Real Time Language Recognition on 2D Cellular Automata: Dealing with Non-Convex Neighborhoods

Martin Delacourt* and Victor Poupet**

LIP (UMR 5668 — CNRS, ENS Lyon, UCB Lyon, INRIA), ENS Lyon, 46 allée d'Italie, 69364 LYON cedex 07 FRANCE

Abstract. In this paper we study language recognition by two-dimensional cellular automata on different possible neighborhoods. Since it is known that all complete neighborhoods are linearly equivalent we focus on a natural sub-linear complexity class: the real time.
We show that any complete neighborhood is sufficient to recognize in real time any language that can be recognized in real-time by a cellular automaton working on the convex hull of V.

1 Introduction

Cellular automata are a widely studied computing model, very well suited for studying parallel computing (as opposed to most other models such as Turing machines or RAM machines). It is made of infinitely many elementary machines of finite memory (the cells) that evolve synchronously at discrete times according to the states of their neighbors. All cells have the same transition rule and can only see their neighbors. Because of the parallel behavior, it is easy to consider cellular automata in any dimension $d \in \mathbb{N}$ (the cells are arranged on $\mathbb{Z}^d$). It is known that cellular automata are Turing universal [1, 8].

The neighborhood of a cellular automaton (the set of cells whose states a given cell can see before changing its own) defines the possible communication between all the cells, and therefore the "geography" of the machine: the neighborhood of a cell is the set of cells from which it can get information in one time step, the neighborhood of the neighborhood is the set from which it can receive information in two time steps, and so on. In that way, the neighborhood defines the shortest paths to exchange information from one point to the other. As such, it can have a great impact on the possible computations that are held on an automaton.

An important result concerning computations on different neighborhoods is due to S. Cole [2] and states that two neighborhoods are either linearly equivalent (any computation that can be done in time T on one can be done in time $k \cdot T$ on the other for some constant k) or that there exists a cell $c \in \mathbb{Z}^d$ such that

* martin.delacourt@ens-lyon.fr
** victor.poupet@ens-lyon.fr

information can go from c to the origin in one of the neighborhoods but not in the other. If we consider neighborhoods that allow communications between any two cells (which are the most interesting because they can perform all possible computations), we will want to consider sub-linear complexity classes in order to distinguish them. The *real time* is especially well suited for this study: it corresponds to the shortest possible time so that any letter of the input word could have an impact on the acceptance of the word (we will only deal with language recognition here). This real time depends on the chosen neighborhood.

In one dimension (when the cells are arranged on $\mathbb{Z}$) most studies were done on the *standard* neighborhood $\{-1, 0, 1\}$ (each cell can see its own state and that of its left and right neighbor) and the *one-way* neighborhood $\{0, 1\}$ (cells can only see their own state and their right neighbor's). It has been shown that these two neighborhoods are different [4, 6, 11](mainly because information can only go in one direction on the *one-way* neighborhood) and many algorithmic results are known [3, 5, 9, 10]. If we only consider neighborhoods that are "complete enough" to perform language recognition (all letters of the input word can affect the outcome of the computation), we have shown in 2005 a stronger version of Cole's equivalence: all neighborhoods are real-time equivalent to either the one-way or standard neighborhood [7]. This was done by showing that it was possible to recognize the same languages in real time on non-convex neighborhoods (neighborhoods that had "holes", for example when a cell c can see $(c + 2)$ but not $(c + 1)$) than on convex ones.

In two dimensions, the situation is more complicated. Even by only considering complete neighborhoods it is known that the two more studied neighborhoods (the von Neumann and the Moore neighborhoods) are not real time equivalent [12].

In this article we will generalize the work that we had previously done in one-dimension and show that any language that is recognized in real time by a cellular automaton working on the convex hull of a complete neighborhood V can be recognized in real time by a cellular automaton working on V.

To alleviate the notations, we will only consider two-dimensional cellular automata in this article. All results can however easily be generalized to higher dimensions. The only result that might seem complicated to generalize would be Theorem 1, but it can be stated and proved similarly, by considering periodic volumes, surfaces, *etc.* and finite sets in the vincinity of the vertices. Theorem 1 is itself a two-dimensional generalization of Theorem 2.1 from [7].

2 Language Recognition by Cellular Automata

2.1 Cellular Automata

Definition 1. *A* two-dimensional cellular automaton *(2DCA) is a triple* $\mathcal{A} = (\mathcal{Q}, V, f)$ *where*

- $\mathcal{Q}$ *is a finite set called* set of states *containing a special* quiescent *state #;*
- $V = \{v_1, \ldots, v_{|V|}\} \subseteq \mathbb{Z}^2$ *is a finite set called* neighborhood *that contains 0.*

– $f : \mathcal{Q}^{|V|} \to \mathcal{Q}$ *is the* transition function. *We have* $f(\#, \ldots, \#) = \#$.

