Mobile Networks and Applications 10, 99–113, 2005

CARD: A Contact-based Architecture for Resource Discovery in Wireless Ad Hoc Networks

AHMED HELMY *, SAURABH GARG and NITIN NAHATA
3740 McClintock Avenue, EEB 232, Electrical Engineering Department, University of Southern California, Los Angeles, CA 90089-2562, USA

PRIYATHAM PAMU
Computer Science Department, University of Southern California, Los Angeles, CA 90089-2562, USA

Abstract. Traditional protocols for routing in ad hoc networks attempt to obtain optimal or shortest paths, and in doing so may incur significant route discovery overhead. Such approaches may be appropriate for routing long-lived transfers where the initial cost of route discovery may be amortized over the life of the connection. For short-lived connections, however, such as resource discovery and small transfers, traditional shortest path approaches may be quite inefficient. In this paper we propose a novel architecture, CARD, for resource discovery in large-scale wireless ad hoc networks. Our mechanism is suitable for resource discovery as well as routing very small data transfers or transactions in which the cost of data transfer is much smaller than the cost of route discovery. Our architecture avoids expensive mechanisms such as global flooding and complex hierarchy formation and does not require any location information. In CARD resources within the vicinity of a node, up to a limited number of hops, are discovered using a proactive scheme. For resources beyond the vicinity, each node maintains a few distant nodes called *contacts*. Contacts help in creating a small world in the network and provide an efficient way to query for distant resources. Using contacts, the network view (or reachability) of the nodes increases, reducing the discovery overhead and increasing the success rate. On the other hand, increasing the number of contacts also increases control overhead. We study such trade-off in depth and present mechanisms for contact selection and maintenance that attempt to increase reachability with reduced overhead. Our schemes adapt gracefully to network dynamics and mobility using soft-state periodic mechanisms to validate and recover paths to contacts. Our simulation results show that CARD is scalable and can be configured to provide desirable performance for various network sizes. Comparisons with other schemes show overhead savings reaching over 93% (vs. flooding) and 80% (vs. bordercasting or zone routing) for high query rates in large-scale networks.

Keywords: energy efficient, sensor networks, routing

1. Introduction

Ad hoc networks are wireless networks composed of mobile devices with limited power and transmission range. These networks are rapidly deployable as they neither require a wired infrastructure nor centralized control. Because of the lack of fixed infrastructure, each node also acts as a relay to provide communication throughout the network. Applications of ad hoc networks include coordination between various units (e.g., in a battlefield), search and rescue missions, rapidly deployable networks, and vehicular networks, among others. Although research on mobile ad hoc networks (MANets) has attracted a lot of attention lately, little attention has been given to resource discovery in large-scale MANets. In addition, a very important mode of communication that has been largely ignored in the ad hoc networks literature is that of short flows and small transactions, where the communication cost of discovering shortest routes is usually the dominant factor (not the data transfer as in long flows). For such short flows reducing overhead (not route optimization) is the main design goal. Current routing protocols in general attempt to discover optimal (shortest path) routes. In our study, instead of obtaining shortest paths, we focus on reducing the overhead of resource (or route) discovery for short flows. Examples of resource discovery and small transfers in ad hoc networks include discovering servers, objects and capabilities (e.g., GPS capable nodes), instant and text messaging, short transactions, DNS-like queries, paging, and dissemination of sensory data in sensor and vehicular networks.

In ad hoc networks, lack of infrastructure renders resource discovery a challenging problem. In addition, mobility induces frequent route changes. Traditional protocols proposed for resource discovery employ either global flooding or complex hierarchy formation schemes. While flooding is inefficient and does not scale well, hierarchy formation involves complex coordination between nodes and therefore may suffer significant performance degradation due to frequent, mobility induced, changes in network connectivity.

To overcome these limitations we propose a new architecture for efficient resource discovery in large-scale ad hoc networks, called CARD. Our study targets resource discovery and routing for short flows. CARD is *not* a general routing protocol, as we make a design decision to trade-off shortest paths for drastic reduction in discovery overhead. CARD,

* A. Helmy was supported by NSF CAREER Award 0134650, and research grants from Intel Corp. and Pratt & Whitney Institute for Collaborative Engineering.

however, may be integrated easily with zone routing protocols to compose a general routing solution.

Nodes in ad hoc networks are usually portable devices with limited battery power. Therefore to save power the resource discovery mechanism should be efficient in terms of communication overhead. Our architecture is designed to meet requirements for power-efficient resource discovery and small transfers in large-scale ad hoc networks with (potentially) thousands of wireless devices. Scalability is one of our main design goals.

Our architecture is based on the concept of *small worlds* [8,26,27] where the addition of a small number of short cuts in highly clustered networks results in significant reduction in the average path length (or degrees of separation) to approach that of random networks. In our architecture we adopt a hybrid approach in which a node uses periodic updates to reach its *vicinity* within a limited number of hops, R, and reactive querying beyond the vicinity via *contacts*. *Contacts* act as *short cuts* that attempt to transform the network into a small world by reducing the degrees of separation between the source and destination of the transfer. They help in providing a view of the network beyond the vicinity during resource discovery. Each node maintains state for a few contacts beyond its vicinity. Contacts are polled periodically to validate their presence and routes. For discovering resources efficiently, queries are sent to the contacts that leverage the knowledge of their vicinity. As the number of contacts increases, the network view (or reachability) increases. However, at the same time the overhead involved in contact selection and maintenance also increases. Our results show this trade-off. We introduce and study alternative mechanisms for contact selection and identify a novel scheme (called the *edge method* for contact selection) that is able to achieve a balanced trade-off and good performance in terms of increased reachability and reduced overhead.

Once the contacts are selected by a node they are used in the resolution of resource discovery queries. Only the contact nodes are queried without the need for flooding, resulting in drastic reduction in per-query communication overhead. The total overhead, however, is the resultant of

(i) the query overhead, which is a function of the per-query overhead and the query rate,

(ii) the vicinity establishment and maintenance overhead, which is a function of the node mobility, and

(iii) the contact selection and maintenance overhead.

Our study elaborates on the interplay between these various overhead components, the query rate, and the mobility rate using the call-to-mobility-ratio (CMR) metric.

Extensive simulation-based comparisons with flooding and bordercasting [5,20] show our architecture to be more efficient, especially for high query rates. Simulation results also show that our protocol is scalable and can be configured to provide good performance for various network sizes. Overhead savings are function of the query rate, reaching 93% (vs. flooding) and 80% (vs. bordercasting) in communication savings for high query rates in large-scale networks; a drastic improvement in performance.

The rest of this document is organized as follows. Section 2 discusses related work. Section 3 describes our design goals and provides an overview of our architecture, CARD, and introduces the contact selection, maintenance and query algorithms. Section 4 presents analysis of CARD, and compares it to flooding, smart flooding and bordercasting. We conclude in section 5.

2. Related work

Related research lies in the areas of routing and resource discovery in ad hoc networks. Due to lack of infrastructure in ad hoc networks, resource (and route) discovery is a challenging problem. Most of the routing protocols can be broadly classified as: proactive (table-driven), reactive (on-demand), hybrid, or hierarchical.

Proactive schemes such as DSDV [21], WRP [18] and GSR [2] flood periodic updates throughout the network. This is resource consuming, especially for large-scale networks. Reactive schemes such as AODV [22] and DSR [13] attempt to reduce the overhead due to periodic updates by maintaining state only for the active resources and using route caching. In these schemes a search is initiated for new discovery requests. However, the search procedure generally involves flooding (or expanding ring search), which also incurs significant overhead. Furthermore, the performance of on-demand routing with caching has been shown to degrade significantly with small transfers in large-scale mobile networks.

