ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 8(34), DOI: 10.17485/ijst/2015/v8i34/85284, December 2015* ISSN (Online) : 0974-5645

A Message Efficient Group Membership Protocol in Mobile Ad Hoc Distributed Systems

Sung Hoon Park*

School of Electrical and Computer Engineering, Chungbuk National University, Korea;
spark@chungbuk.ac.kr

Abstract

In distributed systems, a group of computer should continue to do cooperation in order to finish some jobs. In such a system, a group membership protocol is especially practical and important elements to provide processes in a group with a consistent common knowledge of the membership of the group. Whenever a membership change occurs, processes should agree on which of them should do to accomplish an unfinished job or begins a new job. The problem of knowing a stable membership view is very same with the one of agreeing common predicate in a distributed system such as the consensus problem. Based on the stopping investigating protocol that is traditional one in asynchronous distributed systems, we present the new group membership protocol in mobile wireless networks.

Keywords: Fault Tolerance, Group membership, Mobile Ad Hoc Environment, Synchronous Distributed Systems

1. Introduction

In distributed systems, a group of computer should continue to do cooperation in order to finish some jobs. A group membership protocol is especially helpful tools to allocate processes in a same group with a same view of the membership of the group. Whenever a membership change occurs, processes can consent to which of them should do to finish a waiting job or begin a new job. The problem of getting a stable membership view is very same with the one of getting common knowledge in a synchronous distributed system such as the consensus problem[1].

The Group membership protocol[2] is that every process connected in a network requires getting a stable same group membership view if all connected process are belong to just one group. The problem was widely simulated at the study community. The reason for this great simulation is that many distributed systems need a group membership protocol[3-7]. In spite of such practically usefulness, to our knowledge there is only a few labor that have been committed to this problem in a mobile ad hoc computing environment.

Depending on process mobility, network topologies are changed and process may dynamically connect and disconnect over a wireless network. In such wireless networks, group membership can be changed so much, making it a special critical module of system software part. In mobile ad hoc systems, a lot of environmental adversities are more common than the wired network systems such as that can cause loss of messages or data[8]. In particular, a mobile process can easily get to fault by battery problem or disconnect from the wireless network. Implementing fault-tolerant distributed mobile applications in such an environment is a complex and difficult behavior[9,10].

In this paper, we propose a new protocol to the group membership protocol in a specific ad hoc mobile computing system. Based on the stopping investigating protocol that is traditional one in asynchronous distributed systems, we address the new group membership protocol. We make up of the rest of this paper as follows. In Section 2 we address the mobile system model we use. In Section 3, we describe a solution to the group membership problem in a traditional asynchronous distributed system. We also address a new protocol to solve the group

*Author for correspondence

membership problem in a mobile ad hoc computing system in Section 4. In Section 5, we address conclude.

2. Computing System Model, Definition and Assumption

In this section, we describe our models for capturing behavior of distributed systems. We use these models foe reasoning about correctness of our protocol as well as for analysis of distributed computations. Our model for distributed systems is based on notice passing, and all of protocol is around that concept. Many of these kinds of protocol have analogs in the shared memory world but will not be addressed in this paper.

First, we define our system model based on some assumptions and after that we address our goals. We model a distributed system as a loosely coupled messing-passing system without shared memory and a global clock. Our distributed computation model for an ad hoc network is made up of as an undirected graph. That is, the undirected graph is described as $G = (V, E)$, in which vertices V facing each other with set of mobile process $\{1, 2,\ldots,n\}$ ($n >1$) with unique identifiers and edges E between a pair of process correspond the fact that the two process are in each other's transmission radii. Hence, our distributed system a channel to directly communicate with each other which changes over time when processes move.

Every process i has a variable N_i, which denotes the neighboring processes, with that i can *directly* communicate the neighboring processes. Every process communicates with a channel that is bidirectional; $j \in N_i$ iff $i \in N_j$. More accurately, in the network $G = (V, E)$, we decide E such that for all $i \in V$, $(i, j) \in E$ if and only if $i \in N_j$. Depending on process's movement, the graph could be disconnected that means that the network is partitioned. Because the processes may alternate their position, N_i position would be unexpectedly changed and therefore G also may be changed accordingly. The assumptions about the processes, wireless network and system architecture are followings.

Every process is distinguished by a unique identifier. The unique identifiers are used to distinguish processes during operating the group membership search process. Channels and links are bidirectional that means first n first out, i.e. every process receives notices based on the sequence that are delivered over a link between two neighboring processes. Many topology changes may be arbitrary occurred when the process moves in wireless networks. That makes a lot of network partitioning and merging. Processes can make a fault to be crash arbitrarily at random and can recover again at any time.

Without network partition, the sender and the receiver do successful notice delivery that means the notice would be successfully delivered only when the two processes remain connected for the all period of notice transfer. Every process has a big receiving buffer enough to avoid buffer overflow all the time in its lifetime. Even though a finite number of topology changes, every process i *eventually* has a same view of group membership of the group to which i belongs.

