Dedicated to my Family, Friends and Guruji...

Certificate

This is to certify that this thesis entitled "Some Observations on Leader Election Algorithms for Distributed Systems" submitted by Amit Biswas (Roll No.: 17071510) for the award of the degree of doctor of philosophy to the Indian Institute of Technology (Banaras Hindu University), Varanasi, is a record of bona fide research works carried out by him under my direct supervision and guidance and it has not been submitted elsewhere for a degree. It is further certified that the student has fulfilled all the requirements of Comprehensive Examination, Candidacy and SOTA for the award of Ph.D. Degree.

Signature of Supervisor
Anil Kumar Tripathi
Professor
Department of Computer Science and Engineering
Indian Institute of Technology (BHU)
Varanasi - 221005, India

Declaration

I hereby declare that the work embodied in this thesis is my own bona fide work and carried out by me under the supervision of **Prof. Anil Kumar Tripathi** from **December 2017** to **November 2021**, at the Computer Science and Engineering department, Indian Institute of Technology (BHU), Varanasi. The matter embodied in this thesis has not been submitted for the award of any other degree/diploma. I declare that I have faithfully acknowledged and given credits to the research workers wherever their works have been cited in my work in this thesis. I further declare that I have not willfully copied any other's work, paragraphs, text, data, results, etc., reported in journals, books, magazines, reports dissertations, theses, etc., or available at websites and have not included them in this thesis and have not cited as my own work.

Date: 11/11/2021
Place: Varanasi

Signature of Student
(Amit Biswas)

Certificate by the Supervisor

It is certified that the above statement made by the student is correct to the best of my/our knowledge.

Signature of Supervisor
(Anil Kumar Tripathi)

Signature of Head of Department
(Prof. Sanjay Kumar Singh)

Copyright Transfer Certificate

Title of the Thesis: **Some Observations on Leader Election Algorithms for Distributed Systems**
Name of Student: **Amit Biswas**

Copyright Transfer

The undersigned hereby assigns to the Indian Institute of Technology (Banaras Hindu University), Varanasi, all rights under copyright that may exist in and for the above thesis submitted for the award of the Doctor of Philosophy.

Date: 11/11/2021 **Signature of Student**
Place: Varanasi **(Amit Biswas)**

Note: However, the author may reproduce or authorize others to reproduce material extracted verbatim from the thesis or derivative of the thesis for author's personal use provided that the source and the Institute's copyright notice are indicated.

Acknowledgments

"No one who achieves success does so without acknowledging the help of others. The wise and confident acknowledge this help with gratitude." -Alfred North Whitehead

Though only my name appears on the cover of this thesis, a great many people have contributed to produce it. I sincerely thank all the people who helped me to make this thesis possible.

Foremost, I would like to express my sincere gratitude to my supervisor, Prof. Anil Kumar Tripathi, for giving me the opportunity and continuous support to undertake my Ph.D. I am deeply grateful for his invaluable guidance, advice, mental and academic support, and motivation throughout my candidature. His guidance helped me in all the time of research, conducting experiments, and writing of this thesis.
I would like to express my gratitude to the Research Program Evaluation Committee members Prof. Sanjay Kumar Singh, and Dr. Ashok Ji Gupta, for their insightful comments, encouragement, and suggestions, which help me improve my research from various perspectives. Special thanks to Prof. Samir Aknine and Dr. Gaurav Baranwal for their support during my Ph.D. Their guidance and directions help me throughout my journey of Ph.D. I would like to convey my sincere gratitude to the other faculty members of the Department of Computer Science and Engineering, Prof. K.K Shukla, Prof. Rajeev Shrivastava, Dr. R.S Singh, Dr. Bhaskar Biswas, Dr. Sukomal Pal, Dr. Pratik Chattopadhyay, Dr. H.P Gupta, Dr. Ruchir Gupta, Dr. Amrita Chaturvedi, Dr. R.N. Chaudhary, Dr. A.K Singh, Dr. Tanima Dutta, Dr. Prasenjit Chanak and Dr. Vinayak Shrivastava for their guidance and support throughout this tenure.