Hybrid schemes such as the zone routing protocol (ZRP) [5,20] try to combine the benefits of both the proactive and reactive schemes. ZRP limits the overhead of periodic updates to a limited number of hops (called the zone radius). Resources beyond the zone are discovered in a reactive manner by sending queries through nodes at the edges of the zones (bordercasting). The zone concept is similar to the vicinity concept in our study. However, instead of bordercasting we use contact queries. The design principles upon which our CARD architecture was designed – employing contacts as short cuts to create a small world, and trading off optimal paths for energy efficiency – are fundamentally different from those used for ZRP bordercast. In our study, through detailed comparison we show that the contact-based approach is much more efficient than bordercasting for our purposes. Furthermore, CARD maybe easily integrated with ZRP to provide a complete routing protocol in which ZRP is used to discover routes for long-lived flows and CARD is used for resource discovery and small transfers.

Hierarchical schemes, such as CGSR [3,15], tend to have good scalability, but involve election of cluster-heads, that have greater responsibilities than other nodes. A cluster-head is responsible for routing traffic in and out of the cluster. Cluster-based hierarchies rely on complex coordination and thus are susceptible to major re-configuration due to mobility and node failure, leading to serious performance degradation

in highly dynamic networks. Also, a cluster head may be a single point of failure and a potential bottleneck. In our architecture each node has its own view of the network, and hence there is very little coordination between various nodes. This enables our architecture to adapt gracefully to network dynamics. GLS [14] provides a location-discovery service for geographic routing. GLS requires nodes to know of a network grid map and assumes knowledge of node locations (via GPS or other). CARD does not require location information.

Related work on smart or efficient flooding has been proposed in [4,6,16,19]. These techniques attempt to reduce the redundancy inherent in flooding, and may be integrated in our work to provide more efficient vicinity establishment instead of regular link state protocol. One major difference between smart flooding and CARD is that smart flooding reduces the redundant messages in querying every node in the network, whereas CARD attempts to create a small world and only queries a small number of nodes on the order of the degrees of separation from source to target. In relatively sparse networks (some of which we include in our study) smart flooding will not be very effective since there is no significant redundancy in flooding anyway. Section 4.3 discusses this issue further.

In [8] we have shown the relationship between small worlds and wireless networks. In this paper, we build upon that relationship by introducing the contacts to act as short cuts in the highly clustered multi-hop wireless network, proposing and evaluating – in details – two proactive contact-selection mechanisms. We first introduced the high level idea of using contacts in [7]. The initial work on the CARD architecture was presented in [11]. This work extends the analysis of the CARD architecture and explores the important interplay between the query rate and mobility rate. The MARQ architecture [9] provides a mobility-assisted contact selection mechanism, the efficiency of which increases with mobility. In cases of static networks (e.g., sensor networks), or when mobility is low, CARD may be used in conjunction with MARQ for efficient query resolution. TRANSFER [10] provides a reactive (on-the-fly) contact selection mechanism to reduce node-contact vicinity overlaps, but does not explicitly reduce the contact-contact vicinity overlap because contacts are selected in parallel. CARD, by virtue of selecting contacts proactively and using the edge method for contact selection in serial is able to guarantee non-overlapping node-contact vicinities and reduce the contact-contact vicinity overlap, but may incur more overhead for periodic contact maintenance. ACQUIRE [23,24] is an architecture for multi-variable query resolution in sensor networks, that uses the look-ahead technique to optimize overhead. The query is forwarded from one querying node to another d hops away, randomly. The work provides an analytical framework to get optimal d for given level of network and event dynamics. A variant of CARD's contact selection may be used to reduce the overlap between the look ahead zones for successive querying nodes to improve the performance of ACQUIRE.

3. CARD architectural overview

In this section we provide an overview of the CARD architecture. In particular, we describe the design requirements for our architecture, present definitions and terminology used in this document, and introduce and investigate alternative contact selection, maintenance and query mechanisms.

3.1. Design requirements

The design requirements of our CARD resource discovery architecture for large-scale Ad hoc networks include (a) scalability, (b) power-efficiency, (c) robustness, (d) decentralized self-organization, and (e) independence of location information.

(a) Scalability. Applications of large-scale ad hoc networks include military and sensor network environments that may include thousands of nodes. Therefore the resource discovery mechanism should be scalable in terms of control overhead with increase in network size. We shall show that CARD may be configured to perform very well over a wide array of network sizes and conditions.

(b) Power and communication efficiency. Ad hoc networks include portable devices with limited battery power. Therefore, resource discovery mechanisms should be power-efficient. CARD achieves dramatic reduction in communication overhead (in terms of transmitted and received messages) over the several existing schemes considered in our study.

(c) Robustness. The mechanism should be robust in the face of network dynamics. A periodic soft-state mechanism is provided to handle node failures and frequent link failures due to mobility.

(d) Decentralized operation. For the network to be rapidly deployable, it should not require any centralized control. CARD does not require or assume any centralized entity or special infrastructure.

(e) Independence of location information. GPS (or other location information) may not be available in many context (e.g., indoors, or in simple devices and sensors). Hence, assuming availability of location information limits the applicability of the proposed scheme. We avoid such limitation in our design and do not assume or require any location information.

3.2. Definitions

An overview of the CARD architecture is shown in figure 1. Following are some terminology definitions we use throughout this document.

- *Vicinity (of a node).* All nodes within a particular number of hops (R) from the node. R is the radius of the vicinity.
- *Edge nodes (of a node's vicinity).* All nodes at a distance of exactly R hops away from the node.
- *Maximum contact distance (r).* The maximum distance (in hops) from the source within which a contact is selected.

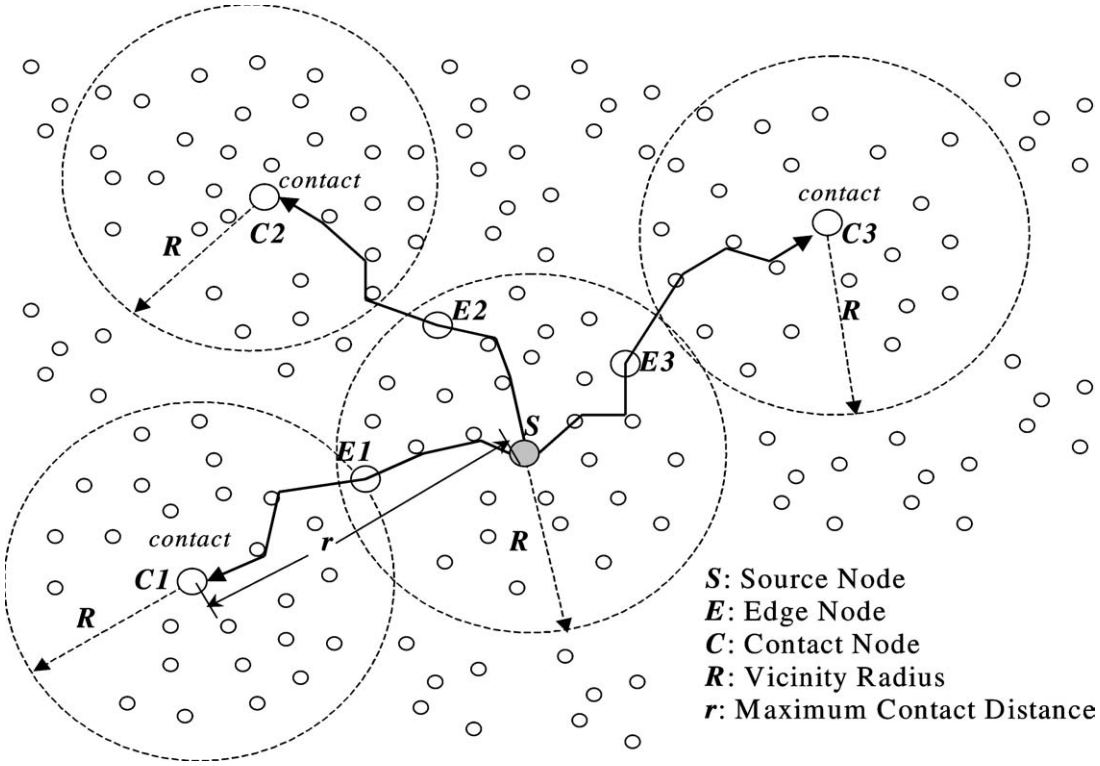


Figure 1. Architectural overview for CARD: Node S (potentially any source) keeps track of nodes and resources in its vicinity, up to R hops away. S also elects and maintains routes to a small number of *contacts* (*NoC*) (in this case $NoC = 3$ contacts: $C1$, $C2$, and $C3$). *Contacts* are selected within r hops away from S. Nodes exactly R hops away from S are called the edge nodes (*Ei*).

