Fundamental Distributed Protocols in Mobile Networks *

Kostas P. Hatzis[1,2] George P. Pentaris[1,2] Paul G. Spirakis[1,2]
Basilis T. Tampakas[1,3] Richard B. Tan[4]

April 19, 1999

Contact author: Kostas Hatzis

Abstract

In this work we propose simple and efficient protocols for counting and leader election in mobile networks. For mobile networks with fixed base stations we provide a new and very efficient protocol for counting the number of mobile hosts. The main part of the work concentrates on *ad-hoc networks* (no fixed subnetwork). We provide a *model* for these networks and leader election (and a special form of counting) protocols for both named and anonymous mobile hosts. In this work we define two protocol classes, the *Non-Compulsory* protocols, which do not affect the motion of the hosts and the *Compulsory*, which determine the motion of some or all the hosts. By assuming that the mobile hosts move as if each one is doing a continuous random walk on their allowable space S of motions, and by assuming a universal time, we show that our leader election protocol terminates (with high probability and also on the average) in time asymptotically *linear* to the size of the space S, measured as its volume divided by the volume of the sphere defined by the range of transmission of each mobile host. We also provide a simple but very efficient Compulsory (forced random walks) Las Vegas protocol for leader election in ad-hoc networks which also allows counting, with termination detection. Our analysis techniques for the meeting time of concurrent random walks extend the known facts and are tight. They may be used as an analysis tool in the design of many other distributed protocols. This is the first algorithmic and characterization work, to our knowledge, for ad-hoc networks.

1 Introduction

A new computing environment which is referred as *mobile* or nomadic computing has already created an entire new class of distributed applications and new massive markets, ranging from personal computing to consumer electronics. A wide variety of information servers are (or will be) accessible to mobile computers. New information infrastructures, like the "information superhighway", organized in a highly decentralized and distributed way, are expected to emphasize the nomadic approach (see [14]).

Mobile computing is characterized by four constraints (see [22]):

- Mobile elements are resource-poor relative to static elements.

*This work was partially supported by the EU ESPRIT LTR ALCOM-IT. (contract No. 20244).

[1] Computer Technology Institute Kolokotroni 3, 26221 Patras, Greece. Tel:+3061273496, Fax:+3061222086, E-mail: {hatzis,pentaris,spirakis,tampakas}@cti.gr

[2] Computer Engineering and Informatics Department, Patras University, 26500 Rion, Patras, Greece.

[3] Technological Educational Institute (TEI) of Patras, M Alexandrou 1, 26334, Koukouli, Patras, Greece.

[4] Universiteit Utrecht, Department of Computer Science, Centrumgebouw Noord, Padualaan 14, De Uithof, 3584 C4 Utrecht, The Netherlands. Email: rbtan@cs.uu.nl

- Mobility is inherently hazardous (concerning physical security loss or damage).
- Mobile connectivity is highly variable in performance and reliability.
- Mobile elements rely on a finite energy source.

The above constraints complicate the design of mobile information systems and require the redesign of traditional approaches to information access and to some fundamental distributed computing problems.

Until now, two basic system models have been proposed for mobile computing. The "fixed backbone" mobile system model assumes two distinct sets of entities in a mobile network: a large number of mobile hosts, and relatively fewer but more powerful, fixed hosts ([2]). All fixed hosts and the communication paths between them constitute the *static* or *fixed* network. The geographical area that is served by the fixed network is divided into smaller regions called cells. Each cell is served by a fixed host. A fixed host communicates with the mobile hosts within its cell via a wireless medium. Host mobility is represented in this model as migration of mobile hosts between cells.

The "*ad-hoc*" system model assumes that mobile hosts can form networks without participation of the fixed infrastructure. The structure of an ad-hoc mobile network is highly dynamic. Routing tables may change frequently due to changing communication conditions and power levels. Such networks arise in rapid-deployment situations, like emergency services at a disaster site, or military operations in a remote area or business situations such as meetings held in venues without network infrastructure (see [14]).

In this work we deal with two fundamental problems of distributed computing, the problems of process or node counting and the leader election problem. Both problems were extensively studied in the case of networks consisted only of static hosts. In the case of mobile networks these problems preserve their importance, since they are building blocks for solution to more complicated problems, and can be used both by the control (i.e. routing, data management) and the application level (i.e. sales and inventory applications, see [14]) of the network.

In our work, we deal both with mobile networks with fixed wired infrastructure and with ad-hoc networks. In the first case we give an efficient protocol for counting the number of mobile hosts, that has a total cost of $mC_{wireless} + O(n^2)C_{fixed}$, where m and n are correspondingly the number of mobile hosts and the fixed nodes, and $C_{wireless}$ and C_{fixed} are the costs of wireless and fixed network message exchange. Our protocol is based on the well known *Echo* algorithm ([19]) and it moves successively most of the network activity on the static portion of it, as proposed by the *two tier* principle (see [2]).

The main contribution of this work concentrates on ad-hoc networks. We extend the existing model in two directions. First we give a graph theoretic concept of the movement of mobile hosts in the three-dimensional space. Under the new model the space of movement S can be mapped to a graph $G(V, E)$, $|V| = n, |E| = \epsilon$. In this graph, n approximates the ratio between the volume of space S and the space occupied by the transmission range of a mobile host and $\epsilon = O(n)$. Second we propose and distinguish two classes of mobile ad-hoc protocols. *Non Compulsory* protocols do not affect the natural movement of the mobile host. *Compulsory* protocols require the hosts to perform certain moves in order to ensure the correct protocol execution. The sense of orientation directly affects Compulsory protocols and thus it is seriously considered by the new model.

We present a Non Compulsory protocol for leader election and a special form of counting (where the leader finally knows the number of the hosts). This protocol does not include termination detection. We analyze the performance of the protocol by assuming that the hosts are performing concurrent random walks on G and also that there is a universal time not necessarily known to the hosts and we show that the execution time of the protocol is linearly bounded (on the average) on n. Our results are asymptotic in nature. Our analysis also extends the known results ([4]) for meeting times in coalescing random walks. Based on this result we present Las Vegas Compulsory protocols for leader election and counting for ad hoc networks where the mobile hosts have no sense of orientation. These protocols force the hosts to perform random walks on G. In this case we provide a termination detection mechanism if the size of the space S is known and we show how our protocols can be applied to the case of anonymous mobile networks (where the hosts do not have distinct identities). Finally, we study the behavior of the steady state of the Markov chain induced by the random walks of the hosts on G and we show that it follows the behavior of the chain studied by Saloff-Coste and Diaconis in [10]. Our analytic results do not take into account details of the structure of the graph G. However, our methods show how more knowledge of the structure can tighten the results.

To our knowledge, there is no related previous work especially in the case of ad-hoc networks. However random walks is a powerful tool and has been extensively studied and used by many researchers in the past, in the area of distributed computing. Among the others we refer Israeli and Jalfon ([16]), who provided a solution for the token management problem on general graphs, introducing the tool of random walks. Anagnostou and El-Yaniv in [3] presented a self-stabilizing randomized protocol for the Unique Naming problem, using random walks. Their protocols work properly even under a powerful scheduler that besides controlling processor activations, knows at any stage the results of any bounded number of future random choices that will be made by the processors and, in addition, can bias by any constant all future random choices. Spirakis and Tampakas in [23], and Spirakis et al in [24] proposed and studied the pursuit-evasion problem in distributed environments. Their protocols also made use of accidental meetings of random walks.

The rest of the paper is organized as follows: In Section 2 we present a counting protocol for mobile networks with fixed base station infrastructure. In Section 3 we present the new graph theoretic model for ad-hoc networks. We describe the basic Non Compulsory protocol for leader election and we study its performance. Then we describe the termination detection mechanism and we give Compulsory protocols for leader election and counting. Finally, in Section 4 we present simulation results verifying the protocol performance.

