Fault Tolerant Range Grouping Routing in Dynamic Networks

Akanksha Bhardwaj, Prachi Badera, and K. Rajalakshmi

Department of Computer Science,
Jaypee Institute of Information Technology, Noida
{busy.akanksha,prachbadera}@gmail.com,
k.rajlakshmi@jiit.ac.in

Abstract. A characteristic feature of dynamic networks is the notion of failure. A failure can be a partial failure [as in distributed systems] or total failure. A partial failure may happen when one component in a system fails. This failure may affect the proper operation of other components, while at the same time leaving yet other components totally unaffected. In contrast, a failure in any system is often total in the sense that it affects all components, and may easily bring down the entire system. Hence, it becomes important to design a system which can work even if (partial) failures occur. This paper proposes various approaches which help in making the designed system fault tolerant. Mainly the routing mechanism is focused upon with the help of the concept of acknowledgment and negative acknowledgment.

Keywords: Routing, Crash failure, Omission failure, Fault detection, Fault recovery.

1 Introduction

Dynamic networks have received significant attention in recent years due to the widely increasing application of Dynamic Networks in various fields. The nodes in such a network collaborate with each other to perform tasks such as data communication, data processing, data analyzes, etc. Since the network formed is dynamic in nature, its components have high probability to fail under given scenario [1].

Fault tolerance is the ability of a system to deliver a desired level of functionality in the presence of faults. Actually, extensive work has been done on fault tolerance and it has been one of the most important topics in Dynamic Networks. The objective of this research paper is to investigate the current research work on fault tolerance in Dynamic Networks.

Failures determined by nature are the identity of the faulty processes and the details of their faulty behavior. These depend on the particular failure model being assumed. In this application, we consider two closely related failure models, called crash and omission. Here crash implies departure of the node from the topology. In the omission model, a faulty process may omit to send/receive messages [2].

K. Singh, A.K. Awasthi, and R. Mishra (Eds.): QSHINE 2013, LNICST 115, pp. 218–227, 2013.

Another problem faced in dynamic network is arrival of a new node. In a node arrival /departure [crash model], it is necessary to provide the user with a new topology.

A brief overview of the network designed is given in section 2. In section 3 how fault happens in different levels of Dynamic network is discussed. Fault detection and recovery algorithms are introduced in section 3.4.1. Results of evaluation performed for this approach are shown in section 4.

2 Network Designed

In order to create large networks, involvement of large nodes is required. However, maintenance of large number of nodes is not easy. Therefore, for proper maintenance of large nodes, we first cluster the nodes and then perform routing among the nodes. A network with 9 nodes has been created in our previous work is described in detailed manner in [3]. A short description of the network is as follows:

2.1 Clustering

Let node A be the master and suppose it first runs the application to detect its nearby devices. Node A finds out nodes B, C, D in its cluster and store the address and the name in their data structures. Hence it forms an INTRA cluster with the nodes B, C, D. Similarly, nodes B, C, D, E, F, G, H, I will find the nodes lying in its range and store this information in their lists respectively. Next, these lists are exchanged with the nearby nodes. At each node, a comparison between the nodes of its list and the list received from its neighbors takes place. A node that is found to be different is appended in its list. These newly added nodes form the INTER cluster where the nodes of inter cluster becomes gateway to reach the nodes of inter cluster. This creates an initial topology of the network. The topology is prone to changes [since it is a dynamic Network], therefore addition and deletion of node takes place according to their Bluetooth ranges.

2.2 Routing

After the formation of the clusters, the main task that remains is forwarding of the data. In order to accomplish this, the nodes have the list available with itself which it can refer to know if the node exists in the network or not. After mentioning the destination node, the node looks at the list. The data are sent to the neighboring node via *hoping*. When the data reach the ***gateway*** node (i.e. the node which is a part of more than one cluster), it looks for the route with the help of the lists again and then again routes the data. If the gateway crashes it looks for another path and sends the packets via another gateway.