- *Overlap*. Overlap between nodes represents number of common nodes between their vicinities.
- *Number of Contacts (NoC)*. *NoC* specifies the value of the maximum number of contacts to be selected by each source node. The actual number of contacts chosen is usually less than this value. This is due to the fact that for a particular value of R and r, there is only a limited region available for choosing contacts. Once this region has been covered by vicinities of the chosen contacts, choosing more contacts in the same region is not possible, as their vicinities would overlap with the vicinities of the already chosen contacts. This is according to our contact selection policy to minimize overlap.
- *Depth of search (D)*. D specifies the levels of contacts (i.e., contacts of contacts) queried by a source.
- *Reachability*. The *reachability* of a source node refers to the number of nodes that can be reached by the source node. This includes the nodes within the vicinity that can be reached directly and the nodes that lie in the contacts' vicinities, and their contacts' vicinities, and so on, up to D levels of contacts. This is also considered a measure of the discovery success rate.

3.3. Establishing and maintaining vicinity information

Our architecture employs a hybrid of proactive and reactive approaches for resource discovery. As shown in figure 1, all nodes within R hops from a node form the node's vicinity. Each node proactively (e.g., using a link state protocol) maintains state for resources within its vicinity. Alternatively, a smart flooding scheme (e.g., based on dominating sets) may be used to reduce the vicinity establishment and maintenance overhead. For comparison reasons, however, in this study we shall use a link state protocol similar to that used in ZRP to maintain vicinity information. The overhead of such link state protocol increases with node mobility and the number of nodes within the vicinity. Such overhead under mobility scenarios will be thoroughly studied later in section 4.3, and will be factored into the overall overhead of CARD. As we shall see, when the vicinity overhead is amortized over a reasonable number of queries the overall gain is still quite significant.

Each node also maintains state for (a few) nodes that lie outside the vicinity. These nodes serve as contacts for accessing resources beyond the vicinity. Contacts are selected, maintained and queried using the mechanisms described below.

3.4. Contact selection, maintenance and query mechanisms

Contacts are key to the efficient resolution of resource discovery queries. In CARD, the contacts are selected proactively in anticipation of queries, and paths to these contacts are maintained using a periodic soft state mechanism to capture network dynamics and mobility effects. Since the contacts are selected proactively, the contact selection delays become less of a concern (than they would otherwise in a reactive, on-the-fly, scheme, for example). Hence, contacts are selected in

a serial fashion, one after the other, with information about previously selected contacts being utilized to effectively select new contacts. When a resource discovery query is issued, and the resource is not found in the vicinity, the source node queries its contacts first, and the contacts may query their contacts, and so on, until the query is resolved. Below, we introduce the details of the contact selection, maintenance and query mechanisms.

3.4.1. Contact selection mechanism

Any potential source of query or small transfer may choose to select contacts. The procedure starts when a node ***s*** sends a Contact Selection (*CS*) message through each of its edge nodes (*Ei*), one at a time, until *NoC* number of contacts are selected or until all edge nodes have been attempted. An edge node receiving a *CS* forwards it to a randomly chosen neighbor (X).

A node receiving a *CS* decides whether or not to be a contact for ***s*** based on a contact selection method. This decision is made using either a probabilistic method (PM) or edge method (EM). These methods are described later in this section. After using either procedure PM or EM for deciding whether (or not) to be a contact, if the node receiving a *CS* does not choose to be the contact, it forwards the *CS* to one of its randomly chosen neighbor (excluding the one from which the *CS* was received).

The *CS* traverses in a depth-first manner until a contact is chosen or the distance traversed by the *CS* from ***s*** reaches r hops. If a contact is still not chosen (due to overlap), *CS* backtracks to the previous node, which forwards it to another randomly chosen neighbor. When a contact is selected, the path to the contact is returned and stored at ***s***.

3.4.2. Contact selection methods

We introduce and compare two different methods for contact selection: (a) the probabilistic method (PM), and (b) the edge method (EM).

(a) *Probabilistic Method (PM).* Contacts increase a node's view (reachability) of the network beyond its own vicinity. To increase the reachability of a node, the vicinities of that node, call it ***s***, and its contacts should be disjoint, i.e., there should be reduced (or no) overlap between the vicinity of ***s*** and the vicinity of any of its contacts. The vicinities of different contacts of the same node should also be non-overlapping, to achieve good increase in reachability. To achieve this, the *CS* contains the following information: (i) ID of node ***s***, (ii) a list of already-selected-contacts of ***s*** (*Contact_List*; typically small of ∼5 IDs), and (iii) the hop count d.

This information is used as follows. When a node X receives a *CS*, it first checks if ***s*** lies within its vicinity. This check is easily performed since each node has complete knowledge of its vicinity. So a node knows the IDs of all the other nodes in its vicinity. X also checks if its vicinity contains any of the node IDs contained in the *Contact_List*.

If neither ***s*** nor any of its already-selected-contacts lie in the vicinity of X, then X probabilistically chooses itself as the contact. This probability (P) of choosing to be a contact is defined as follows:

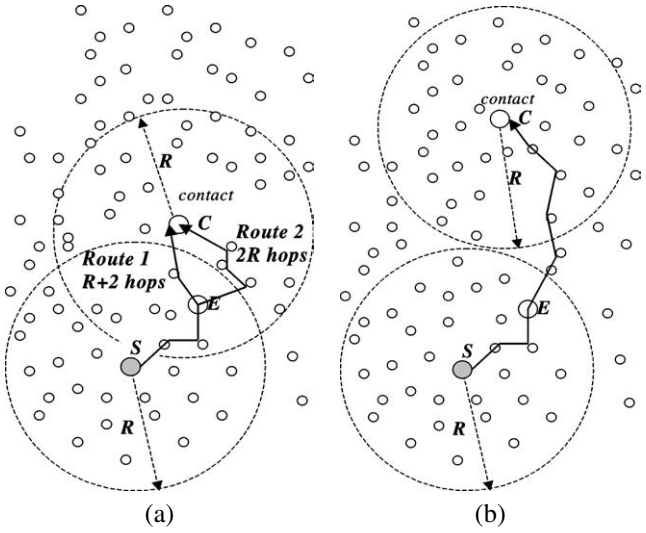


Figure 2. Overlap in (a) due to the use of P. (a) Heavy overlap; (b) no overlap.

$$P = \frac{d - R}{r - R}, \tag{1}$$

where d is the number of hops traversed from ***s*** to X. The value of d is included in the *CS* as hop count. From the above equation, when $d = R$, $P = 0$, and when $d = r$, $P = 1$. This aims to select contacts between R and r hops away from ***s***, and is formulated to provide an increase in reachablility with the addition of new contacts outside the vicinity of ***s***, i.e., with distance $> R$ hops from ***s***. The probability P increases with the number of hops traversed, d. However, there are cases where equation (1) does not provide the maximum benefit of adding a contact. An example case is shown in figure 2(a) where ***c*** is the contact for node ***s*** and the contact route (route 1 in the figure) is $R + 2$ hops.

In this figure although the distance between ***s*** and its contact ***c*** is greater than R hops, there is still heavy overlap between the two vicinities. Such situations will arise whenever a node within R hops from the edge node becomes the contact. To alleviate this effect, equation (1) is modified to:

$$P = \frac{d - 2R}{r - 2R}. \tag{2}$$

In this equation $P = 0$ when $d = 2R$ and $P = 1$ when $d = r$. Hence, contacts are chosen after traversing between $2R$ and r hops from the source ***s***.

