MEMORYLESS SEARCH ALGORITHMS IN A NETWORK WITH FAULTY ADVICE

Nicolas Hanusse*

LaBRI - CNRS - Université Bordeaux I, 351 Cours de la Liberation, 33405 Talence, France.

hanusse@labri.fr

Dimitris Kavvadias

Department of Mathematics, University of Patras, Rio, Greece.

kavadias@ceid.upatras.gr

Evangelos Kranakis†

Carleton University, School of Computer Science, Ottawa, ON, K1S 5B6, Canada.

kranakis@scs.carleton.ca

Danny Krizanc‡

Department of Mathematics, Wesleyan University, Middletown CT 06459, USA.

dkrizanc@caucus.cs.wesleyan.edu

Abstract In this paper, we present a randomized algorithm for a *mobile agent* to search for an item t stored at a node 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 get to 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

*Research supported in part by CNRS-Jemstic "Mobicoop" grant.

†Research supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems) grants.

‡Research supported in part by NSERC and MITACS grants.

edges to follow in order to reach t is bounded from above by $O(d+r^k)$, where d is the distance between the initial and target node 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, the torus, and the spider graph.

Keywords: Data Structure and Algorithms, Distributed Computing, Randomized Algorithms, Graph , Random Walks, Faulty Networks

1. Introduction

Searching for a piece of information is one of the most common tasks in a distributed environment. The evolution of the World Wide Web, for example, has led to the development and the wide use of specialized sites called search engines, that are able to provide the user with information on almost any possible query. Therefore, seeking a piece of information in a distributed environment is an algorithmic problem of major significance.

The usual mathematical model of such an environment is a graph (directed or undirected) whose nodes are computers that perform routing of message packets and/or store information. This information can be retrieved by an interrogating program that can be seen as an agent (*mobile agent*) that visits the nodes of the graph in a certain way moving from node to node using the edges. The purpose of the mobile agent is to locate a specific piece of information or *token* that resides in a certain node [KK99, KKKS00].

Graph searching is a central procedure in many computational problems. There are numerous techniques and algorithms for this problem that vary from purely deterministic (e.g., depth or breadth-first search) to purely randomized, (e.g. random walks, [AKL$^+$79, KR95]). A variety of versions of the problem have been proposed that capture the requirements of different problem settings. For example, the agent may have limited knowledge of the topology of the network (e.g., searching in an unknown graph [MHG$^+$88]) or it may exploit some geometric properties of the network (e.g., searching in the plane [BYCR93]). The efficiency of the search algorithm that the agent executes is measured, as usual, by the time it takes to find the token and/or its memory requirements.

In this paper we study the efficiency of searching algorithms where the advice is based on *shortest path* information that an agent receives when it arrives at a node. That is, the computer that resides at the node advises the agent by revealing to it an edge that is the beginning of a shortest path from the current node to the token. This kind of advice is common in the IP protocol where routing is done usually through the shortest available path. However, as is common in practice, we also assume that certain nodes–called *liars*–may give faulty information. This may model a situation where a computer is malfunctioning or its knowledge about the topology of the graph and/or the position of the token, is outdated. The searching problem in the presence of liars has been proposed and studied in [HKK00]. 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 the memory requirements. The reason for this is simple: the size of a network such as the Web is such that any agent that accumulates information in its traversal of the network can end up having devastating memory requirements. We are therefore interested in *memoryless* algorithms. It is easily seen however, that any memoryless algorithm that only receives shortest path information has to use randomization in order to avoid deadlocks. 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. Using our memoryless randomized algorithm, on average $O(1)$ steps are sufficient.

1.1. Definitions

A *network* is for our purposes an undirected graph $G = (V, E)$ where V is the set of nodes and E the set of edges. There are two specific nodes in the graph, the *start node* which we denote by s and the *token* which we denote by t. All nodes point to an edge incident to them. Each node is either a *truth–teller* or a *liar*. A node of the former kind points to an edge which is the start of a shortest path from that node to the token. A node of the latter kind may point to any incident edge except the start of a shortest path.

