Lightweight Replication Middleware for Data and Service Components in Dense MANETs

Paolo Bellavista, Antonio Corradi, Eugenio Magistretti
Dip. Elettronica, Informatica e Sistemistica - Università di Bologna
Viale Risorgimento, 2 - 40136 Bologna - ITALY
Phone: +39-051-2093001; Fax: +39-051-2093073
{pbellavista, acorradi, emagistretti}@deis.unibo.it

Abstract

The increasing diffusion of wireless-enabled portable devices is pushing towards service provisioning over dense Mobile Ad-hoc NETworks (MANETs), i.e., limited spatial regions, such as shopping malls, railway stations and airports, where a high number of mobile wireless peers autonomously cooperate, without the need for statically deployed network infrastructures. Dense MANET deployment scenarios can take advantage of high node population to replicate common-interest resources to increase their availability, by overcoming the unpredictable node exit from the dense region. The paper proposes a lightweight middleware, called REDMAN, to manage, retrieve and disseminate replicas of data/service components made available by cooperating nodes in a dense MANET. In particular, the paper focuses on the REDMAN original solutions both to determine the nodes belonging to dense MANETs without exploiting any positioning system and to dynamically elect a suitable replica manager node in charge of enforcing the desired resource replication degree in a lazy consistent way. Experimental results show that REDMAN solutions are lightweight and effective in dense MANET scenarios with almost constant node density and even high node mobility.

1. Introduction

The mass market of wireless devices suggests novel service deployment scenarios where there are no constraints on device mobility and distributed applications are the result of impromptu collaborations among wireless peers. In these scenarios, not only wireless nodes should have location-aware access to distributed resources without requiring static knowledge about their execution environment, but also resources should be permanently available, not depending on node movement, disconnection, and battery shortage [1].

Some research activities are investigating MANET-specific solutions to bind/rebind to newly discovered distributed resources, thus enabling wireless clients to automatically redirect requests to service components also by considering their mutual location [2, 3]. On the contrary, the idea of increasing the availability of MANET applications by replicating data and service components close to their clients is still at its very beginning. Only few state-of-the-art proposals have started to face the very challenging issue of resource replication in mobile environments, with the goal of increasing access probability and effectiveness [4]. Most investigations have recently addressed the issue of information availability in cellular and infrastructure-mode IEEE 802.11 networks. However, in that context, it is possible to exploit the fixed part of the network infrastructure, e.g., to store and replicate personal data at highly available servers on wired stable links. The most critical issue in infrastructure-based scenarios is to properly manage user disconnections during update operations on shared data, possibly by automatically handling the reconciliation of multiple modified copies [5, 6].

Due to the complete lack of a static support infrastructure, the effective replication of data and service components in MANET dynamic environments is a hard challenge, which requires re-thinking and significantly modifying traditional replication approaches. So far, the research has mainly focused on replication to ensure data availability in case of network partitions. Most proposals assume that wireless nodes are aware of their physical position, e.g., by imposing the constraint of hosting Global Positioning System hardware at any participant [7]. Other research activities aim at answering strict requirements about replica synchronization, and therefore impose a heavy overhead, in

terms of both network traffic and requested time to ensure the consistency of all replicas [8].

We claim that, due to connectivity issues, e.g., high loss rates and frequent network partitioning, the management of data/service component replicas is a very hard task to perform in an effective and lightweight way when dealing with general-purpose MANETs and with strict consistency requirements. Therefore, we focus on a specific deployment scenario of increasing relevance for the service provisioning market, called dense MANET in the following. We use the term *dense MANET* to indicate a MANET that:

- includes a large number of wireless devices located in a relatively small area at the same time, e.g., as it will probably happen in the near future in shopping malls, airport waiting rooms, and university campuses;
- has a node density (the average number of wireless nodes at single-hop distance from any dense MANET participant) that is almost invariant during long time intervals.

In particular, the paper proposes a middleware solution, called REDMAN (**RE**plication in **D**ense **MAN**ETs), that transparently disseminates, manages, and retrieves replicas of common interest resources among cooperating nodes in the dense MANET. REDMAN has the main goal of improving the availability of data and service components, by exploiting lightweight solutions specifically suitable for the characteristics of dense MANETs. The primary idea is of maintaining, within the dense MANET, a fixed replication degree for the needed resources, independently of possible (and unpredictable) exits of replica-hosting nodes from the dense region. REDMAN addresses the primary challenging issues of dense MANETs, i.e., the determination of nodes belonging to the dense region, the sensing of nodes entering/exiting the dense MANET, and the dynamic election of replica managers responsible for maintaining the required resource replication degrees, in a completely decentralized way via original and lightweight protocols specifically designed for dense MANETs.

Let us point out that REDMAN takes into account client device resource constraints by choosing not to respect strict consistency requirements on resource replication degree in order to limit its overhead. In particular, REDMAN proposes lightweight lazy-consistent protocols, capable of obtaining acceptable performance while reducing the number of exchanged messages. For instance, the manager election protocol exploits a heuristic-based algorithm that does not always permit to reach the optimum, but that generally discovers quasi-optimal solutions by exploring only a restricted subset of dense MANET nodes.

