On the Solvability of Anonymous Partial Grids Exploration by Mobile Robots

Roberto Baldoni, François Bonnet, Alessia Milani, Michel Raynal

I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1892

ON THE SOLVABILITY OF ANONYMOUS PARTIAL GRIDS EXPLORATION BY MOBILE ROBOTS

R. BALDONI F. BONNET A. MILANI M. RAYNAL

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

On the Solvability of Anonymous Partial Grids Exploration by Mobile Robots

R. Baldoni* F. Bonnet** A. Milani*** M. Raynal****

Systèmes communicants
Projet ASAP

Publication interne n ° 1892 — Mai 2008 — 19 pages

Abstract: Given an arbitrary partial anonymous grid (a finite grid with possibly missing vertices or edges), this paper focuses on the exploration of such a grid by a set of mobile anonymous agents (called robots). Assuming that the robots can move synchronously, but cannot communicate with each other, the aim is to design an algorithm executed by each robot that allows, as many robots as possible (let k be this maximal number), to visit infinitely often all the vertices of the grid, in such a way that no vertex hosts more than one robot at a time, and each edge is traversed by at most one robot at a time.

The paper addresses this problem by considering a central parameter, denoted ρ, that captures the view of each robot. More precisely, it is assumed that each robot sees the part of the grid (and its current occupation by other robots, if any) centered at the vertex it currently occupies and delimited by the radius ρ. Based on such a radius notion, the paper investigates the following cases and presents associated results. It first shows that there is no solution (i.e., $k = 0$) when $\rho = 0$. It then shows that $k \leq p - q$ is a necessary and sufficient requirement when $\rho = +\infty$, where p is the number of vertices of the grid, and q a parameter whose value depends on the actual topology of the partial grid. The paper finally analyzes the case $\rho = 1$ showing that it is the borderline from which the considered problem can be solved.

Key-words: Anonymity, Grid exploration, Partial grid, Mobile agent, Mutual exclusion, Robot, Synchronous system.

(Résumé : tsvp)

* Università di Roma "La Sapienza", Italy, baldoni@dis.uniroma1.it
** IRISA, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes Cedex, France Francois.Bonnet@irisa.fr
*** Università di Roma "La Sapienza", Italy, Alessia.Milani@dis.uniroma1.it
**** IRISA, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes Cedex, France raynal@irisa.fr

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Exploration avec contraintes de graphes par des robots

Résumé : Ce rapport présente (1) des bornes sur le nombre maximal de robots lorsque ceux-ci doivent visiter infiniment souvent chaque sommet d'un graphe connexe sous les contraintes d'exclusion mutuelle sur les arcs et les sommets du graphe, ainsi que (2) des algorithmes réalisant ces bornes.

Mots clés : Algorithme distribué, collision, contrainte d'exclusion mutuelle, robot, système synchrone.

1 Introduction

Graph exploration by robots The graph exploration problem consists in making one or several mobile entities visit each vertex of a connected graph. The mobile entities are sometimes called agents or robots (in the following we use the word "robot"). The exploration is perpetual if the robots have to revisit forever each vertex of the graph. Perpetual exploration is required when robots have to move to gather continuously evolving information or to look for dynamic resources (resources whose location changes with time). If nodes and edges have unique labels, the exploration is relatively easy to achieve.

The graph exploration problem becomes more challenging when the graph is anonymous (i.e., the vertices, the edges, or both have no label). In such a context, several bounds have been stated. They concern the total duration needed to complete a visit of the nodes (e.g. [6, 11, 13]), or the size of the memory of the robot necessary to explore a graph (e.g., it is proved in [8] that a robot needs $O(D \log d)$ bits of local memory in order to explore any graph of diameter D and maximum degree d). Impossibility results for one or more robots with bounded memory (computationally speaking, a robot is then a finite state automaton) to explore all graphs have been stated in [5, 15]. The major part of the results on graph exploration consider that the exploration is made by a single robot. Only very recently, the exploration of a graph by several robots has received attention also from a practical side [10]. This is motivated by research for more efficient graph explorations, fault-tolerance, or the need to overcome impossibilities due to the limited capabilities of a single robot.

The constrained exploration problem Considering the case where the graph is an anonymous partial grid (the grid is connected but has missing vertices/edges), and where the robots can move synchronously but cannot communicate with each other, the paper considers the following instance of the graph exploration problem, denoted the *Constrained Perpetual Graph Exploration* problem ($CPGE$). This problem consists in designing an algorithm executed by each robot that (1) allows as many robots as possible (let k be this maximal number), (2) to visit infinitely often all the vertices of the grid, in such a way that the following mutual exclusion constraints are always satisfied: no vertex hosts more than one robot at a time, and each edge is traversed by at most one robot at a time. These constraints are intended to abstract the problem of collision that robots may incur when moving within a short distance from each other or the necessity for the robots to access resources in mutual exclusion (This mutual exclusion constraint has been considered in [12] in a robot movement problem in a grid).

Results exposed in the paper rest on three parameters, denoted p, q and ρ. The first parameter p is related to the size of the grid, namely, it is the number of vertices of the partial connected grid. The second parameter q is related to the structure of the partial grid. This parameter is defined from a *mobility tree* (a new notion introduced in the paper) that can be associated with each partial grid. So, each pair (p, q) represents a subset of all possible partial grids with p vertices. Finally, the third parameter ρ is not related to the grid, but captures the power of the robots when we consider the part of the grid they can see. More precisely, a robot sees the part of the grid centered at its current position and covered by a radius ρ. From an operational point of view, the radius notion allows the robots that are at most ρ apart one from the other to synchronize their moves without violating the vertex and edge mutual exclusion constraints.

The paper analyzes the solvability of the $CPGE$ problem with respect to the number of robots k that can be placed in a partial grid characterized by the pair p and q when considering a given ρ for robots. The paper shows the following results.

- Case $\rho = 0$. In that case, the $CPGE$ problem cannot be solved (i.e., we have $k = 0$) for any grid such that $p > 1$ (a grid with more than one vertex). This means that, whatever the grid, if the robots cannot benefit from some view of the grid, there is no algorithm run by robots that can solve the $CPGE$ problem.
- Case $\rho = +\infty$. In that case, $k \leq p - q$ is a necessary and sufficient requirement for solving the $CPGE$ problem. Let us observe that $\rho = +\infty$ means that the robots knows the structure of the grid and the current position of the robots on that grid. (The initial anonymity assumption of the vertices and the robots can then easily be overcome.)
- Case $\rho = 1$. In that case, assuming a grid with more than one vertex, $k \leq p - 1$ when $q = 0$, and $k \leq p - q$ otherwise, are necessary and sufficient requirements for solving the $CPGE$ problem.

Finally, the paper discusses issues related to solvability of the $CPGE$ problem when $1 < \rho < +\infty$. It is important to notice that the previous investigations show that $\rho = 1$ is a critical radius value as it defines the fundamental demarcation line for the solvability of the $CPGE$ problem.

Roadmap The paper is made up of 7 sections. Section 2 presents related works. Section 3 first presents the computation model, and defines formally the $CPGE$ problem. Then, Section 4, 5 and 6 address the cases $\rho = 0$, and $\rho = +\infty$, $\rho = 1$, respectively. Section 7 concludes the paper by discussing the case $1 < \rho < +\infty$.

2 Related work

On the initial assumptions As already indicated, graph exploration is the process by which each vertex of a graph is visited by some entity. A great research effort on graph exploration by robots has been done on the minimal assumptions (in terms of robots/network requirements) required to explore a graph (e.g., [4, 9]). Some works focus on robots endowed with a finite persistent storage and direct communication with other robots (e.g., [2]). Some works assume that each robot has the capability to see where other robots are currently placed (e.g., [7]). Some other works study how the knowledge of the map (graph) by the robots reduces the complexity of the exploration (e.g., [14]).

