REDMAN: an Optimistic Replication Middleware for Read-only Resources in Dense MANETs

Paolo Bellavista, Antonio Corradi, Eugenio Magistretti

Dipartimento di Elettronica, Informatica e Sistemistica - University of Bologna
Viale Risorgimento, 2 - 40136 Bologna - ITALY
Phone: +39-051-2093001; Fax: +39-051-2093073

Elsevier use only: Received date here; revised date here; accepted date here

Abstract

The spread of wireless portable devices is pushing towards service provisioning over dense Mobile Ad-hoc NETworks (MANETs), i.e., limited spatial regions, such as shopping malls and airports, where a high number of mobile peers can autonomously cooperate without a statically deployed network infrastructure. The paper proposes the REDMAN middleware to manage, retrieve, and disseminate replicas of data/service components to cooperating nodes in a dense MANET. The guideline is to exploit high node population to enable optimistic lightweight resource replication capable of tolerating node exits/failures. REDMAN adopts original approximated solutions, specifically designed for dense MANET, that have demonstrated good scalability and limited overhead for dense MANET configuration (node identification and manager election), for replica distribution/retrieval, and for lazily-consistent replica degree maintenance.

Keywords: Optimistic Replication; Middleware; Mobile Ad Hoc Networks; IEEE 802.11
PACS: Type your PACS codes here, separated by semicolons ;

1. Introduction

The mass diffusion of wireless-enabled portable devices makes possible novel service market opportunities and innovative challenging 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, wireless nodes should have access to distributed resources, possibly offered by other mobile wireless peers, without requiring static knowledge about their execution environments. In addition, resources should be permanently available, not depending on possible 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 challenging issue of resource replication in mobile environments with the goal of increasing accessibility and effectiveness [4]. Most recent investigations have focused on enhancing information availability in both 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 [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 MANET partitions. Most proposals assume that wireless nodes are aware of their physical position, e.g., by imposing the constraint of hosting Global Positioning System (GPS) 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 such as high loss rates and frequent network partitioning, the effective and lightweight management of replicas of data and service components is unfeasible when dealing with both general-purpose MANETs and strict consistency requirements on distributed copies. 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 mobile 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, railway stations, and university campuses;
- has a node density, defined as 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, retrieves, and manages replicas of common interest resources among cooperating nodes in dense MANETs [9]. REDMAN has the main goal of improving the availability of data and service components, by adopting original lightweight solutions that exploit the peculiar characteristics of dense MANETs to achieve replication effectiveness and limited overhead. The primary idea is of maintaining, within a dense region, a desired replication degree for needed resources, independently of possible (and unpredictable) exits of replica-hosting nodes from the dense MANET.

Let us point out that REDMAN takes into account the usual resource limitations of portable client devices by choosing not to respect strict consistency requirements on the number of resource replicas available at any instant, which could be temporarily lower than the desired one. Achieving non-optimal but sufficiently accurate solutions is the main guideline adopted in all REDMAN protocols; the goal is to probabilistically guarantee the full availability of replicated resources, by simultaneously limiting the middleware overhead. For instance, the REDMAN middleware benefits from assigning a privileged role in replica management to wireless nodes currently located close to the topologic center of the dense MANET. The determination of these "central nodes" is performed in an approximated and lightweight way, in order to limit the generated network traffic and the overhead imposed to collaborating 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 [10]. 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 (and consequently the desired replication degree) of shared resources.

The paper presents how 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 orchestrating resource replication and maintenance, in a completely decentralized way via original protocols specifically designed for dense MANETs. In addition, the paper details the higher-level REDMAN functionality, based on the above mechanisms, to effectively distribute, retrieve, and maintain the requested replication degree for distributed copies. All REDMAN facilities exploit approximated topologic information about dense MANETs, obtained in a lightweight way without imposing any requirement on the hardware equipment of participant nodes, e.g., to be GPS-enabled.

The remainder of the paper is organized as follows. Section 2 sketches a practical case study scenario to better clarify, from the beginning, the motivations and the functionality needed to enable replica distribution in dense MANETs. Section 3 rapidly overviews the REDMAN architecture and its primary solution guidelines, while, in the following, the paper extensively describes the different REDMAN facilities. In particular, Section 4 presents the REDMAN protocols for dense MANET identification and manager election. On top of these mechanisms, REDMAN performs replica distribution/retrieval and replica degree maintenance, as described in Section 5. Section 6 reports extensive simulation results about the REDMAN prototype, whose performance shows the effectiveness and the limited overhead of the proposed solution, by confirming the suitability of the application-level middleware approach even in conditions of high node mobility. Notes about potential security issues, related work, and conclusions end the article.

2. REDMAN at Work in a Practical Case Study: Collaborative Game Playing at the Railway Station

To practically introduce the REDMAN middleware functions in a simple case study scenario, let us consider travelers with REDMAN-enabled Personal Digital Assistants (PDAs) who are interested in playing a strategy game while spending their time 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 notwithstanding continuous arrivals and departures. Therefore, wireless devices in a waiting room are suitable candidates to compose a dense MANET.

Several travelers could be keen on spending their 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 be interested in playing new videogames (or new versions with new additional features and playing sceneries), by dynamically discovering them in the community of cooperative travelers and by dynamically downloading them 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 subset of nodes that host server components.

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 needed resources, thus providing travelers with a set of locally available games that dynamically grows

depending on the new resources that visiting travelers bring into the waiting room and decide to share with other dense MANET cooperating nodes.

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 support game availability in the dense MANET with no need of explicit and static installations, even if mobile nodes continuously enter/exit the waiting room. Consider the example of deployment scenario depicted 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 region hosting a shared resource (or one of its replicas).

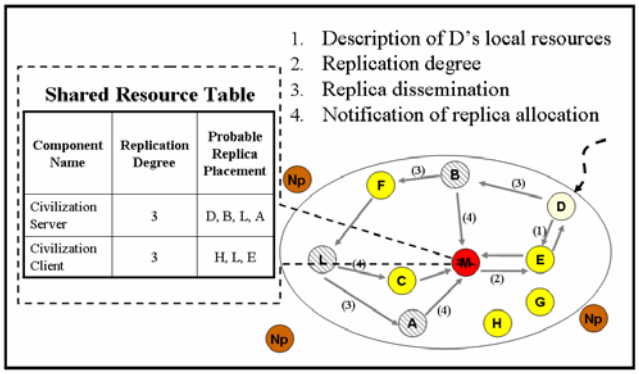

Figure 1. REDMAN-based replication of the Civilization server.

For instance, when the delegate D with the Civilization server engine enters the waiting room, the REDMAN middleware component running on D's device sends the descriptions of D's resources to the REDMAN replica manager M, running in another node of the dense MANET (message 1). The replica manager is in charge of enforcing the needed replication degree and of maintaining information about shared resources replicated in the dense region (Shared Resource Table - SRT). For any shared resource, the corresponding SRT entry includes the associated target 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 reacts by forwarding a resource replica to a randomly chosen neighbor (message 3 to B), according to the replica distribution protocol detailed in the following, by including the still requested number of resource 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 to cooperate and notify their decisions to M (message 4).

Let us note that the sketched case study already points out two primary technical challenges for replication in dense MANETs: 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 in the dense MANET topology. These two issues are at the basis of all other REDMAN middleware features, which directly relate to the actual distribution and retrieval of resource replicas. The addressed deployment scenarios require considering highly decentralized and lightweight solutions for dense MANET configuration and resource replication, capable of scaling well in execution environments with several hundreds of mobile wireless peers. Therefore, it is necessary to design and implement original mechanisms specialized for the peculiar characteristics of dense MANETs.

In addition, to maintain the replication degree unchanged after first replica distribution, the REDMAN replica manager should react to possible exits of Civilization delegates from the dense MANET: it should be notified of such kind of events, 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 there is not the need to guarantee that, for any replicated resource, at any moment, the desired replication degree is exactly enforced; it is sufficient to try to maintain the replication degree in a lazy consistent way, by avoiding that all the replicas of a resource leave the dense region before performing a further distribution of resource copies.

When a new user enters the waiting hall, she can decide to play Civilization even if she has not yet installed the needed game components on her PDA. REDMAN supports her distributed discovery to understand which games are currently available in the dense MANET and which nodes host the desired resource replicas. Then, REDMAN is in charge of downloading both the Civilization client and the playing scenery to her device. Once downloaded them locally and once identified via REDMAN a suitable node with a Civilization server engine replica, the client can interact with the remote engine by adopting usual MANET communication protocols, by bypassing the REDMAN middleware intermediation. The

assumption of dense MANET guarantees that network partitioning will not occur during service provisioning. In the case that the used engine delegate leaves the waiting hall, the client can ask the REDMAN middleware for retrieving another engine replica in the dense MANET; then, the client can restart her gaming session by connecting to the newly retrieved replica.

