A K-Neighbor-based, Energy Aware Leader Election Algorithm (KELEA) for Mobile Ad hoc Networks

Alaa N. Alslaity
Department of Computer Science
School of Computer and Information Technology
Jordan university of Science and Technology
Irbid 22110, Jordan

Sanaa A. Alwidian
Department of Computer Science and Applications
Faculty of Prince Al-Hussein Bin Abdallah II for IT
The Hashemite University
Zarqa 13115, Jordan

ABSTRACT

Leader election is the process of electing a node as a coordinator or a leader to the whole distributed system. This paper proposes a new leader election algorithm called: "***K**-neighbor-based, **E**nergy Aware **L**eader **E**lection **A**lgorithm (KELEA)*" that works efficiently in the ad hoc distributed systems. KELEA is an energy-efficient algorithm which aims to save energy by reducing the number of exchanged messages. The main idea is to assign every node in the ad hoc network with a unique ID, where the ID represents a performance value such as density and energy. Then maintain a descending-ordered list of nodes according to their IDs. When a node detects a leader crash, it instantiates a leader election process by sending an ELECTION message to only a specific number of neighbouring nodes (K) to participate further in the election process, where K represents a ratio of the whole number of nodes. The paper shows through mathematical analysis and a practical example that the proposed algorithm KELEA outperforms other algorithms that perform election using traditional flooding (i.e. by sending messages to the entire neighbouring nodes). KELEA reduces the message overhead and minimizes energy consumption in comparison with other flooding-based algorithms.

General Terms

Mobile Ad hoc Networks, Leader Election Algorithms, Distributed Systems.

Keywords

Distributed Systems, MANETs, Leader, Leader Election, ELECTION message, energy efficient.

1. INTRODUCTION

In the domain of distributed systems, leader election operation is known as electing a process (or a node) as a leader that manages the tasks of other processes that are distributed along the system. The leader is required to act as both a coordinator and an initiator to manage particular jobs such as directory server, token regenerator and central lock coordinator [1].

In distributed systems, if the current leader becomes crashed, all other nodes have to elect another leader. The process of assigning leadership to a unique node is known as leader election problem [2].

In the literature, many leader election algorithms have been proposed to manage the problem of leader election in both classical (wired) and wireless distributed systems. For the classical distributed systems, a plenty of algorithms have been proposed [3], [4], [5], one of the popular algorithms is the Garcia- Molina's bully algorithm [6]. Bully algorithm is a simple algorithm in terms of the concept and implementation, however, it imposes a high message overhead which is of order $O(n^2)$ where n is the total number of nodes in the system. In addition, it is not well-applicable in the mobile ad hoc distributed systems.

The emergence of wireless ad hoc distributed systems required a new set of leader election algorithms that adapt well with the dynamic topology of such mobile ad hoc systems. Leader election process is not trivial in ad hoc environments due to their unique characteristics such as the limited battery power (energy), limited bandwidth and nodes mobility [7]. Several election algorithms were proposed for mobile ad hoc networks, nevertheless, most of them assume that the node with the largest *ID* number should be chosen as a leader regardless to the other previously mentioned factors (as nodes mobility, power or density status.)

This paper proposes a new leader election algorithm that is aware about both the network density and energy. By the term *density*, we mean the number of neighbours that surrounds a particular node. We refer to our proposed algorithm as the: **K**-neighbour-based **E**nergy Aware **L**eader **E**lection **A**lgorithm for Mobile Ad Hoc Network (KELEA).

The main idea of KELEA is to assign every node in the ad hoc network with a unique *ID*, where the *ID* represents a performance value such as density, velocity, battery power, or signal strength (or a combination of two or more value). In our paper, we rely on nodes' *density* as the major performance value. Then we maintain a descending-ordered list of nodes according to their IDs, such that the first node in the list represents the node with the largest number of neighbors, the successor node is the node with the next higher neighbor, and so on. When a node detects a leader crash, it instantiates a leader election process by sending an *ELECTION* message to only a specific number of 1-hop neighbouring nodes (K) to participate further in the election process, where K represents a ratio of the whole number of nodes. Through extensive empirical, we find out that $K = \left\lceil \frac{N}{3} \right\rceil$ achieves the best results (in comparison with other ratios) for different densities.

The rest of this paper is organized as the following: Section 2 discusses the related work. Section 3 discusses the main objectives of the proposed protocol. The assumptions are discussed in Sections 4. Section 5 illustrates the proposed algorithm, and a mathematical analysis of the proposed algorithm is provided in Section 6. Finally Section 7 concludes the paper and provides future directions.

