How to beat the random walk when you have a clock?

Nicolas Hanusse, David Ilcinkas, Adrian Kosowski, Nicolas Nisse

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

How to beat the random walk when you have a clock?

Locating a target with an agent guided by unreliable local advice

Nicolas Hanusse — David Ilcinkas — Adrian Kosowski — Nicolas Nisse

N° 7210

February 2010

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--7210--FR+ENG

INRIA
SOPHIA ANTIPOLIS

How to beat the random walk when you have a clock?

Locating a target with an agent guided by unreliable local advice

Nicolas Hanusse* , David Ilcinkas* , Adrian Kosowski**adrian@kaims.pl** , Nicolas Nisse†

Thème COM — Systèmes communicants
Projets Mascotte

Rapport de recherche n° 7210 — February 2010 — ii pages

Abstract: We study the problem of finding a destination node t by a mobile agent in an unreliable network having the structure of an unweighted graph, in a model first proposed by Hanusse *et al.* [19,20]. Each node is able to give advice concerning the next node to visit so as to go closer to the target t. Unfortunately, exactly k of the nodes, called *liars*, give advice which is incorrect. It is known that for an n-node graph G of maximum degree $\Delta \geq 3$, reaching a target at a distance of d from the initial location may require an expected time of $2^{\Omega(\min\{d,k\})}$, for any $d, k = O(\log n)$, even when G is a tree.

This paper focuses on strategies which efficiently solve the search problem in scenarios in which, at each node, the agent may only choose between following the local advice, or randomly selecting an incident edge. The strategy which we put forward, called R/A, makes use of a timer (step counter) to alternate between phases of ignoring advice (R) and following advice (A) for a certain number of steps. No knowledge of parameters n, d, or k is required, and the agent need not know by which edge it entered the node of its current location. The performance of this strategy is studied for two classes of regular graphs with extremal values of expansion, namely, for rings and for random Δ-regular graphs (an important class of expanders). For the ring, R/A is shown to achieve an expected searching time of $2d + k^{\Theta(1)}$ for a worst-case distribution of liars, which is polynomial in both d and k. For random Δ-regular graphs, the expected searching time of the R/A strategy is $O(k^3 \log^3 n)$ a.a.s. The polylogarithmic factor with respect to n cannot be dropped from this bound; in fact, we show that a lower time bound of $\Omega(\log n)$ steps holds for all $d, k = \Omega(\log \log n)$ in random Δ-regular graphs a.a.s. and applies even to strategies which make use of some knowledge of the environment.

Finally, we study oblivious strategies which do not use any memory (in particular, with no timer). Such strategies are essentially a form of a random walk, possibly biased by local advice. We show that such biased random walks sometimes achieve drastically worse performance than the R/A strategy. In particular, on the ring, no biased random walk can have a searching time which is polynomial in d and k.

Key-words: Distributed Computing, Mobile Agent, Random Walks, Expanders, Faulty Networks

Work of N. Hanusse and D. Ilcinkas was partially suported by ANR ALADDIN, and N. Nisse was partially funded by ANR AGAPE and ANR DIMAGREEN.

* CNRS, LaBRI/INRIA, Université de Bordeaux 1, Bordeaux, France **firstname.lastname@labri.fr**
† MASCOTTE, INRIA, I3S, CNRS, Univ. Nice Sophia, Sophia Antipolis, France. **firstname.lastname@sophia.inria.fr**

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Comment battre la marche aléatoire avec une montre ?

Résumé : Nous étudions le problème de trouver une destination t grâce à un agent mobile, dans un réseau non fiable modélisé par un graphe. Ce problème a initialement été étudié par Hanusse *et al.* [19,20]. 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. Pour un graphe 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 présente des stratégies pour r'esoudre efficacement le problème de recherche. Suivant ces strat'egies, à chaque nœud, l'agent peut choisir de suivre le conseil local ou choisir aléatoirement une arête incidente. La stratégie considérée, appelée R/A, utilise un compteur (d'étapes) pour alterner entre les phases aléatoires (R) et celles où le conseil est suivi (A). 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 régulers aléatoires (une importante classe d'*expanders*). Pour l'anneau, R/A requiert un temps moyen de $2d + k^{\Theta(1)}$ (polynomial en d et k) pour une distribution des menteurs la plus défavorable. Pour les graphes régulers aléatoires, le temps de recherche moyen de 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 régulers aléatoires a.a.s. qui s'applique même pour des stratégies utilisant un sens de l'orientation.

Pour finir, nous étudions des stratégies sans mémoire (n'utilisant pas de compteur). Ces stratégies sont des marches aléatoires possiblement biaisées par les conseils locaux. Nous montrons que le temps de recherche moyen atteint par ces stratégies est sensiblement plus long que celui atteint par R/A. En particulier, dans un anneau, une marche aléatoire biaisée requiert un temps moyen exponentiel en d et k.

Mots-clés : Calcul distribué, Agent mobile, Marche aléatoire, Expanders

1 Introduction: the search problem

Walking in the streets of Paris, you decide to visit the famous *Musée du Louvres*, 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 museum?

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 [29,26,24]. 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. Another way for the agent to find an item without being provided any information consists in exploring the whole network until the desired item is encountered. Several works have been devoted to the problem of exhaustive network exploration and numerous algorithms, both deterministic (e.g., Bread First Search, Universal Traversal Sequences [2,35], Universal Exploration Sequences [28,34]) and randomized (e.g., random walks [1,30,25], biased random walks [4,6]), have been designed. In the context of large scale networks like the World Wide Web or the Internet, it is impractical to have full knowledge of the network because of its size and dynamicity. It is also impossible to fully explore the network since such an approach requires a time complexity (at least) in the order of the number of nodes of the network. Another important constraints to the complete exploration of a network are the memory of the agent (e.g., see [15]) and the notion of sense-of-direction [12,13,14].

In this paper, we consider the problem of locating an item hosted by some node of the network when each of the nodes maintains a database storing the first edge on a shortest path to the node hosting the desired item (the *destination*). 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, 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 the source to the destination, with the length of a shortest path between these nodes.

In this paper, we investigate the search problem in the class of regular graphs in the presence of a bounded number 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). We show that Algorithm R/A improves upon previous algorithms for the search problem in paths 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 [20]. 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 the other classes are highly symmetric graphs with poor expansion. Note, however, that Algorithm R/A is generic and works for any topology.

1.1 Related Work

The search problem in the presence of liars was first investigated by Kranakis and Krizanc [29]. 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 [29]. The case when the number k of liars is bounded was first considered in [19], 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 is proved that the search time is lower-bounded by $\Omega(d + 2^{\min\{k,d\}})$ [19]. Simple randomized and memoryless algorithms are designed in [20] 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 $r = \frac{q}{1-q}$ [20]. Moreover, this bound is tight since they proved a lower bound of $\Omega(d+r^k)$ in the torus [20]. 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.