On the type of graph exploration Graph exploration is mainly divided into *perpetual* graph exploration (e.g., [6]) where the robots have to travel the graph infinitely often, and graph exploration *with stop* (e.g., [9]) where each robot has to eventually stop after having explored the graph. Some papers focus on the exploration of the graph by a single robot (e.g., [1, 2, 8]). Cooperative graph exploration by a team of mobile robots has also received attention (e.g., [4, 7, 12]). As an example of multi-robot exploration, it is shown in [7] that the minimum number of robots required to solve the exploration with stop of a ring of size n is $O(\log n)$ when the exploration is done by oblivious anonymous robots that move asynchronously.

Lower bounds have been established for the perpetual graph exploration problem (e.g., [6, 11]). These bounds concern the period necessary for a robot to complete the visit of the graph, assuming constraints either on the robots, or on the graph. Differently, the upper bound introduced in the Section 3.3 concerns the maximum number of robots that can visit the graph infinitely often without ever colliding.

Constrained graph exploration The $CPGE$ problem defined in Section 3.2 is the perpetual graph exploration problem augmented with the mutual exclusion property on the vertices and the edges of the graph. These mutual exclusion constraints have been already stated and used [12] where the graphs considered are grids. The problem addressed in that paper is different from $CPGE$. More precisely, in [12], each robot has to visit some target vertices of the grid, and any two distinct robots have different targets. That paper establishes a lower bound on the time (number of rounds in a synchronous system) necessary to solve that problem and presents an optimal algorithm.

The problem of robots collision in also addressed in [17], where is proposed a collision prevention algorithm for robots moving on the plane.

3 Computation model, Problem specification and Mobility tree

3.1 Computation model

The grid The underlying grid is made up of a finite set of vertices, each vertex being connected to at least one and at most four other vertices according to the classical grid pattern. If two vertices are connected, we say there is an edge connecting them. The grid is anonymous in the sense no vertex has an identity. Moreover, there is a global sense of direction present in the grid: each vertex is able to distinguish its north, east, south and west neighbors The grid is represented as graph $G = (S, E)$ with $|S| = p$. An example of a partial grid (with $p = 25$ vertices) is depicted in Figure 1.

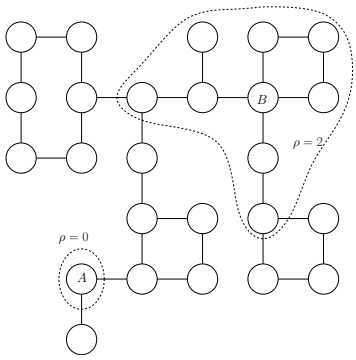


Figure 1: Example of a partial finite grid

The robots A mobile agent (robot) is an automaton whose computational power is a Turing machine. The moves of a robot are defined by the algorithm it executes. All the robots execute the same algorithm. It is assumed that local computation takes no time.

The finite set of robots R is such that $|R| \leq |S|$. The robots are anonymous. The robots have a common clock, and the time is divided into synchronous rounds [16]. At each round a robot can move from the vertex where it currently stays to a neighbor vertex, or stays at the same vertex. A move of a robot from a vertex to a neighbor vertex is done in one round.

Notation 1 *Given a robot a and a round r, $V(a,r)$ denotes the (single) vertex where the robot a is located at the beginning of round r.*

Radius The radius an algorithm is instantiated with is a non-negative integer ρ that provides each robot with the following information.

Let us first consider the case $\rho \neq 0$. At the beginning of any round r, $\forall a \in R$, the robot a, that is currently located at the vertex $V(r,a)$, sees the sub-grid centered at $V(r,a)$, including the vertices whose distance to $V(r,a)$ is at most ρ. It also sees whether these vertices are currently occupied by robots or not (i.e., for any such vertex v, whether the predicate $\exists\ x \in R :\ V(r,x) = v$ is true or false). An example of radius $\rho = 2$ is depicted in Figure 1: the robot located in the vertex denoted B knows the part of the grid surrounded by the corresponding dotted line (for the vertices at distance $\rho = 2$, it knows only their edges within distance $\rho = 2$).

With a light abuse of the previous notation of radius we consider the following definition for $\rho = 0$: a robot knows the edges of the vertex it is located in. An example is depicted in Figure 1: the robot in the vertex denoted A knows that this vertex has a east edge and a south edge. The fundamental difference with $\rho = 1$ lies in the fact that, when $\rho = 0$, the robot located in A cannot know whether the end vertices associated with these edges are occupied or not by robots.

More generally, the radius notion captures the possibility for robots to synchronize their moves when they are apart from each other at a distance $\leq \rho$.

3.2 The constrained perpetual grid exploration problem

The *Constrained Perpetual Grid Exploration* Problem ($CPGE$) can be formally defined by the following three properties.

- Perpetual Exploration. $\forall v \in S :\ \forall a \in R :\ \ \{r \mid V(a,r) = v\}$ is not finite.
  For any vertex v and any robot a, there are infinitely many rounds where a visits v.
- Vertex Mutual Exclusion. $\forall r \geq 0 :\ \forall (a,b) \in R \times R :\ \ (a \neq b) \quad \Rightarrow \quad V(a,r) \neq V(b,r)$.
  At the beginning of any round, no two robots are at the same vertex.

- Edge Mutual Exclusion.

$$\forall r \geq 0 : \forall (a,b) \in R \times R : \quad (a \neq b) \quad \Rightarrow \quad \big((V(a,r+1) = V(b,r)) \Rightarrow (V(b,r+1) \neq V(a,r))\big).$$

During a round, no two robots move on the same edge (i.e., they cannot exchange their positions).

This paper is on solving the $CPGE$ problem for as many robots as possible. More precisely, we are interested in finding the greatest number of robots and designing an algorithm A (executed by each robot) that solves the $CPGE$ problem whose precise definition appears in Section 3.4.

3.3 The mobility tree associated with a grid

The notion of mobility tree defined in this section is instrumental to extract from a grid a parameter q associated with the grid structure. This parameter contributes to state an upper bound (on the number of processes) beyond which the $CPGE$ problem cannot be solved. We originally expressed it for arbitrary undirected connected graph [3]. As we consider here that the map on which the robots move is an incomplete grid, the results exposed in [3] are still valid when we replace the word "graph" by the word "grid".

3.3.1 Preliminary definitions

A vertex v is a *leaf* of a graph $G = (S, E)$ if there is a single vertex v' such that $(v, v') \in E$. The degree d of a vertex v is the integer $\big|\{v' \mid (v, v') \in E\}\big|$. A *bridge* is an edge whose deletion disconnects the graph. A graph without bridge is a *bridgeless* graph. A path from a vertex v to a vertex v' is *simple* if no vertex appears on it more than once.

A graph $G' = (S', E')$ is a *subgraph* of a graph $G = (S, E)$ if $S' \subseteq S$ and $E' \subseteq E$. In that case, we also say that $G = (S, E)$ is a *supergraph* of $G' = (S', E')$. A *non-singleton* subgraph contains at least two vertices. The subgraph $G' = (S', E')$ is *induced* by the set of vertices S', if E' contains all the edges of E whose end-points are in S'. As, in the following, all the subgraphs we consider are induced subgraphs we omit the term "induced" to not overload the presentation.

A subgraph G' is *maximal* with respect to a property P if G' satisfies P, while none of its supergraphs satisfies P. So, "bridgeless" and "non-singleton" are properties that a (sub)graph satisfies or does not satisfy.

