Research Journal of Applied Sciences, Engineering and Technology 11(4): 440-446, 2015
DOI: 10.19026/rjaset.11.1800
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: April 27, 2015 Accepted: May 28, 2015 Published: October 05, 2015

Research Article

Fault Tolerant Tree Management for Node Failures in Wireless Sensor Network

[1]S. Kamalesh, [2]P. Ganesh Kumar and [3]A. Wazim Raja
[1]Anna University, Chennai, India
[2]Linyi Top Network Co., Ltd., Linyi City, Shangdong Province, China
[3]Department of IT, ME-Network Engineering, Velammal College of Engineering and Technology, KLNCE, Sivaganga Dt., Madurai, India

Abstract: Today, Wireless Sensor Network (WSN) is a vital technology required in every phase of development. During data aggregation in Wireless Sensor Network (WSN), there may be possibility of various types of failures. Apart from hardware failures, failures may occur due to bad channel conditions and energy drain. Hence a routing tree should be built towards each sink which is fault tolerant. In this study, we propose to develop a fault tolerant tree management technique in which every node selects a path towards the sink effectively and also maintains a backup path simultaneously. This can be achieved by mutual communication between the neighboring nodes in the network. The backup path is selected based on the link quality, residual energy and load metrics. Simulation results show that the proposed technique reduces the delay and overhead and improves packet delivery ratio.

Keywords: Fault tolerant tree, packet delivery ratio, wireless sensor network

INTRODUCTION

Wireless Sensor Network (WSN): Efficient design and implementation of wireless sensor networks has become a hot area of research in recent years, due to the vast potential of sensor networks to enable applications that connect the physical world to the virtual world. By networking large numbers of tiny sensor nodes, it is possible to obtain data about physical phenomena that was difficult or impossible to obtain in more conventional ways (Stankovic, 2006). A sensor network is composed of a large number of sensor nodes, which are densely deployed either inside the phenomenon or very close to it. The position of sensor nodes need not be engineered or pre-determined (Alobaisat and Braun, 2007).

The applications of WSN are in military, environment, health, home and commercial areas, space exploration, chemical processing and disaster relief. In many WSN applications, the deployment of sensor nodes is performed in an ad hoc fashion without careful planning and engineering. Once deployed, the sensor nodes must be able to autonomously organize themselves into a wireless communication network. Sensor nodes are battery-powered and are expected to operate without attendance for a relatively long period of time (Bhoopathy and Parvathi, 2011).

Types of faults in WSN:

Node faults: Nodes have several hardware and software components that can produce malfunctions. Due to stress from the environment and inadequate enclosures, the sensor nodes were exposed to direct contact with water causing short circuits (Aron *et al*., 2010).

Link faults: In WSNs, communication links between nodes are highly volatile (Khan *et al*., 2009). Instability of the link between nodes causing network partitions and dynamic changes in network topology leads to network level faults (Khan *et al*., 2009).

Sink faults: Failure of the sink leads to a massive failure of the network. At the sink level, software, that store and process data are subject to bugs and can lead to loss of data within the period when fault occurs (Khan *et al*., 2009).

Fault management: Fault management approaches can be classified into three main phases: fault detection, fault diagnosis and fault recovery. Fault detection is the first phase, where faults and failures in the network are properly identified by the management system. The aim of the fault detection is to ensure that the services are functioning properly (Liu *et al*., 2009). Fault diagnosis is a stage that the causes of detected fault can be properly identified and distinguished from the other irrelevant or spurious alarms. The failure recovery phase is the stage at which the sensor network is restructured or reconfigured, in such a way that failures or faulty nodes do not impact further on network performance (Yu *et al*., 2007).

Corresponding Author: S. Kamalesh, Anna University, Chennai, India

Most existing fault management solutions mainly focus on failure detection and there is still no comprehensive solution available for fault management in WSNs. Different mechanisms proposed for fault recovery are not directly relevant to fault recovery in respect of the network system level management i.e., network connectivity and network coverage area etc. Some management frameworks require the external human manager to monitor the network management functionalities (Asim *et al*., 2010).

Problem identification: Some of the existing issues faced by Chakraborty *et al*. (2013) even after having a backup parent node are:

- The alternate parent node is selected randomly and hence its behavior cannot be predicted in terms of reliability, latency and possible link quality.
- The messages in transit when the node failure occurs are lost and cannot be retrieved.
- There is a delay in the delivery of application messages, which affects node communication.

