Probabilistic Distributed Algorithm for Uniform Election in Triangular Grid Graphs

El Mehdi Stouti
FS–Abdelmalek Essaâdi University
P.O. Box. 2121 M'Hannech II
93030 Tetuan Marocco
Email: stouti@uae.ma

Ismail Hind
FS–Abdelmalek Essaâdi University
P.O. Box. 2121 M'Hannech II
93030 Tetuan Marocco
Email: ismailhind@gmail.com

Abdelaaziz El Hibaoui
FS–Abdelmalek Essaâdi University
P.O. Box. 2121 M'Hannech II
93030 Tetuan Marocco
Email: hibaoui@uae.ma

***Abstract*—Probabilistic algorithms are designed to handle problems that do not admit deterministic effective solutions. In the case of the election problem, many algorithms are available and applicable under appropriate assumptions, for example: the uniform election in trees, $k-$trees and polyominoids.**

In this paper, first, we introduce a probabilistic algorithm for the uniform election in the triangular grid graphs, then, we expose the set of rules that generate the class of the triangular grid graphs. The main of this paper is devoted to the analysis of our algorithm. We show that our algorithm is totally fair in so far as it gives the same probability to any vertex of the given graph to be elected.

***Keywords*—Uniform Election, Distributed Algorithms, Probabilistic Election, Markov Process, Randomized Algorithm Analysis.**

I. Introduction

Election in a network is to chose one and only one element of this network. The elected element may be used to manage such shared resources (printer, connection, etc.), or to centralize some network informations (size, diameter, etc.).

The election problem holds the attention of many researchers since it was first proposed by LE LANN [1]. Therefore, it has been studied under various assumptions: the proposed network could be oriented or not, synchronous or asynchronous (no shared global clock), anonymous or with identifiers (no unique identity is attributed to elements), size knowing or not, etc.

The solutions take also into account the network topology. Some solutions are deterministic while others are probabilistic. Another aspect is that we want also to study the uniform election. In this type of election, we attribute the same chance to all nodes and at any position in the network to be elected. The algorithms known in the literature are probabilistic and run on well defined topologies. We quote for trees [2][3], for k-trees [4] and for polyominoids [5]. The work presented here is a continuation of this researches. Thus, we introduce a probabilistic algorithm for uniform election in a network with the topology of triangulated grid graphs.

The triangular grid graph is, in graph theory, a finite sub-graph induced from the infinite graph associated with the two-dimensional triangular grid [6]. It is a subclass of planar graphs [7]. However, networks discussed in this work have the topology of a triangular grid graphs. We assume that the network can be synchronous or asynchronous, and it is anonymous; no unique identity is attributed to its vertices.

The main objective behind this study is to suggest and analyse the uniform election of a probabilistic distributed algorithm in the triangular grid graphs. So and for a given graph G, each vertex $v \in G$ generates its lifetime duration depending to its weight w_v. The lifetime of a vertex v is an exponential random variable of parameter λ_v equals to its weight w_v. According to our algorithm, when the lifetime of a vertex expired, it removed with its incident edges, and its neighbour in the standard spanning tree recovers its weight.

The analysis of this algorithm proves that, whatever the vertex is situated in the studied graph, it has the same probability to be elected. We can consider our algorithm as a probabilistic variant of the distributed algorithm introduced in [8], where random delays are presented.

We consider local computations in the cells. At each step of computation, the vertices of a cell can change their status (or labels). Indeed, the new label of a vertex depends on its previous label (state) and the labels of its neighbours. In our approach we used a random delay for labelling; a vertex can not change its state if its associated lifetime duration is not over. These delays are exponential random variables, independently defined, for active vertices.

The parameter of a random variable associated with a vertex is equal to the weight assigned to this vertex. The weight is locally calculated in term of initial weight and the weights collected from the vanishing neighbours. The labelling process continues until no transformation is possible, that is to say, the last configuration is reached. In this configuration, there was only one vertex which has a different tag (label) from the others, this vertex is considered as the elected one (leader) [5].
For the analyse of this algorithm we model the elimination process with a continuous time of a Markov death Process.

This paper is organized as follows. In Section II, we give

some definitions required to understand the rest of the paper.In section III we give a set of rules generating the class of triangular grid graphs. The section IV devoted to the analysis of a probabilistic algorithm for uniform election in this family of graphs. In section V we presents the operation of the algorithm, providing some tools for its analysis (section 5).

II. Preliminaries and notation

A Triangular Grid Graph (**TGG**) is a finite graph where its vertices are points in $\mathcal{Z} = \mathbb{Z} \times \mathbb{Z}$, where $\mathbb{Z}$ denotes the set of decimal integers. They are linked by neighbourhood relationships [7].

The edges are the links between pairs of points. They are of the forms : $\{(x, y), (x, y + 1)\}$ or $\{(x, y), (x + 1, y)\}$ or else $\{(x, y), (x + 1, y - 1)\}$ for all $x \in \mathbb{Z}$ and for all $y \in \mathbb{Z}$ (see Fig 1).

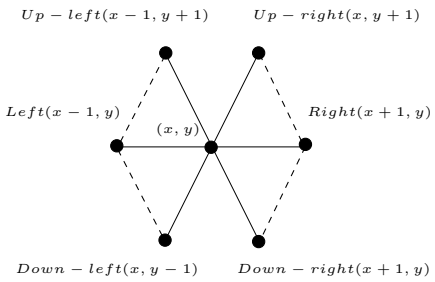


Figure 1. Vertex's neighbours in TGG.

Two vertices $v = (x, y)$ and $v' = (x', y')$ of $\mathcal{Z}$ are neighbours if one of the following conditions are satisfied:

- $y = y'$ and $|x - x'| = 1$, or
- $x = x'$ and $|y - y'| = 1$, or
- $|x - x'| = 1$ and $|y - y'| = 1$.

Two neighbours are the ends of an edge.

For each vertex v of coordinates (x, y), we use the usual terms such as 'up' to denote the neighbour of coordinates $(x, y + 1)$, 'down' for the neighbour $(x + 1, y - 1)$, 'right' for $(x + 1, y)$ and 'left' for $(x - 1, y)$ neighbour, see Fig 1.