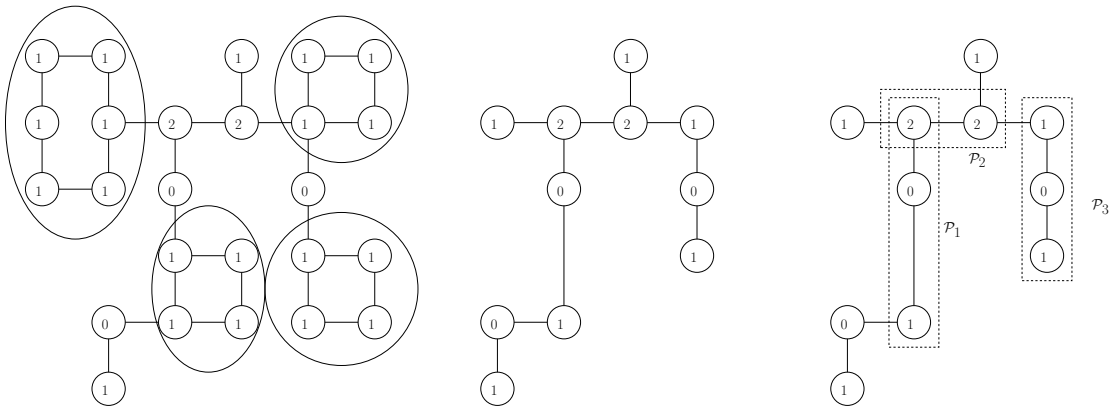
3.3.2 From a graph to a tree: the reduction procedure

Definition 1 *(Mobility Tree) Let the* labeled mobility tree *associated with a graph $G = (S, E)$ be the labeled tree $G' = (S', E')$ derived from G through the following reduction procedure:*

1. Initial labeling. *Each vertex $v \in G$ is first labeled as follows:*
   - *Label* 0*: if v does not belong to a bridgeless subgraph of G and its degree is two;*
   - *Label* 1*: if v is a leaf of G or belongs to a non-singleton bridgeless subgraph of G;*
   - *Label* 2*: otherwise.*
2. Compression. *Each maximal non-singleton bridgeless subgraph of G is reduced to a vertex with label* 1*.*

Figure 2 shows an example of the previous reduction procedure. The initial grid G is the grid depicted in Figure 1. The result of the initial labeling of its vertices is described in Figure 2(a). The non-singleton maximal bridgeless subgraphs of G are surrounded by a circle in that figure. Finally, the resulting labeled mobility tree obtained from the compression of the non-singleton maximal bridgeless subgraphs is shown in Figure 2(b).

The mobility tree of a graph G is intended to point out the noteworthy features of G as far the solvability of $CPGE$ is concerned. First, it points out those subgraphs of G (corresponding to vertices with label 1 in the mobility tree) where $CPGE$ could be solved in each of such subgraphs in isolation with a number of robots equal to the number of vertices of the subgraph. These subgraphs are indeed either leafs of G or non-singleton bridgeless subgraphs of G. Second, the mobility tree shows those paths of G that have to be traversed by a single robot at a time to move from one of the previous subgraphs of G to another one in order to extend the solvability of $CPGE$ in G. Let us therefore introduce the notion of Mutual Exclusion Path.



(a) Initial labeling and identification of subgraphs to be compressed

(b) Labeled mobility tree G'

(c) Three mutual exclusion paths

Figure 2: A graph, its labeled mobility tree, and exclusion paths

Definition 2 *(Mutual Exclusion Path) Let $\mathcal{P}$ be a path $(v, v^1), (v^1, v^2) \dots (v^m, v')$ of the mobility tree G' from vertex v to $v^{'}$. $\mathcal{P}$ is a mutual exclusion path of G' iff:*

- *The labels of v and $v^{'}$ are different from 0;*
- *If there are vertices v^h, $1 \leq h \leq m$, (i.e., the path from v to v' contains more than one edge), those vertices are labeled* 0.

As an example, Figure 2(c) shows three mutual exclusion paths, $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$, of the labeled mobility tree shown in Figure 2(b).

Definition 3 *(Length of a Mutual Exclusion path) In a labeled mobility tree $G' = (S', E')$, let the length of a Mutual Exclusion path between any two vertices v, $v' \in G'$ be the number of edges from v to v' augmented with j, where j is the number of vertices with label 2 in that path.*

The length of $\mathcal{P}_1$ depicted in Figure 2(c) is $2 + j = 3$ (as $j = 1$). The length of $\mathcal{P}_2$ is $1 + j = 3$ (as $j = 2$) while the length of $\mathcal{P}_3$ is $2 + 0 = 2$ (as $j = 0$). Intuitively, *the length of a mutual exclusion path* represents the minimum number of vertices that have to be initially empty (i.e., without robot assignment) in order for the robots to be able to solve the $CPGE$ problem *with respect to that path.* Therefore computing the maximal length of the mutual exclusion paths of a mobility three associated with a graph G becomes a key factor to compute the upper bound on the number of robots to keep $CPGE$ solvability in G.

Definition 4 *For any $p > 0$ and $q \geq 0$, let $\mathcal{G}(p, q)$ the set of graphs such that $\forall\ G \in \mathcal{G}(p, q)$: (1) G has p vertices, and (2) q is the maximal length of the mutual exclusion paths of the mobility tree associated with G.*

Two graphs belong to the same class $\mathcal{G}(p, q)$ if they both have the same number of vertices p and the same maximal length q of the mutual exclusion path of their respective mobility trees. The following theorem defines a bound on the number of robots beyond which $CPGE$ cannot be solved.

Theorem 1 [3] *Let G be a graph of the class $\mathcal{G}(p, q)$. There exists no algorithm solving the $CPGE$ problem for G when there are more than $k = p - q$ robots.*

Proof The proof is by contradiction. Let us assume that there is an algorithm A that solves $CPGE$ for a graph $G \in \mathcal{G}(p, q)$ and any initial configuration with at least $p - q + 1$ robots. There is no restriction on A: its computational power is the one of a Turing machine with an unbounded memory.

The contradiction for the case $q = 0$ is obvious: it is not possible to place $p + 1$ robots on a graph with p vertices without violating the vertex mutual exclusion property. So, the rest of the proof considers $q > 0$.

Let us observe that, due to the vertex mutual exclusion property, any configuration reachable from the initial configuration, contains $q - 1$ vertices without robots. According to definition of $\mathcal{G}(p, q)$, q is the maximal length of the mutual exclusion paths of the mobility tree of G. Let u be such a path, with X and Y being its end-point vertices. According to the labels of X and Y, three cases can be distinguished (but thanks to the labeled mobility graph abstraction, the reasoning is simple and identical in all cases).

- Both X and Y have label 1. This means that, if the path from X to Y in G contains more than one edge, its vertices different from X and Y have degree 2 and are labeled 0.

  Let G_X and G_Y be the the maximal subgraphs of G that include X and Y, respectively, and satisfy the following property P: any simple path in G from any $v \in G_X$ to any $v' \in G_Y$ includes both X and Y. (See an illustration in Figure 3.) As both X and Y have label 1, it follows from Definition 3 that u is a sequence of $q + 1$ vertices (and, as already observed, each vertex in this sequence different from X and Y has degree 2).

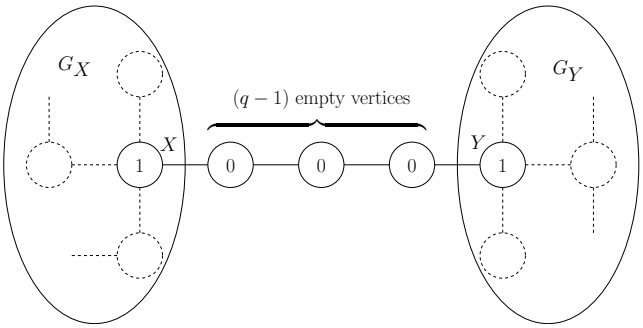


Figure 3: Mutual Exclusion path, from a label 1 to a label 1, of length $q = 4$

  Let us consider the initial configuration where the $q-1$ vertices without robots are exactly the vertices of $u \setminus \{X, Y\}$. Then, the $p - (q - 1)$ robots fill completely all the vertices of both subgraphs G_X and G_Y. As there are only $q - 1$ vertices without robots (the vertices of $u \setminus \{X, Y\}$), it follows that no robot in a vertex $v \in G_X$ can move to a vertex $v' \in G_Y$ (or vice-versa) without violating either the vertex mutual exclusion property or the edge mutual exclusion property, which proves the theorem for that case.

- X and Y have label 2 and 1, respectively. We conclude from Definition 3 and the labels of X and Y that the path u is a sequence of q vertices. Let $G1_X$, $G2_X$ (and possibly $G3_X$, $G4_X$, ...) denote the maximal subgraphs of G that do not contain X and such that any path in G from any vertex $v \in G1_X \cup G2_X \cup \cdots$ to any vertex $v' \in G_Y$ (where G_Y includes Y), includes both X and Y (see Figure 4).

