Leader Election by d Dimensional Cellular Automata

Codrin Nichitiu, Eric Remila

Laboratoire de l'Informatique du Parallélisme

École Normale Supérieure de Lyon
Unité Mixte de Recherche CNRS-INRIA-ENS LYON n° 8512

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

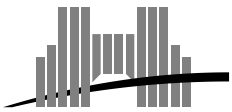
SPI

Leader Election by d Dimensional Cellular Automata

Codrin Nichitiu
Eric Rémila

January 1999

Research Report N° 1999-01



École Normale Supérieure de Lyon
46 Allée d'Italie, 69364 Lyon Cedex 07, France
Téléphone : +33(0)4.72.72.80.37
Télécopieur : +33(0)4.72.72.80.80
Adresse électronique : lip@ens-lyon.fr

INRIA

Leader Election by d Dimensional Cellular Automata

Codrin Nichitiu
Eric Rémila

January 1999

Abstract

We present a cellular algorithm in $O(w^2)$ for the leader election problem on a finite connected subset F of $\mathbb{Z}^d$ of diameter w, for any fixed d. The problem consists in finding an algorithm such that when setting the elements of F to a special state, and all the others to a state #, the cellular automaton iterates a finite number of steps and eventually sets only one precise element of F to a special state called leader state. We describe the algorithm in detail, prove it and its complexity, and discuss the possible extensions on more general Cayley graphs.

Keywords: graph automata, leader election, finite state automata, d-dimensional integer grid, cellular automata

Résumé

Nous présentons un algorithme cellulaire en $O(w^2)$ pour le problème d'élection d'un général sur un sous-ensemble fini connexe $F \subset \mathbb{Z}^d$ de diamètre w, pour n'importe quel d fixé. Le problème consiste à trouver un algorithme tel que lorsqu'on met les éléments de F dans un état spécial et tous les autres dans un état #, l'automate cellulaire itère un nombre fini de pas, et met un seul élément de F dans un état spécial nommé état général. Nous décrivons l'algorithme en détail, démontrons sa correction et sa complexité et discutons des extensions possibles à des graphes de Cayley plus généraux.

Mots-clés: graphes d'automates, éléction d'un général, automate fini, grille entière de dimension d, automates cellulaires

Leader Election by d Dimensional Cellular Automata

Codrin Nichitiu* and Eric Rémila **

Ecole Normale Supérieure de Lyon

Abstract We present a cellular algorithm in $O(w^2)$ for the leader election problem on a finite connected subset F of $\mathbb{Z}^d$ of diameter w, for any fixed d. The problem consists in finding an algorithm such that when setting the elements of F to a special state, and all the others to a state #, the cellular automaton iterates a finite number of steps and eventually sets only one precise element of F to a special state called leader state, and all the others to a different state. We describe the algorithm in detail, prove it and its complexity, and discuss the possible extensions on more general Cayley graphs.

1 Introduction

The leader election problem, or queen bee problem, has been introduced in 1971, in dimension two by [3]. This problem can be informally seen as a reverse firing squad synchronization problem, because here we have all the cells in the same state at initialization, and we want a unique cell to be in a special state at the end of the computation (assimilated to "general" state), while all the others have to be in another state, sort of "soldier" state.

In dimension two, several authors have proposed different solutions, and among the last and the best ones we have to cite [1] and [4] for two dimensions. Their idea is to follow the perimeter of the figure, using a compass. This way the time complexity is linear in the perimeter, thus in the number of cells.

In arbitrary dimensions, the problem is much more difficult, and the best solution we knew of before this paper was the one of [2]. He proposed the use of search algorithms in finite labyrinths. However, the time complexity is $O(n^5)$, where n is the number of cells.

We present here another solution, using a new approach. We make use of a spanning lattice, similar to the spanning tree introduced by [6]. This lattice is constructed through signals starting from points susceptible to be elected as global leaders. We actually have several lattices growing at the same time, so when they hit each other they compete for survival, only one wins, and at the end, the best makes its leader the global leader.

