-bit rules with 3- and 4-bit rules. We cannot cover all parity preserving 3-/4-bit rule variants, but limit ourselves to cases with promising convergence properties. First, we extend the 2-bit ML rule into a 3-bit rule:

3-bit M_3L_3 rule: M_3 : $11 \rightarrow 00$ (i.e., $110 \rightarrow 000$; $111 \rightarrow 001$); L_3 : $010 \rightarrow 100$.

This 3-bit rule omits transitions $011 \rightarrow 101$, which separate pairs of 1-bits. Then more options remain to apply transition M_3 yielding an earlier reduction of the number of 1-bits. The simulation results in Figure 1 confirm a faster classification of the 3-bit rule by about 15% for cyclic, and about 25% for acyclic CA rules. The maximum

convergence time of the acyclic 3-bit rule is $n^2/4$ for even n and $(n^2-1)/4$ for odd n , and thus about half of the worst case $n(n-1)/2$ of the acyclic 2-bit rule for large n .

A study by N. Fatès [5] evaluates a cyclic 4-bit balanced update rule for the parity check of an asynchronous cellular automaton, which involves four neighboring cells in the decisions to invert the middle pair of those cells per update step:

Fatès' rule: $(x_k, x_{k+1}) \rightarrow (1 - x_k, 1 - x_{k+1})$, if and only if $x_{k-1} \neq x_k$ or $x_{k+1} \neq x_{k+2}$.

The CA rule is again applied in random and uniformly chosen positions $k = 1, \dots, n$ of the current configuration. Favourable convergence properties are proven for the 4-bit balanced rule towards two absorbing states $1 \dots 1$ for odd, and $0 \dots 0$ for even parity with a restriction to cases of an odd ring size n [5].

The 4-bit rule implies that the number of bit changes in the configuration, i.e. the number of positions k with $x_k \neq x_{k+1}$ is not increasing in each update step. Therefore, Fatès' rule implies a Markovian convergence process with a monotonously decreasing number of bit changes in subsequent configurations. This is similar to the decreasing trend in the number of 1-bits of the previous 2-bit rules MLR and ML.

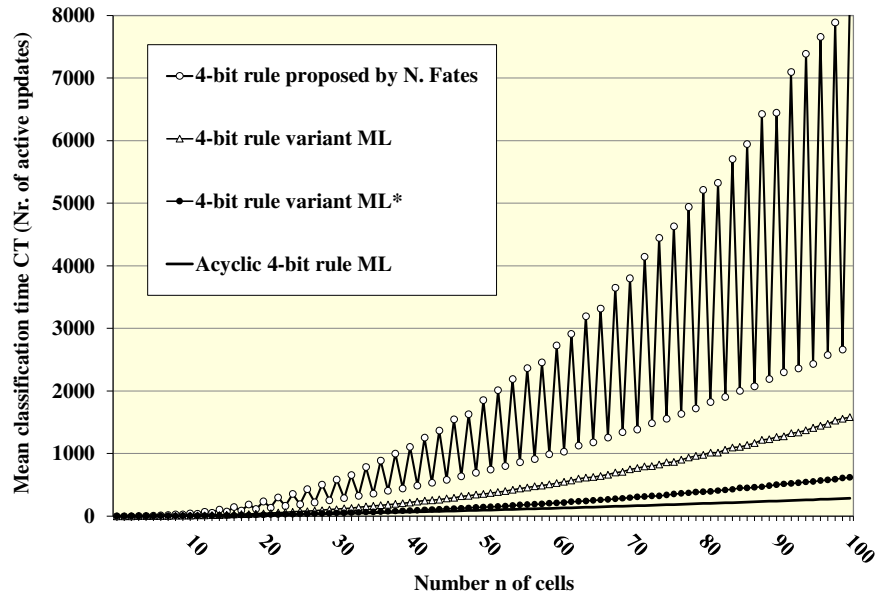


Figure 2: Mean time to reach a classifying state by 4-bit update rules

Our simulations of the classification time of Fatès' balanced rule [5] yield the results shown in Figure 2 when active transitions are counted. For odd ring sizes, Fatès' 4-bit rule is fast converging in contrast to the balanced 2-bit PMLR rule, but is about 1.5-fold slower than the 2-bit MLR rule, see Figure 1.