In our model, once a node is characterized it always gives the same response throughout the execution of the algorithm. 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. We will be working mainly in the *strong adversary* model in which the adversary can also modify the advice of each truthteller as long as they continue to provide an edge on a shortest path. We show in Section 4.2 a spectacular difference between a weak and a strong adversary model whenever there are many paths to go to the destination. Intuitively, a strong adversary can break any coordination of truth-tellers that might be of help to the agent. The agent is unaware of the kind of node in which it arrives.

Parameters that will be used are the total number of liars k and the initial distance d between s and t. We shall also be interested in the topology of the graph and in its diameter which we denote by D.

1.2. Markov Chains and zero memory algorithms

We use the terminology of [Nor97] of Markov Chains and random walks. Let $P = (0 \leq p_{x,y} \leq 1 | x, y \in V)$ 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 $V_i \in V$ and 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.* In the context of a biased random walk in a graph, $p_{x,y}$ denotes the probability to go from node x to node y in one step.

A node x *leads* to a node y if $\Pr(V_j = y \text{ for some } j \geq i | V_i = x) > 0$. A state y is *absorbing* if y leads to no other state. The *expected hitting time* or *hitting time*[1] $\mathbb{E}_x^y$ is the expected or mean number of steps starting from node x to reach node y. In our paper, the token t is the only absorbing state and the expected number of steps to reach t from s is noted $\mathbb{E}_s^t$.

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:*

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

1.3. 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 in a node but at the same time avoid deadlocks that may arise in some cases. The algorithm SEARCH is as follows:

1 The agent arrives at a node of degree, say, 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 of probability $1 - q$), it selects uniformly another edge among the remaining $\Delta - 1$ incident edges.

4 It then moves to the other endpoint of the selected edge.

5 The above steps are repeated at the new node.

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 faster to the token. Therefore the probability q is taken to be more than $1/2$. 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 resembles that of a *biased random walk* as studied by Azar et al. [ABK$^+$96]. In that case however, on each step a coin is flipped which determines whether the current node 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 on the next visit of the agent, a truthteller. In our case, the number and positions of the liars remains fixed throughout the execution of the algorithm. The analysis of both cases relies on Markov chains.

1.4. Results and structure of the paper

Our main task is to analyze the probabilistic algorithm SEARCH presented in Section 1.3. 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 consider that $q > 1/2$.

In Section 2, we consider the chain graph where we show the expected number of steps of our algorithm is $O(d + r^k)$, for $r = \frac{q}{1-q}$. In Section 3 we show that we can generalize this result to an arbitrary graph. We use this to show that if the mobile agent has approximate knowledge of the distance and the number of liars, the token can be reached on average in $d + O(dk/\ln(dk^{-6}))$ steps.

In Section 4, we consider specific topologies. In Section 4.1, 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 a constant. However, in the complete graph, the distance is the constant 1. In the torus, studied in Section 4.2, we prove a lower bound of $\Omega(d + r^k)$

steps in the strong adversary model. For the same distribution of liars, the expected number of steps is $O(d)$ in the weak adversary model. In Section 4.3 we study the case of having different shortest paths to reach the token. For spider graphs of N nodes per layer, the algorithm leads to an expected number of steps of $O(d + N^2 + r^{O(k/N)})$ using the fact that the mobile agent has the possibility to avoid obstacles, i.e. liars, to reach an area with only truth-tellers.

Due to space limitations, the proofs are not given.

2. Chain

It turns out that a very simple graph, namely the chain – or line – behaves as bad as any graph with the same number of liars. This is of no surprise as in a line all nodes –and therefore all liars– have to be visited as there is only one way to reach the token. We begin our study therefore with the chain graph.

Consider the n-vertex chain graph $0, 1, \ldots, n-1$. Each node i except the first and the last, is only connected to its predecessor $i-1$ and its successor $i+1$. We suppose that $0 \leq s \leq t$. If i is therefore 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 $w_{i+1} = \frac{p_{i,i+1}}{p_{i,i-1}} w_i$. So $w_{i+1} = r w_i$ if i is a truth teller and $w_{i+1} = \frac{w_i}{r}$ otherwise. For convenience, let $W_j = \sum_{i=0}^{j} w_i$.

