Theoretical Computer Science 402 (2008) 190–198

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Memoryless search algorithms in a network with faulty advice[☆]

Nicolas Hanusse[a], Dimitris Kavvadias[b], Evangelos Kranakis[c], Danny Krizanc[d,*]

[a] CNRS, LaBRI, Université Bordeaux I, 351 Cours de la Libération, 33405 Talence, France
[b] Department of Mathematics, University of Patras, Rio, Greece
[c] Carleton University, School of Computer Science, Ottawa, ON, K1S 5B6, Canada
[d] Department of Mathematics and Computer Science, Wesleyan University, Middletown CT 06459, USA

ARTICLE INFO

Keywords:
Distributed computing
Mobile agent computing
Randomized algorithms
Graph searching
Random walks
Faulty networks

ABSTRACT

In this paper, we present a randomized algorithm for a *mobile agent* to search for an item stored at a node t of a network, without prior knowledge of its exact location. Each node of the network has a database that will answer queries of the form "how do I find t?" by responding with the first edge on a shortest path to t. It may happen that some nodes, called *liars*, give bad advice. We investigate a simple memoryless algorithm which follows the advice with some fixed probability $q > 1/2$ and otherwise chooses a random edge. If the degree of each node and number of liars k are bounded, we show that the expected number of edges traversed by the agent before finding t is bounded from above by $O(d+r^k)$, where d is the distance between the initial and target nodes and $r = \frac{q}{1-q}$. We also show that this expected number of steps can be significantly improved for particular topologies such as the complete graph and the torus.

1. Introduction

Searching (for information, services, etc.) is one of the most common tasks in a distributed environment. Such an environment is usually modeled by a graph where the nodes are computational devices and the edges communication links between them. The problem of searching for a specific file, service, etc. or in general a *token* in this setting, becomes that of *graph searching*.

In a distributed environment rather than considering centralized solutions to graph searching, one imagines that the search is being performed by an autonomous *mobile agent*, essentially a software entity that is able to move from node to node in the network. Depending upon the properties of the network, the properties of the agent and the information available to it, a variety of techniques have been developed to perform search. The efficiency of the search algorithm that the agent executes is usually measured by the time it takes to find the token and/or the memory requirements of the agent. For a survey of advances in mobile agent graph searching (as well as related problems) see [7].

The searching can be executed without any sort of advice as to where the token resides, or the agent may receive advice and directions from the nodes that it visits. This latter situation is by far the most common in environments like the Web in which we are mostly interested. The kind of advice that a mobile agent may receive may also vary: the agent may have a total or partial idea of the topology of the network, it may exploit some geometric properties of an embedding of the graph, etc.

☆ A preliminary version of the paper has appeared in the Proceedings of the 2nd IFIP International Conference on Theoretical Computer Science, pp. 206–216. (TCS 2002), Montreal, August 25–30, 2002.

* Corresponding author.
E-mail address: dkrizanc@wesleyan.edu (D. Krizanc).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.04.034

In this paper we study the efficiency of searching where it is assumed that each of the nodes maintains information concerning the whereabouts of the token being searched for. In particular, we assume that each node of the network can provide *advice* to the searcher by revealing the first edge on a shortest path to the node containing the token. However, we also assume that certain nodes – called *liars* – may give faulty information. This may model a situation where a node is malfunctioning or its knowledge concerning the whereabouts of the token is outdated.

1.1. Related work

The problem of searching for a token in a network where the nodes provide possibly faulty advice was introduced in [6] and studied in [3–5]. In particular the model studied here was proposed and studied in [3]. In that paper, several deterministic algorithms were proposed for various network topologies. In the present paper we propose a randomized algorithm for the same problem focusing mainly on its memory requirements. In order that our solution scale to arbitrary size networks we are interested in *memoryless* algorithms, i.e., algorithms that do not maintain any state when moving from node to node. It is fairly easy to see that a memoryless algorithm must use randomization in order to avoid deadlocks in the presence of even a small number of liars. For specific topologies, our randomized algorithm is more competitive than any deterministic algorithm. For example, for the worst-case distribution of k liars, to find deterministically a token in the complete graph of n nodes, we need at least $O(k)$ steps and $O(\log k)$ bits of memory (see [3]). Using our memoryless randomized algorithm, on average $O(1)$ steps are sufficient.

1.2. Intuitive description of the algorithm

Consider a start node s and a terminal node t containing the information we are looking for and at distance d from each other. All nodes have a piece of *advice* which points to one of the edges incident to the node. Each node is either a *truth-teller* or a *liar*. The advice of a truth-teller points to an edge which is the start of a shortest path from that node to t. A liar may point to any incident edge except the start of a shortest path. We assume that t is a truth-teller and that its advice reveals that it contains the token.

In the rest of this paper we will formalize and analyze a search algorithm whose intuitive description is as follows. An agent starting from s is looking for t. Anytime the agent finds the token it will halt. Otherwise it asks the node it is currently visiting for advice, which responds by pointing to an edge incident to it. The agent does not always follow the node's advice. Instead it flips a biased coin and accepts the advice with probability q and rejects it with probability $1 - q$ in which case it also selects any of the remaining incident edges with uniform probability in the number of remaining incident edges. The above is repeated until the token is found.

