Comparison of Tarry's Algorithm and Awerbuch's Algorithm

Mike Yuan

Department of Computer Science, Kent State University, Kent OH 44242

myuan@cs.kent.edu

***Abstract*— Tarry's algorithm and Awerbuch's algorithm are two important algorithms for traversing connected networks. This paper compares the two algorithms by doing experiments on them. The experiments measure their time complexity and message complexity while varying the number of nodes in the network and the density of the network. The results shows that Awerbuch's algorithm is more effective than Tarry's algorithm in time complexity, and Tarry's algorithm is more effective than Awerbuch's algorithm in message complexity.**

***Index Terms*—Tarry's algorithm, Awerbuch's algorithm, time complexity, message complexity.**

I. Introduction

A distributed system is a collection of independent processes that communicate with each other in order to accomplish certain tasks. It has many applications, for example, wireless networks, distributed databases, etc.

It is very important to traverse connected networks in distributed systems. And the efficiency of the traversal algorithms is measured by time complexity and message complexity.

Tarry's algorithm and Awerbuch's algorithm are two well known traversal algorithms. It is interesting to compare the two algorithms in order to see in what scenario one algorithm performs better than the other. It is clear that the number of nodes in the network and the density of the network affect the performance of the algorithms.

This paper will experiment and compare the time complexity and message complexity of the two algorithms and focus on the influence of the number of nodes in the network and the density of the network.

Section II highlights some preliminary knowledge in this paper. Section III describes the setup of the experiments conducted to evaluate the relative performance of the algorithms, demonstrates the results obtained and analyze them. Section IV concludes the research conducted and highlights future work.

II. Preliminary knowledge

A wave algorithm is a distributed algorithm if for each computation (also called wave) C:

1. Termination: C is finite.
2. Decision: C contains at least one decide event.
3. Dependence: in each computation C each decide event is causally preceded by an event in each process.

Traversal algorithm is a wave algorithm with the following properties:

1. Each computation contains one initiator which starts computation by sending one message.
2. When a process receives a message it either sends out one message or decides.

And the metrics for measuring efficiency of algorithms are:

1. Time complexity is the number of messages in the longest chain of causally dependent events.
2. Message complexity is number of messages it takes the algorithm to carry out specified task.

A traversal algorithm for arbitrary connected networks was given by Tarry in [4]. Initiator forwards the token to one of its neighbors, each neighbor forwards the token to all other nodes and when done returns the token. A node is free to choose any node to forward the token to. The algorithm is formulated in the following two rules:

1. A node never forwards the token through the same channel twice.
2. A non-initiator only forwards the token to its father (the neighbor from which it first received the token) only if there is no other channel possible according to the above rule.

Tarry's algorithm is a traversal algorithm, and the time complexity and message complexity of Tarry's algorithm are 2E.

Awerbuch gave an improved algorithm of the classical depth first search algorithm in [1]. A node holding the token for the first time informs all neighbors except its father (and the node to which it will forward the token). It constructs spanning tree in time proportional to the number of nodes in linear time, and prevents token forwarding over frond edges because each process knows which neighbors were visited

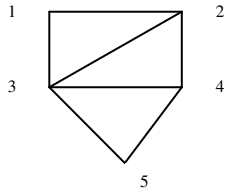
before it forwards the token. The node notifies its neighbors that it is visited by sending <vis> messages to them. The token is only forwarded when these neighbors all acknowledged reception. The token is only forwarded to nodes that were not yet visited by the token (except when a node sends the token to its father). And its time complexity is 4N-2, message complexity is 4E.

III. EXPERIMENTS

We describe some experiments on the time complexity and message complexity of Tarry's algorithm and Awerbuch's algorithm while varying the number of nodes and density of the network.

A. *Experimental Setup*

Because we need to vary the number of nodes and density of the network, we use random generated graphs to do the experiments. Random graph is represented by adjacency matrix. For example, Fig. 1 shows a random graph and its adjacency matrix.