I want to thank my friends and colleagues Mr. Sushant Kumar Pandey, Ms. Dipty Tripathi, Ms. Manisha Singh, Dr. Ashish Kumar Maurya, Mr. Tribikram Pradhan, Mr. Chitranjan Singh, Mr. Ankit Jaiswal, Mr. Nirbhay Tagore, Ms. Shruti Bajpai, Mr. Saurabh Arora, Mr. Shashank Kumar Singh, Ms. Pratishta Verma, Mr. Chintoo Kumar, Mr. Shivang Agarwal, Mr. Amit Kumar, Mr. Ashish Shrma, Mr. Kartick Sutradhar, Mr. Abhilash Gondane and Mr. Raviraj Vashishtha for their motivation and valuable comments. I also extend my thanks to other colleagues members of our department Ms. Naina Yadav, Mr. Ramakant Kumar, Mr. Rakesh Kumar Yadav, Mr. Supriya Chanda and Mr. Rupjyoti Baruah along with other departmental colleagues of computer science

and engineering department and different departmental colleagues. This thesis would not have been possible without their invaluable remarks and persistent help. I extend special thanks to the non-teaching and technical staff in the department, particularly, Mr. Ravi Kumar Bharti, Mr. Ritesh Singh, Mr. Shubham Pandey, Mr. Prakhar Kumar, Mr. Akhilesh Kumar Pal, Mr. Viplav Biswas.
Finally, I would like to thank my family for their love, encouragement, and moral and emotional support to achieve this great goal. It would not have been possible without their help and support. To them, I am eternally grateful.

Amit Biswas

Preface

With the ever-growing technological expansion in the world, distributed computing is becoming popular and widely used in several fields. From scientific research to everyday life applications, everywhere we are using distributed computing and enjoying the benefits of this computing paradigm. Distributed computing helps improve the performance of large-scale projects by combining the power of multiple computing devices. Distributed systems are the backbone of distributed computing. A distributed system is a collection of multiple independent computing components or nodes that work together to attain a common goal and appear as a single coherent system to the user. These independent nodes are connected through a network, and they perform the tasks of the system with collaboration. In a distributed system, to complete an enormous task fast, the task is partitioned into a set of possible sub-tasks and allocated to multiple nodes. Then the nodes execute their assigned sub-tasks in a concurrent and consistent manner to complete the task in a faster way. The banking system, manufacturing industry, artificial intelligence, e-commerce, transport, health care, agriculture, defense systems have been using distributed computing for a long time. Various modern technologies like smart city, blockchain, cloud computing, edge computing, and so on are constructed based on the concept of decentralized systems. Recently, distributed machine learning and deep learning technologies are inclined to adopt the concept of distributed systems to fulfill their need for a reliable, high-performance, and large-scale computational platform. Though a distributed system is supposed to be scalable, cost-effective, fault-tolerant, highly available and reliable, and provides a high-performance computing environment, several issues and challenges are also associated with such a system. System management, synchronization, consistency maintenance, and efficient resource utilization are some significant issues. In a centralized system, a central node controls and manages the whole system, synchronizes the events of the system, utilizes the resources efficiently, and maintains consistency among the several entities of the system. However, no such central node exists in a truly distributed system. Hence, a node needs to be elected as the system leader to manage, coordinate, synchronize, and efficiently utilize a distributed system. The leader helps distribute the computing load, aggregate the computing results, and present a single system image for processes and users. It also simplifies the system management, coordination, and operational complexity and reduces the message and time overhead of the system. Thus, the leader election concept helps reduce the complexity of a decentralized system by effectuating control through a leader node elected by an appropriate leader election algorithm and improves overall system performance. A suitable system leader can help to improve a distributed system's resource utility, reliability, and fault tolerability that directly improve the performance of the applications of the various fields that adopt this system. So, the leader election plays an important role in the distributed systems.

In this thesis, through a detailed literature survey on existing Leader Election Algorithms (LEAs), we first find out some major issues and challenges with them. Then we propose possible solutions to those issues. While doing that, to ease out the survey process, we first divide the existing LEAs into two categories based on network topology. One, algorithms for regular network topology, and two, algorithms for arbitrary network topology. Then we study the LEAs designed considering various distributed system models for these two categories and find the research gaps. Finally, we propose, analyze and characterize four different self-stabilizing leader election methods to overcome those research gaps. The first two algorithms are designed for two regular network topologies (ring and 2D torus), and the subsequent two algorithms are designed for the arbitrary network topology.