The above scenario only represents a possible REDMAN use case. Let us observe that any other service where resources to replicate are read-only data (the list of train departures/arrivals, of movies on in town, of utility phone numbers, and of pictures taken at a rock concert that fans would like to share with other people among the public) is a simplification of the sketched case study, with no need of dynamic composition and distributed coordination of service components.

3. The REDMAN Middleware

REDMAN is a middleware solution that 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 protocols that achieve approximated non-optimal solutions, e.g., they do not guarantee the strict any-time consistency of replication degree. In addition, to reduce the overhead and the complexity of distributed replica management, REDMAN 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 already pointed out in the previous section. 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].

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 low-layer implementation details from application developers. Thanks to the REDMAN middleware, in fact, developers of dense MANET services should only provide each service component with descriptive metadata and suggest the desired replication degree depending on application-specific resource criticality. Clients in the dense MANET transparently perform discovery and retrieval operations; REDMAN-based resource replication does not affect at all their application logic.

Figure 2 depicts the REDMAN middleware architecture organized in two layers of facilities: Dense MANET Configuration (DMC) includes mechanisms for participant identification and manager election; on top of DMC, REDMAN provides facilities for Replica Distribution (RD), Replica Retrieval (RR), and Replica Degree Maintenance (RDM).

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 optimistically identifies when nodes enter/exit the dense MANET and triggers the dynamic election of replica managers by trying to minimize the number of hops required for their messages to reach any dense MANET participant.

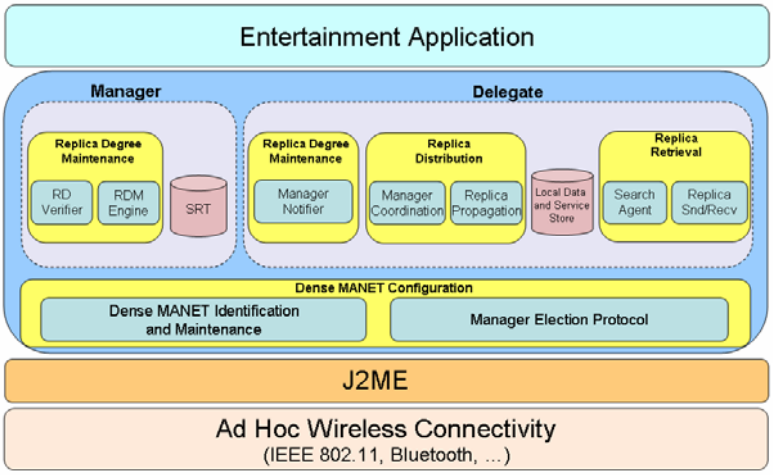

Figure 2. The REDMAN middleware architecture of facilities.

The **RD** facility operates to transparently distribute resource replicas in the dense MANET. When a delegate enters a dense region, it communicates the metadata of its shared resources to the replica manager, which decides the replication degree to enforce, creates new SRT entries for newly arrived resources, and commands the delegate to start the replication operations. REDMAN distributes resource replicas according to a simple and original gossip-based strategy: essentially, each node randomly decides whether to host a replica on its storage and successively forwards the resource to be replicated to one of its neighbors.

The **RR** facility has the goal of effectively retrieving resource replicas, by exploiting a lightweight distributed search protocol. RR implements a simple retrieval solution based on the distribution of non-strictly consistent replica placement information on a subset of nodes in the dense MANET. The REDMAN guideline is to exploit the process of replica distribution itself to disseminate also placement data in a lightweight way. In the RR phase, when a dense MANET node receives a search message, it directly replies to the searcher if it owns either a copy of the needed resource or the associated placement information. Otherwise, it forwards the request to all its neighbors according to a simple "expanding ring search" strategy, which limits search message flooding.

The **RDM** facility works to maintain unchanged the replication degree enforced for each shared resource, without strictly guaranteeing consistency: it is possible to have time intervals when the requested replication degree differs from the actual number of replicas in the dense MANET. RDM mainly works by reacting to resource delegates leaving the dense region. To face also delegate abrupt and non-detected failures/exits, at large time intervals RDM checks the number of available replicas and possibly commands the manager to coordinate the distribution of additional copies.

In the following, Section 4 and Section 5 delve into the detailed technical description of the design and implementation choices followed by REDMAN original solutions for, respectively, the DMC facility layer and the high facility one.

4. Dense MANET Configuration in REDMAN

The REDMAN low-layer DMC facilities are in charge of determining which nodes belong to the dense MANET and which ones among them have to play the role of replica managers. These low-layer mechanisms 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 are currently willing to 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 or not. 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.

By delving into finer details, 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 original REDMAN protocol for dense MANET identification by broadcasting a discovery message that includes the number of neighbors (*NoN*) required to belong to the dense region and the identity of the sender. That number can be autonomously decided depending on the desired degree of connectivity redundancy: typically, a value between 10 and 20 ensures a sufficiently large set of alternative connectivity links. When receiving the discovery message, each node willing to participate replies by forwarding the message to its single-hop neighbors, if it has not already sent that message, and by updating a local list with IP addresses of detected neighbors. The current DMC implementation assumes that each participant node adopts a zero configuration solution, such as [13, 14]; guaranteeing IP address uniqueness is out of the scope of our research. After a specified time interval, any node autonomously checks whether its list contains more than *n* nodes, and autonomously decides whether it belongs or not to the dense MANET.

Let us observe that discovery broadcasts could provoke packet collisions; the problem is well-known in the literature and usually identified as the "broadcast storm" issue [15]. In order to avoid the problem, any REDMAN node defers node broadcasts of a random and limited time interval. Techniques for collision and overhead reduction such as the ones presented in [15, 16] do not fit dense MANET identification, since local density evaluation is based on the assumption that each node receives messages from all its neighbors. The dense MANET identification protocol has demonstrated to scale well also in large deployment scenarios, with a completion time linearly dependent on the dense MANET diameter, i.e., on the maximum hop distance among any couple of nodes belonging to the dense region, and an overall number of exchanged messages equal to the number of dense MANET participants, as illustrated in Section 6.

Given that dense MANET nodes can move during and after the identification process, the proposed algorithm achieves an approximated solution and requires including a lightweight lazily-consistent maintenance phase. Nodes periodically exchange Hello packets with their neighbors; each node receiving a Hello message records its source in a table entry, with an associated timeout; next Hello packets 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.

Let us rapidly observe that dense MANETs are virtual and dynamic group organizations of MANET nodes. Therefore, there is also the possibility to have more than one coexisting dense MANET, each one with a different NoN, involving the same MANET peers, e.g., in the case that several initiators trigger the dense MANET identification process with different NoN values. However, since the addressed deployment scenarios generally exhibit high density gradients close to boundaries, several different NoN values usually determine the same *DM(NoN)* sets. Given the limited applicability and the additional overhead of maintaining more than one dense MANET over the same set of physical nodes, the current REDMAN implementation does not support multiple overlapping dense MANETs with different NoN: the NoN value for a dense area cannot be changed for a long time interval after the first identification procedure.

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, the REDMAN middleware works to assign the manager role to 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. The proposed protocol for replica manager election has not the goal of finding the optimal solution: the idea is to exploit 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 that goes from the considered IN to its farthest node. 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 a practical 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, which is chosen as replica manager by respecting the termination criteria described in the following of the section. Node D can reach any other node in the depicted dense MANET with a maximum path of two hops.

Let us observe that REDMAN provides a simple way to react also to manager exits from the dense MANET. If the manager realizes it is going to exit, e.g., because its battery power is lower than a specified threshold, it delegates its role to the first neighbor node found with suitable battery charge and local memory. In the case the manager abruptly fails, any resource delegate that senses the manager unavailability can trigger a new election procedure.