Let $\mathcal{S}_\mathcal{E}$ be the set of all edges whose ends are neighbours and $\mathcal{I}_\mathcal{G}$=$(\mathcal{Z}, \mathcal{S}_\mathcal{E})$ the infinite graph consisting of the set of vertices $\mathcal{Z}$ and the set of edges $\mathcal{S}_\mathcal{E}$. A cell is a sub-graph of $\mathcal{S}_\mathcal{E}$, induced by a set of three pairwise neighbour vertices $\{(x, y), (x + 1, y), (x, y + 1)\}$ having the form $\triangle$, called *up triangle cell*, or else $\{(x, y), (x + 1, y), (x + 1, y - 1)\}$ for the vertices with form $\bigtriangledown$, called *down triangle cell*.

A path is a finite alternated sequence $\sigma = v_0, e_1, v_1, \cdots, v_{k-1}, e_k, v_k$ of $k + 1$ vertices and k distinct edges $(k \geq 0)$, such that v_{i-1} and v_i are the ends of the edge e_i, for $1 \leq i \leq k$.

We recall that the length of a path σ is the number of its edges k. It should be noted that a path may pass several times through a vertex, but can not borrow an edge more than once. A cycle is a path of length $k \geq 3$ in which the first vertex v_0 and the last vertex v_k coincide. For a given cycle, we can easily and according to [9] define the vertices or edges inside this cycle.

Definition 2.1: A vertex (x, y) is inside the cycle γ=$(x_0, y_0), (x_1, y_1), \cdots, (x_{k-1}, y_{k-1}), (x_0, y_0), ((x_k, y_k) = (x_0, y_0))$ if (card $\{i \mid y = y_i$ and $y \neq y_{i+1}$ and $x \leq x_i\}$) is odd. Therefore, the boundary vertices of γ are inside γ.

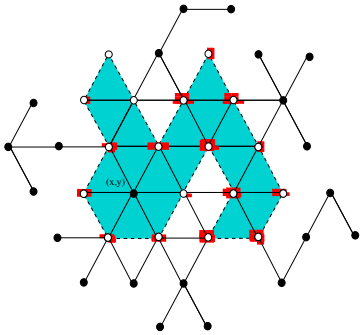


Figure 2. Example of vertex inside a cycle.

Proposition 2.1: For a given TGG and for a given vertex v inside a cycle γ in this graph, we have for each *linear* path (LP) including v, the number of vertices belonging to both LP and γ is even.

Example 2.1: The vertex (x, y) of the figure 2 is inside the cycle consisting by the white vertices and the edges in bold. For the linear path $LP = (x, y), (x + 1, y), (x + 2, y), (x + 3, y), ...$, we have the cardinal of vertices set inside the said cycle and belonging to LP is even.

Definition 2.2: A triangular grid graph $G = (V, E)$ is a sub-graph of $\mathcal{I}_\mathcal{G}$ if the following conditions are satisfied:

1) V is finite,
2) G is connected and
3) G does not contain holes, i.e. for any cycle γ in G, the vertices inside γ are contained in V and if two neighbours are inside γ, so the edge connects these two vertices belongs to E.

We define the size of $G = (V, E)$ as the cardinal of V.

Definition 2.3: A triangular grid graph $G_s = (V_s, E_s)$ is called sub-triangular grid graph of the triangular grid graph $G = (V, E)$ if only if $V_s \subseteq V$ and $E_s \subseteq E$ such that $E_s = E \cap \{\{u, v\} \mid \exists! e = (u, v)$, for all $(u, v) \in V_s^2\}$.

III. Distributed constraction of triangular grid graph

Let $\mathcal{S}_\mathcal{P}$ be the set of partial sub-graphs of the infinite graph $\mathcal{I}_\mathcal{G}$ obtained by the following inductive rules:

a) For all $(x, y) \in \mathcal{Z}$, the graph $G = (\{v = (x, y)\}, \phi)$ is in $\mathcal{S}_\mathcal{P}$.

b) Let $G = (V, E) \in \mathcal{S}_\mathcal{P}$. Consider two neighbouring vertices v and v' such that $v \in V$ and $v' \notin V$, then $G' = (V \cup \{v'\}, E \cup \{\{v, v'\}\})$ is in $\mathcal{S}_\mathcal{P}$.

c) Let $G = (V, E) \in \mathcal{S}_\mathcal{P}$. Suppose that V contains three neighbouring vertices $v_1 = (x, y)$, $v_2 = (x+1, y)$ and $v_3 = (x, y + 1)$ or else $v_1 = (x, y)$, $v_2 = (x + 1, y)$ and $v_3 = (x + 1, y - 1)$ located in a cell of $\mathcal{I}_\mathcal{G}$, such as two edges of this cell are in E and the third one, called e, is not then the graph $G' = (V, E \cup \{e\})$ is in $\mathcal{S}_\mathcal{P}$.

The construction is totally distributed and applying rewrite rules, as seen in [10], requires only knowledge of neighbouring areas that are in a ball of radius 1. Therefore, the local construction can be expressed by considering transformations assigned to a vertex v and the set of all its neighbours. In this case, it is difficult to show that the set $\mathcal{S}_\mathcal{P}$ is the class of all the triangular grid graphs on $\mathcal{I}_\mathcal{G}$. The following proposition proves the equivalence of the two definitions.

Proposition 3.1: A partial sub-graph $G = (V, E)$ of $\mathcal{I}_\mathcal{G}$ is a triangular grid graph iff it belongs to $\mathcal{S}_\mathcal{P}$.

Proof:

$\Longleftarrow$ Let $G = (V, E) \in \mathcal{S}_\mathcal{P}$ and prove that G is a TGG. We just need to prove that the constructions given by the rules (b) and (c) preserving the structure of the triangular grid graphs. So, suppose that G is a TGG and prove that the sub-graph G'_b of $\mathcal{I}_\mathcal{G}$ obtained by (b) and the sub-graph G'_c of $\mathcal{I}_\mathcal{G}$ obtained by (c) are also triangular grid graphs. The properties of the connectivity and the finiteness are obvious. We will show that no hole is created during the application of the rule (b) or the rule (c).