1.3. Results and structure of the paper

Our main task is to analyze the probabilistic algorithm Search presented informally in Section 1.2 (and formally in Section 2.4). Our algorithm is memoryless and we are interested in the time complexity to reach the destination for the worst distribution of liars. We count each edge used by the mobile agent as a single step. We assume the majority of the nodes are truth-tellers. As a consequence, we only consider $q > 1/2$.

After introducing several preliminary issues in Section 2, in Section 3 we consider the line graph where we show the expected number of steps of our algorithm is $O(d + r^k)$, for $r = \frac{q}{1-q}$. This result is tight in that there exists a distribution of liars such that $\Omega(d + r^k)$ steps are required by our algorithm on the line. In Section 4 we show that we can generalize this result to an arbitrary graph. We use this to show that if the mobile agent has knowledge of the distance and the number of liars, the token can be reached on average in $d + O(dk/\ln(dk^{-6}))$ steps for sufficiently large d.

In Section 5, we consider specific topologies. In Section 5.1, we deal with the complete graph for which we prove that, even if the number of liars is large, the expected number of steps to reach the destination is in $O(d)$ which in this case in $O(1)$. This is in contrast to a result in [3] that shows that for a complete graph with k liars any deterministic algorithm requires $\Omega(k)$ steps. For the torus, studied in Section 5.2, we prove a lower bound of $\Omega(d + r^k)$ steps for our algorithm in the strong adversary model.

2. Preliminaries

In this section we discuss several preliminary concepts including network definitions, Markov chains, the formal model and search algorithm.

2.1. Definitions

A *distributed network* is for our purposes an undirected graph $G = (V, E)$ where V is the set of nodes and E the set of edges. The size of V is n and the diameter of G is D. There are two specified nodes in the graph, the *start node* which we denote by s and the node containing the sought after *token* which we denote by t. The distance between s and t is d.

We will consider several simple topologies. In the complete network each node is incident to all other nodes. The chain (or line) is an n-vertex graph with nodes $0, 1, \ldots, n-1$ whereby each node i except the first and the last, is connected to its predecessor $i-1$ and its successor $i+1$. A torus of $n = n_1 n_2$ nodes is a the graph obtained by the Cartesian product of two cycles of length n_1 and n_2. For convenience, we label each node $u = (i, j)$ with $i = [-\lceil n_1/2 \rceil, \lfloor n_1/2 \rfloor]$ and $j = [-\lceil n_2/2 \rceil, \lfloor n_2/2 \rfloor]$. A node $v = (i', j')$ is a neighbor of u if $|i-i'| = 1 \pmod{n_1}$ and $|j-j'| = 1 \pmod{n_2}$. The torus is an appropriate example as it is a common network architecture with symmetry that allows relatively easy calculations.

2.2. Markov chains

We use the terminology of [8] on Markov Chains and random walks. Let $P = (0 \le p_{x,y} \le 1 | x, y \in V)$ be a *stochastic* matrix, i.e., the sum of every row is equal to 1. A *discrete-time Markov Chain* on a finite set of *states* V is a sequence of random variables $V_0, V_1, \ldots$ where $V_i \in V$ such that V_{i+1} depends only on V_i and $\Pr(V_{i+1} = x | V_i = y) = p_{x,y}$. The matrix P is called the *transition probability matrix*.

A node x *leads* to a node y if $\Pr(V_j = y$, for some $j \ge i | V_i = x) > 0$. A state y is *absorbing* if y leads to no other state. The *expected hitting time* or *hitting time* $\mathbb{E}_x^y$ is the expected or mean number of steps starting from node x to reach node y. In our paper, the node t is the only absorbing state and the expected number of steps to reach t from s is denoted by $\mathbb{E}_s^t$.

We will make use of the following well-known theorem for Markov chains:

Theorem 1. *The vector of hitting times $\mathbb{E}^t = (\mathbb{E}_x^t : x \in V)$ is the minimal non-negative solution to the system of linear equations:*

$$\begin{cases} \mathbb{E}_t^t = 0 \\ \mathbb{E}_x^t = 1 + \sum_{y \neq t} p_{xy} \mathbb{E}_y^t \quad \text{for } x \in V. \end{cases}$$

2.3. Description of the model

Before the execution of the algorithm all nodes are truth-tellers. An adversary then selects k nodes out of the n and substitutes their advice with bad advice. In the *strong* model of the adversary, it can also modify the advice of each truth-teller (except t) as long as they continue to provide an edge on a shortest path. Intuitively, a strong adversary can break any co-ordination of truth-tellers that might be of help to the agent. In the *weak* adversary model, information concerning such co-ordination may be available to the agent. Once a node's advice is set it always gives the same response throughout the execution of the algorithm. The agent is unaware of the kind of node (liar or truth-teller) it is in at anytime. Thus, in the weak adversary, one could say that first the adversary selects s and the locations of the liars, and then the truth-tellers select their answers in favour of the algorithm.

2.4. Description of the algorithm

In the model above the mobile agent executes a simple, memoryless algorithm whose goal is to take advantage of the advice that it gets from a node but at the same time avoid cycles that may arise by following the advice of liars. The algorithm Search is as follows:

(1) The agent arrives at a node of degree Δ and if it discovers the token it halts. Otherwise, it asks the node for advice.
(2) The node responds by pointing to one of the edges incident to it.
(3) The agent then flips a biased coin and with probability q it follows the advice. That is, it moves to the adjacent node which is the other endpoint of the edge. If it decides not to follow the advice (an event with probability $1-q$), it selects uniformly another edge among the remaining $\Delta - 1$ incident edges and follows that edge.
(4) The above steps are repeated at the new node until the token is found.

