B.Tech. Project Report

on

Edge exploration of port labeled graph with advice

Submitted by

Thota Goutham

201451034

under the supervision of

Dr.Barun Gorain

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY VADODARA

2017

Declaration

I, Thota Goutham, declare that this written submission represents my ideas in my own words and where other′s ideas or words have been included, I have adequately cited and referred the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not mispresented or fabricated any idea/data/fact/source in my submission. I fully understand that any violation of the above will cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained.

(Thota Goutham)

(Thota Goutham)

Date:26/05/18

201451034

Abstract

We consider the problem of deterministic edge exploration of graphs. An n-node graph has unlabeled nodes and all ports at any node of degree d are arbitrarily numbered from $0...., d-1$. A mobile agent, situated at the start at some node v, must visit all the edges and stop. The exploration time is defined as the number of edges traversed in this process. Some apriori information is necessary, which is given by a map oracle as a binary string. The map oracle knows the port-numbered map of the graph but not the starting node. In this report we investigate the following problem- What is the minimum size of advice that permits the exploration of all the edges of a graph in time T.

We prove that with advice of size $O(n \log n)$, edge exploration can be done in $O(n^3)$ time. For the lower bound, we show that even if the entire map of the graph is provided to the agent as advice, the exploration time of any algorithm is $\Omega(n^{8/3})$.

Contents

1 Introduction **1**

2 Related work **2**

3 Previous work **3**

4 Results **4**

5 Technical Contribution **5**

5.1 Algorithm which takes time $O(n^3)$ 5

5.2 Lower Bound 5

6 Summary and Conclusions **10**

6.1 Scope for future work. 10

Chapter 1

Introduction

We consider a simple connected undirected graph $G = (V, E)$ with n nodes. Nodes have no labels, the ports at any node of degree d are arbitrarily numbered from $0, ..., d-1$. The agent starts at some node v and when it gets to another node u it learns the port j which it has used to enter u along with the degree of u.

It can be easily observed that some apriori knowledge is needed by the agent for traversing all the edges. Consider the class of oriented rings: Rings in which ports at all nodes are numbered $0, 1$ in clockwise order, specifically a ring which takes t steps for edge exploration and another ring which has $t + 1$ edges. The agent cannot distinguish between these two rings as whenever it enters a node it learns that the node has degree 2 and the port number used for entering is 0. Hence it will stop in the larger ring after t steps only and traversal fails

As is the case with algorithms with advice, this apriori information, necessary for edge exploration is provided to the agent by an oracle as a binary string whose length is called the size of advice. For our problem the map oracle is considered which is aware of the port-numbered map of the graph but not the starting node of the graph.

Mathematically, a map oracle is a function Φ:$\Gamma \rightarrow S$, where Γ is the set of all graphs and S is the set of binary strings. The advice s is an input to the exploration algorithm.

Exploration in time T with advice of size of x is possible if there exists an exploration algorithms which traverses (edges) all the graphs with an input advice of size x. Proving that such an exploration is not possible lies in showing that no matter what exploration algorithm is used with advice of size x there exists a graph which takes time more than t.

Chapter 2

Related work

A tangential form of research is one which involves exploration of anonymous graphs. Herein the agent is still unable to mark the nodes visited but also has no apriori information which causes the agent to be unable to stop after the exploration.
Some authors[1,2] have dealt with this by using the mechanism of pebbles which the agent drops to mark a node and later picks up to drop on another node.
Providing agents with information to enable effective completion of tasks has been extensively studied in [3,4,5,6,7,8] in various situation such as graph coloring and leader election. All the aforementioned algorithms follow the paradigm of algorithms with advice.
In[9], the number of bits necessary for communication in order to verify the solution which has been represented in a distributed manner was calculated. In [16.18] algorithms with advice were used for online algorithms.
In [10] the topology of graphs was evaluated using advice given to the nodes.
The efficiency measure adopted in most of these papers in the context of graph exploration was the number of edge traversals. However, even more restrictions are imposed on the agent in [11,12] wherein the agent is assumed to have a limited tank and has to return to the refuelling station after some time.

Chapter 3

Previous work

There are some important results related to node exploration in anonymous graphs[14], which are as follows-

Proposition 3.0.1. *There exists an exploration algorithm, working in time $O(n^2)$ and using advice of size $O(n \log n)$, provided by a map oracle, for n-node graphs.*

Proposition 3.0.2. *There exists an exploration algorithm, working in time $O(n)$ and using advice of size $O(n \log n)$, provided by an instance oracle, for n-node graphs (an instance oracle knows the port-numbered map and the starting node of the agent).*

Proposition 3.0.3. *Regardless of the size of the advice, exploration time of $\Theta(n^2)$ cannot be beaten, for some n-node graphs for a map oracle.*

Proposition 3.0.4. *For any $\delta < 1/3$, any exploration algorithm using advice of size $o(n^\delta)$ must take time $\omega(n^2)$ on some n-node graph for arbitrarily large n, for a map oracle.*

Proposition 3.0.5. *Any exploration algorithm using advice of size $o(n \log n)$ must take time $\Omega(n^\epsilon)$, on some n-node graphs for some constant $\epsilon < 2$, for arbitrarily large n, for an instance oracle.*

Chapter 4

Results

One result is that with advice of size $O(n \log n)$, edge exploration can be done in $O(n^3)$ time for any n-node graph.

