Comment battre la marche aléatoire en comptant ?

Nicolas Hanusse, David Ilcinkas, Adrian Kosowski, Nicolas Nisse

Comment battre la marche aléatoire en comptant ?[†]

N. Hanusse[1] and D. Ilcinkas[1] and A. Kosowski[2] and N. Nisse[3]

[1] *CNRS, LaBRI/INRIA, Université de Bordeaux 1, Bordeaux, France,*
[2] *Dept. of Alg. and Syst. Modeling, Gdańsk University of Technology, Gdańsk, Poland,*
[3] *MASCOTTE, INRIA, I3S(CNRS/UNS), Sophia Antipolis, France*

Nous étudions le problème consistant à trouver une destination t dans un réseau, non fiable, grâce à un agent mobile. Chaque nœud du réseau peut donner un conseil quant au prochain sommet à visiter pour se rapprocher de t. Malheureusement, k nœuds, appelés *menteurs*, donnent de mauvais conseils. Il est connu que pour un graphe G de n sommets et de degré maximum $\Delta \geq 3$, atteindre une cible à distance d de la position initiale peut demander un temps moyen de $2^{\Omega(\min\{d,k\})}$, pour tout $d,k = O(\log n)$, même lorsque G est un arbre. Ce papier étudie une stratégie, appelée R/A, utilisant un compteur (d'étapes) pour alterner entre les phases aléatoires (R) où l'agent choisit aléatoirement une arête incidente, et celles (A) où l'agent suit le conseil local. Aucune connaissance des paramètres n, d, ou k n'est requise, et l'agent n'a pas besoin de se rappeler par quel lien il est entré dans le sommet qu'il occupe. Nous étudions les performances de cette stratégie pour deux classes de graphes, extrêmes pour ce qui est de l'expansion : les anneaux et les graphes réguliers aléatoires (une importante classe d'*expanders*). Pour l'anneau, l'algorithme R/A requiert un temps moyen de $2d + k^{\Theta(1)}$ (polynomial en d et k) pour une distribution des menteurs la plus défavorable. A l'opposé, nous montrons que dans un anneau, une marche aléatoire biaisée requiert un temps moyen exponentiel en d et k. Pour les graphes aléatoires réguliers, le temps de recherche moyen de l'algorithme R/A est $O(k^3 \log^3 n)$ a.a.s. Le terme polylogarithmique de cette borne ne peut pas être amélioré, puisque nous montrons une borne inférieure de $\Omega(\log n)$ pour $d,k = \Omega(\log\log n)$ dans les graphes aléatoires réguliers a.a.s. qui s'applique même lorsque l'agent a le sens de l'orientation.

Keywords: Calcul distribué, Agent mobile, Marche aléatoire, Expanders, Réseaux non fiables

1 The search problem

Walking in the streets of Belle Dune, you decide to go to its famous golf, but do not know where it is situated. You first ask some people who say you should go to the North. After a short walk, you ask a policeman who is almost sure it is to the East. At the next intersection, you are told to go to the South. At least one of the persons you have met is mistaken. What is the best strategy to quickly find the golf ?

Locating an item (a piece of information, data, services, etc.) is one of the most common tasks in a distributed environment. This is, for instance, the role of search engines in the World Wide Web. One idea for a user of a network to locate an item at some node is to send agents out to search for the desired item [KK99, KKKS03]. If the mobile agent is provided complete information about the network and the location of the item, it can quickly find it by following a shortest path from its current position to the node hosting the item. Without any assumption on the level of knowledge of the agents/node, the only solution consists in exploring the whole network without a map until the desired item is encountered. Numerous algorithms dealing with exhaustive network exploration , either deterministic (e.g., Bread First Search, Universal Traversal Sequences [AKL$^+$79], Universal Exploration Sequences [Rei08]) or randomized (e.g., random walks [Lov93], biased random walks [ABK$^+$96]), have been designed.

In the context of large scale and dynamic networks like the World Wide Web or the Internet, the two previous approaches are either pessimistic (no knowledge) or optimistic (full reliable knowledge). In this paper, we assume an intermediate level of knowledge : we consider the problem of locating an item hosted by some node t of the n-node network $G = (V,E)$ when each of the nodes maintains a database storing the

[†]Works of N. Hanusse and D. Ilcinkas are partially supported by ANR ALADDIN and N. Nisse is partially supported by ANRs AGAPE and DIMAGREEN. These three authors are partially supported by the DCR project.