The idea behind this algorithm is that the agent expects, as a general rule, the advice to be correct. That is, the agent assumes that the majority of the nodes are truth-tellers and that following their advice will bring the agent closer to the token. The agent however cannot trust completely the advice that it receives as this may lead to a deadlock. Consider for example a case where the endpoints of an edge are nodes pointing to each other. It is clear that at least one of the nodes is a liar and that if the agent chooses to always follow the advice it will move back and forth between these two nodes forever. By allowing the agent not to trust the advice with some positive probability, we expect that, eventually, it will be able to get out of situations like this.

We note that the actions of the algorithm SEARCH resemble those of a *biased random walk* as studied by [2]. In that case however, on each step a coin is flipped which decides whether the current node is a liar or not. Thus the number of liars is a random variable which is a function of the bias and a node can in one instance be a liar and on the next visit of the agent, a truth-teller. In our case, the number and positions of the liars remain fixed throughout the execution of the algorithm. The analysis of both cases relies on Markov chains.

3. The chain

It turns out that a very simple graph, namely the line, behaves as badly as any graph with the same number of liars. This is of no surprise as in a line all nodes between s and t – and therefore all liars – have to be visited as there is only one way to reach the token.

Consider the n-vertex line graph $0, 1, \dots, n-1$. Each node i except the first and the last, is connected to its predecessor $i-1$ and its successor $i+1$. We suppose that $0 \le s \le t$. If i is a truth-teller, $i < t$, the probability that the agent will move to node $i+1$ is $p_{i,i+1} = q$ (trust the advice) and to move to node $i-1$ is $p_{i,i-1} = 1-q$ (not trust the advice). The corresponding probabilities for i being a liar are of course $p_{i,i+1} = 1-q$ and $p_{i,i-1} = q$. We assume that $p_{0,1} = p_{n-1,n-2} = 1$.

Set $r = \frac{q}{1-q}$. Since $q > 1/2$, $r > 1$. To each edge i between node i and $i+1$, we assign a weight w_{i+1} with $w_1 = 1$ and $w_i = \frac{p_{i,i+1}}{p_{i,i-1}} w_{i-1}$. So $w_i = rw_{i-1}$ if i is a truth-teller and $w_i = \frac{w_{i-1}}{r}$ otherwise. For convenience, let $W_j = \sum_{i=0}^{j} w_i$.

The following result is given in [1, Chapter 5], and constitutes a version of Theorem 1 mentioned above:

Lemma 1. *Let $s < t$ be two nodes of the weighted line $C = (w_1, w_2, \dots, w_{n-1})$. For a random walk in C,*

$$\mathbb{E}_s^t = t - s + 2 \sum_{j=s+1}^{t} \sum_{i=1}^{j-1} w_i w_j^{-1}.$$

Our main result for the line follows from the following two lemmas.

Lemma 2. *The worst-case distribution of liars before and including s that maximizes $\mathbb{E}_s^{s+1}$ is to place them consecutively behind s.*

Proof. By Lemma 1

$$\mathbb{E}_s^{s+1} = 1 + 2\frac{W_s}{w_{s+1}}.$$

Let L (resp. L') denote a distribution of liars such that $i < s$ is a liar (resp. truth-teller) and node $i+1$ a truth-teller (resp. liar). It is easy to see that $W_s' - W_s = rw_i - w_i/r > 0$. ■

Lemma 3. *Let l_i be the number of liars between nodes 0 and i. Let β_i be the number of consecutive truth-tellers between the last liar and i. Then, for $l_i \ge \beta_i$, we have $\mathbb{E}_i^{i+1} < 1 + (\frac{6r^{l_i-\beta_i+1}}{r-1})$ and for $l_i < \beta_i$, $\mathbb{E}_i^{i+1} < 1 + \frac{6}{r-1}$. For r constant, we have $\mathbb{E}_i^{i+1} = O(1 + r^{l_i-\beta_i})$.*

Proof. Using Lemma 2, we know that $\mathbb{E}_i^{i+1}$ is maximal if and only if all the liars are consecutive. Let us consider the sequence of types of nodes composed of $\alpha+1$ truth-tellers (by definition, node 0 is a truth-teller), l_i liars and β_i truth-tellers (i included). For $\beta_i > 0$, we have:

$$\begin{aligned} W_i &= (1 + r + \dots + r^{\alpha}) + (r^{\alpha-1} + \dots + r^{\alpha-l_i+1}) + (r^{\alpha-l_i} + \dots + r^{\alpha-l_i+\beta_i-1}) \\ &= \frac{r^{\alpha+1}-1}{r-1} + r^{\alpha-l_i+1}\frac{r^{l_i-1}-1}{r-1} + r^{\alpha-l_i}\frac{r^{\beta_i}-1}{r-1}, \end{aligned}$$

and

$$\begin{aligned} \mathbb{E}_i^{i+1} &= 1 + 2\frac{W_i}{r^{\alpha+\beta_i-l_i}} \\ &= 1 + \frac{2}{r-1}(r^{1+l_i-\beta_i} - r^{l_i-\alpha-\beta_i} + r^{l_i-\beta_i} - r^{1-\beta_i} + 1 - r^{l_i-\alpha-\beta_i}) \\ &< 1 + \frac{2(1+2r^{l_i+1-\beta_i})}{r-1}. \end{aligned}$$