- Applying the rule (b), a new vertex v' is added to G. Since v' is of degree 1, there is no new cycle in G'_b and all the vertices inside a cycle in G remain inside the same cycle in G'_b. Obviously, the same fact is verified for every edge whose ends are in G'_b.

- Let $G'_c = (V, E'_c)$ be an extension of the triangular grid graph $G = (V, E)$ obtained by applying the rule (c). Let v be a vertex inside the cycle γ included in G'_c. If all edges are in E, then v should be in V. Otherwise, we use an edge of a cell formed by the set of the vertices $\mathbf{S} = \{v_1, v_2, v_3\}$, say $\{v_1, v_2\}$, which does not belong to E. In addition $E'_c = E \cup \{v_1, v_2\}$. In this case, it is possible to transform γ into another cycle γ' included in E avoiding v_1 and borrowing other vertices of the set $\mathbf{S}$.

$\Longrightarrow$ Let $G = (V, E)$ be a TGG. We show by induction proof on the cardinal of the set V that G belongs to $\mathcal{S}_\mathcal{P}$.

- If V is of cardinality 1, then obviously $G \in \mathcal{S}_\mathcal{P}$.

- Now suppose that a graph G with size $n \geq 2$ is a TGG, then $G \in \mathcal{S}_\mathcal{P}$. We prove that it's true for a graph G' with size equals to $n + 1$.

  Let $G' = (V', E')$ be a TGG of size $n + 1$. If G' has a vertex v of degree 1, then when we delete v and its incident edge, G' will transform to G. It is clear that G preserves the properties (1)–(3) of Definition 2.2 and therefore, by the recurrence assumption, it belongs to $\mathcal{S}_\mathcal{P}$. Indeed, an application of Rule (b) allows that G' is also in $\mathcal{S}_\mathcal{P}$.

  Suppose now that all vertices of a triangular grid graph $G = (V, E)$ are of degree greater than or equal to 2. We have $|E| - |V| \geq 0$, if not, G is a tree and admits a vertex of degree 1.
  We use now a second recurrence on $|E| - |V|$. It is clear that G has at least one cycle. Let γ be a maximum cycle in G. It is easy to see that if we remove an edge from γ, the residual graph obtained, denoted $\mathcal{R}$, preserves properties (1)–(3) seen in Definition 2.2. Thus the induction assumption on $|E| - |V|$ gives $\mathcal{R} \in \mathcal{S}_\mathcal{P}$.
  An application of the rule (c) on the triangular grid graph $\mathcal{R}$ allows to reconstruct the graph G as a member of $\mathcal{S}_\mathcal{P}$.

■

IV. Uniform election in triangular grid graph

A. Model used

We represent a communication network by a graph, where a nodes (stations) are represented by a vertices and the edges represent communication links.

The election algorithm presented here is designed for anonymous networks which have the topology of a triangular grid graph. So, We assume that each vertex does not know the size of the graph neither its own coordinates in the plan. Its only knowledge is the directions of its incident edges.

We use the asynchronous system where no global clock is shared. This means that the transmitter sends the message but there is no information on when the receiver actually receives it. Hence, the processes execute the instructions with arbitrary speeds and the messages reach their destinations in a finite time but also arbitrary.

B. Distributed election

In this section, we describe our election algorithm by a graph rewriting system. The rewriting systems or

more generally the local computations in graph are a powerful models providing general tools to encode distributed algorithms, to understand their power, and to prove their validity [8][10][11][12].
Our distributed algorithm is based on the rewriting systems presented in [12]. Each vertex (resp. edge) has a label that represents its state. In fact, the labels attached to the vertices and the edges are locally modified.

So, initially, all vertices of the TGG have the same weight 1 according to the anonymity condition imposed on the graph. The election process behaves as a continuous-time Markov process.

Each vertex has a weight local knowledge and depending to the situation it changes its state by applying one of the set of R_i and R'_i rules described below.

The random delay associated to each removable vertex are independent and can locally be generated. These vertices may be removed in a random delay which is an exponentially distributed random variable with a parameter equal to the weight of the vertex. Whenever the lifetime (delay removal) of a vertex has expired, it is removed with all its incident edges. The weight of the removed vertex is collected by one of its neighbours according to the R'_i-rules.

The rewriting is performed step by step, then after a number of rewriting steps, we obtained an irreducible graph where no rule is applicable. In this graph there is a special label attached to exactly one vertex. This vertex will be considered as elected one (called *leader*).

The rewriting system applied here uses the forbidden contexts [11][13][14]. The idea is to prevent the application of a rewriting rule whenever the related occurrences are included in some special configurations, called *forbidden contexts*. Thus, a rule can be applied if the two conditions are satisfied:

1) the rule does not occur in a prohibit context already mentioned, and
2) its associated delay has expired.

Formally, let G_R be a connected graph and two marking functions of G_R: the initial labelling λ_R and the final labelling λ'_R.

The rewrite rule with forbidden contexts is a quadruple **R** = (G_R,λ_R,λ'_R,$\mathcal{F}_R$) such that (G_R,λ_R,λ'_R) is a rewrite rule and $\mathcal{F}_R$ is a finite set of forbidden contexts (G_R,λ_R).

$$\mathbf{R} : \{\mathcal{F}_R; (G_R, \lambda_R) \longrightarrow (G_R, \lambda'_R)\}\,.$$

For our case, let $G = (V, E)$ be a TGG and $\mathcal{S}_L$ = {**N**,**A**,**B**,**L**} the set of labels. The label **N** encodes the neutral state, **A** encodes the active state, and **B** encodes the beat state, and **L** encodes the leader state (elected).

We denote by **X** any state except **B**, i.e., **X**$\in \mathcal{S}_L \backslash \{\mathbf{B}\}$.