2. RELATED WORK

One of the most important operations in the distributed systems is the leader election which concentrates on selecting a leader node among the existing nodes to be the coordinator of the network. In the literature, many leader election algorithms had been proposed.

Gracia- Molina's bully algorithm [6] is considered as one of the most popular leader election algorithms for traditional distributed systems. In Bully, when a node *N* detects that the leader is crashed, it sends an *ELECTION* message for all other node with *IDs* larger than its *ID*. If one of those nodes is alive, it takes over the election process and sends *OK* message to *N*. once *N* receives an *OK* message it takes over. If no node responds to the *ELECTION* message, *N* wins the election and declares itself as a leader. This process is repeated until all nodes give up but only one, which is eventually the newly elected leader. In the worst case, that is, when the lowest-*ID* process detects the crash, the algorithm requires $O(n^2)$ messages. Many modifications have been proposed to enhance the performance of the bully algorithm.

In [8], Mamun et al have proposed a new algorithm that is more efficient than bully algorithm in terms of messages and time complexity. By minimizing the number of redundant election messages, the proposed algorithm requires, in the best case, n-1 message, while in the worst case, it requires 2(n-1) messages.

Another modification of the bully algorithm has been proposed in [9], in which an overhead aware leader election algorithm had been proposed. This algorithm aims to perform leader election process using the minimum number of *ELECTION* messages as much as possible. To do so, the algorithm depends on sorting the nodes in descending order based on their *IDs*. Once node N detects a leader failure, it sends an *ELECTION* message to the node with the largest *ID* only, and the node with the next higher *ID* (after the crashed leader) is elected as the new leader. Applying this algorithm for leader election sufficiently reduces the number of election messages (*O(N)* in the worst case).

M. Gholipour et al [10] have followed a new strategy that depends on selecting a leader and alternatives. The main goal of this protocol is to reduce the number of exchanged messages using candidate nodes as alternatives to the leader. When a process *P* notice that the leader is crashed it sends Crash-leader message to the Alternative1; if alternative1 is alive, it ensures that the leader is crashed, if so it is selected as the next leader. If alternative1 is not alive, *P* sends Crash-leader to the next alternative. This process is continued until *P* detects that all the alternatives are not alive; at this point *P* starts an election process like bully algorithm to select the new leader and its alternatives. However, maintaining a list of alternatives needs more messages exchanging which in turn makes extra load on the system.

In [11] the researchers proposed a stable election protocol based on unreliable failure detector. The proposed algorithm redesigns the bully algorithm for asynchronous distributed systems. It is composed of three phases; proposing a new leader, then all processes that agree on the new leader acknowledges the reservation of the potential leader, finally the new leader is elected. To ensure safety, the proposed algorithm requires the processes to agree with the current leader crash before the election of a new leader. Although this algorithm ensures safety, it needs extra messages exchanging which in turn makes additional load on the system.

Some other protocols assume that the processes are ordered and they view them as a ring; each node knows its successor. The first leader selection algorithm for unidirectional rings was proposed by Le Lann [12]. The main idea of such algorithm is that each node prepares an election message that contains its own *ID* and circulates this message around the ring clockwise. The process with the highest priority should be elected as a coordinator.

All of the above protocols and much more others have been proposed to solve the problem of electing a leader for the distributed systems, particularly, the traditional distributed systems in which the network topology is static and do not changed along the time. Most of those algorithms are extrima finding algorithms in which nodes are assumed to have a unique *ID* numbers, the leader which is elected is the node which has the largest *ID* number regardless to performance criteria. Such protocols are not applicable in the case of Mobile Ad-hoc Networks (MANETs) in which the topology changed frequently. The nodes in such networks have limited resources such as power and signal strengths. Due to those challenges many protocols have been proposed to solve the leader election problem in MANETs.

An energy efficient leader election algorithm for MANET had been proposed in [13]. It is based on reducing the number of exchanged messages in order to save energy, and it is processed as follow; each process holds an identifier called *LeaderP*, each process *P* starts by checking if *LeaderP*= p; if so, *P* broadcasts a message (ALIVE,P). When a process *P* receiving a message (ALIVE, q), *P* compares its leader with *q*, if q < LeaderP, *P* updates the identifier of its leader and sends (ALIVE,q) to all of its neighbor N such as q < N.