For the other cases, depending on the numbers of liars and truth-tellers between nodes 0 and i, an entirely similar calculation leads to the result. ■

Theorem 2. *Let $s < t$ be two nodes of the line with k liars and q be the probability of trusting the advice. Let $r = \frac{q}{1-q}$ and let d be the distance between s and t. Then the expected number of steps to reach t from s using algorithm* Search *is less than $d\left(1 + \frac{6}{r-1}\right) + \frac{6r^{k+3}}{(r-1)^3}$. If q is constant this is $O(d + r^k)$.*

Proof. By definition of the line, $\mathbb{E}_s^t = \sum_{i=s}^{t-1} \mathbb{E}_i^{i+1}$. For each $\beta \in \{1, 2, \dots, k-1\}$, all nodes i with $l_i - \beta_i = k - \beta$ are considered and their $\mathbb{E}_i^{i+1}$ is appropriately bounded. Between s and t, there are two kinds of nodes: nodes for which the liars are far from them, $\beta_i > l_i$ or close, $\beta_i \le l_i$. We consider that the effect of each node between s and t is similar to an effect of a truth-teller and for a certain number of nodes (at most $O(k^2)$), we add the effect of liars close to them. For the second type of nodes, we can prove that we have at most $\beta + 1$ nodes i for which $l_i - \beta_i = k - \beta$. Let i be a node such that $l_i - \beta_i = k - \beta$ and $\beta_i \le l_i$. Since $l_i \le k$, we have $0 \le \beta_i \le \beta$. Since the node i is completely determined by the pair (l_i, β_i), we have at most $\beta + 1$ nodes i of this kind. By Lemma 3, for these $\beta + 1$ nodes, $\mathbb{E}_i^{i+1} < 1 + (\frac{6r^{l_i-\beta_i+1}}{r-1})$. It may happen that for l_i and β_i given,

$\mathbb{E}_i^{i+1} = 1 + o(\frac{r^{l_i-\beta_i+1}}{r-1})$ (if the liars are not consecutive). We will only count the constant term with the truth-teller effect. We upper bound $\mathbb{E}_s^t$ as follows:

$$\begin{aligned}
\mathbb{E}_s^t &\leq \sum_{\beta=0}^{k-1}(\beta+1)\frac{6r^{k-\beta+1}}{r-1} + \sum_{i=s}^{t-1} 1 + \frac{6}{r-1} \\
&\leq \frac{6}{r-1}\left(\sum_{\beta=0}^{k-1}\sum_{i=0}^{\beta} r^{i+2}\right) + (t-s)\left(1+\frac{6}{r-1}\right) \\
&\leq \frac{6}{r-1}\left(\sum_{\beta=0}^{k-1} r^2\frac{r^{\beta+1}-1}{r-1}\right) + (t-s)\left(1+\frac{6}{r-1}\right) \\
&\leq \frac{6r^2}{(r-1)^2}\left(r\frac{r^k-1}{r-1} - k\right) + (t-s)\left(1+\frac{6}{r-1}\right) \\
&< d\left(1+\frac{6}{r-1}\right) + \frac{6r^{k+3}}{(r-1)^3}. \qquad \blacksquare
\end{aligned}$$

Theorem 3. *For r constant and for the worst distribution of k liars in the line, $\mathbb{E}_s^t = \Omega(d + r^k)$.*

Proof. Take a distribution of k consecutive liars that include s and apply Lemma 1. $\blacksquare$

4. Arbitrary graphs

Consider an arbitrary network $G = (V, E)$. As usual, t denotes the node where the token resides and s the initial node. First, we prove that we reach the destination in linear number of steps with respect to the distance between s and t if the network does not contain a liar.

Proposition 1. *Let s, t be two nodes of an arbitrary graph G and constant $q > 1/2$ be the probability of trusting an advice. If G is without liars, the expected number of steps to reach t from s using algorithm* Search *is $\mathbb{E}_s^t \leq \frac{d}{2q-1}$ steps where d is the distance between s and t.*

Proof. Let X_i be a random variable associated with step i of the algorithm. $X_i = -1$ if the mobile agent goes closer to the destination and $X_i = +1$ otherwise. Let $S_i = d + X_1 + X_2 + \cdots + X_i$, i.e., S_i is the distance from the token. Let $\mathcal{E}_i$ be the event the mobile agent trusts an advice at time i and $\overline{\mathcal{E}_i}$ its complement. Let $\delta \geq 1$ be the degree of the current node x. Let $\alpha \geq 1$ be the number of edges of x on a shortest path between x and the token t. Using Search,

$$\begin{aligned}
\mathbb{P}(X_i = -1) &= \mathbb{P}(X_i = -1|\mathcal{E}_i)\mathbb{P}(\mathcal{E}_i) + \mathbb{P}(X_i = -1|\overline{\mathcal{E}_i})\mathbb{P}(\overline{\mathcal{E}_i}) \\
&= q + \tfrac{\alpha-1}{\delta-1}(1-q).
\end{aligned}$$