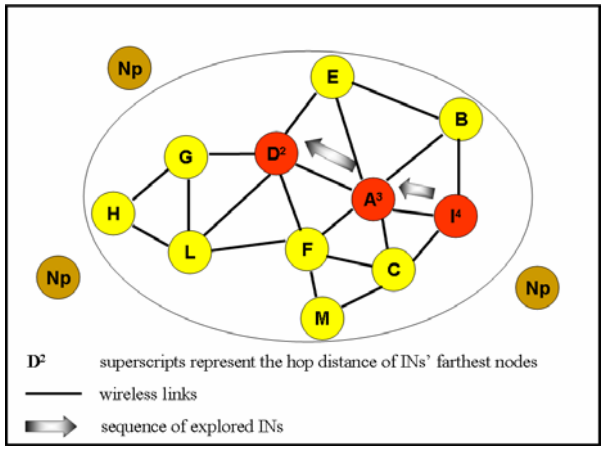

Figure 3. REDMAN exploring the sequence of INs I→A→D.

After having informally introduced the main guidelines of the protocol, let us now precisely specify 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 solution which is considered satisfying according to the adopted heuristics. 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*); and iii) it autonomously chooses the next IN among all the unexplored forwarding neighbors of already explored INs with lowest associated values.

To take possible device heterogeneity into account, REDMAN promotes the exploration only of INs suitable to play the role of replica manager once elected. For instance, if a potential IN device has insufficient memory and too low battery life (if compared with configurable REDMAN thresholds), it is excluded from the manager election protocol.

The protocol ends when either the REDMAN heuristic criterion of 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.

The frequency of node entering/exiting dense regions, as well as their speed, are generally low in the addressed dense MANET deployment scenarios if compared with the time interval to complete the election protocol operations. This consideration has led us, at first, to adopt the simplifying assumption of fixed nodes, by approximating the actual deployment environment with a stationary ad-hoc wireless network. However, also the REDMAN simple protocol for manager election can tolerate node mobility to some extent. First, let us observe that the value of each explored IN is determined exclusively by anyone of its farthest nodes. Thus, within each exploration round, it is necessary that only the current IN and one of its farthest nodes do not change their distance in the network topology. Moreover, since our middleware does not assume that participant nodes support MANET-specific routing protocols to cooperate in REDMAN identification and manager election, it is necessary that all nodes placed along the path from the current IN to farthest nodes remain within mutual communication range at least for the duration of a single protocol round. The number of nodes involved in that condition is very low with regards to the overall number of dense MANET participants. All these assumptions are largely acceptable because, even in very large dense MANET deployment scenarios with several hundreds of nodes, each round of the election protocol lasts for less than a few seconds.

After the end of the election process, REDMAN works to ensure that node mobility does not degrade too much the central position of the elected manager. With the goal of limiting the protocol overhead, REDMAN proposes a lazy strategy that allows short time intervals when the manager could have slightly moved from the dense MANET topologic center. In particular, REDMAN combines two different strategies against manager assignment degradation, one proactive and one reactive, which operate, respectively, over large and medium time periods (*Tp* and *Tr* with *Tp>>Tr*). The proactive maintenance strategy establishes that the current manager always triggers a new manager election after *Tp* seconds. In addition to probabilistically improving the centrality of the manager position, the periodical re-execution of the election process contributes to distribute the burden of the role among different nodes, thus avoiding to deplete the energy resources of a single participant. Moreover, let us rapidly observe that only the nodes located in the proximity of the dense MANET topology center have high probability to assume the manager role. Therefore, a *targetINvalue* can be easily determined equal to the *INvalue* of the current manager, thus speeding up manager election and reducing the protocol overhead.

In addition to the above proactive degradation counteraction, REDMAN exploits a reactive strategy that consists in repeating the farthest node determination at regular *Tr* periods, with the goal of understanding the current manager distance from the optimal placement. If the distance of manager farthest nodes, i.e., its *newINvalue*, has increased if compared with the distance estimated at the moment of its election, i.e., its *INvalue*, REDMAN realizes that the manager has moved from the topologic center in a significant way; in that case, the manager itself triggers a new election process.

The following subsections detail the adopted heuristics 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-based approach that has experimentally demonstrated to reach high quality solutions, close to the optimal choice of the manager (see Section 6).