For a given automaton $\mathcal{A}$, we call *configuration* of $\mathcal{A}$ any function $\mathfrak{C}$ from $\mathbb{Z}^2$ into $\mathcal{Q}$. The set of all configurations is therefore $\mathcal{Q}^{\mathbb{Z}^2}$. From the local function f we can define a global function F

$$\begin{array}{rl} F : \mathcal{Q}^{\mathbb{Z}^2} \to \mathcal{Q}^{\mathbb{Z}^2} & \\ \mathfrak{C} \mapsto \mathfrak{C}' & \mid \quad \forall x \in \mathbb{Z}^2, \mathfrak{C}'(x) = f(\mathfrak{C}(x + v_1), \ldots, \mathfrak{C}(x + v_{|V|})) \end{array}$$

Elements of $\mathbb{Z}^2$ are called *cells*. Given a configuration $\mathfrak{C}$, we'll say that a cell c is in state q if $\mathfrak{C}(c) = q$.

If at time $t \in \mathbb{N}$ the 2DCA is in a configuration $\mathfrak{C}$, we'll consider that at time $(t + 1)$ it is in the configuration $F(\mathfrak{C})$. We can therefore define the *evolution* of a 2DCA from a configuration. This evolution is completely determined by $\mathfrak{C}$.

2.2 Two-Dimensional Language Recognition

Definition 2. *Given a finite alphabet Σ and two integers n_1 and n_2, we define the set of two-dimensional words of size (n_1, n_2) over the alphabet Σ as:*

$$\Sigma^{(n_1,n_2)} = \Sigma^{[\![0,n_1-1]\!]\times[\![0,n_2-1]\!]}$$

The set of all two-dimensional words over Σ is defined as:

$$\Sigma^{**} = \bigcup_{n_1 \in \mathbb{N}, n_2 \in \mathbb{N}} \Sigma^{(n_1,n_2)}$$

Two-dimensional words over Σ can be seen as rectangular grids of size $n_1 \times n_2$ containing letters of Σ.

Definition 3. *A language over an alphabet Σ is a subset of Σ^{**}.*

Definition 4. *We consider a 2DCA $\mathcal{A} = (\mathcal{Q}, V, f)$ and a set $\mathcal{Q}_{acc} \subseteq \mathcal{Q}$ of* accepting states. *Let $w \in \Sigma^{(n_1,n_2)}$ be a word over a finite alphabet $\Sigma \subseteq \mathcal{Q}$. We define the configuration $\mathfrak{C}_w$ as follows.*

$$\begin{array}{rl} \mathfrak{C}_w : & \mathbb{Z}^2 \to \mathcal{Q} \\ \begin{cases} (x, y) & \mapsto w(x, y) \\ (x, y) & \mapsto \# \end{cases} & \begin{array}{ll} \text{if } (x, y) \in [\![0, n_1 - 1]\!] \times [\![0, n_2 - 1]\!] \\ \text{otherwise} \end{array} \end{array}$$

We'll say that the 2DCA $\mathcal{A}$ recognizes *the word w with accepting states $\mathcal{Q}_{acc}$ in time t_w if, starting from the configuration $\mathfrak{C}_w$ at time 0, the cell 0 is in a state in $\mathcal{Q}_{acc}$ at time t_w.*

Definition 5. *Let $\mathcal{A} = (\mathcal{Q}, V, f)$ be a 2DCA and $L \subseteq \Sigma^{**}$ a language on the alphabet $\Sigma \subseteq \mathcal{Q}$. For a given function $T : \mathbb{N}^2 \to \mathbb{N}$, we'll say that the language L is* recognized by $\mathcal{A}$ in time T *if there exists a set $\mathcal{Q}_{acc} \subseteq \mathcal{Q}$ such that, for all words w of size (n_1, n_2) in Σ^{**}, the 2DCA $\mathcal{A}$ recognizes w with accepting states $\mathcal{Q}_{acc}$ in time $T(n_1, n_2)$ if and only if $w \in L$.*

3 Iterated Neighborhoods

Definition 6. *Given two neighborhoods $V_1, V_2 \subseteq \mathbb{Z}^2$, we define*

$$V_1 + V_2 = \{v_1 + v_2 \mid v_1 \in V_1 \text{ and } v_2 \in V_2\}$$

Given a neighborhood V, we define its iterations as $V^0 = \{0\}$ and for all $k \in \mathbb{N}$, $V^{k+1} = V^k + V$ and its multiples as $kV = \{k \cdot v \mid v \in V\}$.