N. Fatès has proposed and analyzed the 4-bit parity rule for odd ring sizes. For even ring size n and configurations with even parity, this rule has two absorbing states $0 \dots 0$ and $1 \dots 1$. For even ring size n and odd parity, Fatès' rule approaches a class of $n^2/2$ configurations with bit changes $x_k \neq x_{k+1}$ in only two positions, i.e., configurations of

the type $1^{2m-1}0^{n-2m+1}$ for $m = 1, \dots, n/2$ and their cyclic shifts. Figure 2 indicates about 3-fold more active update steps for even n than for an odd ring size $n-1$ until a first entrance to the state $10 \dots 0$ as a representative of the state class with two bit changes.

We evaluate variants of Fatès' rule [5] in Figure 2, which invert only a subset of the four cases PMLR similar to the 2-bit rule variants evaluated in Figure 1. First, the 4-bit ML variant of Fatès' rule [5] converges to the state $0 \dots 0$ for even parity and approaches the class of states $10 \dots 0$, $\dots$, $0 \dots 01$ with a single 1-bit for odd parity, except for the state $1 \dots 1$, which persists as an isolated absorbing state. The results for the 4-bit ML rule in Figure 2 are similar to the 2-bit ML rule in Figure 1.

Moreover, we evaluate the acyclic ML variant of Fatès' balanced rule, which converges to the absorbing states $0 \dots 0$ for even and $10 \dots 0$ for odd parity, except for the isolated state $1 \dots 1$. This rule shares the essentially faster convergence properties of the acyclic 2-/3-bit rules as described in Section 3.2.

Finally, we consider a 4-bit update rule ML^* with a monotonous decrease in the number of bit changes as well as the number of 1-bits in each subsequent configuration, thus combining favourable properties of Fatès' rule and the 2-bit ML rule. Therefore, all 4-bit pattern with even parity are mapped to 0000 and all pattern with odd parity to 1000 , except for $0111 \rightarrow 0001$ and $0001 \rightarrow 0001$. Both exceptions avoid an increasing number of bit changes, that otherwise could occur in the entire configuration by an update to 1000 . This rule transforms states with two bit changes, i.e. with a single 1-region, in less than $n/2$ active updates to the absorbing state $0 \dots 0$ or to the final set of states with a single 1-bit, whereas the analysis of Fatès' rule confirms slower $O(n^2)$ convergence especially for this case [5]. However, configurations with several interspaced 1-bits still slow down the convergence of the update process. The 4-bit ML^* rule improves the convergence, see Figure 2, but remains in the same complexity class.

3.4 Active and passive update pattern

The previous classification time results take only active rule updates into account, which modify the configuration. However, except for the 2-bit PMLR rule with active updates on all pattern, the random choice of update positions often stumbles upon passive patterns, which are checked without a change of the configuration.

We evaluate the factor of the number of passive over active updates for the previous update rules in Figure 3. This factor is linear increasing with the ring size n for the considered cyclic rules in a range of $0.2n - 0.4n$. We observe that the fraction of active pattern is higher for random configurations than for configurations with a direct transition into the final states $0 \dots 0$ and $10 \dots 0$, because then the passive pattern $00 \dots$ is dominant throughout the latter configurations. The rule MLR has more active pattern than ML and therefore yields a higher fraction of active updates.

In general, the number of passive update checks until a next active update is geometrically distributed. The probability of an active pattern in the next step is given by the fraction of positions with active update pattern in the current configuration. Each active update is extended by an independent, geometrically distributed number of passive updates. A check of a passive pattern adds to the energy consumption, but active update steps are expected to consume more energy for performing modifications.