2 Definitions

We consider cellular automata over $\mathbb{Z}^d$, where $d \in \mathbb{N}$, $d > 1$, ordered by the lexicographic order $<_L$: for two elements $c_1 = (x_1, x_2, \ldots, x_d)$ and $c_2 = (y_1, y_2, \ldots, y_i)$, we say that

$$c_1 <_L c_2 \iff \exists i \text{ with } 1 \leq i \leq d \text{ with } \forall j, 1 \leq j < i, x_j = y_j \text{ and } x_i < y_i.$$

2.1 General definitions

Definition 1. *Let Σ be a finite set called alphabet, Q be a finite set called set of states, and δ be a function $\delta : Q \times \Sigma \to Q$, called the transition function. Let also $s \in Q$ and $F \subset Q$ be respectively the start state and the final states. Then a finite state automaton (FSA) is a 5-uple $(\Sigma, Q, s, F, \delta)$.*

Definition 2. *Let $x \in \mathbb{Z}^d$, $x = (x_1, x_2, \ldots, x_n)$, and let $e_i = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ the point with 1 as the i-th coordinate. We call neighbor of x any element of the set $N_x = \{x - e_i, x + e_i \mid 1 \leq i \leq d\}$. We see that we can order the elements of N_x according to the sign: $x - e_i$ before $x + e_i$ and thereafter according to i, writing, for example, $x - (1, 0, 0, \ldots, 0) <_N x + (0, 1, 0, \ldots, 0)$.*

* LIP, ENS-Lyon, CNRS URA 1398 46 Allée d'Italie, 69364 Lyon cedex 07, France. e-mail codrin@ens-lyon.fr

** Grima, IUT Roanne, Université J. Monnet 20 avenue de Paris, 42334 Roanne cedex, France. e-mail Eric.Remila@ens-lyon.fr

Definition 3. *A cellular automaton over $\mathbb{Z}^d$ is a couple (Q, δ) with the notations above, where $\Sigma = Q^{2d}$. A configuration is a function $c : \mathbb{Z}^d \to Q$. Given two configurations c_1 and c_2, we note $c_1 = \delta(c_2)$ if $c_2(x) = \delta(c_1(x), c_1(y_1), c_1(y_2), \ldots, c_1(y_{2d}))$, where $y_i \in N_x$ and $y_i <_N y_{i+1}$ for $1 \leq i < 2d$.*

2.2 Leader election

The problem of leader election is defined as follows: Find a cellular automaton on $\mathbb{Z}^d$ such that given a finite connected subset $F \subset \mathbb{Z}^d$, the cellular automaton starts with all the cells members of F in a certain state, exactly the same all over, and with the cells outside F marked with a special symbol #, and after t steps, with t finite, only one cell of F is in a special state, called global leader state, and all the others are in another state, meaning that they cannot be the leader. We also impose that the cells marked with # never change this state, thus confining the computation to the set F.

3 Algorithm

3.1 Idea

We present the algorithm through examples in the integer plane, i.e. in two dimensions. However, the general rules and the proofs are extended to $\mathbb{Z}^d$.

We call a cell a *local leader* when the only neighbors it has in the figure are on the positive senses of the coordinate axis.

Each cell which is a local leader (here, for $d = 2$, this means the most south-western with respect to its neighbors), becomes the root of a spanning lattice, constructed through signals issued from it and propagated across the surface occupied with cells. This lattice expands in pulses until it hits another similar lattice (because in an arbitrary simply-connected plane figure there can be several local leaders) or until it fills the whole region. In the first case, one of the two lattices dies off, and in the second one, it makes the leader the unique global one. We decide which of the lattice wins based on the lexicographic ordering $<_L$ of the two local leaders generating the lattices, by making so that the smaller win. This way, the smallest local leader cell of the figure is elected as global leader.

3.2 Examples

Let us first consider the simplest figure : a rectangle. There is only one local leader, the south-western corner, and from this cell a lattice starts growing, until it reaches the borders. The local leader sends a lattice expansion signal to its northern and eastern neighbors, which is propagated farther. Each cell receiving a signal from a direction among north, south, east and west, sends it in the other three directions. When a cell receives several signals, it considers them all as fathers, and sends further the signal to the others left, named sons. This way, we can speak of the "building" of the lattice.