  Similarly to the previous case, let us consider the configuration where the $q-1$ vertices without robots are the vertices of $u \setminus \{Y\}$. As previously, the $p - (q - 1)$ robots fill completely G_Y, $G1_X$ and $G2_X$ (and $G3_X$, $G4_X$, ... if they exist). As before, the chain from X to Y does not contain enough vertices without robots in order for a robot located at a vertex $v \in G1_X \cup G2_X \cup G3_X \cup \cdots$ to move to a vertex $v' \in G_Y$ (or vice-versa), without violating either the vertex mutual exclusion property or the edge mutual exclusion property. This proves the theorem for the second case.

- Both X and Y have label 2. Due to Definition 3, u is then a sequence of $q - 1$ vertices. Let $G1_X$, $G2_X$ (and possibly $G3_X$, $G4_X$, ...), and $G1_Y$, $G2_Y$ (and possibly $G3_Y$, $G4_Y$, ...) be the maximal subgraphs of G such that $X \notin G1_X \cup G2_X \cup \ldots$, $Y \notin G1_Y \cup G2_Y \cup \ldots$, and the maximal subgraphs of G such that any path in G from any $v \in G1_X \cup G2_X \cup \cdots$ to any $v' \in G1_Y \cup G2_Y \cup \cdots$ includes both X and Y (see Figure 5).

  Let us consider the configuration where the $q-1$ vertices without robots are all the vertices of u. The $p-(q-1)$ robots fill completely $G1_X$, $G2_X$, $G1_Y$ and $G2_Y$ (and $G3_X$, $G3_Y$, ... if they exist). It follows

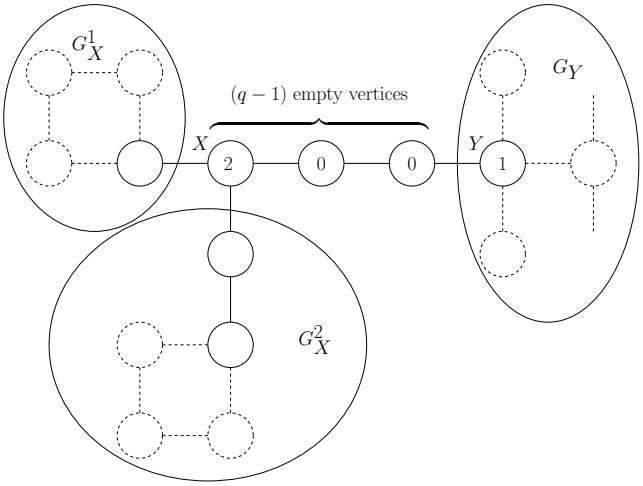


Figure 4: Mutual Exclusion path, from a label 1 to a label 2, of length $q = 4$

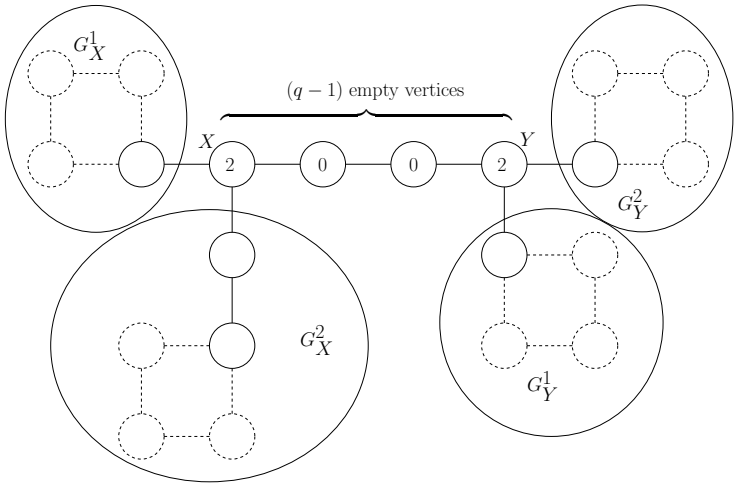


Figure 5: Mutual Exclusion path, from a label 2 to a label 2, of length $q = 5$

that, despite the fact that all the vertices of the path u have no robot, there are not enough vertices without robots to allow a robot in a vertex $v \in G1_X \cup G2_X \cup \dots$ to move to a vertex $v' \in G1_Y \cup G2_Y \cup \dots$, which proves the theorem for this last case.

$\square_{Theorem\ 1}$

3.4 f-Solving the $CPGE$ problem

Let A_ρ denote an algorithm instantiated with the radius value ρ, and let y, $1 \leq y \leq p$, denote the actual number of robots initially placed on a grid G of p vertices.

Definition 5 *Let f be a function from the set of classes $\mathcal{G}(p,q)$ to the set of non-negative integers, i.e., $f(p,q) \in \{0,1,\dots\}$, and y $(0 \leq y \leq p)$ the number of robots initially placed on the grid under consideration.*

Given such a function f, an algorithm A_ρ f-solves *the $CPGE$ problem if A_ρ (1) never violates the vertex and edge mutual exclusion properties (i.e., whatever the value of $y \in \{1\dots,p\}$), and (2) solves the $CPGE$ problem for any graph in the class $\mathcal{G}(p,q)$ and any number y of robots such that $0 \leq y \leq f(p,q)$.*

PI n ° 1892

Let us notice that Theorem 1 states that, whatever the value of ρ and the class $\mathcal{G}(p,q)$, there is no A_ρ algorithm that f-solves the $CPGE$ problem when $f(p,q) > p - q$.

As we can see the algorithms we are interested in always preserve the safety properties (here vertex and edge mutual exclusions) whatever the number y of robot, $0 \leq y \leq p$, i.e., even when $y > f(p,q)$. From a practical point view, always preserving the safety properties is crucial as it guarantees that, whatever their number, the robots never collide.

4 f-Solvability of the $CPGE$ problem when $\rho = 0$

This section shows that, when $\rho = 0$ (i.e., when a robot does not know whether an adjacent vertex is occupied or not by another robot), the $CPGE$ problem can be f-solved only in the very particular case where the grid is made up of a single vertex.

Theorem 2 *There is an algorithm A_0 that f-solves the $CPGE$ problem iff (1) $f(p,q) = 0$ when $(p,q) \neq (1,0)$, and (2) $f(1,0) \leq 1$ otherwise.*

Proof The direction $\Leftarrow$ follows directly from the value of $f(p,q)$. If the grid contains a single vertex ($p = 1$), we have $q = 0$, and the algorithm that never moves the single robot ($f(1,0) = 1$) from the unique vertex solves trivially the problem. So the main part of the proof consists in proving the direction $\Rightarrow$. The proof is by contradiction. Let A be an algorithm that f-solves the $CPGE$ problem for $f(p,q) > 0$ robots. We consider three cases.

- Case $f(1,0) \geq 2$. In that case, $p = 1$, i.e., the grid has a single vertex. It is trivially impossible to place more than one robot on a unique vertex without violating the vertex mutual exclusion property.
- Case where there is a pair $p > 0$ and $q > 0$ such that $f(p,q) > 0$. Let G be a grid in the class $\mathcal{G}(p,q)$. As A f-solves the $CPGE$ problem, it follows from Definition 5 that A (1) solves the $CPGE$ problem for y robots when $1 \leq y \leq f(p,q)$, and (2) preserves the vertex and edge mutual exclusion properties when $f(p,q) < y \leq p$.

  As A solves the $CPGE$ problem for $f(p,q) > 0$ robots, it solves it for one robot. Let us consider this case (a single robot) and let α be this robot. Due to the value of the radius A is instantiated with (namely, $\rho = 0$), the behavior of α cannot depend on the actual number of robots, which means that, be α the single robot in the grid, or be the grid filled with p robots, A imposes the same behavior to α.

  As $q > 0$, there is a at least one bridge in the grid. If the grid if full of robots, the traversal of this bridge by α violates the vertex mutual exclusion property, which proves that, in that case, A cannot f-solves the $CPGE$ problem.
- Case where there is a pair $p > 0$ and $q = 0$ such that $f(p,q) > 0$. Let G be a grid in the class $\mathcal{G}(p,q)$. As $f(p,q) > 0$, A solves $CPEG$ for one robot (say α) whatever its initial position in G. As in the previous case, as A is valid and $\rho = 0$, the algorithm executed by α is the same, be G filled with p robots or a single robot.