We also go on to prove that the lower bound for edge traversal is of order $\Omega(\mathrm{n}^{8/3})$. We describe a construction of a graph which forces the agent to take steps in the order of $\Omega(\mathrm{n}^{8/3})$ regardless of the size of the advice given. This is an improvement on the trivial lower bound $\Omega(\mathrm{n}^2)$ which is arrived at by counting all the edges in a graph.

Chapter 5

Technical Contribution

5.1 Algorithm which takes time $O(n^3)$

Theorem 5.1.1. *There exists an exploration algorithm, working in time $O(n^3)$ and using advice of size $O(n \log n)$, provided by a map oracle for n-node graphs.*

Proof. Let E(u) be a Euler tour of the spanning tree of the graph starting from some vertex u, and $E_1(u)$ be its reverse, there are n such tours one for each vertex. The concatenation of E(u) and $E_1(u)$ is considered and all the n aforementioned tours are concatenated and given as advice to the agent for traversal.
It is possible that a suggested port may be unavailable because the actual starting node of the agent may not match the initial node of the current Euler tour being performed, in which case the agent backtracks.
During traversal when a node is visited all the edges incident on it are traversed. Thus in each tour $O(n^2)$ edges are traversed, so the total edges traversed would be $n \times$O(n^2) which is $O(n^3)$. □

5.2 Lower Bound

Consider an $m/2$- regular m-node bi-partite graph H. Let the disjoint sets of vertices be O={1,3,5,....} and E={2,4,6,....}. Every vertex in E is adjacent to every vertex in O as it is

an $m/2$ regular graph.

We use the following construction from [13], let T be the set of edges in a spanning tree of H. Let S be the set of edges in H aside from T.

We know s=$|S|$=$m^2/4$-m+1, let S=$\{e_1, e_2, e_3, ..., e_s\}$

Take 2 copies of H. Let them be H^1 and H^{11}. Construct a one to one correspondence between x $\in \{0,1\}^s$-$\{0^s\}$ and the set of edges S, if x_i in the sequence x is 1, then the edge e_i in the copies of H^1 and H^{11} are crossed.

If e_i=$\{v_1, v_2\}$ then crossing it entails 2 new edges $\{v_1^1, v_2^{11}\}$ and $\{v_2^1, v_1^{11}\}$ respectively and the old edges $\{v_1^1, v_2^1\}$ and $\{v_2^{11}, v_1^{11}\}$ are removed. It is to be noted that no matter how many edges are crossed in this manner the degree of the nodes does not change as the number of edges incident on each vertex remains unchanged. Let the new graph constructed in this manner be $H_{x(e_i)}$ where $e_i \in$ S.

The new graph $H_{x(e_i)}$ would have the set of vertices $V^1 \cup V^{11}$ and set of edges E_x where E_x= $\{(v_i,v_j)\}$ where in the edge is either crossed or not depending on the sequence x. $V^1 \cup V^{11}$ is the union of the sets E^1, O^1, E^{11} and O^{11}.

From [13] it follows that for every edge e $\in$ S there exists a sequence x $\in \{0,1\}^s - \{0^s\}$ such that if a sequence W of port numbers starting from node v_1 does not visit e atleast (s+1) times then W would not visit either e^1 or e^{11} in one of the copies of H in $H_{x(e)}$.

Construct an oriented ring with s nodes y_1, y_2,....,y_s. Create a family of graphs $H_{x(e_i)}$ where e $\in$ {1,2,3...,s} and join y_1, y_2,...., y_s to $H_{x(e_1)}$, $H_{x(e_2)}$,.....,$H_{x(e_s)}$ respectively at node 1 of these graphs. The port numbers at the node 1 are assigned as m/2 and the port number at node y_i is given as 2.

Each of these $H_{x(e_i)}$ have vertices which can be divided into 4 sets E_1, O_1, E_{11} and O_{11}. Any $H_{x(e_i)}$ can be constructed from these 4 sets of vertices by just joining the required vertces. We endeavour to recreate each $H_{x(e_i)}$ using the vertices of every other $H_{x(e_i)}$.