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

Lemma 1 (Essential edge lemma) *Let $s < t$ be two nodes of the weighted chain $C = (w_1, w_2, \ldots, w_{n-1})$. 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 chain follows from the following two lemmas, whose proof can be found in the appendix.

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

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 $l_i \geq \beta_i$, 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, it turns out $\mathbb{E}_i^{i+1} = O(1+r^{l_i-\beta_i})$.*

Theorem 2 *Let $s < t$ be two nodes of the chain 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 $O(d+r^k)$.*

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 the essential edge lemma, Lemma 1. □

3. Arbitrary Graph

In this section we study the expected number of steps to reach the token in an arbitrary graph. Let G be any graph. Starting from t, arrange all nodes of G in layers according to their distance 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 $(Q_j), j = 0, \ldots, m$. Q has $m+1$ 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: When the agent is on a layer with truth-tellers only, the probability of moving closer is of course q and moving further is $1-q$. When the agent is on a layer with at least one liar then it may happen that it resides on a node i which is actually a liar, in which case the probability of moving closer is at least $\frac{1-q}{\Delta_i - 1}$. 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 or equal than the expected number of steps to reach state Q_0 when starting from state Q_d.

The so constructed Markov chain (Q) is a line in which the distance between the initial and target nodes is d and with at most k liars. Notice that while the truth-tellers in (Q) preserve their usual probabilities (i.e. q to go closer and $1-q$ to go furher), the liars are "stronger" in the sense that the probability of going closer is now only $p_{i,i-1} = \frac{1-q}{\Delta_i - 1}$. (Notice the change in the numbering of nodes in the current chain (Q).) Let $\Delta = \max_i \Delta_i$. Going back to the proof of Theorem 2 of the previous section, we see that the effect of a liar in the application of Lemma 1 is to divide the weight of the next edge by r. In our case $r = \frac{p_{i,i+1}}{p_{i,i-1}} = \frac{\Delta_i - 2 + q}{1-q}$. In the case of the current chain (Q) the parameter r is always less than $\frac{\Delta - 1}{1-q}$. The above observation along with Theorem 2 therefore 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 $r = \frac{\Delta - 1}{1-q}$.*

3.1. Tuning the mobile agent

As algorithm SEARCH is memoryless, the only parameter that can be adjusted to improve performance is the amount of "belief" of the agent to the advice it receives, that is, probability q. 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 consider the worst topology, i.e the line, the worst distribution of liars and we assume that the order of the distance and the number of liars are known.

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 $d + O(dk/\ln(dk^{-6}))$.*

4. Special Graphs

The above disapointing 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 do make a difference in the expected number of steps.

4.1. 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 $\frac{1}{(q-qc)(1-q)} + O(\frac{1}{n})$.*

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

4.2. 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. 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 $|i - i'| = 1(n_1)$ and $|j - j'| = 1(n_2)$. The torus is an appropriate example as it is a common network architecture with symmetry that allows relatively easy calculations.

Consider the following situation: Graph G is a torus with diameter D much 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.

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.

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

Theorem 7 *The expected number of steps to reach the token in Case 1 is $\Omega(d+{r'}^k)$ where r' is a constant depending on q. In contrast, in the second case the expected number of steps is proportional to d.*

4.3. Spider Graph

We now turn our attention to a particular graph which reveals some additional merits of our searching algorithm. We call this graph *spider graph* because it resembles a spider's web. A node u of the spider graph has the polar coordinates $(x,y) \in \{1,..,n/N\} \times \{0,\dots,N-1\} \cup \{(0,0)\}$ where x represents the *radius*, i.e. the distance to the token and y the angle. Node $u \neq (0,0)$ has four neighbors $(x,y+1)$, $(x,y-1)$, $(x-1,y)$ and $(x+1,y)$. The node $(0,0)$ is called the *center* of the spider graph. We assume that the token lies on the center node and is connected to N nodes $(1,0),\dots,(1,N-1)$. N is called the *density* which is the number of nodes on a cycle of the graph. Less important parameter is the number of cycles, or equivalently the number of nodes on any radius of the graph. This is because we assume, as usual, that the majority of the nodes are truth tellers and consequently far enough from the center all nodes are truth tellers. We are therefore interested in the part of the spider graph around the center and at a radius approximately the maximum distance between d (the distance of the initial node) and the distance of the furthest liar.