  Considering the case where G is filled with p robots, let us examine the first round r where a robot moves. As G is full of robots, it follows that more than one robot has to move during r (in order not to violate the vertex mutex exclusion property). More precisely, this can be obtained only if there is at least one cycle C of G on which the robots move are coordinated (in order they all synchronously move to the next vertex on the cycle).

  Let G' be G from which one vertex v of C is suppressed. Assuming an arbitrary orientation of C, let v_p (resp., v_s) be the vertex that is the predecessor (resp., successor) of v on the cycle C. As $\rho = 0$, when they execute the first r rounds, the robots located in $C \setminus \{v_p, v, v_s\}$ have the same behavior in G and G'. It follows that there is at least one cycle C' in G' on which the robots move are coordinated during round $r' \leq r$.

  Let G'' be G' from which one vertex v' of C' is suppressed. The same reasoning as before can be recursively done, at the end of which we obtain a grid G^ω without cycle, and with one robot per

vertex. Trivially, during the round r, no robot can move in G^ω without violating the vertex mutual exclusion property, which completes the proof of the theorem.

$\square_{Theorem\ 2}$

5 f-Solvability of the $CPGE$ problem when $\rho = +\infty$

This section shows that, when $\rho = +\infty$ (i.e., when, at each round, each robot sees the whole grid and its current occupation), $p - q$ is the maximal number of robots for which the $CPGE$ problem can be f-solved.

Theorem 3 *There is an algorithm $A_{+\infty}$ that f-solves the $CPGE$ problem iff $f(p, q) \leq p - q$.*

Proof The fact that there is no algorithm that f-solves the $CPGE$ problem when $f(p, q) > p - q$ follows from Theorem 1. So, let us consider the case where $f(p, q) \leq p - q$.

Let us first observe that, as $\rho = +\infty$, any algorithm A executed by the robots "sees" the whole grid and the current position of the robots. Moreover, as the notion of *north, east, south* and *west* is global (chirality notion), it is the same for all the robots when they "see" the grid. Consequently, A can always guarantee the vertex and edge mutual exclusion properties. To complete the proof an algorithm A that solves the $CPGE$ problem when $f(p, q) = p - q$ has to be designed.

```
operation explore_∞ ():
( 1)  Obtain a global view of the partial grid (thanks to ρ = +∞);
( 2)  Assign deterministically an id to each robot (from top-left to bottom-right);
( 3)  Compute a canonical directed cycle C including all the vertices;
( 4)  while true do
( 5)      for x from 1 to number_of_robots do % move the robot x along the cycle C %
( 6)          Determine the vertex v where the robot x currently is;
( 7)          Compute the rank i of the first occurrence of v in the cycle C;
( 8)          while (x has not reached the i − 1 vertex of C) do
( 9)              Compute the next edge e of C that the robot x has to traverse;
(10)              if e belongs to a cycle
(11)              then Compute a canonical directed elementary cycle C_e that includes e;
(12)                   All robots of C_e moves one step ahead on C_e;
(13)              else Move robots to obtain an empty vertex at the end of e;
(14)                   Move the robot x along e
(15)              end if
(16)          end while
(17)      end for
(18)  end while
```

Figure 6: An algorithm that f-solves the $CPGE$ problem for $\rho = \infty$

An algorithm that f-solves the $CPGE$ problem when $\rho = \infty$ The algorithm is described in Figure 6. Its underlying principles are the following ones. There is first an initialization part that benefits from $\rho = +\infty$, followed by a second part that deterministically (in order to prevent collisions) rules the moves of the robots.

As $\rho = +\infty$, each robot initially knows the whole grid and the initial position of each robot. Fed with that information, each robot can simulate an omniscient daemon that, at each round, coordinates the moves of all the robots, directing each of them either to move to another position or to stay at the same vertex, in such a way that each robot visits each vertex infinitely often.

Init part At line 01, the assignment of distinct identities to each robots (they are at most $f(p, q)$) can be done according to a top-down, left-right strategy. A similar deterministic strategy can be used to assign distinct identities (from 1 to p) to each vertex (line 02).

The "canonical directed cycle" C introduced at line 03 is defined as follows. Let the "canonical directed path" from vertex i to vertex j be the shortest (directed) path connecting i to j in the partial grid[1]. The

[1]If there are several shortest paths from i to j, one of them is selected according to a deterministic rule. This is required in order all the robots select the same path.

canonical directed cycle C starts at the vertex 1, and is made up of the concatenation of the following directed canonical paths: first the path from vertex 1 to vertex 2, then the path from vertex 2 to vertex 3, etc., then path from vertex $(p-1)$ to vertex p, and finally the from vertex p to vertex 1. (When the path terminating in y is concatenated to the path starting at y, the vertex y is considered only once.)

Coordinate move part This part is made up of an infinite loop (lines 04-18). It considers each robot x, one after the other (line 05), and directs it to traverse entirely the partial grid (lines 06-16).

First, the vertex v where the robot x currently stays is computed (line 06), and then the rank i of the first occurrence[2] of v in C is computed (line 07). Once this rank i has been computed, the aim is to adopt a greedy strategy that moves the robot x along the cycle C from i until $i-1$, thereby directing it to visit all the vertices of the partial grid G. According to the next oriented edge e of C that the robot x has to traverse, there are two cases to consider. Let v_1 be the start vertex of e and v_2 its end vertex.

- e belongs to a cycle (lines 11-12). In that case, the algorithm computes a directed cycle e belongs to[3]. Let C_e be this cycle. Let us notice that, as all the robots agree on C_e, they can synchronously coordinate their move in order x progresses along e from v_1 to v_2.
- e does not belong to a cycle (lines 13-14), so e is a bridge. Let us define three sets of vertices (denoted $Z1$, $Z2$ and $Z3$) as described in Figure 7[4]. We consider two cases.

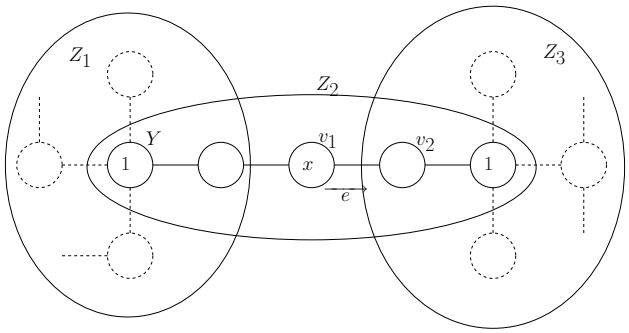


Figure 7: Configuration used in the description of the algorithm

  - There is an empty vertex in $Z3$. Let us consider a simple path from v_2 (end vertex of e) to an empty vertex of $Z3$ (as before, this vertex and the path are deterministically chosen). All the robots on this path synchronously coordinate their moves in order the vertex v_2 becomes empty. The robot x can then progress from v_1 to v_2.
  - There is no empty vertex in $Z3$. In that case, there are at least q empty vertices in the set $Z1$. Let us observe that the vertex v_1, on which the robot x is currently located, belongs to a chain of length at most q (this follows from the definition of the length of a mutual exclusion path in the mobility tree associated to G).
    * The robots in $Z1$ synchronously coordinate their moves to make empty the vertices in $Z1 \cap Z2$.
    * After these moves, the robot x moves to the vertex denoted Y in Figure 7 (end vertex of the chain x belongs to).
    * As Y is labeled 1, it belongs to a cycle γ and consequently, the robots on this cycle move in order the robot x does not remain in the vertex Y (in order not to "block" that vertex).
    * The robots in $Z1$ (without involving x) coordinate their moves in order all the vertices in the chain $Z2$, but Y, become empty.

[2] As v can appear several times in the canonical cycle C, it is sufficient to deterministically select one of its occurrence, e.g., its first occurrence in C.

[3] In case there are several cycles, one is deterministically selected.