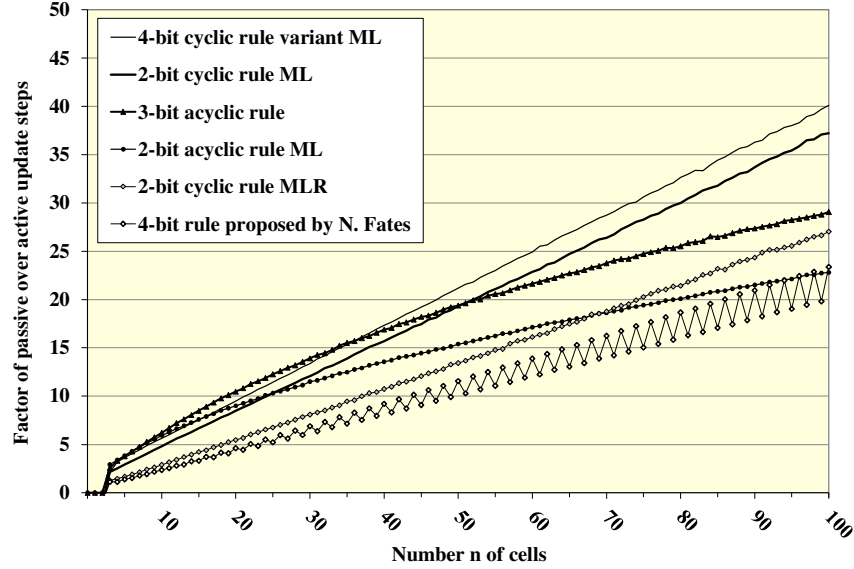


Figure 3: Factor of passive to active update steps required for the classification

3.5 Summary of the convergence properties of the rules

We summarize the convergence properties of the evaluated update rules for asynchronous CA in Table 3, which includes the set of final states and an estimate of the mean classification time for a large number n of cells.

Table 3: Final (set of) states and mean convergence time of CA update rules

Update Rules	Final state or class of states		Classification Time CT (*) until first entering $0 \dots 0$ or $10 \dots 0$
	for even parity	for odd parity	
Cyclic Updates			Mean CT
2-bit rule PMLR	-	-	(*) $\approx 0.525 \cdot 2^n$
4-bit rule by Fatès for odd n [5]	$0 \dots 0$	$1 \dots 1$	$\approx 0.333 \cdot n^{1.96}$
for even n	$0 \dots 0$ and $1 \dots 1$	$n^2/2$ states with 2 bit changes	(*) $\approx 0.9 \cdot n^{2.04}$
2-bit rule MLR	$0 \dots 0$	$100 \dots 0,$	(*) $\approx 0.3 \cdot n^{1.99}$
2-bit rule ML		$010 \dots 0,$	(*) $\approx 0.163 \cdot n^{2.02}$
3-bit rule M_3L_3		...	(*) $\approx 0.13 \cdot n^{2.04}$
4-bit rule ML*		$0 \dots 01$	(*) $\approx 0.04 \cdot n^{2.1}$
4-bit ML var. of [5]	$\rightarrow$ with isolated	state $1 \dots 1$	(*) $\approx 0.142 \cdot n^{2.03}$
Acyclic Updates			Mean CT Worst Case
2-bit rule ML	$0 \dots 0$	$10 \dots 0$	$\approx 0.375 \cdot n^{1.5}$ $n(n-1)/2$
3-bit rule M_3L_3			$\approx 0.21 \cdot n^{1.56}$ $n^2/4$

The acyclic rules and the 4-bit rule proposed by N. Fatès [5] for odd ring size n show strict convergence towards two absorbing states to distinguish even and odd parity. For the cyclic rule variants ML, the Markovian process of active updates is ending up in a set of states with a single 1-bit for odd parity, which are periodically passed through. Then one of those states, e.g., $10 \dots 0$ has to be detected and established as the final absorbing state by blocking further transitions from this state.