2 Mobile networks with fixed base stations

2.1 The system model

A host that can move while retaining its network connections is a *mobile host* ([8]). The geographical area that is served by the fixed base station network is divided into smaller regions called *cells*. Each cell has a base station also referred to as the *Mobile Service Station (MSS)* of the cell. All mobile hosts that have identified themselves with a particular MSS are considered to be *local* to the cell of this MSS. At any instance of time, a mobile host belongs to *only one* cell. When a mobile host enters a new cell it sends a $< join(mobile-host-id) >$ message to the new MSS. Therefore it is added in the list of local mobile hosts of this MSS. Each MSS is connected to the service stations of neighboring cells by a fixed high bandwidth network. The communication between a mobile host and its MSS is based on the use of low-bandwidth wireless channels. If a mobile host h_1 wants to send a message to another mobile host h_2 it first sends the message to the local MSS over a wireless channel. The MSS then forwards through the fixed network the message to the local MSS of h_2 which then forwards it to h_2 over its local wireless channels. Since the location of a mobile host within the network is neither fixed nor universally known (its "current" cell may change with every move), the local MSS of h_1 needs first to determine the MSS that currently serves h_2. This means that each message transmission between two mobile hosts incurs also a *search cost*. The following notation has been proposed in [1] for the description of the cost of messages exchanged in the network:

- C_{fixed}: The cost of sending a point-to-point message between any two fixed hosts.
- $C_{wireless}$: The cost of sending a message from a mobile host to its local MSS over a wireless channel (and vice versa). An extra cost may incur in order to allocate the wireless channels ([20], [21]).
- C_{search}: The cost (messages exchanged among the MSSs of the fixed network) to locate a mobile host and forward a message to its current local MSS. We consider that $C_{search} = aC_{fixed}$ where a depends on the location management strategy used (e.g. [5], [9]).

2.2 Notation and definitions

Let $G(V, E)$, where $|V| = n$ and $|E| = \epsilon = O(n^2)$, be the graph which describes the fixed part of the network (the network of the MSSs). Each vertex models an MSS. There exists an edge between two vertices if and only if the corresponding MSSs communicate directly (point-to-point) in the fixed network. Let m be the number of mobile hosts (we assume that usually $m >> n$). We define by D the diameter of G.

Based on the above notation, the search cost C_{search} is approximately $O(D)$. A message sent from a mobile host to another mobile host incurs a cost $2C_{wireless} + C_{search}$. This means that any algorithm based on the communication between mobile hosts requires a large number of messages to be exchanged over the fixed network and the wireless channels.

2.3 Definition of the problem

Suppose that m mobile hosts are moving throughout a fixed base station mobile network. One of the mobile hosts (the initiator of the algorithm) wants to find the size of the mobile network (the number of the mobile hosts, m). This is called the *Counting Problem.*

2.4 Assumptions

We assume that the communication between the MSSs is based on the *asynchronous timing* model. An operational mobile host responds to messages broadcasted by its local MSS immediately. Each mobile host has its own distinct identity which is transmitted to the local MSS when it enters a new cell.

2.5 The protocol

Suppose that the mobile hosts are willing to control by themselves the execution by avoiding the participation of the MSSs. Note that, sometimes, this would be the case if the protocol requires computational power that will increase the overhead of the MSSs (since their basic activity is to control the communications in the network). A fundamental algorithm for the solution of the counting problem in fixed networks can be found in [19] (p.190). It is basically a wave algorithm which can be used if a unique starter process (an initiator) is available. The application of this algorithm in a mobile network implies the definition of a "virtual" topology on the mobile hosts (each mobile host should be somehow assigned a set of "neighbors" at the beginning of the execution). By assuming that the virtual topology has $O(m)$ "edges", the total cost of the protocol would approximately be $O(m)C_{wireless} + O(Dm)C_{fixed}$. The appearance of the D factor is due to the search cost in the fixed network.

```
int size=0; the final size of the network
boolean counted=false; a flag indicating if the host has been counted
the initiator mobile host:
begin
   broadcast < count >;
   on receive < size, s >
    size=s;
end
the other mobile hosts:
begin
   on receive < count >
    if (not counted)
      begin
       broadcast < count_me >;
       counted=true;
      end
   on receive < size, s >
    size=s;
end
```

Figure 1: The counting protocol executed by the mobile hosts

Besides the high cost of message transmissions in the fixed network, another drawback of this approach is that it requires the participation of every mobile host in order to maintain the connectivity of the virtual topology and cannot therefore permit any one of them to disconnect during the execution (which is very usual in the case of mobile hosts). The MSSs act mainly as routers by forwarding messages from one mobile host to another and updating their routing tables when a mobile host joins their cell.

If the protocol execution is not very complex to require much of the MSSs computational power, more efficient solutions can be provided. A guiding principle for this case was presented in ([2]) and is called the *two tier* principle:

Computation and communication costs of an algorithm should be based on the static portion of the network. This leads to avoid locating a mobile participant and lowers the total search cost of the algorithm. Additionally, the number of operations performed at the mobile hosts is decreased and thereby consumption of battery power (which is critical resource for mobile hosts) is kept to a minimum.

The application of this simple principle on the design of distributed algorithms for mobile hosts has been studied in [2] in the case of a classical algorithm for mutual exclusion in distributed systems (Le Lann's token ring, [17]). In this section we propose and study the behavior of a protocol based on this principle that solves the counting problem in a mobile network with fixed base stations.

The proposed counting scheme is based on the execution of the *Echo* protocol. The protocol executed by the mobile hosts is presented in Figure 1. As can be seen, the execution is started by the *initiator* mobile host which broadcasts a $< count >$ message. Afterwards, the initiator itself does not respond to any $< count >$ message. The execution on the fixed part of the network (the MSSs) is described as follows:

1. In the first phase, the initiator MSS (the MSS serving the initiator) broadcasts a $< count >$

message in its cell and then spreads along the base station network the request for counting by using the *Echo* algorithm. The algorithm starts by sending $< count\_tok >$ messages to all neighboring MSSs.

2. After a base station has been informed about the execution of a counting protocol (by receiving a $< count\_tok >$ message) in the network it behaves as follows: It broadcasts a $< count >$ message to its cell and waits to collect answers from mobile hosts in this cell. The MSS also forwards a $< count\_tok >$ message to its neighbors in order to continue the execution of the *Echo*. By receiving a $< count\_me >$ message it increases $size_p$, the number of counted mobile hosts in its cell. If a base station receives a $< join(mobile - host - id) >$ message (from a mobile host joining its cell) it broadcasts again a $< count >$.

3. After the completion of the *Echo*, all of the MSSs have been informed about the execution of a counting algorithm in the network and have broadcasted a $< count >$ message in their cell.

4. The initiator base station starts a second execution of the *Echo* algorithm by sending a $< size\_tok, 0 >$ message to its neighbors. This execution aims to collect the $size_p$ variables from all MSSs to the initiator MSS. After completing the execution of the second *Echo* (by receiving answers from all children and sending its $size_p$ variable to its parent), each MSS stops to broadcast $< count >$ messages when a new mobile host joins its cell.

5. After the second completion of the *Echo* the initiator MSS knows the total number of mobile hosts in the network (it is its $< size_p >$ variable).