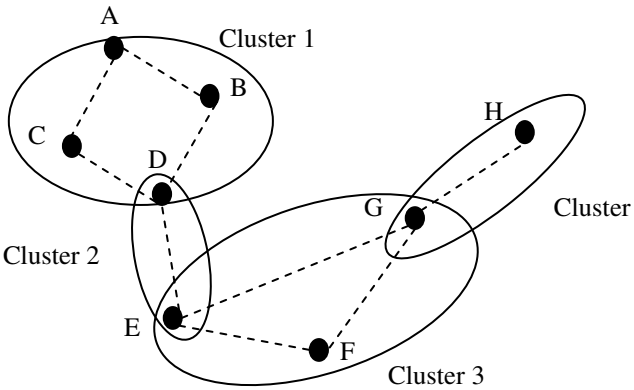

Fig. 1. Four clusters are formed covering all the nodes in the network

3 Fault Tolerance

Five levels of fault tolerance were discussed in [5]. They are physical layer, hardware layer, system software layer, middleware layer, and application layer. On the basis of study, we classify fault tolerance in WSNs into four levels from the system point of view.

To tackle faults the system should follow two main steps. The first step is fault detection. It is to detect that a specific functionality is faulty, and to predict it will continue to function properly in the near future. After the system detects a fault, fault recovery is the second step to enable the system to recover from the faults.

We consider two closely related failure models, called crash and omission. Here crash implies departure of the node from the topology. In the omission model, a faulty process may omit to send/receive messages.

3.1 Assumptions

- Synchronous system
- Communication delay is bounded.
- Message delivery is ordered.
- Uni-casting.

3.2 Hardware Layer

Faults at hardware layer can be caused by malfunctioning of any hardware components. As in the network designed, the hardware component used is Bluetooth device. Due to power limitations of Bluetooth devices, excessive use of a Bluetooth device can result in its malfunctioning. Secondly, due to the environment which may contain other radio wave radiations which might result in interfere with Bluetooth's working and hence create problem for Bluetooth users.

Solution. Try to use the Bluetooth in the environment where there are no other radiations. And for power consumption, always put the device on charging while using it.

3.3 Software Layer

The network has two software components:

a. Operating system
b. Middleware such as routing.

Solution. In order to reduce the probability of operating system failure, the designed application should be tested with different versions so that even if one results in a failure, the application can still run in other version.

For fault tolerant routing, the concept of acknowledgement and negative acknowledgement is introduced. The detailed algorithm is explained in the following sections.

3.4 Network Communication Layer

The routing in Dynamic Networks is the most prone area to faults. There may be a possibility that the receiver doesn't receive the sent packets due to link failure. Or there may be a possibility that the sender is not able to send the packets to the desired node.

As described in the related research, the fault tolerance can be implemented either with the help of check-pointing or with the help of message logging.

3.3.1 Proposed Approach

In our approach, we combine the two concepts for fault tolerant routing. When the data to be forwarded is packetized, it is piggy-backed with a randomly generated number called the transaction ID and the total number of packets to be sent. All the generated packets are assigned a serial number so that the receiver can keep a track of lost messages.

Thus, the packet is shown in fig(2)

Packet Type	File format	Data	Transaction Id	Seq no.	Total no. of packets	Receiver Name

Fig. 2. Packet formed before forwarding the data

A sender log is maintained which records the filename and the receiver's address.

Filename	Receiver's [Bluetooth] address

Fig. 3. Sender Log

On sending a message

1) A file/data is broken into small chunks of data.
2) Each chunk is then packetized as shown in fig (2).
3) An entry is made into sender log as shown in fig (3).
4) The packet is sent to the receiver.

3.3.1.1 Omission Failure

Omission failure[4] occurs when a message inserted in sender hosts message buffer never arrives at the receiver hosts incoming message buffer.