LITERATURE REVIEW

Khedo *et al*. (2010) has proposed "Redundancy Elimination for Accurate Data Aggregation in Wireless Sensor Networks". In monitoring systems, multiple sensor nodes can detect a single target of interest simultaneously and the data collected are usually highly correlated and redundant. If each node sends data to the base station, energy will be wasted and thus the network energy will be depleted quickly. Data aggregation is an important paradigm for compressing data so that the energy of the network is spent efficiently. In this study, a novel data aggregation algorithm called Redundancy Elimination for Accurate Data Aggregation (READA) has been proposed. By exploiting the range of spatial correlations of data in the network, READA applies a grouping and compression mechanism to remove duplicate data in the aggregated set of data to be sent to the base station without largely losing the accuracy of the final aggregated data. One peculiarity of READA is that it uses a prediction model derived from cached values to confirm whether any outlier is actually an event which has occurred.

Ozdemir and Xiao (2013) have proposed fault-tolerant data aggregation scheme that eliminates the false data sent by malfunctioning and/or compromised sensor nodes. To conserve energy while eliminating false data, an in-network outlier detection technique that is based on locality sensitive hashing scheme is used. it is also observed that if sensor data are highly correlated FTDA can eliminate redundant data transmissions and reduce the overall data transmission in the network.

Ozdemir and Cam (2010) have presented a Data Aggregation and Authentication protocol (DAA) to integrate false data detection with data aggregation and confidentiality. To support data aggregation along with false data detection, the monitoring nodes of every data aggregator also conduct data aggregation and compute the corresponding small-size message authentication codes for data verification at their pairmates. To support confidential data transmission, the sensor nodes between two consecutive data aggregators verify the data integrity on the encrypted data rather than the plain data.

Hosseini and Haghparast (2011) have proposed a cluster based method for fault detection and network connectivity recovery. It uses some of the nodes as gateway nodes in the network for implementing of voting mechanism. It deals with the fault detection and network connectivity recovery mechanisms after the stage of cluster formation.

Chakraborty *et al*. (2013) have proposed a Convergecast tree management technique during arbitrary node failure in sensor network. In this study, a set of distributed algorithms has been proposed to construct a BFS tree for data gathering. Each node computes its alternate path to the sink, based on neighborhood information collected during the tree construction. Thus when a node fails, all its neighbors can take actions in constant time to repair the tree locally. If both the parent and alternate parent fail at a time, the proactive approach of repairing does not work. An extended repairing scheme has been proposed that works reactively to find the alternate path to the root in this adverse scenario.

METHODOLOGY

Proposed BFS tree management technique:

Overview: In this study, we propose to develop a scheme that builds a data gathering tree rooted at the sink (Fig. 1). The tree eventually becomes a Breadth First Search (BFS) tree where each node maintains the shortest hop-count to the root to reduce the routing delay. Each node collects some extra neighborhood information during the tree construction. Thus a little pre-processing at each node helps in taking prompt actions to repair the tree through local adjustment if any arbitrary single or multiple nodes fail in future. On failure of a node, each affected node in its vicinity fixes the parent through a pair of control message transmissions. But, while selecting a backup parent node, Residual Energy of the node, Link Quality or channel condition and Average Load are the parameters to be considered in order to assure that the alternate parent node does not fail due to energy drain, bad channel condition (link quality).

Formation of a data tree:

Developing a BFS tree: Breadth-First Search (BFS) is a graph search algorithm that begins at the root node