In [14] the researchers proposed an extrema finding algorithm that is adaptive to the topological changes of the network, and hence it is suited for MANET. The proposed algorithm can be used in scenarios where just a unique leader is desired [14]. In [15] another technique was proposed. The proposed algorithm depends on using mobile agents in both of the design and implementation levels to provide a solution for leader election problem for several network topologies.

Two algorithms for mobile ad hoc networks were proposed in [16]. The proposed algorithms where built based on the routing algorithm TORA [17]. The algorithms viewed the ad hoc network as Directed Acyclic Graph (DAC) and impose that each connected component of the graph should have a single leader. The first algorithm designed to adapt to a single topology changes (i.e. topology change occurs only after the algorithm has terminated its execution). While the second algorithm is designed for concurrent topology changes. Both algorithms are deployable in ad hoc environments, however, they have been provided with no proof of correctness.

Neepa Biswas [18] proposed a leader election algorithm for MANET that aims to reduce the message and time complexity of the election process. It depends on the position, battery life, and load of the nodes to elect a middle node (i.e. which has a good battery life with comparatively low load) as a leader. Although this algorithm depends on a comprehensive weight to elects the leader, it has not considered the topology changes of the ad-hoc networks [18].

3. ALGORITHM OBJECTIVES

The main Objective of KELEA is to reduce the number of nodes that participate in leader election process by selecting a ratio of neighbouring nodes depending on the number of their 1-hop neighbours. Such that only the nodes with large number of neighbours (i.e. high density) will be nominated to participate in the leader election process. By doing so, the algorithm decreases the communication overhead associated with the redundant exchange of ELECTION messages between all nodes in the network. As a result, the level of energy consumption is reduced.

Reducing the number of ELECTION and OK messages during leader election process ensures not only communication overhead reduction and energy saving, but also saves the limited resources of the ad hoc distributed network such as bandwidth, battery power and buffers capacity [7].

It is worthwhile to mention here that in some cases, two or more nodes will have the same density (i.e. the same number of neighbours). To handle such cases, another performance value is deployed in KELEA to solve the tie. This value represents the node's *Energy*. If two nodes *X* and *Y* have the same density, node *X* will be preferred among node *Y* if it has higher energy. Relying on energy as a second performance value ensures high stability and survivability of the network and reduces the occurrences of link breakages, therefore, enhancing the performance of the entire network

4. ALGORITHM ASSUMPTIONS

As in [19], we assume that the mobile ad hoc network is composed of a set of nodes that are connected in the form of connected graph. In this graph, nodes are represented as vertices and communications between nodes are represented by lines. Each node, say *N*, has a transmission range (which is viewed as circles), and other nodes that wish to communicate directly with *N* should be positioned within its transmission range. At any time, some nodes can get into the transmission range of the node *N* while other nodes might get out of that transmission range. Therefore, it is irrational to assume a fixed topology of the network, rather, the topology continuously changes based on the mobility of nodes. As in [19], we only consider those nodes that belong to the graph. That is, nodes that still exist within the transmission radius of some other node. While nodes that leave all of the available transmission ranges and go out of the network scope will not be considered at all.

Furthermore, we assume that each node is assigned with a unique identifier ID which represents a performance value. In KELEA, we used density as a performance value to make preferences among nodes, such that the node with higher density will be targeted to participate in the election process. As mentioned previously, there are situations where two or more nodes will have similar number of neighboring nodes (i.e. densities). In such cases, battery power of nodes is used to solve the tie and make a final selection decision.

5. THE PROPOSED ALGORITHM

In mobile ad hoc networks, a network-wide broadcasting is used to broadcast and disseminate messages between nodes [20]. Flooding, is the major technique that is used to perform the network-wide broadcasting [21]. Although flooding is a not complicated process that is easy to implement and guarantees delivery of messages to every targeted destination; it is of a major expense. Since in flooding, all nodes (without any exception) participate in broadcasting messages, then the leader election process will suffer a lot of redundant messages that inherently increase the message overhead–collision and contention, and delay in the network which in turn consume nodes' resources [21].

In flooding based algorithms, as illustrated in Figure 1, if a node *S* detects the leader crash, it will initiate a leader election process by broadcasting an *ELECTION* message to all of its 1-hop neighbors. In turn, the 1-hop neighboring nodes will rebroadcast the *ELECTION* message to all of their neighbors (i.e. 2-hop neighbors of node *S*) and so on. This process continues until the desired leader is found and elected. It is clear from the figure that number of messages required to perform leader election is high, and this redundant messages will cause heavy traffic on the network. This problem becomes more serious for larger network with larger densities.