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 (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 INvalue. 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 REDMAN election protocol.

```
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.

4.2.2. *Determination of Farthest Nodes*

Let us formally define *farthest nodes*, with respect to a node d_k in the dense MANET, the set $FN(d_k)=\{d_{f0}, \ldots, d_{fF}\}$, where:

- $\forall i \in [0, F]$, $d_{fi} \in DM(n)$, and
- $\forall d_j \in DM(n)$, $sp(d_k,d_{fi}) \geq sp(d_k,d_j)$

where $sp(d_x,d_y)$ is the length of the shortest path that connects the nodes d_x and d_y.

REDMAN proposes a simple broadcast-based strategy to detect the length of 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 the dense MANET identification protocol. Figure 5 shows the message propagation from node I (the current IN) to all other nodes in the dense MANET. 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 regards to I.

Let us observe that the farthest node identification protocol works as if all dense MANET nodes were fixed during the determination of $FN(d_k)$ and, consequently, usually reaches an approximated non-optimal solution. However, given the limited time interval needed to complete the $FN(d_k)$ set determination, that stationary assumption has demonstrated to be acceptable and to permit the set identification with adequate accuracy for the lightweight and lazily-consistent REDMAN form of replication.

Given that the protocol is flooding-based, it could incur in the broadcast storm issue, as explained in Section 4.1. However, state-of-the-art approaches to avoid broadcast storm, such as gossip-based strategies, are not applicable to the REDMAN farthest node identification protocol because in REDMAN there is the need to determine shortest paths of maximum length between investigated farthest nodes and the current IN [15]. Consequently, it is mandatory to adopt solutions that permit the identification of shortest paths: it is easy to demonstrate that traditional flooding complies with this property, while gossip-based flooding does not. At the moment, similarly to what described for dense MANET identification, REDMAN simply adopts a trivial technique based on delaying message broadcasts of a random time interval. We are currently working to experimentally evaluate the effectiveness of storm-prevention flooding techniques similar to [16] when applied to REDMAN farthest node identification and to possibly integrate them in the next release of our middleware prototype.

By delving into finer technical details about the REDMAN protocol solution, 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 a 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 determination message include not only the actual farthest nodes for the current IN, but also other nodes situated at the dense region boundaries. Figure 6 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 do not reply because they are prevented by single-hop neighbors placed at greater distance, e.g., nodes E, F, and D in the case of farthest node determination for node A. 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 ones. The IN assumes the distance of the determined farthest node(s) as its INvalue.

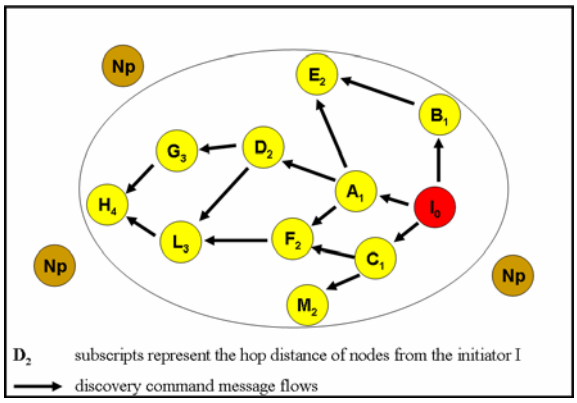

Figure 5. The current IN (I) broadcasts exploratory messages for manager election.

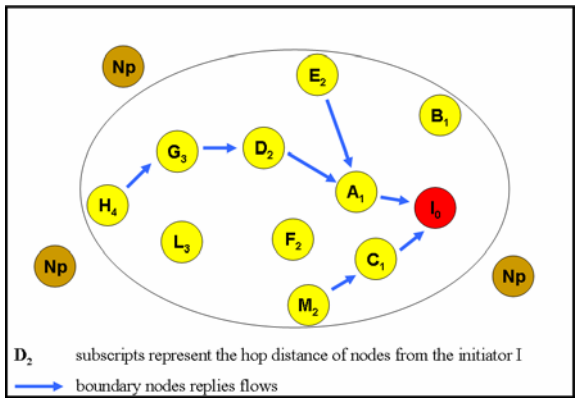

Figure 6. Only nodes at dense MANET boundaries (E, F, H) reply to the current IN.

5. The REDMAN High-Layer Facilities

On top of the DMC layer, REDMAN provides a high layer of facilities that exploit dense MANET configuration mechanisms to provide replica distribution, replica retrieval, and replica degree maintenance, as described in the following three sub-sections.

5.1. Replica Distribution

The RD facility operates to transparently distribute resource replicas in the dense MANET. When a delegate enters a dense region, it communicates the descriptions of its shared resources, represented according to the Resource Description Framework (RDF) [17], to the replica manager, which autonomously decides the values of some parameters that influence the resource replication process.

First, the manager establishes the target replication degree (*deg)*, depending on the delegate suggestion included in the resource RDF-based description. More sophisticated and conservative strategies to choose the proper resource replication degree, e.g., based on a lightweight and approximated estimation of the current number of nodes in the dense MANET with enough storage resources to store a replica, are currently under investigation. In addition, to limit the problem of uselessly consuming the storage of cooperating nodes, the manager also specifies a suggested replica expiration time (*expTimer*); after that time interval, resource delegates can autonomously decide to discard their hosted copies. Finally, the manager associates the resource to be replicated with a third parameter, the replication hops (*repHops*), better explained in the following. Then, the manager creates new SRT entries for new resources in the dense MANET and commands the delegate to start the replication operations. Each SRT entry contains the replication degree to be lazily enforced and weakly consistent information about replica placement.

When a delegate receives the manager reply, it starts the resource distribution according to a distributed gossip-based strategy, driven by the three parameters above. First, the delegate forwards a replication message (with a copy of the resource and its RDF-based description with the three distribution parameters) to one randomly chosen neighbor node. Let us point out that the DMC facility provides the delegate with a lazily-consistent view of its neighbor set (see Section 4.1). Then, the replication message is randomly forwarded node-by-node at least *repHops* times before a new replica is placed. This favors a scattered distribution of replicas in the dense MANET, by avoiding that many copies are adjacently placed on neighbor nodes. When a node receives a replication message already forwarded *repHops* times, it has to decide whether to collaborate on

replica sharing by hosting a local copy of the resource. In the affirmative case, it establishes the expiration date of the replica depending on both *expTimer* and its local preferences, and notifies the replica manager. Consequently, the manager updates its information in the SRT. Then, the new delegate decreases the replication degree *deg* and, if other replicas are required, forwards the replication message. On the contrary, if the potential delegate is not willing to host a resource copy, e.g., because of limited storage space or scarce battery power, it simply forwards the replication message, with no modifications on the carried distribution parameters.

5.2. Replica Retrieval

The RR facility has the goal of retrieving resource replicas at provision time on the basis of the resource RDF-based descriptions, i.e., to dynamically determine the IP address of one suitable node hosting the requested resource and the unique name of the replica on that node. Resource retrieval is a hard task in MANETs, where a static infrastructure is not continuously available, thus preventing the usage of a fixed centralized lookup service known by all participants [18]. The usage of a single centralized repository with replica placement information is not viable: i) a large number of requests could overwhelm the single point of centralization; ii) the repository would represent a single point of failure; and iii) the repository should be updated with strict consistency requirements not to hinder resource accessibility.

In most deployment scenarios, it makes sense to improve the RR performance by paying the overhead of disseminating Information about Replica Placement (IRP), i.e., information about the position of resource delegates, to a subset of nodes belonging to the dense MANET. Let us briefly recall that, according to the RD strategy, a REDMAN replication message reaches the node that is going to become a new resource delegate only after having traversed *repHops* nodes, which are currently located along the path between the replica-disseminating delegate and the new potential one. The REDMAN primary guideline to improve RR effectiveness is to exploit that message forwarding process to distribute also the current IRP data of the replicated resource. In other words, some dense MANET nodes participating to RD message forwarding are required to store the IRP traversing them, together with the time of the associated replication message reception, in a local table (*IRPTable*). When replicas are discarded after *expTimer* or when delegates fail/exit the dense MANET, IRP data is not consistently updated to limit the middleware overhead. In the case of stale IRP information retrieved by a client, simply the client is responsible for re-starting the RR process. The only mechanism to discard possibly stale IRP data supported in REDMAN is the automatic removal of *IRPTable* entries older than *expTimer*.

That very simple strategy permits both to store IRPs on a limited number of nodes and, at the same time, to limit message overhead during both IRP distribution and resource retrieval. In fact, thanks to REDMAN IRP dissemination, even to a limited number of hosts (see Section 6.5), replica searchers can probabilistically achieve one suitable IRP within a few hops. In most cases, this is sufficient to prevent flooding the dense MANET with RR packets.

Based on distributed IRP data, the REDMAN RR solution proposes a simple “expanding ring search” retrieval strategy, similar to that employed in [19] to find a route toward a destination, by limiting control messages and achieving good scalability. RR sends search packets with an increasing Time To Live (*TTL*): first, a searcher client floods RR packets within its neighborhood (*TTL*=1); then, in the case that no neighbors reply with the requested IRP data, the searcher repeats RR flooding by increasing the message *TTL*. The process goes on until any useful IRP data is returned back to the searcher.