Definition 7. *A neighborhood $V \in \mathbb{Z}^2$ is said to be* complete *if $\bigcup_{k\in\mathbb{N}} V^k = \mathbb{Z}^2$.*

Definition 8. *The* continuous convex hull *of a neighborhood V, denoted $\mathrm{CCH}(V)$, is the smallest convex polygon (in $\mathbb{R}^2$) that contains V. The (discrete)* convex hull *of V, denoted $\mathrm{CH}(V)$, is the set of all points of $\mathbb{Z}^2$ that are in the continuous convex hull of V.*

Definition 9. *For a given neighborhood V, the* vertices *of the polygon $\mathrm{CCH}(V)$ are all elements of V (and therefore elements of $\mathbb{Z}^2$). We will call them the vertices of V.*

When considering the set $\{s_1, \ldots, s_p\}$ of vertices of a neighborhood, we will always order them as they appear when going clockwise around $\mathrm{CCH}(V)$. We will also consider the indexes modulo p (the number of vertices), meaning that $s_0 = s_p$ and $s_{p+1} = s_1$.

3.1 General Form of Iterated Complete Neighborhoods

In this whole subsection V is a complete neighborhood. We will study the shape of the successive iterations of V. First of all, we define the integer $t_c = \min\{t \in \mathbb{N} \mid \mathrm{CH}(V)^2 \subseteq V^{t_c+2}\}$.

We know that t_c is correctly defined because V is complete so there exists an integer t such that $\mathrm{CH}(V)^2 \subseteq V^t$. We have the following proposition:

Proposition 1. *For all integers $t \geq 2$,*

$$\mathrm{CH}(V)^t \subseteq V^{t_c+t} \subseteq \mathrm{CH}(V)^{t_c+t}$$

Proof. The rightmost inclusion is immediate because $V \subseteq \mathrm{CH}(V)$. The other inclusion can be shown by induction using the fact that $V + \mathrm{CH}(V)^2 = \mathrm{CH}(V)^3$.

We have the following theorem:

Theorem 1. *For any two-dimensional complete neighborhood V, if we denote by $(s_1, s_2, \ldots, s_p)$ its vertices, there exists an integer t_s such that:*

- *for all $i \in [\![1, p]\!]$, there is a set $A_i \subseteq V^{t_s+t_c} \setminus V^{t_s}$,*
- *for all integer $i \in [\![1, p]\!]$, there is a set B_i included in the trapezoid of sides $h_i = (s_{i+1} - s_i)$, $t_c \cdot s_{i+1}$, $-t_c \cdot h_i$ and $-t_c \cdot s_i$.*

– *for any integer $t \in \mathbb{N}$, the neighborhood $V^{t_c+t_s+t}$ is exactly the union of the following sets:*
  - $\mathrm{CH}(V)^{t_s+t}$,
  - $(A_i + t \cdot s_i)$ *for all* $i \in [\![1, p]\!]$,
  - *copies of B_i arranged regularly (translation of vector h_i) on the outer strip of the cone (s_i, s_{i+1}) to cover the area that isn't covered by the A_i.*

The general form of $V^{t_c+t_s+t}$ (as described by Theorem 1) is illustrated by Figure 1 for two different values of t.

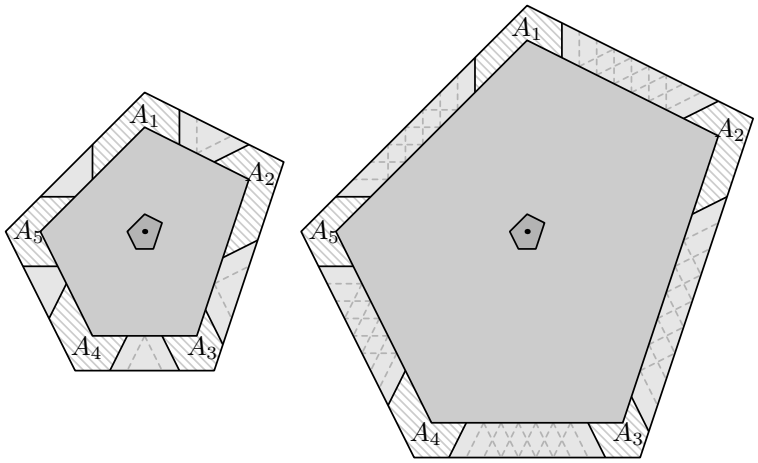

Fig. 1. General form of $V^{t_c+t_s+t}$ (the fillings of the dashed trapezoids are all identical on a given strip).