Figure 3 explains the contact selection procedure with an example. In the figure $R = 3$ and $r = 6$. Nodes ***a***, ***b***, ***c*** and ***d*** are the edge nodes for node ***s***. Node ***s*** sends a Contact Selection (*CS*) message through its edge node ***a***. Node ***a*** randomly chooses one of its neighbors, ***e***, and forwards the *CS* to that node. Node ***e*** calculates the probability P, say according to equation (1). If the probability of being the contact fails at ***e***, it forwards the *CS* to one of its neighbors ***f*** (chosen randomly). Node ***f*** again forwards the *CS* to ***g***. As ***g*** is at r hops from ***s***, the probability P at ***g*** is 1. However, ***g*** still

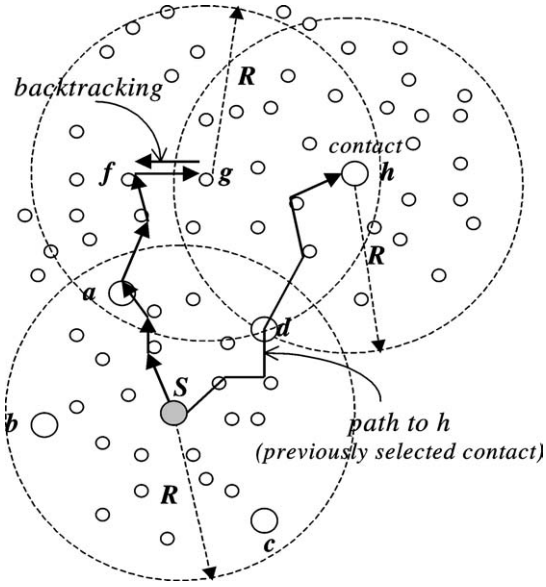


Figure 3. Selecting contacts.

cannot become a contact for s as there already exists another contact h (which was selected through a previous selection via another edge node d) in the vicinity of g. So g returns the *CS* to f (*backtracking*). Node f then forwards *CS* to another neighbor, and so on.

(b) *Edge Method (EM).* Even with equation (2) the probabilistic method can result in a situation where there is some overlap between the vicinity of the contact and the vicinity of s. This is possible due to the fact that the nodes do not have a sense of direction once the CS message is forwarded out of the vicinity (i.e., $d > R$). Therefore, it is possible that a contact may be selected at a location where the *CS* has traversed more than $2R$ hops, but the contact may in fact be closer than $2R$ hops from the source, as shown in figure 2(a) route 2, leading to heavy overlap.

More seriously, the probabilistic method for contact selection can be expensive in terms of the amount of traffic generated by the *CS*. This is due to the extra traffic generated due to backtracking, and lost opportunities when the probability fails, even when there is no overlap. To reduce the possibility of such a situation, the probability equations (1) and (2) are not used. The probability equations were formulated to have a higher possibility of choosing the contact that lies either between R and r hops (equation (1)) or between $2R$ and r hops (equation (2)). To maintain this non-overlapping property without the probability equations, the contact selection procedure is modified as follows.

The list of all edge nodes (*Edge_List*) of s is added to the *CS*. Also, the query and source IDs are included to prevent looping. Note that the *Edge_List* is readily available through the vicinity information and obtaining it does not require any extra overhead. Upon receiving a *CS*, in addition to checking for overlap with s's vicinity and the vicinities of all the already-selected-contacts (*Contact_List*), the receiving node also checks for overlap with the vicinities of any of the nodes on the *Edge_List* as well. It can be easily proven that this scheme guarantees non-overlap between the node and the contacts vicinities, as follows. Any node that lies at a distance of R hops or less from the edge will have an overlapping vicinity with the s's vicinity, and hence will have at least one of s's edge nodes in its vicinity. Thus, checking for non-overlap with the edge nodes ensures that a contact is chosen at least $2R+1$ hops away from s. This eliminates the possibility of an overlap due to the lack of direction. The *Edge_List* may be added to the *CS* in a communication-efficient manner by using bloom filters [17] to represent membership in the edge-list. Figures 4 and 5 show a comparison of the probabilistic and edge methods. As can be seen from figure 4 the reachability saturates in both PM and EM. However the saturation occurs much earlier in the case of probabilistic method. Also as compared to EM, the reachability achieved is less for PM, for the same values of *NoC*. Figure 5 shows the backtracking overhead for PM and EM. Due to the reasons explained earlier, overhead is significantly reduced for EM.

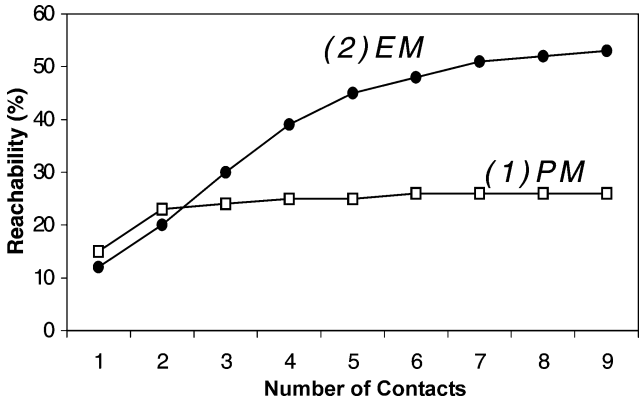


Figure 4. Reachability for (1) PM and (2) EM. [1]

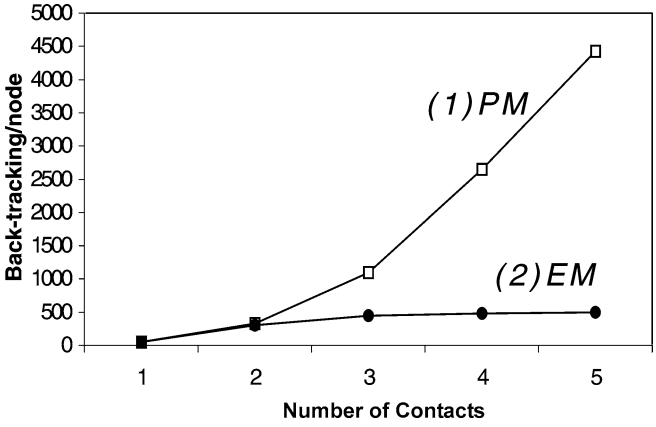


Figure 5. Overhead for (1) PM and (2) EM.[1]

3.4.3. Contact maintenance mechanism

Node mobility may cause the path to a contact to change. Therefore a node needs to keep track of its contacts and their paths. This is done using soft-state periodic polling of the contacts as follows.

[1] Shown: 500 nodes, 710 m × 710 m, Tx range = 50 m, $R = 3$, $r = 20$, $D = 1$. Similar trends were obtained for other simulation scenarios.

(1) Each node periodically sends a *validation* message towards each of its contacts. These validation messages contain the path from a node s to the contact.

(2) Each node on the path that receives the validation message checks if the next hop in the path is a directly connected neighbor. If so, it forwards the validation message to the next hop node. If the next hop is missing, the node tries to salvage the path using *local recovery*, discussed later in this subsection.

(3) If a path cannot be salvaged using local recovery, the contact is considered to be lost.

(4) If the path to a contact is validated but the number of hops to the contact does not lie between $2R$ and r, the contact is considered to be lost.

(5) After validating all the contacts, if the number of contacts left is less than the specified *NoC*, then a new contact selection procedure is initiated.

