The Mutual Exclusion Problem in Cellular Wireless Networks

Young-Whan Cho, Sung-Hoon Park, and Seoun-Hyung Lee
Dept. of Computer Engineering
Chungbuk National Univ., Chungbuk, Korea
{yhcho,spark}@cbnu.ac.kr

Abstract

The mutual exclusion (MX) paradigm can be used as a building block in many practical problems such as group communication, atomic commitment and replicated data management where the exclusive use of an object might be useful. The problem has been widely studied in the research community since one reason for this wide interest is that many distributed protocols need a mutual exclusion protocol. However, despite its usefulness, to our knowledge there is no work that has been devoted to this problem in a mobile computing environment. In this paper, we describe a solution to the mutual exclusion problem from mobile computing systems. This solution is based on the token-based mutual exclusion algorithm.


1. Introduction

The wide use of small portable computers and the advances in wireless networking technologies have made mobile computing today a reality. There are different types of wireless media: cellar (analog and digital phones), wireless LAN, and unused portions of FM radio or satellite services. A mobile host can interact with the three different types of wireless networks at different point of time. Mobile systems are more often subject to environmental adversities which can cause loss of messages or data [8]. In particular, a mobile host can fail or disconnect from the rest of the network. Designing fault-tolerant distributed applications in such an environment is a complex endeavor.

In recent years, several paradigms have been identified to simplify the design of fault-tolerant distributed applications in a conventional static system. Mutual exclusion, simply MX, is among the most noticeable, particularly since it is closely related to accessing shared resource called the critical section (CS) [7], which (among other uses) provides an exclusive access basis for implementing the critical section. The mutual exclusion problem [1] requires two properties, safety and liveness, from a given set of processes. The problem has been widely studied in the research community [2, 3, 4, 5, 6] since one reason for this wide interest is that many distributed protocols need an mutual exclusion protocol. However, despite its usefulness, to our knowledge there is no work that has been devoted to this problem in a mobile computing environment.

The aim of this paper is to propose a solution to the mutual exclusion problem in a specific mobile computing environment. This solution is based on the token-based mutual exclusion algorithm that is a classical one for distributed systems. The rest of this paper is organized as follows: in Section 2, a solution to the mutual exclusion problem in a conventional synchronous system is presented. Section 3 describes the mobile system model we use. A protocol to solve the mutual exclusion problem in a mobile computing system is presented in Section 4. We conclude in Section 5.

2. Mutual Exclusion in a Static System

2.1 Model and Definitions

We consider a synchronous distributed system composed of a finite set of process \( \Pi = \{p_1, p_2, \ldots, p_n\} \) connected by a logical ring. Communication is by message passing, synchronous and reliable. A process fails by simply stopping the execution...
(crashing), and the failed process does not recover. A correct process is the one that does not crash. Synchrony means that there is a bound on communication delays or process relative speeds. Between any two processes there exist two unidirectional channels. Processes communicate by sending and receiving messages over these channels.

The mutual exclusion problem is specified as following two properties. One is for safety and the other is for liveness. The safety requirement asserts that any two processes connected the system should not have permission to use the critical section simultaneously. The liveness requirement asserts that every request for critical section is eventually granted. A mutual exclusion protocol is a protocol that generates runs that satisfy the mutual exclusion specification.

2.2 Token-based Mutual Exclusion Algorithm

As a classic paper, the token-based mutual exclusion algorithm, which was published by M. Raynal, specifies the mutual exclusion problem for synchronous distributed systems with crash failures and gives an elegant algorithm for the system; this algorithm is called the token-based MX Algorithm [2]. The basic idea in the token-based MX algorithm is that the any process holding the token can use the critical section exclusively. The token-based MX algorithm is described as follows.

- A distributed system is connected by a logical ring. Each process has a unique ID that is known by its neighborhood processes.
- The CS is exclusively used by the process holding the token.
- The token is circulated on the logical ring. If a process wants to use the CS, then it just waits until receiving a token from its neighborhood. Only when it has received the token, it has a right to use the CS exclusively.
- When the process with the token finished its use of CS, it immediately passes the token to its neighborhood.
- If a process doesn’t use a CS when it received the token, it just pass the token to it neighborhood.
- There exists only one token and the token is continuously circulated upon the logical ring.
- By doing this, any process eventually receives the token and it can use the CS exclusively, which means that this algorithm satisfies both of the safety and the liveness properties.