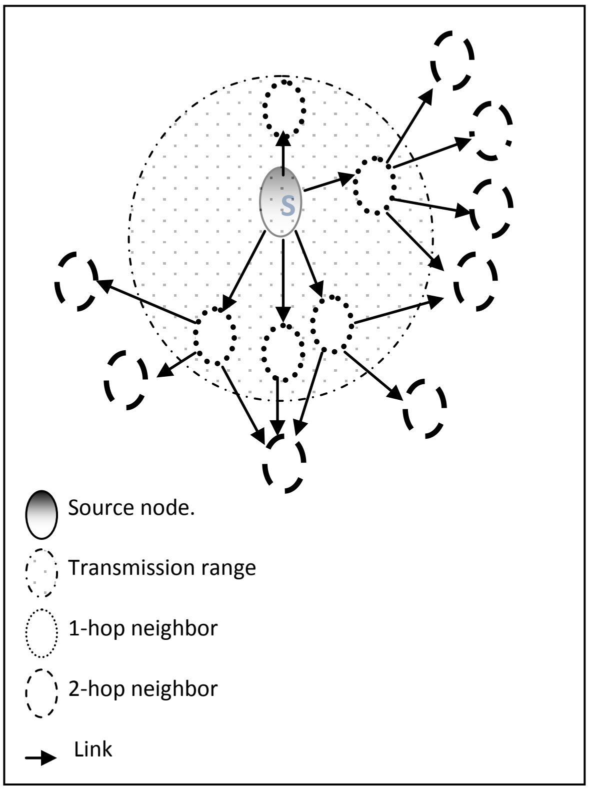

Fig 1: Leader election based on flooding

To overcome these mentioned problems, this paper propose the efficient K-neighbor based leader election algorithm (KELEA) that reduces the number of *ELECTION* messages transmitted between nodes by reducing the number of nodes that participate in the leader election process. The enhancement is achieved by allowing each node in the network to obtain an *Election Set (ES).* The *ES* consists of a set of neighboring nodes that are known to have the largest number of neighbors among all other nodes. Applying KELEA, instead of transmitting *N* (*ELECTION* and *OK* messages) in a network that consist of *N* nodes, only *N/R* messages are transmitted, where *R* is a ratio of *N*. through empirical, *R=3* proved to be the best ratio to be chosen for the *ES*.

The idea of KELEA is illustrated further in Figure 2. According to the figure, when a node *S* detects the crash of the current leader, instead of broadcasting h_1 messages to all it H_1 neighbors (as in the case of flooding-based algorithms), it sends only $h_1/3$ messages, where the $h_1/3$ nodes are known to have larger number of neighbors in comparison with other nodes. Node S prepares its *ES* and appends the addresses of nodes that belong to *ES* with the *ELECTION* message, then it multicasts the *ELECTION* message to its *ES* nodes. In turn, each node in the *ES* will repeat this process in a distributed manner by preparing its own *ES* and multicast the *ELECTION* message further, and so on, until the *ELECTION* message reaches the last hop.

