An Efficient Election Protocol in a Mobile Environment

Sung-Hoon Park
Dept. of Computer Science, NamSeoul Univ., Chung-Nam, KOREA
spark@nsu.ac.kr

Abstract

The Election paradigm can be used as a building block in many practical problems such as group communication, atomic commit and replicated data management where a protocol coordinator 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 an election protocol. However, despite its usefulness, to our knowledge there is no work that has been devoted to this problem in a mobile computing environment. Mobile systems are more prone to failures than conventional distributed systems. Solving election in such an environment requires from a set of mobile hosts to choose a mobile host or a fixed host based on their priority despite failures of both mobile computing and/or fixed hosts. In this paper, we describe a solution to the election problem from mobile computing systems. This solution is based on the Garcia Molina's Bully 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. Election is among the most noticeable, particularly since it is closely related to group communication [7], which (among other uses) provides a powerful basis for implementing active replications.

The Election problem [1] requires that a unique coordinator be elected 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 election 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 election problem in a specific mobile computing environment. This solution is based on the Garcia’s Bully algorithm that is a classical one for synchronous distributed systems. The rest of this paper is organized as follows: in Section 2, a solution to the election problem in a conventional synchronous system, is presented. Section 3 describes the mobile system model we use. A protocol to solve the election problem in a mobile computing system is presented in Section 4. We conclude in Section 5.

2. Election in a Static System

We consider a synchronous distributed system composed of a finite set of process \( \Pi = \{p_1, p_2, ..., p_n\} \) completely connected. 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 Election problem is specified as following two properties. One is for safety and the other is for liveness. The safety requirement asserts that all processes connected the system never disagree on a leader. The liveness requirement asserts that all processes should eventually progress to be in a state of normal operation in which all processes connected to the system agree to the only one leader. An election protocol is a protocol that generates runs that satisfy the Election specification.

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 (MSS) or base stations. The non MSS fixed hosts can be viewed as MSS, 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 MSS 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.

3.1 Characteristics of Mobile Hosts

The bandwidth of the wireless link connecting a MH to a MSS is significantly lower than bandwidth of the links between static hosts [9]. In addition, mobile hosts have tight constraints on power consumption relative to desktop machines, since they usually operate on stand-alone energy sources such as battery cells. Consequently, they often operate in a doze mode or voluntarily disconnect from the network. Transmission and reception of messages over wireless links also consume power at a MH. So, distributed algorithm for mobile systems need to minimize communication over wireless links. Furthermore, mobile hosts are less powerful than fixed hosts and have less memory and disk storage. Hence, while designing distributed algorithms for mobile systems, the following factors should be taken into account [10,11]:

- 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. Election in a Mobile System

In the following, we consider a broadcast group $G = \{G_{MH}, G_{MSS}\}$ 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 $n$ MSS. In so far, local mobile hosts of base station MSS, which currently residing in MSS 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 election problem is defined over the set $G_{MH}$ of mobile hosts. When a mobile host $h_k$ detects a leader failure, it broadcasts the leader crash to all other mobile hosts through its $G_{MSS}$.

In this case, a mobile host with highest priority eventually should be elected as a new leader. Due to the resources constraints of mobile hosts and the limited bandwidth of wireless links, the distributed algorithm to solve election is executed by the set of MSS on behalf of the set $G_{MH}$ of mobile hosts. In a first phase, MSS have to elect their local leaders amongst the subset of $G_{MH}$ of mobile hosts roaming in their respective cells. In the second phase, each MSS starts bully algorithm to elect a global leader in the broadcast group G. Finally each MSS forwards the newly elected leader to the mobile hosts that currently reside in its cell.

4.1 Principle and 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 presents the protocol executed by a base station MSS. It is subdivided in two sub-parts: sub-part B1 (figure 3) and sub-part B2. Sub-part B1 is related to the interactions between a base station and its local mobile hosts, on one hand and the election coordinator on the other hand. Sub-part B2 is the Bully protocol adapted to our environment to elect an election coordinator among base stations. So, it is abbreviated. 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 an election.