Even though it is hard to state clearly, Theorem 1 is very important because it shows that no matter how irregular V is, it becomes "regular" after a certain number of iterations.

4 Main Theorem

This whole section will be dedicated to the proof of the following theorem

Theorem 2. *Given a complete neighborhood V in d dimensions ($d \in \mathbb{N}$), any language that can be recognized in real time by a 2DCA working on the convex hull of V can be recognized in real time by a 2DCA working on V.*

To prove this theorem, we'll consider a complete neighborhood V and language L recognized in real time by a 2DCA $\mathcal{A}$ working on $\mathrm{CH}(V)$. We will then describe the behavior of a 2DCA $\mathcal{A}'$ working on the neighborhood V that recognizes L in real time. We define t_c as previously.

4.1 General Behavior of $\mathcal{A}'$

To describe the behavior of $\mathcal{A}'$, we will consider a two-dimensional word w and describe the evolution of $\mathcal{A}'$ on this input. Since the evolution of $\mathcal{A}'$ will mimick that of $\mathcal{A}$, it will be convenient to denote by $\langle c \rangle_t$ the state in which the cell c is at time t in the evolution of $\mathcal{A}$ starting from the initial configuration corresponding to the word w (for instance $\langle 0 \rangle_0$ is the lowest and leftmost letter of w).

For some large enough integer t_0 depending on V (we'll explain later how to choose t_0) the automaton $\mathcal{A}'$ will spend the first t_0 steps gathering all possible information on each cell.

After t_0 generations, any cell c knows therefore all states $\{\langle c + x \rangle_0 \mid x \in V^{t_0}\}$. If V^{t_0} is different from $\mathrm{CH}(V)^{t_0}$ there are some states in $\mathrm{CH}(V)^{t_0}(c)$ that c doesn't know. All cells will however assume that the states that they don't know in their neighborhood $\mathrm{CH}(V)^{t_0}$ are #. Obviously, many of these assumptions are false at time t_0, but for cells close enough to the borders of the input word some of these assumptions are true.

The cells of $\mathcal{A}'$ will now apply the transition rule of $\mathcal{A}$ to all the states they hold (including the ones they *assume*). Hence, at time $(t_0 + t)$ each cell c of $\mathcal{A}'$ holds a set of states that it assumes to be the states $\{\langle c + x \rangle_t \mid x \in \mathrm{CH}(V)^{t_0}\}$.

4.2 Propagation of correct assumptions

As previously, we denote by $\{s_1, \ldots, s_p\}$ the set of all vertices of V (ordered clockwise). For all $i \in [\![1, p]\!]$, we separate the cone (s_i, s_{i+1}) of the neighborhood $\mathrm{CH}(V)^{t_0}$ in four parts:

- the inside triangle C_i of sides $(t_0 - t_c)s_i$ and $(t_0 - t_c)s_{i+1}$ that we know is totally included in V^{t_0};
- a trapezoidal area T_i included in the remaining strip, whose parallel sides lay on the inner and outter borders of the strip and whose two other sides are parallel to the segments $[s_{i-1}, s_i]$ and $[s_{i+1}, s_{i+2}]$ of the convex hull of V;
- the two parts S_i^d and S_{i+1}^g that are left on each side of the central trapezoidal area.

We also define $S_i = S_i^d \cup S_i^g$.

We choose t_0 large enough so that V^{t_0} is of the "stabilized form" described by Theorem 1 (meaning that $t_0 \geq t_c + t_s$) and also that for all i the trapezoid T_i doesn't extend beyond the central periodic area of the outter strip of the cone (s_i, s_{i+1}) on V^{t_0} (when t grows, the central periodic area becomes arbitrarily large so there is a time t_0 such that we can choose T_i entirely inside of it).

Figure 2 illustrates the general form of such a splitting of $\mathrm{CH}(V)^{t_0}$.

If we consider a cell c of $\mathcal{A}'$ at time $(t_0 + t)$, it knows correctly all states $\{\langle c + x \rangle_t \mid x \in C_i\}$ for all i but not necessarily all states in the other regions.