3.3.1.1.1 Fault Detection

For detecting omission failure a receiver log is maintained which stores the transaction ID, the number of packets received, the total number of packets it should receive and the sender's address and the filename.

Transaction ID	Total no. of packets	No.of packets received	Filename	Sender's address	Time at which packet was received

Fig. 4. Reciever Log

Negative Ackowledgement	Transaction Id	Sender Name	Receiver Name

Fig. 5. Negative Acknowledgement

Acknowledgement	Transaction Id	Sender Name

Fig. 6. Acknowledgement

Transaction ID	File Name

Fig. 7. Sender Log

On receiving a packet:

1) If the receiver received a packet with sequence number as 1, A new entry is made in the receiver log as shown in fig (4).
2) On receiving each packet with sequence number greater than 1, based on transaction ID its sequence number is compared with the sequence number of the received packet.
3) If the sequence number is consistent the packet is processed and number of received packets is updated.
4) If the sequence number is not consistent the packet is rejected and a negative acknowledgement (NACK) is created as shown in fig (5).

The receiver log is monitored periodically for incomplete transactions. A transaction is said to be incomplete if its last packet was received 30 sec before the current time. On detecting an incomplete transition a NACK is again generated and sent.

An acknowledgement is sent to the sender, as shown in fig (6), after receiving the last packet. On receiving the acknowledgement entry in sender log is deleted. After sending the acknowledgement the receiver deletes the entry of file from the receiver log.

3.3.1.2 Fault Recovery

On receiving a NACK:

1) Checks the entry in sender log based on transaction Id to get the filename.
2) The file is broken into chunks.
3) Chunks following the sequence number received are packetized.
4) An entry is made into sender log as shown in fig (7).
5) The packet is sent to the receiver.

3.3.2 Crash Fault/Entry of New Nodes

Crash fault occurs when a node due to mobility gets disconnected from the topology. Similarly a new node can also enter the existing topology. All such updations should be diagnosed and communicated.

3.3.2.1 Fault Detection

The receiver checks his receiver log periodically. A node arrival and departure is detected.

3.3.2.2 Fault Recovery

The notification of the arrival and departure of nodes is broadcasted in the topology.

4 Evaluation

Logs are maintained at both sender and receiver end which helps in fault detection and recovery. But these logs can be a load on memory. Hence after complete transfer of packets the entry from the log is deleted. In this way all transactions that are incomplete are stored and rest are deleted to decrease the load on memory.

The most important factor to be dealt with in any routing algorithm is Reliability. The problem faced in most of the applications based on dynamic networks is the problem of failure. Especially for a network type application, failure can occur easily due to change in available bandwidth, failure of links, nodes. Hence the goal here is to make the system work properly under any circumstances either by fixing the problem or to make the system work by neglecting the faults without its performance being affected.

Before the integration of fault tolerance module to the routing algorithm when there is a low link failure rate, as shown in case (a), the packets are dropped in between and when the link sets up again, the dropped packets are not transferred. However, after application of fault tolerance module in the application, the dropped

packets are asked to be sent first and then the latter packets are accepted. Hence, after the integration of fault tolerance, the channels become more reliable.

In case (b), when there is a high link failure rate, more packets are dropped i.e. complete information is not transferred from sender to receiver and large number of packets are dropped. These simulations have been done in ns2.The simulation results for case (a) and case (b) are shown in fig (8) and fig (9) respectively.