3. Group Membership Specification

We assume that our specification is as followings, it is consist of four properties for a group membership protocol.

Safety(1) : At any time, all processes in the group have a stable consistent view.

Progress : If there are no more changes in the each views of the processes in one group, they eventually getting to their stable consistent views.

Validity : If all processes in a event know a view as their local view and they have eventually reached their stable states, then the last process of their sequences of global views are all at same position and must be equal to each other.

Safety(2) : When a view is committed as a global view, it cannot be changed.

The first property describes agreement. Consistent history must be an unchanged one for any program that satisfies the specification. The second property shows termination of global view. When the state and event of all processes are unchanged, the processes are eventually getting to close changing their output results. The third property removes trivial solutions where protocols never getting on any new view or always determine on the consistent view.

4. Group Membership Protocol in Ad Hoc Network

At this section, we address a group membership protocol that was operated upon the stopping investigating protocol, simply SIA, by scattering computations. After these

sections, we will describe in detail the method that this protocol may be accommodated to a mobile system.

4.1 Group Membership in a Wired Network

We first address our group membership protocol in the wired network settings. In which we assume that process and channels have no faults.

The protocol is made up of three phases running at the process that starts the group membership protocol.

1. The first phase that is a diffusing phase and it works by first diffusing the "who" notices.
2. The second phase that is a searching phase and it runs by then accumulating the id of every process that is consist of the wired networks. We represent this computation starting processes as the *start process.*
3. The third phase is a closing phase that is managed by deciding the same view and announcing it as a stable new view to all process.

The start process will have the information enough to decide a uniform group membership view after taking all process' ids completely and the start process will then broadcast it to the rest of the process in the network. The three kinds of notice, *Who*, *Ack* and *View* are used to manipulate the operations.

As the first phase is diffusing computing phase, *Who* notice is used to make a start of the group membership protocol by diffusing the *Who notice*.

4.1.1. The First Phase

When group membership protocol is launched at a start process *s*, the start process makes a replying queue *wl* and a accepted queue *rl* and starts a *scattering computation* by forwarding an *Who notice* to all of its immediate neighboring processes. At the starting point, the replying queue makes up of only its most close neighboring process's ids and the accepted queue has nothing.

When process *i* receives a *Who notice* from the neighboring process for the first time, it immediately sends the *Ack* notice to the start process and propagates the *Who notice* to all its neighboring process except the process from which it first accepted an *Who notice*.

The *Ack* notice sent by process *i* to the start process contains the ids of all its neighboring process that are needed for the start process to decide the stable view of the process connected with a distributed network. After that, any *Who notice* accepted by other neighboring process will be ignored.

4.1.2 The Second Phase

Searching phase- When the start process receives the *Ack* notice was taken out from the process *j*, it takes *j* out from the replying queue and gets *j* into the accepted queue and as soon as possible it detects sequentially the each process's id included in the *Ack* notice. If there is the some process in the *Ack* notice which has already been accepted, i.e. that means it is in the accepted queue, it is dismissed. If it is not in the accepted queue, it is inserted into the replying queue of start process. The start process will be suspends for the *Ack* notice from one.

The replying queue is increasing and decreasing repeatedly when it was accepted based on the accepted *Ack* notices, however the replying queue is continually increasing by accepting the *Ack* notices. But the replying queue at the end could have no element and the replying queue could insert all ids of processes connected to the wired networks whenever the start process accepted the *Ack* notices from all other processes. Therefore the start process eventually has much information enough to decide the stable view of the group based on the replying queue. That is because the replying queue could be eventually unoccupied and it means that the start process has accepted the *Ack* notices from all the process.

4.1.3 The Third Phase

Once the start process has accepted *Ack*s from all other process, it decides the stable view based on the replying queue and forwards a *View* notice to all other process to let know the current view of the group. We show some sample running protocol as the protocol execution to explain more specific features. We address the protocol in synchronous setting even though all the behaviors of the protocol are practically asynchronous. We assume that the network shown in Figure 1(a) is asynchronous. In this shape, and for the all of the paper, thin arrows denote the route of *Who notice*'s move and dotted arrows denotes the way of route of *Ack* notices to the start process.

As shown in Figure 1, process A is a start process that starts wl_a and rl_b with {B,C} and {A} at each and starts a scattering computation with forwarding out *Who notice*s (indicated as "E" in the shape) to its immediate neighbors, viz. process B and C, shown in Figure 1(a). As indicated in Figure 1(b), process *B* and *C* in turn forward the *Who*

notice to its most close neighbors only except the start process. It sends the *Ack* notice with close neighboring process queue to the start process *A*. Hence *B* and *C* also send *Who notice*s to each other.

But B and C do not acknowledge to the start process about the *Who notice*s because process *B* and *C* have already accepted *Who notice*s from the start process at each. The information of neighboring process is piggybacked upon the *Ack* notice sent by all process.