(2) Message INIT_3 is sent to a mobile host either when its local base station is informed (on receipt of INIT_2) that an election has started or when the mobile host enters a cell managed by a base station which is not yet aware of its value. Upon receipt of such a message, each mobile host sends to the MSS its id that represents the priority of the mobile host $h_k$.

(3) When the election protocol terminates, the id of the newly elected leader is forwarded to each mobile host.

% Mobile host $h_k$ is located in MSS, cell %
(1) Upon the program application requires to start an election

   Send INIT_1 to MSS_i

(2) Upon receipt of INIT_3 from MSS_i

   % Let the value of variable id be a value provided by the application program.

   Send PROPOSE($h_k$) to MSS_i

(3) Upon receipt of DECIDE(New_Leader) from MSS_i

   % The result of the election protocol is % delivered to the application program

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

Actions of the protocol in figure 3 numbered from (4) to (8) are executed a mobile support system MSS_i. They have the following meaning:

(4) When a base station is asked by a mobile host to initiate an election, it sends an INIT_2 message to inform the other base station that an election has started. Next, each base station starts to collect values from local mobile hosts, if any, and until time-out holds ($End_{collect_i} = true$). Testing if $Phase_i = 0$ ensures that the election is undergoing.

(5) When a base station receives an INIT_2 from other base stations, it has to forward an has INIT_2 message to all base stations to ensure a reliable broadcast of message INIT_2. So, despite failures of base stations, all correct base stations will be aware that an election been initiated. Next, like (5), each base station starts to collect values from local mobile hosts until time-out holds ($End_{collect_i} = true$).

(6) Each base station MSS_i gathers the ids of its local mobile hosts, while time-out is false.

(7) On time-out, MSS_i lets the variable $End_{collect_i}$ be true and chooses a local host with highest priority as its local leader. After that, MSS_i recommends it as a global new leader to the election coordinator.

(8) A base station MSS_i receives a message DECIDED($h_k$) when the election coordinator has gathered all local host ids proposed from other MSS_s and has decided a host with highest priority as a global leader. MSS_i adapts this host as new global leader, changes its state to decided, forwards the decided global leader to local mobile hosts and terminates ($Phase_i = 0$). To ensure that all correct processes have decided, the message is also forwarded to all the other base stations (reliable broadcast).
When a mobile host
and (10) are executed a mobile support system which is an
election coordinator. They have the following meanings:

(5) || Upon receipt of INIT_1
if Phase, := 0 then
send INIT_2 to all MSS, except MSS;
Phase, := 1;
if ~End_collect, then
W_Broadcast INIT_3
end-if
end-if

(6) || Upon receipt of PROPOSE( h_k )
if ~End_collect then
% Value collection %
P_i := P_i \cup \{ h_k \};
end-if

(7) || On time-out
End_collect := true;
% MSS, propose an estimate for
a new leader %
Phase, := 2;
Leader, := Max (P_i);
Send ESTIMATE(MSS, Leader,) to MSS;

(8) || Upon receipt of DECEIDE (h_k)
if (State, = undecided ) then
State, = decided;
Phase, := 0;
Leader, := h_k;
W_Broadcast DECEIDE(Leader,)
end-if

Figure 3: Protocol Executed by a Base Station MSS_i (Part B)

Actions of the protocol in figure 4 numbered from (9)
and (10) are executed a mobile support system which is an
election coordinator. They have the following meanings:

(9) When time is out, the election coordinator MSS_i,
chooses a host with highest priority among hosts
recommended from other base stations and decides it
as a new global leader. After that, the MSS_i sends the
id of global leader to other base stations, changes its
state to decided and terminates the role of the
coordinator.

(10) When MSS_i has received an id of a local
host proposed by a MSS, it saves the id into the set
while it doing the role of the coordinator.