Let us observe that a critical issue is the decision of how many IRPs should be distributed for a resource at the moment of its replica dissemination. REDMAN permits to configure that number, by enabling to trade the memory/network overhead of IRP distribution against RR performance at provision time; the growth of IRP diffusion costs corresponds to a decrease of runtime RR costs. The choice of the most appropriate number of distributed IRPs mostly depends on the characteristics of the supported applications and of the deployment scenario (primarily, RR time requirements and expected ratio between the number of resource requests and of replica instantiations); secondary elements to consider are the frequency with which replicas expire, the size of RR messages, and the size of IRPs.

The REDMAN RR solution presented in this section is simple, with sufficiently good performance and limited overhead, and is the one currently implemented in the REDMAN prototype. However, we are investigating and evaluating more sophisticated and effective (but still decentralized and lightweight) solutions for RR. Primary guidelines of our on-going research work on RR are: i) how to distribute IRP data over the dense MANET in an almost uniform way from the topology point of view; and ii) how to disseminate IRP data along straight directions so that searchers can rapidly come across them by diffusing search messages along another straight direction, orthogonal to the one of IRP data diffusion. The analysis and comparison of alternative advanced RR protocols is out of the scope of the paper; some preliminary results can be found in [20].

5.3. Replica Degree Maintenance

Proactive RDM solutions available in the literature, such as [7], do not fit the provisioning environments addressed by REDMAN. In fact, proactive RDM approaches usually require GPS-equipped wireless nodes that continuously monitor their mutual positions to foresee network exits, thus producing non-negligible network/computing overhead. REDMAN RDM, instead, implements a reactive solution with very low communication overhead by relaxing the constraint of anytime perfect consistency in the number of available replicas.

After the initial replica distribution phase, the lightweight RDM facility works to maintain unchanged the replication degree decided for each shared resource, without guaranteeing strict 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 aims at maintaining unchanged the replication degree only by reacting to resource delegate movements/failures.

When a delegate realizes it is going to exit the dense MANET, it autonomously offloads its shared resources on neighbors that are still within the dense region; in their turn, these new delegates communicate the occurred change to the replica manager. On the contrary, if a delegate does not succeed in foreseeing its exit from the dense region and realizes to be already out of it, it tries to notify the replica manager by specifying its hosted resources. Once the manager receives the notification, it commands other delegates for those resources in the dense MANET to distribute new replicas. Finally, when a delegate either fails or leaves the network by abruptly interrupting all its connections, to achieve scalability and to limit overhead, REDMAN accepts a temporary inconsistency in the replication degree. Only at large time intervals, delegates are required to confirm their presence to their associated manager, by sending an updated list of their shared resources. In that way, the manager can lazily re-establish the replica degree consistency by commanding still alive delegates to distribute new replicas only when some delegate update messages are missing.

6. Experimental Results

To evaluate the effectiveness of our approach, we have implemented a REDMAN prototype and extensively simulated the behavior of REDMAN solutions for dense MANET identification, manager election, and gossip-based resource retrieval in the NS2 simulator [21]. The simulations have three main goals: i) to determine the network overhead of the proposed protocols in terms of the number of messages 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. For the sake of simplicity, the reported results refer to deployment scenarios where all nodes are equal from the point of view of locally available resources, such as battery power and storage space.

The code of the REDMAN prototype, the full algorithmic description of REDMAN protocols, the exhaustive list of all NS2 simulation parameters used, and a wide set of additional performance figures are available at the REDMAN Web site http://www.lia.deis.unibo.it/Research/REDMAN/

6.1. DMC 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 DMC 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 messages sent by each participant, over a set of more than 1,000 simulations.

The results reported in Figure 7 are normalized to the number of nodes actively participating in the 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 REDMAN solution for manager election, also in this case the number of sent messages is very limited and grows very slightly with the number of participants. In fact, sent messages 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 12 hops for the 550-node case) and grows less than the number of participants.

Given the dependence of manager election network overhead on the number of iterations, we have carefully investigated the convergence of the REDMAN protocol while varying the number of dense MANET nodes (see Figure 8). 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 experimental results about the number of iterations have demonstrated the almost linear dependence on dense MANET diameter and have shown to be negligibly affected by the choice of the initiator node.

In summary, the experimental results about REDMAN DMC 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 in Figure 7 is due to different factors. First, the dense MANET diameter only increases in correspondence with some threshold values of the number of participants, as discussed and experimentally shown in Section 6.3, 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 additional performance results and comments at the REDMAN Web site).

About the latency imposed by the REDMAN manager election protocol, let us briefly observe that it mainly depends on three factors: obviously, one is the number of dense MANET participants; the others are two REDMAN configurable timers that establish, respectively, the message delay for preventing broadcast storm in farthest node determination and the time interval waited by the current IN before delegating its role. All the presented results have been obtained by setting the former to 2.5s and the latter to 5s per hop of the diameter. These values guarantee that the current IN passes node exploration responsibility only after having received most replies from farthest nodes, thus achieving its correct INvalue.

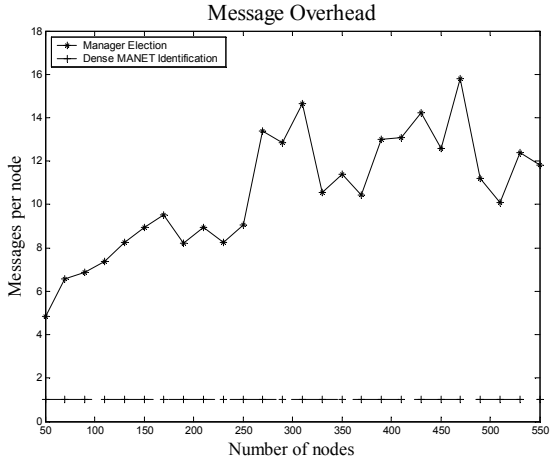

Figure 7. Messages sent/received per node in dense MANET identification and manager election.

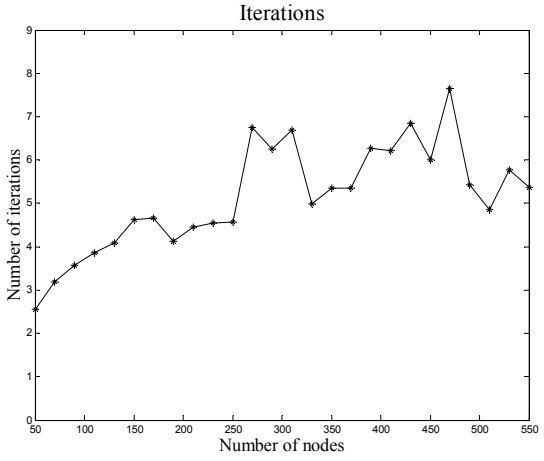

Figure 8. Number of iterations needed for the REDMAN manager election protocol.

6.2. Manager Election Inaccuracy

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 election inaccuracy defined as the hop distance between the manager chosen by the REDMAN protocol and the actual optimal solution.

The results in Figure 9 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.

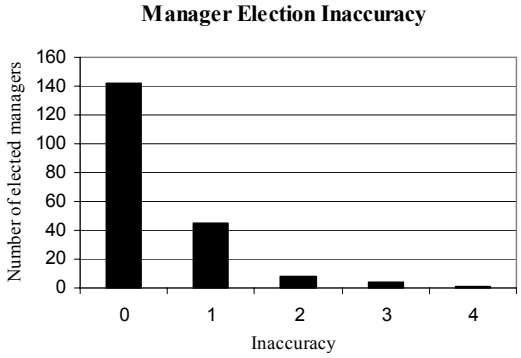

Figure 9. Inaccuracy of the REDMAN manager election protocol.

6.3. Impact of REDMAN Heuristics on Manager Election Accuracy

A decisive parameter affecting the results achieved by the REDMAN manager election protocol has demonstrated to be *MaxConsecutiveEqualSolutions*. Let us recall that this parameter influences the protocol termination, by permitting to stop the

iterations when the specified number of best solutions has already been explored. However, the *INvalue* of the elected IN (*electedINvalue*) could not be still the optimum achievable in that network (*optimalINvalue*).

To determine whether the *electedINvalue* obtained with low values of *MaxConsecutiveEqualSolutions* is acceptable, we have run a large set of simulations in the deployment scenarios described in Section 6.1. Figure 10 plots the average *electedINvalues* obtained over 30 different runs for each scenario, for values of *MaxConsecutiveEqualSolutions* ranging from 2 to 4. The results are compared with two reference traces: the lower represents the *optimalINvalue* in each scenario; the higher the *optimalINvalue* multiplied by 1.25, thus depicting a tolerance strip of 25% from *optimalINvalue*.

As expected, the figure shows that the difference between *electedINvalues* and *optimalINvalues* grows as *MaxConsecutiveEqualSolutions* decreases. However, even for the lowest tested value (*MaxConsecutiveEqualSolutions*=2), most points lie below the higher reference trace. This means that REDMAN achieves solutions not too far from the *optimalINvalue* also for very low values of *MaxConsecutiveEqualSolutions*. For that reason, all the other experimental results in the section refer to simulation runs where *MaxConsecutiveEqualSolutions* has been set to 3.

Let us briefly observe that manager election accuracy is also influenced by *DesiredAccuracy*, the other configurable parameter in the REDMAN election heuristic. For the sake of briefness, we have decided not to report also experimental results interest about *DesiredAccuracy*, which are of minor interest: simulations have confirmed that election accuracy linearly depends on that parameter, as intuitively expected. In all experimental results in the section, we have used *DesiredAccuracy*=100%.