In fact we do the following we do the following with any two $H_{x(e_i)}$'s. We first join every vertex in E_1 of one $H_{x(e_i)}$ with every vertex in O_{11} of the other $H_{x(e_i)}$, now join every vertex in O_1 which is the counter part of teh aforementioned E_1 with every vertex of E_{11} in the other $H_{x(e_i)}$. Now cross some edges so as to create $H_{x(e_i)}$. The port numbers of these new edges start are at least m/2. Let this graph be $\hat{G}$.

Lemma 5.2.1. *From any traversal sequence of $\hat{G}$, we can create a traversal sequence for any $H_{x(e_i)}$.*

Proof. Consider a traversal sequence of $\hat{G}$ namely U. It gives a sequence of port numbers which when followed traverse all edges of G at least once. Isolate all the port numbers which when taken by the agent traverses one of the edges of $H_{x(e_i)}$.
$U = 1.3.4.....(-).....(-).....(-).....(-)....(-).....$
These port numbers (represented by the '-' in the above equation) by themselves do not necessarily constitute a traversal sequence for $H_{x(e_i)}$ but we can surely add some port numbers between these ports as every edge in a $H_{x(e_i)}$ can be reached from every other edge. After padding up the original sequence we get a traversal sequence for $H_{x(e_i)}$. The padding between each edge is the diameter of the graph which is some constant c. This constant c is dominated by other terms larger than it in the asymptotic notation. Also if such a traversal for $H_{x(e_i)}$ does not start from vertex v_1 we can can create the sequence such that the agent goes to v_1 and continues traversal from there, this will just add a constant to the exploration time which will be dominated by other terms.
We do the same thing for all $H_{x(e_i)}$'s and their copies too; we create a traversal sequence for each of them. From [13] edge e_i has to be visited at least s+1 times in $H_{x(e_i)}$ otherwise edge exploration would fail, hence a particular $H_{x(e_i)}$ has a traversal time of order $\Omega(s+1)$.

□

Lemma 5.2.2. *Each $H_{x(e_i)}$ must have a traversal time of order $\Omega(m^4)$*

Proof. Assume that a sequence U is a traversal sequence for , it mus visit each e_i in each $H_{x(e_i)}$ atleast (s+1) times from [13]. Now as it is the result of a deterministic algorithm even upon changing the starting node so that part of the sequence D_i which before traversed $H_{x(e_i)}$ but must now visit $H_{x(e_j)}$ the sequnce U remains unchanged, so D_i must visit all $\Omega(m^2)$ edges atleast (s+1) times.
So each $H_{x(e_i)}$ takes $m^2 \times \Omega(m^2) = \Omega(m^4)$ time.

□

Theorem 5.2.1. *Graph G has an exploration time of the order $\Omega(n^{8/3})$*

Proof. From lemma 3.2.2 it follows that each $H_{x(e_i)}$ has an exploration time in $\Omega(m^4)$. There are $\Omega(m^2)$ copies of a $H_{x(e_i)}$ and there are $\Omega(m^2)$ $H_{x(e_i)}$'s. So the exploratioin time is of the order $\Omega(m^4) \times \Omega(m^2) \times \Omega(m^2)$ which is $\Omega(m^8)$ which in turn is $\Omega(n^{8/3})$.