REDMAN operates at the application level because several replica management decisions, such as the suitable replication degree depending on differentiated resource criticality, are typically at this abstraction layer [9]. Working at the application level also simplifies portability over heterogeneous connectivity technologies and routing protocols. In addition, REDMAN performs replica management transparently from the point of view of service developers/administrators, who only have to indicate the criticality of the shared resources involved. Moreover, REDMAN has been specifically designed for battery/memory-constrained devices (PDAs, smart phones, …), which typically cannot host positioning hardware and cannot store all needed data and service components in their local memory permanently; REDMAN simplifies the mutual interactions of limited portable devices to enable collaborative service provisioning in dense MANETs.

After briefly presenting the REDMAN architecture and goals, better described in [10], the paper focuses on two crucial issues for replica management in dense MANETs: how to determine the nodes that belong to dense regions (with no need of external positioning systems) and how to elect, in a completely decentralized way, replica managers in charge of enforcing the desired replication degree for resources of common interest. The paper reports extensive simulation results about REDMAN performance, which show the effectiveness and the limited overhead of the proposed solutions, by confirming the suitability of the approach even in conditions of high node mobility.

2. A Case Study for REDMAN: Playing a Strategy Game at the Railway Station

To practically introduce the REDMAN middleware functions in a simple case study scenario, let us consider travelers with REDMAN-enabled PDAs who are interested in playing a strategy game while in a waiting room of a railway station. Waiting rooms are usually crowded places, where travelers come and go, and the number of people co-located there is almost constant. Therefore, wireless devices in a waiting room are suitable candidates to compose a dense MANET.

Several travelers could be keen on spending the waiting time by playing videogames. However, their devices cannot have all the desired game components statically pre-installed, also because of their usual memory limitations. In addition, travelers could also be interested in playing new videogames (or new versions, new playing sceneries, …), by dynamically discovering them in the community of cooperative travelers bringing them in the waiting room and by dynami-

cally downloading those games to their devices, if allowed. Moreover, some videogames have a distributed implementation and require the continuous availability of one or more servers; the dense MANET should always include the needed running servers, independently of the unpredictable movements of the nodes hosting them. In this scenario, REDMAN enables the lightweight and transparent distribution of replicas of game components (with the desired resource-associated replication degree) to a subset of devices in the waiting room. In addition, REDMAN supports the dynamic retrieval of the needed resources and their distributed cooperation, thus providing travelers with a set of locally available games, which dynamically grows depending on the new resources that visiting travelers bring into the waiting room and decide to share with other dense MANET participants.

For instance, consider the well-known single-player Civilization strategy game [11] and let us illustrate how REDMAN can distribute replicas of Civilization game components, to provide the game availability in the dense MANET with no need of explicit and static installations, even if mobile nodes continuously enter/exit the waiting room (see the example of deployment scenario in Figure 1). Suppose that a device enters the dense MANET by carrying new game components (the Civilization server engine, the Civilization lightweight client, and some playing sceneries). If the game is considered of interest, REDMAN transparently works to guarantee, via replication, the availability of the different Civilization components in the dense MANET, independently of the unpredictable mobility of nodes hosting the component replicas. We will use the term resource delegates to indicate the nodes in the dense MANET that hold a resource of common interest (or its replica).

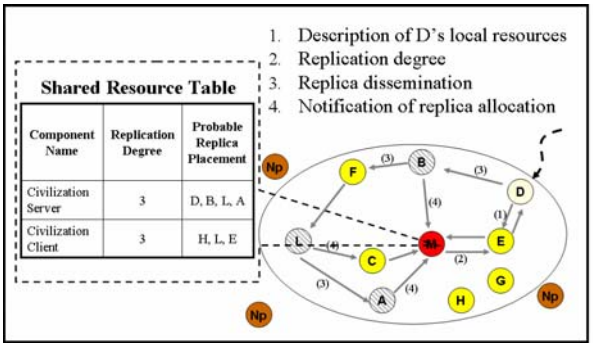

Figure 1. REDMAN-based replication of the Civilization server.

For instance, when the delegate D with the Civilization server engine enters the waiting room, the REDMAN middleware layer on it sends the descriptions of D's resources to the REDMAN replica manager M on another node (message 1). The replica manager is in charge of enforcing the needed replication degree and of maintaining information about all shared resources available in the dense MANET (Shared Resource Table - SRT). For any shared resource, the corresponding SRT entry includes the associated replication degree and weakly consistent information about the nodes where the resource is currently replicated. If M decides that D's resources should be replicated, it commands D to start the replication operations (message 2). D answers by forwarding a resource replica to a randomly chosen neighbor (message 3 to B), by including the requested number of replicas in the message. If the neighbor accepts to locally store the resource, it makes a local copy and recursively forwards the message, by decreasing the number of replicas yet to be instantiated. Otherwise, it only forwards the message. In the figure, node F refuses to host a server engine replica, while nodes B, L, and A accept and notify their decisions to M (message 4).

Let us note that two primary technical challenges for dense MANETs already leak out from the sketched case study: i) how to dynamically identify the nodes belonging to the dense region, given that the nodes continuously move, and ii) how to suitably choose the replica manager node, e.g., by taking into account its position with regards to the dense MANET topology. These two issues should be addressed with highly decentralized and lightweight solutions, specialized for dense MANET; the related REDMAN original proposals are the main focus of the paper and are detailed in the following.

To maintain the replication degree unchanged, the REDMAN replica manager should also react to the exit of Civilization delegates from the dense MANET: it should be notified of that event, should understand the replicas of which resources are leaving, should identify other delegates for those leaving resources still available in the dense region (via the corresponding SRT entries), and should command new replicas. Note that it is not necessary to guarantee that, for any replicated resource, in any moment, the desired replication degree is respected; it is sufficient to try to respect the replication degree in a lazy consistent way, by avoiding that all the replicas of a needed resource leave the dense region before performing a further replica dissemination.