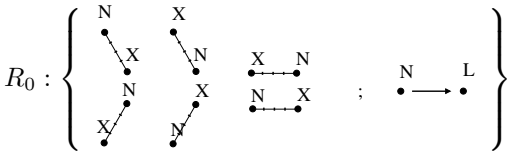
The election process on G runs in distributed manner as follows:
Initially, all vertices have the same weight $w = 1$ and each one is **N**-labelled.

Each **N**-labelled vertex v decides locally whether it is active or not according to the activation rules R_i below. So, if a vertex v becomes active, it generates its lifetime, which is an exponential random variable with parameter equal to its current weight. Once the lifetime of an active vertex is expired, its weight is transmitted to one of its neighbour. In the end, only one vertex is active. This surviving vertex is called the leader.

Activation rules:

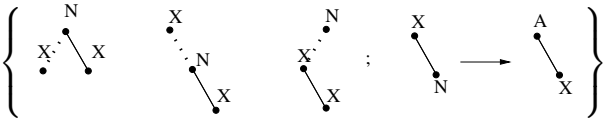
Each vertex in the graph can determine locally if it is active or not according the rules R_i bellows.

- R_0: If the degree of v is zero ($deg(v) = 0$), then the election is over, and v is the elected vertex. It is important to note that this vertex is considered as an active vertex.

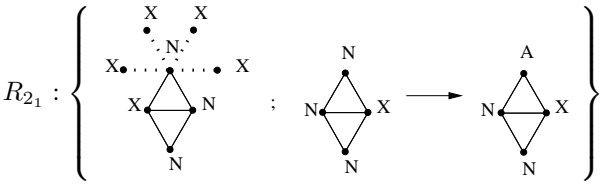


In this rule, the forbidden context shows that v shouldn't have a **X**-labelled neighbour.

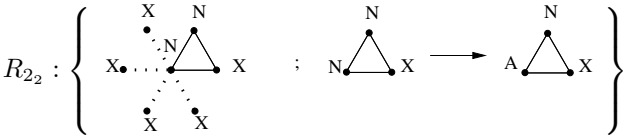
- R_1: If the degree of v is 1, then v becomes active and generates its lifetime. Once the lifetime of the vertex v is expired, it disappears with the incident edge and its neighbour, noted u, recovers its weight.



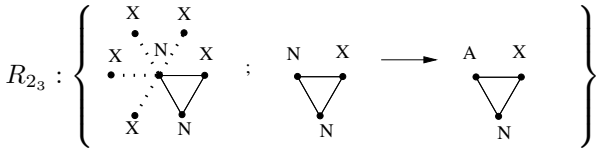
- R_2: If the degree of v is 2 then depending on its position it could become active or not. We distinguish five sub-rules to ensure that v becomes active:



The sub-rule R_{2_1} expresses that if $v = (x, y+1)$ is on the top of a up triangle cell $\{(x, y), (x+1, y), (x, y+1)\}$ with the existence of the down triangle cell $\{(x, y), (x+1, y), (x+1, y-1)\}$, then v becomes active.

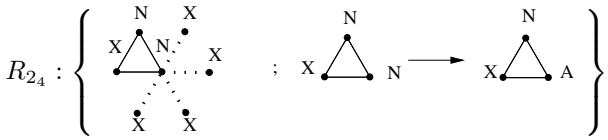


R_{2_3} :

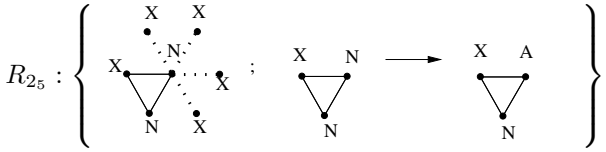


The sub-rule R_{2_2} (resp. R_{2_3}) expresses that if $v = (x, y)$ is on the left of a up (resp. down) triangle cell then v becomes active.

R_{2_4} :



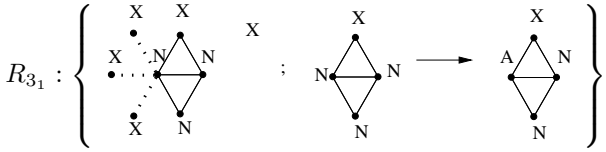
R_{2_5} :



The sub-rule R_{2_4} (resp. R_{2_5}) expresses that if $v = (x + 1, y)$ is on the right of a up (resp. down) triangle cell then v becomes active.

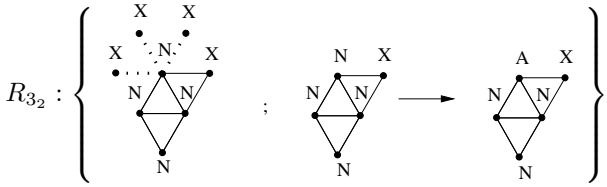
- R_3: If $deg(v) = 3$, then two cases arise:

R_{3_1} :



The sub-rule R_{3_1} describes that if v is either on the left of the up and the down triangle cells, then v becomes active.

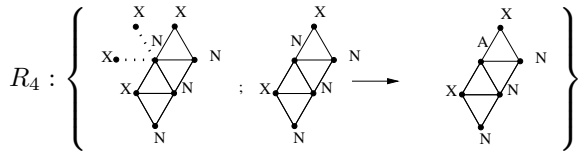
R_{3_2} :



The sub-rule R_{3_2} explains that if $v = (x, y + 1)$ is either on the top of the up triangle cell $\{(x, y), (x, y + 1), (x + 1, y)\}$ and on the left of the down triangle cell $\{(x, y+1), (x+1, y), (x+1, y+1)\}$ and also the the down triangle cell $\{(x, y), (x + 1, y), (x + 1, y - 1)\}$ exists, then v becomes active.

- R_4: If $deg(v) = 4$, then if $v = (x, y + 1)$ belongs to three cells $\{(x, y), (x, y + 1), (x + 1, y)\}$, $\{(x, y + 1), (x, y + 2), (x + 1, y + 1)\}$, and $\{(x, y + 1), (x + 1, y + 1), (x + 1, y)\}$ and the down triangle cell $\{(x, y), (x + 1, y), (x + 1, y - 1)\}$ exists, then v becomes active.