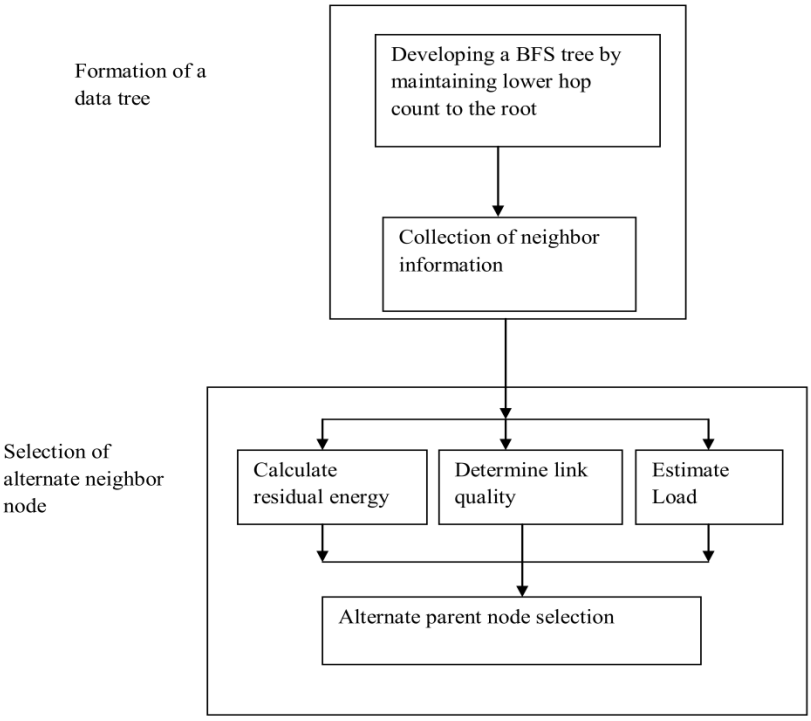


Fig. 1: Data gathering tree rooted at the sink

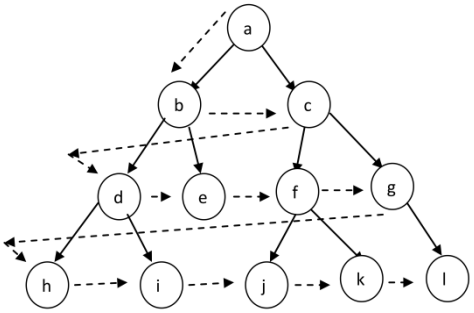


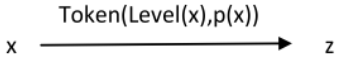
Fig. 2: Breadth-First Search (BFS) tree

and explores all the neighboring nodes. Then for each of those nearest nodes, it explores their unexplored neighbor nodes and so on, until it finds the goal. It has been proved by induction that the breadth first search tree is a shortest path tree starting from its root. Every vertex has a path to the root, with path length equal to its level (just follow the tree itself) and no path can skip a level so this really is a shortest path (Mahajan and Malhotra, 2011).

The BFS formation process is described in algorithm 1 (Fig. 2):

- The nodes at each level of the graph are stored using queues.
- These stored nodes are then treated one by one and their adjacent nodes are visited.
- As the nodes are visited, the terminating condition is reached when the queue is empty.
- In the beginning all nodes will be in ready state. A node that has not been visited yet and waiting to be processed will be in ready state.
- As soon as the node is added on to queue, it will be in waiting state.
- After processing the node i.e., whose neighbors have been added on to queue will be in processed state. Nodes that have been processed once will not be considered again.

Collection of neighbor information: Nodes send Token message to the neighboring nodes. Upon receiving Token (mLevel, pid) message for the first time from a node x, the node z first updates the variables like id of node x and level of x in its neighbor table (Chakraborty *et al*., 2013):



Here, x is the sending node and z is the receiving neighbor node of x. On receiving the Token message from x, the neighbor node z receives the node id and level value of x and updates this information in its neighbor table.

In this way, each node collects the information of every neighboring node and stores this information in its neighbor table.

Alternate parent node selection: A node with high residual energy, good link quality and less average load should be selected as the alternate parent node.

Calculation of residual energy: A probabilistic based prediction model is used (Shenkutie and Patil Shinde, 2011). Where the modes of operation of sensor nodes are represented by the states of a Markov chain and the probability of entering into each state is denoted by a random variable.

The residual energy estimation is described in algorithm 2.

Algorithm-2:

1. Each sensor node has L modes of operations and each node is modeled by a Markov chain with L states.
2. For a node currently in state i, the probability of being in state j in the next time-step is represented by p_{ij}.
3. The n-step transition probability, p_{ij}, that a node currently in state i will be in state j after n transitions, can be defined as given by:

   $p_{ij}^{(n)} = \sum_{k=1}^{m} P_{ik}^{(r)} P_{kj}^{(n-r)}$ where 0<r<n

4. With p_{ij}, the possibility to predict the amount of time steps a node initially in state x_0, spends in state j, in the next T time steps is given by $\sum_{t=1}^{T} P_{ij}^{(t)}$.
5. The amount of energy the node will consume in the next T time steps E^T is also calculated as:

   $$E^T = \sum_{a=1}^{L} (\sum_{t=1}^{T} P_{ia}^{(t)}) * E_a$$

   where E_a, is the energy consumed by a node in state a.
6. Since each node has to follow its current state at each time-step, it has to maintain a probability matrix and update it at each time-step.
7. The residual energy is calculated as follows:

   Residual energy, $E^R = E^{Total} - E^T$

where, E^{Total} is the initial total available energy at the node.