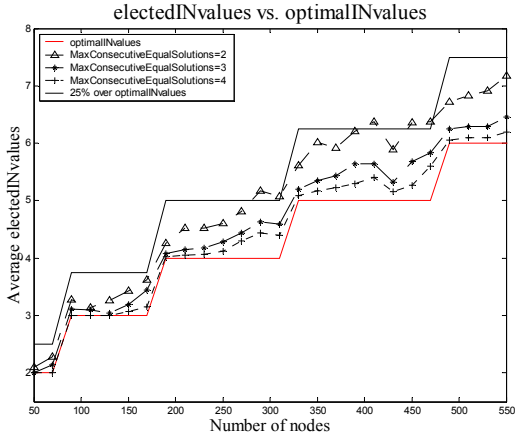

Figure 10. Accuracy of the manager election protocol as a function of *MaxConsecutiveEqualSolutions*.

6.4. Impact of Node Mobility on the Accuracy of the REDMAN Dense MANET Identification Protocol

To test the robustness of REDMAN solutions, we have evaluated the accuracy of 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, the sub-section focuses on the behavior of our dense MANET identification protocol as a function of node mobility.

We have considered the same 110-node scenario of the tests in Section 6.1. 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 arrival/departure in/out the dense MANET does not affect at all the behavior of the REDMAN identification solution.

Figure 11 reports the dense MANET identification inaccuracy, defined as the difference between the number 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 over 30 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 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.

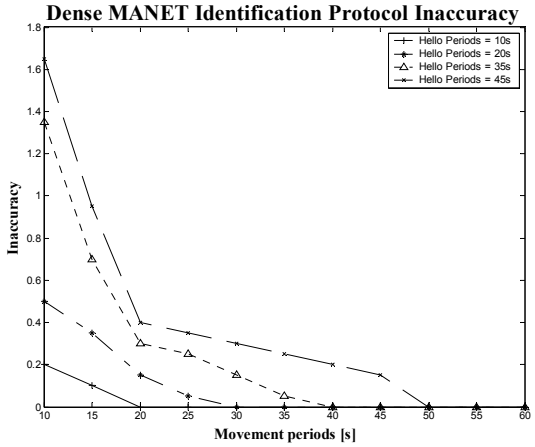

Figure 11. Accuracy of the REDMAN dense MANET identification protocol as a function of node mobility.

6.5. Replica Retrieval Message Overhead

We have also performed a wide set of simulations to evaluate the message overhead of the REDMAN RR strategy, i.e., the overall number of search messages exchanged within the dense MANET, on the average, when a requester looks for the IP address of a node hosting a replica of the needed resource. To this purpose, we compare the REDMAN RR results on message traffic with the network overhead that would be generated by a trivial retrieval strategy based on flooding query messages to all nodes belonging to the dense MANET (in the following we call that strategy Query Flooding – QF).

Figure 12 reports the REDMAN RR plot as a function of the *repHops* parameter (the configurable number of IRPs disseminated in the dense MANET) in the most populated scenario of 550 nodes. The figure shows the average number of search messages necessary to find the first IRP, normalized to the number of dense MANET participants. As expected, the number of search messages per node decreases as the number of disseminated IRPs grows. But, most important, even for a very limited number of IRPs, the REDMAN expanding ring RR strategy largely outperforms QF, by providing a suitable trade-off between RR complexity, provision-time RR traffic, and IRP distribution overhead.

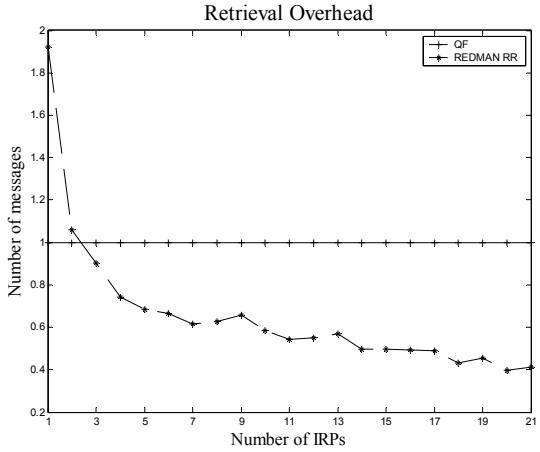

Figure 12. Message overhead for resource retrieval depending on the number of disseminated IRPs.

7. REDMAN Security Concerns

REDMAN is supposed to operate in open environments and, as any other MANET middleware/application, potentially presents several security issues [22-24]. Even if not the primary focus of our research, after having demonstrated the feasibility and effectiveness of the REDMAN approach, we are now analyzing the main security weaknesses affecting our solution and investigating how to provide lightweight countermeasures against these issues.

By passing over link-layer security concerns, such as denial of service threats leveraging traffic jamming (which frequency hopping or spread spectrum techniques can eliminate), we have mainly identified two different protocol layers suffering from potential security problems. On the one hand, DMC protocols (in particular, manager election) undergo the same security issues of any MANET network-layer solution, such as in the case of routing. On the other hand, RD/RR strategies are prone to the security concerns typical of any peer-to-peer system for content distribution. In the following, we rapidly discuss the main characteristics of primary security concerns specific to dense MANET replication and identify possible guidelines of solution that are currently under investigation.

About the DMC layer, the main security issue is represented by malicious participants aiming at either becoming managers or saving their own battery/network resources. For instance, a malicious node M can force its election as manager by forging replies with a fake (higher) hop-counter during the REDMAN farthest node identification protocol. In that case, the current IN is likely to delegate the exploration to its neighbor placed in the M direction. M can repeat the same procedure until it becomes an IN, and then can falsely assume the manager role. In the case of non-colluding attackers, misbehavior detection strategies can easily face up with this problem, as deeply investigated within ad-hoc routing research. Several solutions in the literature are based on passive acknowledgement, i.e., on the implicit control by each participant of the behavior of its successor through channel overhearing [25], even if there are well-known situations where the approach does not work properly [26]. Once identified a manager election attack, the culprit needs to be penalized to discourage further security attacks. That raises additional trust issues since it can be difficult to decide whether the accuser or the accused is the actual rogue. To this specific purpose, many solutions have been proposed [27], also based on reputation methods [26].

Selfish node behavior represents another potentially security issue for REDMAN [28]: nodes could aim at saving their battery, by refusing to forward farthest node detection packets or by forging distant node replies with a fake (lower) hop-counter, thus avoiding to be chosen as the next explored IN. Let us observe that the high node density of dense MANETs makes the problem less critical because there is sufficient redundancy in link connectivity to guarantee a correct protocol behavior anyway, at least when the number of simultaneous attackers is limited. Moreover, also in this case, overhearing techniques can help in defending from the attack. An additional approach to overcome selfish behavior is represented by incentives, sorts of virtual monetary rewards to stimulate and motivate user cooperation. For instance, intermediate nodes could be refunded for the energy dissipated in packet forwarding. [29] presents a tamper-resistant device in charge of maintaining a virtual wallet, by also preventing from selfish money forgeries. An auction-based solution aiming at establishing how much a node should pay for a MANET service is presented in [30].

About the security concerns of high-layer REDMAN protocols, RD/RR/RDM solutions are subject to well-known content distribution and peer-to-peer attacks, widely investigated for traditional wired-network deployment scenarios [31]. With regards to resource distribution and retrieval, malicious intermediate nodes could exhibit a selfish misbehavior: they could not forward replica distribution/retrieval packets or could not store replicas when designated as resource delegates. Security solutions similar to the ones explained above can deal with the former attack. Again, let us observe that flooding-based REDMAN RR benefits from path redundancy within the dense MANET. The latter problem of "node storing quotas", i.e., the selfish behavior consisting in exploiting the storage of other participants while not lending one's own, has been addressed in many research works. For instance, [32] proposes to equip each device with an anti-tampering smartcard.

Additional problems relate to resource access control and authorization [31], by taking into account also the digital rights that replicated resources could be subject to. While generally disregarded in traditional peer-to-peer protocols, in wired networks these concerns can be solved by trusting a certification authority (CA) for key distribution [33]. In infrastructure-less MANETs, where it is unfeasible to assume CA availability, the problem is harder: recent research is proposing novel and completely decentralized key management systems for these scenarios [24, 34]. Finally, also Digital Rights Management (DRM) has undergone deep research in the last years: [35] proposes DRM architectures for infrastructure-based networks that can be easily adapted to MANET when assuming the temporary availability of wide-range connections, such as GPRS, to off-line download licenses directly from clearinghouses.

8. Related Work

The section organizes the wide state-of-the-art research related to replica distribution and retrieval in wide-scale distributed networks along three primary perspectives. First, we provide an overview of recent research activities on content distribution in MANETs. By following the main focus of the article, the second part presents solutions that address DMC-similar issues: since, to the best of our knowledge, REDMAN is original in determining which MANET nodes belong to a dense area, we concentrate our discussion on leader election solutions in multi-hop MANETs. Finally, the last part examines distributed resource retrieval issues in infrastructure-less environments, by surveying solutions for MANETs and wireless sensor networks.

8.1. Content Distribution in MANETs

The continuous growth of Internet traffic has promoted the idea of increasing availability by replicating service contents depending on popularity [36, 37]. Resource replication is even more crucial in MANETs where continuous node availability is unfeasible: replication is essential both to maintain resource availability in the case of MANET partitioning and to reduce access latency and battery consumption by placing replicas close to requesters. However, due to the novelty of MANET service provisioning, a very few approaches have already emerged.

A couple of proposals address latency reduction for resources on the fixed Internet and accessed by MANET nodes. [38] describes a cooperative strategy for caching Web contents in wireless localities composed by mobile terminals with also cellular-phone capabilities. A similar goal is addressed in [39], which tends to decrease access latency of Web Services and to optimize energy consumption at the same time: each participant maintains a local cache with recently accessed Web Services components; when searched components are not in cache, the requester first explores the ad hoc network and, as the last chance, exploits long-range telecom connectivity.