For all i, we will say that a cell is (s_i, s_{i+1})-*correct* if all the assumptions it makes in the area $(c + T_i)$ are correct. We will say that it is s_i-*correct* if it is (s_{i-1}, s_i)-correct, (s_i, s_{i+1})-correct and that all the assumptions it makes in the area $(c + S_i)$ are also correct. Figure 3 illustrates these definitions.

We can now prove the two following lemmas:

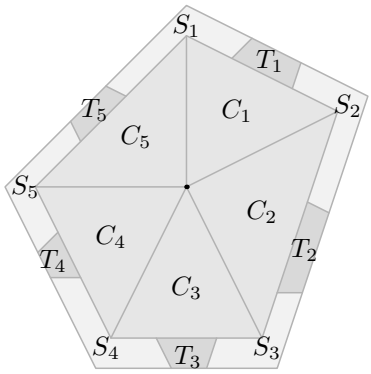

Fig. 2. Splitting of $\mathrm{CH}(V)^{t_0}$.

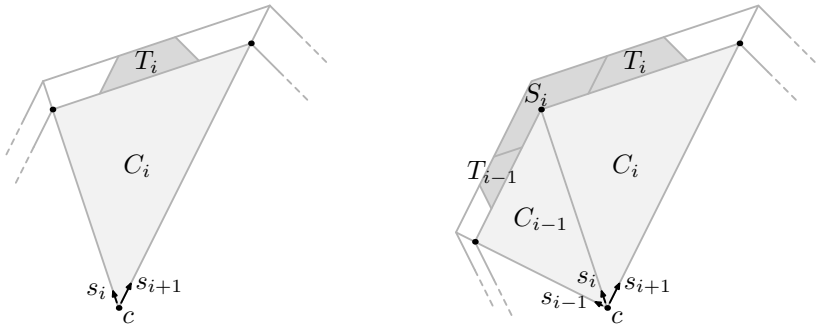

Fig. 3. Correct hypothesis of a (s_i, s_{i+1})-correct (left) and a s_i-correct (right) cell.

Lemma 1. *If at time (t_0+t) the cells $(c+s_i)$ and $(c+s_{i+1})$ are both (s_i, s_{i+1})-correct then at time (t_0+t+1) the cell c is (s_i, s_{i+1})-correct.*

Lemma 2. *If at time (t_0+t) the cell $(c+s_i)$ is s_i-correct and that both cells $(c+s_{i-1})$ and $(c+s_{i+1})$ are (s_{i-1}, s_i)-correct and (s_i, s_{i+1})-correct respectively then at time (t_0+t+1) the cell c is s_i-correct.*

Figures 4 and 5 illustrate the proofs of these two lemmas. In both cases we have represented on the left side the area on which the cell must have correct information to be correct at the next step and on the right side the areas on which it can see correct information according to the hypothesis of the lemma.

We see that in both cases the cell has enough information at time (t_0+t) to compute correct states at time (t_0+t+1) (the slope of the central trapezoid has been chosen so that everything works correctly, and we use the fact that $\mathrm{CH}(V)^{t_0}$ is convex). We also use the fact that if there is a conflict between the information held by a cell and its neighbors, the priority is given to the information held by the neighbor that is the closest to the disagreeing point.

We know that at time t_0 all cells of $\mathcal{A}'$ that are "close enough" to the border of the word w are correct in the direction pointing outside of the word. Lemmas 1

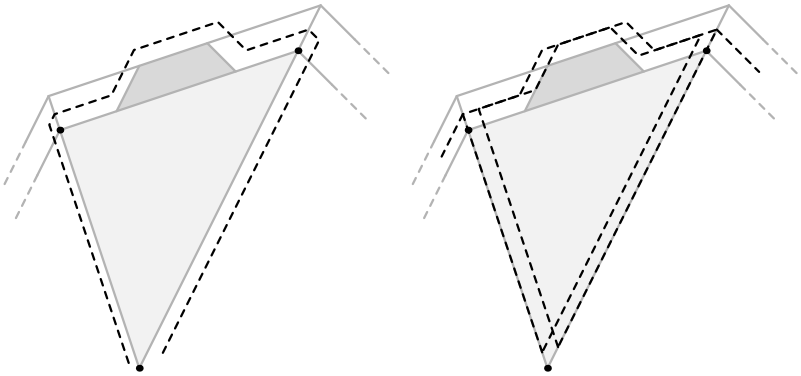

Fig. 4. States that the cell c must know to be (s_i, s_{i+1})-correct at the next time (left) and the correct information held by its neighbors (right).

