Exploration de réseaux par un robot (Network exploration with a robot)

P. Fraigniaud[1], D. Ilcinkas[1], G. Peer[2], A. Pelc[3] et D. Peleg[2]

[1] *CNRS, LRI, Université Paris-Sud, France*
[2] *Dept. of Computer Science, Weizmann Institute, Israel*
[3] *Dép. d'informatique, Univ. du Québec en Outaouais, Canada*

Résumé. Cet article traite de l'exploration d'un réseau par un agent mobile, ou *robot*, modélisé par un automate fini. Le réseau est anonyme, au sens que les nœuds ne disposent pas nécessairement d'un étiquetage global. Le robot ne dispose pas de connaissances préalables sur la topologie du réseau, ni même sur sa taille. Sa tâche consiste à traverser chacun des liens du réseau. Nous montrons que, pour tout robot à K états, et pour tout $d \geq 3$, il existe un réseau de degré maximum d et d'au plus $K+1$ nœuds que le robot ne réussit pas à explorer. Cette borne $K+1$ améliore toutes les bornes connues jusqu'aujourd'hui. Nous montrons également que, pour explorer tous les graphes de diamètre D et de degré maximum d, un robot doit disposer de $\Omega(D \log d)$ bits de mémoire, et ce même si l'on restreint l'exploration aux réseaux planaires. Enfin, nous montrons que cette borne est asymptotiquement optimale, en décrivant un algorithme qui permet à un robot possédant une mémoire de $O(D \log d)$ bits d'explorer tous les réseaux de diamètre D et de degré maximum d. Ceci nous permet donc de conclure que la complexité mémoire du problème de l'exploration de réseaux est $\Theta(D \log d)$ bits.

Abstract. A finite automaton, simply referred to as a *robot*, has to explore a graph whose nodes are unlabeled and whose edge ports are locally labeled at each node. The robot has no a priori knowledge of the topology of the graph or of its size. Its task is to traverse all the edges of the graph. We first show that, for any K-state robot and any $d \geq 3$, there exists a planar graph of maximum degree d with at most $K+1$ nodes that the robot cannot explore. This bound improves all previous bounds in the literature. Moreover, we show that in order to explore all graphs of diameter D and maximum degree d, a robot needs $\Omega(D \log d)$ memory bits, even if we restrict the exploration to planar graphs. Finally, we describe an algorithm that meets this latter bound, i.e., an algorithm that allows a robot to explore any graph of diameter D and maximum degree d using a memory of size $O(D \log d)$ bits. We thus prove that the worst case space complexity of graph exploration is $\Theta(D \log d)$ bits.

Keywords: exploration, labyrinthe, automate fini, agent mobile, robot.

1 Background and motivation

A mobile entity, e.g., a software agent or a robot, has to *explore* an undirected graph by visiting all its nodes and traversing all its edges, without any a priori knowledge of the topology of the graph or of its size (see, e.g., [1, 2, 3, 7, 8, 9, 10, 14]). The task of visiting all nodes is fundamental in searching for data stored at unknown nodes of a network, and traversing all edges is often required in network maintenance and when looking for defective components. More precisely, we consider the task of "perpetual" exploration in which the robot has to traverse all edges of the graph but is not required to stop. That is, the robot moves from node to node, traversing edges, so that eventually all edges have been traversed. Perpetual exploration is of practical interest, e.g., if regular control of a network for the presence of faults is required, and all edges must be periodically traversed over long periods of time.

If nodes and edges have unique labels, exploration can be easily achieved (e.g., by depth-first search). However, in some navigation problems in unknown environments, such unique labeling may not be available, or limited sensory capabilities of the robot may prevent it from perceiving such labels. Hence it is important to be able to program the robot to explore *anonymous* graphs, i.e., graphs without unique labeling of nodes or edges. Clearly, the robot has to be able to locally distinguish ports at a node: otherwise it is impossible to explore even the star with 3 leaves (after visiting the second leaf, the robot cannot distinguish the port leading to the first visited leaf from that leading to the unvisited one). Hence we make a natural assumption that all ports at a node are locally labeled $1,\ldots,d$, where d is the degree of the node. No consistency between those local labelings is assumed.