□

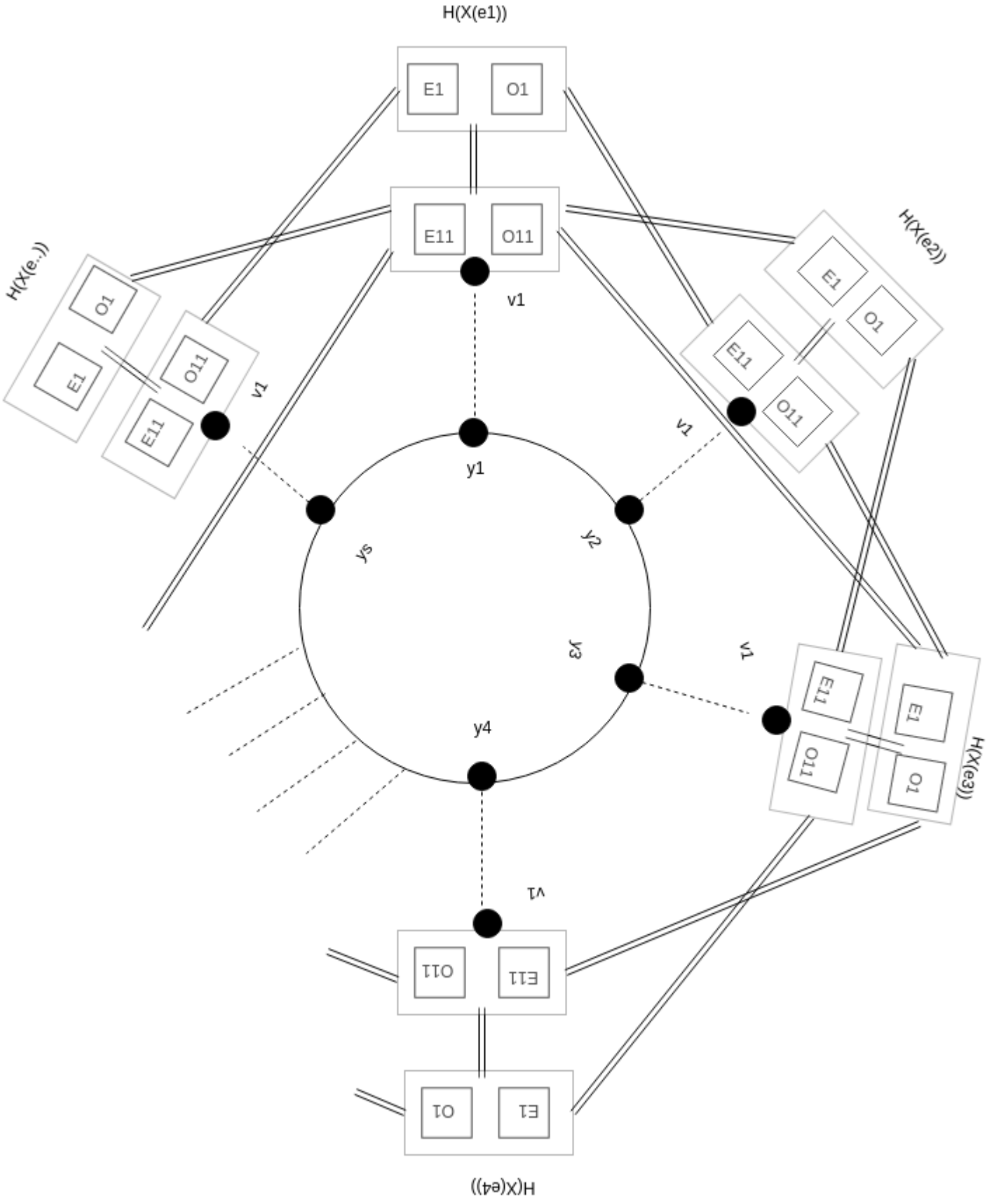

Figure 5.1: Graph $\hat{G}$

Chapter 6

Summary and Conclusions

Proposition 6.0.1. *There exists an edge exploration algorithm which takes advice of size* $O(n \log n)$ *and completes traversal in time* $O(n^3)$.

Proposition 6.0.2. *Regardless of the size of advice any exploration algorithm will take* $\Omega(n^{8/3})$ *time on some* n*-node graph.*

6.1 Scope for future work.

It is strongly felt that the lower bound can be further improved to $\Omega(n^3)$, a guiding principle to achieve this would be increase the number of edges in a graph without increasing the number of nodes, this same principle has been used in the construction of the graph in this report.

Bibliography

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

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

[3] D. Dereniowski, A. Pelc, Drawing maps with advice, Journal of Parallel and Distributed Computing 72 (2012), 132143.

[4] Y. Emek, P. Fraigniaud, A. Korman, A. Rosen, Online computation with advice, Theoretical Computer Science 412 (2011), 26422656.

[5] P. Fraigniaud, C. Gavoille, D. Ilcinkas, A. Pelc, Distributed computing with advice: Information sensitivity of graph coloring, Distributed Computing 21 (2009), 395403.

[6] P. Fraigniaud, D. Ilcinkas, A. Pelc, Communication algorithms with advice, Journal of Com- puter and System Sciences 76 (2010), 222232.

[7] P. Fraigniaud, D. Ilcinkas, A. Pelc, Tree exploration with advice, Information and Computation 206 (2008), 12761287.

[8] P. Fraigniaud, A. Korman, E. Lebhar, Local MST computation with short advice, Theory of Computing Systems 47 (2010), 920933.

[9] A. Korman, S. Kutten, D. Peleg, Proof labeling schemes, Distributed Computing 22 (2010), 215233.

[10] E. Fusco, A. Pelc, R. Petreschi, Use knowledge to learn faster: Topology recognition with advice, Proc. 27th International Symposium on Distributed Computing (DISC 2013), 31-45.

[11] B. Awerbuch, M. Betke, R. Rivest and M. Singh, Piecemeal graph learning by a mobile robot, Proc. 8th Conf. on Comput. Learning Theory (1995), 321-328.

[12] E. Bar-Eli, P. Berman, A. Fiat and R. Yan, On-line navigation in a room, Journal of Algorithms 17 (1994), 319-341.

[13] A. Borodin, W. Ruzzo, M. Tompa, Lower bounds on the length of universal traversal sequences, Journal of Computer and System Sciences 45 (1992), 180-203.

[14] Barun Gorain, Andrzej Pelc, Deterministic graph exploration with advice.