Global Journal of Computer Science and Technology
Volume 12 Issue 6 Version 1.0 March 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

A Quick Termination Detection Protocol by Reducing Overload for Mobile Ad Hoc Networks

By Subrata Kumar Das, Md. Asif Nashiry & Md. Alam Hossain
Jessore Science & Technology University, Jessore, Bangladesh

Abstract - An ad hoc network is characterized by the fact that there is no fixed topology due to the mobility of nodes, interference, multipath propagation and path loss. Execution of applications in such kind of networks typically consists of a number of successive phases such as network reprogramming, localization, power monitoring, and parameter updates. Termination detection of a phase is therefore a critical operation to safely execute a new phase on some or all of the network nodes. In resource constrained network environment the overhead should be minimum in order to increase throughput and minimize delay. This paper studies the existing solutions for termination detection by analyzing their effectiveness. Moreover, in this paper, we propose an efficient algorithmic solution to encounter termination detection by minimizing the network overloads.

Keywords : *Ad hoc networks, termination detection, network overloads, diffusion-based approach, distributed system.*

GJCST Classification: *C.2.2*

Strictly as per the compliance and regulations of:

A Quick Termination Detection Protocol by Reducing Overload for Mobile Ad Hoc Networks

Subrata Kumar Das [α], Md. Asif Nashiry [σ] & Md. Alam Hossain [ρ]

Abstract - An ad hoc network is characterized by the fact that there is no fixed topology due to the mobility of nodes, interference, multipath propagation and path loss. Execution of applications in such kind of networks typically consists of a number of successive phases such as network reprogramming, localization, power monitoring, and parameter updates. Termination detection of a phase is therefore a critical operation to safely execute a new phase on some or all of the network nodes. In resource constrained network environment the overhead should be minimum in order to increase throughput and minimize delay. This paper studies the existing solutions for termination detection by analyzing their effectiveness. Moreover, in this paper, we propose an efficient algorithmic solution to encounter termination detection by minimizing the network overloads.

Keywords : Ad hoc networks, termination detection, network overloads, diffusion-based approach, distributed system.

I. Introduction

Ad hoc networks are formed opportunistically among devices equipped with wireless communication capabilities. Links are established as devices move within communication range and are broken as hosts move away. All of this takes place without assistance from any wired resources.

Numerous kinds of control information need to be exchanged among the nodes of the networks to cope with this dynamic nature of ad hoc environment such as event detection, tracking, routing, power monitoring, parameters up gradation, location detection and so on. The total processing and memory requirements to complete these tasks exceed the resources available on most nodes. Multi-phase execution is therefore a common approach in such complex network situation. In multi-phase execution, the application is divided into multiple logical tasks that are executed at different times. For instance, to determine absolute or relative positions of nodes can be one application phase. Based on this information, the re-programming of location aware application programs onto all nodes can be a subsequent application phase. Power monitoring and parameter updates can be regarded as separate phases. Hence the importance of termination detection across a diverse range of mobile applications is very crucial.

Minimizing these sorts of control messages is very important when dealing with ad hoc networks. Because nodes participating in ad hoc networks require need to store their battery power for further processing and longer stay in the network. So reducing overhead packets not only improve the throughput but also enhance network performance. So exchanging minimum number of control messages is required in resource constrained dynamic ad hoc networking environment.

There are many proposed algorithm for termination detection of distributed system considering on static system i.e. for systems comprising of a fixed set of nodes. There is relatively less work on dynamic systems, where nodes may be created as well as destroyed while the computation is in progress. As mobile ad hoc network is a dynamic process we have to develop a termination detection algorithm for dynamic systems permitting unrestricted connection and disconnection of nodes. A distributed computation is assumed to be structured in the form of a set of concurrent devices/ nodes {Ni}, with each node performing a specific computational task. Nodes can be connected or disconnected during the course of the computation. In according to (Dhamdhere, Reddy & Iyer, 1992) a node becomes idle on completing the computational task assigned to it, and awaits one of the following events:

i. assignment of a new computational task,
ii. receipt of a report killing it, or
iii. declaration of termination.

In case (i), the node becomes active again,
In case (ii), it informs other nodes of its destruction and dies, while
In case (iii), it simply terminates.