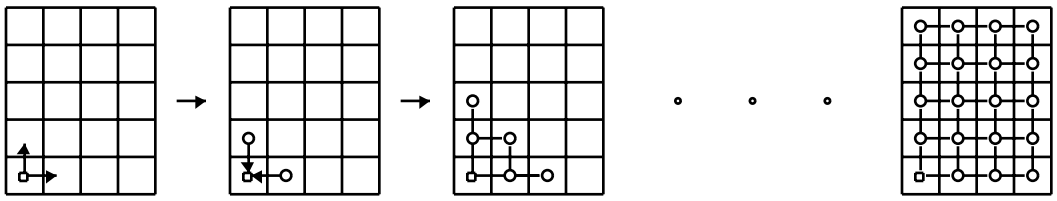


Fig. 1: initial phase expansion steps region filled

We said the growth is pulsed ; this means after one expansion step we have a report step, when the lattice border cells may send back a "report" signal, each cell to its fathers, and these propagate them back to the root of the lattice. We need this in case of conflict, in order to make the conflicts be synchronously solved, as we explain farther in the paper.

A father thus may send back a report, to its fathers. For this example, we only need to consider the termination of the algorithm. During the expansion step, if a cell has no neighbor to send further the

expansion signal, then it switches to a "dead-end" state, otherwise it sends a "border signal". On the other hand, when a father has only dead-end sons, it switches also to dead-end state. Thus, when the root has only dead-end sons, it knows that it is the global leader.

This algorithm also works when the figure is more complicated than a rectangle, yet it has only one local leader. We now have to explain how we deal with several local leaders. At "mid-way" at some point of the expansions, there is a conflict between the two lattices. Therefore we have to identify and order the lattices in a certain way. Since we can have an unbounded number of local leaders and we want a finite number of states, we cannot decide to have different states for different lattices (we wouldn't even know how to assign them locally). However, we can borrow an idea from [4], which originally consists in measuring the coordinate difference between the emitter of a signal (in this case the expansion signal) and its receiver using the time difference between a reference signal and a coordinate signal, retarded when going say positively along the coordinate axis, and accelerated when going the other way, here negatively.

We will explain in the section 3.3 how this works. The idea is briefly illustrated in figure 2, where the cell with two concentric circles has to decide whether $\ell_1 <_L \ell_2$ or $\ell_2 <_L \ell_1$.

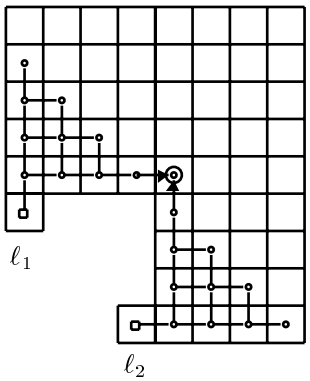


Fig. 2: conflict in the cell with two concentric circles

3.3 Solving the conflicts

First, we forget about the fact that the expansion is pulsed. We consider some "grown" lattices of the same maximum length (from each root to the respective border), and we study the expansion step leading to their touching, thus to a conflict. We have to make the border cells able to measure the coordinate difference of the conflicting local leaders.

Measuring the coordinate difference using signals We do this using, as we have said, an idea from [4], based on signals. Informally speaking, a signal is a special state (or substate) s which "propagates" from a cell x to another cell y, along a specific path $(x = z_1, z_2, \ldots, z_p = y)$.

For each coordinate, thus here for the two, x and y, the local leader sends a signal, at variable speed, to all its children. When the coordinate signals go along the positive sense of the coordinate axis, they are handed from a cell to another at one time unit difference, that is, if the cell receives the signal at time t, then at time $t+1$ the neighbor receives it, and the signal is said to travel at speed 1 (maximal speed). If the coordinate signals go along the negative sense, the reverse happens : they are received at time t and sent (thus received by the neighbor) at time $t+3$, traveling at speed $\frac{1}{3}$. And, when a coordinate signal travels along a different axis, it goes at a speed in between, which is $\frac{1}{2}$ (being transmitted at the second time unit to the neighbors).

Thus if we consider the speed $\frac{1}{2}$ as a "reference" speed, each coordinate signal gets accelerated when going in the positive sense of the respective coordinate axis, decelerated in the opposite and not perturbed (i.e. traveling at speed $\frac{1}{2}$) when going perpendicularly.