Upon hearing *Ack* notices from B and C, process A renews $wl_a = \{ B,C \}$, $rl_b = \{ A \}$ with the close neighboring process information piggybacked at the *Ack* notices.

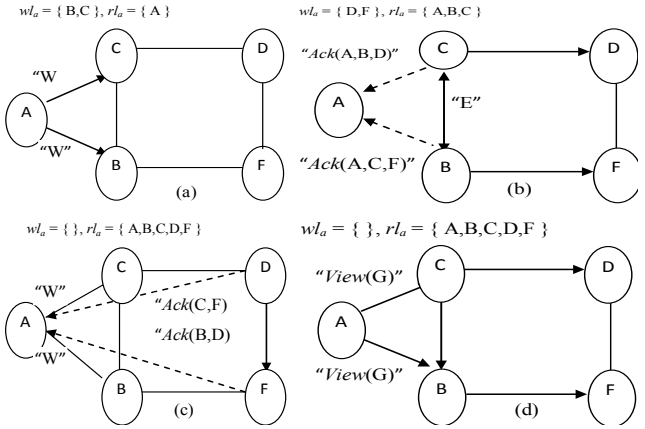

Figure 1. An example of group membership protocol execution on the process search protocol.

The *Who notice*s is transmitted over the arrows at the edges and the dotted arrows going parallel with the edges denotes *Ack* notices. In Figure 1(c), the process D and F also send the *Ack* notices to the starts process at the time they accepted the *Who notice* s from the B and C one by one.

Each of these *Ack* notices includes the ids of the neighbor. All the time, the start *A* accepts all acknowledgments from all of other process except itself in Figure 1(d) and then determines the stable view between the group and forwards it, that is the *View* notice displayed in Figure 1(d).

5. Conclusion

We have addressed here the study of distributed group membership protocol for mobile, ad hoc networks and proved it to be correct based on the symbolic dynamics of finite state machine obtained by linear probability model. We have also shown that the symbolic formal specification of property in our group membership protocol based on linear temporal logic is to reason the protocol correctness.

In real world, the wireless network topology is actively and lively changing at random and that dynamic network changed configuration causes frequent connection and disconnection of process over the wired network. In spite of weakness about wireless networks, our group membership protocol specification guarantees the safety and progress property could be always satisfied.

As mentioned in the introduction, our main goal has been to design group membership search protocol and prove decidability of consistent view in as simple a fashion as possible without paying much attention at wireless networks to the consistent membership view on every process even though complexity issues of ad-hoc networks.

We are however convinced that process search protocol and linear set theory techniques can considerably relax the strong property of safety in many of our constructions. In particular, a more careful logical design of group membership protocol for specific classes of ad-hoc networks can be performed using the results from our design of convergence properties of consistent group membership view. This could lead to a significant improvement of our protocol from a practical environment point.

Finally, in a practical setting one may generalize the group membership protocol to more fit in some distributes systems according to network environments. It will be interesting to explore whether safety or progress could be weakened depending on distributed computing environmental factors.

6. References

1. Amir Y, Moser E, Melliar-Smith P, Agarwal D, Ciarfella P. The totem single-ring ordering and membership protocol. ACM Trans. Computer Systems. 1995; 13(4): 311–42.
2. Anceaume E, Charron-Bost B, Minet P, Toueg S. On the formal specification of group membership services. Technical Report Computer Science Dept. Cornell Univ. 1995; 95(3):1534–59.
3. Anker T, Chockler G, Dolev D, and Keidar I. Scalable group membership services for novel applications. Proceedings of Workshop on Networks in Distributed Computing (DIMACS 45); 1998 Aug 21-23. p. 23–42.
4. Keidar I, Sussman J, Marzullo K, Dolev D. A client server oriented protocol for virtually synchronous group membership in WANs. Proceedings of 20th Int'l Conf. Distributed Computing Systems. 2000 Apr 15-17; 3(1):234–44.

5. Brunekreef J, Katoen J, Koymans R, Mauw S. Design and analysis of dynamic leader group membership protocols in broadcast networks. Distributed Computing. 1996; 9(4):157–71.
6. Bottazi D, Montanari R, Rossi G. A self-organizing group management middleware for mobile ad-hoc networks. Computer Communications. 2008; 31(13):3040-304.
7. David P. Special section on group communication. Communications of the ACM. 1996; 39(4):50–97.
8. Pradhan D, Krishna P, Vaidya N. Recoverable mobile environments: design and tradeoff analysis. Proceedings of Annual Symposium on Fault Tolerant Computing; 1996 Oct 12-15. p. 16–25.
9. Briesemeister L, Hommel G. Localized group membership service for ad hoc networks. Proceedings of International Conference on IEEE Parallel Processing Workshops; 2002 Mar 05-18. p. 94–100.
10. Hatzis K, Pentaris G, Spirakis P, Tampakas V, Tan R. Fundamental control protocols in mobile networks. Proceedings of 11th ACM SPAA; 1999 Oct 19-21. p. 251–60.