When a new user enters the waiting hall, she can decide to play Civilization even if she has not yet installed any suitable game component on her PDA. REDMAN supports her distributed discovery to find out which games are currently available in the dense MANET and which nodes host the needed resource

replicas. Then, REDMAN is in charge of downloading both the Civilization client and the playing scenery to her device; in addition, it supports distributed cooperation between resources, e.g., between the client and a replica of the Civilization engine dynamically retrieved in the dense MANET.

The above scenario only represents a possible REDMAN use case. Let us point out that any other service where resources to replicate are simple read-only data (the list of train departures/arrivals, of movies on in town, of utility phone numbers) is a simplification of the sketched case study, with no need of dynamic composition of distributed service components.

3. The REDMAN Middleware

REDMAN addresses the issue of disseminating replicas of resources of common interest in dense MANETs, independently of unexpected node exit from the dense region, e.g., due to node mobility and battery shortage. More formally, a dense MANET is defined as the set of MANET nodes DM(n) = $\{d_0, \ldots, d_{N-1}\}$, where i) $\forall j \in [0, N-1]$ d_j has at least *n* neighbors at single-hop distance, and ii) the spatial node density in the area where DM(n) nodes are is almost constant with regards to time. Given a resource with a desired replication degree *k*, REDMAN is in charge of instantiating and distributing *k* replicas of it, and of maintaining the *k* replication degree notwithstanding the changes in the composition of the DM(n) set.

In particular, to suit resource-limited nodes, REDMAN proposes dense MANET-specific lightweight solutions. REDMAN decides not to guarantee the strict any-time consistency of the replication degree of shared resources. Instead, it employs reactive strategies to counteract the reduction of replicas in DM(n) when resource delegates either fail or leave the network. In addition, to reduce the overhead and the complexity of distributed replica management, REDMAN currently manages the replication of read-only resources, thus permitting to exclude heavy and expensive operations for possible reconciliation of concurrently updated resource replicas. Dealing with read-only resources is sufficient for guaranteeing the availability of a large class of services of primary interest in MANETs, as pointed out in Section 2. General-purpose protocols for replica reconciliation in traditional wired systems are not suitable, even in adapted forms, for dense MANETs, due to their relevant overhead and connectivity requirements [12].

In addition, we claim the suitability of providing REDMAN facilities at the application level to improve flexibility, configurability, and portability over different MANET communication solutions, and to hide lower layer implementation details from application developers. Developers of services for dense MANET should only provide each service component with metadata to describe the shared resource and to suggest the suitable replication degree depending on application-specific resource criticality. Clients in the dense MANET transparently perform discovery and retrieval operations, i.e., resource replication does not affect at all their application logic.

Figure 2 depicts the REDMAN middleware architecture organized in four facilities: Dense MANET Configuration (DMC), Replica Distribution (RD), Replication Degree Maintenance (RDM), and Resource Retrieval (RR). In the following, the paper presents a rapid overview of these facilities; additional details about the REDMAN organization can be found in [10] and at the REDMAN Web site `http://lia.deis.unibo.it/Research/REDMAN/`

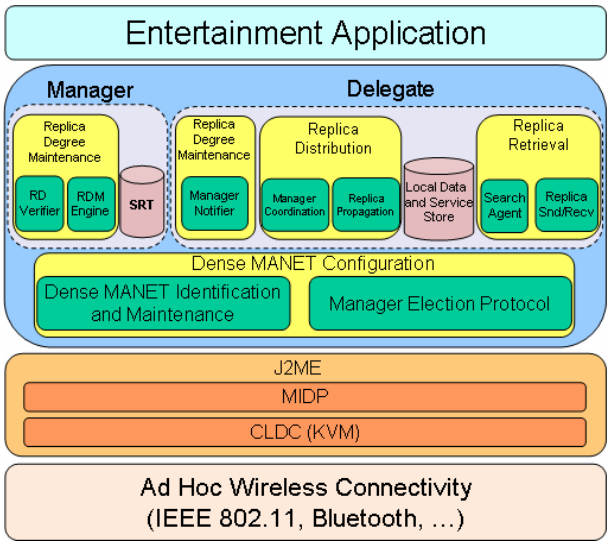

Figure 2. The REDMAN middleware architecture of facilities.

The **DMC** facility is in charge of determining the participants of the dense MANET *DM(n)*, i.e., the subset of nodes that have more than *n* nodes at single-hop distance. In addition, DMC identifies when nodes enter/exit the dense MANET, and triggers the dynamic election of the replica manager node, by minimizing the number of hops required for its messages to reach any dense MANET participant. The implementation and experimental evaluation of the REDMAN DMC are extensively described in Sections 4 and 5.

The **RD** facility operates to transparently distribute resource replicas in the dense MANET. When a delegate enters a dense region, it communicates the meta-

data of its shared resources to the replica manager, which decides the replication degree to enforce, creates new SRT entries for the newly arrived resources and commands the delegate to start the replication operations. At the moment, the replica manager simply chooses the target replication degree depending on the delegate suggestion; more sophisticated and conservative strategies to choose the proper resource replication degree are currently under investigation. Nodes willing to host a copy of the resource notify the replica manager; in this way, the manager can update the information in its SRT. Since the replica manager node has the duty of coordinating replica operations, it tends to consume more resources than other nodes. For this reason, REDMAN periodically reallocates the role by forcing a new manager election (as detailed in the next section). This also contributes to improve the suitability of nodes acting as replica managers because, during long time intervals, an elected node could lose the properties that motivated its election, e.g., its central position in the dense MANET topology.