The nodes which consume higher energy, will have lesser residual energy. Hence, the node which consumes the least amount of energy will possess the highest amount of residual energy.

Estimation of the channel condition/Link Quality (LQ): The Received Signal Strength Indicator (RSSI) feature is used to distinguish links which have approximately the same Packet Reception Rate (PRR). We thus define a metric (Rondinone *et al*., 2008) which can take into account the RSSI information obtained from the radio by:

Link Quality Indicator, LQI = PRR × norm (RSSI_avg) (1)

The average value of the RSSI, RSSI_avg varies over the range (-100, -40) dBm values by the radio chip.

The normalized value of RSSI_avg is calculated according to the equation:

$$norm\ (RSSI\_avg) = \frac{meanRSSI}{60} + \frac{100}{60} \quad (2)$$

It is normalized to (0, 1) so that PRR and the mean RSSI are combined in a fair way.

Estimating average load: Every node estimates its effective queue length, EQL based on the virtual queue length (Basaran *et al*., 2010) observed packet drops and the effect of probabilistic load balancing. The load at the node is directly proportional to the EQL. Hence, if the EQL increases then the load at the node also increases. The EQL/load at node i is estimated as follows:

$$EQL_i = \sum_{j=1}^{N_i} (P_j \times q_j)$$

where,

$$q_j = MAC_i + NQ_j + \beta \times D_i$$

where,

MAC_i = The number of packets in node i's MAC layer queue

NQ_j = The number of the packets in node j's neighbor Queue

D_i = The number of packets dropped by n_i due to an excessive number of retransmissions after the most recent successful transmission

β = The limit for retransmitting a single packet

Using the virtual queue length q_j, the weight w_j for node j ∈ FS_i is computed:

$$w_j = \alpha^{qj}$$

where, 0<a<1. w_j becomes smaller as q_j increases, because a<1.
And,

$$P_j = \frac{w_j}{\sum_{j=1}^{N_i} w_j}$$

where N_i is the cardinality of FS_i.

Alternate parent node selection: Using the techniques described in above sections, every candidate node calculates the residual energy, Link Quality and load condition at the neighboring nodes. The neighboring node which possess highest residual energy, highest link quality and optimum load balance among the several other nodes is selected as the alternate parent node.

Data tree management algorithm: The Overall Data Tree management process is described in algorithm 3.

Algorithm-3:

1. Developing a data gathering tree to form a Breadth First Search (BFS) Tree.
2. Maintaining lower hop count to the root, also called as sink in order to reduce routing delay.
3. Collection of the neighboring node information by every node.
4. Parent node is selected using Token and Add messages.
   Based on the information present in the neighbor table of node z, it selects a node with the lowest level value in its neighbor table as its parent. Then z sends an Add (FALSE) message to its parent for confirmation. The FALSE argument with the Add message forces the parent node to add z immediately in its Child set. z also sends Token message with its updated information to all its neighbors:

   $y \xleftarrow{\text{Add(FALSE)}} z$

5. Alternate parents are selected for emergency conditions, based on three parameters: Residual Energy of the node, Link Quality or channel condition and Average Load.

SIMULATION RESULTS

Simulation setup: Fault Tolerant Tree Management (FTTM) technique is evaluated through NS-2 (Chakraborty *et al*., 2013) simulation. A random network deployed in an area of 500×500 m is considered. We vary the number of nodes as 20, 40….100. Initially the nodes are placed randomly in the specified area. The base station is assumed to be situated 100 m away from the above specified area. The initial energy of all the nodes is assumed as 10.1 joules. In the simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. The Distributed Coordination Function (DCF) of IEEE 802.11 is used for wireless LANs as the MAC layer protocol. The simulated traffic is CBR with UDP source and sink. The performance of FTTM technique is compared with Convergecast (Chakraborty *et al*., 2013) protocol. Both the techniques FTTM and Convergecast are tested in grid and random network topologies. Table 1 and 2 summarizes the simulation settings used for Grid and Random topologies, respectively.