Biased random walks. Roughly speaking, the algorithms we present in this paper consist of alternation of phases of given duration: either the agent keeps on following the advice provided by the nodes or it walks choosing the next visited node uniformly at random in the neighboorhood of the current position. 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 [4]. More formally, biased random walks are used in network exploration in order to speed up the time required to visit the whole network without an a-priori knowledge of the topology and without an edge/node labeling requirement. For instance, Ikeda *et al.* proved in [22] that, assuming the knowledge of the degrees of the neighbors, a biased random walk can explore any graph within $O(n^2 \log n)$ edge traversals whereas a uniform random walk takes $\Theta(n^3)$ steps for some graphs. In our context, however, the bias may be erroneous due to the presence of liars.

Expanders and random regular graphs. Expander graphs are highly connected sparse graphs that play an important role in computer science and the theory of communication networks (see [21] for a survey). Formally, a graph $G = (V, E)$ is a c-expander if, for any $X \subset V$ with $|X| \leq |V|/2$, then $|N(X) \setminus X| \geq c|X|$ where $N(X)$ is the set of neighbours of X. Expanders arise in questions about designing networks that connect many users while using only a small number of switches (e.g., see [3]). They also arise in constructions of error-correcting codes with efficient encoding and decoding algorithms, derandomisation of random algorithms, embeddings of finite metric spaces, etc.

Expanders and random regular graphs have been extensively studied for the design of optimal networks and algorithms for routing [32,27,17]. Because of their low diameter and high connectivity, random regular graphs are also of interest in Peer-to-Peer networks (e.g., see [18]). More generally, it can be observed that many interaction networks like peer to peer overlay networks, small worlds and scale-free networks are expanders despite this is not proved in the original papers. For instance, Bourassa and Holt [8] proposed a fully decentralized protocol based on random walks for the nodes to join and leave the network. They conjectured that their protocol produces random regular graphs, which was proved formally in [10]. On the other hand, Cooper [9] *et al.* show that random regular graphs are expanders.

1.2 Terminology and the model

Throughout the paper, a distributed network is for our purposes an undirected n-node graph $G = (V, E)$. There are two distinguished nodes in the graph, the *source* $s \in V$, and the *destination* $t \in V$, hosting the desired item. The distance $d(u, v)$ between two nodes u and v corresponds to the number of edges of a shortest path from u to v. We set $d(s, t) = d$. The number of liars is denoted by $k \geq 0$. For any $v \in V$ and $r \geq 0$, let $N_v(r)$ denote the *distance-r neighborhood of* v, i.e., the set of all nodes $u \in V$ such that $\text{dist}(u, v) \leq r$. We will call the subgraph $B_v(r) = G[N_v(r)]$ the *ball* with center v and radius r. The $deg(u)$ edges incident to any node u are labeled by *port numbers*, from 1 to $deg(u)$, so that the searchers can distinguish the different edges incident to a node. There is no *sense-of-direction* [12,13,14], meaning that the local labeling of the edges satisfies no global consistency constraints (e.g., right/left in a path, or North/South/East/West in a grid). In most cases, we assume that the agent does not know the label of the port by which it entered the current node. Note that in a Δ-regular graph, all nodes are indistinguishable for the agent.

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 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. In the following, the set of liars is denoted by $\mathcal{L}$ and the set of truth-tellers by $\mathcal{TT}$. *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.

1.3 Results and structure of the paper

We design and study Algorithm R/A $[t_R,t_A]$ defined as follows. The mobile 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). Note that this algorithm can fundamentally work for any topology. However, a cautious

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.

design of duration is necessary. For instance, 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 and following Algorithm R/A 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 (Section 2), and
- in random Δ-regular graphs, an agent using a counter of $O(\log k + \log\log n)$ bits and following Algorithm R/A 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 (Section 3).

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.

Tables 1 and 2 establish 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. Note that, if $p = 1/2$, BRW is the pure Random Walk.

2 Searching the path

In this section, let us assume that the initial graph is a path P of n nodes. For this topology, we look at the behavior of the generic algorithms, which we later study for expanders. Recall that k denotes the number of liars.

Note that whereas it is of course possible to design extremely simple and efficient algorithms specifically for searching a path, such algorithms will usually not perform well in general. Consider for example the

Table 1: Searching in a path with liars. The listed strategies have no knowledge of inbound ports used by the agent.

Strategy	Expected searching time	Memory	Reference
BRW $[p < 1/2]$	$2^{\Omega(d)}$	–	[30]
BRW $[p = 1/2]$	$\Omega(dn)$	–	
BRW $[p > 1/2]$	$\Omega(d + 2^{\Omega(k)})$	–	[20]
R/A	$2d + k^{\Theta(1)} + o(d)$	timer	Thm. 2

Table 2: Searching with liars in random Δ-regular graphs. Results marked with (*) allow the agent to make use of labels of inbound ports and to have knowledge of n and k.

Strategy	Expected searching time (a.a.s.)	Memory	Reference
* lower bound	$\Omega(\min\{(\Delta–1)^k, (\Delta–1)^d, \log_{\Delta\text{-}1} n\})$		Thm. 6
BRW $[p > 1/2]$	$\Theta(\min\{(\Delta–1)^k, n^{\Theta(1)}\})$	–	
BRW $[p = 1/2]$	$\Omega(\log^2_{\Delta\text{-}1} n)$	–	[9]
BRW $[p < 1/2]$	$n^{\Theta(1)}$	–	
* R/A/E	$O(k \log n)$	$\Theta(\log k + \log\log_{\Delta\text{-}1} n)$	Thm. 3
R/A	$O(k^3 \log^3 n)$	timer	Thm. 4

1D Cow Path strategy proposed in [5], which does not take into account the advice. For $i \in [1, \lfloor \log_2 d \rfloor]$, it just consists in going 2^i steps in one given direction, then in going 2^{i+1} steps in the opposite direction and repeats the sequence until the target is found. This algorithm is efficient in the case of the path, always reaching its target in at most $9d$ steps [5]; however, this is not the case when the expansion of the graph is large (e.g., the d-Dimensional Cow Path approach will require $\Omega(d^D)$ steps for D-dimensional grid). Moreover, this algorithm requires some sense-of-direction, or at least knowledge of the port by which the agent enters each node.

It is interesting to note that the Biased Random Walk (BRW) performs badly on the path, see Table 1. When the probability of following advice is $p \leq 1/2$, the strategy proves ineffective even when there are no liars in the network [30]. For $p > 1/2$, the searching time is sometimes exponential in the number of liars [20]. Regardless of the value of p, the searching time of BRW in the path is always lower-bounded by $\Omega(d + 2^{\min\{k,d\}})$.

In this section, we study the performance of Algorithm R/A$[L, L]$ on a path. Recall that, following this algorithm, the search consists of rounds of length $2L > 0$. During each round, the mobile agent first executes a random walk during L steps (phase R), and then it follows the advices during the next L steps (phase A).