The inter devices/nodes communication is assumed to be asynchronous with arbitrary. This ensures that all existing nodes participate in the termination detection.

The solution of termination detection problem can be classified into the following two categories:

Author α : Department of Computer Science & Engineering, Jessore Science & Technology University, Jessore, Bangladesh.
E-mail : sdas_ce@yahoo.com
Author σ : Department of Computer Science & Engineering, Jessore Science & Technology University, Jessore, Bangladesh.
E-mail : asif.nashiry@yahoo.com
Author ρ : Department of Computer Science & Engineering, Jessore Science & Technology University, Jessore, Bangladesh.
E-mail : alamcse_iu @yahoo.com

weight throwing protocol (Mattern, 1989), (Sameeruddin, Sharma, Nandi & Dutta, 2007) & (Mittal, Venkatesan & Peri, 2007) and diffusion based protocol (Dijkstra & Scholten, 1980) & (Misra & Chandy, 1982). In the former one, a real number, called weight, is associated with every node involved in computation. The later one is a tree based scheme in which the root node initiates and coordinates the computation. Other nodes are added in the tree as and when they get their job. Weight throwing protocol has constant time, space and communication complexities and it is more efficient and scalable for the distribute system having mobility. But having least computation complexity the diffusion based protocol best suits the diverse nature of resource constrained mobile ad hoc networks. This paper is concerned with the development of a termination detection algorithm based on diffusion based protocol for use in ad hoc networking environments that reduces the overload of the network by exchanging lowest number of control packets among the nodes in the network.

The reminder of this paper is organized as follows: section 2 depicts the related works, section 3 introduces the major problem of diffusing computation and its solution, section 4 presents our proposed algorithm and its functionalities. In section 5, we mention an example to provide a more concrete understanding of the algorithm. Conclusion and future works are in section 6.

II. Related Works

Mobile ad hoc network is an emerging technology. Several researches in different field relating this issue have been conducted. As termination detection is one of the fundamental concerns in ad hoc networking, a number of researches (Baker & Ephremides, 1981), (Sato, Inoue, Masuzawa & Fujiwara, 1996) & (Chen & Murphy, 2001) have been done in the past to analyze and improve the performance of termination detection protocols. The algorithms of Dhamdhere, Reddy & Iyer(1992), Misra & Chandy(1982) & Cohen & Lehmann(1982) are concerned with special cases of dynamic systems, viz. systems with synchronous communication in which nodes may be created but not destroyed. Lai (1986) gives an algorithm for dynamic systems where nodes may be created and destroyed. In (Tseng &Tan, 2001), the authors used a hybrid protocol for termination detection in mobile distributed environment by combining weight throwing and diffusion based approaches. This protocol works by applying weight throwing approach to all static processes and diffusion based scheme to all mobile processes.

Roman and Payton (2005) use a diffusion based scheme to detect termination in mobile ad hoc networks. This protocol requires that each node maintains a list of nodes that it activates and will be responsible to take further information till it becomes passive. The basic principle of this protocol is to use of a partial ordering of across all active nodes in the network for delivery of termination notices. This method is reliable indeed. But we have identified some sort of problems in the research works done in the past based on diffusion based approach of termination detection. We address and present these problems in section 3. The possible solutions of those problems also follow in this section. Thus our work complements previous works and can be combined to help in resource constrained ad hoc networking environment for termination detection.

III. Problems In Diffusion Based Computation & Its Solutions

Most of the research works done earlier based on diffusion computation depend on the relationship between root and all other nodes in the network. In those algorithms, to detect termination, root is explicitly and implicitly depend on all other nodes in the network and responsible for collecting all status information of other nodes to process and memorize these information. In this way the task of root becomes complex to execute. In this circumstance the problems that may create and their solutions are defined below.

Problem: Receiving an idle report from a node, the root node cannot prune that node due to having lack of knowledge about its parent.

According to (Roman & Payton, 2005), after getting the idle report from some nodes the root cannot prune those nodes until it has not got the status of the parent of those nodes. This increases the overload of the network and to memorize the receiving idle report from the nodes until it has collected the information from all nodes of the network. If it is possible to prune the node after getting the idle report the overload of the network is decreased. For instance, consider the following figure 1.