For comparing the convergence of the rules from a random start configuration to a final absorbing state, we approximate the mean classification time CT as active updates by a function $\alpha \cdot n^\beta$. The parameters α and β are determined to match simulation results $CT(n)$ on the boundaries $[n_{\min}, n_{\max}]$ of a range of ring sizes n :

$$CT(n) \approx \alpha \cdot n^\beta, \text{ where } \beta = \log(CT(n_{\max})/CT(n_{\min}))/\log(n_{\max}/n_{\min}), \alpha = CT(n_{\min})/n_{\min}^\beta.$$

Beyond the results in Figure 1 and Figure 2 for $n \leq 100$, we performed simulations up to $n_{\max} = 500$ for cyclic and $n_{\max} = 1000$ for acyclic rules, where $n_{\min} = 100$ is assumed in the approximations obtained in Table 3. Those approximations of the mean classification times are close to a quadratic complexity $O(n^2)$ for most of the cyclic rules, whereas the acyclic rules are close to $O(n^{1.5})$.

The results for cyclic rules are subject to considerable standard deviation σ up to $\sigma/\mu = 10\%$ of the mean value μ for cyclic rules for a ring size $n_{\max} = 500$. For the acyclic rules, the convergence times are essentially shorter, thus enabling for simulation results with $\sigma/\mu < 1\%$. Then the relative deviations of the approximation $CT(n) \approx \alpha \cdot n^\beta$ from the simulation results are below 2% on the range $100 \leq n \leq 1000$. However, we have no evidence whether this behavior continues for larger ring sizes $n > 1000$. The acyclic rules have a proven quadratic complexity $O(n^2)$ of the worst case convergence time with the explicit results summarized in the last column of Table 3.

4 Transient Analysis of the Classification Time

Next, we apply a transient analysis to the update processes. Let X_t , $t = 1, 2, \dots$ be the Markov chain of configurations after t steps of the update process with transition probabilities $p_{ij} = P(X_{t+1} = j | X_t = i)$, where configurations are represented as a number $X_t \in \{0, 1, \dots, 2^n - 1\}$ corresponding to the binary value of the cell sequence.

The state transitions of the Markov chain are predefined by a rule, which is applied per update step in a position of the current configuration with uniform probability $1/n$. We denote the successor state by an update in position m of a configuration i by a rule R as $S_R(i, m)$. Then we obtain the transition probabilities for a cyclic rule:

$$p_{ij} = \sum_{m=1}^n \delta_{j, S_R(i, m)} / n$$

where δ_{ij} is Kronecker' Delta: $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$. For acyclic rules, the summation limit n is reducing due to cells without updates. In most evaluations, we focus on the embedded Markov chain Y_t of active updates with transition probabilities:

$$q_{ij} = P(Y_{t+1} = j | Y_t = i) = \begin{cases} p_{ij}(1 - \delta_{ij}) / (1 - p_{ii}) & \text{if } p_{ii} < 1; \\ \delta_{ij} & \text{if } p_{ii} = 1. \end{cases}$$

Finally, we can iteratively calculate the state probabilities $P(X_{t+1} = j)$ after $t+1$ updates and $P(Y_{t+1} = j)$ after $t+1$ active updates:

$$P(X_{t+1} = j) = \sum_{i=0}^{2^n-1} P(X_t = i) p_{ij}; \quad P(Y_{t+1} = j) = \sum_{i=0}^{2^n-1} P(Y_t = i) q_{ij}.$$

We start the transient analysis from a uniform distribution $\forall j: P(X_0 = j) = 2^{-n}$ on all 2^n configurations, which was also the starting point of the previous simulative evaluations.