3. Mobile System Model

A distributed mobile system consists of two set of entities: a large number of mobile hosts (MH) and a set of fixed hosts, some of which act as mobile support stations (MSSs) or base stations. The non MSS fixed hosts can be viewed as MSSs whose cells are never visited by any mobile host. All fixed hosts and all communication paths connect them from the static network. Each MSS is able to communicate directly with mobile hosts located within its cell via a wireless medium. A cell is the geographical area covered by a MSS. A MH can directly communicate with a MSS (and vice versa) only if the MH is physically located within the cell serviced by the MSS. At any given instant of time, a MH can belong to one and only one cell. In order to send message to another MH that is not in the same cell, the source MH must contact its local MSS which forwards the messages to the local MSS of the target MH over the wireless network. The receiving MSS, in its turn, forwards the messages over the wireless network to the target MH. When a MH moves from one cell to another, a Handoff procedure is executed by the MSSs of the two cells. Message propagation delay on the wired network is arbitrary but finite and channels between a MSS and each of its local mobile hosts ensure FIFO delivery of messages.
- The amount of computation performed by a mobile host should be kept low
- The communication overhead in the wireless medium should be minimal.
- Algorithm should be scalable with respect to the number of mobile hosts.
- Algorithm should be able to easily handle the effect of mobile host disconnections and connections.

4. Mutual exclusion in a Mobile System

In the following, we consider a broadcast group $G = (G_{MSS}, G_{MH})$ of communicating mobile hosts, where $G_{MH}$ and $G_{MSS}$ are respectively a set of $m$ mobile hosts roaming in a geographical area (like a campus area) covered by a fixed set of $n$ MSSs. In so far, local mobile hosts of base station $MSS_i$, which currently residing in $MSS_i$ cell, will refer to mobile hosts that belong to group $G$.

A mobile host can move from one cell to another. If its current base station fails, the connection between the mobile host and the rest of system is broken. To recover its connection, a mobile host must move into another cell covered by an operational or correct base station. So, unless it crashes, a mobile host can always reconnect to the network. A mobile host may fail or voluntarily disconnect from the system. When a mobile host fails, its volatile state is lost.

In this environment, the mutual exclusion problem is defined over the set $G_{MH}$ of mobile hosts. When a mobile host $h_k$ wants to use the CS, it sends the request message to a $MSS$. In this case, the mobile host eventually should get the permission from the $MSS$ and use the CS. Due to the resources constraints of mobile hosts and the limited bandwidth of wireless links, the distributed algorithm to solve mutual exclusion is executed by the set of $MSS$ on behalf of the set $G_{MH}$ of mobile hosts. In a first phase, the $MH$ which wants to use the CS has to request the permission from the $MSS$ in the cell which it belongs to. The $MSS$ receiving those requests from the subset of $G_{MH}$ of mobile hosts roaming in their respective cells keeps them in its queue. A token is circulated through the logical ring which consists of the fixed $MSS$s. In the second phase, when each $MSS$ receives the token from its neighborhood, it sends the token to a mobile host $h_k$ to give permission for the CS.

Finally, the $h_k$ received the permission from the $MSS$ uses the CS and after using it returns the permission to the $MSS$. The $MSS$ which has got the permission back from the $h_k$ sends the token to the next turn of $MSS$s.

4.1 Principle

The mutual exclusion protocol proposed in this paper is based on the solution described by
Raynal in Token-based MX algorithm [2]. The outlines of their protocol have been described in Section 2. In this section, we give an overview of our protocol and identify the major differences compared with the original token-based MX algorithm. We assume that the mutual exclusion is initiated by a mobile host which requests its current base station a token to use the CS. The contacted base station saves the request into the queue until it receives the token from its neighborhood.

During the mutual exclusion, each base station on one hand interacts with the mobile hosts located in its cell to gather the request of each mobile host for CS and on the other hand interacts with the other neighboring base stations to send and receive a token. In our approach, a base station \( \text{MSS} \) which participates in the mutual exclusion protocol, always acts on behalf of a subset of mobile hosts.

More precisely, the initial value of \( \text{Token}_{\text{Holdekk}} \) is false but the value of it is changed true as a mobile host \( h_k \) that resides in \( \text{MSS}_i \) receives the token from its \( \text{MSS}_i \). After returning the token to its base station, the mobile host \( h_k \) changes the value of its \( \text{Token}_{\text{Holder}} k \) into false again.

The mutual exclusion algorithm among base stations is similar to the token-based MX in static distributed systems. That is, only the base station holding the token has a permission to use the CS. In the second case, that is when a mobile host received a token from its host base station. Then it just uses the CS for a while and returns the token to its host base station after finishing it.

In above scenario, we don’t consider the mobility of the mobile host in the MX algorithm. But if we consider the mobility of the mobile host, then it makes the MX problem more complicated than the one of static distributed systems.