In many applications, robots and mobile agents are meant to be simple, often small and inexpensive devices. This limits the amount of memory with which they can be equipped. As opposed to numerous papers that imposed no restrictions on the memory of the robot and sought exploration algorithms minimizing time, i.e., the number of edge traversals, we investigate the minimum memory size of the robot that allows exploration of graphs of given (unknown) size, regardless of the time of exploration. That is, we want to find an algorithm for a robot performing exploration, using as little memory as possible.

A robot with a k-bit memory is modeled as a finite automaton. The first known finite automaton algorithm designed for graph exploration was introduced by Shannon [17] in 1951. Since then several papers have been dedicated to the graph exploration problem (cf. [11] and the references therein). In particular, in 1967, during his talk at Berkeley, Rabin [15] presented a proof that no finite automaton with a finite number of pebbles can explore all graphs (a pebble is a marker that can be dropped at and removed from nodes). In 1971, Müller [13] gave some formal arguments to support Rabin's claim, in the restricted case of a robot without pebble. In 1977, Coy [6] presented another proof, but some parts of it are fuzzy. The first formal proof of Rabin's claim is generally attributed to Budach [5], in 1978, for a robot without pebble. Actually, the long and technical paper by Budach is concerned with labyrinths. A *labyrinth* is a two-dimensional obstructed chess-board (i.e., $\mathbb{Z}^2$ with forbidden cells). The forbidden cells in $\mathbb{Z}^2$ are described by a set L. If L (resp., $\mathbb{Z}^2 \setminus L$) is finite, then the labyrinth is called finite (resp., co-finite). Exploring a finite labyrinth means that the automaton is able to go arbitrarily far away from its starting position, for any starting position. The edges of the labyrinth are consistently labeled North, South, East, West. (Budach's result applies also to graphs because a co-finite labyrinth is a finite graph.) The same year, Blum and Kozen [4] improved Budach's result by proving that three finite automata cannot cooperatively perform exploration of all graphs. In 1979, Kozen [12] proved that four cooperative robots cannot explore all graphs. Finally, in 1980, Rollik [16] gave a complete proof of Rabin's claim. More precisely, Rollik proved that no finite set of finite automata can cooperatively perform exploration of all cubic planar graphs. Since a finite automaton is more powerful than a pebble, Rabin's claim is a corollary of Rollik's theorem.

In all proofs, including the one by Budach and the one by Rollik, the size of the smallest *trap* for an automaton with no pebble (i.e., the smallest graph that an automaton with no pebble cannot explore) is large. One of the objectives of the current paper is to revisit Rabin's claim in the case of a robot with no pebble, specifically for improving the size of traps, and for designing traps with specific topological properties.

2 Terminology and model

An anonymous undirected graph with locally labeled ports is a graph whose nodes are unlabeled and where the edges incident to a node v have distinct labels $1,\ldots,d_v$, where d_v is the degree of v. Thus every undirected edge $\{u,v\}$ has two labels which are called its *port numbers* at u and at v. Port numbering is *local*, i.e., there is no relation between port numbers at u and at v. Unless specified otherwise, all considered graphs are supposed to be connected.

We are given a mobile entity traveling in an anonymous graph with locally labeled ports. The graph and its size are a priori unknown to the entity. The mobile entity is referred to as a *robot*. More precisely, a K-state robot is a finite Moore automaton $\mathcal{R} = (X,Y,\mathcal{S},\delta,\lambda,S_0)$ where $X \subseteq \mathbb{N}^2$, $Y \subseteq \mathbb{N}$, $\mathcal{S}$ is a set of K states among which is a specified state S_0 called the *initial* state, $\delta : \mathcal{S} \times X \to \mathcal{S}$, and $\lambda : \mathcal{S} \to Y$. Initially the robot is at some node u_0 in the initial state $S_0 \in \mathcal{S}$. S_0 determines a local port number $p = \lambda(S_0) \in Y$, by which the robot leaves u_0. When incoming to a node v, the behavior of the robot is as follows. It reads the number i of the port through which it entered v and the degree d_v of v. The pair $(i,d_v) \in X$ is an input symbol that causes the transition from state S to state $S' = \delta(S,(i,d_v))$. S' determines a local port number $p = \lambda(S')$, by which the robot leaves v. The robot continues moving in this way, possibly infinitely.