Let $u_0, u_1, \ldots, u_L$ be the sequence of nodes traversed by the mobile agent in a phase R and let $v_0 = u_L, v_1, \cdots, v_L$ be the sequence of nodes traversed by the mobile agent in a phase A. We are interested in the *gain* $X = X_A + X_R$ of v_L with respect to u_0 defined by $X = d(u_0, t) - d(v_L, t)$, X_R (resp. X_A) being the gain during phase R (resp. A).

In the following, we will make use of an assumption: u_L is almost always close to u_0. More formelly, let $\mathcal{C}_\ell$ be the event defined by u_L is at distance at most $\sqrt{2L\ell}$ from u_0, i.e., $u_L \in B_{u_0}(\sqrt{2L\ell})$. We get:

Lemma 1 (Random Phase)**.** *Let $c > 0$ and $k > 0$. At the end of Phase R, we have:*

1. *With probability at least $1 - 2/k^c$, event $\mathcal{C}_{c \log k}$ holds;*
2. *With probability at least $1 - 2/e$, event $\mathcal{C}_1$ holds.*

Let $|i| \leq L$ with same parity as L.

3. $\Pr(d(u_L,u_0)=i) \leq p_{\max} = 0.8/\sqrt{L}$*;*

4. If $|i| \leq \sqrt{2L}$*,* $\Pr(d(u_L,u_0)=i) \geq p_{\min} = 0.1/\sqrt{L}$*.*

Proof. Let $X_1, X_2, \ldots, X_L$ be independent random variables such that $\Pr(X_i = 1) = \Pr(X_i = -1) = \frac{1}{2}$. The gain $X_R = d(u_0,t) - d(u_L,t)$ during phase R is $X_R = \sum_{i=1}^{L} X_i$, that is, u_L is at distance $|X_R|$ from u_0. From Chernoff's bound, we have:

$$\Pr(|X_R| \geq a) \leq 2e^{\frac{-a^2}{2L}}, \quad a > 0.$$

For the first statement (resp. the second), take $a = \sqrt{2cL\log k}$ (resp. $a = \sqrt{2L}$).

Let $-L \leq i \leq L$ such that $L - i \equiv 0[2]$. The probability that $X_R = i$ is exactly $\binom{L}{L/2+|i|/2}/2^L$. We recall that we should have $L/2 + i/2$ moves in one given direction and $L/2 - i/2$ moves in the opposite direction. We prove the lemma in the case $L = 2\ell$, the case L odd is similar. The function $f(2j) = \binom{L}{\ell+j}/2^L$ is a decreasing function ($0 \leq j \leq \ell$). Hence, for all $0 \leq j \leq \ell$, $f(2j) \leq f(0) = \binom{L}{\ell}/2^L$. By the Stirling Formula, $f(0) \sim \sqrt{\frac{2}{\pi L}} \leq 0.8/\sqrt{L}$.

The last statement is similar. Indeed, since the function f is decreasing, we get that, for any $i = 2j$ with $|i| \leq \sqrt{2L}$, $f(i) \geq f(2\lfloor\sqrt{2L}/2\rfloor) \geq f(2\sqrt{L})$. Moreover, $f(2\sqrt{L}) = \binom{L}{\ell+\sqrt{L}}/2^L = \frac{\sqrt{(2)}\cdot(e^{ln(2)-1})^2}{4\sqrt{\pi}\sqrt{L}} + o(L^{-3/2})$. Hence, $f(2\sqrt{L}) \geq 0.1/\sqrt{L}$. □ □

Consider the subpath of $P \setminus \{t\}$ containing u_0. It can be decomposed into at most $2k+1$ maximal subpaths of consecutive nodes such that each subpath is composed only of nodes of same state, that is liars or truth-tellers. We assume that the subpaths (or sets of nodes) are ordered from the subpath furthest from t to the subpath closest to t. Define $\mathcal{L}_i$ (resp. $\mathcal{T}_i$) as the i-th set of liars (resp. truth-teller) of the path.

Lemma 2 (Advice Phase)**.** *Let* $c \geq 1$ *such that* $\sqrt{c+1} - \sqrt{c} > 1/\sqrt{2k\log k}$*, and* $L > 32k^3$*. Let* k' *be the number of liars at distance at most* $\sqrt{2(c+1)L\log k}$ *from* u_0*. Condition on* $\mathcal{C}_{c\log k}$*, we have:*

1. $-k' \leq X_A < 0$ *with probability at most* $p_{\max}\cdot k' = O(k'/\sqrt{L})$*;*

2. $X_A \geq \sqrt{2L}/k' - 1$ *with probability at least* $\sqrt{2L}p_{\min}(1-2/e) = \Theta(1)$*;*

3. $\mathbb{E}(X) = \mathbb{E}(X_A)$ *and* $\mathbb{E}(X_A) \geq \frac{\sqrt{L}}{40k'} - 1$*.*

Proof. By hypothesis, $u_L \in B_{u_0}(\sqrt{2cL\log k})$, the path of $2\sqrt{2cL\log k}+1$ consecutive nodes centered at node u_0.

We first prove that after phase A, the distance can only increase by at most k' units. If u_L is a liar belonging to $\mathcal{L}_i$, we know that during phase A, the distance to t can increase by at most $|\mathcal{L}_i| \leq k$ units, i.e., $X_A = d(u_0,t) - d(v_L,t) \leq d(u_0,t) - d(u_L,t) + |\mathcal{L}_i|$. In this case, since $d(u_L,v_L) \leq k$, we have that $v_L \in B_{u_0}(\sqrt{2(c+1)L\log k})$ because, by the choice of L and c, $\sqrt{2(c+1)L\log k} - \sqrt{2cL\log k} > k$. Hence, $|\mathcal{L}_i| \leq k'$.

If u_L is a truth-teller, the distance to t decreases by a number of units which is the minimum of L and the distance to the next set of liars (towards t).

Let us focus of the gain X_A during phase A. $X_A < 0$ if and only if u_L is a liar node. Otherwise $X_A \geq 0$. $X_A = 0$ is possible as soon as u_L is a truth-teller being a neighbor of a liar and L is even.

From Lemma 1, we know that for any given node v at distance at most L from u_0 and such that $d(u_0,v)$ has same parity as L, $\Pr(u_L = v) \leq p_{\max}$. Moreover, by hypothesis, u_L belongs to $B_{u_0}(\sqrt{2cL\log k})$ that contains at most k' liars. Hence, with probability at most $p_{\max}k'$, u_L is a liar and thus $-k' \leq X_A < 0$.