In very dynamic MANET environments it is unfeasible to assume that long-range base stations, e.g., connecting to GPRS/UMTS core networks, are always available. In any case, multi-hop packet forwarding exacerbates MANET networking issues, especially when exploited to download multimedia flows and large files [40]. SPAWN proposes a cooperative strategy for content delivery in vehicular ad hoc networks where gateways are only intermittently available [41]. Client nodes start downloading files from gateways installed on freeway service stations while the client is within the gateway coverage area. If not yet terminated before losing the gateway connectivity, the download can continue with a peer-to-peer strategy: SPAWN leverages gossip-based control message exchanges to discover cars that own missing file chunks; strategies based on chunk rareness and replica distance are used to select which file pieces to download first.

By considering pure MANETs, Chen et al. propose a well-known solution to counteract network partitioning [7]. They place resource replicas on the basis of node positions/movements, by assuming, differently from REDMAN, that all nodes are GPS-equipped and aware of their physical positions. To the best of our knowledge, [42], [43], and [44] are the research proposals more similar to REDMAN from the point of view of content distribution. [42] enhances data availability via an adaptive replication protocol suitable only for deployment scenarios with nodes in direct single-hop visibility. In addition, it assumes that any node knows the position of all replicas and the memory/battery/mobility state of all other nodes. [43], instead, proposes proactive replication depending on resource access frequency; however, its coordination and synchronization solutions do not scale well in wide deployment scenarios, thus making it unsuitable for REDMAN-addressed dense MANET scenarios. Finally, Cao et al. propose collaborative caching of nodes along client-to-server paths during resource forwarding with the goal of reducing resource access latency [44]. Due to the number of alternative paths connecting any node pair, the solution is not effective when applied to unstructured, non-hierarchical, and wide-scale MANETs such as dense ones. If compared with the above solutions, REDMAN is original in addressing a multi-hop and dense MANET scenario composed by resource-limited devices. That calls for the design of lightweight replica distribution/retrieval and maintenance protocols, whose main goal is not to counteract network partitioning or to decrease access latency, but to increase resource availability, notwithstanding node mobility and with a very limited coordination overhead.

8.2. Leader Election

Leader election is a largely investigated research issue in both traditional wired networks [45] and single-hop ad-hoc wireless ones [46, 47]. Only a few recent research efforts have addressed leader election in the context of multi-hop MANETs. To the best of our knowledge, all proposed protocols address leader election as an “extrema finding” problem, i.e., they aim at electing the node with the maximum identifier, where identifiers can also depend on local node capabilities (processing-power, battery state, …) and local topology (number of neighbors), regardless of its current position. REDMAN deals with the manager election problem from a completely different perspective: the goal is to determine suitable managers by only evaluating the node topologic position within the dense MANET, without imposing any additional positioning hardware constraint on participating nodes.

Four very relevant approaches emerge in the state-of-the-art research on MANET “extrema finding” leader election [48-51]. [48] assumes that nodes are equipped with positioning devices and move in a three-dimensional space quantized in adjacent polyhedrons, whose size is determined to have all mobile hosts in one polyhedron at single-hop distance the one from the other. The proposed election algorithm exploits node movements to opportunistically exchange election-related information: in particular, one node can eliminate another participant in the competition to become leader when they are in the same polyhedron.

[49] takes into consideration possible MANET partitioning and merging and proposes a novel formulation of the election problem as “Any [network] component whose topology is static sufficiently long will eventually have exactly one leader”. In this perspective, it proposes two algorithms, based on the TORA routing protocol [52], that create a virtual leader-oriented Directed Acyclic Graph. The first algorithm can manage only single topology changes, i.e., it does not allow any change in the MANET topology before terminating the process of recovering from the previous modification; it reacts to MANET partitioning by declaring leader the node that detected the partitioning; in the case of network merging, instead, the leader of the subnet with the smallest identifier becomes the leader of the merged MANET. The second algorithm introduces modifications to relax the single change assumption, but formal proofs of correctness are given only for the first solution.

An original tree-based self-stabilizing algorithm has been proposed in [50]. It aims at identifying the node in the MANET with the highest value according to a metric that considers node performance and characteristics. A source node starts the election by diffusing a message to create a spanning tree: as leafs are identified, they propagate their values to their parents; parents, in their turn, propagate upstream, up to the source node, the maximum value of nodes belonging to their sub-trees. Periodic probe exchanges between neighbors and heartbeats from the leader are exploited to detect and counteract node mobility.

All the above solutions for MANETs do not take into consideration malicious or selfish behavior of participant nodes. Only a recent work presents some cheat-proof algorithms for synchronous systems, with the goal of choosing the best-performance-valued node in a network [51]. [51] assumes that all participating nodes own two certificates, one binding the node identity to its public key, the other binding performance indicators to the node identity. The proposal exploits a round-based bottom-up approach to build a node hierarchy: at round k, undefeated candidates broadcast challenge-messages to other undefeated nodes at k-hop-distance; any loser assumes its winner as its parent in the hierarchy, while winners carry on the protocol until a unique leader is chosen. Challenge-messages with sender performance indicators are hashed and signed with the sender's private key to guarantee transmission integrity and authenticity.

The above cited research projects include formal proofs of correctness/termination and specifically focus on leader election. A number of other solutions for MANETs, instead, include election protocols as a secondary aspect of their proposals, often by relaxing the constraint of leader uniqueness [53-55]. In particular, network clustering calls for original solutions for cluster-head designation [56, 57], especially in the context of wireless sensor networks [58-60].

8.3. Resource Retrieval

Infrastructure-free and completely decentralized MANETs pose innovative challenging issues also for resource retrieval. A very few research activities have investigated MANET-specific broadcast-based peer-to-peer solutions for resource retrieval, with limited scalability and excessive overhead for resource-constrained clients [61]. Some recent proposals aim at limiting broadcast communications by exploiting quorum-based solutions [62-64]: the primary idea is to disseminate resource placement information on a node subset that can be determined without message flooding.

Some interesting research results on resource retrieval have been achieved in the partially similar deployment scenario of stationary wireless sensor networks. [62] assumes GPS-equipped nodes to spread advertisement/search packets along orthogonal directions. On the contrary, [63] considers provisioning environments where geographic routing cannot apply and proposes the dissemination of placement data along paths with approximately constant directions, determined by choosing, as the next hop, a node that is not the neighbor of any node belonging to the already built sub-path.

Some of the above work is suggesting solution guidelines that can also apply to MANET environments. [64] extends the GPS-based solution in [62] for replication in mobility-enabled deployment scenarios: each node is asked for storing placement data about its nearest replica. A different approach is presented in [65]: any resource associates with its origin place, which is in charge of distributing all replicas to nodes placed at k-hop distance from it. In that proposal, replica retrieval exploits geographical routing to forward queries toward the resource origin place.

Most referred solutions exploit GPS to disseminate and retrieve resources in MANET. [61] and [63] are the approaches most similar to the REDMAN RR proposal, by not requiring any positioning system. To the best of our knowledge, however, the usage of gossip-based replication to favor expanding ring searches represents a simple and original proposal in MANET replica retrieval, by fitting the REDMAN goals of simplicity and limited coordination.

9. Conclusive Remarks

The challenge of supporting distributed services over dense MANETs can benefit from the assumption of high node population to enable lazily consistent forms of replication for read-only resources of common interest. This can significantly increase resource availability notwithstanding unpredictable node movements in/out dense MANETs. The REDMAN project demonstrates how it is possible to provide middlewares for lightweight and effective replica management: REDMAN protocols impose a limited overhead in terms of exchanged messages and rapidly converge to high quality solutions even in very large-scale deployment scenarios.