In figure 1, at any instance of time, node E moves into the communication range of node D and passes its idle report. Then node D becomes idle and adds its information to the idle report, the node D eventually establish communication with node A and passes its idle report. In this situation, root A knows that E and D are idle. It also knows that E has no children and node D is the parent of nodes F and G. But root A does not know that B is the parent of nodes E and D. Also, root node A does not know the status of nodes B, F and G.

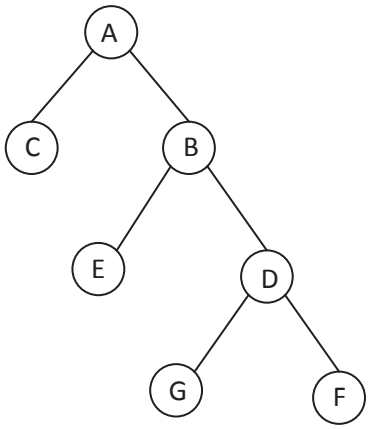

Fig.1: Networks among nodes

In this case, according to our observation, though the root A has already got the idle reports from nodes E and D, it cannot prune the node E, because root A has not yet received the status of their parent node B. So, node A have to keep the node E, which is not necessary at this stage, due to the absent of status of node B.

Solution approach: To solve the problem mentioned above we propose a decentralized activity. Now, the root nodes only maintain the information of its children only. These children while behaving as parent will maintain the status of their children only and so on. So, in figure 1, node D maintain the status of nodes F and G, node B keeps track the status of E and D and at last root A knows the status of its children B and C. In this way, the root node A can know all idle reports and can easily detect the termination. By following this process, root A has not to maintain the status of all other nodes rather it maintain only the status of its children. For instant, if parent nodes had to maintain the status of its own children then after getting the idle report from node E (figure 1), the parent node B can prune its child E because it has no children at all. The advantage here is that it is not necessary to keep a node for long time, which reduces network overload, minimize the computation complexity of the root and requires small buffer to process; resulting performance enhancement.

IV. Algorithm and its Functionalities

Our proposed algorithm belongs to the category of diffusion-based algorithm. However, to overcome the drawbacks of the diffusion-based algorithm mentioned in section 3, we have designed the decentralized computation instead of centralized one.

In this system, each wireless device is considered as node except a node which is called the root initiate the computation. Other nodes are categorized two types: parent (defined as the node which has child/leaf) and child (defined as the node without any child that means leaf). In previous centralized system, the initiator which is considered as the root performed all the tasks to detect and declare termination. But in our designed system the others nodes are given the responsibility to keep the status of its own children. At first, the root will distribute the task among its children. Then the children of the root while behaving as parents will distribute the task to its children and so on. On the other hand, the idle reports are collected in the same way, but in reverse order.

Here each node maintains a data structure to store identities of its connecting nodes. Whenever two nodes Ni and Nj communicate with one another, their status are updated with each other's id's. When a process Ni kills process Nj, or Nj kill itself, Nj informs its parent. This leads to the deletion of Nj from the network. New nodes may be added to parent (Ni) without the concern of the root which will not make any hassle to detect termination.

An important issue is that every active node is reachable along a path from the root, but many of the links may no longer be up since nodes may have moved out of range with respect to the root. It may be that a node is the out of range of the root, but it is under the range of another parent node. Then the root cannot control that node or the node cannot send its status directly to the root. In this case, the parent of our proposed protocol can properly access such nodes that are the outside of the range of the root but within the range of the parent and help the root to detect termination quickly. So we have divided the functionalities of our protocol in three sections. Firstly, the execution will be started from the root which checks whether there are children or not. If the root does not find any of its children then it declares the system as terminated. If the children (nodes) are presented then the root executes the lines from 2 to 7. A child finishing its work sends the idle report to its root, then the root merge that report with its previous ones (line 3) and the child become idle and leave the network as it has not connected to any other node.