These signals serve the double purpose of expanding the lattice and measuring the coordinate difference.

The former is done by memorizing the neighboring source cells (as fathers) by each cell receiving a coordinate signal. Actually, the (possibly) fastest coordinate signals "open" the path, followed by the subsequent coordinate signals. We will see this in detail in the next section.

The latter is done as follows: suppose we have an emitter cell $c_0 = (x_0, y_0, \ldots)$ and a receiver cell $c = (x_c, y_c, dots)$. The x-coordinate signal is sent at time 0, it arrives at the cell c time t_c, and it travels a path of length l (i.e. it crosses l edges to get there). Then the simple law of speed and distance tells us that

$$x_c - x_0 = 2l - t_c \tag{1}$$

The following figure shows the path of the x and y coordinate signals from ℓ_1 to the cell marked with two concentric circles in the figure 2. We call this a *beam* of coordinate signals.

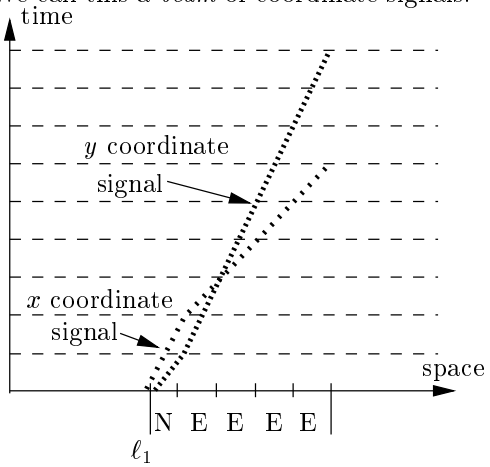


Fig. 3 : coordinate measure signals: a beam

Conflicts Suppose now we have two synchronous emitter cells $\ell_1 = (x_1, y_1, \ldots)$ and $\ell_2 = (x_2, y_2, \ldots)$, as in figure 2. Let c be the "conflict cell" (the one with two concentric circles on the figure 2). It happens that the length l of the path from the emitters to it is the same. We have from the equality (1)

$$x_c - x_1 = 2l - t_1 \text{ and } x_c - x_2 = 2l - t_2$$

where t_1 (respectively t_2) denotes the arrival time of the x coordinate signal from ℓ_1 (resp. ℓ_2) in c.

Then,

$$x_1 < x_2 \iff t_1 < t_2 \tag{2}$$

Therefore, the cell c is able to find the minimun in sense of the lexicographic order $<_L$ (see figure 6). One method is to store the order of arrival for each coordinate signal, and at the end, to make the right choice.

We actually see that it is enough to know which signal arrived first from which of the beams, all this for each coordinate. We explain the implementation of this in the section 3.4, paragraph Conflict.

However, we need that the emitters be synchronous and that they be at equal distances from the conflict cell. All this is actually possible because

- at the beginning all the local leaders are synchronized, because the initialization is synchronous
- the lattices expand "uniformly"
- we ensure the further synchronization as wee see in the next paragraph.

Synchronization We set up a second signal, called a sweep signal, going slower than any coordinate signal, i.e. at speed $\frac{1}{3}$, and traveling along the same paths. Thus, a cell "takes" the decision when all sweep signals have arrived, sending it back to the local leaders through other signals: border signal to the winner, destroy signal to the losers. On the other hand, with the sweep signal system we make sure that all the border cells send back their report at the same time.

We will see that the sweep signals arrive all at the same time (or at exactly three time units difference, in a special case), thus ensuring the synchronization and pulse of the lattice expansion process.