6. The initiator base station broadcasts a $< size, size_p >$ message in its cell and then starts the third execution of the *Echo* by sending a $< inform\_tok, size_p >$ message to its neighbors. Upon receiving such a message, an MSS broadcasts a $< size, size_p >$ message to its cell and forwards an $< inform\_tok, size_p >$ message to its neighbors in order to continue the execution. If a base station receives a $< join(mobile - host - id) >$ message, it broadcasts the $< size, size_p >$ again. After the third completion of the *Echo*, all the MSSs have broadcasted the size of the mobile network in their cells. The initiator starts a fourth execution of the *Echo* to inform the MSS about the completion of the counting. After the completion of the fourth *Echo* an MSSs stops to broadcast $< size >$ messages when a new mobile host joins its cell.

Note that with slight variations the basic protocol mechanism can be applied to elect a unique leader among the mobile hosts.

2.6 Correctness proof of the *two tier* protocol

It is easy to see that a mobile host cannot be counted twice even if it moves throughout the network during the execution of the protocol (it will respond only to one $< count >$ message). The major problem when executing a distributed protocol by using the MSSs is to inform all of the mobile hosts to participate in the execution. A faulty behavior may arise as follows. Suppose that MSSs S_1 and S_2 complete the execution of the MSS protocol at time t_1 and t_2 respectively, where $t_2 >> t_1$. It is possible that a mobile host starts moving from the cell of S_2 towards the cell of S_1 at time $t_1' < t_1$ and appears at the cell of S_2 at time t_2', where $t_1 < t_2' < t_2$. In this case the mobile host will not participate in the algorithm even if it is willing to do so. The solution to the problem is to maintain each MSS active (transmit $< count >$ and receive $< count\_me >$ messages) until all other MSSs are informed of the protocol execution (have received $< count\_tok >$ messages).

Lemma 2.1 *The* two tier *protocol counts the number of mobile hosts correctly.*

Proof: We consider an MSS as *notified* if it has received a $< count\_tok >$ message and therefore collects answers and broadcasts $< count >$ messages in its cell. If a mobile host travels from an MSS S_2 which is *not notified* to an MSS S_1 which is *notified*, S_1 is still waiting to collect answers (the execution of the first *Echo* has not been completed yet since S_2 is *not notified*) and therefore the mobile host will receive a $< count >$ from S_1 and reply to S_1 with a $< count\_me >$ message. □

2.7 Protocol analysis

By the nature of the *Echo* algorithm which is executed four times the protocol needs $8|E| = O(n^2)$ messages to be exchanged in the fixed network yielding a total cost of $mC_{wireless} + 8|E|C_{fixed}$ which approximately is still $mC_{wireless} + O(n^2)C_{fixed}$. The protocol is completed in time $8D$.

3 Ad hoc mobile networks

3.1 The system model

An ad hoc mobile network ([14]) is a collection of mobile hosts with wireless network interfaces forming a temporary network without the aid of any established infrastructure or centralized administration. Such a network may be formed from any type of vehicles and even simply persons, equipped with radio transmitters/receivers moving in a certain geographical area. It is expected to be a common form of mobile networks in the future (especially in the case where fixed base stations are not available). In an ad hoc network two hosts that want to communicate may not be within wireless transmission range of each other, but could communicate if other hosts between them are also participating in the ad hoc network and are willing to forward packets for them. Indeed, the routing problem in a real ad hoc network is much more complicated due to the propagation characteristics of wireless transmissions and due to the possibility that any or all of the hosts involved may move at any time (see [14]).

Suppose that m mobile hosts equipped with wireless transmitters and receivers are moving in a geographical area forming an ad hoc network. Suppose further that these hosts want to execute a simple distributed protocol such as *leader election*. One way to perform this is to utilize an underlying routing protocol which delivers (if possible) a message from one mobile host to another, regardless of their position. This is done by using the other mobile hosts as intermediate routers. If the identities of the mobile hosts that participate in the algorithm are known in advance, it is easy to define a "virtual" topology on the network by defining a set of "neighbors" for each host and then execute the protocol. This approach does not guarantee the successful completion of the whole computation since a mobile host may be unreachable from the rest of the network at some time instances and thus the protocol execution will fail. Another drawback of this approach is the large number of broadcasts, since each message transmission of the application protocol between two "neighbors" may require $m - 1$ broadcasts of intermediate hosts (regardless of the routing policy). Note also the extra cost of broadcasts used by the routing protocol in order to keep the routing tables updated. This scheme, in the case of increased mobility of the hosts, could lead to a situation where most of the computational and battery power of the hosts is consumed for the routing protocol.

The other way is to design protocols dedicated to solve specific problems in ad hoc networks. These protocols can be divided in two major categories. *Non-Compulsory* protocols are the ones whose execution does not affect the movement of the mobile hosts. On the other hand, *Compulsory* protocols are the ones that require the hosts to perform certain moves in order to ensure the correct protocol execution. Non-Compulsory protocols try to take advantage of the mobile hosts natural movement by exchanging information whenever mobile hosts meet incidentally. Compulsory protocols force the mobile hosts to move according to a specific scheme in order to meet the protocol demands (i.e. meet more often, spread in a geographical area, etc.). Furthermore, Compulsory protocols can be categorized into three cases according to the sense of orientation of the mobile hosts. In the first case the mobile hosts have *no sense of orientation.* In the second case each mobile host has an *individual sense of orientation* while in the third case the mobile hosts have a *common sense of orientation.*

3.2 A graph theoretic model for ad hoc networks

In order to design protocols for mobile hosts it is necessary to use an appropriate model that formulates the notion of movement and the formation of the ad hoc network.

In real world applications the most usual case is that the mobile hosts are moving in a two-dimensional space on the surface of a geographical area. The model that we propose is more general and suites also the case of hosts moving in the three-dimensional space. Suppose that in the three-dimensional space a mobile host has a transmission range represented by a sphere tr centered at itself. This means that any message broadcasted by this host can be received by any other host inside tr. We approximate this sphere by a regular polyhedron tc with volume $V(tc)$, where $V(tc) < V(tr)$. The size of tc can be chosen in such a way that its volume $V(tc)$ is the maximum that preserves $V(tc) < V(tr)$ and if a mobile host inside tc broadcasts a message, this message is received by any other host in tc. In Figure 2 for example, tc is a cube. In this example, given that the hosts are moving in the space S, S is divided into consecutive cubes of volume $V(tc)$ (see Figure 2). The graph $G(V, E)$, $|V| = n$, $|E| = \epsilon$, which corresponds to a quantization of S is constructed in the following way: a vertex $u \in V$ represents a cube of volume $V(tc)$. An edge $(u, v) \in V$ if the corresponding cubes share a face.

A host can move anywhere in S but at any instance of time it is inside a specific cube tc. Thus, at any time instance, a host resides in only one vertex of G. If the host knows the distance it has covered, it can determine the change of its position from one cube to another (from a vertex of G to another vertex). If a mobile host resides on a vertex u and broadcasts a message, this message will be received by all other hosts residing also on u. The number of vertices n, actually approximates the ratio between the volume of space S, $V(S)$, and the space occupied by the transmission range of a mobile host $V(tr)$. In the extreme case where $V(S) \approx V(tr)$ (the transmission range of the hosts approximates the space that they are moving), then $n = 1$. Given the transmission range tr, n depends linearly on the volume of space S regardless of the choice of tc, and $n = O(\frac{V(S)}{V(tr)})$. Since the edges of G represent neighboring polyhedra each node is connected with a constant number of neighbors, which yields that $\epsilon = O(n)$. In our example where tc is a cube, G has maximum degree of six and $\epsilon \leq 6n$.

3.3 Definition of the problem

Suppose that m mobile hosts are moving in a space S. The hosts want to elect a unique leader. This means that starting from a configuration where all mobile hosts are in the same state, a configuration should be reached where exactly one host is the *leader* while all other hosts are in a state *lost.* Furthermore, the leader should know the size of the mobile network, m.