[4] This figure considers a path from a vertex labeled 1 to a vertex labeled 1. It could easily be adapted if the labels of the end vertices of the path are in $\{1, 2\}$.

- Finally, The robots in the cycle γ (now including x) coordinate their move to move x to the vertex Y.
- Finally, the robot x moves along the empty vertices of $Z2$ from Y to v_2.

It follows that, according to the previous greedy strategy, the robots can synchronously coordinate their moves in order the robot x eventually moves from v_1 to v_2, i.e., progresses along the edge e.

After it has moved from v_1 to v_2, the algorithm directs x to move from v_2 to v'_2, the next edge e of the canonical cycle C (while loop, lines 08-16). When the robot x has visited all the vertices of C, the algorithm proceeds to the next robot, until all robots have traveled along C, after which the algorithm restarts from the beginning, etc. $\square_{Theorem\ 3}$

6 f-Solvability of the $CPGE$ problem when $\rho = 1$

This section considers the case where the (vision) radius of each robot is $\rho = 1$. The result of that section is the following main theorem.

Theorem 4 *There is an algorithm A_1 that f-solves the $CPGE$ problem iff $p = 1 \wedge f(1,0) \leq 1$, or $p > 1 \wedge f(p,0) \leq p-1$, or $q \neq 0 \wedge f(p,q) \leq p-q$.*

This theorem shows two noteworthy things. First, $\rho = 1$ is the borderline from which the $CPGE$ problem has a non-trivial solution. Second, it shows that, except for $q = 0$, the maximal number of robots for which it can be f-solved is the same as for $\rho = +\infty$, namely, $p - q$.

6.1 Two simple lemmas

Lemma 1 *When $p = 1$ (single vertex grid), the $CPGE$ problem can be f-solved iff $f(1,0) \leq 1$.*

Proof The proof follows from the fact that the grid is made up of a single vertex. $\square_{Lemma\ 1}$

Lemma 2 *There is no algorithm A_1 that f-solves the $CPGE$ problem when $p > 1 \wedge f(p,0) > p-1$ (then $q = 0$) or $q \neq 0 \wedge f(p,q) > p-q$.*

Proof For the case $q \neq 0$, the proof follows directly from Theorem 1. Let us now consider the case $q = 0 \wedge p > 1$. Then, $f(p,0) > p-1$ implies that each vertex is occupied by a robot. As $q = 0$, the robots have to move through along cycles to f-solve the $CPGE$ problem. Moreover, as $\rho = 1$ there is no possibility for a robot, without moving, to detect a cycle. It follows that there is no possibility of agreement among the robots to move synchronously along a cycle, which proves the lemma. $\square_{Lemma\ 2}$

So, Lemma 1 proves Theorem 4 for the case of the trivial grid (only one vertex), while Lemma 2 proves its "only if" direction. It remains to prove its "if" part when $p \neq 1$.

6.2 An algorithm that f-solves the $CPGE$ problem

This section presents an algorithm that f-solves the $CPGE$ problem when $p > 1 \wedge f(p,0) \leq p-1$, and when $q \neq 0 \wedge f(p,q) \leq p-q$.

The algorithm is based on the following two intuitions: (i) to let robots move while avoiding collision, a single vertex without robots (hole) is sufficient if at any given round all the robots that move, move towards a same direction and a robot moves only if the destination vertex is free (this requests $\rho \geq 1$); (ii) if it exist a round where every robot knows the map of the partial grid, the fact that robots are at most $p-q$ and where they are located; then, even without any vision, they can globally synchronize their moves to perpetually explore the grid.

Thus, the whole algorithm consists of the following three steps (precisely described in the following):

Step 1: Map Building. First, each robot runs an algorithm to know (1) the map (i.e., the whole structure of the partial grid, so; it then knows p, q and the grid diameter d) and (2) to attain a round in which each robot knows that all the robots know the map (Section 6.2.1). Each robot progressively learns the map (1) directly by moving and (2) by making deduction from observing the fact that either another robot in a given round reaches or leaves a vertex in its current neighborhood or it does not.

Step 2: Evaluate if there are at most $p-q$ robots. Once robots attain the round in which each robot knows that all the robots know the map, each robot runs an algorithm to learn whether there are more than $p-q$ robots. The algorithm implementing this step is described in Section 6.2.2 and at round $4d(p-q)$ every robot is able to evaluate if there are at most $p-q$ robots on the partial grid. If the number of robots bypasses $p-q$, a robot stops, which f-solves the $CPGE$ problem. Otherwise, the algorithm moves robots in the first R vertices of the partial grid (where R is the number of robots). This positioning takes $p \times 4d$ rounds.

Step 3: Perpetual Exploration. If the number of robots is $\leq p-q$, each robot supposes there are exactly $p-q$ robots in the system, completing the R real robots by "virtual" robots ($R \leq p-q$ is unknown). Thanks to the repositioning, each robot agrees on the current (real or virtual) robots location. Then, each robot can run the exploration algorithm $\mathsf{explore}_\infty()$ depicted in Figure 6 with $p-q$ robots starting from line (3) to ensure the perpetual exploration property.

6.2.1 Every robot learns the map of the grid

Context-sensitive moves For each given vertex $v \in S$, its context is a subset $C_v \subseteq \{north, west, south, east\}$ such that for each direction in C_v, v has a neighbor in such direction. We define a *context-sensitive move*, denoted cs-move, to be a move of a robot to a given vertex in its neighborhood (only if this latter is free), provided that the vertex where the robot is located has a given context.

For example, the cs-move requiring a robot to move east from a vertex with a east and south neighbors, is different from the cs-move requiring it to move east from a vertex with a east, south and west neighbors. There are 32 possible cs-moves: 4 from the vertices with one edge, 12 from the vertices with two edges, 12 from the vertices with three edges, and 4 from the vertex with four edges.

Lemma 3 *Consider a partial grid with p vertices and a number of robots $\leq p-1$, and radius $\rho = 1$. There is an algorithm and an integer k_{max} such that after k_{max} rounds, every robot knows (1) the map (i.e., the structure of the grid), (2) the value of k_{max}, and (3) the fact that each other robot knows both the map and k_{max}.*

An algorithm proving the lemma is described in Figure 8. The main idea behind this algorithm is that, for any $1 \leq k < k_{max}$, all robots incrementally perform all possible sequences of k *cs-moves*. k_{max} is the minimum value such that for any initial configuration with at most $p-1$ robots, every robot completely knows the grid after trying all the possible sequences of k_{max} cs-moves.

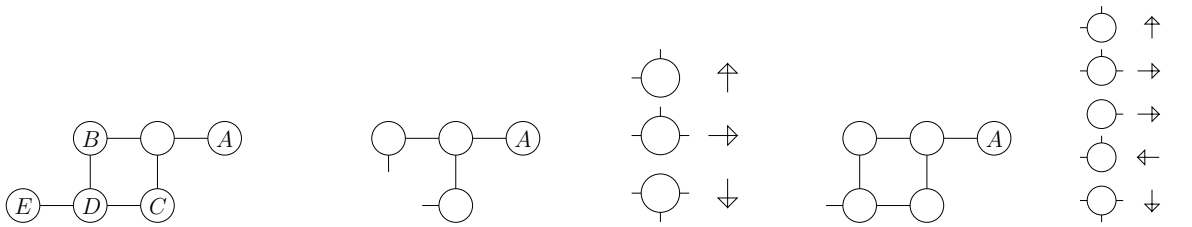
In particular, after executing a given sequence L (the same for all robots), robots return to their initial position by executing the sequence $\overline{L}$, i.e. L in the reverse order. It is possible to show that, after exhausting all the sequences of cs-moves of length k, a robot knows the part of the grid that is at most at distance $\lfloor \frac{k+3}{2} \rfloor$ from its initial position.

We explain here the underlying principles of the algorithm by presenting its behavior in a particular case, the one depicted in Figure 9(a) where 5 robots are located on a grid of 6 vertices. At the initial state, there is a robot A located in a leaf of the grid, and its only neighbor vertex is free. The goal is to prove that after exhausting all possible sequences composed of up to k_{max} cs-moves, A is able to deduce all the vertices of the grid.