It follows that $\mathbb{P}(X_i = +1) = (1-q)\frac{\delta-\alpha}{\delta-1}$ and the expectation of X_i, $\mathbb{E}[X_i] = -\mathbb{P}(X_i = -1) + \mathbb{P}(X_i = +1) = \frac{(\delta-1)(1-2q)+2(1-\alpha)(1-q)}{\delta-1} \leq 1-2q$. This implies that $\mathbb{E}[S_i] \leq d - i(2q-1)$ and $\mathbb{E}[S_i]$ reaches the value 0 in at most $i = \frac{d}{2q-1}$ steps. $\blacksquare$

We now consider the case of $k > 0$ liars. Starting from t, arrange all nodes of G in layers according to their distance from t. This is easily done by executing a breadth-first search starting from t. Denote by L_i, $i = 0, \ldots, m$ the set of nodes that are at distance i from t. Here, m is the maximum distance between t and any other node. Construct the following Markov chain (Q_j), $j = 1, \ldots, m$. Q has m states each corresponding to a layer of the graph. As for the transition probabilities let the probability of moving from state Q_i to state Q_j, $p_{i,j}$, be:

$$p_{i,j} = \begin{cases} q & \text{if } i = j+1 \text{ and all nodes in layer } L_i \text{ are truth-tellers;} \\ \frac{1-q}{\Delta_i - 1} & \text{if } i = j+1 \text{ and there exists at least one liar in layer } L_i; \\ 1 - p_{i,i-1} & \text{if } i = j-1; \\ 0 & \text{if } |i-j| \neq 1. \end{cases}$$

In the above, we denote by Δ_i the maximum degree among all nodes of layer L_i. In effect, the Markov chain that we defined has one state for each layer of the graph and its transition probability from state (layer) Q_i to state Q_{i-1} is the minimum among the probabilities to move from any node in layer L_i to a node in layer L_{i-1}, that is, one step closer to the token. The initial state of (Q) is of course the state that corresponds to the layer which includes the starting node s. This is state Q_d since we have assumed that the token is at distance d from our initial position.

By our choice of transition probabilities the event of moving closer to state Q_0 from any state of Q, is less probable than moving from any node of G to another node closer to t. It is therefore clear that the expected number of steps to reach node t when starting from s, is less than or equal to the expected number of steps to reach state Q_0 when starting from state Q_d.

The Markov chain so constructed is a line in which the distance between the initial and target nodes is d and with at most k liars. Let $\Delta = \max_i \Delta_i$. If we look back at the proof of Theorem 2 of the previous section, we see that the effect of a liar on the application of Lemma 1 is to divide the weight of the next edge by r. In the case of the current chain (Q) the parameter r is always less than $(\Delta - 1)\frac{q}{1-q}$. The above observation along with Theorem 2 gives:

Theorem 4. *Let G be any network of maximal degree Δ with k liars in which the distance between the initial node s and the token t is d. Then the expected number of steps of a mobile agent to reach t is less than $d\left(1 + \frac{6}{r-1}\right) + \frac{6r^{k+3}}{(r-1)^3}$ where $r = (\Delta - 1)\frac{q}{1-q}$.*

Corollary 1. *Let s, t be two nodes of an arbitrary graph G and $q > 1/2$ be the probability of trusting an advice. If G consists only of truth-tellers, the expected number of steps to reach t from s using algorithm* Search *is $\mathbb{E}_s^t = O(d)$ where d is the distance between s and t.*

4.1. Tuning the mobile agent

As the algorithm search is memoryless, the only parameter that can be adjusted to improve performance is the probability q with which it accepts the advice of a node. Even so, there can be many different versions of this problem depending on what we might consider known to the agent. Interesting choices include the distance, the number of liars, the distribution of liars, the location of the initial node in the network, the topology etc. As an easy example, we we assume that the distance and the number of liars are known to the agent, i.e., the value of q can be set with these values in mind.

Theorem 5. *Let G be any network with k liars in which the distance between the initial node s and the token t is d. For $d > \frac{(k+3)^6}{2}$, taking $r = 1 + \frac{\ln(2d/(k+3)^6)}{2(k+3)}$, the expected number of steps of a mobile agent to reach t is $d + O(dk/\ln(dk^{-6}))$.*

Proof. Take $r = 1 + \frac{\ln(2d/(k+3)^6)}{2(k+3)}$. Then we have, from Theorem 2 that

$$\begin{aligned}
\mathbb{E}_s^t &\le d + \frac{6d}{r-1} + O\left(\frac{r^{k+3}}{(r-1)^3}\right) \\
&\le d + \frac{12d(k+3)}{\ln(2d/(k+3)^6)} + O\left(\left(1 + \frac{\ln(2d/(k+3)^6)}{2(k+3)}\right)^{k+3} \cdot \left(\frac{2(k+3)}{\ln(2d/(k+3)^6)}\right)^3\right) \\
&= d + O(dk/\ln(dk^{-6})).
\end{aligned}$$

The last equation follows from the fact that asymptotically in k the first factor inside the O is easily seen to be bounded above by $\exp[\ln(2d/(k+3)^6)/2]$ which in turn is at most $O(d^{1/2}/(k+3)^3)$. ■

5. The complete graph and the torus

The above disappointing bound of $O(d + r^k)$ comes from the fact that there may exist a bottleneck between the initial node and the token where all k liars may reside. If however the topology of the graph allows multiple paths, then things can become much better as for example in the complete graph. Moreover multiple paths give rise to new interesting problems as a truth-teller may now have a choice of shortest paths to point to. We show, by giving a specific example in the torus, that different advice from the truth-tellers can make a difference in the expected number of steps for search.