The performance 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. In particular, we are carefully evaluating in which deployment conditions (dense MANET diameter and number of participants, node mobility, resource distributions/requests ratio, average replica size, …) the traffic reduction due to the central position of the replica manager offsets the overhead incurred in its election. In addition, we are going to extensively test our J2ME REDMAN prototype in a campus-wide dense MANET environment consisting of a number of PDAs that exploit IEEE 802.11b connectivity in ad-hoc mode. Finally, as already stated in previous parts of the paper, we are working on more sophisticated replica retrieval solutions, which also take into account simple security mechanisms for authorized resource access and incentive-based promotion of node collaboration.

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 Journal on Ad Hoc Networks*, July 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] P. Bellavista, A. Corradi, E. Magistretti, "Lightweight Replication Middleware for data and Service Components in Dense MANETs", *1st IEEE Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2005.

[10] 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.

[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] C.E. Perkins, J.T. Malinen, R. Wakikawa, E.M. Belding-Royer, Y. Sun, "IP Address Autoconfiguration for Ad Hoc Networks", Internet Engineering Task Force, Zeroconf Working Group, Jul. 2000.

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

[15] 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.

[16] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, L. Viennot, "Optimized Link State Routing Protocol", *IEEE Int. Multi Topic Conference (INMIC)*, Dec. 2001.

[17] S. Decker, P. Mitra, S. Melnik, "Framework for the Semantic Web: an RDF Tutorial", *IEEE Internet Computing*, Vol. 4, No. 6, Nov.-Dec. 2000.

[18] wwws.sun.com/software/jini

[19] S.-J. Lee, E. Belding-Royer, C. Perkins, "Scalability study of the ad-hoc on-demand distance vector routing protocol", *Wiley Int. Journal of Network Management*, Vol. 13, No. 2, Mar. 2003.

[20] P. Bellavista, A. Corradi, E. Magistretti, "Comparing and Evaluating Lightweight Solutions for Replica dissemination and Retrieval in Dense MANETs", *10th IEEE Int. Symp. on Computers and Communications (ISCC)*, July 2005.

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

[22] F. Stajano, R. J. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks", *7th Int. Workshop on Security Protocols*, Apr. 1999.

[23] L. Zhou, Z. Haas, "Securing Ad Hoc Networks", *IEEE Network*, Vol. 13, No. 6, Nov.-Dec. 1999.

[24] J.-P. Hubaux, L. Buttyan, S. Capkun, "The Quest for Security in Mobile Ad Hoc Network", *ACM Symp. on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Oct. 2001.

[25] S. Buchegger, C. Tissieres, J.-Y. Le-Boudec, "A Test-Bed for Misbehavior Detection in Mobile Ad-Hoc Networks – How Much Can Watchdogs Really Do?", *6th IEEE Wireless Mobile Computing and Systems and Applications*

(WMCSA), Dec. 2004.
[26] P.-W. Yau, C. J. Mitchell, "Reputation Methods for Routing Security for Mobile Ad hoc Networks", *Joint IST Workshop on Mobile Future and Symp. on Trends in Communications (SympoTIC)*, Oct. 2003.
[27] K. Paul, D. Westhoff, "Context Aware Detection of Selfish Nodes in DSR based Ad-hoc Networks", *IEEE Global Telecommunications Conf. (GLOBECOM)*, Nov. 2002.
[28] S. Marti, T. J. Giuli, K. Lai, M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks", *6th ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, Aug. 2000.
[29] L. Buttyan, J.-P. Hubaux, "Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks", *ACM/Kluwer Mobile Networks and Applications (MONET)*, V. 8, N. 5, Oct. 2003.
[30] K. Chen, K. Nahrstedt, "iPass: an Incentive Compatible Auction Scheme to Enable Packet Forwarding Service in MANET", *24th IEEE Int. Conf. on Distributed Computing Systems (ICDCS)*, Mar. 2004.
[31] S. Adroutselli-Theotokis, D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies", *ACM Computing Surveys*, Vol. 36, No. 4, Dec. 2004.
[32] P. Druschel, A. Rowstron, "PAST: A Large-scale, Persistent Peer-to-peer Storage Utility", *8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, May 2001.
[33] N. Daswani, H. Garcia-Molina, B. Yang, "Open Problems in Data-Sharing Peer-to-Peer Systems", *9th Int. Conf. on Database Theory*, Jan. 2003.
[34] N. Asokan, P.Ginzboorg, "Key Agreement in Ad Hoc Networks", *Computer Communication Review*, Vol. 23, No. 17, Nov. 2000.
[35] W. Liu, R. Safavi-Naini, P. Sheppard, "Digital Rights Management for Content Distribution", *Australasian Information Security Workshop (AISW)*, Feb. 2003.
[36] M.J. Freedman, E. Freudenthal, D. Mazières, "Democratizing Content Publication with Coral", *1st USENIX/ACM Symp. on Networked Systems Design and Implementation*, Mar. 2004
[37] http://bittorrent.com/.
[38] F. Sailhan, V. Issarny, "Energy-Aware Web Caching for Mobile Terminals", *22nd Int. Conf. on Distributed Computing Systems Workshops*, July 2002.
[39] R. Friedman, "Caching Web Services in Mobile Ad-hoc Networks: Opportunities and Challenges", *2nd ACM Int. Workshop on Principles of Mobile Computing*, Oct. 2002.
[40] K. Tang, M. Gerla, R. Bagrodia, "TCP Performance in Wireless Multi-hop Networks", *2nd IEEE Workshop on Mobile Computer Systems and Applications (WMCSA)*, Feb. 1999.
[41] A. Nandan, S. Das, G. Pau, M. Gerla, M. Y. Sanadidi, "Co-operative Downloading in Vehicular Ad-hoc Wireless Networks", *2nd IEEE Conf. on Wireless On demand Network Systems and Services (WONS)*, Jan. 2005.
[42] M. Boulkenafed, V. Issarny, "A Middleware Service for Mobile Ad Hoc Data Sharing, Enhancing Data Availability", *4th ACM/IFIP/USENIX Int. Middleware Conf.*, June 2003.
[44] T. Hara, "Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility", *20th Int. Joint Conf. IEEE Computer and Communications Societies*, Apr. 2001.
[44] G. Cao, L. Yin, C.R. Das, "Cooperative Cache-based Data Access in Ad Hoc Networks", *IEEE Computer*, Vol. 37, No. 2, Feb. 2004.
[45] J. Villadangos, A. Cordoba, F. Farina, M. Prieto, "Efficient Leader Election in Complete Networks", *13th Euromicro Conf. on Parallel, Distributed and Network-Based Processing (PDP)*, Feb. 2005.
[46] T. Jurdzinski, M. Kutylowski, J. Zatopianski, "Efficient Algorithms for Leader Election in Radio Networks", *21st ACM Symp. on Principles of Distributed Computing (PODC)*, July 2002.
[47] K. Nakano, S. Olariu, "A Survey on Leader Election Protocols for Radio Networks", *IEEE Int. Symp. on Parallel Architectures, Algorithms and Networks (ISPAN)*, May 2002.
[48] K.P. Hatzis, G.P. Pentaris, P.G. Spirakis, V.T. Tampakas, R.B. Tan, "Fundamental Control Algorithms in Mobile Networks", *11th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, June 1999.
[49] N. Malpani, J. L. Welch, N. Vaidya, "Leader Election Algorithms for Mobile Ad Hoc Networks", *4th Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, Aug. 2000.
[50] S. Vasudevan, J. Kurose, D. Towsley, "Design and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks", *12th IEEE Int. Conf. on Networks Protocols (ICNP)*, Oct. 2004.
[51] S. Vasudevan, B. DeCleene, N. Immerman, J. Kurose, D. Towsley, "Leader Election Algorithm for Wireless Ad Hoc Networks", *IEEE DARPA Information Survivability Conf. and Exposition (DISCEX)*, Apr. 2003.
[52] V.D. Park, M.S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", *17th Annual Joint Conf. IEEE Computer and Communications Societies (INFOCOM)*, Apr. 1997.
[53] E.M. Royer, C. E. Perkins, "Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol", *5th ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, Aug. 1999.
[54] X. Hong, M. Gerla, "Dynamic Group Discovery and Routing in Ad Hoc Networks", *1st Mediterranean Ad Hoc Networking Workshop (Med-hoc-Net)*, Sept. 2002.
[55] K. Weniger, M. Zitterbart, "IPv6 Autoconfiguration in Large Scale Mobile Ad-Hoc Networks", *European Wireless*,

Feb. 2002.
[56] M. Gerla, J.T.-C. Tsai, "Multicluster, Mobile, Multimedia Radio Network", *ACM/Baltzer Journal of Wireless Networks*, Vol. 1, No. 3, Mar. 1995.
[57] F.G. Nocetti, J.S. Gonzales, I. Stojmenovic, "Connectivity Based k-Hop Clustering in wireless Networks", *Kluwer Telecommunication Systems*, Vol. 22, No. 1-4, Jan. 2003.
[58] W. Rabiner Heinzelman, A. Chandrakasan, H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", *33rd IEEE Hawaian Int. Conf. on System Sciences (HICSS)*, Jan. 2000.
[59] D. Estrin, R. Govindan, J. Heidemann, S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks", *5th ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, Aug. 1999.
[60] K. Sohrabi, J. Gao, V. Ailawadhi, G. J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network", *IEEE Personal Communications*, Oct. 2000.
[61] A. Helal, N. Desai, V. Verma, L. Choonhwa, "Konark - a Service Discovery and Delivery Protocol for Ad-Hoc Networks", *3rd IEEE Conf. Wireless Communications Networks,* Mar. 2003.
[62] I. Aydin, C.-C. Shen, "Facilitating Match-Making Service in Ad Hoc and Sensor Networks using Pseudo Quorum", *11th IEEE Int. Conf. Computer Communications and Networks*, Oct. 2002.
[63] D. Braginsky, D. Estrin, "Rumor Routing Algorithm for Sensor Networks", *1st ACM Int. Workshop Wireless Sensor Networks and Applications,* Sept. 2002.
[64] J. Tchakarov, N. Vaidya, "Efficient Content Location in Mobile Ad hoc Networks", *IEEE Int. Conf. Mobile Data Management (MDM)*, Jan. 2004.
[65] M. Tamori, S. Ishihara, T. Watanabe, T. Mizuno, "A Replica Distribution Method with Consideration of the Positions of Mobile Hosts on Wireless Ad-hoc Networks", *22nd Int. Conf. on Distributed Computing Systems Workshops*, July 2002.