As mentioned before, we consider the task of "perpetual" exploration in which the robot has to traverse all edges of the graph but is not required to stop. That is, it is not required that a final state be in $\mathcal{S}$. A robot is said to perform an *exploration* of a graph G, if starting at *any* node of G in the initial state S_0, it completes traversing all edges of G in finitely many steps.

3 Summary of our results

Our first result is the design of a trap with at most $K+1$ vertices for any K-state automaton. More precisely, we prove that, for any $d \geq 3$ and for any K-state automaton, there exists a planar graph of $K+1$ nodes and maximum degree d that the automaton cannot explore. (We assume $d \geq 3$ since, obviously, all connected graphs of maximum degree $d \leq 2$ can be explored by a robot with a constant memory size.) This construction improves –in terms of size– the ones by Budach and Rollik, as well as all other previous constructions in the literature.

Theorem 1 *For every K-state robot and every $d \geq 3$, there exists a planar graph of maximum degree d with at most $K+1$ nodes that the robot cannot explore.*

Due to lack of space, most of the proofs are omitted. They can be found in the complete version of the paper [11]. Roughly speaking, the proof of Theorem 1 places the robot in the infinite d-ary tree, and considers the first time the robot returns in a state experienced before. Then, the proof "cuts" and "folds" the infinite tree to obtain a finite planar graph G in which the behavior of the robot is the same as in the infinite tree. The way the folding is done insures that at least one edge of G is not traversed. See [11] for more details.

We can rephrase Theorem 1 as follows:

Corollary 1 *A robot that explores all n-node planar graphs requires at least $\lceil \log n \rceil$ memory bits.*

Our construction methodology is quite generic and can be adapted for the minimization of other graph parameters. In particular, we prove that, for any $d \geq 3$ and for any K-state automaton, there exists a planar graph of $O(K)$ nodes, maximum degree d, and diameter $O(\frac{\log K}{\log d})$ that the automaton cannot explore.

Theorem 2 *For every K-state robot and every $d \geq 3$, there exists a planar graph of maximum degree d and diameter at most $4\lceil \log_{d-1} K \rceil + 2$ that the robot cannot explore.*

The proof of Theorem 2 uses the construction in the proof of Theorem 1. The graph is "completed" by a binary tree, insuring that all nodes become close to each other thanks to the binary tree. The difficulty is to show that this completion is indeed possible. See [11] for more details.

Theorem 2 has an important corollary, namely:

Corollary 2 *A robot that explores all graphs of diameter D and maximum degree d requires at least $\Omega(D \log d)$ memory bits.*

This latter lower bound is tight. Indeed, we design an algorithm for a robot with $O(D \log d)$ memory bits enabling the robot to explore all graphs of maximum degree d and diameter D. More specifically, we present an algorithm called `Increasing-DFS`, that enables a robot to explore all graphs of sufficiently small diameter and maximum degree. Roughly speaking, exploration is achieved by using a sequence of *depth-first search (DFS)* operations at increasing depths from the initial position u_0 of the robot. The robot keeps in memory the current sequence of port numbers leading back to u_0 in the DFS tree. At Phase i, $i \geq 1$, the robot performs a DFS of depth bounded by i. In the case where one is given a robot $\mathcal{R}$ with k memory bits, we use the variant k-`Increasing-DFS`, that is `Increasing-DFS` in which the robot perpetually checks the size of the currently allocated memory. If this size exceeds k bits, then the robot stops. See [11] for a complete description of Algorithm `Increasing-DFS`.