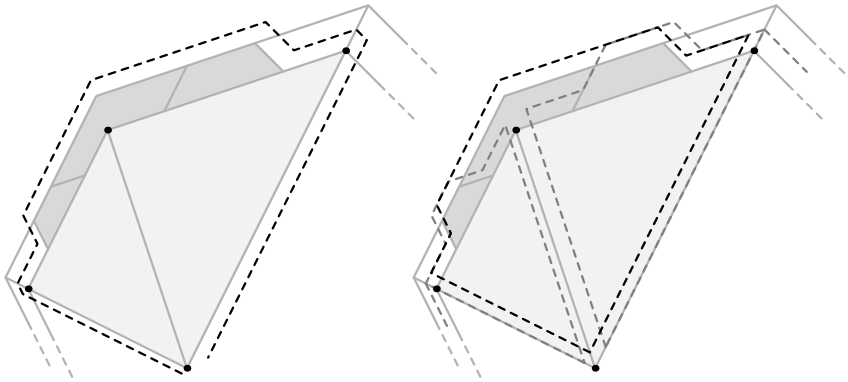

Fig. 5. States that the cell c must know to be s_i-correct at the next time (left) and the correct information held by its neighbors (right).

and 2 show that the the correctness of thes cells "propagates" to their neighbors towards the origin along the vectors s_i until eventually at some time $(t_0 + t_f)$ the origin is correct in all possible directions. At this time, the origin knows correctly all the states in $\{\langle x \rangle_{t_f} \mid x \in \mathrm{CH}(V)^{t_0}\}$ and can hence anticipate the simulation of $\mathcal{A}$ of t_0 steps. At time $(t_0 + t_f)$ the origin is therefore capable of knowing the state $\langle 0 \rangle_{t_0+t_f}$ in which the origin of $\mathcal{A}$ would be at time $(t_0 + t_f)$.

The 2DCA $\mathcal{A}'$ can therefore compensate for the initial t_0 steps that were "lost" at the beginning. Now we have to show that $(t_0 + t_f)$ is exactly the real time corresponding to the input word w.

4.3 The Real Time

Given a word w in Σ^{**}, we denote by M the set of cells on which the word spans when "written" on the initial configuration of $\mathcal{A}$ and $\mathcal{A}'$. In other terms, if w is of size (n, m), we have $M = [\![0, n-1]\!] \times [\![0, m-1]\!]$.

By definition of the real time, we have $\mathrm{TR}_V(n, m) = \min\{t \in \mathbb{N} \mid M \subseteq V^t\}$. To alleviate the notations in this section, we will define $t_r = \mathrm{TR}_V(n, m)$.

We want to show that at time t_r the origin of $\mathcal{A}'$ is correct in all possible directions. We have to consider both cases of angles (s_i-correctness) and cones ((s_i, s_{i+1})-correctness).

Lemma 3. *If $t_r \geq t_0$ then for any vertex s_i of V the cell $(t_r - t_0)s_i$ is s_i-correct at time t_0.*

Proof. According to the definition of real time, we have $M \subseteq V^{t_r}$. If $t_r \geq t_0$, the neighborhood V^{t_r} is of the "stabilized" form described by Theorem 1. Moreover, since V^{t_0} is also "stabilized", we know that the area corresponding to the vertex s_i in both neighborhoods V^{t_0} and V^{t_r} is identical (see Figure 6).

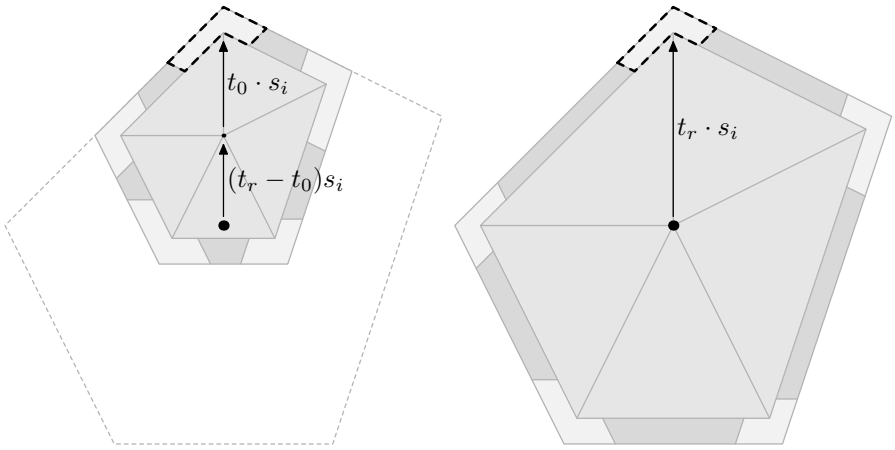

Fig. 6. The sets $V^{t_0} + (t_r - t_0)s_i$ (left) and V^{t_r} (right) coincide on the black dashed area.