Whenever u_L is a truth-teller, u_L belongs (or not) to $B_{u_0}(\sqrt{2L})$. By Lemma 1, any nodes of $B_{u_0}(\sqrt{2L})$ has almost the same probability to be chosen (there is a multiplicative factor of at most $p_{\min}/p_{\max}$). $X_A = i$ if there are i consecutive truthtellers before the next liar. Let us prove that $X_A = \Omega(\sqrt{L}/k')$ with constant probability. We just focus on truth-tellers belonging to $B_{u_0}(\sqrt{2L})$.

For $y \geq 1$, let t_y be the number of disjoint sequences of consecutive truth-tellers of length y of $B_{u_0}(\sqrt{2L})$. Note that $\sum_{y\geq 1} t_y \leq k'$. By definition, we have:

$$\sum_{t_y \geq 1} y \cdot t_y \geq 2\sqrt{2L} - k' + 1 \tag{1}$$

Since at least $\sum_{y\geq x}(y-x)t_y$ vertices in $B_{u_0}(\sqrt{2L})$ are at distance at least x from the next liar in the direction of t, we have

$$\Pr(X_A \geq x) \quad \geq \quad p_{\min}(1-2/e)\sum_{y\geq x}(y-x)t_y \tag{2}$$

Taking $x = \sqrt{2L}/k' - 1$, we have $\sum_{y=0}^{x} xt_y \leq \sqrt{2L} - k'$. From Eq. (1), it follows that $\sum_{y\geq x}(y-x)t_y \geq \sqrt{2L}+1$ and $\Pr(X_A \geq \sqrt{2L}/k' - 1) \geq \sqrt{2L}p_{\min}(1-2/e) = \Theta(1)$.

The expectation of X_A is

$$\begin{aligned}
\mathbb{E}(X_A) &\geq -k'\Pr(X_A < 0) + (\sqrt{2L}/k' - 1)\Pr(X_A \geq \sqrt{2L}/k' - 1) && (3)\\
&\geq -k'^2 p_{max} + (\sqrt{2L}/k' - 1)\sqrt{2L}p_{\min}(1-2/e) && (4)\\
&\geq -\frac{0.8k'^2}{\sqrt{L}} + \frac{0.1\sqrt{2}(1-2/e)(\sqrt{2L}-k')}{k'} && (5)
\end{aligned}$$

Since $L > 32k'^3$, we get:

$$\begin{aligned}
\mathbb{E}(X_A) &\geq \frac{\sqrt{L}}{k'}(-1/40 + 1/20) - 1 && (6)\\
&\geq \frac{\sqrt{L}}{40k'} - 1 && (7)
\end{aligned}$$

Since $X = X_R + X_A$ and $\mathbb{E}(X_R) = 0$, we get $\mathbb{E}(X) = \mathbb{E}(X_A)$ □

We now need some definitions. The *advice cone* C_u of a liar u is the set of nodes v such that there is a path $v_0 = v, v_1, \ldots, v_i = u$ from v to u with $v_j = \mathsf{Adv}(v_{j-1})$. For $r \geq 1$, we note the *r-restricted advice cone* of a liar u by $C_u(r) = C_u \cap N_u(r)$. The *r-zone* of a liar u is $N_{C_u(r)}(r)$, that is the set of all vertices at distance at most r from $C_u(r)$ (including $C_u(r)$). Finally, a *r-box* is a connected component of the subgraph induced by all r-zones.

In other words, consider one of the two components of $P \setminus \{t\}$, and let $\{u_1, \cdots, u_k\}$ be the liars in this component, ordered by decreasing distance to t. The cone of u_1 consists of u_1 plus the component of $P \setminus \{u_1\}$ that does not contain t. For any $i > 1$, C_{u_i} consists of the subpath of truth-tellers between u_{i-1} and u_i, plus u_i. It follows that, on the path, a r-box containing $k' \geq 1$ liars $\{u'_1, \cdots, u'_{k'}\}$ consists of a subpath $P' = \{v_1 \cdots, v_p\}$, such that the neighbors of v_1 and v_p not in P' are truth-tellers, $u'_1 = v_{2r+1}$, $u'_{k'} = v_{p-r-1}$, and for any $1 < i < k'$, u'_i is between u'_{i-1} and u'_{i+1}, and for any $1 \leq i < k'$, $d(u_i, u_{i+1}) \leq 2r$. In particular, a r-box containing k' liars is a subpath of length at least $3r+1$, and at most $(2k'+1)r + k'$.

Lemma 3 (Inside a box)**.** *Let $L = \Omega(k'^3)$, $c \geq 1$, and $r = \sqrt{2cL\log k}$. Let us assume that event $\mathcal{C}_{c\log k}$ always occurs. Given a r-box $\mathcal{B}$ of $k' \leq k$ liars, the agent, initially located within any node of $\mathcal{B}$, leaves the box after $O(k'^2\sqrt{c\log k})$ expected rounds of Algorithm $R/A[L,L]$ in the direction of the target. The exit time is $O(\max\{c\log k, k'^2\sqrt{c\log k}\})$ with probability $1 - 2/k'^c$.*

Proof. First, let us make some easy remarks.

- If, at the end of a phase A, the mobile agent still stands in $\mathcal{B}$, then it is on a liar u or on the neighbor of u in its cone (indeed, in this case, at the end of phase A, the agent oscillates between these two nodes).
- Condition on event $\mathcal{C}_{c\log k}$, either the agent exits from the box in the direction of the target or stay within the box. Once the agent is outside the box after one phase R, during phase A, it increases its distance from $\mathcal{B}$ of at least $r+1$ (the distance minimum to the first liar of the next box) and at most L (if no liars are encountered).

Now, we prove that the gain during i rounds is very close to the lower bound of its expectation $\mathbb{E}(iX) \geq i(\sqrt{L}/40k' - 1)$ obtained in Lemma 2 (note that, at each round starting in some vertex u of the box, $B_u(r)$ contains at most k' liars, and thus Lemma 2 applies). More formally, let $X[i]$ (resp. $X_R[i]$ and $X_A[i]$) be the gain obtained during the first i-th rounds of algorithm $\mathsf{R/A}$ (resp. phase R and phase

A) within the box. By definition, $X[0] = 0$ and as soon as $X[i] > (2k'+1)r + k'$, the agent exits from the box.

Due to the $\mathcal{C}_{c\log k}$ assumption, $|X_R[i]| \leq \sqrt{2c(iL)\log k})$ since the i-th iteration of phases R of length L can be seen as one iteration of phase R of lenght iL. Hence, $X[i] = X_A[i]+X_R[i] \geq X_A[i]-\sqrt{2c(iL)\log k}$. So as soon as $X_A[i] \geq \sqrt{2c(iL)\log k} + ((2k'+1)r + k')$ with high probability, the agent exits from the box.