Theorem 3 *Algorithm* `Increasing-DFS` *allows a robot to explore every graph. Moreover, Algorithm k-*`Increasing-DFS` *explores all graphs of diameter D and maximum degree d, whenever $k \geq \alpha D \log d$, for some positive constant α.*

Proof. Let the robot $\mathcal{R}$ start from node u_0 in graph G. After $\mathcal{R}$ has performed a DFS of depth i, it has visited all nodes at distance at most i from u_0. Let $i = D+1$ where D is the diameter of G. Thus, after the ith phase of Algorithm `Increasing-DFS`, all edges have been traversed, and thus exploration has been completed. If $k \geq \alpha D \log d$, then a stack of $D+1$ elements on $\log d$ bits, and a constant number of scalar variables, can be stored in the robot's memory, for $\alpha = O(1)$ large enough. Thus, when $i = D+1$, the exploration is completed using no more than k bits. Hence any graph of diameter D and maximum degree d can be explored. □

A direct consequence of Theorem 3 is the following:

Corollary 3 *All graphs of diameter D and maximum degree d can be explored by a robot using $O(D \log d)$ memory bits.*

To summarize, we prove that the worst case space complexity of graph exploration is $\Theta(D \log d)$ bits.

As a final remark, observe that algorithm `Increasing-DFS` uses an infinite memory to explore some graphs of bounded size. Nevertheless, this phenomenon cannot be overcome by any exploration algorithm. Indeed, surprisingly, any infinite automaton that explores all graphs is required to use an infinite amount of memory to explore some finite graphs. In particular, for $d \geq 0$, let $\mathcal{G}_d$ be the set of all edge-colored d-regular graphs ($\mathcal{G}_d \neq \emptyset$ as witnessed by, e.g., the hypercube Q_d, or two nodes linked by d parallel edges). We have the following:

Theorem 4 *For any (infinite deterministic) automaton $\mathcal{R}$ that explores all graphs, and for any $G \in \mathcal{G}_d$, $\mathcal{R}$ uses infinitely many memory states when exploring G.*

Proof. Let $\mathcal{R}$ be a (deterministic) automaton that explores all graphs, and let $G \in \mathcal{G}_d$. As a consequence of Theorem 1, $\mathcal{R}$ is an *infinite* automaton $(X, Y, \mathcal{S}, \delta, \lambda, S_0)$, i.e., $|\mathcal{S}|$ is unbounded. Assume, for the purpose of contradiction, that $\mathcal{R}$ uses K states of $\mathcal{S}$ when executed in G, starting from some node, say u_0. Let $\mathcal{R}'$ be the automaton obtained by restricting $\mathcal{R}$ to the diagram induced by these K states of $\mathcal{S}$. More precisely, $\mathcal{R}' = (X, Y, \mathcal{S}', \delta', \lambda', S_0)$ where $\mathcal{S}'$ is the set of the K states visited by $\mathcal{R}$ when exploring G starting from u_0, λ' is λ restricted to $\mathcal{S}'$, and δ' is δ restricted to $\mathcal{S}' \times X$. Let $\mathcal{G}_d(\mathcal{R}')$ be the set of pairs (H, v_0) where $H = (V, E)$ is an edge-labeled graph and $v_0 \in V$, such that, starting at v_0 in H, $\mathcal{R}'$ visits only nodes of degree d and traverses only edges that have identical labels at their two extremities. Let (H, v_0) be the trap for $\mathcal{R}'$ constructed in the proof of Theorem 1. By our construction, we have $(H, v_0) \in \mathcal{G}_d(\mathcal{R}')$. Moreover, since $G \in \mathcal{G}_d$, we also have $(G, u_0) \in \mathcal{G}_d(\mathcal{R}')$. Let $(S_i)_{i \geq 0}$ be the sequence of states of $\mathcal{R}'$ when exploring G starting from u_0. By construction of $\mathcal{R}'$, $(S_i)_{i \geq 0}$ is also the sequence of states of $\mathcal{R}$ when exploring G starting from u_0. In fact, we have $\{S_i, i \geq 0\} = \mathcal{S}'$, and $S_{i+1} = \delta'(S_i, \lambda'(S_i, d)) = \delta(S_i, \lambda(S_i, d))$. Therefore, the sequence $(S_i)_{i \geq 0}$ is independent of any instance (graph, starting node) $\in \mathcal{G}_d(\mathcal{R}')$, and is independent of which automaton $\mathcal{R}$ or $\mathcal{R}'$ is exploring that instance. In particular, the sequence $(S_i)_{i \geq 0}$ is the same for $\mathcal{R}$ and $\mathcal{R}'$ in (H, v_0). Therefore, the sequences of nodes visited by $\mathcal{R}$ and $\mathcal{R}'$ when exploring H starting from v_0 are identical. Since (H, v_0) is a trap for $\mathcal{R}'$, this latter fact is in contradiction with the fact that $\mathcal{R}$ is universal, and thus explores all graphs, including H. Hence $\mathcal{R}$ uses an infinite number of states when exploring G. □