	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	1	1
4	0	1	1	0	1
5	0	0	1	1	0

Fig. 1. An example random graph and its adjacency matrix.

The probability of there is an edge between two nodes is called connection probability. The experiments in this paper will use connection probability 30% as sparse graphs, 50% as moderate graphs, and 70% as dense graphs.

Because two nodes are trivial, the random graphs used in the experiments have at least three nodes. The experiments do all data in range of 3 to 10 because the Tarry's algorithm and Awerbuch's algorithm differentiate there. And the maximum number of nodes of the examples in this paper is 50 because the difference is more and more pronounced and stable in range 30-50.

B. *Experiment Results*

First of all, this paper shows experiment results of time complexity. The first experiment is the time complexity of connection probability 30% shown in Fig. 2. It shows that Tarry's algorithm is better than Awerbuch's algorithm in range 3-9 nodes, Tarry's algorithm's complexity is almost same as Awerbuch's algorithm's complexity in range 10-11 nodes, Awerbuch's algorithm is better than Tarry's algorithm in range equal to or more than 12 nodes, and the difference is more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm is more stable than Tarry's algorithm.

The second experiment is the time complexity of connection probability 50% shown in Fig. 3. It shows that Tarry's algorithm is better than Awerbuch's algorithm in range 3-7 nodes, Tarry's algorithm's complexity is almost same as Awerbuch's algorithm's complexity in range 7-8 nodes, Awerbuch's algorithm is better than Tarry's algorithm in range equal to or more than 9 nodes, and the difference is more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm is more stable than Tarry's algorithm.

The third experiment is the time complexity of connection probability 70% shown in Fig. 4. It shows that Tarry's algorithm is better than Awerbuch's algorithm in range 3-4 nodes, Tarry's algorithm's complexity is almost same as Awerbuch's algorithm's complexity in range 4-5 nodes, Awerbuch's algorithm is better than Tarry's algorithm in range equal to or more than 6 nodes, and the difference is more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm is more stable than Tarry's algorithm.

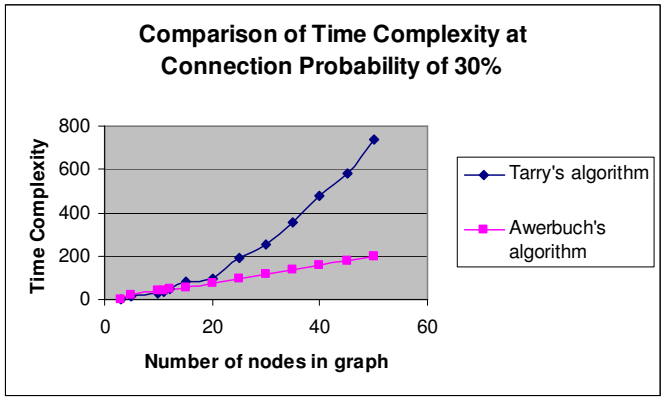


Fig. 2. Time complexity at connection probability of 30%.

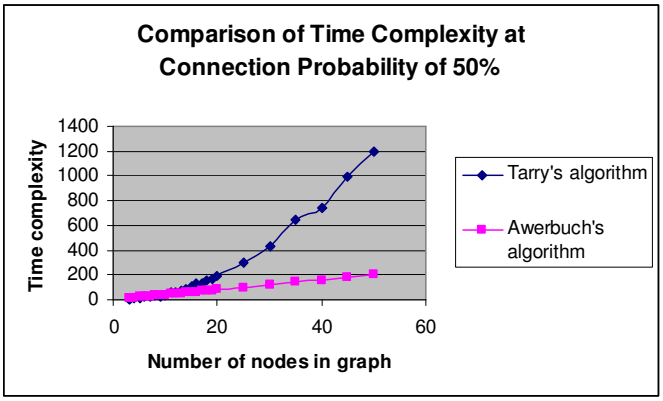


Fig. 3. Time complexity at connection probability of 50%.