a. Low Link Failure Rate

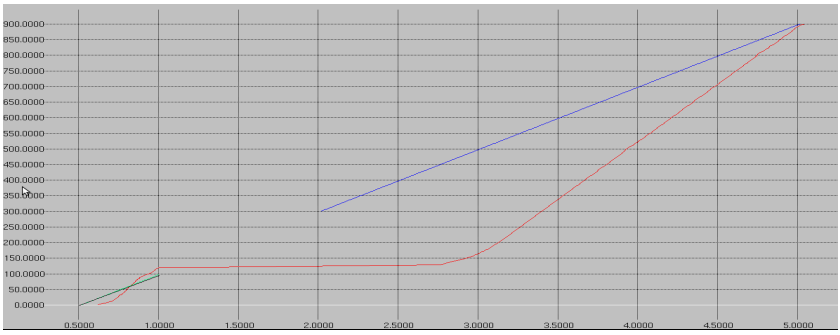

Fig. 8. Case with Low Link failure Rate showing before and after the implementation of the module

b. High Link Failure Rate

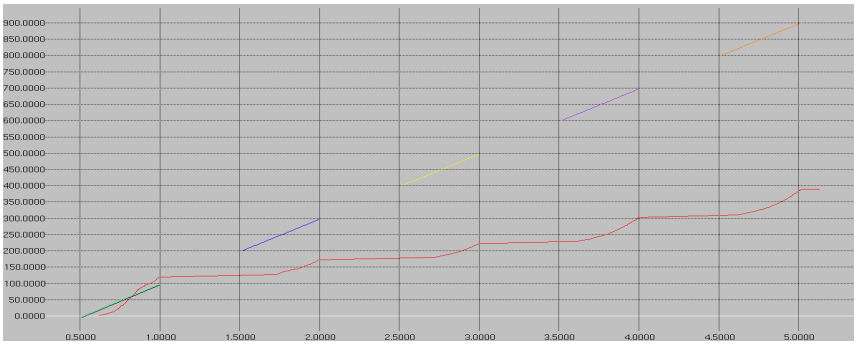

Fig. 9. Case Showing High link failure rate before and after implementation of the module

For any network based application, the transmission time and response time are the most important characteristics.

In this application, for transmission of the data, the data is first packetized in the application and then forwarded. Computation time with respect to this application is defined as the time it takes to form the packet initially. Delay is the time introduced due to the hoping of the data which includes comparing the address with its own address and correspondingly either forward or keep it with itself or delay introduced due to the link failure between nodes. The response time per packet is defined as the

sum of delay introduced and transmission time. And the total response time i.e. the total time taken to transfer the file via hoping or directly is derived by multiplying the number of packets with the response time per packet.

The following table depicts different scenarios with different data input files with respect to total time taken in the network.

Table 1. Response time with respect to Link Failure Rate in different scenarios. n–Number of hops done to reach the receiver.

Link Failure Rate	**No. of hops**	**Computation Time at sender (per packet) (sec:milisec)**	**Transmission Time (per packet) (sec:milisec)**	**Delay (per packet) (sec:milisec)**	**Response Time (Delay + Transmission time + Computation time) (per packet) (sec:milisec)**
0	0	01:22	00:03	N/A	01:22
0	1	01:22	00:03	00:32	01:22 + 00:35 =01:57
0	N	01:22	00:03	00:32*n	01:22+00:32*n +00:03 = 01:25+00:32*n
1	0	01:22	00:03	00:07 + 00:03[#]+ 01:30 = 01:40	01:22+01:40 +00:03 = 03:05
1	1	01:22	00:03	00:07+ 00:03+ 01:30+ 00:03 = 01:43	01:22+01:43 +00:03 = 03:08
1	N	01:22	00:03	00:07 + (01:30*n-1)+ 00:03*n+ 1 =03:33*n –00:93	01:22+00:03 +03:33*n-00:93 = 2:12+3:33*n

[#] Assuming the link breaks after transferring one packet only.

5 Related Research

Faults in any system can be classified in three categories: Transient, Intermittent and Permanent faults. Transient faults are the ones which occur once and then disappear. An intermittent fault occurs, vanishes of its own accord, then reappears, and so on. Intermittent faults cause a great deal of aggravation because they are difficult to diagnose. A permanent fault is one that continues to exist until the faulty component is replaced [2].