If any node is connected to the report sending child, mentioned above, will respond as a parent. Working as a parent, the node completes the tasks in sections (b) and (c) in the algorithm, then it sends the report to the root and only after that the root can declare the termination of the system. But if there is no node joined with the child, then it will be idle and disconnected. In our proposed decentralized protocol the parent plays an important role getting the idle report from its child. At first update the status of its idle report following prune the child (if it is a leaf) from the network then check whether any others nodes are exist or not. If there exist any other child then the precious process will repeat. Otherwise it will response as a child. In third section, the child sends the idle report to its parent as a leaf becomes idle and leaves the network.

State characterization for node

root	- the node is the initiator of the diffusing computation
parent	- the node having child of the diffusing computation
child	- the node connected with other nodes.
idle	- the node is in an idle state, initially true except of the initiator of the diffusing computation
idleReport	- the status of a node initially zero.
PRESENT	- a node is present in the network and active
leave	- the node prune from the parent or the network.
leaf	- a node has no child.

a) Actions as the *root*

1. If no *child* are connected to the *root* Then "system terminated"
2. else on receipt of an *idleReport* from the *child*
3. merge *idleReport*
4. if the *child* is a *leaf*
5. *child* ← *idle and leave* ;
6. else the *child* act as a *parent*
7. end if
8. check whether any other *child* is *PRESENT*
9. if *child PRESENT* repeat steps 2 to 8
10. else "system terminated"
11. end if
12. end if

b) Actions as a *parent*

13. on receipt of an *idleReport* from the *child*
14. merge *idleReport*
15. if the *child* is a *leaf*
16. *child* ← *idle* & *leave* ;
17. else the *child* act as a *parent*
18. end if
19. check whether any other *child* is *PRESENT*
20. if *child PRESENT* repeat steps 13 to 19
21. else *parent* acts as a *child*
22. end if

c) Actions as a *child*

23. Send *idleReport* to its *parent*, become idle and leave

V. Example

In this section, we present an example to explain the fundamental idea behind our proposed algorithm to detect the termination in the network.

Consider a network consists of ten hosts/nodes, shown in figure 2. These hosts do some tasks using some particular software installed on them. Suppose at any time, there is a need to update on of the software installed on all of the hosts by replacing its old version. Before processing any further task each and every host must be informed about that the up gradation of software reflects to all other nodes in that network.

According to our algorithm, the root node initiates the software up gradation process. With the reference of figure 2, at first, root node install the new version of software on it and sends it to its neighbours that lie within the communication range of the root; A, B, C, D and E, only.

After receiving this information node D install the new version on it and transmit this message to its neighbours, F and G. Similarly, node E first installs the software and then sends it to node H. At any instant of time if node I comes within the communication range of node E then it will be the child of node E. Then node E sends the up gradation request to node I.

The nodes that are out of range from root node such as F, G, H and I, send the confirmation report to their corresponding parent. The parent nodes D and E then send this to root node. In the mean time, root has the confirmation report of all nodes that lie within its communication range. So that, root has the confirmation reports of all nodes in the network. It can terminate the process of up gradation and initiates further tasks. Here each parent node is responsible to send and receive the information to/from its immediate children only. It is not worried about the rest. In this case, the task is totally distributed.

According to our example the problems mentioned in section 3 can easily be solved. Because, when child nodes F and G send the confirmation report to their parent node D then node D can prune those nodes, F and G from the network (solution of problem 1). Moreover, root node does not need to memorize the

report of the nodes that lie outside its communication range. It only needs to keep track the reports of its neighbours, which reduces the computation complexity (solution of problem 2).