R_4 :



Whenever a vertex v of weight w_v becomes active, it generates its lifetime $L_t(v)$ which is an exponentially distributed random variable (r.v.).

$$Pr(L_t(v) \geq t) = e^{w_v(t)}.$$

This random variable has the expected value $1/w_v$.

Weight transmission rules:

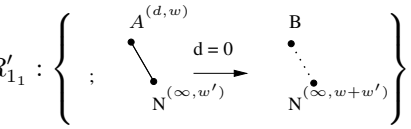
Once the lifetime has expired the vertex will be no longer **A**-labbled and the algorithm removes the vertex with all incident edges, giving its weight to the selected neighbours. The choice of the weight receiver neighbour is done according to rules R'_i. In those rules, d denotes the vertex lifetime and when a vertex is removed all incident edges are removed but we conserve the edges through which the weights are transmitted. Those edges are dotted.

- R'_0: The election is terminated, the remaining vertex is considered as the leader.

- R'_1: The neighbour vertex u of v recovers the weight of v, and it is either in the active state, or in the neutral state. In both cases, we have:

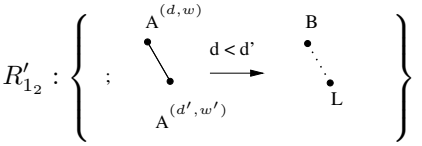
  - R'_{1_1}: If u is neutral then when it recovers the weight of v, it decides locally if it becomes active in the residual graph $G' = (V\backslash\{v\}, E\backslash\{v, u\}, u \in V)$.

R'_{1_1} :



  - R'_{1_2}: Otherwise, u is active before the time when v is removed. So, u becomes the elected one. We will be partially in the case of the rule R_0.

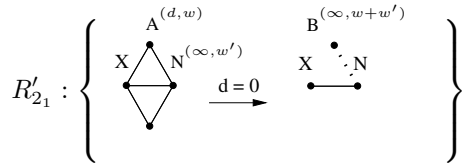
R'_{1_2} :



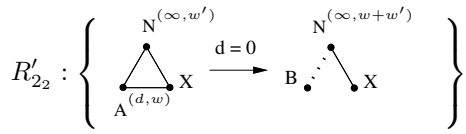
- R'_2: If the lifetime of the vertex of degree 2 is expired, it is removed with its incident edges and the right

neighbour $(x+1, y)$ recovers its weight (in a up triangle case), or else the down neighbour $(x+1, y-1)$ recovers its weight (in a down triangle case). We have:

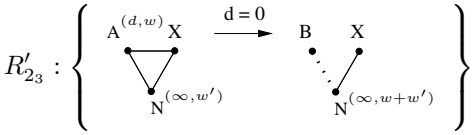
- R'_{2_1}: If the active vertex $v = (x, y+1)$ is on the top of $T_u = \{(x, y), (x, y+1), (x+1, y)\}$ with condition of the existence of $T_d = \{(x, y), (x+1, y), (x+1, y-1)\}$, then the neighbour $u = (x+1, y)$ recovers its weight.

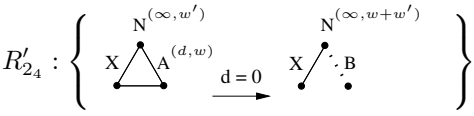


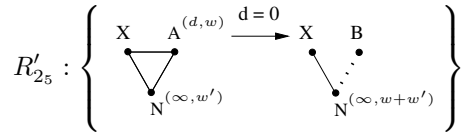
- R'_{2_2}: If the active vertex $v = (x, y)$ is the right-down vertex of the up triangle $T_u = \{(x, y), (x, y+1), (x+1, y)\}$, then its neighbour $u = (x, y-1)$ recovers its weight when v is removed.



- R'_{2_3}, R'_{2_4} and R'_{2_5}: Like the rules R'_{2_1} and R'_{2_2} the transmission of the weight pass through the diagonal indecent edge of the removed vertex.

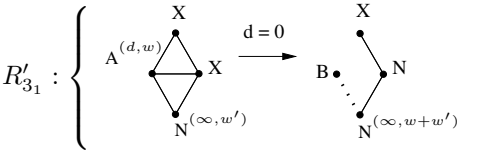




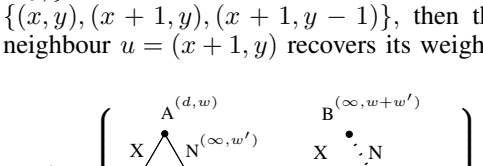


- R'_3: If the $deg(v) = 3$ then v disappears and the right-down neighbour gets its weight. In this case we distinguish the three following sub-rules.

  - R'_{3_1}: If the active vertex $v = (x, y)$ belongs to the two pairs of cells $\{(x, y), (x, y+1), (x+1, y)\}$ and $\{(x, y), (x+1, y), (x+1, y-1)\}$, then when its lifetime ends, its neighbour $u = (x+1, y-1)$ recovers its weight.



  - R'_{3_2}: If the active vertex $v = (x, y)$ belongs to the two pairs of cells $\{(x, y), (x+1, y), (x+1, y-1)\}$ and $\{(x, y), (x+1, y-1), (x, y-1)\}$, then its neighbour $u = (x+1, y-1)$ collects its weight when v is vanished.



- R'_4: If the active vertex $v = (x, y)$ belongs to the tree cells $(\{(x, y), (x, y+1), (x+1, y)\}, \{(x, y), (x+1, y), (x+1, y-1)\}, \{(x, y), (x+1, y-1), (x, y-1)\})$, then its neighbour $u = (x+1, y-1)$ recovers its weight at its disappearance.

C. Invariant proprieties

Our algorithm removes an active vertex once its lifetime expired. Thus, to ensure the continuity of the removal process, we must prove that the residual graph preserve the specific properties of TGG.

Proposition 4.1: Let $G = (V, E)$ be a TGG with size ≥ 2, v an active vertex in G. The graph $G' = (V \backslash \{v\},\ E \backslash \{\{v, u\}, \forall u \in V\})$ is a TGG.