3.4 Assumptions

We assume that the transmission delay while broadcasting a radio message is negligible (i.e. the message will be received by all other hosts in its range immediately). Each mobile host has its own distinct identity, i.e. the ad-hoc mobile network is *named.* In a later section we present a protocol

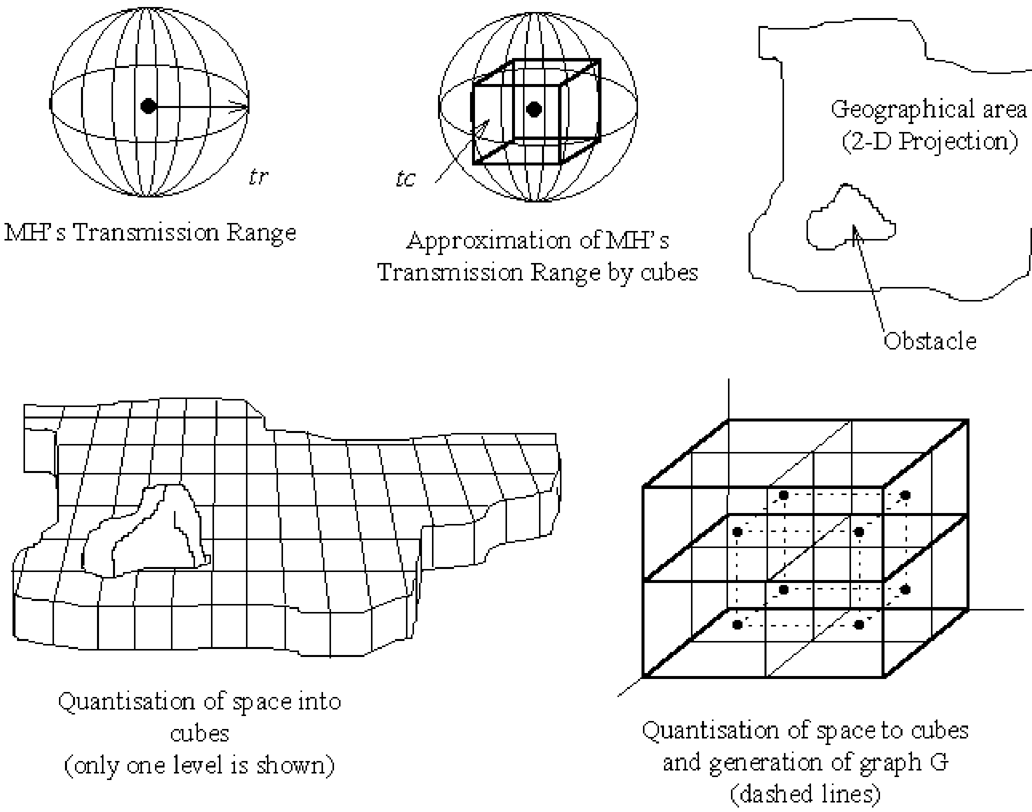

Figure 2: The construction of G when tc is a cube

for the case of *anonymous* ad-hoc mobile networks. Finally we assume that the mobile hosts are able to measure the distance that they cover when they move.

3.5 The protocol

The proposed protocol executed by the mobile hosts is presented in Figure 3. This protocol is *Non-Compulsory.* The hosts should know in advance (e.g. from hardware) the type and the dimensions of the polyhedron that is used for the quantization of S in order to be able to determine whether they have covered enough distance to reach a new vertex of G.

Each mobile host keeps a local counter which counts the other mobile hosts that it has met. At the beginning this counter is equal to one (the mobile host knows the existence of itself only). Two hosts can meet at any point of S but they transmit information related to the protocol only when they reach a new vertex (i.e. when they enter into a new polyhedron). In this manner we reduce the number of broadcasts needed by the protocol and therefore the battery power consumed for message transmissions. When two or more mobile hosts meet on a vertex u of G they exchange their identities. The winner is the one with the higher identity. The winner receives the counter of the loser and adds this to its local counter. It continues to participate in the protocol execution (it remains in state *participate*). The mobile host that lost, changes its state into *inactive* and no longer responds to messages concerning the protocol execution.

The protocol uses messages of size $O(log\ m)$ bits since the only information carried by each

```
boolean my_state=participate; the state of the mobile host regarding the protocol (participate or
inactive)
int my_id; the identity of the mobile host
int counter=1; the local counter of the host

on arriving at a new vertex u of G
if (my_state ≠ inactive) then
 begin
   broadcast < host, my_id >;
   on receive < host, i > decide_on(i);
  end

on being on a vertex u and receive < host, i >
if (my_state ≠ inactive) then
 begin
   broadcast < host, my_id >;
   decide_on(i);
  end

procedure decide_on(id)
  if (id > my_id) then
   begin
     broadcast < my_counter, counter >;
     my_state=inactive;
   end
  else
    begin
     receive < my_counter, k >;
     counter=counter+k;
    end;
```

Figure 3: The counting protocol executed by the mobile hosts in the ad hoc network

message is a counter. If the given bandwidth of radio communication enables the transmission of messages of size $O(m \log m)$ bits, the hosts may also exchange lists of identities. Each mobile host may keep a local list of mobile hosts that it has defeated (initially this list contains only itself). When two hosts meet, the winner concatenates its local list with the list transmitted by the loser. In this way, the final winner will know not only the size of the mobile network but also the identities of the hosts that move inside S.

3.6 The correctness of the protocol

As can be seen from the previous section, in order for the protocol to be executed correctly the mobile hosts are required to meet and exchange information. If this is not the case, the protocol may never elect a unique leader. Another observation is that the protocol does not include any type of termination detection mechanism. Note that these weaknesses are common for all Non-Compulsory protocols. For example, suppose that the hosts are spread in different remote areas of S in such a way that communication among them is impossible. If they do not move beyond these areas, there is no way to execute globally any protocol and provide termination detection mechanisms. In a later section we present a *Las Vegas* variation of this simple protocol and a variation which includes termination detection if the size of space S is known to the mobile hosts.

In this section we assume that the hosts move in such a way that the total number of mobile hosts that participate in the protocol execution decreases in time and there exists a time instance t where only one mobile host is still in state *participate* (the one with the highest identity). This mobile host is the final winner and the size of the mobile network is contained in its *counter* variable.

A mobile host transfers its knowledge about the network size whenever it meets another host and loses (if its identity is smaller than the winner's identity). This is done by transmitting its counter to the winner. This happens only once, since after that, the host is *inactive* and no longer responds to messages concerning the protocol. Thus, a host cannot be counted twice. The *counter* variable of a mobile host contains its current knowledge about the size of the network (i.e. the sum of the counters of all other hosts it has met plus itself).

Definition 3.1 *We define the relation* $\xrightarrow{l}$ *as follows: If* m_1 *and* m_2 *are two mobile hosts then* $m_1 \xrightarrow{l} m_2$ *if at some time instance* t m_1 *was defeated by* m_2 *and* m_2 *added the value of the counter of* m_1 *to its counter.*

Fact 3.1 *For any sequence* $m_1 \xrightarrow{l} m_2 \xrightarrow{l} m_3 \xrightarrow{l} \ldots m_{k-1} \xrightarrow{l} m_k$ *the counter of* m_k *contains the sum of the counters of* $m_1, m_2, \ldots, m_{k-1}$.

Fact 3.2 *There exists no sequence* $m_1 \xrightarrow{l} m_2 \xrightarrow{l} m_3 \xrightarrow{l} \ldots m_{k-1} \xrightarrow{l} m_k \xrightarrow{l} m_1$.

Fact 3.3 *If at some time instance* t*,* m_k *is the only mobile host in state "participate" then for any other mobile host* m_i*, there exists a mobile host* m_j *for which* $m_i \xrightarrow{l} m_j$.