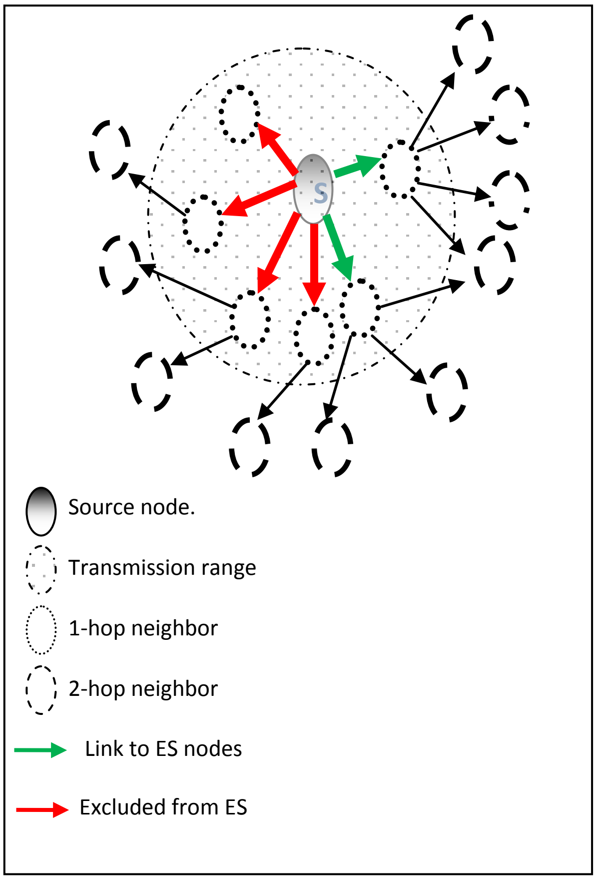

Fig2: Leader election based on KELEA

6. MATHEMATICAL ANALYSIS

This section provides a mathematical analysis of the proposed algorithm, KELEA, and shows its superiority over any algorithm that performs flooding as a basic technique for leader election.

As mentioned previously, in KELEA, if a node S detects the crash of the current leader, it prepares its ES and sends *ELECTION* message to the nodes that belong to this set instead of sending the *ELECTION* message to the entire nodes in its 1-hop transmission range (as in the case of flooding-based techniques).

To analyze the performance of our algorithm, and compare it with the flooding-based algorithms, we have the following assumptions: each node in the network has a specific transmission range (represented as a circle) and all other nodes are distributed and located randomly within these transmission ranges, such that the set of nodes that are positioned within the transmission range of a particular node *x* is known to be the 1-hop neighbors of *x*. To make the organization of nodes within transmission ranges easier, we view the levels of neighborhood in relation with a node S as hops. Where the first hop (1-hop or alternatively H_1) is the first level of neighborhood that includes the direct neighbors of *S*, the second hop (2-hop or H_2) is the second level of neighborhood that includes the neighbors of the direct neighbors of S (i.e. the second neighbors of S), and so on. The last hop (last-hop or H_i) therefore, is known to be the hop that includes nodes with the farthest distance from *S*.

In addition, we assume that the number of nodes in the first hop H_1 is equal to h_1 the number of nodes in the second hop H_2 is equal to h_2, and so on. So the number of nodes in the last hop H_i is equal to h_i.

Regarding energy, the paper assumes that at the early stage of the network construction (i.e. at time= t_0), a network *H* with a number of nodes *N* will have an amount of energy equal to *E*, where each node has an *E/N* units of energy. As the time advances, (that is, at *t1*, *t2*, …) each node consumes part (or all) of its energy. For simplicity, we refer to the *E/N* units of energy that each node possess as *U*. therefore, the total amount of energy *E* in a network of *N* nodes is equal to *N*U.*

Furthermore, the amount of energy consumed for sending a message is assumed to be equals to *X*, so, the total amount of energy consumption in a network of *N* nodes is equal to *N*X*

The following two subsections provide an analysis of the message complexity and energy consumption for flooding-based leader election algorithms and our K-neighbor Based Leader Election Algorithm, KELEA.

6.1. Performance of Flooding-based Leader Election Algorithms:

In an ad hoc network that consists of *N* nodes, if a source node S detects the crash of the leader, it will initiate a leader election process by sending a number of ELECTION messages equals to h_1 to all of its H_1 neighbors. Further, each one of the H_1 neighboring nodes will re-send the ELECTION message to their H_1 neighbors (i.e. the H_2 or 2-hop neighbors of S), with a total number of messages sent equals to H1* h2.

This process continues until reaching, in the worst case, the last hop *i*, where the number of messages that are sent by H_{i-1} nodes will be $H_{i-1} * h_i$.

Along this process, to send a single message, each node consumes an amount of its energy equals to *X*. Therefore, the total amount of consumed energy is equal to *X* multiplied by the number of messages exchanged during the leader election process.

Equation1 shows message complexity associated with flooding-based leader election algorithms. And Equation 2 shows the resulting energy consumption.

$$\text{Complexity}=H_1+H_1\times h_2+H_2\times h_3+\ldots+H_{i-1}\times h_i \quad \ldots(1)$$

$$\text{Energy Consumption}= X(H_1+H_1\times h_2+H_2\times h_3+\ldots+H_{i-1}\times h_i)\ldots(2)$$