If we consider the figure 3, and show the space-time diagram of the sweep and x (East-West) coordinate signals issued from the two local leaders ℓ_1 and ℓ_2 on the path containing the conflicting cell, we get the following figure :

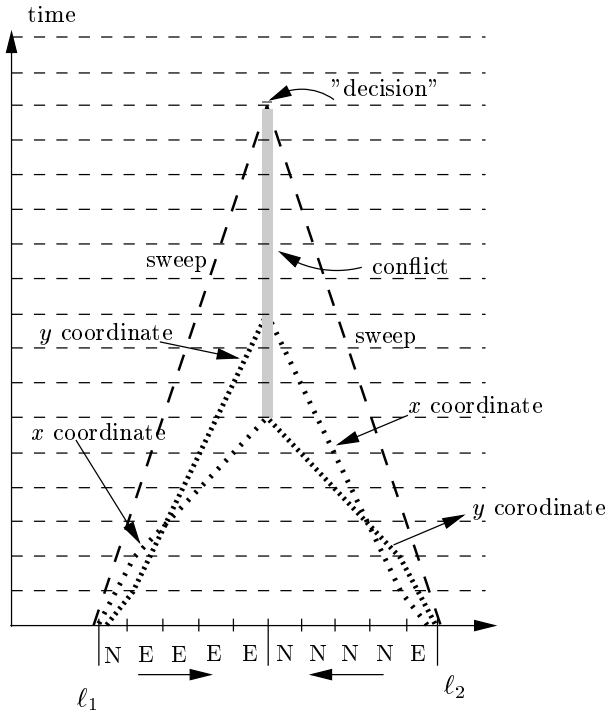


Fig. 4 : conflict and decision; the edge labels are in the sense of the arrows

3.4 Memory structure

We actually see the states of each cell of the figure as elements of a cartesian product, each component taking care of a specific aspect : action to be taken by the cell, information about the signals, information about the neighbors, information to solve the conflicts. By abuse of language, we call states the components having to do with the action or the status of the cell.

Main components of states A cell can be in any of the following states : start state, inactive state, potential global leader state, lattice state, conflict state, decision state, dead-end state and global leader state. This makes the automaton look as shown in the following figure:

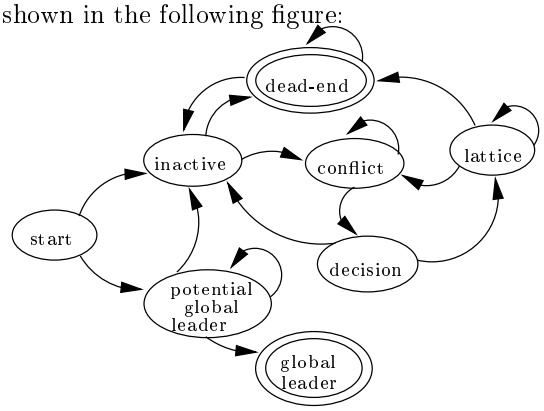


Fig 5: Finite State Automaton schematic diagram

Signals We associate with each cell a signal vector SV, to handle the signal transmission, composed of counters for each signal. As we said earlier, the counter $SV[i]$ encodes how long the signal i has been held

in the cell. The sweep, destroy and border signals are three supplemental signals, so SV has $d+4$ elements. A value of 0 in $SV[i]$ means the signal i (coordinate, sweep, destroy or border) is not in the cell, a value of 1 means the signal just arrived, and so on, with values no greater than 3.

Neighborhood We also need to encode information about the $2d$ neighbors. Thus, each cell has also an array NV of $2d$ elements with values from $\{?, S, F, FW, FL\}$ meaning no information, son, father, father winner, father loser. The ith element of the array gives information about the ith neighbor (in the order defined in section 2). Also, when the cell receives the sweep signals and takes the decision, it has to communicate it to its fathers, and this is done using the values FL and FW put in the elements of the array NV.

Conflict In case of conflict, the cell has to be able to find the minimum in lexicographic order of arrivals of coordinate signals, so we also need a $d \times 2d$ boolean matrix CM such that $CM[i,j] = 1$ means the neighbor j (in the same order as above) is among the very first to have sent the i-coordinate signal to the conflict cell, and otherwise, $CM[i,j] = k$ means the neighbor j is among the ones having sent the i-coordinate signal one time unit *after* the j' with $CM[i,j'] = k-1$ Here, k is 1, 2 or 3. In any other case, $CM[i,j] = 0$.

3.5 Transition rules

We now describe how the cells switch from a state to another, updating at the same time their NV and SV arrays.

Initialization At starting time, all cells are in start state. When a cell is in start state, if it is a local leader, it switches to potential global leader state, otherwise, it switches to inactive state.