5.1. Complete graph

For the complete graph, we prove that, even when the number of liars is large, the expected number of steps to reach the destination is a constant.

Theorem 6. *Assume that in K_n (the complete graph with n nodes) the number of liars is $k = cn$, where c is a constant, $0 < c < 1$. Then the expected number of steps to reach the token is $\frac{1}{(q-qc)(1-q)} + O(\frac{1}{n})$.*

Proof. Let us compute the expected number of steps for the worst distribution of k liars. We claim that the worst distribution is obtained when each liar returns as advice another liar. Indeed, it is easy to show that starting from any node s_0 which gives as advice node s_1, the probability that the token is reached after 2 steps is smallest whenever both s_0 and s_1 are liars.

Therefore, for this distribution, the behavior of the mobile agent in K_n can be described by a random walk in a triangle T with three nodes: truth-teller, liar and token, denoted $\mathcal{TT}$, $\mathcal{L}$ and t. T contains two self-loops: the edges $(\mathcal{L}, \mathcal{L})$ and

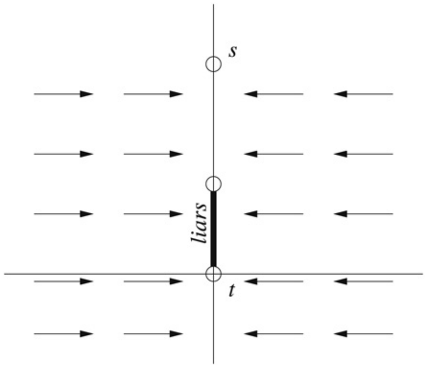

Fig. 1. Distribution of liars and advice on a torus in Case 1. The starting node s is located at $(0, d)$, the token at $(0, 0)$ and the k liars at $(0, 1), (0, 2), \ldots, (0, k)$.

$(\mathcal{TT}, \mathcal{TT})$ since in K_n, we have the possibility to go from a liar (respectively truth-teller) to a liar (resp. truth-teller). The probabilities between these 3 nodes are easily seen to be:

$$\begin{cases} p_{t,t} = 1 \\ p_{t,\mathcal{L}} = p_{t,\mathcal{TT}} = 0 \\ p_{\mathcal{TT},t} = q + \frac{1-q}{n-1} \\ p_{\mathcal{TT},\mathcal{TT}} = \frac{(1-q)(n-k-2)}{n-1} \\ p_{\mathcal{TT},\mathcal{L}} = \frac{(1-q)k}{n-1} \\ p_{\mathcal{L},t} = \frac{1-q}{n-1} \\ p_{\mathcal{L},\mathcal{TT}} = \frac{(1-q)(n-1-k)}{n-1} \\ p_{\mathcal{L},\mathcal{L}} = q + \frac{(1-q)(k-1)}{n-1}. \end{cases}$$

Using Theorem 1, we prove the maximal value for $\mathbb{E}_s^t$ is $\mathbb{E}_{\mathcal{L}}^t$ and whenever $k = cn$ for $0 < c < 1$ and for n large enough, we have $\mathbb{E}_{\mathcal{L}}^t = \frac{1}{(q-qc)(1-q)} + O(\frac{1}{n})$. Indeed, using Theorem 1, we have in T, knowing that $\mathbb{E}_t^t = 0$,

$$\begin{cases} \mathbb{E}_{\mathcal{TT}}^t = 1 + p_{\mathcal{TT},\mathcal{TT}}\mathbb{E}_{\mathcal{TT}}^t + p_{\mathcal{TT},\mathcal{L}}\mathbb{E}_{\mathcal{L}}^t \\ \mathbb{E}_{\mathcal{L}}^t = 1 + p_{\mathcal{L},\mathcal{TT}}\mathbb{E}_{\mathcal{TT}}^t + p_{\mathcal{L},\mathcal{L}}\mathbb{E}_{\mathcal{L}}^t. \end{cases}$$

Substituting the probabilities and solving we obtain:

$$\begin{cases} \mathbb{E}_{\mathcal{TT}}^t = \frac{n(n-1)}{n+qn^2-qnk-2\,qn+qk} \\ \mathbb{E}_{\mathcal{L}}^t = \frac{(n-1)(n-q)}{\left(n+qn^2-qnk-2\,qn+qk\right)(1-q)}. \end{cases}$$

The maximal value for $\mathbb{E}_s^t$ is $\mathbb{E}_{\mathcal{L}}^t$. Whenever $k = cn$ for $0 < c < 1$ and for n large enough, we have $\mathbb{E}_{\mathcal{L}}^t = \frac{1}{(q-qc)(1-q)} + O(\frac{1}{n})$. ∎

Corollary 2. *In the complete network K_n containing $\Theta(n)$ truth-tellers, the expected number of steps to reach the destination is $O(1)$.*

5.2. Multiple paths: The case of the torus

The effect of liars on the required time to reach the token may sometimes be affected by the advice of the truth-tellers. This can happen in cases where multiple shortest paths exist from some nodes to the token and consequently multiple possibilities exist for the advice that the agent receives from a truth-teller. The purpose of this section is to demonstrate this by studying the running time of the algorithm for a specific setup on the torus.