Sequences composed of a single cs-move. Among these sequences there is one that will move the robot A to the west. A then discovers the context of the vertex adjacent to its initial position. It is a vertex where the only missing neighbor is the one at the north. After this, A comes back to its initial position. Among the sequences of length 1, there is also one that will move robot C to its north (and then back) and another that will move robot B to its east neighbor (and then back). After each of those moves, A sees a robot located in its west neighbor vertex. From this observation A can deduce the type of the initial vertex both of robot C and B. This is because all the robots execute the same algorithm, and thus A knows the cs-moves that

```
operation get_map1 ():
( 1)  k ← 1; k_max ← +∞;
( 2)  while (k < k_max)
( 3)        Try deterministically all possible moves composed of k cs-moves
                memorizing all the states;
( 4)        if (the graph can be entirely deduced)
( 5)        then Compute the parameters p and q of the graph;
( 6)             Compute k_max
( 7)        end if;
( 8)        k ← k + 1
( 9)  end while
```

Figure 8: The algorithm get_map$_1$()



(a) Initial state

(b) A's knowledge after all sequences of 1 *cs-moves*

(c) Sequence of cs-moves

(d) A's knowledge after 3 steps

(e) Sequence of cs-moves

Figure 9: A simple example

respectively forced C and B to move to its west neighbor. Figure 9(b) summarizes the knowledge of A after all sequences of a single cs-move.

Sequences composed of two cs-moves. During these sequences, A does not increase its knowledge of the map.

Sequences composed of three cs-moves. A is able to deduce the context of the vertex where D is initially located. Indeed, among all the sequences of three cs-moves, there is the one described in Figure 9(c). The execution of this sequence of cs-moves entails first a move of C to the north, then (during the next round) a move of D to the east, and finally (during the third and last round of the sequence) no move. During the last of those three rounds, only C is located in a vertex that has the context required to move, but it cannot move to the south because the corresponding vertex is occupied. This sequence of 3 cs-moves allows A to learn that a robot D has moved to the initial position of C. Otherwise, C would have moved to the south during the third move. Hence, A deduces the type of D's initial vertex. Figure 9(d) summarizes the knowledge of A after executing the sequences composed of 3 cs-moves. .

Sequences composed of four cs-moves. A does not increase its knowledge during these sequences of cs-moves.

Sequences composed of five cs-moves. A is able to deduce E's initial vertex when the algorithm will execute the sequence of cs-moves described in Figure 9(e). Indeed, in this sequence of cs-moves, C moves south, but is not able to go back north on the fifth move. This means that the robot E has blocked D which in turn has blocked C. After the moves composed of five cs-moves, A knows entirely the graph.

After discovering the graph. As soon as A knows the graph, it can simulate the behavior of all the robots in the system and consequently know their knowledge. More precisely, (1) A simulates the execution of the algorithm with any initial state of at most $p-1$ robots. Then (2), A computes the maximum value $k_{\max}$ needed for any robot in any configuration to know entirely the graph. (3) After testing all the sequences of moves composed of k_{max} cs-moves, A knows that all robots knows (i) the map of the partial grid, and also that (ii) each robot has computed the same k_{max}. The robots can then enter the second step of the algorithm as described in the next section.

6.2.2 Evaluating the predicate $\mathcal{P}_1 \equiv$ "are there at most $p-q$ robots?"

Once each robot knows that all the robots know the map of the partial grid, they can compute the value of the parameters p and q. Then, the robots have to evaluate the predicate $\mathcal{P}_1 \equiv$"are there at most $p-q$ robots?".

To this end, each robot executes the following sequence of operations, where d is the diameter of the grid and R is the actual number of robots:

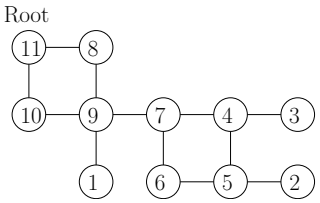
operation evaluating_$\mathcal{P}_1$ ():
(1) Deterministically assign an id to each vertex and define a tree;
(2) Move the robots to vertices whose ids goes from 1 to R (in $p \times 4d$ rounds);
(3) Evaluate the predicate $\mathcal{P}_1$.
(4) Move the robots to vertices whose ids goes from 1 to R (in $p \times 4d$ rounds);

evaluating_$\mathcal{P}_1$**: Line 01** Due to the global sense of direction (as defined in Section 3.1), the robots can agree on a same predetermined vertex of the partial grid (e.g., the more "north-west" vertex of the grid). Starting from that vertex, each robot can assign an identifier to each vertex using a *Depth First Search* (DFS) algorithm where the identifiers are assigned according to the *Post-Ordering* rule. (Note that this labeling, from 1 to p, is done locally without any move.)

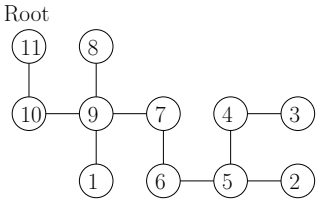
Due to the fact that the labeling is produced by a post-ordering DFS algorithm, it satisfies the following properties:

Property 1 *If all vertices labeled from* 1 *to* l *(with* $l < p$*) are suppressed from the grid, the remaining grid (made up of the vertices labeled* $l+1$ *to* p*) remains connected. (An example is depicted in Figure 10.)*

Property 2 *There is a tree rooted at* p *such that each vertex has for father his adjacent vertex with the smallest id among the id greater than itself. (For example, the vertex labeled* 4 *with adjacent vertices* 3*,* 5 *and* 7 *has the vertex* 5 *for father.)*



(a) Post Ordering



(b) Corresponding Tree

Figure 10: Example of labeling from a post-ordering DFS

evaluating_$\mathcal{P}_1$**: Line 02** During this step, the robots move[5] to reach the vertices with the lowest $|R|$ ids. Initially, each robot moves in order to try to reach the vertex 1. Each robot computes the shortest path from its current position to vertex 1 and then moves along this path.

In the following, we explain how each robot follows its path without colliding with the other robots. Each direction is associated with a value in the set $\{0, 1, 2, 3\}$. Let s be the direction where a robot has to move according to the next step of its path. Then this robot moves to its s neighbor at round r if (i) (r mod 4) $= i$ where i is the value associated to the direction s and (ii) the s neighbor vertex is free.

As a result, after at most $4d$ rounds, there is a robot that occupies vertex 1. Starting from the $4d + 1$ round, the remaining robots try now to reach the vertex 2. This process can continues until the *p-th* vertex thanks to the Property 1. Therefore, after $p \times 4d$ rounds, the R robots occupy the vertices of the grid labeled $1, 2, \ldots, R$. It is important to notice that, up to now, no robot knows the value of R.

evaluating_$\mathcal{P}_1$**: Line 03** If the graph belongs to a (p, q) class of graphs with $q = 0$, the evaluation of the predicate is easy: since all robots know the graph, they know this value of $q = 0$ and then they evaluate the predicate to true. The following text concerns then only cases when $q > 0$.

The main idea is that at the beginning of this step, there is a set of robots (possibly empty) that is able to trivially evaluate the predicate $\mathcal{P}_1$. Then these robots (if any) coordinate to communicate to the remaining robots the result of the evaluation. Remember that the only way for robots to communicate is to move and to observe the occupation of vertices.

At the beginning of this step, the robots occupy the vertices of the grid labeled $1, 2, \ldots, R$, if there are some robots on the vertices labeled from $p - q + 1$ to p, they can trivially compute the predicate to false: indeed since any robot of this set occupy a vertex whose label is greater than $p - q$, it means that there are more than $p - q$ robots in the grid, thus these robots immediately evaluate the predicate to false.

In $(p-q)$ phases (starting at phase 1) all the other robots in the grid will be able to evaluate the predicate. Each phase takes $4d$ synchronization rounds. In particular, in the i-th phase the only robot which evaluates $\mathcal{P}_1$ is the one (if any) located at the vertex $p - q - i + 1$ (notice that for the first phase it corresponds to the vertex $p - q$ which is the highest one that could not compute immediately the predicate). This latter evaluates the predicate $\mathcal{P}_1$ as false if at the end of the i-th phase, it observes that its father vertex (as defined in Property 2) is occupied by a robot; true otherwise (i.e. its father vertex is free at that point).