4.1 Transient analysis of the acyclic 2-bit ML update rule

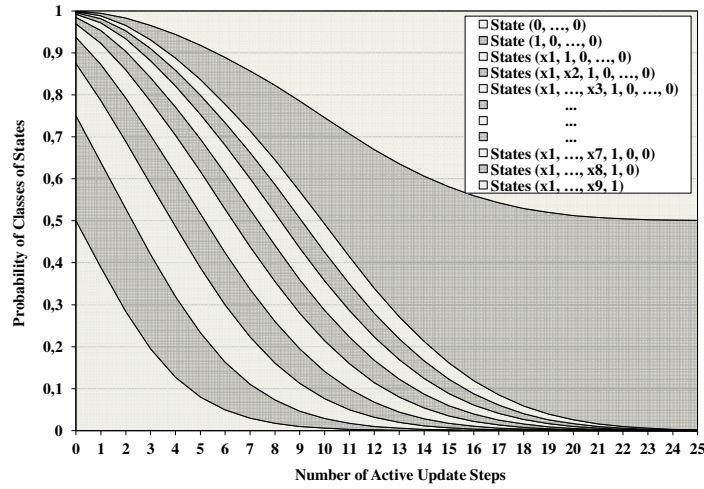


Figure 4: Transient analysis of probabilities shifting towards the absorbing states

Figure 4 evaluates results of the transient analysis for the acyclic 2-bit rule ML for $n = 10$. The probabilities of both absorbing states $0 \dots 0$ and $10 \dots 0$ for even and odd parity are shown, as well as the probabilities of 9 classes of states, whose binary representation ends with 1 or with $10 \dots 0$ for $m = 1, \dots, 8$ zeros. The class of states ending with $10 \dots 0$ comprises 2^{n-m-1} states, which initially aggregate a total probability of 2^{-m-1} . Figure 4 shows how those state probabilities are shifting towards 49.8% probability in each of both absorbing states within 25 active update steps. The acyclic 2-bit ML rule has a worst case convergence time of $n(n-1)/2$, but the maximum of 45 for $n = 10$ is seldomly encountered. The mean convergence time is 10.572 active updates.

In general, the transient analysis provides numerical exact results of the state probabilities and derived measures, such as the distribution, mean value, variance and quantiles of the convergence time, whereas simulation results are subject to a standard deviation. We focus on the mean values and leave other measures for future study.

However, the transient analysis requires storage for the probabilities of 2^n states with a computational complexity of $n \cdot 2^n$ per update step. Thus, the analysis becomes increasingly complex for large n , whereas simulations are scalable even for large ring sizes more or less independent of n .

4.2 Transient analysis of the 4-bit update rule proposed by N. Fatès

In a second case study, we apply the transient analysis to Fates' balanced update rule [5] with absorbing states $0 \dots 0$ for even parity and $1 \dots 1$ for odd parity, see Section 3.3. Figure 5 illustrates shifts of the probability distribution towards both absorbing states within the first 100 update steps for a ring of size 21.

Starting again from a uniform distribution, Fatès' rule leads to a decreasing number of bit changes in the configurations during the convergence process. Initially, over 40% of all possible configurations include more than 10 bit changes. Figure 5 reveals a fast reduction of the probability of states with >10 bit changes from 40% down to 1% already after 10 updates. Moreover, the probabilities of state classes with 2, 4, ..., 10 bit changes are shown. It is apparent that states with 2 bit changes are dominant after 50 - 100 active update steps with a slow convergence towards both attracting states.

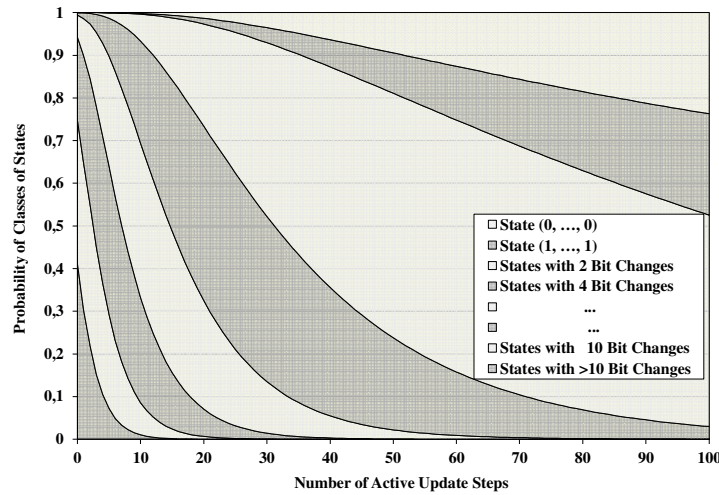


Figure 5: Transient analysis for active updates by Fatès' rule [5] for ring size 21

The fast convergence from configurations with many bit changes is stemming from a high percentage of pattern which reduce the number of bit changes. About 42% of the active update pattern in all positions of states with >10 bit changes lead to a configuration with less bit changes.