The *local recovery* mechanism is illustrated using an example of a contact path ($a \rightarrow b \rightarrow c \rightarrow d \rightarrow e$). Assuming reasonable values of node velocities and validation frequency (section 1 in our study), there is a high probability that if a node (say c) has moved out of a contact path (i.e., moved out of transmission range of b), that it is still within the vicinity of the previous hop (b) in the path. Even in the case when the moving node (c) is completely lost (because it has moved out of the vicinity of the previous hop, b), some other node further down the path (say $d$or e) might have moved into the vicinity of the previous node (b). Local recovery takes advantage of these cases to recover from changes in the path when possible, without having to initiate new searches from s. Thus local recovery provides an efficient mechanism for validating contacts and recovering from changes in the contact paths. If the next hop on the path (node c) is missing, the node that received the validation message (node b) looks for the next hops (c, d and e) in its vicinity routing table. If any of the next hops (c, d or e) is found the vicinity, the path is updated and the validation message is forwarded to that next hop. If the lookups for all next hops fail, an error message is returned to the source s, and another contact selection is initiated. Figure 6 further illustrates an example of local recovery when two nodes along the path to the contact (nodes c and d in this case) move.

3.4.4. Query mechanism

When a source node s (potentially any node), needs to reach a destination or target resource T it first checks its vicinity table to see if T exists in its own vicinity. If T is not found in the vicinity, s sends a Destination Search Query (DSQ) to its contacts. The DSQ contains the following information: (1) depth of search (D), and (2) target resource ID (T). Upon receiving a DSQ, each contact checks the value of D. If D is equal to 1, the contact performs a lookup for T in its own vicinity. If T exists, then the path to T is returned to s, and the query is considered successful. Otherwise, if $D > 1$, the contact receiving the DSQ decrements D by 1 and forwards the DSQ to its contacts. In this way the DSQ travels through multiple levels of contacts until D reduces to 1.

The source node s first sends a DSQ with $D = 1$ to its contacts. So only the first level contacts are queried with this DSQ. After querying all its contacts if the source does not

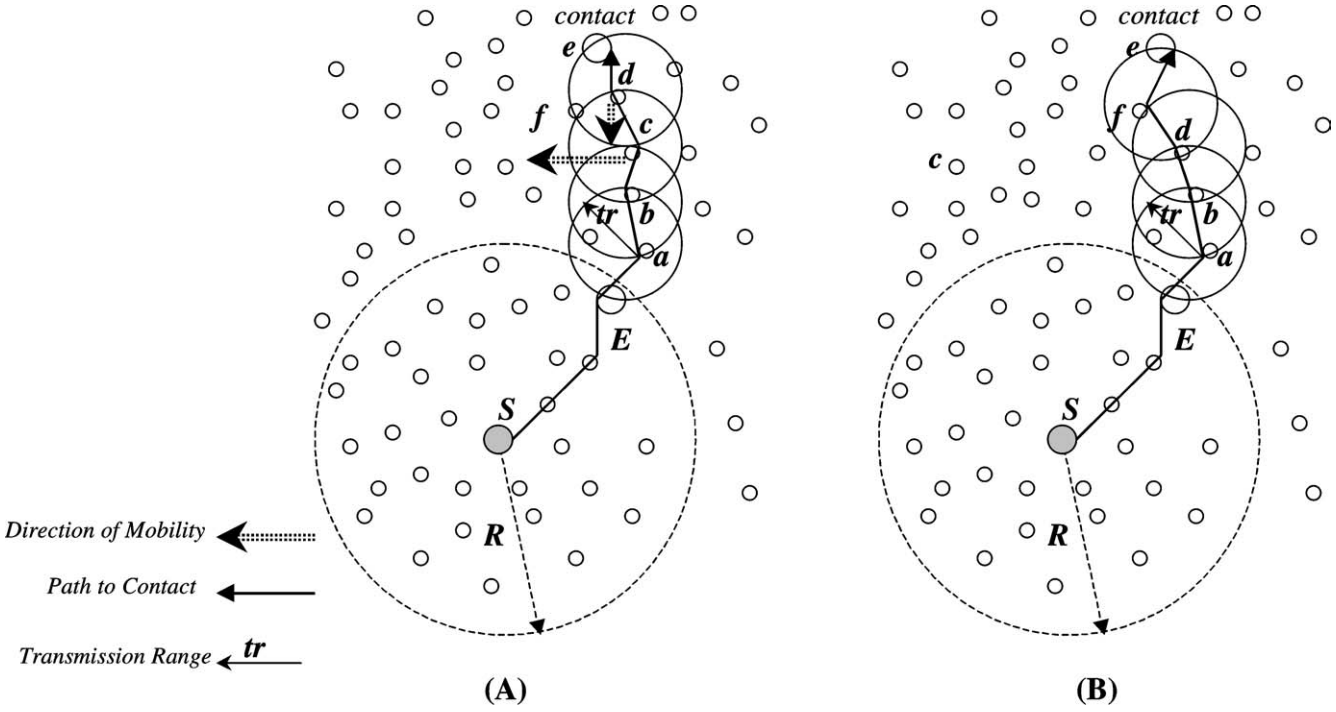


Figure 6. Contact maintenance using local recovery: (A) Path to the contact node e goes through $a \rightarrow b \rightarrow c \rightarrow d \rightarrow e$. Node c is moving away from b's transmission range, and node d is moving away from e. (B) During *validation*, node b loses contact with node c but finds node d in its range. Also, node d loses direct contact with e but finds a path in its vicinity to node e through node f. The updated part of the contact path is thus $a \rightarrow b \rightarrow d \rightarrow f \rightarrow e$.

Table 1
Description of the various scenarios used for simulating CARD.

Scenario	Nodes	Area	Transmission range	No. of links	Aver. node degree	Network diameter	Aver. path length (hops)
1	250	500×500	50	837	6.75	23	9.378
2	250	710×710	50	632	5.223	25	9.614
3	250	1000×1000	50	284	2.57	13	3.76
4	500	710×710	30	702	4.32	20	5.8744
5	500	710×710	50	1854	7.416	29	11.641
6	500	710×710	70	3564	14.184	17	7.06
7	1000	710×710	50	8019	16.038	24	8.75
8	1000	1000×1000	50	4062	8.156	37	14.33

receive a path to the target within a specified time, it creates a new DSQ with $D = 2$ and sends it again to its contacts. Each contact observes that $D = 2$ and recognizes that this query is not meant for itself. So it reduces the value of D in the DSQ by 1 and forwards it to its contacts. These contacts serve as second level contacts for the source. Upon receiving the DSQ, a second level contact observes that $D = 1$ and it does a lookup for the target T in its own vicinity and returns the path to T, if found. In this way the value of D is used to query multiple levels of contacts in a manner similar to the expanding ring search. However, querying in CARD is much more efficient than the expanding ring search as the queries are not flooded with different TTLs but are directed to indiviual nodes (the contacts). Contacts leverage knowledge of their vicinity (gained through the proactive scheme operating within the vicinity) to provide an efficient querying mechanism.

4. Evaluation and analysis

In this section we present detailed simulation based evaluation and analysis of our architecture. NS-2 [1] along with our CARD extensions and other utilities were used to generate various scenarios of ad hoc networks. The mobility model used for these simulations was the random way-point model. Our simulations so far did not consider MAC-layer issues. In random way point model a node is assigned a random velocity from $[0, V_{\max}]$ and assigned a destination location randomly. Once the node reaches its destination it is assigned a random velocity and random destination again, so on. In the reachability analysis experiments the mobility was set to '0' to understand the basic effects of the various architectural parameters on reachability characteristics. For the maintenance overhead, total overhead and comparison experiments, continuous mobility was used (with no pauses) with $V_{\max} = 20$ m/s.

First we try to understand the effect of various parameters such as vicinity radius (R), maximum contact distance (r), the number of contacts (NoC), the depth of search (D) and network size (N) on reachability and overhead. Reachability here is defined as the percentage of nodes that are reachable from a source node. For overhead we consider the number of control messages; the contact selection (CS) messages and the periodic contact maintenance *validation* messages. Having developed an understanding of the various parameters in our architecture, we then compare it to other schemes such as flooding and bordercasting in terms of query overhead and query success rate.

Table 1 shows the scenarios used in our simulations. These scenarios vary in number of nodes, network size, node density and transmission range. The variation is considered to capture the effect of these factors on CARD. As was shown in figures 4 and 5, the edge method outperforms the probabilistic method. Therefore, we use the edge method (EM) for contact selection in the rest of our study.