Coordiator := true; V_i := \emptyset;
Cobegin
(9) || On time-Out
% MSS, decide a new leader %
Leader, := Max (V_i);
Send DECIDE(Leader,) to all MSS;,
Coordinator := false;
(10) || Upon receipt of ESTIMATE(h_k)
if Coordinator = true then
V_i := V_i \cup \{ h_k \};
end-if
end-if

Figure 4: Protocol Executed by MSS_i

As shown in Figure 5, the handoff protocol is reduced.
When a mobile host h_k moves from MSS_j cell to MSS_i 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,
MSS_j learns that a new mobile host h_k, coming from MSS_j,
cell has entered in its cell. MSS_j, informs MSS_i which
removes h_k from the set of mobile hosts that reside in its
cell and eventually transfers information about the last
state of h_k to MSS_j. MSS_j queries h_k to send back its id
value, if an election has already started and the value
collection is still possible.

% Role of h_k %
Upon entry in MSS_i cell
Send Guest(h_k, MSS_j) to MSS_j,
% Role of MSS_j
Upon receipt of GUEST(h_k, MSS_j)
Local_MH_i := Local_MH_i \cup \{ h_k \};
Send BEGIN_HANOFF(h_k, MSS_j) to MSS_j;
if Phase, = 0 \land State, = undecided
then send INIT_3 to h_k
end-if
if Phase, = 0 \land State, = decided
then send V_i to h_k
end-if
% Role of MSS_j
Upon receipt of BEGIN_HANOFF(h_k, MSS_j)
Local_MH_i := Local_MH_i \setminus \{ h_k \}

Figure 5: Handoff Procedure
4.2 Correctness Proof

As our protocol is based on the Bully algorithm proposed by Garcia Molina, some statements of lemmas and theorems that follow are similar to the ones encountered in [2].

Theorem 1 All the mobile hosts in the system never disagree with a leader when the hosts are in the state of a normal operation (safety property).

Proof A mobile host decides (action 3) a leader only if its base station has decided (action 6) in that case the mobile host adapts the leader broadcasting by this base station. Consequently, theorem 1 is valid if no two base stations decide differently. Assume that at least one base station has decided (action 8). In that case, a coordinator has previously broadcast a message DECIDED (action 9). So, at least a majority of base stations have adopted the host as a new leader sent by the election coordinator. Eventually, all base stations in the system reach to the decision adopting a mobile host as a new leader since the base stations received the leader broadcasting send it again to other base stations. Therefore, all base stations never disagree with a leader when the base stations are in the state of a normal operation. \( \square \) Theorem 1

Theorem 2 All the processes should eventually progress to be in the state of a normal operation in which all hosts in the system agree to the only one leader (liveliness property).

Proof If at least one base station decides and does not crash then all collect base stations eventually deliver such a DECIDE message. This is due to the fact that a base station forwards the decided leader when it delivers such a message (action 8). Consequently, any mobile host will receive the decided leader either when its base station decides (action 8) or when it enters in the cell of a base station that previously decided. \( \square \) Theorem 2

5. Conclusion

The communication over wireless links are limited to a few messages (in the best case, two messages: one to inform the initial value and the other to get the decided leader) and the mobile hosts CPU time is low since the actual election 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. So, the protocol is scalable and can not be affected by mobile host failures.

Another interesting characteristics of the protocol are as follows. 1) During the election period, a base station that plays a role of the election coordinator should be elected to reduce the message traffics between base stations before electing a global mobile host which acts as a leader. 2) In such a mobile computing environment, a handoff algorithm is needed to perform elections efficiently, but it is not needed in static distributed systems.

The election algorithm in a mobile computing environment consists of two important phases. One is a local election phase in which all MSS, have to elect local leaders amongst the set of \( G_{MH} \) of mobile hosts in their respective cells. The other is a global election phase in which each MSS takes part in the election of a global leader among all MSS, in the broadcast group \( G \).

References