In order to get an upper bound on the exit time from the box, we define another random variable Y_A.

$$\begin{aligned} Y_A &= -k' \text{ with probability } p_{\max}k' \\ &= \frac{\sqrt{2L}}{k'} - 1 \text{ with probability } \sqrt{2L}p_{\min}(1-2/e) \\ &= 0 \text{ otherwise} \end{aligned}$$

From Lemma 2, $X_A \geq Y_A$, i.e., for any m, $\Pr(X_A \geq m) \geq \Pr(Y_A \geq m)$.

It follows that $\mathbb{E}(Y_A) \geq \sqrt{L}/40k' - 1 = \Theta(\frac{\sqrt{L}}{k'})$. We note $Y_A[i]$ the sum of Y_A's on i iterations. Since, the iterations of Y_A are independent, $\mathbb{E}(Y_A[i]) = \Theta(i\frac{\sqrt{L}}{k'})$. Using Hoeffding bound, we have:

$$\Pr\left(|Y_A[i] - \mathbb{E}(Y_A[i])| \geq \delta\mathbb{E}(Y_A[i])\right) \leq 2\exp\left(-\frac{2\,\delta^2\mathbb{E}(Y_A[i])^2}{i(k'+\sqrt{L}/k'-1)^2}\right).$$

By taking $\delta = 1/2$, we get that $Y_A[i] \geq \frac{i\sqrt{L}}{80k'}$ with probability at least $1 - 2\exp\left(-\frac{iL}{2\cdot 40^2(k'^2+\sqrt{L}-k')^2}\right) \approx 1 - 2\exp\left(-\frac{i}{2\cdot 40^2}\right)$. Hence, as soon as $i \geq c\log k$, we get that $X_A[i] \geq Y_A[i] = \Theta(i\frac{\sqrt{L}}{k'})$ with probability at least $1 - 2k^{-c}$. Finally, for $i = \Omega(k'^2\log k)$, we get that $\Theta(i\frac{\sqrt{L}}{k'}) = \Omega(\sqrt{2c(iL)\log k} + ((2k'+1)r + k'))$.

To conclude, as soon as $i = \Omega(k'^2\log k)$, we get that $X_A[i]-((2k'+1)r+k') \geq |X_R[i]| \geq \sqrt{2c(iL)\log k}$ with probability at least $1 - 2k'^{-c}$. Hence, $X[i] \geq ((2k'+1)r + k')$ and the mobile agent leaves the box with high probability. By remark above, it will never go back inside. □ □

Lemma 4 (Outside the box)**.** *Assume that the agent stays outside of any box during one iteration. With probability* 1*,* $X = d(u_0,t)-d(v_L,t) \geq 0$ *and* $\mathbb{E}(X) = L$*. Moreover, given event* $\mathcal{C}_{c\log k}$*,* $X \geq L-\sqrt{2Lc\log k}$

Proof. Since the agent stays outside a box, it implies that it only encounters truth-tellers during L steps of phase A. Since $X_R \geq -L$ with probability one, we have $X = X_A + X_R \geq 0$ and $\mathbb{E}(X) = \mathbb{E}(X_A) + \mathbb{E}(X_R) = L + 0$. Moreover, given $\mathcal{C}_{c\log k}$, $X_R \geq -\sqrt{2Lc\log k}$ □

Theorem 1. *Assuming the knowledge of* k*, if* $L > 32k^3$*, Algorithm* R/A *finds the target within* $2d + O(Lk^2\log k) + o(d)$ *steps with probability* $1 - \Theta(1/k^{c-3})$*.*

Proof. Let us decompose the unique path into sequence of safe area and r-boxes with $r = \sqrt{2cL\log k}$. We just focus on the set of boxes that are between s and t. Boxes $\mathcal{B}_1, \ldots, \mathcal{B}_i$ are ordered by the distance toward the target, that is $\mathcal{B}_1$ is closer to t than $\mathcal{B}_2$ and so on.

The proof is based on one property: conditioning on $\mathcal{C}_{c\log k}$, once the mobile agent exits from a box $\mathcal{B}$, it will never come back to $\mathcal{B}$ with high probability. Then Lemmas 3 and 4 will be applied to compute exactly the amount of steps to cross the safe area and boxes.

Let k_i be the number of liars in box $\mathcal{B}_i$. By definition, $k \geq \sum k_i$. By definition, the total number of vertices of the boxes is less from $\sum 2k_i r \leq 2kr$ and the expected number of iterations to cross all of them is less that $\sum O(k_i^2\log k) = O(k^2\log k)$. In our analysis, the event $\mathcal{C}_{c\log k}$ is assumed within the boxes and the first round R after exiting a box. The probability to have this event during $\Theta(k^2\log k)$ iterations is greater than $1 - 2/k^{c-3}$.

To traverse a safe area of length larger than L, event $\mathcal{C}_{c\log k}$ is not required. Once the distance decreases by L units in a safe area, the agent is unable to come back to a already visited box.

By construction, the agent exits from a box during a phase R. During the next phase A, either it reaches the next box in one iteration or it stays in a safe area. In this last case, the distance can not

increase and decreases by L units on expectation. From Lemma 4, we know that the gain in the next iteration is non-negative with probability 1.

Since the safe areas between s and t have at most $d = d(s,t)$ vertices, it takes $2d/L$ iterations on average to traverse them and, from Lemma 1, $2d/L + o(d/L)$ iterations with probability $1 - 2k^{-c}$. We just have to add at most $2k$ extra iterations for the transitions between safe areas and boxes. The total number of iterations is at most $2d/L + O(k^2 \log k) + o(d/L)$ with probability $1 - \Theta(k^{-c+3})$. □

To conclude this section, we prove that the knowledge of k is not necessary to achieve the same performance as the one of Theorem 1.

Theorem 2. *The iterated execution of Algorithm* R/A$[L_i = 2^{3i}, L_i]$*, with parameter i starting at $i = 0$ and increasing by one unit every $t_i = i \cdot 2^{2i}$ phases, finds the target within $2d + O(k^5 \log k) + o(d)$ steps with probability $1 - \Theta(1/k^{c-3})$, without assuming knowledge of k.*

Proof. Whenever the number of iterations ($2d/L$ on average) required to cross safe areas is larger than the ones to traverse the boxes of liars (roughly $40k^2\sqrt{\log k}$ iterations), the time to reach the target is dominated by the time of traversal of the safe areas.

For $i = 5/3 + \log_2 k$, the i-th execution of the proposed algorithm corresponds to the execution of R/A[L,L], with $L = 2^{5/3+\log_2 k} = 32k^3$ during $O(i \cdot 2^{2i}) = O(k^2 \log k)$ iterations. According to Theorem 1, this finds the target with probability $1 - \Theta(1/k^{c-3})$.