Proof: Let $G = (V, E)$ be a TGG with size ≥ 2, and v an active vertex in G and let the graph $G' = (V \backslash \{v\},\ E \backslash \{\{v, u\}, \forall u \in V\})$. To prove the proposition, we must show that G' is a connected graph.

- If $deg(v) = 1$, then the removal of v and its incident edge in G doesn't introduce the disconnection of G' neither the creation of a hole in G'.

- If $deg(v) = 2$, $deg(v) = 3$ *or* $deg(v) = 4$, then let v, v_1, v_2 be a three vertices in the triangular grid graph G such as v is the active vertex whose lifetime has expired (in case of rules R_k and R'_k, $k = 2, 3, 4$). Consider the vertex $u \in V \backslash \{v, v_1, v_2\}$. then, if u is accessible to a node v_i; $1 \leq i \leq 2$, via a path passing through v, then when v is deleted, u will still accessible to v_i in another way by taking the vertices $v_{j \neq i}$; $j = 1, 2$.

■

V. Analysis of the Algorithm

A. Standard spanning tree

Let $G = (V, E)$ be a TGG and let F the set constituted by only the edges of E on which the weights of the vanishing vertices are transmitted. The set F can be built in advance with a distributed way as follows:

- If $e = \{(x, y), (x + 1, y - 1)\}$ is an edge in E then e belongs to F, i.e., any edge of the form $\{(x, y), (x + 1, y - 1)\}$ in E belongs to F.
- If $e = \{(x + 1, y - 1), (x + 1, y)\}$ belongs to E and to a single cycle γ of the form $\gamma = (x, y), (x, y + 1), (x + 1, y), (x + 1, y - 1)$. Then $e \in F$.

The graph **T**$= (V, F)$ connects all vertices of G and it is acyclic. Then it is a spanning tree.

Proposition 5.1: The graph **T**$= (V, F)$ as described above is a spanning tree of the triangular grid graph G.

Proof: We can prove this proposition by an inductive construction of **T** on G :

1) If $G = (\{(x, y)\}, \emptyset)$, the triangular grid graph consists of only one vertex, then the proposition is asserted **T**$= (\{(x, y)\}, \emptyset)$.

2) Let $G = (V, E)$ be a TGG and let **T**$= (V, F \subseteq E)$ be the spanning tree of G obtained by the above rules. Consider two adjacent vertices v and v' such as $v \in V$ and $v' \notin V$. According to the inductive rules seen in Section IV-B , the graph $G' = (V \cup \{v'\},\ E \cup \{\{v, v'\}\})$ is a TGG. So, it remains to prove that the tree **T'**$= (V \cup \{v'\},\ F \cup \{\{v, v'\}\})$ is the spanning tree of G'.
We can easily see that no cycle is created when the new edge $\{v, v'\}$ is added. Now let the tree $A = (V_A, F_A)$ where $V_A = \{v, v'\}$ and $F_A = \{\{v, v'\}\}$, then when we join the spanning tree **T** with the tree A the residual graph is acyclic. So it is a spanning tree of the triangular grid graph G'.

3) Let $G = (V, E)$ be TGG and let **T**$= (V, F \subseteq E)$ be its spanning tree. Suppose now that V contains three adjacent vertices $v_1 = (x, y)$ and $v_2 = (x + 1, y)$ and $v_3 = (x, y + 1)$ or else $v_1 = (x, y)$ and $v_2 = (x + 1, y)$ and $v_4 = (x + 1, y - 1)$ located in a cell such as two edges of the cell are in E and the third one, called e, is not. So according to inductive rules seen in section IV-B, the residual graph $G' = (V, E \cup \{e\})$, after the insertion of the new edge e, is a TGG.

   However, it remains to prove that the weight transmission occurs through the spanning tree **T'**$= (V, F')$ of G'.

   Let $C_1 = \{v_1, v_2, v_3\}$ or $C_2 = \{v_1, v_2, v_4\}$ be a cell of $G' = (V, E \cup \{e\})$ and let $e_1 = \{v_1, v_2\}$, $e_2 = \{v_2, v_3\}$, $e_3 = \{v_2, v_4\}$, $e_4 = \{v_1, v_3\}$, and $e_5 = \{v_1, v_4\}$.
   We have:

   - If $e = e_1$ or $e = e_3$ or $e = e_4$ then $F' = F$. (In this case the spanning tree does not change.)
   - If $e = e_2$ then $F' = F \backslash \{e_4\} \cup \{e_2\}$.
   - If $e = e_5$ then $F' = F \backslash \{e_3\} \cup \{e_5\}$.

   We can easily notice that the graph **T'** is a connected graph and, moreover, no cycle is generated when e is added. Thus, **T'** is a spanning tree of G.

■

Remark 5.1: The spanning tree constructed by these rules is unique.

Definition 5.1: The spanning tree **T**$= (V, F)$ is called standard spanning tree of the triangular grid graph G.

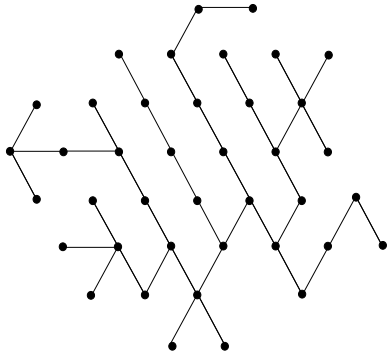


Figure 3. Standard spanning tree of the TGG given in Fig 2.

Proposition 5.2: Let $G = (V, E)$ be a TGG and **T**$= (V, F)$ its standard spanning tree. The vertex $v \in V$ is an active vertex in G iff it is a leaf in **T**.