The original lightweight **RDM** facility works to maintain unchanged the replication degree decided for each shared resource, without guaranteeing absolute consistency, i.e., it is possible to have time intervals when the requested replication degree differs from the actual number of replicas in the dense MANET. RDM works by reacting when resource delegates leave the dense MANET. If a delegate realizes its departure, it distributes owned resources to willing neighbors. To prevent possible inconsistencies due to delegate abrupt failures/exits, the manager periodically checks resource replication degrees and commands the diffusion of additional copies when needed. REDMAN manages also the case of the manager abandoning the dense MANET. If the manager realizes in time it is going to exit, it delegates its role to the first neighbor node found with suitable characteristics to act as manager, e.g., battery charge and local memory greater than a threshold. In the case the manager abruptly fails, when any delegate senses the manager unavailability, it triggers a new election.

The **RR** facility has the goal of retrieving resource replicas effectively, by exploiting a lightweight distributed protocol. RR implements a simple retrieval solution based on client flooding of resource requests. When a node receives a search message, if it owns a copy of the needed resource, it directly replies to the requester; otherwise, it forwards the request to all its neighbors. The above retrieval solution is quite trivial, but the proposal of novel and effective RR algorithms is not the current focus of the REDMAN research activity.

4. Dense MANET Configuration in REDMAN

DMC is the REDMAN facility in charge of determining which nodes belong to the dense MANET and play the role of replica managers. These functions are crucial for the realization of all other REDMAN facilities and require original solutions that fit the specific characteristics of the dense MANET deployment scenario.

4.1. Dense MANET Identification

REDMAN proposes an original solution to dynamically determine the nodes that currently participate to a dense MANET. The primary idea is not to maintain a centralized, global, and always up-to-date vision of all the nodes and of their network topology, but to design a simple, lightweight, and decentralized protocol where any node autonomously determines whether it belongs to the dense MANET. One node is in the dense MANET DM(n) only if the number of its neighbors, i.e., the nodes at single-hop distance, is greater than *n*. Each node autonomously discovers the number of its neighbors by exploiting simple single-hop broadcast discovery messages.

In more detail, at any time one REDMAN node can start the process of dense MANET identification/update; in the following, we will call that node the *initiator*. The initiator starts the protocol by broadcasting a discovery message that includes the number of neighbors required to belong to the dense region. When receiving this message, each node willing to participate replies by forwarding the message to its single-hop neighbors, if it has not already sent that message. After a specified time interval, any node autonomously checks whether the number of received discovery messages is greater than the specified number of neighbors to belong to the dense region, and autonomously decides whether it belongs to the dense MANET. Let us observe that discovery broadcasts could provoke packet collisions (broadcast storm issue [13]): to avoid this problem, any REDMAN node defers node broadcasts of a random time interval. Notwithstanding this introduced random delay, the time needed to complete the dense MANET identification protocol is limited and largely acceptable; in fact, it shows a linear dependence on the dense region diameter, not on the number of its participants.

Since dense MANET nodes can move after the identification process, the proposed algorithm includes a lightweight lazy-consistent maintenance phase. Nodes periodically exchange Hello packets; each node

receiving a Hello message records its source in a table entry, with an associated timeout; next Hellos received from the same source restart a new timeout. Dense MANET nodes periodically check whether their table entries are still valid; if an entry has expired, the node removes it from the table, and verifies whether the condition for dense MANET belonging still holds.

4.2. Replica Manager Election

REDMAN proposes an original, lightweight, and decentralized protocol to elect the replica manager. To reduce the communication overhead (both in terms of dissipated energy and elapsed time) for the manager to get in touch with all dense MANET participants, it could be useful to choose a node located in a topologically central position. More precisely, REDMAN aims at electing one node that minimizes the number of hops required to reach its farthest nodes belonging to the dense MANET. In addition, the proposed protocol for replica manager election has not the goal of finding the optimal solution with regards to the above minimization criterion: REDMAN exploits some heuristics to relevantly limit the election overhead while achieving a good quality manager designation. The proposed solution explores, as manager candidates, only a subset of nodes in the dense MANET, called Investigated Nodes (INs). To avoid the overhead of exhaustive search, REDMAN adopts an exploration strategy that limits the IN number, by only choosing successive INs to get closer to the dense MANET topology center at each exploration step, thus decreasing the distance of each successive IN from farthest participants.

Therefore, a primary issue is how INs can autonomously determine the direction towards the dense MANET topology center by exclusively exploiting information about MANET farthest nodes. To this purpose, the adopted guideline is to explore the nodes located along the direction of farthest nodes from previously considered INs. In fact, by moving toward that direction, each protocol step considers an IN that is placed one-hop closer to the previously identified farthest nodes; therefore, the IN distance from farthest nodes tends to decrease and to converge close to the best solution. Figure 3 shows an example of application of that guideline. The first step of the protocol considers node I: its farthest node is H, located at 4-hop distance; so, I is tagged with the value of that distance (I^4 in the figure). Then, the REDMAN manager election protocol considers A because it is the first node along the path from I to H: A's farthest node is H, at 3-hop distance (A^3). At the next iteration, the protocol explores node D and elects it as the replica manager; in fact, D can reach any other node in the dense MANET with a maximum of two hops.