Now, the total number of steps of the proposed algorithm until it terminates its $(5/3 + \log_2 k)^{th}$ iteration is

$$O(\sum_{i=1}^{5/3+\log_2 k} L_i t_i) = O(\sum_{i=1}^{5/3+\log_2 k} 2^{3i} \cdot i \cdot 2^{2i}) = O(\sum_{i=1}^{\log_2 k} i \cdot (2^5)^i) = O(2^{5\log_2 k} \log k) = O(k^5 \log k)$$

Either the target is reached during these steps ($d < \frac{32 \cdot 40 \cdot k^5 \sqrt{\log_2 k}}{2}$) or the agent spend more time in the safe areas and the target is reached within $2d + o(d)$ steps. □

3 Searching in random Δ-regular graphs

3.1 Preliminaries: Properties of Random Δ-regular graphs

A particular class of graphs with powerful expansion properties is that of random Δ-regular graphs. 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 [23]. It is well known that a random regular graph does not have the same properties as those of the standard random graphs model, namely the Erdős-Rényi random graphs $\mathcal{G}'_{n,p}$ with parameter p [1], even when the parameter p is chosen so that both graphs have the same expected number of edges ($p = \Delta/n$). For $\Delta = o(\log n)$, $\mathcal{G}_{n,\Delta}$ is connected asymptotically almost surely (a.a.s.), whereas this is not the case for $\mathcal{G}'_{n,\Delta/n}$. Random Δ-regular graphs were proved to be very powerful expanders, a.a.s. [33, 18]. Moreover, there exists a quick randomized algorithm for generating such graphs [33].

We recall here some properties of random Δ-regular graphs that will be useful in the paper.

Lemma 5 (diameter [7])**.** *There exists a constant D such that the diameter of $\mathcal{G}_{n,\Delta}$ a.a.s. satisfies the relation $|\operatorname{diam} \mathcal{G}_{n,\Delta} - (\log_{\Delta\text{-}1} n + \log_{\Delta\text{-}1} \log n)| < D$.*

Lemma 6 (tree-like neighbourhood [31, 11])**.** *There exists a constant $c > 0$, such that for any node u of $\mathcal{G}_{n,\Delta}$, the subgraph induced by $B_u(c \log n)$ is a.a.s. a tree.*

Lemma 7 (mixing time [16, 9])**.** *For any starting node u of $\mathcal{G}_{n,\Delta}$ and after following a random walk of $M_{n,\Delta} = 8\frac{\log n}{\log(\Delta/4)}$ steps, for every node v, the probability that the walk ends at v is at least $1/n - 1/n^3$.*

This last lemma implies that we will restrict considerations to the case $\Delta > 4$ in the following discussion.

[1] In the Erdős-Rényi model, a random graph is built in the following way: for any pair of nodes, flip a biased coin and it exists an edge between these two nodes with probability p

3.2 Upper Bounds

Before discussing the behavior of the R/A algorithm for random regular graphs, we introduce a new algorithm called R/A/E (Algorithm 2). Algorithm R/A/E is formulated in a way which assumes that we have knowledge of the port by which we entered each node in order to locally explore a ball in the exploration phase. Then, we show how to apply an analogous analysis to the R/A algorithm, for an agent with no knowledge of inbound ports.

Algorithm 2 The R/A/E algorithm with parameters t_R, t_A, r_E.

Algorithm R/A/E $[t_R, t_A, r_E]$: Repeat the following sequence of three 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;
3. *Exploration phase* (E): the mobile agent explores locally, that is, visits all the nodes at distance up to a given radius r_E from its location at the start of the phase.

The values of parameters t_R, (resp. t_A and r_E), which are used for determining the duration of phase R (resp. A and E), are all set deterministically *a priori* and will be described later.

Let $r_1 = \log_{\Delta\text{-}1} n - 1$, let $r_2 = \log_{\Delta\text{-}1} n + \log_{\Delta\text{-}1} \log n + D$, where D is the constant from Lemma 5, and let the target t be an arbitrarily chosen node in graph $G \in \mathcal{G}_{n,\Delta}$.

Theorem 3. *For a graph $G \in \mathcal{G}_{n,\Delta}$, the* R/A/E *algorithm with parameters: $t_R = M_{n,\Delta}$, $t_A = r_1 - \log_{\Delta\text{-}1} k$, and $r_E = \log_{\Delta\text{-}1} k + (r_2 - r_1)$, completes any search in expected time $O(k \log n)$, a.a.s.*

Proof. We will call an execution of three successive phases (random walk, advice, exploration) an *iteration* of the algorithm. The number of steps within each iteration is deterministically bounded by the sum of the durations of its three phases:

$$t_R + t_A + O((\Delta\text{–}1)^{r_E}) = O(\log_{\Delta\text{-}1} n) + O(\log_{\Delta\text{-}1} n) + O(k \log n) = O(k \log n).$$

To achieve $O(k \log n)$ expected search time, it now suffices to prove that the probability of locating the target in any iteration is $\Theta(1)$.

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 s of the graph with probability at least $1/n - O(1/n^3)$ by the definition of the mixing time $M_{n,\Delta}$. We remark that by Lemma 5, the distance between s and the target t is bounded by $d(s,t) \leq \mathrm{diam}(G) < r_2$, a.a.s.

The proof is completed by showing that if indeed $d(s,t) < r_2$, then the probability of locating the target in any iteration of algorithm R/A/E is $\Theta(1)$. Let κ denote the random variable describing the number of liars encountered by the agent during the advice (A) phase. Let s' be the location of the agent at the end of the advice phase. Observe that if $\kappa = 0$, then either the target is reached in the advice phase, or in each of the t_A steps of the advice phase, the agent moves towards the target by a distance of 1, and consequently:

$$d(s',t) = d(s,t) - t_A \leq r_2 - (r_1 - \log_{\Delta\text{-}1} k) = r_E.$$

Then, the target will be reached in the exploration (E) phase. So, it suffices to show that the event $\kappa = 0$ occurs with probability $\Theta(1)$. Observe that a sufficient condition for this event to occur is that the neighborhood $N_s(t_A)$ does not contain any liars, or equivalently, that we have $s \in V \setminus \bigcup_{l \in \mathcal{L}} N_l(t_A)$, where $\mathcal{L} \subseteq V$ is the set of liars. Since the graph is of maximum degree Δ, we have for any node l:

$$|N_l(t_A)| \leq (\Delta\text{–}1)^{t_A} \leq \frac{1}{k}(\Delta\text{–}1)^{r_1} \leq \frac{n}{2k}.$$

So, $|\bigcup_{l \in L} N_l(t_A)| \leq |\mathcal{L}|\frac{n}{2k} = \frac{n}{2}$, and $|V \setminus \bigcup_{l \in L} N_l(t_A)| \geq \frac{n}{2}$. Recall that s is chosen as any node from V with probability $1/n - O(1/n^3)$, hence the event $s \in V \setminus \bigcup_{l \in L} N_l(t_A)$ occurs with probability $(n/2)(1/n - O(1/n^3)) = 1/2 - O(1/n^2) = \Theta(1)$, which completes the proof. □