Lemma 3.1 *If at some time instance* t*,* m_k *is the only mobile host in state "participate" then the counter of* m_k *contains the size of the mobile network.*

Proof: From Fact 3.3 observe that at time t the $\xrightarrow{l}$ relation forms a graph $G_l(V_l, E_l)$ as follows: A vertex $m_i \in V_l$ represents a mobile host. An edge $(m_i, m_j) \in E_l$ if $m_i \xrightarrow{l} m_j$. Fact 3.3 also implies that for any mobile host m_i, $m_i \in V_l$. From Fact 3.2, G_l is actually a tree rooted at m_k. Fact 3.1 yields that the counter of each "parent" in the tree contains the sum of the counters of all its "children" and thus ,finally, the counter of the root m_k contains the size of the mobile network. □

3.7 Protocol analysis using the random walk assumption

3.7.1 Preliminaries

The Non-Compulsory nature of the protocol makes the analysis of the protocol performance impossible. In order to be able to carry out this analysis **we make the following assumption:** We assume that **the movement of each mobile host is a random walk on the graph** G. We also assume that all random walks are concurrent and that there is a global time t, not necessarily known to the hosts. We are then able to apply powerful mathematical tools that allow us to analyze the protocol's performance. The random walk of a mobile host on G induces a continuous time Markov chain M_G as follows: The states of M_G are the vertices of G. Let s_t denote the state of M_G at time t. Given that $s_t = u$, $u \in V$, the probability that $s_{t+dt} = v$, $v \in V$, is $p(u,v)dt$ where

$$p(u,v) = \begin{cases} \frac{1}{d(u)} & if\ (u,v) \in E \\ 0 & otherwise \end{cases}$$

where $d(u)$ is the degree of vertex u.

We denote by $Pr_i[\,]$ and $E_i[\,]$ the probabilities and expectations for the chain started at state i (the random walk of the mobile host has begun from vertex i) and by $Pr_\mu[\,]$ the probability for the chain started at time 0 from any vertex with distribution μ (the initial distribution of the Markov chain). Let

$$T_i = min\{t \geq 0 : s_t = i\}$$

be the first hitting time on state i (the first time that the mobile host visits vertex i). We denote by $M_{i,j}$ the first meeting time of two mobile hosts that started their walk from i and j respectively. Finally, let C_f be the time that only one mobile host is still in state *participate* (and thus its *counter* variable contains the size of the network).

According to our model, a broadcasted message may be received not only by hosts residing on the same node of G but also by hosts residing on neighboring nodes. The analysis focuses on the worst case scenario where a broadcasted message is received only by hosts residing on the same node (since the reception of a message by hosts residing on neighboring nodes speeds up the protocol execution). We first consider the case where $m = n$. We assume that at the beginning of the protocol execution (at time 0) there exists one mobile host on each vertex of G. We remove this assumption later.

3.7.2 A first upper bound on the execution time of the protocol

We follow the analysis of the *voter model* as described by *Aldous and Fill* in [4]. We order the vertices of G as $i_1, ..., i_n$ by giving the lower label i_1 to the vertex occupied by the mobile host which is the final winner and by assigning the other labels arbitrarily to the remaining vertices. Recall that when two hosts meet only one continues to participate in the protocol execution. According to this construction,

$$C_f \leq max_j M_{i_1,j}$$

since the time C_f is less than the maximum time required for the winner to meet any other host. Let $m^* \equiv max_{i,j} E_i[M_{i,j}]$.

Lemma 3.2 $Pr[M_{i,j} > t] \leq exp(-\lfloor \frac{t}{em^*} \rfloor)$

Proof: Starting from time 0, divide the time axis into equal time intervals of size s. For any initial distribution μ, any $s > 0$ and any integer $k \geq 1$,

$$Pr_\mu[M_{i,j} > ks \mid M_{i,j} > (k-1)s] = Pr_\theta[M_{i,j} > s]$$

for some initial distribution θ (due to the memoryless property of the Markov chain). By definition

$$Pr_\theta[M_{i,j} > s] \leq max_i Pr_i[M_{i,j} > s]$$

By applying the Markov inequality $Pr[x > t] \leq \frac{E[x]}{t}$,

$$max_i Pr_i[M_{i,j} > s] \leq \frac{max_{i,j} E_i[M_{i,j}]}{s}$$

and finally

$$max_i Pr_i[M_{i,j} > s] \leq \frac{m^*}{s}$$

By induction on k,

$$Pr_\mu[M_{i,j} > \lambda s] \leq (\frac{m^*}{s})^\lambda$$

implying (by substituting $\lambda s = t$) that

$$Pr_\mu[M_{i,j} > t] \leq (\frac{m^*}{s})^{\lfloor \frac{t}{s} \rfloor}$$

.

Using the above observations, the tail of the distribution can be bounded in the following way:

$$\begin{aligned}
Pr_\mu[M_{i,j} > t] &\leq \left(\frac{m^*}{s}\right)^{\lfloor \frac{t}{s} \rfloor} \\
&\leq e^{\lfloor \frac{t}{s} \rfloor \log \frac{m^*}{s}} \\
&\leq e^{\lfloor \frac{t}{m^* e} \rfloor \log \frac{1}{e}} \quad (by\ substituting\ s = m^* e) \\
&\leq e^{-(loge)\lfloor \frac{t}{m^* e} \rfloor} \\
&\leq e^{\lfloor -\frac{t}{m^* e} \rfloor}, \quad 0 < t < \infty
\end{aligned}$$

□

Theorem 1 *The expected value of* C_f, $E[C_f]$, *is bounded according to the inequality* $E[C_f] \leq e(logn + 2)max_{i,j}E_i[T_j]$.

Proof: Recall that

$$C_f \leq max_j M_{i_1,j} \quad (1)$$

and

$$Pr[max_j M_{i_1,j} > t] \leq \sum_j Pr[M_{i_1,j} > t] \quad (2)$$

By definition,

$$\begin{aligned}
E[C_f] &= \int_0^\infty Pr[C_f > t]dt \\
&\leq \int_0^\infty Pr[max_j M_{i_1,j} > t]dt \quad (from \quad (1)) \\
&\leq \int_0^\infty min(1, \sum_j Pr[M_{i_1,j} > t])dt \quad (from \quad (2)) \\
&\leq \int_0^\infty min(1, n\ e^{-\frac{t}{em^*}})dt \quad (from \quad Lemma\ 3.2) \\
&\leq \int_0^\infty min(1, n\ e^{-\frac{t}{em^*}+1})dt \\
&\leq \int_0^\infty min(1, n\ e\ e^{-\frac{t}{em^*}})dt \quad (3)
\end{aligned}$$

By applying the formula

$$\int_0^\infty min(1, Ae^{-\alpha t})dt \equiv \alpha^{-1}(1 + logA), A \geq 1$$

(3) yields

$$\begin{aligned}
\int_0^\infty min(1, n\ e\ e^{-\frac{t}{em^*}})dt &= em^*(2 + logn) \\
&= e\ max_{i,j}E_i[M_{i,j}](2 + logn)
\end{aligned}$$

Since $max_{i,j}E_i[M_{i,j}] \leq max_{i,j}E_i[T_j]$,(see [4]), we finally conclude that

$$E[C_f] \leq e(logn + 2)max_{i,j}E_i[T_j]$$

□

Corollary 3.1 *For the expected value of C_f, $E[C_f]$, $E[C_f] \leq e(logn + 2)2\epsilon$ where ϵ is the number of edges of the graph G.*

Proof: From [18] it is known that $max_{i,j}(E_i[T_j] + E_j[T_i]) \leq 2\epsilon$ where $E_i[T_j] + E_j[T_i]$ is the commute time between i and j (the expected time for a process to start from i and return to i after at least one visit to j) and $\epsilon = |E|$. Thus,

$$\begin{aligned} E[C_f] &\leq e(logn + 2)max_{i,j}E_i[T_j] \\ &\leq e(logn + 2)max_{i,j}(E_i[T_j] + E_j[T_i]) \\ &\leq e(logn + 2)2\epsilon \end{aligned}$$

□

3.7.3 A tighter upper bound on the execution time of the protocol

A key observation that leads to a tighter bound on the execution time of the protocol is that the basic inequality of the proof of Theorem 1 , $C_f \leq max_j M_{i_1,j}$, is quite rough. Specifically, this inequality corresponds to the extreme case where the winner host meets all other hosts one at a time, while in the average case, the processes exclude each other in parallel and thus accelerate the counting procedure.