We design our first algorithm for a ring network topology considering a failure-recovery and partially synchronous distributed system model. The main aim of designing this algorithm is to reduce the message complexity and time complexity of the election process and elect a low-loaded and higher reliable node as the system leader. The second algorithm is designed for a 2D torus network. Here, we propose a lower bound message complexity of a comparison-based leader election algorithm in a 2D torus network. The objectives of this work are to design a more fault-tolerant leader election algorithm and reduce the message and time overhead of the election process. We introduce several message-sending patterns that make the algorithm more link and node failures tolerant, reduce the number of exchanged messages and time steps of the election process, and elect a good quality leader for the system.

Our last two algorithms are designed for arbitrary network topologies. The first one of these two algorithms is designed considering a distributed real-time system. Here, considering the characteristics of a distributed real-time system, the proposed algorithm not only elects a leader but also identifies some higher potential nodes for leadership such that if a leader crashes, the system can select another potential node as the system leader. Using the eccentricity of the nodes, we divide the network into two layers, i.e., the inner layer and the outer layer, and only the inner layer nodes take part in the election directly, which helps reduce the message and time overhead of the election process.

A good quality leader can manage a distributed system in a better way and utilize the resources efficiently that improve the overall system performance. Different distributed systems are designed for different purposes. So the definition of a good quality leader may vary from system to system. On the other hand, a distributed system may consist of heterogeneous nodes. A node can have multiple attributes, and different nodes can have different values of those attributes. So it is challenging to determine which type of quality a leader of a distributed system should have and which attributes are responsible for that quality of the leader. It is also difficult to determine how much priority has to be given to those pertinent attributes. Hence electing a good quality leader for a system considering multiple attributes is a pretty intricate task and that is why the fourth algorithm is designed to elect a good quality leader according to the system requirements by introducing the concept of the quality factor of the nodes. We modify the concept of the TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) MCDM method to calculate the quality factor of the nodes. Here, we involve experts to identify the appropriate attributes of a good quality leader (according to the system requirements) and assign weight to identified attributes according to their importance to elect the suitable leader for the system. This algorithm can tolerate multiple links and node failures during the election and reduce the time and message overhead of the election process.

List of Publications

1. **Amit Biswas**, Ashish Kumar Maurya, Anil Kumar Tripathi and Samir Aknine, “FRLLE: a failure rate and load-based leader election algorithm for a bidirectional ring in distributed systems”, The Journal of Supercomputing, 2021, 77, pp.751-779. DOI: https://doi.org/10.1007/s11227-020-03286-y, (SCI indexed, Publisher: Springer, Impact Factor: 2.474)

2. **Amit Biswas**, Anil Kumar Tripathi and Samir Aknine, “Lea-TN: leader election algorithm considering node and link failures in a torus network”, The Journal of Supercomputing, 2021. DOI: https://doi.org/10.1007/s11227-021-03803-7, (SCI indexed, Publisher: Springer, Impact Factor: 2.474)

3. **Amit Biswas**and Anil Kumar Tripathi, “Preselection Based Leader Election in Distributed Systems”, The 14th International Symposium on Intelligent and Distributed Computing (IDC 2021), Italy, September 16-18, 2021 (SCOPUS indexed, Publisher: Springer) (Accepted)

4. **Amit Biswas**, Manisha Singh, Gaurav Baranwal, and Anil Kumar Tripathi “Multi-Attribute based Self-Stabilizing Leader Election in Distributed Systems: An Application to Cloud Computing”, IEEE Transactions on Cloud Computing, (SCI indexed, Publisher: IEEE, Impact Factor: 5.938) (Communicated)

Contents

List of Figures

List of Tables

Abbreviations

DC	**D**istributed **C**omputing
DS	**D**istributed **S**ystem
RQ	**R**esearch **Q**uestion
LEA	**L**eader **E**lection **A**lgorithm
FRLLE	**F**ailure **R**ate and **L**oad-based **L**eader **E**lection
DRTS	**D**istributed **R**eal-**t**ime **S**ystem
TOPSIS	**T**echnique for **O**rder of **P**reference by **S**imilarity to **I**deal **S**olution
MCDM	**M**ulti-**C**riteria **D**ecision-**M**aking
MADM	**M**ulti-**A**ttribute **D**ecision-**M**aking
DML	**D**istributed **M**achine **L**earning
DDL	**D**istributed **D**eep **L**earning
FD	**F**ailure **D**etector
IoT	**I**nternet **o**f **T**hings