The differences of mutual exclusions between mobile computing environments and static distributed systems are as follows:

1) During the period of the MH using the CS, the MH changes its base station from the one that it received the token to the other base station. In this case, the MH simply sends the token to the base station of the cell in which it resides. But the base station that received the token takes some action to keep the fairness of the MX. The base station that did not send the token but received the token from its MH simply sends it to the base station which waits for the token to keep the fairness of the MX. Because, as a big difference between mobile computing environments and static distributed systems, the mobile host with token will appear in any cell whenever mutual exclusion protocol has started. Therefore, every base station should check all other base stations to know which base station cover the mobile host holding the token in the cell. That causes message traffics among base stations.

2) In mobile computing environment, a handoff algorithm is needed to perform mutual exclusions correctly, but it is not needed in static distributed systems.

3) Due to the resource constraints of mobile hosts and the limited bandwidth of wireless links, the distributed algorithm to solve mutual exclusion is executed by the set of \( \text{MSS}_i \) on behalf of the set \( G_{\text{MH}} \) of mobile hosts.

4.2 Protocol

The protocol is composed of three parts and each part contains a defined set of actions. Part A
(figure 2) describes the role of an arbitrary mobile host \( h_k \). Part B (figure 3) presents the protocol executed by a base station \( MSS_i \).

\[
\begin{align*}
% \text{Mobile host } h_k \text{ is located in } MSS_i \text{ cell } \\
(1) & \text{Upon receipt of the request for CS from the application, send } Req\_Token \text{ to } MSS_i \\
(2) & \text{Upon receipt of Token from } MSS_i, \text{ the mobile host } (h_k) \text{ gets into CS} \\
(3) & \text{Upon receipt of the release for CS from the application, send } Release\_Token \text{ to } MSS_i
\end{align*}
\]

Figure 2: Protocol Executed by a Mobile Host \( h_k \) (Part A)

Part B is related to the interactions between a base station and its local mobile hosts on one hand and the other base station on the other hand. Thus, Part B is based on the traditional Token-based MX protocol adapted to our environment.

Finally, the part C of the protocol is the handoff protocol destined to handle mobility of hosts between different cells. In figure 2, the three actions performed by an arbitrary mobile host are:

1. A mobile host executes this action when it receives a request from an upper application program to initiate a mutual exclusion.
2. Token message is sent to a mobile host \( h_k \) by the mobile support systems \( MSS_i \) when it had requested a token from the local base station where it resides. Upon receipt of such a message, the mobile host gets into the Critical Section.
3. When the application program terminates the mutual exclusion protocol, the Token is released to the mobile support system, \( MSS_i \).

Actions of the protocol in figure 3 numbered from (4) to (7) are executed a mobile support system, i.e., a base station \( MSS_i \). They have the following meaning:

4. When a base station is asked by a mobile host to send a Token, it inserts the request into the rear of its queue.

\[
\begin{align*}
\text{My}\_\text{Status}_i & := 0; \\
\text{My}\_\text{Queue}_i & := \emptyset; \\
\text{Cobegin} \\
(4) & \text{Upon receipt of } Req\_\text{Token}( h_k ) \\
& \text{insert } Req\_\text{Token}( h_k ) \text{ to rear (My}\_\text{Queue}_i); \\
(5) & \text{Upon receipt of } Token(MSS_i) \\
& \text{if My}\_\text{Queue}_i \neq \emptyset \text{ then} \\
& \quad \text{My}\_\text{Status}_i := 1; \\
& \quad \text{send Token to front (My}\_\text{Queue}_i); \\
& \quad \text{delete front (My}\_\text{Queue}_i); \\
& \quad \text{else} \\
& \quad \text{send Token to } MSS_{i+1}; \\
& \text{end-if} \\
(6) & \text{Upon receipt of } Token( h_k ) \\
& \text{if ( Phase}_i = 0 \land \text{My}\_\text{Queue}_i \neq \emptyset \text{ ) then} \\
& \quad \text{My}\_\text{Status}_i := 1; \\
& \quad \text{send Token to front (My}\_\text{Queue}_i); \\
& \quad \text{delete front (My}\_\text{Queue}_i); \\
& \quad \text{else} \\
& \quad \text{My}\_\text{Status}_i := 0; \\
& \quad \text{send Token to } MSS_{i+1}; \\
& \text{end-if} \\
(7) & \text{Upon receipt of } Req\_\text{Token}( MSS_j ) \\
& \text{insert } Req\_\text{Token}( h_k ) \text{ to Rear(My}\_\text{Queue}_i);
\end{align*}
\]

Figure 3: Protocol Executed by a mobile support station \( MSS_i \) (Part B)

5. In case of receiving a Token from other base station, the base station checks its queue My\_Queue, to confirm whether the queue is empty or not. If the queue is not empty, then the base station sends the Token to the mobile host that is positioned at the front of the queue. And it deletes the element from the queue and sets its status to true that means it holding Token, i.e., \( My\_\text{Status}_i := 1 \). But if the queue is empty, then the base station just passes the Token to the next base station.