Definition 3.2 *Let $X, X_1, X_2 \ldots X_m$ be independent random variables on the same distribution $F(x)$, probability density function $f(x)$ and mean value $E[f(x)]$. Let $F_{min(m)}(x)$ denote the distribution of the minimum of these variables, $f_{min(m)}(x)$ its probability density function and $E\left[f_{nim(m)}(x)\right]$ its mean value.*

Lemma 3.3 *There exists a constant $m_0 < m$ for which*

$$\forall\ m > m_0,\ E\left[f_{min(m)}(x)\right] \leq \frac{1}{m}E[f(x)]$$

Proof:

$$\begin{aligned} F_{min(m)}(x) &= Pr[minX_i < x, \quad 0 < i \leq m] \\ &= 1 - Pr[minX_i > x] \\ &= 1 - Pr[\forall i\ \ X_i > x] \\ &= 1 - Pr[X > x]^m \quad (since\ X_i\ are\ indepenrent) \\ &= 1 - (1 - Pr[X < x])^m \\ &= 1 - (1 - F(x))^m \end{aligned}$$

Derivation of both sides of the equation results in

$$\begin{aligned} \left[F_{min(m)}(x)\right]' &= [1 - (1 - F(x))^m]' \ \Rightarrow \\ f_{min(m)}(x) &= m\ f(x)\ (1 - F(x))^{m-1} \end{aligned}$$

By definition, $E[f(x)] = \int_0^\infty xf(x)dx$. Therefore,

$$\begin{aligned} E[f_{min(m)}(x)] &= \int_0^\infty x f_{min(m)}(x)dx \\ &= \int_0^\infty xf(x)m(1-F(x))^{m-1} \end{aligned}$$

Since $E[f_{min(m)}(x)] \neq \infty$ there exists a real λ for which

$$\begin{aligned} E[f_{min(m)}(x)] &= 2\int_\lambda^\infty xf(x)\, m(1-F(x))^{m-1}dx \\ &\leq 2\int_\lambda^\infty xf(x)\, m(1-F(\lambda))^{m-1}dx \quad (\textit{since } F(x) \textit{ is monotonically increasing}) \\ &= 2m(1-F(\lambda))^{m-1}\int_\lambda^\infty xf(x)dx \\ &\leq 2m(1-F(\lambda))^{m-1}E[f(x)] \end{aligned}$$

In order to get $E\left[f_{min(m)}(x)\right] \leq \frac{1}{m}E[f(x)]$ it suffices to have $(1-F(\lambda))^{m-1} \leq \frac{1}{2m^2}$. This holds for every $m \geq m_0$, where m_0 is a constant depending on $F(\lambda)$, since $F(x) \leq 1$ always holds.
□

Lemma 3.4 *Let S_1 and S_2 be two (not necessarily) independent random variables on the same distribution and $S = S_1 + S_2$. Then, $E[S] = 2E[S_1] = 2E[S_2]$.*

Proof: By linearity of expectation. □

Definition 3.3 *Let $C_f(k)$ be the time required by the processes that still participate in the protocol to have defeated k other processes. Obviously, $C_f(m) = C_f$. Furthermore, let $M_{i,j}(k)$ be the time required for the first interaction of two processes that still participate in the protocol, when k processes still participate in the protocol.*

According to Lemma 3.4, each time two mobile processes meet, the counter of the winner process is doubled on the average. This yields that the final winner is the product of the interaction of two mobile processes that, on the average, have counted (and excluded) $m/2$ mobile processes each. The average duration of this final phase is $E[M_{i,j}]$. Furthermore, each one of the last two processes is a product of the interaction of two processes, each of which has counted $m/4$ processes on the average, while the duration of this phase is $E[M_{i,j}(2)]$ on the average.

Theorem 2 *$E[C_f] \leq (2 + log(m_0))2\epsilon$, where m_0 is a constant.*

Proof:

$$\begin{aligned} E[C_f] = E[C_f(m)] &= E[M_{i,j}(2)] + E\left[C_f\left(\frac{m}{2}\right)\right] \\ &= E[M_{i,j}(2)] + E[M_{i,j}(4)] + E\left[C_f\left(\frac{m}{4}\right)\right] \\ &= E[M_{i,j}(2)] + E[M_{i,j}(4)] + \ldots + E[M_{i,j}(m)] \\ &= \sum_{k \leq log(m_0)} E\left[M_{i,j}(2^k)\right] + \sum_{log(m_0) < k < log(m)} E\left[M_{i,j}(2^k)\right] \end{aligned}$$

$$
\begin{aligned}
&\leq \; log(m_0)E[M_{i,j}] + \sum_{log(m_0)<k<log(m)} \frac{2^k}{m} E[M_{i,j}] \quad (using\ Lemma\ 3.3) \\
&\leq \; log(m_0)E[M_{i,j}] + E[M_{i,j}] \sum_{log(m_0)<k<log(m)} \frac{2^k}{m} \\
&\leq \; log(m_0)E[M_{i,j}] + 2\ E[M_{i,j}] \\
&\leq \; (2 + log(m_0))E[M_{i,j}] \\
&\leq \; (2 + log(m_0))2\epsilon
\end{aligned}
$$

□

3.7.4 The case of $m \neq n$

Up to this point we considered that the number of mobile hosts equal to the number of nodes of the underlying graph G. In this section we will prove the rather counter-intuitive fact that the execution time of the protocol is not affected seriously by the number of mobile hosts or their initial distribution on G.

The case of $m < n$

Let us first consider the case where the initial placement of the mobile hosts is a configuration chosen uniformly and at random out of all configurations where the mobile hosts do not overlap, occupying exactly m vertices. It is easy to observe that this configuration can be regarded as an instance of the execution of the protocol with n initial mobile hosts. Therefore, on the average, the execution time of the algorithm in this case is less than C_f, as described in the previous sections. Furthermore, by following the arguments of Section 3.6, only the first stages of the protocol execution are omitted. These stages have little contribution to the total execution time ($O\left(\frac{1}{n}E[M_{i,j}]\right)$) and thus, the protocol execution time is not seriously affected. The extreme case of $m = 2$ can provide a clear insight on the previous argument. In this case, the protocol execution time is on the average $E[M_{i,j}]$, which differs from the case of $m = n$ only by a constant factor. The chosen initial configurations can be shown to be the worst case, since in all other cases the overlapping mobile processes will exclude each other in one step, leading to a configuration with even less participating mobile hosts.

The case of $m > n$

In this case, the mobile hosts that reside on the same node exclude each other in the first step, leading thus to a sub-case of the previous paragraph.