Symbols

Π	Set of all nodes in the system
N	Number of nodes in the system
L	Set of all links that connect the nodes of the system
N_Id_i	Node Id of the i^{th} node
L_Id_i	Leader Id stored by the i^{th} node
Fr_i	Failure rate of the i^{th} node
A_Cu_i	Average CPU utilization of the i^{th} node
A_Mu_i	Average memory utilization of the i^{th} node
A_Bu_i	Average bandwidth utilization of the i^{th} node
Lc_i	Leader coefficient of the i^{th} node
Lcc	Leader coefficient of the election message creator node
M_Lc_i	Minimum leader coefficient stored by the i^{th} node
$N\_Id\_Mlc_i$	Id of the minimum leader coefficient node stored by the i^{th} node
Lrt_i	The time-stamp when the i^{th} node got the last response from the leader
$Emat_i$	Time-stamp of the received election message when it arrives at node i
P_{ij}	A message propagation delay between the i^{th} and j^{th} nodes
P_i	The election message processing delay of the i^{th} node
Ttd_i	Total time delay of an election message calculated by the i^{th} node
Emc_Id	Election message creator's Id
Fl_Id	Failed leader Id

$Frem_i$	First received election message by the i^{th} node
Ein_i	It denotes whether the i^{th} node is an election initiating node
El_Id	Elected leader Id
Nfl_i	Number of failed links of the i^{th} node
Nnf_i	Number of non-faulty links of the i^{th} node
Lfi	Link failure indicator
$Lfact_i$	Leader factor of the i^{th} node
$Lfact_c$	Leader factor of the election message creator node
M_Lfact_i	Maximum leader factor stored by the i^{th} node
$T_{n,n}$	Represents a $(n \times n)$ $2D$ torus network that has N nodes and $N = n \times n$
$N\_Id\_Mlf_i$	Id of a node with the maximum leader factor that is maintained by the i^{th} node
Lsb	Link status bits
T_d	The diameter of the $T_{n,n}$ network
$Flist_i$	List of adjacent node and link failures information maintained by a node i
Lft	Number of link failures tolerability
Msp	Message sending pattern
Amc_Id	Acknowledgment message creator Id
$Remc\_Id$	Received election message creator Id
$Fimc\_Id$	Failure information message creator Id
Emc_Lf_Id	The Id of the election message creator that helped to identify the link failure
W_{in}	Width of the inner-layer
R	Radius of the arbitrary network
D	Diameter of the arbitrary network
Ect_i	Eccentricity of the i^{th} node
Qc_i	Quality-coefficient of the i^{th} node
Qc_c	Quality-coefficient of the election message creator node

Deg_i	Degree of the i^{th} node
Pl_i	When a provisional leader is selected as the system leader, a node i sets its Pl_i to 1. Otherwise node i sets its Pl_i to 0
T_{in}	A message takes T_{in} time to travel between two farthest inner layer nodes
T_f	A message takes T_f time to travel between two farthest nodes in the system
INL	Set of inner-layer nodes of the system
OLN	Set of outer-layer nodes of the system
P_List_i	List of potential nodes stored by the i^{th} node
Np_List	List of new potential nodes
Tl	The type of leader. $Tl = 1$ indicates the provisional leader and $Tl = 0$ indicates the primary leader
S_eim	It refers to the Id of an EIM message sender
C_ack	It refers to the Id of an ACK message creator.
C_agm	It refers to the Id of an AGM message creator.
$Status_i$	It refers to the status of a node i
Ein_Id_i	A node i stores the Id of an election initiating node in it
Qf_i	It is the quality factor of a node i
S_mqfm	It refers to the Id of an $MQFM$ message sender
Mqf	It refers to the maximum quality factor
Mqf_Id	It refers to the Id of the node with maximum quality factor
$Pnode_i$	It refers to the parent Id of a node i
$Cpnode_i$	It refers to the co-parent Id(s) of a node i
$Cnode_i$	It refers to the child Id(s) of a node i
$Tf_{e\text{-}m}$	The time to exchange an EIM message and its corresponding $MQFM$ message between the two farthest nodes in the system