All truth tellers point of course at the center of the graph while we assume for simplicity that all liars point at the opposite direction. Hence with probability $p_r = q+(1-q)/3$ the agent will move along its current radius and with probability $p_c = 1-p_r = 2(1-q)/3$ it will change radius and move to a neighboring node on the current cycle. Since the probabilities to move on a cycle clockwise and counterclockwise are equal, if we ignore the moves that the agent does along a radius, its resulting motion is an unbiased random walk on a ring with N nodes. This last observation explains how the agent is able to avoid areas with high concentration of liars: It "probes" the area in front (toward the token) in an attempt to get closer to the token. We know from Section 2 that if there are k liars in front, then it will take the agent $O(r^k)$ steps to pass them. But since the agent also moves around on the current ring continuously trying to get closer to the token, it will eventually discover areas with no liars or with only a few liars and it will pass from there.

We label the set of radii from 0 to $N-1$ in the clockwise direction. We denote by (X_i, Y_i) the polar coordinates of the mobile agent at time i where X_i corresponds to the distance from the center and Y_i is the label of the current radius. By convention, $(X_0, Y_0) = (d, 0)$. The following result comes from the *mixing time* in a ring of N nodes [AF]:

Lemma 4 *If q is constant then for $i = \Omega(N^2)$ and for all $y \in [0, N-1]$, $|\Pr(Y_i = y) - 1/N| = \epsilon$.*

Let t_x be the number of truth-tellers of layer x (at distance x from the destination). Let p_x^i be the probability, knowing that $X_i = x$, to go from layer x to layer $x-1$ at time i.

Lemma 5 *After $i = \Omega(N^2)$ steps, $p_x^i > 1/2$ if $t_x > cN$ with $c = \frac{1+2q}{2(4q-1)}$.*

The next result shows that if the density of the spider graph is significant, the time to reach the token can become much better.

Theorem 8 *In a spider graph of density N, if q is constant, the expected number of steps to reach the token is upper bounded by $O(d + N^2 + r^{\lfloor \frac{2k(4q-1)}{(6q-3)N} \rfloor})$.*

5. Conclusion

We have presented "memoryless" randomized algorithms to search for an item t contained in 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. We have provided an algorithm and studied its performance in an arbitrary network and also considered different topologies like the complete graph, the torus, and the spider graph. It would be interesting to consider techniques similar to [KR95, BKRU89] and [MR95][Chapter 6] in order to study time-memory tradeoffs, as well as search algorithms for multiple mobile agent systems in our model.

Notes

1. In Norris' book [Nor97], the hitting time is the minimum amount of time needed to reach a node.

References

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

[AF] D.J. Aldous and J.A. Fill. Reversible markov chains and random walks on graphs. (book in preparation, available on the web from http://www.stat.berkeley.edu/users/aldous/book.html).

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

[BKRU89] A. Broder, A. K. Karlin, P. Raghavan, and E. Upfal. Trading space for time in undirected $s-t$ connedtivity. In *Proc. STOC*, pages 543–549, 1989.

[BYCR93] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 1993.

[HKK00] N. Hanusse, E. Kranakis, and K. Krizanc. Searching with mobile agents in networks with liars. In *Proc. EUROPAR'2000, LNCS 1900*, pages 583–590, Munich, 2000.

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

[KKKS00] L. M. Kirousis, E. Kranakis, D. Krizanc, and Y. Stamatiou. Locating information with uncertainty in fully interconnected networks. In *Proc. DISC'2000, LNCS 1914*, pages 283–296, Toledo, Spain, 2000.

[KR95] A. R. Karlin and P. Raghavan. Random walks and undirected graph connectivity: A survey. In *Discrete Probability and Algorithms, Institute of Mathematics and its Applications, Vol., 72, Springer Verlag, 1995*, pages 95–101, 1995.