A more formal approach for the case $m < n$

Let $\mathcal{S}$ be the set of identities of the initially participating mobile hosts and

$$\mathcal{L}_i(t) = \{m \in \mathcal{S} : m\ is\ on\ node\ i\ at\ time\ t\}$$

Define an *instance* of the protocol execution at time t as

$$\mathcal{I}(t) = \{\mathcal{L}_1(t), \mathcal{L}_2(t), \ldots, \mathcal{L}_n(t)\}$$

Furthermore, define $\delta = \{r_1, r_2\}$ to be the set of rules according to which a transition from instance $\mathcal{I}(t)$ to $\mathcal{I}(t')$, $t \leq t'$ occurs.

$$
\begin{aligned}
\mathcal{I}(t) \xrightarrow{r_1} \mathcal{I}(t') \quad &\Longleftrightarrow \quad \exists\ \mathcal{L}_i(t)\ :\ \mathcal{L}_i(t') \subset \mathcal{L}_i(t)\ and\ |\mathcal{L}_i(t')| = 1 \\
\mathcal{I}(t) \xrightarrow{r_2} \mathcal{I}(t') \quad &\Longleftrightarrow \quad \exists\ \mathcal{L}_i(t)\ and\ \mathcal{L}_i(t')\ :\ (i,j) \in E\ and\ \exists\ m\ : m \in \mathcal{L}_i(t)\ and\ m \in \mathcal{L}_i(t')
\end{aligned}
$$

Finally, define the *history* $\mathcal{H}_\mu$ of a protocol execution with initial mobile host distribution μ, the sequence of instances $\mathcal{I}(0)$ to $\mathcal{I}(C_f)$ of the protocol execution, according to δ.

Let $\mathcal{H}_\mu$ be the history of the protocol execution with initial mobile host distribution

$$\mu \; : \; |\mathcal{S}| = m < n$$

consisting of instances $\mathcal{I}$. Furthermore, let $\mathcal{H}_\theta$ be the history of the protocol execution with initial mobile host distribution $\theta \; : \; |\mathcal{S}| = n$, consisting of instances $\mathcal{I}'$. Then

$$\exists \, \theta \; : \; \mathcal{I}(0) = \mathcal{I}'(t), t > 0$$

A more "practical" approach for the case $m < n$

Suppose that m mobile hosts, $m < n$, are placed on m vertices of G with initial distribution μ. This case can be reduced to the case of $m = n$ by using the following construction: We consider $m - n$ new "virtual" mobile hosts. These hosts are identified with $vm_1 = -1, vm_2 = -2, \ldots, vm_{m-n} = -(m-n)$ and are placed on the remaining $m - n$ vertices of G with an initial distribution μ'. These "virtual" hosts are also participating in the protocol. Obviously they do not affect the execution and the final winner since they lose and become inactive the first time they meet one of the m hosts. According to Theorem 2, $E[C_f]$ is at most equal to $(2 + log(m_0))2\epsilon$ in this case too.

3.7.5 Discussion of the result

The results of sections 3.7.3 and 3.7.4 are imlpying implies that the execution time of the protocol is linear on the average with respect to the ratio between the volume of space S, $V(S)$, and the space occupied by the transmission range of a mobile host $V(tr)$. This is a direct consequence of the construction of G (see Section 3.2) since $\epsilon = O(n)$ and $n = O(\frac{V(S)}{V(tr)})$. This result holds regardless of the choice of the polyhedron that is used for the quantization of S and the initial positions of the mobile hosts on G. It also leads to the (rather intuitive) fact that powerful transmitters can speed up the protocol's execution.

3.8 A variation of the basic protocol with termination detection

The calculation of the average protocol execution time allows us to construct a new protocol that provides a unique leader with a given probability of success. We consider that the execution of the protocol fails if the mobile hosts stop the execution of the protocol without having a unique leader. Note that in this case there is no mobile host whose counter contains the correct network size. Let p_f denote the protocol failure probability ($p_f \equiv Pr[C_f > t]$).

Lemma 3.5 *For any given failure probability p_f, running the protocol for time $t = \frac{c\epsilon}{p_f}$ where c is a constant, suffices to ensure that the result will be correct with failure probability at most p_f.*

Proof: By applying the Markov inequality for the random variable C_f we have

$$\begin{aligned} Pr[C_f > t] &< \frac{E[C_f]}{t} \Rightarrow \\ t &\leq \frac{E[C_f]}{Pr[C_f > t]} \\ &\leq \frac{c\epsilon}{p_f} \end{aligned}$$

□

Lemma 3.5 implies that the mobile hosts may decide on the duration of the protocol execution only if they know the total number of edges ϵ of the underlying graph G. Recall that in G, the number of edges ϵ is linear to the number of vertices n (in the case of a cube-based grid ϵ is related to n by the inequality $\epsilon \leq 6n$). Intuitively and by taking into consideration the discussion in Section 3.2, a mobile host must have a feeling of the available space in order to be able to determine whether the protocol has terminated or not. Consider, for example, a mobile host moving in the area of Manhattan, that at time t remains in state *participate* having defeated k other mobile hosts. Its estimation of being a unique winner would be different if the total area in question is Manhattan, New York or North America.

Finally, note that in the case of time constraints (i.e. the mobile hosts have to execute the protocol for a specific time period), Lemma 3.5 may be applied by the mobile hosts to calculate the probability of success for a given run time t, in order to find out how much they can rely on the result.

3.9 Las Vegas Compulsory protocols for leader election and counting in an ad hoc network without sense of orientation

Since a major weakness of the basic Non-Compulsory protocol is that it may never elect a unique leader, the previous results lead to the specification of a *Las Vegas Compulsory* protocol that elects a unique leader in the ad hoc mobile network in time linear to the size of S with a given probability of success. This protocol behaves as the one described in Section 3.5, but in addition, it forces the mobile hosts to perform a random walk on G. The protocol includes also termination detection as presented in the previous section. The host can decide on a failure probability p_f and use it in order to find the required time t to run the protocol. If the host after time t is still in state participate, it is (with probability at least $1 - p_f$) the unique leader. On the other hand, if it is in state inactive it knows that a unique leader has been probably elected. The proposed Las Vegas Compulsory protocol can be applied even in the case where the mobile hosts have no sense of orientation. We further conjecture that this algorithm is **optimal in time** in the case of mobile hosts with no sense of orientation. A straightforward extension of this protocol is a Compulsory counting protocol. After time t, the leader initiates a flooding phase as follows: It starts moving in a random walk. Every time it meets another mobile host it transmits its counter. Each mobile host that has been informed about the counter behaves in exactly the same way. The duration of this phase is obviously bounded by $(2 + log(m_0))2\epsilon$ too.

3.10 Anonymous networks

The protocols described in the previous section can also be applied when the mobile network is *anonymous*, i.e. the mobile hosts do not have distinct identities. In the latter case the proposed variation leads to *Las Vegas Compulsory* protocols for *leader election* and *counting in anonymous ad hoc networks without sense of orientation.* The proposed variation is as follows. Whenever a mobile host meets another host on a vertex of G they choose random identities over a predefined set and then exchange them. If there is a conflict, they repeat the procedure. The winner is finally the one that selected the higher identity. We remark that the time required for two hosts to select unique identities when they meet (regardless of the number of conflicts) is negligible compared to the time required for a mobile host to move from one vertex of G to another and thus does not affect the protocol execution time.

3.11 Comparison theorems and the steady-state of the random motion

Let us note that a related "many objects" motion process is the so called Symmetric Exclusion (SE) process. Let G be an undirected graph of n vertices. To start, r unlabeled particles are placed in an initial configuration, $1 \leq r \leq n$. At each step, a particle is chosen at random. Then one of the neighboring sites of this particle is chosen at random. If the neighboring site is unoccupied, the chosen particle moves there; if the neighboring site is occupied the system stays as it was. This is a reversible Markov chain on the r-sets of $1, 2, \ldots, n$. The reader can easily recognize that this is the discrete analog of our proposed model of motions of the mobile hosts, provided that the "time" in the concurrent movement case is replaced by rounds of steps (i.e. a step sequence in which all particles move at least once). Let us denote our continuous model by RM. Fill ([11]) gave bounds on the second eigenvalue of SE (but for labeled particles) on the finite circle of n vertices.