Since, in a large network, the machines [servers/clients] are highly dependent on each other, failure in one system can lead to failure in others and so on. Therefore, it is a necessity to define failure models. Gottfried Fuchs has described various failure models in his paper [10].

MNLIR scheme proposed in [6] uses interval routing. Interval routing is a space-efficient routing method for networks, but the method is static and determinative, and it cannot realize fault-tolerance.

A drawback of the topology construction in method of [7] is that it is not particularly efficient for very dynamic environments. A node that joins or leaves the network could trigger a complete restructuring of the topology. Although algorithm proposed in [8] generates topologies with low total weight, the question of their sub-optimality with respect to the unique minimal k-connected and k-regular overlay graph arises. M.K, Das, et al. [11] has given a comparative study of various routing algorithms along with their analysis. However, they have not focused on fault tolerant routing. A new routing approach named DRS has been described in [12]. This protocol basically looks for faults from time to time in the system.

6 Conclusion

The goal of this paper is to propose a fault tolerant routing algorithm which deals with two types of failure models, crash fault and omission fault. The focus mainly has been on network communication layer. Our approach periodically monitors the network to form the most updated topology with the nodes. It also checks for alternate routes if the gateway is disconnected from the sender node. The application uses minimum memory for the application of fault tolerance. This fault tolerance approach brings redundancy but it is a reasonable trade off for providing reliability.

References

1. Cao, G.: Designing Efficient Fault-Tolerant Systems on Wireless Networks. In: The Proceedings of 8th Annual International Conference on Computer Networks, pp. 263–270 (2004)
2. Tanenbaum, A.S., Van Steen, M.: Distributed Systems, 2nd edn. Pearson (2006)
3. Badera, P., Bhardwaj, A., Rajalakshmi, K.: Range grouping for routing in dynamic networks. In: Parashar, M., Kaushik, D., Rana, O.F., Samtaney, R., Yang, Y., Zomaya, A. (eds.) IC3 2012. CCIS, vol. 306, pp. 95–105. Springer, Heidelberg (2012)

4. Dolev, D., Friedmant, R., Keidar, I., Malkhi, D.: Failure detectors in omission failure environments. In: PODC 1997 Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, p. 286 (1997)
5. Koushanfar, F., Potkonjak, M., Sangiovanni-Vincentelli, A.: Fault Tolerance in Wireless Sensor Networks. In: Mahgoub, I., Ilyas, M. (eds.) Handbook of Sensor Networks, Section VIII, vol. 36. CRC press (2004)
6. Feng, X., Han, C.: A Fault-Tolerant Routing Scheme in Dynamic Networks. Journal of Computer Science and Technology 16(4), 371–380 (2001)
7. Thallner, B.: Fault tolerant communication topologies for wireless ad hoc networks. In: 1st Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS 2004), Florence, Italy (June 2004)
8. Thallner, B., Moser, H.: Topology Control for Fault-Tolerant Communication in Highly Dynamic Wireless Networks. In: Third International Workshop on Intelligent Solutions in Embedded Systems, pp. 89–100 (2005)
9. Thallner, B., Moser, H., Schmid, U.: Topology control for fault-tolerant communication in wireless ad hoc networks. Published in: Journal Wireless Networks 16(2), 387–404 (2010)
10. Fuchs, G.: Implications of VLSI Fault Models and Distributed Systems Failure Models – a Hardware Designer's view. In: The Proceedings of the Conference on Design, Automation and Test, vol. 2, pp. 300–310 (March 2009)
11. Marina, M.K., Das, S.R.: On-demand Multipath Distance Vector Routing in Ad Hoc Networks. In: 9th International Conference on Network Protocols, vol. 4(6), pp. 14–23 (2001)
12. Chowdhury, A., Friederi, O., Burger, E., Grossman, D., Makki, K.: Dynamic Routing System (DRS): Fault Tolerance in Network routing. In: The Proceedings of 10th International Conference on Computer Networks, vol. 3(8), pp. 23–25 (2004)