After having informally introduced the above main guidelines of the protocol, let us now better detail how the manager election works. The protocol considers the initiator as the first IN; then, it re-iterates the farthest node determination process (see Section 4.2.2) to evaluate other promising nodes, until it reaches a satisfying solution. Each IN executes three operations: i) it determines the number of hops of the shortest paths connecting it to any farthest node in the dense MANET (the maximum of those hop numbers is called *INvalue*); ii) it identifies its neighbors located in the direction of its farthest nodes (*forwarding neighbors*); iii) it autonomously chooses the next IN among all the unexplored forwarding neighbors of already explored minimum-valued INs. To take devices heterogeneity into account, the algorithm promotes the exploration only of nodes suitable to carry manager tasks, e.g., with sufficient memory and expected battery life. The manager election protocol ends when either the REDMAN heuristic criterion presented in Section 4.2.1 determines there are no more promising nodes, or the *current INvalue = MinInt((worst explored INvalue)/2)*, where *MinInt(x)* returns the least integer greater than *x*. Since REDMAN considers bi-directional links among MANET nodes, when the above equation is verified, it is easy to demonstrate that REDMAN has reached the optimal solution for the manager election.

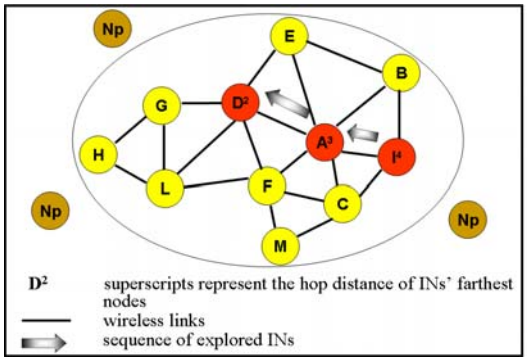

Figure 3. REDMAN exploring the sequence of INs I→A→D.

Since the duration of election protocol operations is significantly lower than the usual time for mobile nodes to enter/exit the dense MANET, during the election process REDMAN works as if the participating nodes were fixed, by approximating the actual deployment environment to a stationary wireless network. In addition, if the manager moves after the completion of the election process, either by leaving the

dense MANET or by relevantly going far from its central position, REDMAN senses the events and enforces suitable strategies for manager replacement [10].

The following subsections detail the adopted heuristic to limit the number of explored INs and the exploited solution to determine, given a node, its farthest nodes in the dense MANET.

4.2.1. Heuristic-based Overhead Reduction

To reduce the overhead due to a large number of re-iterations of the manager election protocol, REDMAN exploits a heuristic approach that has experimentally demonstrated to reach high quality solutions, close to the optimal choice of the manager (see Section 5.2).

To improve the flexibility and adaptability of the REDMAN election strategy to the peculiar characteristics of the dense MANET where it is deployed, REDMAN provides two tuning parameters that enable dense MANET administrators to trade between the quality of the manager election protocol and its performance. The first parameter, *DesiredAccuracy*, permits the initiator to tune the approximation considered acceptable for the election solution already found (see Figure 4). The second parameter, *MaxConsecutiveEqualSolutions*, is introduced by observing that, when the REDMAN election protocol approaches the optimal solution, it often explores other candidate nodes without improving the current best IN value. For each explored solution equal to the current best, REDMAN increases a counter; the counter resets when REDMAN finds a new solution outperforming the old best. The adopted heuristic stops the iterations when the counter reaches *MaxConsecutiveEqualSolutions*. Figure 4 shows the pseudo-code of the election protocol.

4.2.2. Determination of Farthest Nodes

REDMAN proposes a simple broadcast-based strategy to detect the lengths of the shortest paths connecting the current IN to the farthest participants in the dense MANET. The current IN starts the protocol by broadcasting a farthest node determination message including a counter initialized to 0. Every node receiving that message and belonging to the dense MANET increases the counter and forwards the message, without resending an already sent message, similarly to the case of dense MANET identification. Figure 5.a shows the message propagation from node I (the current IN) to all the dense MANET nodes. Each node is marked with the value of its counter, i.e., its distance from I in number of hops. For the sake of simplicity, the figure does not show all broadcast messages exchanged, but only those from closer nodes to farther ones with regard to I.

```
exploredList = ø; forwarderList = ø;
bestNode = ø; bestValue = MAX; worstValue = 0;
unexploredList = Initiator;
while (unexploredList != ø) {
    IN = Head(unexploredList);
    INValue = DistanceFromFarthest(IN);
    exploredList = exploredList U IN;
    unexploredList = unexploredList – IN;
    forwarderList = GetPromisingNeighbors(IN);
    forwarderList = forwarderList – exploredList;
    if ((INValue == MinInt(worstValue/2) || (INValue <=
    worstValue * desired_accuracy)) exit;
    if (INValue < bestValue) {
        bestNode = IN; bestValue = INValue;
        consecutiveEqualSolutions = 0;
        unexploredList = forwarderList ; }
    if (INValue > worstValue) { worstValue = INValue;
        if ((bestValue == MinInt(worstValue/2) || bestValue
        <= worstValue * desired_accuracy)) exit; }
    if (INValue == bestValue) {
        consecutiveEqualSolutions++ ;
        if (conecutiveEqualSolutions == max_consecutive_
        equal_solutions) exit;
        unexploredList = unexploredList U forwarderList ; }
    } Print(bestNode)
```

Figure 4. Pseudo-code of the REDMAN manager election protocol.