Proof:

$\Longrightarrow$ Let the six vertices of a TGG $G = (V, E)$ defined as follows:
- $v_1 = (x, y)$
- $v_2 = (x, y + 1)$
- $v_3 = (x + 1, y)$
- $v_4 = (x + 1, y - 1)$
- $v_5 = (x + 1, y + 1)$
- $v_6 = (x, y + 2)$,

and let $C_1 = \{v_1, v_2, v_3\}$, $C_2 = \{v_1, v_3, v_4\}$, $C_3 = \{v_2, v_3, v_5\}$ and $C_4 = \{v_2, v_5, v_6\}$ four cells.
If v one active vertex of those vertices then we will show that v is a leaf in the spanning tree **T** of G.

- If $deg(v) = 1$, then certainly, v is a leaf in **T**.

- If $deg(v) = 2$, then we enumerate the following cases:

  (i) if v belongs to the cell C_1 and C_2 doesn't form a cell in G, then v is an extremity of the horizontal edge $\{v_1, v_3\}$, and since the weight transmission doesn't pass through this edge, thus, v is of degree 1 in **T** (i.e. it is a leaf).

  (ii) If v belongs to the cell C_2, then v is an end of the horizontal edge $\{v_1, v_3\}$, and since the transmission of weight does not pass through this edge, thus, v is a leaf in **T**.

  (iii) If $v = v_2$ and the cells C_1 and C_2 exist in G then v is one end of the edge $\{v_2, v_3\}$. While the weight transmission doesn't pass through the edge $\{v_1, v_2\}$, it becomes a leaf in **T**.

- If $deg(v) = 3$ and v belongs both to C_1 and C_2, then we have the bellow cases:

  (i) If $v = v_1$ then the weight transmission does not pass through the edge $\{v_1, v_2\}$ either $\{(v_1, v_3\}$. So v becomes a leaf in **T**.

  (ii) If $v = v_2$ then the weight transmission does not pass neither through the edges $\{v_1, v_2\}$ and $\{v_2, v_5\}$. So v becomes a leaf in **T**.

- If $deg(v) = 4$, and v belongs to the three cells C_1, C_3, and C_4 of G, then the weight transmission pass only through $\{v_2, v_3\}$. So v (equals v_2) becomes a leaf in **T**.

$\Longleftarrow$ Suppose now that v is a leaf in **T**=(V, F) and prove that v is an active vertex in G.

- If $deg(v) = 1$, then clearly v is an active vertex in G.

- If $deg(v) = 2$, then the two incident edges to v in G couldn't be in the same line, otherwise v is not a leaf in **T**. In the case where those edges are in a different orientations, only the edge $\{v_1, v_4\}$ or else the edge $\{v_2, v_3\}$ is in F, in addition, v is in the context of the rules R_2, then it is an active vertex.

- If $deg(v) = 3$, then with similar reasoning to the previous case, only one of the incident edges of v is in F. Thus, v is in the context of the rules R_3. So it is active.

- If $deg(v) = 4$ and the cells C_1, C_3, and C_4 are in G, then v is in the context of the rule R_4. Consequently, only the edge $\{v_2, v_3\}$ is in F, and according to the construction rules of F the vertex v is active.

■

B. Uniform election algorithm

Based on the results of the previous sections, we can summarize the distributed probabilistic election algorithm in triangular grid graph G as follows.

While G is not reduced to a single vertex **do**
- Each active vertex (rules R_0-R_4) generates its lifetime according to its weight.
- Once the lifetime of an active vertex has expired, it is removed with its incident edges and its neighbour in the standard spanning tree collects its weight.

The election algorithm in a TGG is seen as an election algorithm in its standard spanning tree. The Proposition 5.2 shows that each active vertex in a triangular grid graph is a leaf in its standard spanning tree, and the weights of this vertex in the both configurations are equals.

Let $G = (V, E)$ be a TGG. Initially, all vertices have the same weight 1: $w(v) = 1$, $\forall v \in V$. According to the rules introduced in Section IV-B, when an active vertex disappears, its *successor* collects its weight and adds it to its current weight. At the time t when a vertex v becomes active, its weight is the number of the vanishing vertices of its sides in the standard spanning tree. The lifetime $L(v)$ of a vertex v is a exponential random variable of parameter $\lambda(v)$ such that:

$$\lambda(v) = w(v) : Pr(L(v) \geq t) = e^{-\lambda(v)t}, \forall t \geq 0.$$

This property is equivalent to say that the probability of the disappearance of v in the time interval $[t, t + h]$ is $\lambda(v)h + o(h)$, when $h \to 0$ at each instant t, and this is independently of what happening elsewhere and what happened in the past. The random process is a variant of pure death process which is, in its turn, a special example of the Markov process in continuous time.

C. Election process

Probabilistic election can be mathematically modelled by a Markov process in continuous time. The initial state of the process is $G = (V, E)$ (the entire TGG). Let $\mathcal{S}_G$ be the set of all sub-triangular grid graphs of G and $G' \in \mathcal{S}_G$

We define **R** by:

R= $G' \cup (\{v\}, \{\{v, u\}\})$, u adjacent with v in **T**, i.e. the remove of the vertex v and all its incident edges from **R** leads to G'.

The transition probability from the triangular grid graph **R** to the G' is:

$$P_{(R,G')} = \frac{w(v)}{\sum_{u \ active \ in \ R} w(u)}$$

The following properties characterize the process of elimination in a TGG.

– The death rate of the triangular grid graph G is:

$$\lambda(G) = w(G) = \sum_{u \ active \ in \ G} w(u)$$

– The lifetime of G is: $L(G) = min_u\{L(u),\ u$ active in $G\}$ has the following distribution function:

$$Pr(L(G) \leq t) = 1 - Pr(L(G) \geq t) = 1 - e^{-\lambda(G)t},$$
$$\forall t \in \mathbb{R}^+$$

Proposition 5.3: Let G' be a TGG in $\mathcal{S}_G$, and let $P_{G'}(t)$ the probability that G' is the state of the election at time t. We have:

(i) $\frac{dP_G(t)}{dt} = -w(G)P_G(t)$,

(ii) for $G' \neq G$ of size ≥ 2,
$$\frac{dP_{G'}(t)}{dt} = -w(G')P_{G'}(t) + \sum_{v \ active \ in \ R} w(v)P_{\mathbf{R}}(t),$$

(iii) $\frac{dP_{(\{v\},\emptyset)}(t)}{dt} = \sum_{u \ adjacent \ to \ v \ in \ G} w(u)P_{(\{u,v\},\{\{v,u\}\})}(t)$

with the initial condition $P_G(0) = 1$.