Since M is included in V^{t_r}, there is no point of M in the black dashed area that isn't in V^{t_r}. By Theorem 1 we know that all points in that area are also in $V^{t_0} + (t_r - t_0)s_i$. Thus the cell $(t_r - t_0)s_i$ makes only correct assumptions in that area at time t_0 when it considers that all the states it doesn't see are #.

Since we have seen that (s_i, s_{i+1})-correctness propagates from $(c + s_i)$ and $(c + s_{i+1})$ to c and that the only cell we are really interested in is the origin, it is sufficient to work on the cells of the form $(a \cdot s_i + b \cdot s_{i+1})$ for $a, b \in \mathbb{N}$.

Lemma 4. *If $t_r \geq t_0$ then for any $i \in [\![1, p]\!]$, all cells of the form $(a \cdot s_i + b \cdot s_{i+1})$ with $a, b \in \mathbb{N}$ and $a + b = t_r - t_0$ are (s_i, s_{i+1})-correct at time t_0 (these cells are all on the segment $[(t_r - t_0)s_i, (t_r - t_0)s_{i+1}]$ as shown by Figure 7).*

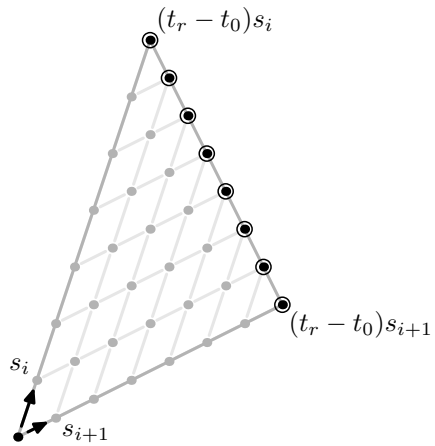

Fig. 7. The cells $(a \cdot s_i + b \cdot s_{i+1})$ where $a, b \in \mathbb{N}$ and $a + b = t_r - t_0$ (represented by black circled dots).

Proof. As previously, if $t_r \geq t_0$ the neighborhood V^{t_r} is of the form described by Theorem 1, as is V^{t_0}. This means that the central trapezoids in the strip of width t_c on both neighborhoods are superpositions of identical trapezoidal fillings periodically translated by a vector $(s_{i+1} - s_i)$.

This means that for any cell $c = (a \cdot s_i + b \cdot s_{i+1})$ with $a, b \in \mathbb{N}$ and $a + b = t_r - t_0$ the filling of $V^{t_0}(c)$ on the trapezoidal area corresponding to (s_i, s_{i+1}) coincides with the neighborhood V^{t_r} (see Figure 8)

Since M in included in V^{t_r}, all letters of the word that are in the area that the cell c has to know in order to be (s_i, s_{i+1})-correct are visible to the cell. It only makes true assumptions in this area when is assumes that all states it cannot see are #. All cells $(a \cdot s_i + b \cdot s_{i+1})$ for $a, b \in \mathbb{N}$ and $a + b = t_r - t_0$ are therefore (s_i, s_{i+1})-correct at time t_0.

4.4 End of the Proof

Using Lemmas 1, 2, 3 and 4 we can now show by induction the following results:

Lemma 5. *If $t_r \geq t_0$, for any vertex s_i of V and any $t \in \mathbb{N}$, the cell $(t_r - t_0 - t)s_i$ is s_i-correct at time $(t_0 + t)$.*

Lemma 6. *If $t_r \geq t_0$, for any vertex s_i of V and any $t \in \mathbb{N}$, all cells $(a \cdot s_i + b \cdot s_{i+1})$ where $a, b \in \mathbb{N}$ and $a + b = t_r - t_0 - t$ are (s_i, s_{i+1})-correct at time $(t_0 + t)$.*