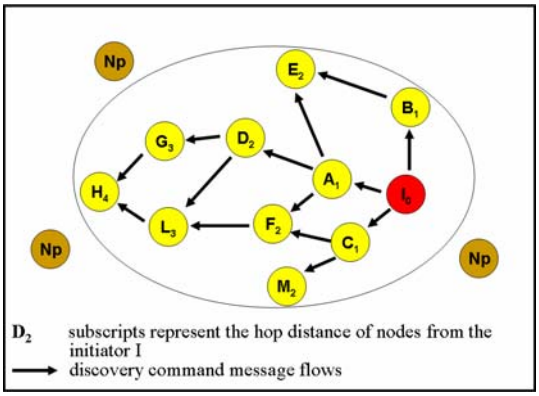

Figure 5.a. The current IN (I) broadcasts exploratory messages for manager election.

To limit bandwidth consumption, each node replies to the IN by communicating its distance if and only if it cannot detect any node farther than itself at single-hop distance. In fact, when a node receives a farthest node determination request, it starts a timeout; at timeout expiration, it replies if it has not received any other broadcast from a node farther than itself (with a greater counter). Let us rapidly observe that the choice of that timeout is simple because it represents the time for single-hop neighbors to re-broadcast the message and does not depend on the number of participants and on the dense MANET diameter [14].

The nodes replying back to the farthest node de-

termination message include not only the actual farthest nodes for the current IN, but also other nodes situated at the dense region boundaries. Figure 5.b shows that not only H (the only farthest node) replies to I, but also E and M, which are at the boundaries of the dense MANET. All other nodes, e.g., node A, do not reply because they are prevented by single-hop neighbors placed at greater distance, e.g., nodes E, F, and D. Every time the IN receives a reply, it records the message source identity, its distance, and the incoming direction, i.e., the neighbor that last forwarded the message. Finally, the IN determines the identity of the farthest nodes, by excluding non-farthest nodes. The IN assumes the distance of the determined farthest node(s) as its INvalue.

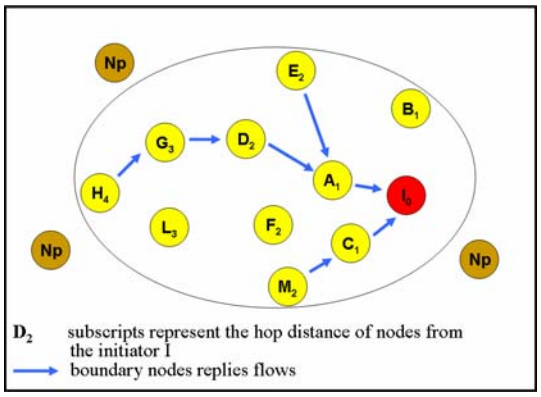

Figure 5.b. Only nodes at dense MANET boundaries (E, F, H) reply to the current IN.

5. Experimental Results

To evaluate the effectiveness of our approach, we have implemented a REDMAN prototype and extensively simulated the behavior of the dense MANET identification and the manager election solutions in the NS2 simulator [15]. The simulations have three main goals: i) to determine the network overhead of the proposed protocols, in terms of the number of packets exchanged among MANET nodes; ii) to evaluate the accuracy of the manager election protocol notwithstanding the heuristic-based overhead reduction; iii) to evaluate the robustness of the dense MANET identification protocol while increasing node mobility.

Our simulation deployment scenario consists of randomly positioned nodes in a square area. The area is split in two zones, a smaller Internal Square (IS) at the center of the area, and the remaining area around the IS, called External Square (ES). Nodes are distributed so that the dense MANET almost coincides with the IS area (the IS node density is relevantly higher than the ES one). In addition, if not differently specified, we have used default NS2 values for simulation parameters, e.g., constant 250m circular transmission ranges, bi-directional connectivity, and IEEE 802.11 link layer protocol.

The code of the REDMAN protocols, the exhaustive description of all NS2 simulation parameters used, and additional performance figures about REDMAN are available at the REDMAN Website.

5.1. Network Overhead

First, we have carefully evaluated the network overhead of the REDMAN protocols for dense MANET identification and manager election, to verify that the proposals are lightweight enough for the addressed deployment scenario. We have tested the two protocols in different simulation environments, with a number of nodes ranging from 50 to 550 (increasing of 20 nodes each step). The size of ES/IS areas have been changed with the changing number of nodes involved, to maintain the same ES/IS node densities in all simulations. For both protocols we have measured the average number of packets sent by each participant, over a set of more than 1,000 simulations.

The results reported in Figure 6 are normalized to the number of nodes actively participating in the related protocol. The dense MANET identification protocol is designed to determine participant nodes by requiring only one local broadcast from each node reachable from the initiator. With regard to the replica manager election protocol, also in this case the number of sent packets is very limited and grows very slightly with the number of participants. In fact, sent packets tend to be proportional not to the total number of nodes, but to the number of iterations required to identify an acceptable solution. The number of iterations is roughly proportional to the dense MANET diameter (approximately 3 hops for the 50-node case and 10 hops for the 550-node case) and grows less than the number of participants.