[MHG+88] N. Megiddo, S. Hakimi, M. Garey, D. Johnson, and C. Papadimitriou. The complexity of searching a graph. *Journal of the ACM*, 1988.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[Nor97] J.R. Norris. *Markov Chains.* Cambridge University Press, 1997.

Appendix

This appendix contains details and calculations of some of our proofs to help the reader.

Proof of Lemma 2:

Proof.

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

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's easy to see that $W'_s - W_s = rw_i - w_i/r > 0$. □

Proof of Lemma 3:

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 type 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:

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

For the other cases, a similar calculation leads to the result. □

Proof of Theorem 2:

Proof. By definition of the line, $\mathbb{E}_s^t = \sum_{i=s}^{t-1}\mathbb{E}_i^{i+1}$. 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, $\beta_i \leq l_i$. From Lemma 3, a node s' of the first kind is so that $\mathbb{E}_{s'}^{s'+1} < 1 + \frac{6}{r-1}$. For a node s' of the second kind, $\mathbb{E}_{s'}^{s'+1} < 1 + \frac{6r^{l_i-\beta_i+1}}{r-1}$. We can prove that we have at most $\beta+1$ nodes i of the second kind for which $l_i - \beta_i = k - \beta$. Let i be a node such that $l_i - \beta_i = k - \beta$ and $\beta_i \leq l_i$. Since $l_i \leq k$, we have $0 \leq \beta_i \leq \beta$. 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. From Lemma 3, for these $\beta+1$ nodes, $\mathbb{E}_i^{i+1} = 1 + O(\frac{r^{k-\beta+1}}{r-1})$.

We upper bound $\mathbb{E}_s^t$ claiming:

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

□

Proof of Theorem 5:
Proof. Take $r = 1 + \frac{\ln(2d/(k+3)^6)}{2(k+3)}$. Then we have, from Theorem 2 that

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

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 $O(d^{1/2}/(k+3)^3)$. This proves the theorem. □

Proof of Theorem 6:
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, we can prove 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 behaviour 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 $(\mathcal{TT},\mathcal{TT})$ 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: $p_{t,t}=1$, $p_{t,\mathcal{L}}=p_{t,\mathcal{TT}}=0$, $p_{\mathcal{TT},t}=q+\frac{1-q}{n-1}$, $p_{\mathcal{TT},\mathcal{TT}}=\frac{(1-q)(n-k-2)}{n-1}$, $p_{\mathcal{TT},\mathcal{L}}=\frac{(1-q)k}{n-1}$, $p_{\mathcal{L},t}=\frac{1-q}{n-1}$, $p_{\mathcal{L},\mathcal{TT}}=\frac{(1-q)(n-1-k)}{n-1}$, $p_{\mathcal{L},\mathcal{L}}=q+\frac{(1-q)(k-1)}{n-1}$.

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 $\mathbb{E}_{\mathcal{L}}^t=\frac{1}{(q-qc)(1-q)}+O(\frac{1}{n})$.

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:

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

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 $\mathbb{E}_{\mathcal{L}}^t=\frac{1}{(q-qc)(1-q)}+O(\frac{1}{n})$. □

Proof of Theorem 7:

Proof. *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 does on 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:

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

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 on the y axis only, but with the additional feature that after moving at 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 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 3) but with the biasing probability q, now reduced to a smaller value q'. Recalling Theorem 2 therefore completes the proof of Case 1 with $r' = (1-q')/q'$ substituting r.