Diaconis and Saloff-Coste ([10]) study the chain SE by comparison with a second Markov chain on r-sets that proceeds by picking an unoccupied site at random (not necessarily a neighboring site) and moving the particle to the unoccupied site. This second process which corresponds to mobile hosts moving "very fast" or e.g. by teleportation (or other means), is a well-studied chain (the Bernoulli-Laplace model for diffusion). Its eigenvalues are known. We denote this process by TP.

The method of comparison of [10] is based on the minimax characterization of eigenvalues based on the Dirichlet forms of the chains, viewed as a self-adjoint operator because of reversibility (see e.g. Horn and Johnson, [15]).

Diaconis and Saloff-Coste developed a method for comparison of the Dirilecht forms, which provides tight upper and lower bounds on the eigenvalues of SE (and thus on the eigenvalues of RM) by using the eigenvalues of TP and structural (path) properties of the graph G. The method gives bounds for the mixing properties of RM on any particular network G. A crude but universal estimate for the second largest eigenvalue β_1 of SE (the first is unity) is

$$\beta_1 \leq 1 - 1/rn^2 d_0$$

where d_0 is the maximum degree of the graph. This bound is easily obtained by the above method.

4 Simulation Results

The basic protocol was implemented and tested using the Distributed Systems Platform (DSP) [1](see [25]). The DSP tool allows the specification, implementation and testing of distributed algorithms for fixed and mobile networks. The simulations focused on the case of $m = n$, with the mobile processes performing random walks on the underlying graph. An example of the simulation results for 10 runs and three different topologies are presented in Figure 2.

Topology	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Mean**	**Variance**
Regular 30	99	249	44	256	160	60	72	53	183	97	126.9	6255.211
Irregular 30	902	217	333	171	96	212	57	250	162	755	315.4	80211.822
Regular 15	22	56	19	16	39	56	45	20	42	32	34.7	229.566

Figure 4: Simulation results: Protocol execution time for 10 runs on three different topologies

In general, the simulations confirmed the theoretical results. The number of participating hosts decreased exponentially in time. In fact, on the average, one third of the mobile hosts were defeated immediately after the beginning of the protocol. The protocol execution time was on the average less than the theoretical result, because of the fact that the analysis holds for every underlying graph G, while the quantization of the space leads to regular graphs, for which it is known that the random

[1]The DSP tool was designed and implemented during the EU ESPRIT LRT ALCOM-IT project

walks mix faster. As expected, the average protocol execution time increased for more irregular topologies as compared to the execution time for regular topologies of the same size. Finally, the protocol execution time increased linear to the number of nodes for regular topologies of the same degree.

5 Conclusions

Our present work proposes a way to deal with fundamental aspects of Mobile Distributed Computing while abstracting away details of geometric nature. The random walks assumption, viewed either as a crude approximation of many hosts or as a compulsory motion element of the protocols, allowed us to get powerful analytic results. Our future work will concentrate on modeling issues that will allow an analytic or combinatorial treatment of the solvability and perhaps the discovery of efficient protocols for other fundamental problems such as coordination, termination detection or failures detection.

References

[1] "Impact of mobility on Distributed Computations", A. Acharya, B. R. Badrinath, T. Imielinski, Operating Systems Review, April 1993.

[2] "Structuring distributed algorithms for Mobile Hosts", A. Acharya, B. R. Badrinath, T. Imielinski, In Proc. of 14th International Conference on Distributed Computing Systems, June 1994.

[3] "More on the Power of Random Walks: Uniform Self-Stabilizing Rand omized Algorithms", E. Anagnostou and R. El-Yaniv, In Proc. of 5th International Workshop on Distributed Algorithms WDAG '91, Delphi, Greece, 1991.

[4] "Reversible Markov Chains and Random Walks on Graphs", D. Aldous, J. Fill, unpublished manuscript, http://stat-www.berkeley.edu/users/aldous/book .html

[5] "Concurrent online tracking of mobile users", B. Awerbuch, D. Peleg, Journal of the ACM, 42(5), September 1995.

[6] "Approximating the Size of a Dynamically Growing Asynchronous Distributed Network", B. Awerbuch, S. A. Plotkin, MIT.

[7] "Echo algorithms: Depth parallel operations on general graphs", E. J-H. Chang, IEEE Transactions on Software Engineering, SE-8, 1982.

[8] "IP-based protocols for mobile internetworking", D. Duchamp, G. Q. Maquire, J. Ioannidis, In Proc. of ACM SIGCOM, September 1991.

[9] "The Arrow Distributed Directory Protocol", M. Demmer, M. P. Herlihy, In Proc. of 12th International Symposium on Distributed Computing, DISC '98, September 1998.

[10] "Comparison Theorems for reversible Markov Chains", P. Diaconis, L. Saloff-Coste, The Annals of Applied Probability, Vol. 3, No. 3, 1993.

[11] "Eigenvalue bounds on convergence to stationarity for nonvreversible Markov chains with an application to the exclusion processes", J. Fill, Ann. Appl. Probab., 1, 62-87, 1991.

[12] "Frequency Assignment in Mobile and Radio Networks", D. Fotakis, G. Pantziou, G. Pentaris and P. Spirakis, to appear in the special issue of the American Mathematical Society DIMACS series with the proceedings of the workshop on "Networks in Distributed Computing".

[13] "A Hamiltonian Approach to the Assignment on Non-Reusable Frequencies", D. Fotakis, P. Spirakis, In Proc. of 18th Symposium on Foundations of S/W Technology and Theoretical Computer Science, 1998.

[14] "Mobile Computing", T. Imielinski, H. F. Korth, Kluwer Academic Publishers, 1996.

[15] "Matrix Analysis", R. Horn and C. Johnson, Cambridge University Press, 1985.

[16] "Token Management Schemes and Random Walks yield Self Stabilizing Mutual Exclusion", A. Israeli and M. Jalfon, In Proc. of the 9th ACM Symposium on Principles of Distributed Computing, PODC 1990.

[17] "Distributed Systems towards a formal approach", G. Le Lann, IFIP Congress, 1977.

[18] "Randomized algorithms", R. Motwani, P. Raghavan, Cambridge University Press, 1996.

[19] "Introduction to Distributed Algorithms", G. Tel, Cambridge University Press, 1994.

[20] "Distributed Dynamic Channel Allocation for Mobile Computing", R. Prakash, N. Shivaratri, M. Sighal, In Proc. of ACM Symposium on Principles of Distributed Computing, PODC 1995.

[21] "Competitive Call Control in Mobile Networks", G. Pantziou, G. Pentaris, P. Spirakis, In Proc. of 8th International Symposium on Algorithms and Computation, ISAAC 97, December 1997.

[22] "Fundamental Challenges in Mobile Computing", M. Satyanarayanan, Invited talk, In Proc. of ACM Symposium on Principles of Distributed Computing PODC 1996.

[23] "Distributed Pursuit Evasion: Some aspects of Privacy and Security in Distributed Computing", P. Spirakis and B. Tampakas, short presentation, In Proc. of ACM Symposium on Principles of Distributed Computing, PODC 1994.

[24] "Distributed Protocols Against Mobile Eavesdroppers", P. Spiraki s, B. Tampakas and H. Antonopoulou, In Proc. of 9th International Workshop on Distributed Algorithms WDAG '95, France, 1995.

[25] DSP Web Site, http://helios.cti.gr/alcom-it/dsp