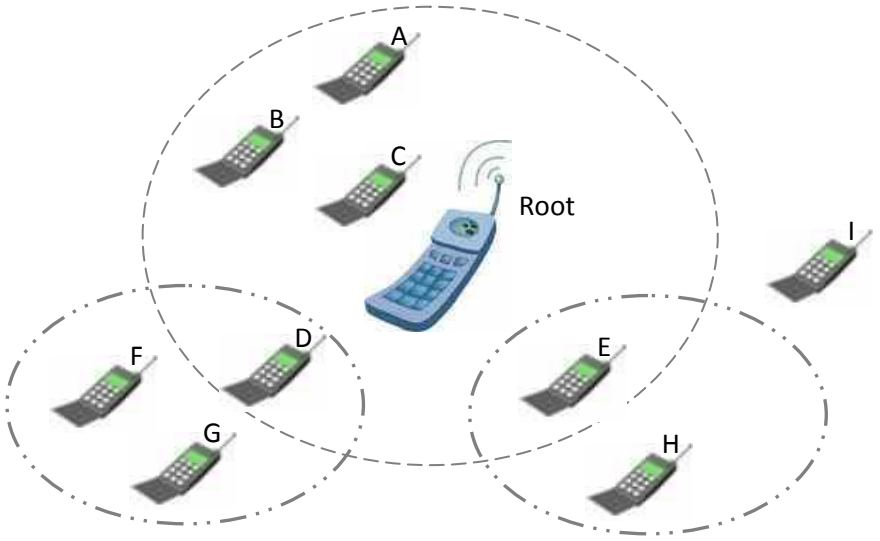

Fig.2 Mobile Networks

If a particular node is out of range from the root node then it can propagate its report via the intermediate nodes using ad hoc relay. Moreover, if a new node comes to the communication range of any other existing node then the former one will be the child of the later one. The new node will receives and/or send information/status to its parent only. In this way the root node can know the status of all nodes on the network directly or via other nodes.

VI. Conclusion and Future Works

In this paper, we have investigated and proposed a termination detection algorithm in mobile ad hoc networks. We have underscored some problems in this regard and provided solutions. As out of range communication between two hosts and link failure due to mobility are the fundamental concerns in ad hoc network, so relying on one node to collect status of all other hosts cannot be an effective formula. In our solution, every host exchange activation and idle information to its own child nodes only. It will send then to the initiator node by direct communication or relay. It minimizes delay, reduces complexity and enhances performance. More research is needed to detect termination in ad hoc networking environment. We plan to investigate the distribution of activation message as well as collecting idle report for multiple sources in future.

References Références Referencias

1. Dhamdhere, D. M., Reddy, E. K. K. & Iyer, S. R. (1992). Distributed termination detection for dynamic systems. IIT Bombay, TR-081-92.
2. Mattern, F. (1989). Global quiescence detection based on credit distribution and recovery. IPL, Elsevier, 195-200.
3. Dijkstra, E. & C. Scholten, C. (1980). Termination Detection for Diffusing Computations. IPL, Elsevier, 1–4.
4. Baker, D. & Ephremides, A. (1981). A Distributed Algorithm for Organizing Mobile Radio Telecommunication Networks. In Proceedings of the Second International Conference on Distributed Computer Systems, 476–483.
5. Misra, J. & Chandy, K. M. (1982). Termination detection of diffusing computations in communicating sequential processes. ACM Transactions on Programming Languages and Systems, 4(1), 37-43.
6. Sato, Y., Inoue, M., Masuzawa, T. & Fujiwara. H. (1996). A Snapshot algorithm for distributed mobile systems. In Proceedings of the 16th International Conference on Distributed Computing Systems, 734–743.
7. Chen, X. & Murphy, A. (2001). Enabling Disconnected Transitive Communication in Mobile Ad Hoc Networks. In Proceedings of the Workshop on Principles of Mobile Computing, 21–27.
8. Cohen, S. and Lehmann, D. (1982). Dynamic systems and their distributed termination. Proceedings of the first Annual ACM Symp. on Principles of distributed computing, Ottawa, 29-33.
9. T. H. Lai, T. H. (1986). Termination detection for dynamic distributed systems with non-first-in-first-out communication. Journal of Parallel and Distributed computing, 3, 577-599.

10. Tseng, Y. & Tan, C. (2001) Termination Detection Protocols for Mobile Distributed Systems. IEEE Transactions on Parallel and Distributed Systems 12, 558–566.
11. Roman, G. C. & Payton, J. (2005). A termination detection protocol for use in mobile ad hoc networks. ASE, Springer, 81-99.
12. [12]S. De, M. Sameeruddin, S. De. M., Sharma, V., Nandi, N. & Dutta, H. (2007). A New Termination Detection Protocol for Mobile Distributed Systems. Proc. ICIT 07, IEEE, 148-150.
13. Mittal, N., Venkatesan, S. & Peri, S. (2007). A Family of Optimal Termination Detection Algorithms. DC, Springer, 141-162.