Figure 7 reports how the number of iterations grows in function of the growing of network participants. The results are average values on a set of simulations where the role of initiator is assigned to a different node at each simulation; the results have shown a very limited dependence on the choice of the initiator node. In summary, the experimental results about REDMAN overhead demonstrate the feasibility and the good scalability of the proposed solutions (the network overhead slowly increases when the number of participants grows). Let us briefly observe that the non-monotonic growth of the overhead trace is due to different factors. First, the dense MANET diameter only increases in

correspondence with some threshold values of the number of participants, thus affecting the number of optimal solutions in the network. Moreover, the adopted heuristic parameters, such as *MaxConsecutiveEqualSolutions*, influence the number of iterations of the manager election protocol (see further details at the REDMAN Web site).

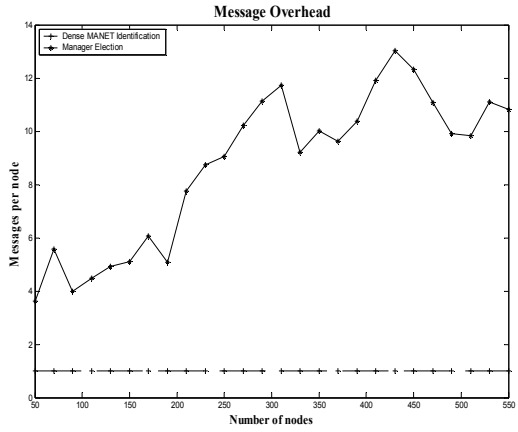

Figure 6. Messages sent/received in dense MANET identification and manager election.

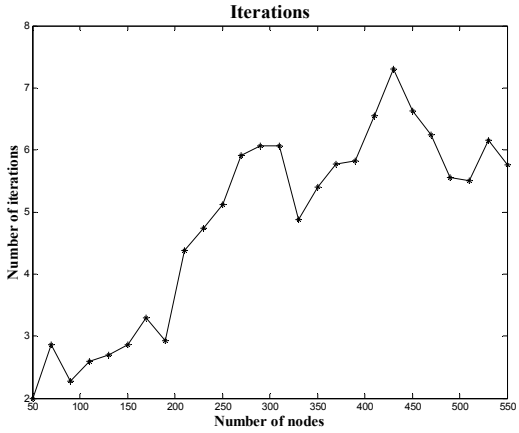

Figure 7. Number of iterations needed for the REDMAN manager election protocol.

5.2. Manager Election Accuracy

To quantitatively assess the effectiveness of the REDMAN election protocol, we have measured its accuracy in assigning the manager role to a node close to the actual topology center of the dense MANET. We have run over 200 simulations in the most populated scenario of 550 nodes and, for any simulation, we have measured the hop distance between the manager chosen by the REDMAN protocol and the actual optimal solution. The results in Figure 8 are obtained by starting each election from a different initiator node. In more than 90% of the runs, the REDMAN protocol has identified either optimal solutions or quasi-optimal solutions at 1-hop distance from the actual optimum. The average inaccuracy is only 0.385 hops, which represents a largely acceptable value for the addressed application scenario.

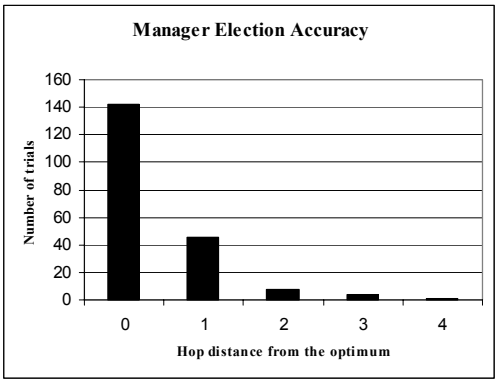

Figure 8. Accuracy of the REDMAN manager election protocol.

5.3. Robustness in Function of Node Mobility

To test the REDMAN dynamic behavior, we have evaluated the dense MANET identification protocol by varying the mobility characteristics of network nodes. Let us rapidly note that the manager election protocol executes for very limited time periods and re-starts its execution only after a long time interval; to a certain extent, it is assumable that it operates under static conditions. On the contrary, the dense MANET identification protocol should continuously work to maintain an almost consistent and updated view of the dense MANET participants, crucial for the effective working of REDMAN solutions. For this reason, we have focused on REDMAN identification behavior in function of node mobility.

We have considered the same 110-node scenario of the tests in Section 5.1. For any node (randomly chosen among the ones close to the IS boundary) that exits the dense region, a new node enters the IS, to keep unchanged the spatial node density according to the dense MANET definition. Any pair of random movements of randomly chosen nodes occurs every *M* seconds, with *M* that varies from 10 to 60. Any other node movement not producing entrances/exits in/from the dense MANET does not affect at all the behavior of the REDMAN identification solution.

Figure 9 reports the dense MANET identification inaccuracy, defined as the difference between the num-

ber of dense MANET participants determined by the REDMAN protocol and its actual value. The inaccuracy is reported as a function of the mobility period M and for different values of the time period used for Hello packets. Each point in the figure represents an average value obtained by capturing the state of the network in 20 different runs. The figure shows that the average inaccuracy is very limited and always within a range that is definitely acceptable for lazy consistent resource replication in dense MANETs. As expected, the inaccuracy grows when node mobility grows, for fixed values of the Hello message period. However, even for relatively high values of the Hello period, the REDMAN identification inaccuracy is negligible for the addressed application scenario (on the average, always less than 1.7), for the whole range of node mobility frequencies that can be of interest for dense MANETs. Let us observe that this permits to set relatively high periods for Hello packets, while obtaining a low inaccuracy for the dense MANET identification, thus significantly reducing the message exchange overhead.