Next, this paper shows experiment results of mesage complexity. The first experiment is the message complexity of connection probability 30% shown in Fig. 5. It shows that Tarry's algorithm is always better than Awerbuch's algorithm,

the difference is not distinguishable in range 3 to 15 nodes, is distinguishable in range more than 15 nodes, and becomes more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm increases almost twice speed of Tarry's algorithm.

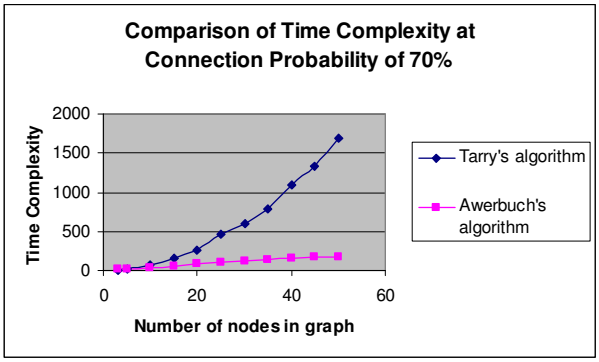


Fig. 4. Time complexity at connection probability of 70%.

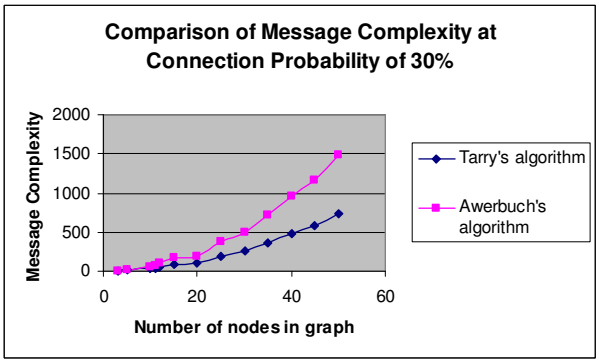


Fig. 5. Message complexity at connection probability of 30%.

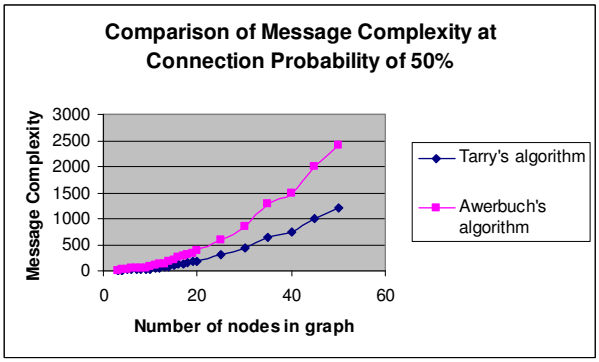


Fig. 6. Message complexity at connection probability of 50%.

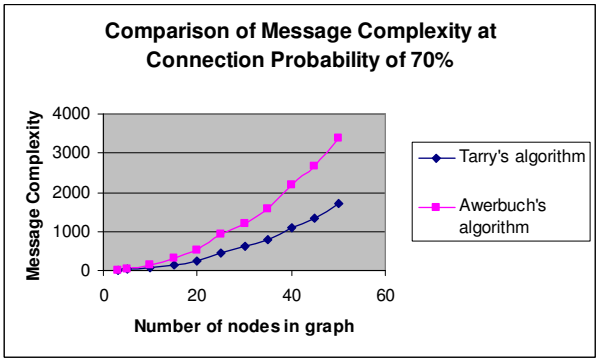


Fig. 7. Message complexity at connection probability of 70%.

The second experiment is the message complexity of connection probability 50% shown in Fig. 6. It shows that Tarry's algorithm is always better than Awerbuch's algorithm, the difference is not distinguishable in range 3 to 10 nodes, is distinguishable in range more than 10 nodes, and becomes more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm increases almost twice speed of Tarry's algorithm.