Table 1: Simulation settings for grid topology

No. of nodes	8, 16, 24, 32 and 40
No. of failed nodes	1 to 5
Area size	1000×1000 m
MAC	802.11
Simulation time	50 sec
Packet size	512
Transmit power	0.660 w
Receiving power	0.395 w
Idle power	0.035 w
Initial energy	10.1 J
Transmission range	75 m
Rate	30 Kb

Table 2: Simulation settings for random topology

No. of nodes	50, 100, 150, 200 and 250
No. of failed nodes	2
Simulation time	50 sec
Transmit power	0.660 w
Receiving power	0.395 w
Idle power	0.035 w
Initial energy	20.1 J
Transmission range	75 m
Rate	30 Kb

Results:

Grid topology: For grid topology, the number of nodes and the number of arbitrary failed nodes are varied and the performance metrics failure recovery delay, packet delivery ratio, residual energy and control overhead are measured.

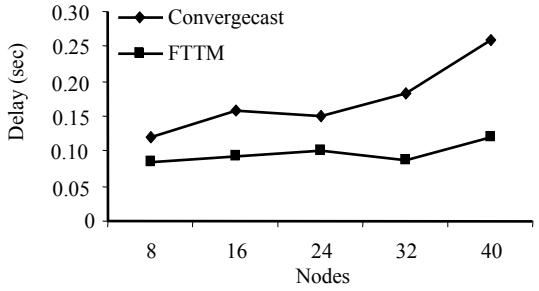


Fig. 3: Nodes vs. delay

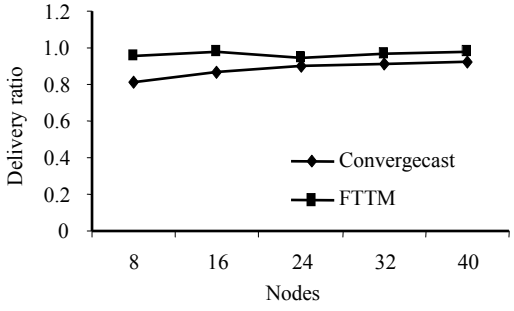


Fig. 4: Nodes vs. delivery ratio

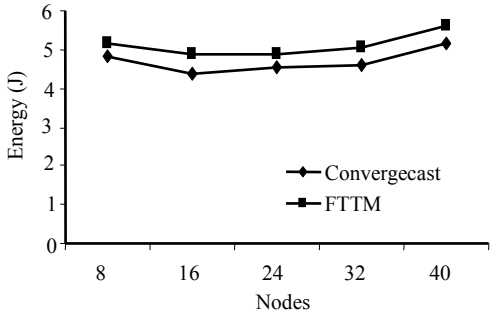


Fig. 5: Nodes vs. residual energy

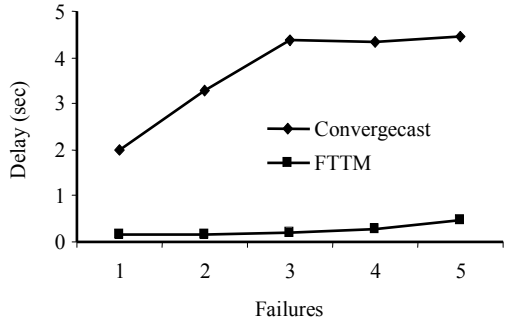


Fig. 6: Failures vs. delay

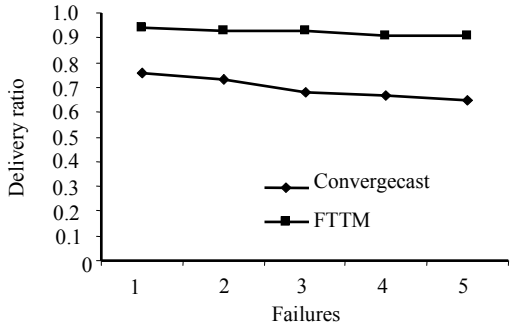


Fig. 7: Failures vs. delivery ratio

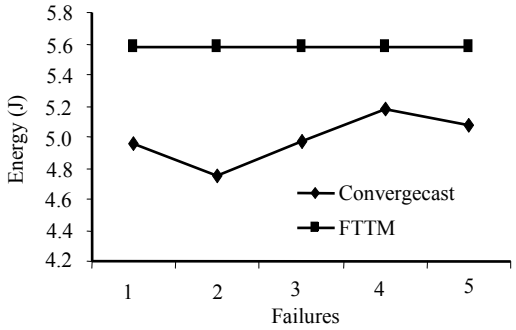


Fig. 8: Failures vs. residual energy

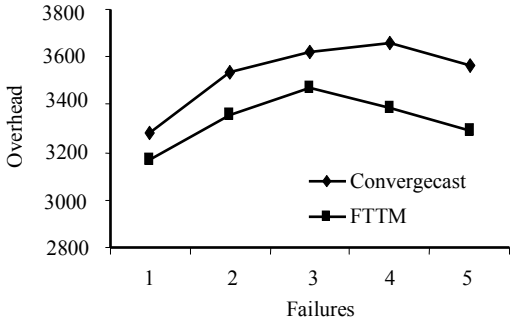


Fig. 9: Failures vs. overhead

Varying number of nodes: The number of grid nodes is varied as 8, 16, 24, 32 and 40, respectively.