The cells bordering the figure (outside it) are all marked with a special symbol #, and the arrays of all the figure cells are all set to zero.

Expansion The cell being in potential global leader state sends the coordinate signals to its neighbors. When a cell being in inactive state receives one coordinate signal, it changes the NV to note the source of the signal as father, and waits for the other coordinate signals and for the sweep signal. It also switches immediately to conflict state. If it receives several coordinate signals from different sources, it notes them all as its fathers in the NV array.

When a cell being in lattice state receives some coordinate signals, it propagates them farther, according to the delay rules (acceleration if in the positive sense, deceleration if the opposite, speed $\frac{1}{2}$ if perpendicular direction) to its children.

There is also another case of switching to conflict state, explained as Special case.

Dead-end and stop Les us call the unknown neighborhood the neighbor cells of a cell which are marked with ? in the NV array of the cell. If, for a cell, all the unknown neighbors are actually in state #, and all its children are in dead-end state, the cell switches to dead-end state as well. If the cell being in potential global leader state sees all its children in dead-end state, then it switches to global leader state.

Conflict and decision We call *conflict period* for a cell the period of time between the switching to conflict state (reception of a first coordinate signal, being in and the switching to lattice. When a cell switched to conflict state from inactive state, it waits to receive all the coordinate and the sweep signals. Whenever it receives a coordinate signal, say for coordinate i, the first time, it writes 1 to $CM[i,j]$ where j runs through the neighbors which sent it this signal the first. This way, only these neighbors are noted as potential winners, and, at the end, the cell takes a decision, according to the lexicographic order of the coordinates, switching to decision state, and setting the NV appropriately: if the i-th neighbor is a winning father, it puts $NV[i]$ to FW, and for all j which denote the losing fathers, it puts $NV[j]$ to FL. Thus, when a cell is in lattice state and sees some of its children switching to conflict state and then to decision state, it looks in their NV arrays at the right elements, to "learn" the outcome of the conflict. If at least one of its children tells it that it is FL, then it propagates the destroy signal back to its fathers, thus back towards its potential global leader, otherwise, it propagates the border signal.

Special case This happens when two local leaders are separated by a path of even length. In this case we have two cells which enter the conflict state and are at the same time neighbors, each one belonging to a lattice and only one. When this happens, the cell "ignore" each other and behave as said before. However, at the next expansion they might receive some coordinate signals from the established fathers *and also* from unknown neighbors (of F (not yet fathers or children). The reasoning would be the same as the one described in section 3.3, but there is a difference : the length l is no longer the same, it is l from the "known" part (the fathers, actually from the potential global leader ℓ_1) and $l+1$ from the "unknown" part (ℓ_2). Thus the equality (2) becomes

$$x_1 < x_2 \Longleftrightarrow t_1 < t_2 + 2 \tag{3}$$

This is why the values in CM go from 1 to 3. Therefore, the rule is enriched with the case of switching to conflict state from lattice state, which is this very case: if a cell being in lattice state and still having unknown neighbors in F (that is marked with ? in its NV array) receives a coordinate signal, then it switches to conflict state. It also starts filling the CM matrix with values from 1 to 3, and at the end decides again about the order, but this time by saying that

1. if the signals with 1 arrived at least from some unknown neighbors, then these neighbors win and only them
2. else if the signals with 2 arrived at least from some unknown neighbors, then the same happens
3. else if the signals with 3 arrived at least from some unknown neighbors, then they win, but *also with* the known (fathers) neighbors which sent signals arriving first (CM value 1).
4. else only the known father neighbors which sent signals first win (CM value 1)

Suppose we have the figure 2 slightly modified, with the row of ℓ_2 being one cell lower, in order to illustrate this case, and call the two new potential global leaders ℓ'_1 and ℓ'_2. Then we have the following space-time diagram:

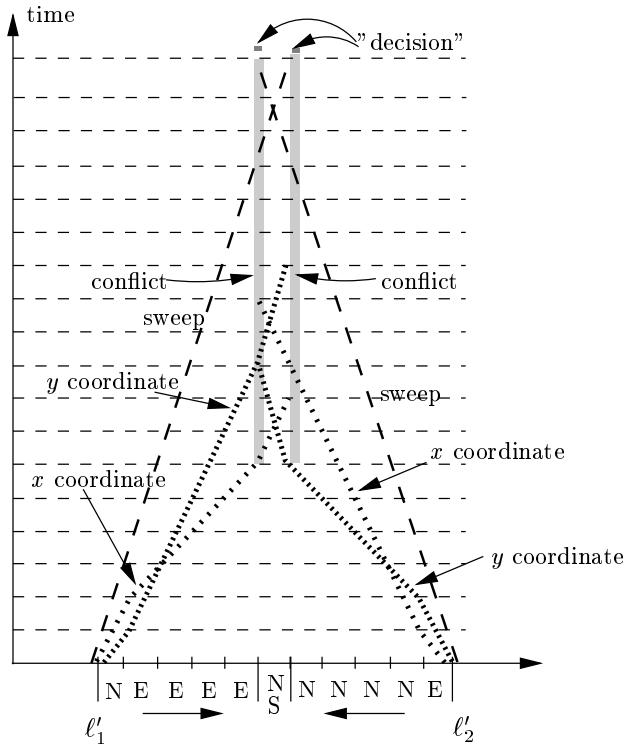


Fig. 5 : conflict and decisions

For the left conflict cell, ℓ'_1 is declared winner because its x coordinate signal arrives before $t-2$ where t is the arrival time of the x coordinate signal from ℓ'_2, which is the unknown part for the left conflict cell (case 4 from above, and the order on x, if any, decides lexicographically). For the right conflict cell, again ℓ'_1 is declared winner because its x coordinate signal, this time from the unknown part, arrives one time unit before the x coordinate signal from ℓ'_2 (case 1 from above).

Decision transmission The decisions are taken when the last sweep arrives. In all cases, the signals arrive (also) from some unknown neighbors, and the decisions are notified only to these unknown neighbors, because in case of signal crossing (that is even length), each of the conflict cell takes care of the other's fathers. Also, the FW values of NV are switched thereafter to F, and all the others (F or FL) to ?.

Report When a cell is in lattice state and it receives from its children a border or destroy signal, it propagates it up to its fathers.

Destruction When a cell being in potential global leader state receives at least one destroy signal from one of its children, then it sends the destroy signal to all its children and it switches to inactive state. When a cell being in lattice state or dead-end state receives a destroy signal from one of its fathers, then it propagates it to all its children, and it switches to inactive state.

Updating and iteration When a cell being in potential global leader state receives from all its children only border signals, it starts a new expansion period, sending again the coordinate signals.

4 Analysis

4.1 Definitions

We have been talking about lattices and about constructed and expanded lattices. We now set this properly.

Definition 4. *By constructed lattice of radius h of a local leader ℓ we mean the intersection of a ball $B(\ell, h)$ of $\mathbb{Z}^d$, centered in a local leader, with the initial figure $F \subset \mathbb{Z}^d$, in which all the cells are in special states : potential global leader state, lattice state or dead-end state, and have the NV appropriately set. The cell being in the local leader state (that is the potential global leader) has no father and is (transitively) the father of all the other members of the constructed lattice. The relations father-son are oriented outwards from the leader to the borders and the cells which have to be in dead-end state (because all their children are in dead-end state or are marked with #) are in this state.*

4.2 Proofs

If we look at the evolution of the algorithm, we can remark a certain periodicity. The start of the h-th period is the moment when the potential global leaders "know what to do", i.e. at the beginning to start the expansion, and afterwards to continue or to destroy themselves. We denote by this the time coordinates $m_i = m_{i-1} + 4i$, where $m_1 = 1$.