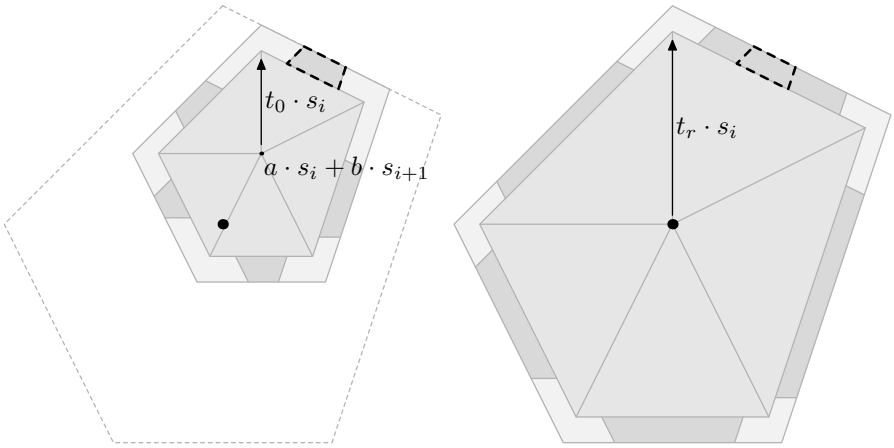

Fig. 8. The sets $V^{t_0}(c)$ (left) and V^{t_r} (right) coincide on the black dashed area.

We finally conclude by saying that if $t_r \geq t_0$, the origin is correct in all possible directions at time $t_0 + (t_r - t_0) = t_r$. Hence, on an input w of size (n, m), the 2DCA $\mathcal{A}'$ working on the neighborhood V can compute at time $\mathrm{TR}_V(n, m)$ the state in which the origin of $\mathcal{A}$ would be at the same time from the same input.

Since there are only a finite number of words $w \in \Sigma^{**}$ such that the real time corresponding to these words is smaller than t_0, we can modify the automaton so that it recognizes also correctly in real time these words that are too small. This ends the proof of Theorem 2.

5 Conclusion

Understanding the impact of the choice of the neighborhood in language recognition on cellular automata is a key point to understanding communication in parallel computations. What we have shown here is that, although it might seem important, the neighborhood needs not be convex since the same computation can be done on non-convex neighborhoods than on convex ones, and there is no other loss of time than the obvious one due to the fact that the neighborhood is smaller. Not only it shows that convexity is never fully used by any parallel algorithm, but it also simplifies our study considerably since we will now be able to consider only convex neighborhoods when proving algorithmic results (speed-up theorems for example).

An interesting thing to notice is that we don't have the converse of Theorem 2. Although it might seem unlikely, there might exist languages that can be recognized in real time on a certain neighborhood V but not on its convex hull. Of course, any computation that can be done on V can be done in the same time on $\mathrm{CH}(V)$, but since the real time on $\mathrm{CH}(V)$ can be smaller than the one on

V, it is not easy to show that we can go faster on CH(V). This comes from the fact that we are unable to prove constant time acceleration theorems on some convex neighborhoods (such as the von Neumann one).

Also it might be interesting to determine precisely which neighborhoods are real time equivalent but very little is known in that direction. If we consider complete neighborhoods, we know that there is a language that is recognized in real time on the von Neumann neighborhood but not on the Moore neighborhood [12], but it's the only example (and the converse is still unknown).

References

1. Albert, J., Čulik II, K.: A simple universal cellular automaton and its one-way and totalistic version. Complex Systems **1** (1987) 1–16
2. Cole, S.N.: Real-time computation by n-dimensional iterative arrays of finite-state machines. IEEE Transactions on Computers **C-18** (1969) 349–365
3. Čulik, K., Hurd, L.P., Yu, S.: Computation theoretic aspects of cellular automata. Phys. D **45** (1990) 357–378
4. Dyer, C.R.: One-way bounded cellular automata. Information and Control **44** (1980) 261–281
5. Ibarra, O., Jiang, I.: Relating the power of cellular arrays to their closure properties. Theoretical Computer Science **57** (1988) 225–238
6. Ibarra, O.: 6. In: Cellular Automata: a Parallel Model. Mathematics and its applications edn. Kluwer, Dordrecht (1999) 181–197
7. Poupet, V.: Cellular automata: Real-time equivalence between one-dimensional neighborhoods. In Diekert, V., Durand, B., eds.: STACS. Volume 3404 of Lecture Notes in Computer Science., Springer (2005) 133–144
8. Smith III, A.R.: Simple computation-universal cellular spaces. J. ACM **18** (1971) 339–353
9. Smith III, A.R.: Real-time language recognition by one-dimensional cellular automata. Journal of the Assoc. Comput. Mach. **6** (1972) 233–253
10. Terrier, V.: Language recognizable in real time by cellular automata. Complex Systems **8** (1994) 325–336
11. Terrier, V.: Language not recognizable in real time by one-way cellular automata. Theoretical Computer Science **156** (1996) 281–287
12. Terrier, V.: Two-dimensional cellular automata recognizer. Theor. Comput. Sci. **218** (1999) 325–346