Consider the following situation: Graph G is a torus of D columns and N/D rows with N/D larger than the distance d between the starting node and the token. The token is placed at the origin $(0, 0)$ while the k liars occupy the nodes from $(0, 1)$ to $(0, k)$. The starting node s is the node $(0, d)$. We distinguish two cases:

Case 1. All truth-tellers with coordinates (x, y) point left if $x > 0$ and right if $x < 0$. They point to the appropriate (unique) direction if they lie on the y axis (see Fig. 1).

Comments: This case is a lower bound for this kind of adversary.

Case 2. All truth-tellers with $y \neq 0$ point either up or down depending on their position. That is, all truth-tellers with $y > 0$ point down and all truth-tellers with $y < 0$ point up. They point to the appropriate direction if they lie on the x axis. (see Fig. 2).

We next study the expected number of steps to reach the token in each of the above cases. We have the following result:

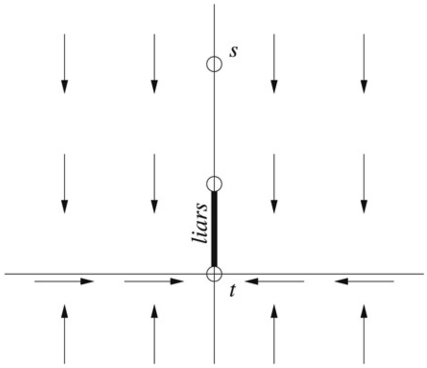

Fig. 2. Distribution of liars and advice on a torus in Case 2. The starting node s is located at $(0, d)$, the token at $(0, 0)$ and the k liars at $(0, 1), (0, 2), \ldots, (0, k)$.

Theorem 7. *If all truth-tellers with coordinates (x, y) point left if $x > 0$ and right if $x < 0$ then the expected number of steps to reach the token is $\Omega(d + r'^k)$, where r' is a constant depending on q. In contrast, when all truth-tellers with $y \neq 0$ point either up or down depending on their position, then the expected number of steps is proportional to d.*

Proof. First we outline the proof of the two cases. In *Case* 1 we show, by first studying the horizontal and then the vertical motion of the agent, that for $q > 1/2$, more than two-thirds of the vertical moves of the agent are on the y axis. Then the time to reach the token is bounded from below by the time of an agent that moves only on the y axis but with the biasing probability of the liars (and of the truth-tellers) now reduced to a smaller value q'. Recalling Theorem 3 for $r' = (1 - q')/q'$ substituting r, completes the proof of Case 1. In *Case* 2 the horizontal motion of the agent is now a pure random walk, i.e., the agent moves to the left and to the right with same probability $(1-q)/3$. By taking a torus with D columns and N/D rows, it is easy to see that almost all the time the agent occupies nodes with $x \neq 0$. The effective motion of the agent is therefore as on a line with only truth-tellers. Hence the time to reach the token is proportional to d. Next we complete the details.

Case 1. Consider the horizontal (left–right) movement of the agent, disregarding any vertical movement. We would like to know the expected number of steps that the agent visits nodes with $x \neq 0$ before returning to the origin. By symmetry, this is equivalent to the random walk on the half-line with starting point the origin, 0. The probability to move closer to the origin is $q_1 = q/(q + (1 - q)/3) = 3q/(1 + 2q)$, since we are only considering the horizontal movement. The probability to move away from the origin is of course $1 - q_1$.

Now the expected number of nodes with positive x coordinate that the agent visits before returning to 0, is the solution of the equation:

$$x_h = \left(\frac{1}{q_1} - 1\right)(x_h + 1) + 1.$$

This follows from the fact that the agent first moves to the node next to the origin from where it will return the first time an experiment with probability q_1 turns out true. This is a geometrically distributed event and therefore its average value is $1/q_1$. It follows that the agent for an average of $1/q_1 - 1$ times goes to the opposite direction from the origin and visits $x_h + 1$ nodes on average each time. Hence $x_h = 1/(2q_1 - 1) = (1 + 2q)(4q - 1)$. Returning to our two-dimensional scene, while the agent visits a node with $x \neq 0$, it may also do a number of vertical moves with probability $2(1 - q)/3$. The expected number of vertical moves that it does before changing its x coordinate, i.e., before doing a horizontal move, is $3/(2q + 1) - 1 = (2 - 2q)/(1 + 2q)$. Combining this with the value of x_h we get that the expected number of vertical moves that the agent does between two consecutive visits to the y axis is: $V_{x\neq 0} = (2-2q)(4q-1)$. Similarly, the expected number of the agent's vertical moves on the y axis, between its arrival and its departure is $V_{x=0} = 3/(2(1-q)) - 1 = (1+2q)/(2(1-q))$. Therefore the fraction of the vertical moves of the agent not on the y axis with respect to those on the y axis is:

$$\lambda = \frac{V_{x\neq 0}}{V_{x=0}} = \frac{4(1 - q)^2}{(4q - 1)(1 + 2q)}.$$

Observe that for $q > 1/2$ this is less than 1/2 which means that the agent does more than twice the number of vertical moves on the y axis than on the rest of the plane. Hence we may view the walk of the agent as restricted to the y axis only, but with the additional feature that after moving to a node it will first flip a biased coin and with probability λ it will decide whether its next move will be unbiased, i.e., it will move up or down with probability 1/2 or biased according to whether its current node is a truth-teller or a liar. Notice that this is a simplification; the actual expected number of steps can be worse. For example, in the real walk there are cases where the agent moves more than one step away from the y axis and it is thus impossible for its next vertical move to be on the y axis.