4.1. Analysis of reachability

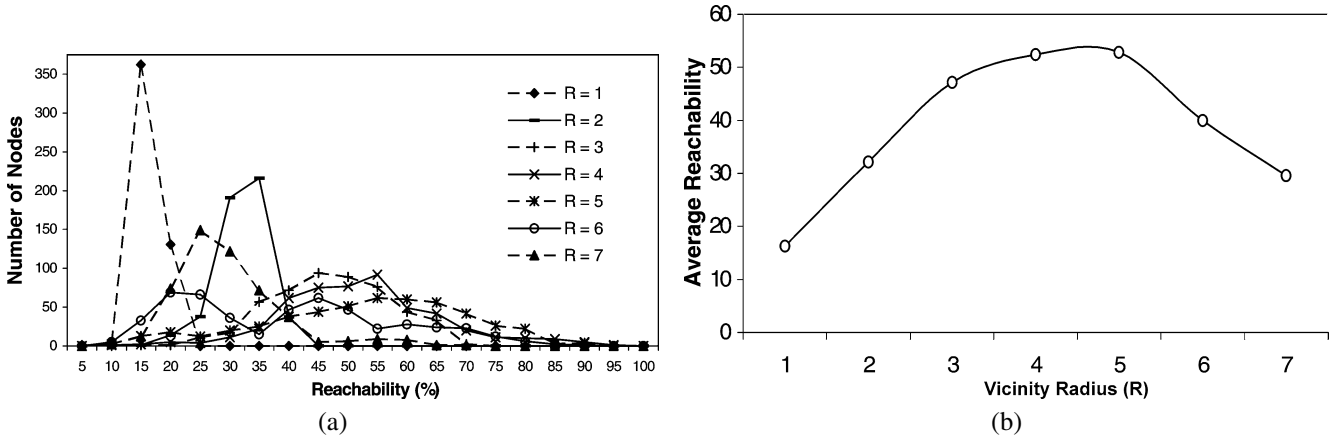
Analysis of the *reachability*, or query success rate, was conducted to understand how contacts help in increasing the view of the network. Here we present results for a topology of 500 nodes spread over area of 710 m $\times$ 710 m. The details can be seen from table 1, scenario number 5. Similar trends were observed for other scenarios.

4.1.1. Varying vicinity radius (R)

Figure 7 shows the effect of increasing the vicinity radius (R) on reachability. As R increases, the reachability increases and the reachability distribution in figure 7(a) shifts to the right; i.e., more nodes achieve higher percentage of reachability. This increase in reachability with the increase in R is due to increase in the number of nodes within the vicinity. As the value $2R$ approaches the maximum contact distance r ($r = 16$ in this experiment), the region available for contact selection (between $2R$ and r) is reduced. This results in less number of contacts being chosen. In figure 5, when $R = 7$, contacts can only be selected between $2R = 14$ and $r = 16$ hops from the source. This small region for contact selection significantly reduces the number of contact and hence the reachability reduces as seen in figure 7(b). At this point most reachability is due to the vicinity of the source.

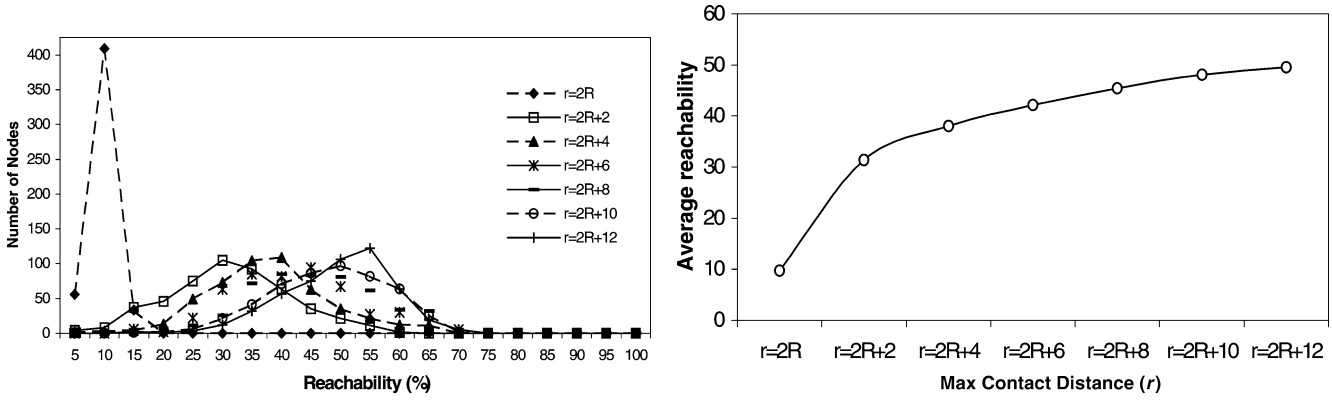
4.1.2. Varying maximum contact distance (r)

Figure 8 shows the effect of increasing r on reachability. Since contacts are selected between $2R$ and r hops from the source, higher values of r provide a wider region for contact selection. The mechanisms for the edge method for contact selection described earlier provide selection of contacts that have vicinities with reduced overlaps. This implies that as r increases a larger number of contacts can be selected without having vicinity overlaps. Therefore reachability increases



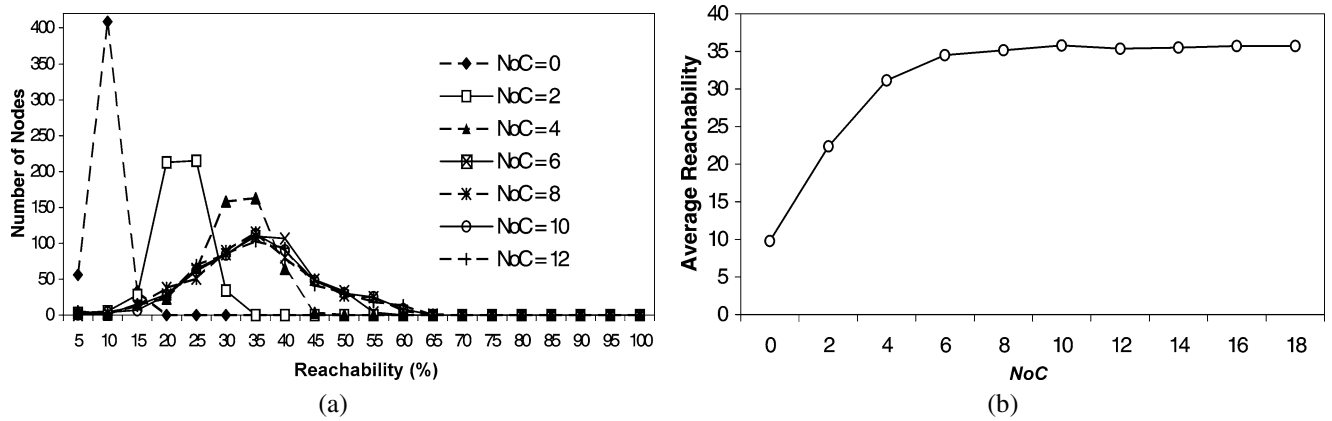
(a) (b)

Figure 7. Effect of vicinity radius (R) on reachability. $N = 500$, area $= 710\,\text{m} \times 710\,\text{m}$, propagation range $= 50\,\text{m}$, $r = 16$, $NoC = 10$, $D = 1$. (a) Histogram of reachability for different values of R; (b) average reachability with R.



(a) (b)

Figure 8. Effect of maximum contact distance (r) on reachability. $N = 500$, area $= 710\,\text{m} \times 710\,\text{m}$, propagation range $= 50\,\text{m}$, $R = 3$, $NoC = 10$, $D = 1$. (a) Histogram of reachability for different values of r; (b) average reachability with r.