first edge on a shortest path to the *destination* t. The search is performed by a mobile agent with a limited perception of the environment and with little memory which starts from some initial node s, the *source*. When occupying a node, the mobile agent can perform a query to the node's database that reveals to it an edge that is the beginning of a shortest path from the current node to the item. We assume, however, that some nodes may provide wrong information, that is, a node v may indicate an edge that does not belong to any shortest path from v to the destination. This is motivated by the fact that inaccuracies occur in the nodes'databases because nodes may malfunction or be malicious, or may store out-of-date information due to the movement of items or the dynamicity of the network. This is also the case when you are searching some place in a city by asking your way to some people you meet. A node providing wrong information is called a *liar*, otherwise it is a *truth-teller*. The problem is then to deal with the potentially incorrect information and to find the desired item. That is, the mobile agent can decide to follow the edge pointed by its current node's database or not. The performance of the search is measured by comparing the *searching time* (a.k.a. *hitting time*), i.e., the length of the walk followed by the agent from s to t, with the distance $d(s,t) = d$ between these two nodes.

We investigate the search problem in the class of regular graphs in the presence of a bounded number $k \geq 0$ of liars. Our main contribution is the design and study of a randomized algorithm, called R/A that alternates phases of pure random walk (R) with phases in which the agent follows the advice (A). This is closely related to biased random walks which are random walks in which nodes have a statistical preference to shift the walker towards the target, or more generally, prevent the walker from staying too long in one vicinity [ABK$^+$96]. We show that Algorithm R/A improves upon previous algorithms for the search problem in rings and random Δ-regular graphs. In particular, in these classes, we prove that the Algorithm R/A achieves searching time much smaller than $\Omega(d+2^k)$, which is the lower bound for general regular graphs [HKKK08]. Note that the graph classes we consider capture the two extreme types of behavior in terms of expansion, since random Δ-regular graphs are good expanders, while rings are highly symmetric graphs with poor expansion. However, Algorithm R/A is generic and works for any topology.

Related Work. The search problem in the presence of liars was first investigated by Kranakis and Krizanc in [KK99]. In this seminal work, they designed algorithms for searching in distributed networks with ring or torus topology, when a node has a constant probability of being a liar [KK99]. The case when the number k of liars is bounded was first considered in [HKK04], where deterministic algorithms were designed for particular topologies like the complete graph, ring, torus, hypercube, and bounded degree trees. In particular, in bounded degree trees, it was proved that the search time is lower-bounded by $\Omega(d+2^{\min\{k,d\}})$ [HKK04]. Simple randomized and memoryless algorithms are designed in [HKKK08] for the case of bounded degree graphs, where the mobile agent follows the advice with some fixed probability $p > 1/2$. In this class of graphs, the authors showed that the expected distance covered before reaching the destination is upper-bounded by $O(d+r^k)$, where $r = \frac{q}{1-q}$ [HKKK08]. Moreover, this analysis is tight since they proved a lower bound of $\Omega(d+r^k)$ in the torus [HKKK08]. While this bound is a bit disappointing, it can be improved for particular graph classes. In this paper, we focus on some particular and widely used topologies.

Expander graphs are highly connected sparse graphs that play an important role in computer science and the theory of communication networks (see [HLW06] for a survey). They arise in constructions of error-correcting codes with efficient encoding and decoding algorithms, derandomisation of random algorithms, embeddings of finite metric spaces, network design, etc. A particular class of graphs with powerful expansion properties is that of random Δ-regular graphs [GHW08]. A *random Δ-regular graph* $\mathcal{G}_{n,\Delta}$ is an element of $\mathcal{G}(n,\Delta)$, the set of Δ-regular graphs with n nodes viewed as a probability space with uniform probability [JRL00]. Because of their low diameter and high connectivity, these graphs are also of interest in Peer-to-Peer networks (e.g., see [GHW08]).

Terminology and model. The $deg(u)$ edges incident to any node u are labeled by *port numbers*, from 1 to $deg(u)$, so that the agent can distinguish the different edges incident to a node. We do not assume that the agent knows or keeps in its memory the previous node (or in-port) it comes from.