6.2. Performance of K-neighbor Based Leader Election Algorithm:

As in [19], the paper assumes that if node *S* has H_1 neighbors in its 1-hop transmission range and H_2 nodes in its 2-hop transmission range and H_i nodes in its i^{th} transmission range, then the number of nodes in the Election Set (ES) for these hops will be $H_1/3$, $H_2/3$, $H_3/3$, … and $H_i/3$, respectively. Where the ratio K=3 is the ratio of nodes in any hop that is chosen to participate in the leader election process and to disseminate *ELECTION* messages further. Therefore, and upon detecting the leader crash by a source node *S*, *S* will send $H_1/3$ *ELECTION* messages to its H_1 neighbors. In turn , each node in the $H_1/3$ set will send $H_2/3$messages to their H_1 neighbors (which represent H_2 neighbors of *S*). This process continues in the same manner until the nodes of the H_{i-1} hop send $H_i/3$messages.. Message complexity associated with the K-neighbor based leader election algorithm is illustrated in equation 2.

$$\text{Complexity} = \frac{H_1}{3} + \frac{H_2}{3} + \frac{H_3}{3} + \ldots + \frac{H_i}{3} \quad \ldots (3)$$

If the ad hoc network has N nodes distributed along n hops, then the number of transmitted messages can be generalized as in Equation 4:

$$\text{Complexity} = \frac{\sum_{i=1}^{n} H_i}{3} \quad \ldots (4)$$

Equation 5 presents the amount of energy consumption associated with KELEA based algorithm:

$$\text{Energy Consumption} = X\left(\frac{\sum_{i=1}^{n} H_i}{3}\right) \quad \ldots (5)$$

6.3. A practical Example

This section provides a practical example that illustrates the message complexity and energy consumptions associated with both flooding-based and KELEA leader election algorithms. The complexity and energy consumption are computed according to the equations that are discussed in the previous section. But before elaborating further with the calculations, it is necessary to illustrate the procedure that we follow to compute the average number of neighbors that are expected to be available within any transmission range. According to [22], the average number of neighbors, *n*, that are reachable within any transmission range *R* in a network consisting of *N* nodes and an area *A*, is calculated according to the following equation:

$$n = (N - 1)\frac{\pi R \times R}{A} \quad \ldots\ldots (6)$$

Table 1 shows the expected average number of neighbors for networks of different number of nodes N distributed among an area of 1000* 1000, and the transmission range is 250 m.

Table 1. Average Number of Neighbors

Area (A)	Nodes (N)	avg # of neighbors (n)
1000* 1000	25	4.7 ≈ 5
1000* 1000	50	9.6 ≈ 10
1000* 1000	75	14.5 ≈ 15
1000* 1000	100	19.4 ≈19

To illustrate complexity and power consumption associated with KELEA and naïve flooding-based algorithms, equations 1, 2, 4, and 5 are applied, respectively. To determine the number of hops in the network, we assume that the source node that needs to initiate a leader election process is located in the farthest angle in relation with the target or destination nodes that needs to receive the ELECTION message. For a network with an area A=1000* 1000 m, the distance between two nodes that are located opposite to each other across the diagonal is 1414 m. therefore, for a transmission range R= 250 m, the minimum number of hops will be 5. For the purpose of computing the power consumption, an amount of power units *U* is assumed to be consumed by each message sending operation, through the experiments in this paper, *U* is assumed to be 0.2. Table 2 shows the performance results for the supposed case where the number of hops = 5 and power consumption/message (*U*) = 0.2.

Table 2. Performance analysis for number of hops=5 and U= 0.2

		Number of nodes (N)			
		25	**50**	**75**	**100**
Complexity	KELEA	10	20	25	35
	Flooding	105	410	915	1463
Power Consumption	KELEA	2	4	5	7
	Flooding	21	82	183	292

7. CONCLUSION and Future Works

This paper proposed an election algorithm that is aware about both density and energy of the ad hoc distributed systems. The proposed algorithm, KELEA, is an energy-efficient algorithm which aims to save energy by reducing the number of exchanged messages through reducing the number of nodes that participate in the leader election process. The use of KELEA introduces a major contribution in our work, since it reduces the message overhead associated with simple flooding-based leader election algorithms and reduces power and resources consumption.

Although the mathematical analysis and the supported practical example show the preference of the proposed algorithm, it still needs more experiments and simulation tests to get more accurate comparison between KELEA and other flooding-based algorithms.

8. REFERENCES

[1] Mehdi Mirakhorli, Amir A. Sharifloo, Maghsoud Abbaspour, "A Novel Method for Leader Election Algorithm". The 7th IEEE International Conference on Computer and Information Technology (CIT 2007), pp.452-456, October 2007.