The third experiment is the message complexity of connection probability 70% shown in Fig. 7. It shows that Tarry's algorithm is always better than Awerbuch's algorithm, the difference is not distinguishable in range 3 to 7 nodes, is distinguishable in range more than 8 nodes, and becomes more and more pronounced as scale of graphs increases. Another observation is that Awerbuch's algorithm increases almost twice speed of Tarry's algorithm.

C. *Analysis of Experiment results*

Awerbuch's algorithm's time complexity is better than Tarry's algorithm when the number of nodes in the network is sufficiently large because the time complexity of Tarry's algorithm is 2E since all processes have been visited and each channel has been used once in both directions, and time complexity of Awerbuch's algorithm is 4N-2 since token traverses N-1 edges twice and is delayed at every root node for two time units. And when the graphs become denser, Tarry's algorithm's time complexity is $O(N^2)$, but Awerbuch's algorithm's time complexity is still O(N).

And Tarry's algorithm's message complexity is better than Awerbuch's algorithm because the message complexity of Tarry's algorithm is 2E since it is same as time complexity, and the message complexity of Awerbuch's algorithm is 4E since <vis> and <ack> messages are sent along each frond edge twice, <vis> is sent from father to son, <ack> is sent from son to father, and <tok> is sent twice along each tree edge. And when the graphs become denser, both Tarry's algorithm's and Awerbuch's algorithms' message complexities are $O(N^2)$.

IV. CONCLUSIONS AND FUTURE WORK

This paper compares the performance of Tarry's algorithm and Awerbuch's algorithm by varying the number of nodes and density of the network. The conclusions are, first Awerbuch's algorithm is more effective than Tarry's algorithm in time complexity. Second, Tarry's algorithm is more effective than Awerbuch's algorithm in message complexity. Third, When the graphs become denser, both time and message complexity of Tarry's algorithm, and message complexity of Awerbuch's algorithm are quadratic to the number of nodes in the network, but the time complexity of Awerbuch's algorithm is still linear to the number of nodes in the network.

And the future plans for this research are, first it is interesting to explore the influence of density of graphs on the algorithms, that is, change the density of the network while the number of nodes in the network is fixed. Second, since we measure time complexity and message complexity by varying the number of nodes in the network, we need to consider

whether we can measure the complexities while varying the number of edges in the network, not nodes. Third, we will do our experiments on real distributed systems, for example, multi hop ad hoc wireless network, wireless sensor network, and Tiny OS, etc.

ACKNOWLEDGMENT

I acknowledge the help given to me by Dr. Nesterenko and Tom Clouser during working on my project.

REFERENCES

[1] A. Baruch, "A New Distributed Depth-First-Search Algorithm," *Information Processing Letters* 20, pp. 147-150, 1985.

[2] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: Of theory and practice," *22nd ACM Symposium on the Principles of Distributed Computing (PODC),* Jul. 2003.

[3] M. Miyashita, M. Nesterenko, "2FACE: Bi-Directional Face Traversal for Efficient Geometric Routing," technical report TR-KSU-CS-2006-06, Kent State University.

[4] G. Tarry, "Le Problem Des Labyrinthes," *Nouvelles Annales de Mathematique 14*, 1895.

[5] G. Tel, Introduction to Distributed Algorithms , 2000.

[6] A. Vora and M. Nesterenko., "Void traversal for guaranteed delivery in geometric routing," *the 2nd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS),* pp. 63–67, Nov. 2005.

[7] D. Watson, M. Nesterenko, "MULE: Hybrid Simulator for Testing and Debugging Wireless Sensor Networks," *Second International Workshop on Sensor and Actor Network Protocols and Applications*, pp. 67-71, Aug. 2004.

[8] Slides of advanced operating systems class, http://deneb.cs.kent.edu/~mikhail/classes/aos.f07/.

[9] TOSSIM User Manual, http://deneb.cs.kent.edu/~mikhail/classes/aos.f06/aos_tos_tutorial/tos_tutorial.html,2006.

[10] TinyOs mailing Archive, http://deneb.cs.kent.edu/~mikhail/classes/aos.f06/aos_tos_tutorial/tos_tutorial.html.