4 Conclusion and future work

We have proved that $\Omega(\log n)$ memory bits are necessary to explore all n-node graphs, and that $\Theta(D \log d)$ memory bits are necessary and sufficient to explore all graphs of diameter D and maximum degree d.

As mentioned in the introduction, Rollik [16] proved that no finite set of finite automata can separately (i.e., non-cooperatively) explore all undirected graphs. For his proof, Rollik constructed a trap for q robots of K states each, that is, a graph that none of the q robots explores completely. This trap is of size $O(K^q)$. Thus, an interesting direction of research is to look for smaller traps. In particular, we raise the question of whether there exists a trap of polynomial size for any set of q robots of K states each.

Acknowledgments: P. Fraigniaud and D. Ilcinkas are supported by the project "PairAPair" of the ACI Masses de Données; A. Pelc is supported in part by NSERC grant OGP 0008136 and by the Research Chair in Distributed Computing of the Université du Québec en Outaouais.

References

[1] S. Albers and M. R. Henzinger, Exploring unknown environments, SIAM Journal on Computing 29:1164-1188, 2000.

[2] M.A. Bender, A. Fernandez, D. Ron, A. Sahai and S. Vadhan, The power of a pebble: Exploring and mapping directed graphs, Proc. 30th Ann. Symp. on Theory of Computing (STOC), pages 269-278, 1998.

[3] M.A. Bender and D. Slonim, The power of team exploration: Two robots can learn unlabeled directed graphs, Proc. 35th Ann. Symp. on Foundations of Computer Science (FOCS), pages 75-85, 1994.

[4] M. Blum and D. Kozen. On the power of the compass (or, why mazes are easier to search than graphs). In 19th Symposium on Foundations of Computer Science (FOCS), pages 132-142, 1978.

[5] L. Budach. Automata and labyrinths. Math. Nachrichten, pages 195-282, 1978.

[6] W. Coy. Automata in labyrinths. In Fund. Computat. Theory (FCT), LNCS 56, 65-71, 1977.

[7] X. Deng and C. H. Papadimitriou, Exploring an unknown graph, Journal of Graph Theory 32:265-297, 1999.

[8] K. Diks, P. Fraigniaud, E. Kranakis, and A. Pelc. Tree Exploration with Little Memory. In 13th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA), pages 588-597, 2002.

[9] C. Duncan, S. Kobourov and V. Kumar, Optimal constrained graph exploration. In 12th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA), pages 807-814, 2001.

[10] P. Fraigniaud and D. Ilcinkas. Exploration with little memory. In Symp. on Theoretical Aspects of Computing Science (STACS), LNCS 2996, pages 246-257, 2004.

[11] P. Fraigniaud, D. Ilcinkas, G. Peer, A. Pelc, and D. Peleg. Graph exploration by a finite automaton. Submitted for publication. See *http://www.lri.fr/˜pierre/gewfa.pdf*.

[12] D. Kozen. Automata and planar graphs. In Fund. Computat. Theory (FCT), 243-254, 1979.

[13] H. Müller. Endliche Automaten und Labyrinthe. Elektronische Informationsverarbeitung und Kybernetic, EIK 7/4, 261-264, 1971.

[14] P. Panaite and A. Pelc, Exploring unknown undirected graphs, Journal of Algorithms 33:281-295, 1999.

[15] M.O. Rabin, Maze threading automata. Seminar talk presented at the University of California at Berkeley, October 1967.

[16] H.A. Rollik. Automaten in planaren Graphen. Acta Informatica 13:287-298, 1980 (also in LNCS 67, pages 266-275, 1979).

[17] CL. E. Shannon. Presentation of a maze-solving machine. In 8th Conf. of the Josiah Macy Jr. Found. (Cybernetics), pages 173-180, 1951.