With this in mind, we see that in effect, the agent moves down with probability $p_t = \frac{1-\lambda}{2} + \lambda q$, if the current node is a truth-teller and $p_l = \frac{1-\lambda}{2} + \lambda(1 - q)$, if it is a liar. Notice that probability p_t is still greater than 1/2 but less than q. That is, the movement of the agent on the y axis is as on a line with k liars (Section 4) but with the biasing probability q, now reduced to a smaller value q'. Recalling Theorem 3 therefore completes the proof of Case 1 with $r' = (1-q')/q'$ substituting r.

Case 2. Take a Markov Chain $(Q_i)_{i\geq 0}$ defined on the position of the mobile agent. We denote by $Q_t = (x_t, y_t)$ the position of the mobile agent at time t. Consider two generic states "$x = 0$" and "$x \neq 0$". We shall show that the mobile agent stays in state "$x = 0$" during a constant number of steps on average whereas it will stay $\Theta(D)$ number of steps on average in state "$x \neq 0$". In state "$x \neq 0$", all nodes are truth-tellers and the mobile agent decreases its distance toward the destination by at least $\frac{D}{2q-1}$ units on average applying Proposition 1. In state $x = 0$, the mobile agent can only increase its distance of $O(1)$ units to the target whatever is the status of the visited nodes (liars or truth-tellers). Since the movement of the mobile agent corresponds to alternate states "$x = 0$" and "$x \neq 0$", by averaging the distance toward the destination, we get the result. To sum up, the influence of the liars is negligible.

For any position, the event of doing an horizontal movement is geometrically distributed with parameter $2(1-q)/3$. It follows that the expected time between two horizontal moves is $3/(2-2q)$ and that the mobile agent stays in state "$x = 0$" during $3/(2-2q)$ steps.

Once $|x_t| = 1$, we can compute the return time to the y axis. Consider the Markov Chain $(X_i)_{i\geq 0}$ whose states correspond to the set of distances from the y axis (assuming for convenience that the maximum distance $D/2$ is an integer). We define X_i as follows: for $X_i = x_t, X_{i+1} = x_{t'}$ for $t' > t$ being the first time step for which $x_{t'} \neq x_t$. We know that $\mathbb{E}(t') = t + 3/(2-2q)$. Applying Lemma 1 by assigning $w_i = 1$ (left and right moves occur with same probability) for a chain of $D/2$ nodes, we get that $\mathbb{E}_1^0 = 1 + 2(D/2 - 1) = D - 1$ in Markov chain (X_i). The return time to state $x = 0$ in the above Markov chain (Q_i) takes $\frac{3(D-1)}{2-2q}$ in expectation. From Proposition 1, during state $x \neq 0$, the distance toward the destination decreases by at least $\frac{3(D-1)}{2(1-q)(2q-1)}$ units on expectation. In a sequence of $x = 0$ followed by $x \neq 0$, the distance toward the destination decreases by at least $\frac{3(D-1)}{2(1-q)(2q-1)} = \Theta(D)$ on expectation for constant $1/2 < q < 1$. Therefore with the same hypothesis on q, the sequence takes $\Theta(D)$ on average. Repeating this argument $\Theta(d/D)$ times, the destination is reached in $O(d)$ expected time. ■

6. Conclusion

We have presented a "memoryless" randomized algorithm to search for an item t contained at a node of a network, without prior knowledge of its exact location and under the assumption that some nodes, called *liars*, may give bad advice. It would be interesting to study time-memory tradeoffs, as well as search algorithms for multiple mobile agent systems in our model.

Acknowledgements

The third author's research was supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems) grants.

References

[1] D.J. Aldous, J.A. Fill, Reversible Markov Chains and Random Walks on Graphs, 2002 (in preparation). Available at: http://www.stat.berkeley.edu/users/aldous/RWG/book.html.
[2] Y. Azar, A. Broder, A. Karlin, N. Linial, S. Phillips, Biased random walks, Combinatorica 16 (1996) 1–18.
[3] N. Hanusse, E. Kranakis, K. Krizanc, Searching with mobile agents in networks with liars, Discrete Applied Mathematics 137 (2004) 69–85.
[4] A. Kaporis, L.M. Kirousis, E. Kranakis, D. Krizanc, Y. Stamatiou, E. Stavropoulos, Locating information with uncertainty in fully interconnected networks with applications to world wide web retrieval, Computer Journal 44 (2001) 221–229.
[5] L.M. Kirousis, E. Kranakis, D. Krizanc, Y. Stamatiou, Locating information with uncertainty in fully interconnected networks, Networks 42 (2003) 169–180.
[6] E. Kranakis, D. Krizanc, Searching with uncertainty, in: Proc. SIROCCO'99, 1999, pp. 194–203.
[7] E. Kranakis, D. Krizanc, S. Rajsbaum, Computing with mobile agents in distributed networks, in: S. Rajasedaran, J. Reif (Eds.), Handbook of Parallel Computing: Models, Algorithms, and Applications, CRC Press, 2007.
[8] J.R. Norris, Markov Chains, Cambridge University Press, Cambridge, England, 1997.