(a) (b)

Figure 9. Effect of number of contacts (NoC) on reachability. $N = 500$, area $= 710\,\text{m} \times 710\,\text{m}$, propagation range $= 50\,\text{m}$, $R = 3$, $r = 10$, $D = 1$. (a) Histogram of reachability for different values of NoC; (b) average reachability with NoC.

with increase in r. Larger values of r also mean that the average contact path length would increase (as more contacts are chosen at larger distances from the source). However, once the vicinities of the contacts and the source become non-overlapping, for $r > (2R + 8)$, we see no significant increase in reachability with further increase in r.

4.1.3. Varying number of contacts (NoC)

NoC specifies the maximum number of contacts to be selected for each node. The actual number of contacts chosen may be less than this value. This is because of the limited region available for choosing contacts for given R and r according to the contact selection mechanism. Once this region has

been covered by vicinities of chosen contacts, choosing more contacts in the same region is not possible as their vicinities would overlap with the vicinities of the already chosen contacts. Therefore the contact selection mechanism prevents the selection of more contacts. This can be seen in figure 9, in which the reachability initially increases sharply as more and more contacts are chosen. However, the increase in reachability saturates beyond $NoC = 6$ as the actual number of contacts chosen saturates due to the effect of overlapping vicinities.

4.1.4. Varying depth of search (D)

D specifies the levels of contacts that are queried in a breadth first manner. When $D = 1$, a source node looking for a resource beyond its vicinity, queries its first level contacts only. When $D = 2$, if none of the first level contacts contain the resource in its vicinity, second level contacts (contacts of the first level contacts) are queried through the first level contacts. As can be seen from the figure 10, reachability increases sharply as the depth of search, D is increased. The depth of search, D, results in a tree-like structure of contacts, improving the reachability and success rate of CARD.

4.1.5. Varying network size

Figure 11 shows the reachability distribution for three different network sizes, $N = 250$, 500 and 1000 nodes. The area of the three networks has been chosen so that the node density is almost same across the three networks. Figure 11 shows that for a given network (specified by the values of N and the area), the values of R and r can be configured to provide a desirable reachability distribution in which most of the nodes have a high value of reachability.

4.2. Contact selection and maintenance overhead analysis

The overhead analysis measures the number of control messages required for contact selection and maintenance. The query overhead is considered in the next section. The overhead considered in this section includes:

1. Contact selection overhead: This is the amount of *CS* traffic generated for selecting new contacts. This includes overhead due to *backtracking* as described earlier.

2. Contact maintenance overhead: This is the traffic generated by the contact path validation messages. *Local recov-*

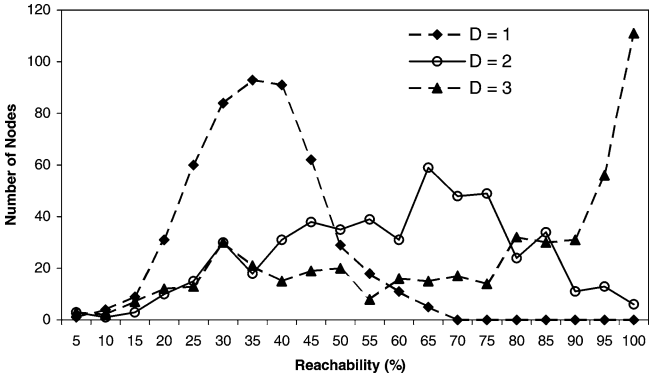


Figure 10. Effect of depth of search (D) on reachability. $N = 500$, area = 710 m $\times$ 710 m, Tx range = 50 m, $R = 3$, $NoC = 10$, $r = 10$.

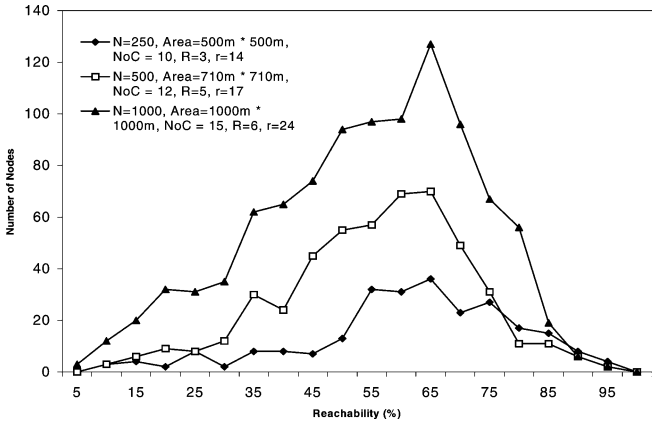
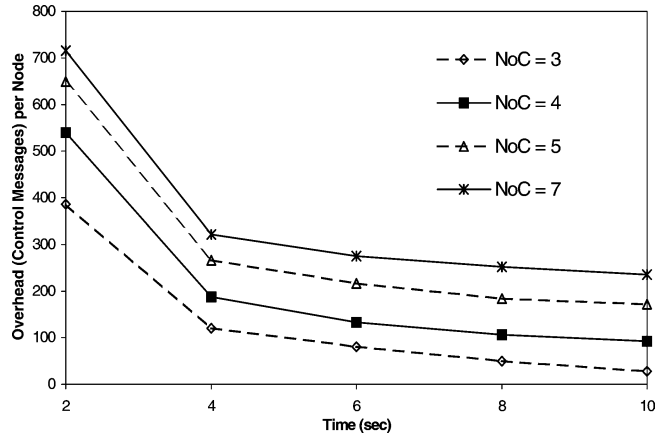
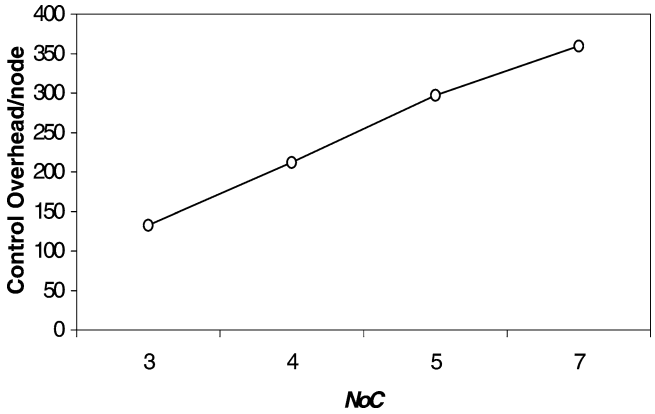


Figure 11. Reachability for different network sizes ($D = 1$).



(a)



(b)

Figure 12. Effect of number of contacts (NoC) on contact selection overhead. $N = 500$, area = 710 m $\times$ 710 m, Tx range = 50 m, $R = 3$, $r = 10$, $D = 1$. (a) Overhead over time for different values of NoC; (b) average (per sec) overhead for different values of NoC.

ery, as described earlier, helps in reducing this part of the total overhead.

Results are shown for scenario number 5 in table 1 ($N = 500$, area $= 710$ m $\times$ 710 m). Similar trends were observed for other scenarios.

4.2.1. Varying number of contacts (NoC)

As shown in figure 12, as the number of contacts increases the maintenance overhead increases sharply as more nodes are periodically maintained through the validation scheme.

4.2.2. Varying maximum contact distance (r)

As r increases the number of selected contacts increases. The increase in the number of contacts is due to the availability of a wider area for choosing contacts. Moreover, with higher values of r, contacts may lie at greater distances from the source. That is, the contact path length is expected to be higher for larger values of r. This suggests that the maintenance overhead should increase with increase in r. However, as shown in figure 13, the overhead actually decreases with increase in r. Figure 14 explains this decrease in maintenance overhead. Figure 14 shows that as the value of r increases the backtracking overhead decreases significantly. Recall that backtracking occurs when a node receiving a *CS* cannot become a contact due to overlap with already existing contacts. As r increases, the possibility of this overlap decreases due to availability of a wider area for contact selection. This decrease in back-tracking overhead is significantly more than the increase in overhead due to increased number of contacts and contact path length. Therefore, the total contact selection and maintenance overhead decreases.