Theorem 4. *For a graph $G \in \mathcal{G}_{n,\Delta}$, the* R/A *algorithm with parameters: $t_R = M_{n,\Delta} + \log_{\Delta\text{-}1} k + (r_2 - r_1)$ and $t_A = r_1 - \log_{\Delta\text{-}1} k$, completes any search in expected time $O(k^2 \log^2 n)$, a.a.s.*

Proof. Observe that the only difference between algorithms R/A/E$[M_{n,\Delta},\ r_1 - \log_{\Delta\text{-}1} k,\ r_E]$ and R/A$[M_{n,\Delta} + r_E,\ r_1 - \log_{\Delta\text{-}1} k]$, where $r_E = \log_{\Delta\text{-}1} k + (r_2 - r_1)$, is that the exploration phase up to a radius of r_E in algorithm R/A/E has been replaced by a pure unbiased random walk of r_E steps at the start of the random phase of the next iteration of algorithm R/A. Observe that if node t is reached from node s' by exploration up to depth r_E, then $d(s', t) \leq r_E$, and so the probability that t is reached by a random walk of r_E steps starting from s' is at least $\frac{1}{(\Delta-1)^{r_E}} = \Omega(\frac{1}{k \log n})$. If the success probability of an iteration of algorithm R/A/E is denoted as p, then the success probability of two consecutive iterations of R/A is $\Omega(\frac{p}{k \log n})$. Recalling from the proof of Theorem 3 that $p = \Theta(1)$, we have that in expectation, after $O(k \log n)$ iterations of algorithm R/A, the target is found. Since the duration of each iteration is $O(k \log n)$, this completes the proof. □

We now show that by appropriately setting the number of phases in each iteration, it is possible to apply Algorithm R/A to the searching problem without knowledge of n or k. We start by proving the following lemma.

Lemma 8. *Let $L_1 = \log_{\Delta\text{-}1} n - \log_{\Delta\text{-}1} k - 1$ and $L_2 = k$. Then, for any fixed integer c, the target is located with probability $1 - 1/2^{\Omega(c)}$ during the execution of $ck \log n$ iterations of Algorithm* R/A$[100L_1, L_1]$ *and $ck \log n$ iterations of Algorithm* R/A$[100L_2, L_2]$*, a.a.s.*

Proof. We consider two cases. If $k \geq \sqrt{n}$, then a single iteration of R/A$[100L_2, L_2]$ includes a random phase of length $100k \geq 100\sqrt{n}$. By the properties of random regular graphs [9], a random walk of this length reaches the target t with probability at least $\Theta(\frac{1}{\sqrt{n} \log n})$. After $ck \log n \geq c\sqrt{n} \log n$ iterations of R/A$[100L_2, L_2]$, the probability of reaching the target is $1 - 1/2^{\Omega(c)}$.

If $k < \sqrt{n}$, then we have $100L_1 = 100(\log_{\Delta\text{-}1} n - \log_{\Delta\text{-}1} k - 1) > 50 \log_{\Delta\text{-}1} n - 100 > 8\frac{\log n}{\log(\Delta/4)} = M_{n,\Delta}$, for all $\Delta \geq 5$ and sufficiently large n. Since increasing the duration of the random phase does not affect the correctness of the reasoning in the proof of Theorem 4, we obtain that 2 consecutive iterations of R/A$[100L_1, L_1]$ reach the target with probability $\Theta(\frac{1}{k \log n})$. Hence, after $ck \log n$ iterations of R/A$[100L_1, L_1]$, the probability of reaching the target is $1 - 1/2^{\Omega(c)}$. □

Theorem 5. *Consider an execution of Algorithm* R/A$[100i, i]$ *which consists of stages numbered with successive integers $j = 1, 2, 3, \ldots$, and the j-th stage includes exactly one iteration of the algorithm with the iteration length parameter set to i, for all $i = 1, 2, \ldots, j$. The target is located within $O(c^3 \cdot k^3 \log^3 n)$ steps with probability $1 - 1/2^{\Omega(c)}$, a.a.s., without assuming the knowledge of k or n.*

Proof. Once the j-th stage of the algorithm, where $j = 2ck \log n$, has been completed, at least $ck \log n$ iterations of the algorithm have been performed with parameter values R/A$[100L_1, L_1]$ and R/A$[100L_2, L_2]$ from Lemma 8 (since $L_1, L_2 \leq ck \log n$). Hence, by Lemma 8, the target has been located with probability $1 - 1/2^{\Omega(c)}$. It suffices to note that the j-th stage is completed within $\Theta(j^2)$ phases, or equivalently within $\Theta(j^3) = \Theta(c^3 \cdot k^3 \log^3 n)$ steps. □

3.3 Lower Bound

Before proving the main theorem, we put forward a lower bound for a search scenario in a tree, which is a modification of the formulation of the known lower bound for trees from [20].

Proposition 1. *Let $T = (V_T, E_T)$ be a complete (Δ–1)-ary tree having l levels with a root at node s, let a ($1 \leq a \leq l$) be a known value, and let t be a target node chosen uniformly at random from the set of nodes of T such that $d(s, t) = a$. Assume that for all nodes of T, the advice is directed towards s, and the direction of the advice of s is arbitrarily chosen. Then in expectation any search strategy requires $\Omega((\Delta\text{–}1)^a)$ steps to reach target t from source s.*

Proof. The distribution of the advice in the given tree is independent of the location of the target t. Hence, the expected search time from s to t cannot be improved with respect to strategies in a model with no advice. □

Theorem 6. *For a graph $G \in \mathcal{G}_{n,\Delta}$, with k liars and source-target distance d, any search strategy requires $\Omega(\min\{(\Delta–1)^k, (\Delta–1)^d, \log_{\Delta\text{-}1} n\})$ steps in expectation, a.a.s.*

Proof. Let us denote $r = \lfloor \frac{c}{2} \log_{\Delta\text{-}1} n \rfloor$, where c is the constant from Lemma 6. The number of required steps is not less than the source-target distance, and so, if $d \geq r$, since $r = \Omega(\log_{\Delta\text{-}1} n)$, the claim clearly holds. We can thus assume that $d < r$.

Consider a searching process in which the source vertex s is such that $B_s(2r)$ is a tree (such a vertex can be found a.a.s. by Lemma 6), and let the target vertex t be chosen uniformly at random subject to the constraint $d(s,t) = d$. Since $B_s(d)$ is a tree, there exists a unique shortest path P of length d between s and t in G. If $k < d-1$, let the k nodes of path P which are closest to s be liars, whose advice is directed along path P to vertex s. If $k \geq d$, let all vertices of P (excluding s and t) be likewise defined as liars, and place the remaining liars at arbitrary nodes of G outside $N_s(2r)$.