At each *step* of the execution of an algorithm, the agent performs a query to its current node $v \in V$. If $v = t$, the item is found and the mobile agent stops. Otherwise, a piece of advice $a \in \{1, \cdots, deg(v)\}$ is given to the agent, representing the port number a of the next edge e incident to v which should be crossed

Strategy	Expected searching time		Memory
	path (ring)	random Δ-regular graphs	
BRW [$p < 1/2$]	$2^{\Omega(d)}$ [Lov93]	$\Theta(\min\{(\Delta\text{–}1)^k, n^{\Theta(1)}\})$	–
BRW [$p = 1/2$]	$\Omega(dn)$ [Lov93]	$\Omega(\log^2_{\Delta\text{-}1} n)$ [CF05]	–
BRW [$p > 1/2$]	$\Omega(d + 2^{\Omega(k)})$[HKKK08]	$n^{\Theta(1)}$	–
R/A*	$2d + k^{\Theta(1)} + o(d)$ a.a.s.	$O(k^3 \log^3 n)$ a.a.s.	timer
Lower bound*		$\Omega(\min\{(\Delta\text{–}1)^k, (\Delta\text{–}1)^d, \log_{\Delta\text{-}1} n\})$	

TABLE 1: The listed strategies have no knowledge of inbound ports used by the agent, but the lower bound allows it.

in order to reach t. If v is a *truth-teller*, e belongs to a shortest path between v and t. Otherwise, v is called a *liar*. Finally, the mobile agent chooses some edge incident to v and traverses it. *A priori*, a search algorithm can take into account the set of advice encountered so far to choose the next edge to cross. In particular, this means that a node should always provide the same advice, otherwise it would easily be identified as a liar. However, for practical applications, it is natural to limit the agent's memory. Here, we consider that the agent only has a timer (whose size is specified below). Hence, a lying node may or may not provide always the same advice (but a node cannot change its status : it is either consistently a liar or a truth-teller). Finally, the agent will also be assumed to have no global knowledge about the size of the network, the number of liars, and the value of d. We are mainly interested in the expected number of edge traversals, named the *searching time*, taken by the mobile agent to reach the target t, and in comparing it with d.

2 Performances of Algorithm R/A

We design and study Algorithm R/A [t_R,t_A] defined as follows. The agent alternates between phases in which it performs a random walk for $t_R \geq 0$ steps and then follows advice for $t_A \geq 0$ steps (Algorithm 1).

Algorithm 1 The R/A algorithm with phase durations t_R, t_A.

Algorithm R/A $[t_R, t_A]$: Repeat the following sequence of two phases until the target is reached :

1. *Random phase* (R) : the mobile agent performs a pure unbiased random walk for t_R steps ;
2. *Advice phase* (A) : the mobile agent follows the advice for t_A steps.

Note that this algorithm can fundamentally work for any topology. However, a cautious design of duration is necessary : if the duration of the phase A is much larger than that of phase R, the mobile agent could be stuck forever in the same area full of liars. By carefully parameterizing the durations, we prove that :

– in the path (similarly in the ring), an agent using a counter of $O(\log k)$ bits ends up at the target t within $2d + O(k^5) + o(d)$ moves with probability $1 - \Theta(1/k^{c-3})$, where c is a constant, and
– in random Δ-regular graphs, an agent using a counter of $O(\log k + \log\log n)$ bits ends up at the target t within $O(c^3 \cdot k^3 \log^3 n)$ steps with probability $1 - 1/2^{\Omega(c)}$, where c is a constant.

In both results, no knowledge of n, d, or k is required. Finally, in the case of random Δ-regular graphs, we provide a lower bound of $\Omega(\log n)$ which holds for all $d, k = \Omega(\log\log n)$ and applies even to strategies which are in some sense not oblivious with respect to the environment.

Table 1 ‡ establishes a comparison between the performances of Algorithm R/A, and the performances of the *Biased Random Walk* (BRW). In the BRW strategy, the agent flips a biased coin and accepts the advice with probability p and rejects it with probability $1 - p$, in which case it also selects any of the remaining incident edges with uniform probability in the number of remaining incident edges. Full proofs can be found in [HIKN10] and for convenience, we just present here the schemes whenever k is known.

‡. In Table 1, the * points out our results.

Sketch of proof (Chain - $t_R = t_A$). Given an integer $r = \sqrt{t_R \log k}$, we decompose the chain into regions : safe area without any liars and boxes containing liars. Informally, any node leading to a liar in phase A or being at a distance at most r from a liar belongs to a box. In safe areas, one iteration of R/A takes the agent closer to the target at a speed of $1/2$ on average (explaining the $2d$ factor) whereas within a box one iteration of R/A decreases the distance of $\Omega(\sqrt{t_R}/k)$ a.a.s., as soon as $t_R > 32k^3$. Hence, by definition of a box, $O(k^2 \log k)$ expected iterations are sufficient to cross all the boxes, leading to an expected search time $O(2d + t_R k^2 \log k)$.