Proposition 1. *Let $h < w$. At the start of the h-th period the constructed lattices are of size h, with h being at most the length of the longest among the shortest path from the future global leader to the borders of the figure.*

The main loop gives the main steps of the h-th period:

1. *at (relative) time coordinate 0, the potential global leaders have decided either to expand or to destroy, and are about to send the appropriate signals. Their constructed lattices of radius h are disjoint, with no signals being held for transmission, and the rest of the cells are in the inactive state.*
2. *at time h, the constructed lattices which had to be destroyed are indeed destroyed, and all their cells switched to inactive state, having no more signals held to be transmitted to neighbors and clearing the neighbor array NV.*
3. *between time $h+1$ and time $3h+3$ the expansion of the surviving constructed lattices is completed, and conflicts detected by the new border cells.*
4. *at time $3h+3$ the sweep signals have reached the borders, and the conflicts are solved.*
5. *at time $4h+4$ the decisions are reported to the local leaders, and a new period starts.*

Proof. Straightforward, by induction, and by the construction of the algorithm. We can emphasize that there is no interference between the expansion and the destruction of two neighboring constructed lattices, because the destroy signal travels as fast as the fastest coordinate signal, but has a one-cell-shorter path to go to a same cell, element of the current border.

Also, since the report signals only travel within each constructed lattice, there is no interference either, and they "safely" get back to the local leaders, informing them about the possibility of expansion or necessity to destroy themselves.

We also insist on the synchronization of the process, all lattices starting the expansion at the same moment, and solving the conflicts, sending back the decisions and restarting the loop again at the same moment. This is ensured by the sweep signals traveling at a constant speed, along the same distance, the radius h, from the local leaders to the borders, and the report signals, traveling again at a constant speed, along the same maximal distance, and by the fact that the local leaders wait for all the report signals to come before restarting the period. □

Proposition 2. *The constructed lattice rooted in the local leader which should be elected according to the lexicographic order of the coordinates never looses any conflict.*

Proof. By the construction of the algorithm, from the way the conflicts are handled. The beam of coordinate signals from the future global leader will arrive "faster" than any other coordinate signals (that is with the appropriate mark), thus making the border cell take the right decision each time. □

Proposition 3. *If there is only one potential global leader at the beginning of a time period (at a m_i moment), then the algorithm finishes electing it as the global one.*

Proof. The dead-end states start propagating themselves towards the root once the border cells are reached by the sweep signal, and by the construction of the algorithm, the potential global leader becomes global. □

Theorem 1. *The algorithm finishes in $2(w^2 + w)$ time steps, where w is the length of the longest among the shortest paths starting from the global leader and going to the border of the figure.*

Proof. From the way the conflicts are handled, we see that after each conflict at least one constructed lattice will dissapear. Since between the conflicts the constructed lattices only grow, if at the beginning there are several ones, during the evolution their number decreases strictly monotonically. We arrive in the case of the previous proposition and thus the algorithm stops.

On the other hand, we can only look at the evolution of the constructed lattice of the future global leader. This constructed lattice keeps expanding itself, without delay, and without being destroyed (previous propositions), and $m_w = 4\sum_{k=1}^{w} k = 4\frac{w(w+1)}{2}$. □

5 Discussion

We can think of extending this algorithm to other types of graphs. We can do this on labeled trees and we can hope to do it on some special Cayley graphs, under specific conditions. M ore work has to be done to relax as much as possible of these conditions, or to classify the graphs and find necessary conditions for the leader election problem to be solvable.

References

1. A. Beckers and T. Worsch. A perimeter-time CA for the queen bee problem. In *Cellular Automata: Research towards Industry. (Proceedings of ACRI'98)*. Springer Verlag London, 1998.
2. A. Hemmerling. Concentration of multidimensional tape-bounded systems of turing automata and cellular spaces. In *Proceedings of FCT 1979*, 1979.
3. A. R. Smith III. Two-dimensional formal languages and pattern recognition by cellular automata. In *IEEE Conference Record of 12th Annual Symposium on Switching and Automaa Theory*, 1971.

4. J. Mazoyer, C. Nichitiu, and E. Remila. Compass permits leader election. In *Proceedings of SODA 1999*. SIAM, 1999.
5. P. Rosenstiehl, J. R. Fiksel, and A. Holliger. Intelligent graphs: Networks of finite automata capable of solving graph problems. In *Graph Theory and Computing*, pages 210–265. R.C. Reed (editor), Academic Press, New York, 1973.
6. A. Wu and A. Rosenfeld. Cellular graph automata. I. Basic concepts, graph property measurement, closure properties. *Information and Control*, 42:305–329, 1979.