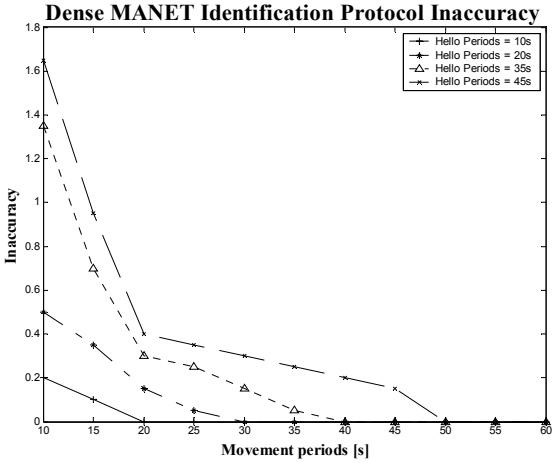

Figure 9. Inaccuracy of the REDMAN dense MANET identification protocol.

6. Conclusions and On-going Research

The challenge of supporting distributed services over dense MANETs can significantly exploit the assumption of high node population to enable lazy consistent forms of resource replication. This can increase the availability of common interest resources notwithstanding the unpredictable node exit from the dense MANET. The REDMAN project demonstrates how it is possible to provide middlewares for lightweight and effective replica management also in dense MANETs. In particular, REDMAN proposes original and completely decentralized protocols, specifically designed for dense MANET, to dynamically determine dense region participants and to elect replica managers by minimizing the distance from farthest nodes. These protocols impose a limited overhead in terms of exchanged messages and rapidly converge to high quality solutions even in very large-scale deployment scenarios with high node mobility. To the best of our knowledge, there are no other research activities that have already proposed and evaluated solutions for dense MANET identification and dynamic determination of the MANET topology center.

The experimental results obtained are encouraging further REDMAN-related research activities. First, we are evaluating the impact of the proposed protocols on the performance of REDMAN-supported applications, e.g., the Civilization strategy game. In particular, we are carefully evaluating in which deployment conditions the traffic reduction due to the central position of the replica manager offsets the overhead incurred in its election. Secondly, we are exploring decentralized security mechanisms to enable the dense MANET participation (and the access to shared resources) only to authorized nodes. Finally, we are completing the implementation of our J2ME REDMAN prototype and going to extensively test it in a wide-scale dense MANET consisting of a number of devices that exploit IEEE 802.11b connectivity in ad-hoc mode.

Acknowledgements

Work supported by the Italian Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) in the framework of the FIRB WEB-MINDS Project "Wide-scale Broad-band Middleware for Network Distributed Services" and by the Italian Consiglio Nazionale delle Ricerche (CNR) in the framework of the Strategic IS-MANET Project "Middleware Support for Mobile Ad-hoc Networks and their Application".

References

[1] I. Chlamtac, M. Conti, J. J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges", *Elsevier Ad Hoc Networks*, Jul. 2003.

[2] Y. Yuan; W. Arbaugh, "A Secure Service Discovery Protocol for MANET", *14th IEEE Conf. on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2003.

[3] A. Helmy, S. Garg, P. Pamu, N. Nahata, "Contact-based Architecture for Resource Discovery (CARD) in

Large Scale MANETs", *17th IEEE Int. Parallel and Distributed Processing Symp. (IPDPS)*, Apr. 2003.

[4] T. Qing, D.C. Cox, "Optimal Replication Algorithms for Hierarchical Mobility Management in PCS Networks", *IEEE Wireless Communications and Networking Conf. (WCNC)*, Mar. 2002.

[5] J. J. Kistler, M. Satyanarayanan, "Disconnected Operations in the Coda File System", *ACM Transactions on Computer Systems*, Feb. 1992.

[6] A. J. Demers, K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, B. B. Welch, "The Bayou Architecture: Support for Data Sharing among Mobile Users", *1st IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, Dec. 1994.

[7] K. Chen, K. Nahrstedt, "An Integrated Data Lookup and Replication Scheme in Mobile Ad Hoc Networks", *SPIE Int. Symp. on the Convergence of Information Technologies and Communications (ITCom 2001)*, Aug. 2001.

[8] K. Rothermel, C. Becker, J. Hahner, "Consistent Update Diffusion in Mobile Ad Hoc Networks", *Technical Report 2002/2004*, CS Dep., Univ. of Stuttgart.

[9] S. Garg, Y. Huang, C.M.R. Kintala, K.S. Trivedi, S. Yajnik, "Performance and Reliability Evaluation of Passive Replication Schemes in Application-level Fault Tolerance", *29th IEEE Int. Symp. on Fault-Tolerant Computing*, June 1999.

[10] P. Bellavista, A. Corradi, E. Magistretti, "REDMAN: a Decentralized Middleware Solution for Cooperative Replication in Dense MANETs", *2nd Int. Workshop on Middleware Support for Pervasive Computing (PerWare)*, Mar. 2005.

[11] Sid Meier's Civilization III, http://www.civ3.com

[12] L.P. Cox, B.D. Noble, "Fast Reconciliations in Fluid Replication", *21st Int. Conf. on Distributed Computing Systems (ICDCS)*, Apr. 2001.

[13] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, J.-P. Sheu, "The Broadcast Storm Problem in Mobile Ad Hoc Networks", *5th ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, Aug. 1999.

[14] S. Nesargi, R. Prakash, "MANETconf: Configuration of Hosts in a Mobile Ad Hoc Networks", *21st Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM)*, Jun. 2002.

[15] The VINT Project, http://www.isi.edu/nsnam/vint/