Case 2. Considering again first the horizontal movement of the agent, it is clear that now it is pure random walk, i.e. the agent moves to the left and to the right with probability $1/2$. Based on that it is easy to see that with probability asymptotically close to 1, the agent will at some point leave the y axis. But then the expected number of steps to return to it can be made arbitrarily large by taking a torus with large enough diameter D. It follows that the agent almost all the time moves at nodes with $x \neq 0$. The vertical therefore movement of the agent is that on a line with only truth tellers and as a result the agent will reach the x axis in expected time which is a small constant times the distance d. It will then reach the origin $(0,0)$ moving almost always on the x axis in time again a small constant times the distance d. It follows that in this case the expected number of steps is proportional to d. □

Proof of Lemma 4:

Proof. This result comes from the *mixing time* in a ring of N nodes [AF]. In the case of a pure random walk in the ring, after $\frac{N^2}{2\pi^2}$ steps, the probability of being at a given node is $1/N + \epsilon_c$ with $|\epsilon_c| = o(1/N)$. It follows that the expected number of steps of the mobile agent to reach any radius with almost the same probability is $\frac{N^2}{2\pi^2 p_c}$. □

Proof of Lemma 5:

Proof. Let $\mathcal{E}_x^i$ be the event of going closer given that at time i we are at layer x. Also let $\mathcal{L}_{x,y}$ be the event that node (x,y) is a liar. We have $p_x^i = \sum_{y=0}^{N-1} \Pr(\mathcal{E}_x^i \cap \{Y_i = y\})$. It follows that:

$$p_x^i = \sum_{y=0}^{N-1} \Pr(\mathcal{E}_x^i \cap \mathcal{L}_{x,y} \cap \{Y_i = y\}) + \Pr(\mathcal{E}_x^i \cap \overline{\mathcal{L}_{x,y}} \cap \{Y_i = y\}) \text{ and}$$

$$\begin{aligned} p_x^i &= \sum_{y=0}^{N-1} \Pr(\{Y_i = y\})((\Pr(\mathcal{E}_x^i | \mathcal{L}_{x,y} \cap \{Y_i = y\}) \Pr(\mathcal{L}_{x,y} | \{Y_i = y\}) + \\ &\quad \Pr(\mathcal{E}_x^i | \overline{\mathcal{L}_{x,y}} \cap \{Y_i = y\}) \Pr(\overline{\mathcal{L}_{x,y}} | \{Y_i = y\})) \end{aligned}$$

Lemma 4 now allows us to ignore the specific radius y that we are at. (Since after $O(N^2)$ steps we can be at any radius with almost the same probability.) Therefore we have:

$$p_x^i \geq (1 - o(1)) \left(\frac{(1-q)(N - t_x)}{3N} + \frac{q t_x}{N} \right)$$

Hence $p_x^i > 1/2$ if $(1 - o(1)) \frac{t_x(4q-1) + (1-q)N}{3N} > 1/2$. It turns out if $t_x > \frac{1+2q}{2(4q-1)} N$, we have $p_x^i > 1/2$. □

Proof of Theorem 8:

Proof. Let us construct a chain C of nodes $0, 1, ..., n/N$ and set the token t in node 0. The idea consists in simulating our random walk in the spider graph by a random walk in the chain C where each layer is associated to a "light" truth-teller or a "light" liar depending on the majority (in the sense of Lemma 5) of truth-tellers at each layer.

For $i = O(N^2)$, in C set $p_{x,x-1} = p_x^i$ and $p_{x,x+1} = 1 - p_x^i$. We upper bound the expected number of steps by observing: (1) After $O(N^2)$ steps, we can be in any radius with the same probability; (2) After $O(N^2)$ steps we can be at a distance at most $d' = d + O(N^2)$ from t and (3) The expected number of steps is upper bounded by the expected number of steps of a random walk in C starting from node d'. This scenario is pessimistic since it's like considering that in the spider graph if we don't go closer, we go further. For this reason, after $O(N^2)$ steps in the spider graph, we claim the behaviour of the random walk is "better" for our goal than a random walk in C. From Lemma 5, a liar in C is associated to a layer in the spider graph with at least $(1-c)N = \frac{6q-3}{2(4q-1)}N$ liars. A truthteller is of course associated to any other layer. It follows that the number of liars k' in C is upper bounded by $\lfloor \frac{k}{(1-c)N} \rfloor$. Using Theorem 2, if q is constant, we obtain that the expected number of steps of the spider graph is upper bounded by $O(N^2 + d' + r^{\lfloor \frac{k}{(1-c)N} \rfloor})$. $\square$