Figure 3 to 5 show the results of delay, delivery ratio and residual energy for Convergecast and FTTM by varying the number of nodes. From the figures, it

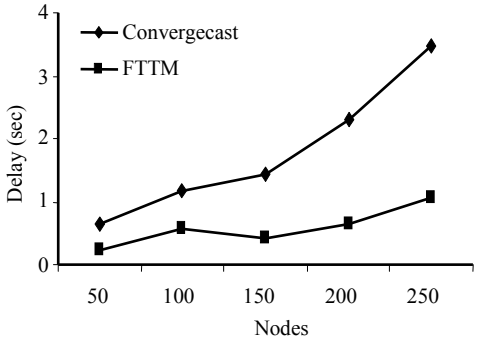


Fig. 10: Nodes vs. delay

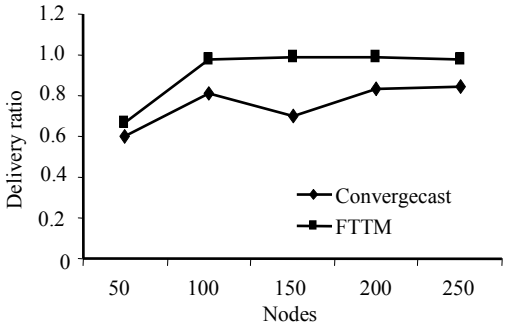


Fig. 11: Nodes vs. delivery ratio

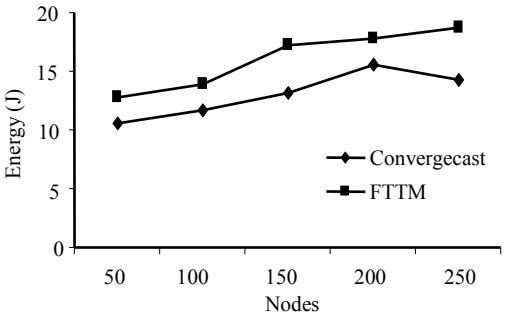


Fig. 12: Nodes vs. residual energy

can be seen that FTTM outperforms Convergecast in terms of delay by 42%, in terms of delivery ratio by 9% and in terms of residual energy by 8%.

Varying number of failures: The number of arbitrary node failures is varied from 1 to 5.

Figure 6 to 9 show the results of delay, delivery ratio, residual energy and overhead for Convergecast and FTTM by varying the number of failed nodes. From the figures, it can be seen that FTTM outperforms Convergecast in terms of delay by 90%, in terms of delivery ratio by 25%, in terms of residual energy by 10% and in terms of overhead by 5%.

Random topology: For grid topology, the number of nodes is varied from 50 to 250 and the performance metrics failure recovery delay, packet delivery ratio and residual energy are measured.

Figure 10 to 12 show the results of delay, delivery ratio and residual energy for Convergecast and FTTM by varying the number of nodes. From the figures, it can be seen that FTTM outperforms Convergecast in terms of delay by 65%, in terms of delivery ratio by 18% and in terms of residual energy by 18%.