However, in the class of states with 2 bit changes only $2n$ out of $n(n-1)$ configurations have a transit to an absorbing state via a single out of three active update pattern. Therefore, the convergence from this prefinal class to the final absorbing states is much slower, as confirmed by the transient analysis results in Figure 5. A detailed analysis by N. Fates [5] represents the prefinal phase, which is spent in configurations with 2 bit changes, as a one-dimensional Markov chain. The mean classification time derived from the Markov analysis shows a quadratic increase $O(n^2)$ for active updates and even a cubic increase $O(n^3)$ including passive update checks.

The mean classification time in the example of Figure 5 requires 136.6 active updates and over 500 updates to reach the final absorbing states with 99% probability.

5 Conclusions and Future Work

We investigated the convergence properties of novel parity rules for local 2-, 3- and 4-bit neighborhood patterns of an asynchronous CA in a broader context with arbitrary final states and acyclic rule variants. Especially the acyclic rules can improve the mean classification time of a parity check already by 2-bit updates, with a worst case performance bound of $n(n - 1)/2$ active update steps. However, the asynchronous principle still involves a considerable number of passive rule checks without a state update. The transient analysis approach gives valuable insights into the shifts of the probability for pattern classes being entered during the update process, although with limited scalability in the ring size.

As topics for future work, the Markov chains of the update process can be analyzed in more depth and for additional performance measures. The search for optimized rules can be widened, e.g., for covering all options of 3- and 4-bit rules. The favourable performance of simple acyclic parity check rules is a promising basis for studying more complex computation tasks beyond a parity check. A closer look at the computation effort and energy consumption of active and passive updates is desirable as well as a comparison with synchronous CA update processes.

References

- [1] S. Achasova et al., *Parallel substitution algorithms, Theory and Applications*, World Scientific (1994)
- [2] H. Betel, P.P.B. de Oliveira, and P. Flocchini. Solving the parity problem in one-dimensional cellular automata, *Natural Computing* 12.3 (2013) 323-337
- [3] A. Nenca, B. Wolnik and J. Dybizbański, Advances in the parity problem for cellular automata, *Proc. AUTOMATA 2024, Exploratory Papers and Extended Abstracts*, Durham, UK (2024)
- [4] I. Donoso Leiva et al., Impact of (a)synchronism on ECA: towards a new classification (2025) arXiv:2505.10645v1
- [5] N.A. Fatès, Asynchronous cellular system that solves the parity problem. *Proc. 30th Workshop on Cellular Automata and Discrete Complex Systems (AUTOMATA)* Durham, UK (2024) 133-145
- [6] N.A. Fatès, A tutorial on elementary cellular automata with fully asynchronous updating: General properties and convergence dynamics, *Natural Computing* 19.1 (2020) 179-197
- [7] R. Hoffmann, *Algorithms with active cells modeled by cellular automata with write-access (CA-w), Automata, universality, computation: Tribute to Maurice Margenstern*, Cham, Springer (2014) 277-295
- [8] A. Nenca, B. Wolnik and B. De Baets, No six-cell neighborhood cellular automaton solves the parity problem. *Theoretical Computer Science*, 1021 (2024) 114923
- [9] R.S. Roy, V.K. Gautam and S. Das, A Note on skew-asynchronous cellular automata, *Proc. AUTOMATA 2024, LNCS 14782*, Durham, UK (2024) 146-158
- [10] S. Wolfram, Statistical mechanics of cellular automata, *Reviews of modern physics* 55.3 (1983) 601-644
- [11] B. Wolnik et al., Cellular automata can really solve the parity problem, arXiv:2501.08684v2 (2025) 1-23