At the beginning of phase i, the robots (if any) that occupy the vertices labeled from p until $p-q-i+2$ knows if the predicate $\mathcal{P}_1$ is true or false, because they evaluated it during the previous phases (or initially). So, they coordinate to ensure that at the end of the i-th phase, the robot located on the $p-q-i+1$ vertex evaluates $\mathcal{P}_1$ correctly. In particular, if the predicate is false, by the end of the i-th phase these robots move to let one of them occupy the father of the vertex $p - q - i + 1$. On the contrary, if the predicate is true, they move in order to make the father of the vertex $p - q - i + 1$ empty.

At round $4d$ of the phase $i = (p-q)$, all the robots agree on the value of the predicate $\mathcal{P}_1$. If the predicate is true, then they can start the last step and run the exploration algorithm explore$_\infty$() defined in Section 5 to ensure the perpetual exploration property.

evaluating_$\mathcal{P}_1$**: Line 04** In order to execute (when $R \leq p - q$) the algorithm explore$_\infty$() starting from line 3 as remarked in step 3 of Section 6.2, each robot has to know the location of every other robots. This is why we repeat line 02 in order to place robots in the first R position and after $p \times 4d$ rounds every robot knows this positioning. However this repositioning is not enough since robots do not know the exact number R of robots, they just know $R \leq p - q$. The solution consists for each robot to suppose there are exactly $p - q$ robots and then the execution of the algorithm explore$_\infty$() is done with $(p - q) - R$ virtual robots that occupy vertices labeled from $R+1$ to $p-q$. This introduction of virtual robots ensure that each robot start the execution with the same configuration.

7 Conclusion

To conclude, the following table summarizes the results of the paper:

[5]Here the algorithm considers simple moves, not cs-moves.

Value of ρ	f-solvability of the $CPGE$ problem
$\rho = 0$	$f(1,0) \leq 1$ and $f(p,q) = 0$ otherwise
$\rho = 1$	$f(p,0) \leq p-1$ when $p > 1$, and $f(p,q) \leq p-q$ otherwise
$1 < \rho < +\infty$	$f(p,0) \leq p-1$ when $(p > 1 \wedge \mathcal{Q}_\rho)$, and $f(p,q) \leq p-q$ otherwise
$\rho = +\infty$	$f(p,q) \leq p-q$

It is easy to see that $f(p,q) \leq p-q$ is an upper bound on the number of robots in all cases (recall that $q = 0$ when $p = 1$). The case $\rho = 0$ requires that the grid be trivial (only one vertex), while there is no requirement on the structure of the grid for f-solving the $CPGE$ problem when $1 \leq \rho \leq +\infty$. It follows that $\rho = 1$ is a strong demarcation line delineating the cases from which f-solving the $CPGE$ problem becomes relevant.

The case $1 \leq \rho \leq +\infty$ shows that, the maximal number of robots for which one can f-solve the $CPGE$ problem, depends on the structure of the grid. This is captured by the parameter q derived from its structure (thanks to the notion of mobility tree).

When $1 \leq \rho < +\infty$ and $q \neq 0$, $(p-q)$-solving the $CPGE$ problem is always possible, and is optimal (in the number of robots). When $q = 0$, there are cases where the maximal number of robots for which the $CPGE$ problem can be f-solved is smaller than $p - q = p - 0$, namely it is $p - 1$. The paper has identified the corresponding grid structures when $\rho = 1$: they are all the non-trivial grids (more than one vertex). As far as the cases $1 < \rho < +\infty$ are concerned, we conjecture that $f(p,0) \leq p-1$ when $(p > 1 \wedge \mathcal{Q}_\rho)$ where $\mathcal{Q}_\rho$ is a property that depends on the cycles that can be seen within the radius ρ. Moreover, we also conjecture that, given a grid, this property is such that $\mathcal{Q}_1 = \mathit{true}$, $\mathcal{Q}_{+\infty} = \mathit{false}$, and $\forall \rho : \mathcal{Q}_{\rho+1} \Rightarrow \mathcal{Q}_\rho$.

References

[1] Albers S. and Henzinger M.R., Exploring Unknown Environments. *SIAM Journal on Computing*, 29(4):1164-1188, 2000.

[2] Awerbuch B., Betke M., Rivest R.L. and Singh M., Piecemeal Graph Exploration by a Mobile Robot. *Information and Computation*, 152(2):155-172,1999.

[3] Baldoni R., Bonnet F., Milani A. and Raynal M., Anonymous Graph Exploration without Collision by Mobile Robots. *Tech Report #1886*, 10 pages, IRISA, Université de Rennes 1, France, 2008. ftp.irisa.fr/techreports/2008/PI-1886.pdf

[4] Bender M.A. and Slonim D., The Power of Team Exploration: Two Robots can Learn Unlabeled Directed Graphs. *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS'94)*, IEEE Computer Press, pp. 75-85, 1974.

[5] Budach L., Automata and Labyrinth. *Mathematische Nachrichten 1948-1999*, 86(1):195-282, 1978.

[6] Dobrev S., Jansson J., Sadakane K. and Sung W.K., Finding Short Right-Hand-on-the-Wall Walks in Undirected Graphs. *Proc. of the 12th Colloquium on Structural Information and Communication Complexity (SIROCCO'05)*, Springer-Verlag, LNCS #3499, pp. 127-139, 2005.

[7] Flocchini P., Ilcinkas D., Pelc A. and and Santoro N., Computing Without Communicating: Ring Exploration by Asynchronous Oblivious Robots. *In Proc. 11th Int'l Conference On Principles Of Distributed Systems (OPODIS'07)*, Springer-Verlag, LNCS #4878, pp. 105-118, 2007.

[8] Fraigniaud P., Ilcinkas D., Peer G., Pelc A. and Peleg D.. Graph Exploration by a Finite Automaton. *Theoretical Computer Science*, 345(2-3):331-344, 2005.

[9] Fraigniaud P., Ilcinkas D., Rajsbaum S. and Tixeuil S., Space Lower Bounds for Graph Exploration via Reduced Automata. *In Proc. 12th Colloquium on Structural Information and Communication Complexity (SIROCCO'05)*, Springer-Verlag, LNCS #3499, pp. 140-154, 2005.

[10] Franchi A., Freda L., Oriolo G., Vendittelli M., A Randomized Strategy for Cooperative Robot Exploration. *Int'l Conference on Robotics and Automation (ICRA'07)*, IEEE press, pp. 768-774, 2007.

[11] Gasieniec L., Klasing R., Martin R.A. Navarra A. and Zhang X., Fast Periodic Graph Exploration with Constant Memory. *In Proc. 14th Colloquium on Structural Information and Communication Complexity (SIROCCO 2007)*, Springer-Verlag, LNCS #4474, pp. 26-40, 2007.

[12] Grossi R., Pietracaprina A. and Pucci G., Optimal Deterministic Protocols for Mobile Robots on a Grid. *Information and Computation*, 173(2):132-142, 2002.

[13] Ilcinkas D., Setting Port Numbers for Fast Graph Exploration. *In Proc. 13th Colloquium on Structural Inf. and Communication Complexity (SIROCCO'06)*, Springer-Verlag, LNCS #4056, pp. 59-69, 2006.

[14] Panaite P. and Pelc A., Impact of Topographic Information on Graph Exploration Efficiency. *Networks*, 36(2):96-103, 2000.

[15] Rollik H.A., Automaten in Planaren Graphen. *Acta Informatica*, 13:287-298, 1980.

[16] Suzuki I. and Yamashita M., Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. *SIAM Journal on Computing*, 28(4):1347-1363, 1999.

[17] Yared R., Defago X. and Wiesmann M., Collision Prevention Using Group Communication for Asynchronous Cooperative Mobile Robots. *In Proc. 21st Int'l IEEE Conference on Advanced Information Networking and Applications (AINA 2007)*, IEEE Computer Press, pp. 244-249, 2007.