Proof: Let G' is a sub-TGG of G and consider the evolution of the elimination process in the interval $[t, t+h]$. Let's calculate the probability of being in the state G' at time $t+h$.

– For $G' \neq G$ and G' is not reduced to a leaf, we have:
$$P_{G'}(t+h) = \sum_{v \ active \ in \ R} P_R(t)\pi_{R,G'}(h) + P_{G'}(t)\pi_{G',G'}(h) + o(h),$$

where $R = G' \cup \{v\}$ and $\pi_{R,G'}(h)$ is the probability of a direct transition from R to G' in a time interval of length h; the summation is performed for each vertex v adjacent to G'.

$$P_{G'}(t+h) = h\sum_v \lambda(v)P_R(t) + P_{G'}(t)[1-\lambda(G')] + o(h).$$

Therefore,
$$\frac{P_{G'}(t+h) - P_{G'}(t)}{h} = -\lambda(G')P_{G'}(t) + \sum_v \lambda(v)P_R(t) + \frac{o(h)}{h}.$$

This proves (ii).

– To prove (i), we remark that in the case of $P_G(t+h)$, the sum $\sum_v(\dots)$ disappeared from the right side, and since G has no predecessor graph. This established (i).

– To prove (iii), we just need to remark that the singleton state $(\{v\}, \emptyset)$ is absorbing, and, thus, $\pi_{\{v\},\{v\}}(h) = 1$. So, in $P_{(\{v\},\emptyset)}(t+h)$, the negative term disappeared. A simple computing gives (iii).

■

Proposition 5.4: The strategy described above leads to a totally fair election: in a TGG, all vertices have the same probability of being elected.

Proof: In [3] the authors give the prove of the uniform election in trees. In our work we have showed that there is a similarity of the election process over a TGG and over its standard spanning tree **T**. Using the similarity between the two structures, we can conclude, based on the results presented in [3], that for a triangular grid graph G of size n, the probability of being elected in G for any vertex $v \in G$ is $\frac{1}{n}$.

■

VI. Conclusion

In this paper, we proposed and analysed a probabilistic algorithm for uniform election in triangular grid graphs (TGG). We have introduce some rules to produce the family of those graphs.

Our algorithms use random delay associated to discovered vertices (active ones). These delays are an independent random variables and are locally generated when the vertices are discovered. To determine locally the active vertices we presented the activation rules. Also, we presented the weight transmissions rules for the successor of the vanishing vertex.

The election process is an elimination process that remove the actives vertices of the TGG until the graph is reduced to only one vertex, called leader. Using a single pass and a local computations, the elimination process is modelled by a pure death Markov process in a continuous time.

Finally, we showed that our algorithm is totally fair, since it gives the same probability to each vertex to be elected.

Our further work will be focussed on the study of the uniform election in the chordal graphs.

References

[1] G. L. Lann, "Distributed systems – toward a formal approach," in *Proceedings of the IFIP Congress 77*, 1977, pp. 155–160.

[2] Y. Métivier and N. Saheb, "Probabilistic analysis of an election algorithm in a tree," in *Colloquium on trees in algebra and programming*, ser. Lecture Notes in Comput. Sci., vol. 787. Spinger-Verlag, 1994, pp. 234–246.

[3] Y. Métivier, N. Saheb, and A. Zemmari, "A uniform randomized election in trees (extended abstract)," in *Proceedings of The 10th International Colloquium on Structural Information and Communication Complexity (SIROCCO 10)*. Carleton university press, 2003, pp. 259–274.

[4] A. E. HIBAOUI, N. SAHEB, and A. ZEMMARI, "A uniform probabilistic election algorithm in $k-$trees," *IMACS : 17th IMACS World Congress : Scientific Computation, Applied Mathematics and Simulation*, Jully 2005.

[5] A. E. Hibaoui, J. M. Robson, N. Saheb-Djahromi, and A. Zemmari, "Uniform election in trees and polyominoids," *Discrete Appl. Math.*, vol. 158, no. 9, pp. 981–987, May 2010.

[6] V. S. Gordon, Y. L. Orlovich, and F. Werner, "Hamiltonian properties of triangular grid graphs," *Discrete Mathematics*, vol. 308, pp. 6166–6188, 2008.

[7] M. Benantar, U. Dogrusoz, J. Flaherty, and N. S. Krishnamoorthy, "Triangle graphs," *Applied Numerical Mathematics*, vol. 17, pp. 85–96, 1995.

[8] I. Litovsky, Y. Métivier, and E. Sopena, "Different local controls for graph relabelling systems," *Math. Syst. Theory*, vol. 28, pp. 41–65, 1995.

[9] R. Sedgewick, *Algorithms in C++*, 1st ed. Addison-Wesley Co., 1992.

[10] I. Litovsky, Y. Métivier, and E. Sopena, "Graph relabelling systems and distributed algorithms," in *Handbook of graph grammars and computing by graph transformation*, H. Ehrig, H. Kreowski, U. Montanari, and G. Rozenberg, Eds. World Scientific, 1999, vol. 3, pp. 1–56.

[11] A. Sellami, "Des calculs lacaux aux algorithmes distribués," Ph.D. dissertation, Université Bordeaux I, 2004.

[12] J. Chalopin, Y. Métivier, and W. Zielonka, "Election, naming and cellular edge local computations," in *Proc. of International conference on graph transformation*, vol. 3256, ICGT'04, LNCS, 2004, pp. 242–256.

[13] I. Litovsky, Y. Métevier, and E. Sopena, "Definition and comparison of local computations on graphs and networks," in *MFCS'92*, ser. Lecture Notes in Comput. Sci., vol. 629, 1992, pp. 364–373.

[14] E. Godard, "Réécritures de graphes et algorithmique distribuée," Ph.D. dissertation, Université Bordeaux I, 2002.