6. When a base station receives a Token from a mobile host \( h_k \), it checks its queue and status. If both ( Phase\(_i = 0 \land My\_\text{Queue}_i \neq \emptyset \) ) are true, which means that it does not hold the token and at the same time the queue is not empty, then the base station sends the Token to the mobile host that is the front element of the queue. And it deletes the element from the queue and sets its status to true. Otherwise it sends the Token to the next base station and sets its status to false.
On receiving the Token request message from other mobile support system, the MSS inserts the request message into its queue.

As shown in Figure 4, the handoff protocol is described.

When a mobile host \( h_k \) moves from MSSj cell to MSSi cell, the handoff protocol execution is triggered. Mobile host \( h_k \) has to identify itself to its base station by sending a message GUEST(\( h_k, MSS_j \)).

Upon receiving this message, MSSi learns that a new mobile host \( h_k \), coming from MSSj cell has entered in its cell. With BEGINHANDOFF(\( h_k, MSS_i \)) message, MSSi informs MSSj that it removes \( h_k \) from the set of mobile hosts that reside in its cell.

Upon receiving such a message, MSSj checks its queue to confirm that the token request of \( h_k \) is in the queue. If it is in its queue, then it transfers the token request to MSSi and deletes the token request from the queue.

4.3 Correctness Proof

As our protocol is based on the Token-based logical ring algorithm proposed by M. Raynal, some statements of lemmas and theorems that follow are similar to the ones encountered in [2].

**Theorem 1** No two different processes can have permission to use the critical section simultaneously (safety property).

**Proof** (proof by contradiction). Let assume that there exist two mobile hosts to get a permission to use the critical section. A mobile host can use the CS only if it received a permission token from the MSS of the cell to which it belonging (action 2). In this case, the assumption means that there exist two MSSs holding the token or one MSS sends the token twice to two different mobile hosts each. The first case is false since there is only one token circulating under the logical ring. The second case is also false since the MSS holding the token sends it to mobile host \( h_k \) only once (action 5). So it is a contradiction. □

**Theorem 2** Every request for the critical section is eventually granted (liveliness property).

**Proof** If a mobile host sends a message to request a token (action 1), at least one MSS eventually receives it and inserts it into the queue (action 4). After that, there are two cases. In first case, if the mobile host \( h_k \) sent the message does not move to other cell, then the message Req-Token eventually will be positioned at the front of the queue and the MSS received the message sends the token. Thus, the mobile host sent the message eventually receives the token and uses the CS. In a second case, when the mobile host \( h_k \) sent a message Req-Token moves from MSSj cell to another MSSi cell before receiving the token, then the handoff protocol execution is triggered (action 8-10). Mobile host \( h_k \) has to identify itself to its base station by sending a message GUEST(\( h_k, MSS_j \)). In this case, by (action 10) the request message will be transferred to the MSS of the cell to which the mobile host has moved. Consequently, the mobile host will receive the Token and use the CS when the MSS sends the Token. □

5. Conclusion

The communication over wireless links are limited to a few messages (in the best case, three messages: one to request a token and the others to get the token and release the token respectively)
and the consumption of mobile hosts CPU time is low since the actual mutual exclusion is run by
the base stations. The protocol is then more energy efficient. The protocol is also independent
from the overall number of mobile hosts and all needed data structures are managed by the base
stations. Therefore, the protocol is scalable and can not be affected by mobile host failures.

In addition, other interesting characteristics of the protocol are as follows. 1) During the mutual
exclusion period, a base station should keep track of every mobile host within its cell to manage
the request messages and the token. 2) In such a mobile computing environment, a handoff
algorithm is needed to perform mutual exclusions efficiently and correctly, but it is not needed in
static distributed systems.

The mutual exclusion algorithm in a mobile computing environment consists of two important
phases. One is a local mutual exclusion phase in which a mobile host holds and uses the CS. The
other phase is a global mutual exclusion phase in which each MSS takes part in the mutual
exclusion by passing the token to another MSS.

References
Computer Systems 3 (2) (1985) 145.159.
distributed systems, in: Proceedings of the ISCA International Conference on Parallel
and tradeoff analysis. FTCS-26, June 1996.
[10]Badache N., Mobility in Distributed Systems, Technical Report #962, IRISA, Rennes,
Octol 1995.