[2] Mina Shirali, Abolfazl Haghighat Toroghi, and Mehdi Vojdani "Leader election algorithms: History and novel schemes". Third 2008 International Conference on Convergence and Hybrid Information Technology(ICCIT'08),pp. 1001-1006, November 2008.

[3] Scott D. Stoller. "Leader Election in Asynchronous Distributed Systems". IEEE Transaction on Computers Journal, volume 49, no. 3 pp.283- 284, March 2000.

[4] G.L. Lann, "Distributed Systems - Towards a Formal Approach", in Proc. IFIP Congress, pp.155-160, 1977.

[5] Randolph Franklin "An improved algorithm for decentralized extrema-finding in circular configurations of processes". Communications of the ACM Magazine. pp.336–337, May 1982.

[6] H.Garcia Molina "Election in a distributed Computing System". IEEE Trans. Comp, 1982, vol31, no. 1, pp. 48-59. 1982

[7] Nourddine E, Mohammed K, Amine B. "Enhancing AODV Performance based on Statistical Mobility Quantification". The IEEE International Conference on Information & Communication Technology (ICTTA06), pp. 2455-2460, 2006.

[8] Quazi Ehsanul Kabir Mamun, Salahuddin Mohammad Masum, and Mohammad Abdur Rahim Mustafa. "Modified bully algorithm for electing coordinator in distributed systems". WSEAS Transactions on Computers, Issue 4, Volum 3, pp. 948-953, October 2004.

[9] Muneer Bani Yassein, Ala'a N Alslaity and Sana'a A Alwidian "An Efficient Overhead-aware Leader Election Algorithm for Distributed Systems." International Journal of Computer Applications, volume 49 no. 6, pp: 10-15, July 2012.

[10] M. Gholipour, M. S. Kordafshari, M. Jahanshahi, A. M. Rahman. A New Approach For Election Algorithm in Distributed Systems. Second International Conference on Communication Theory, Reliability, and Quality of Service. IEEE (2009).

[11] Sung-Hoon Park . A Stable Election Protocol based on an Unreliable Failure Detector in Distributed Systems. Eighth International Conference on Information Technology: New Generations. IEEE 2011

[12] G.L. Lann, "Distributed Systems - Towards a FormaApproach", in Proc. IFIP Congress, pp.155-160, 1977.

[13] Leila Melit, Nadjib Badache. An Energy Efficient Leader Election Algorithm for Mobile Ad Hoc Networks. 10th International Symposium on Programming and Systems (ISPS), IEEE 2011.

[14] Sudarshan Vasudevan, Jim Kurose, Don Towsley. Design and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks. Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP'04). 2004.

[15] M. A. Haddar, A. Hadj Kacem, Y. M´etivier, M. Mosbah, and M. Jmaiel. Electing a Leader in the Local Computation Model Using Mobile Agents. International Conference on Computer Systems and Applications, ICCSA 2008. IEEE/ACS.

[16] N. Malpani, J. Welch and N. Vaidya. "Leader Election Algorithms for Mobile Ad Hoc Networks". DIALM '00 Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications, Boston, MA, pp. 96-103, August 2000.

[17] V.D.Park and Scott.M.Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", INFOCOM '97 Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution, pp. 1405, 1997.

[18] Neepa Biswas, Leader Election In Mobile Ad-Hoc Networks. [Disseration]. Bengal Engineering And Science University, Shibpur. April 2010.

[19] Sanaa A. Alwidian , Alaa N. Alslaity. A Stable Covering Set-based Leader Election Algorithm (SCLEA) for Mobile Ad-hoc Distributed Systems Leader Election Algorithm (SCLEA) for Mobile Ad hoc Distributed Systems" International Journal of Computer Applications, Vol 54, no. 2, pp.:25-32, September 2012.

[20] Paul Rogers, & Nael Abu-Ghazaleh (2005), *Robustness of network-wide broadcasts in MANETs.* Proc. of the 2nd IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2005), Vol. 3, Nov. 2005, pages 161-185

[21] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, & Jang-Ping Sheu (2002), *The broadcast storm problem in a mobile ad hoc network.* Wireless Networks, 8 (2), pages 153-167.

[22] Z. Qi and D. P. Agrawal, "Analysis of Leveled Probabilistic Routing in Mobile Ad hoc Networks," presented at The IEEE International Conference on Communications (ICC'04), Paris, France, 2004.