Observe that, for any $v \in N_s(r)$, we have $d(v,t) \leq d(v,s) + d(s,t) \leq 2r$, and a path of length at most $2r$ connects v and t in $B_s(r)$. This path is the unique shortest path connecting v and t in G, since otherwise there would have to be another path in G of length at most $2r$ connecting v with t and the tree $B_s(2r)$ would contain a cycle, a contradiction. We will say that a node $v \in V$ *gives useless advice* if the following conditions are fulfilled: $v \in N_s(r)$, $v \neq t$, and v directs the agent along the path in the tree $B_s(2r)$ leading from v to s. Since for all $v \in N_s(r) \setminus P$, the unique shortest path from v to t in G leads via s, we obtain that all nodes $v \in N_s(r) \setminus P$ give useless advice. Moreover, all of the liars located on path P give useless advice by construction of the set of advice for liars.

To complete the proof, we will consider two possibilities:

- The search strategy reaches the target t, visiting only nodes which give useless advice. Then, until t is discovered, the visited subgraph must be a subtree of $B_s(r)$. The searching time for t is the same as in a corresponding set-up for the complete $(\Delta–1)$-ary tree with useless advice, and by Proposition 1, we obtain that in expectation the number of required steps is $\Omega((\Delta–1)^{d(s,t)}) = \Omega((\Delta–1)^d)$.

- Some of the nodes encountered in the search do not give useless advice. Let v be the first such encountered node. If $v \notin N_s(r)$, then $d(s,v) > r$, so the search process takes $\Omega(r) = \Omega(\log_{\Delta\text{-}1} n)$ steps. Otherwise, the search process before encountering node v must be confined to the tree $N_s(r)$, and node v must lie on the path P. Node v is necessarily the node of P located closest to s which is not a liar, i.e., $d(s,v) = k+1$. By applying Proposition 1 analogously as before, we have that the number of steps of the search which are performed in expectation before reaching v is at least $\Omega((\Delta–1)^{d(s,v)}) = \Omega((\Delta–1)^k)$.

The claimed bound is the minimum of the obtained bounds, taken over the considered cases. □

4 Conclusions

We have shown that there exists a simple, generic searching strategy which, for network topologies such as the path (and ring) or random regular graphs, allows an anonymous agent to locate the target efficiently. The number of moves is polynomial in the number of faults which appear in the network. The proposed R/A strategy is based on alternating phases in which the agent follows advice and performs a random walk. The precise duration of the phases has to be fine-tuned, depending on the graph class in which the search is performed (Theorems 2 and 4). As a matter of fact, by a minor modification to the proofs, it is possible to select a phase duration which gives good results for both of the specific graph classes studied in this paper.

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 in Section 2 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.

References

[1] 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. 20th FOCS*, pages 218–223, 1979.

[2] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 218–223, 1979.

[3] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. *Networks*. to appear.

[4] Yossi Azar, Andrei Z. Broder, Anna R. Karlin, Nathan Linial, and Steven Phillips. Biased random walks. *Combinatorica*, 16(1):1–18, 1996.

[5] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 106:234–252, 1993.

[6] Roberto Beraldi. Biased random walks in uniform wireless networks. *IEEE Trans. Mob. Comput.*, 8(4):500–513, 2009.

[7] Béla Bollobás and Wenceslas Fernandez de la Vega. The diameter of random regular graphs. *Combinatorica*, 2(2):125–134, 1982.

[8] V. Bourassa and F. Holt. Swan: Small-world wide area networks. In *Proceedings of International Conference on Advances in Infrastructure*, 2003.

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

[10] Colin Cooper, Martin E. Dyer, and Catherine S. Greenhill. Sampling regular graphs and a peer-to-peer network. *Combinatorics, Probability & Computing*, 16(4):557–593, 2007.

[11] Colin Cooper, Alan Frieze, and Tomasz Radzik. Multiple random walks in random regular graphs. Technical report, Dept. Mathematical Sciences of Carnegie Mellon, 2008.

[12] Paola Flocchini, Bernard Mans, and Nicola Santoro. Sense of direction: Definitions, properties, and classes. *Networks*, 32(3):165–180, 1998.

[13] Paola Flocchini, Bernard Mans, and Nicola Santoro. Sense of direction in distributed computing. *Theor. Comput. Sci.*, 291(1), 2003.

[14] Paola Flocchini, Alessandro Roncato, and Nicola Santoro. Computing on anonymous networks with sense of direction. *Theor. Comput. Sci.*, 1-3(301):355–379, 2003.

[15] P. Fraigniaud, D. Ilcinkas, G. Peer, A. Pelc, and D. Peleg. Graph exploration by a finite automaton. *Theor. Comput. Sci.*, 345(2-3):331–344, 2005.

[16] Joel Friedman. A proof of alon's second eigenvalue conjecture. In *STOC*, pages 720–724. ACM, 2003.

[17] Alan M. Frieze. Edge-disjoint paths in expander graphs. *SIAM J. Comput.*, 30(6):1790–1801, 2000.

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

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

[20] Nicolas Hanusse, Dimitris J. Kavvadias, Evangelos Kranakis, and Danny Krizanc. Memoryless search algorithms in a network with faulty advice. *Theor. Comput. Sci.*, 402(2-3):190–198, 2008.

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

[22] Satoshi Ikeda, Izumi Kubo, Norihiro Okumoto, and Masafumi Yamashita. Impact of local topological information on random walks on finite graphs. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 1054–1067. Springer, 2003.

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

[24] A. Kaporis, L. M. Kirousis, E. Kranakis, D. Krizanc, Y. Stamatiou, and E. Stavropoulos. Locating information with uncertainty in fully interconnected networks with applications to world wide web retrieval. *Computer Journal*, 44:221–229, 2001.

[25] A. R. Karlin and P. Raghavan. Random walks and undirected graph connectivity: A survey. In *Discrete Probability and Algorithms*, volume 72, pages 95–101. Institute for Mathematics and Its Applications, 1995.

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

[27] Jon M. Kleinberg and Ronitt Rubinfeld. Short paths in expander graphs. In *FOCS*, pages 86–95, 1996.

[28] Michal Koucký. Universal traversal sequences with backtracking. *J. Comput. Syst. Sci.*, 65(4):717–726, 2002.

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

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

[31] Eran Makover and Jeffrey McGowan. Regular trees in random regular graphs. Technical report, arxiv, 2008.

[32] Sotiris E. Nikoletseas, Krishna V. Palem, Paul G. Spirakis, and Moti Yung. Short vertex disjoint paths and multiconnectivity in random graphs: Reliable network computing. In *Proceedings of the 21st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 508–519, 1994.

[33] Sotiris E. Nikoletseas and Paul G. Spirakis. Expander properties in random regular graphs with edge faults. In *STACS*, pages 421–432, 1995.

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

[35] Michael E. Saks. Randomization and derandomization in space bounded computation. In *IEEE Conference on Computational Complexity*, pages 128–149, 1996.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399