CONCLUSION

In this study, we develop a data gathering tree using the Breadth First Search (BFS) Tree technique. Every Node selects parent nodes using Token and Add messages. Next every node selects an alternate parent node as a backup in case of emergency due to parent node crash or path failure. The alternate parent node is chosen such that it satisfies the three criteria required i.e., higher residual energy, good link quality and reduced load at node. The surrounding node that fulfills the three requirements to a higher extent is chosen as the alternate parent node. Thus there is no delay in data delivery in this technique as we choose alternate parent node with reduced load, use better channels for data transmission and the operating nodes have higher energy to perform this operation. Simulation experiments are conducted for both grid and random network topologies and results show that the proposed technique reduces the delay, overhead and improves delivery ratio.

REFERENCES

Alobaisat, Y. and R. Braun, 2007. On Wireless Sensor Networks: Architectures, Protocols, Applications and Management. Springer-Verlag London Ltd., DOI: 10.1007/978-1-84882-218-4 10.

Aron, Z.C., W. Al-Khateeb and F. Anwar, 2010. The enhanced fault-tolerance mechanism of AODV routing protocol for wireless sensor network. Int. J. Comput. Sci. Netw. Secur., 10(6).

Asim, M., H. Mokhtar and M. Merabti, 2010. A self-managing fault management mechanism for wireless sensor networks. Int. J. Wirel. Mob. Netw., 2(4).

Basaran, C., K.D. Kang and M.H. Suzer, 2010. Hop-by-hop congestion control and load balancing in wireless sensor networks. Proceeding of the IEEE 35th Conference on Local Computer Networks (LCN, 2010), pp: 448-455.

Bhoopathy, V. and R.M.S. Parvathi, 2011. Energy efficient secure data aggregation protocol for wireless sensor networks. Eur. J. Sci. Res., 50(1): 48-58.

Chakraborty, S., S. Nandi and S. Karmakar, 2013. Convergecast tree management from arbitrary node failure in sensor network. Ad Hoc Netw., 11: 1796-1819.

Hosseini, S. and M. Haghparast, 2011. CHFM: Cluster-based and hierarchical fault management to fault detection and network connectivity recovery in wireless sensor networks. Aust. J. Basic Appl. Sci., 5(7): 243-248.

Khan, M.Z., M. Merabti and B. Askwith, 2009. Design Considerations for Fault Management in Wireless Sensor Networks. Proceeding of the 10th Annual PGNET'09 Conference. LJMU, Liverpool, UK.

Khedo, K., R. Doomun and S. Aucharuz, 2010. READA: Redundancy elimination for accurate data aggregation in wireless sensor networks. Lect. Notes Comput. Sc., 2: 300-308.

Liu, H., A. Nayak and I. Stojmenovic, 2009. Fault Tolernat Algorithms/Protocols in Wireless Sensor Networks. In: Misra, S. *et al*. (Eds.), Guide to Wireless Sensor Networks, Computer Communications and Networks. Springer-Verlag, London, pp: 265-295.

Mahajan, S. and J. Malhotra, 2011. Energy efficient path determination in wireless sensor network using BFS approach. Lect. Notes Comput. Sc., 3: 351-356.

Ozdemir, S. and H. Cam, 2010. Integration of false data detection with data aggregation and confidential transmission in wireless sensor networks. IEEE ACM T. Network., 18(3).

Ozdemir, S. and Y. Xiao, 2013. FTDA: Outlier detection-based fault-tolerant data aggregation for wireless sensor networks. Lect. Notes Comput. Sc., 6(6): 702-710.

Rondinone, M., J. Ansari, J. Riihijarvi and P. Mahonen, 2008. Designing a reliable and stable link quality metric for wireless sensor networks. Proceeding of the Workshop on Real-World Wireless Sensor Networks (REALWSN'08). Glasgow, United Kingdom.

Shenkutie, D.K. and P.K. Patil Shinde, 2011. Residual energy monitoring in wireless sensor networks. Technical Report, IDE1152.

Stankovic, J.A., 2006. Wireless Sensor Networks. Department of Computer Science, University of Virginia, Charlottesville, VA.

Yu, M., H. Mokhtar and M. Merabti, 2007. A Survey on Fault Management in Wireless Sensor Networks. ISBN: 1-9025-6016-7 © PGNet.