Sketch of proof (Expanders). To study the behavior of the R/A algorithm for random regular graphs, we introduce a new algorithm called R/A/E$[t_R, t_A, r_E]$ where an extra phase E is added. The agent starting the phase E in node v explores all the $\leq (\Delta-1)^{r_E}$ nodes at distance at most r_E from v (this can be done by adding some memory to the agent). Let us set $t_R = O(\log_{\Delta\text{-}1} n)$, i.e., the mixing time of random Δ-regular graphs, $t_A = \log_{\Delta\text{-}1} n - \log_{\Delta\text{-}1} k$ and $r_E = D - t_A$ where $D = \Theta(\log_{\Delta\text{-}1} n + \log_{\Delta\text{-}1} \log n)$ greater than the diameter of a random Δ-regular graph a.a.s. First, observe that regardless of the initial location of the agent, after completion of the random walk (R) phase, the agent is located at any node v of the graph with probability at least $1/n - O(1/n^3)$ by the definition of the mixing time. Then, the distance between v and the target t is bounded by D, a.a.s. We prove that, with probability $\Theta(1)$, v is at distance at least t_A from any liar. Hence, with probability $\Theta(1)$, during Phase A, the agent reachs a node w at distance at most $D - t_A$ from t, and t is reached during Phase E. Thus, Algorithm R/A/E allows the agent to reach t in at most $t_R + t_A + (\Delta-1)^{r_E} = O(k \log n)$ steps a.a.s. By simulating Phase E by $O(k \log n)$ iterations of Phase R, we prove that Algorithm R/A$[t_R + r_E, t_A]$ finds t in $O(k^2 \log^2 n)$ steps a.a.s. Finally, when the agent has no knowledge of either k or n, applying R/A$[x, y]$ and increasing sequentially x and y allows it to find t in $O(k^3 \log^3 n)$ steps a.a.s.

Improvements and further works. The persistent memory of the agent following the R/A strategy is limited to a timer which counts the number of performed steps. Somewhat surprisingly, the memory requirement cannot be decreased to 0 without affecting the performance of the algorithm even on the path. It is, however, possible to implement a variant of R/A using only a one-bit state, representing the phase currently being performed. At each step, the agent then switches to the other phase with some probability, which may be fixed *a priori* if the topology of the graph, its order, and the number of liars are known in advance.

It is natural to ask if the proposed R/A strategy or its variants may be applied to other graph classes, with a searching time polynomial in the number of liars and the distance to the target. Whereas the approach applied for paths generalizes e.g. to some cases of 2-dimensional grids, we leave this question open, e.g., for the much wider class of graphs of bounded doubling dimension.

Références

[ABK+96] Y. Azar, A. Z. Broder, A. R. Karlin, N. Linial, and S. Phillips. Biased random walks. *Combinatorica*, 16(1) :1–18, 1996.

[AKL+79] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proc. of the 20th Symp. on Foundations of Comp. Sc. (FOCS)*, pages 218–223, 1979.

[CF05] C. Cooper and A. Frieze. The cover time of random regular graphs. *SIAM J. Discret. Math.*, 18(4) :728–740, 2005.

[GHW08] C. S. Greenhill, F. B. Holt, and N. C. Wormald. Expansion properties of a random regular graph after random vertex deletions. *Eur. J. Comb.*, 29(5) :1139–1150, 2008.

[HIKN10] N. Hanusse, D. Ilcinkas, A. Kosowski, and N. Nisse. How to beat the random walk when you have a clock ? Research Report 7210, INRIA, February 2010. http ://hal.inria.fr/inria-00458808/PDF/RR-7210.pdf.

[HKK04] N. Hanusse, E. Kranakis, and D. Krizanc. Searching with mobile agents in networks with liars. *Discrete Applied Mathematics*, 137 :69–85, 2004.

[HKKK08] N. Hanusse, D. J. Kavvadias, E. Kranakis, and D. Krizanc. Memoryless search algorithms in a network with faulty advice. *Theor. Comput. Sci.*, 402(2-3) :190–198, 2008.

[HLW06] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4) :439–561, 2006.

[JRL00] S. Janson, A. Ruciński, and T. Luczak. *Random Graphs*. Wiley-Interscience, 2000.

[KK99] E. Kranakis and D. Krizanc. Searching with uncertainty. In *Proc. SIROCCO'99,*, pages 194–203, 1999.

[KKKS03] L. M. Kirousis, E. Kranakis, D. Krizanc, and Y. Stamatiou. Locating information with uncertainty in fully interconnected networks. *Networks*, 42 :169–180, 2003.

[Lov93] L. Lovász. Random walks on graphs : A survey. *Combinatorics*, pages 1–46, 1993.

[Rei08] O. Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), 2008.