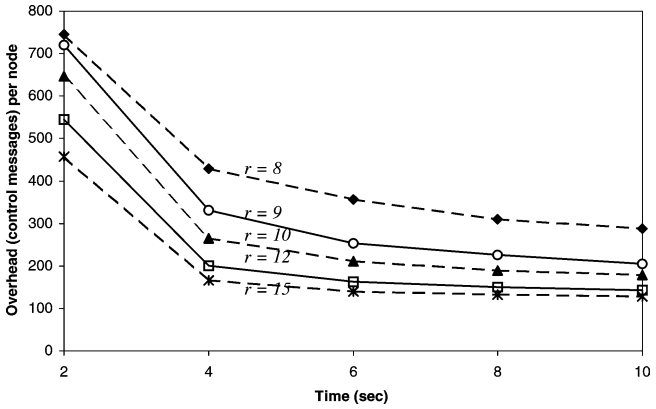


Figure 13. Effect of maximum contact distance (r) on contact selection overhead. $N = 500$, area $= 710$ m $\times$ 710 m, Tx range $= 50$ m, $NoC = 5$, $R = 3$, $D = 1$.

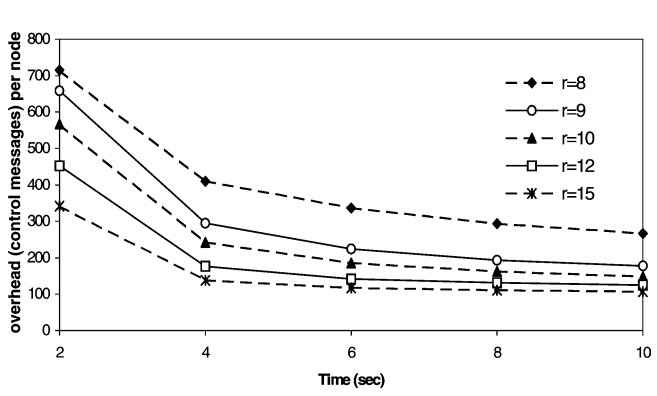
4.3. Maintenance overhead over time

Figure 15 shows the maintenance overhead per node over a 20 s period for $V_{\max} = 20$ m/s. The maintenance overhead decreases steadily with time. However, the number of contacts increases slightly. This suggests that the source nodes find more stable contacts over time. Stable contacts may be defined as those nodes that have low velocity relative to the source node. For example, a node moving in the same direction as source node with similar velocity could prove to be a stable contact. Hence, over time, CARD leads to source nodes finding more stable contacts.

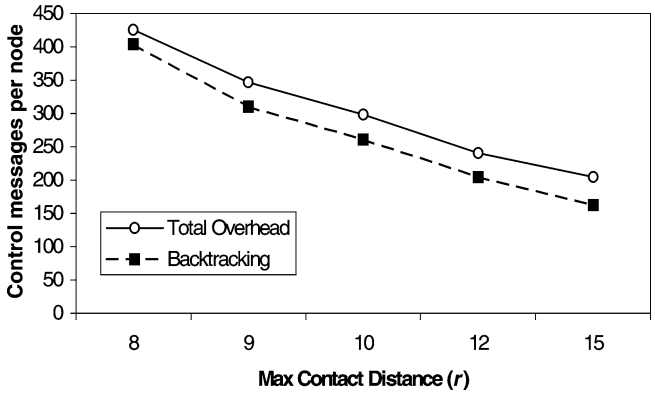
4.4. Comparison with related schemes (query overhead and total overhead)

We compare the performance of CARD to that of flooding, smart flooding [19] and bordercasting [20], in terms of average query overhead and overall overhead. Simulations were repeated several times with various random seeds to filter out the noise.

Figure 16 shows the average traffic generated per query for the three protocols. We select random source-destination pairs in the network (the same pairs were used for all the three protocols). The graph shows the average overhead for random queries with different network sizes, for each protocol. The overhead includes number of transmissions as well as number of receptions. Therefore the overhead for flooding is about twice the number of links (as expected). Bordercasting is implemented as described in [20]. We implemented query detection (QD1 and QD2) and early termination (ET) as described in [20] to improve the performance. For smart flood-



(a)



(b)

Figure 14. Effect of maximum contact distance (r) on backtracking overhead. $N = 500$, area $= 710$ m $\times$ 710 m, propagation range $= 50$ m, $NoC = 5$, $R = 3$, $D = 1$. (a) Backtracking over time for different values of r; (b) contact selection and backtracking overheads (per sec).

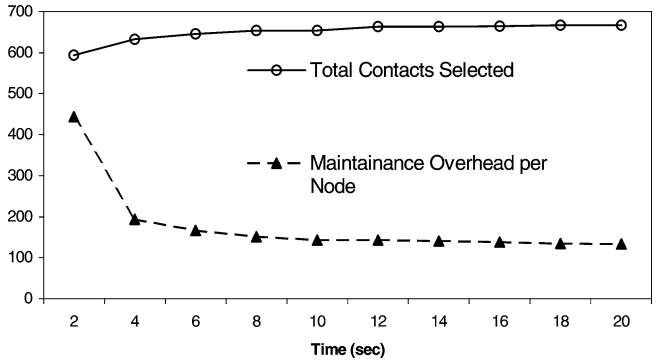


Figure 15. Variation of overhead with time. $N = 250$, area $= 710\,\text{m} \times 710\,\text{m}$, Tx range $= 50$ m, $NoC = 6$, $R = 4$, $r = 16$, $D = 1$.

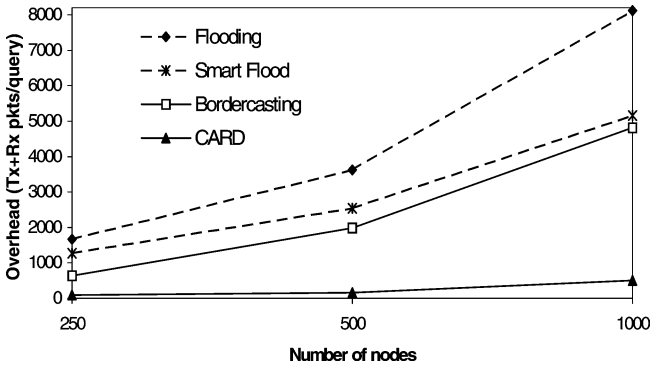


Figure 16. Query overhead for CARD, flooding and bordercasting.

ing we investigated several techniques (probabilistic flooding, minimum dominating set, counter based methods) and we show the results for those settings that achieved success rate of 90%. This was equivalent to probabilistic flooding as in [19] with $p = 0.65$. For CARD the values of R and r used were chosen as the values that gave maximum reachability for that particular network size. This information was obtained from previous results shown under the analysis of CARD with respect to various parameters (see figure 11. Reachability for different network sizes). Flooding and bordercasting result in 100% success in queries, smart flooding achieved 90% success rate, and CARD showed a 95% success rate with $D = 3$. CARD's success rate can be increased by increasing D, or with resource replication. No replication is assumed in our study. As can be seen from figure 16, CARD leads to significant savings in communication overhead over the other two approaches. CARD incurs, on average, around 5% of the query overhead for flooding, and around 10% or more of the query overhead of bordercasting or smart flooding. We note that smart flooding achieves the least success rate. To increase the success rate for smart flooding the overhead approaches that of flooding.

What is not shown in figure 16, however, is the effect of contact and vicinity maintenance. For that we show the following 'total overhead' comparison results. Maintenance overhead (for contacts and vicinity) is a function of mobility and simulation time. Its cost is amortized over the number of queries performed during that period. Hence, we present our results as function of the query rate per mobility per node (i.e., query/sec/(m/s) or query/m); this is referred to as call-to-mobility ratio (CMR) or q query/m per node. We show results for simulations with $V_{\max} = 1$ m/s and 20 m/s, for various query rates q for 20 seconds of simulated time. These results take into consideration the contact selection and maintenance overhead, the vicinity establishment and maintenance overhead and the query overhead. As can be seen from figures 17 and 18, the advantage of using contacts becomes clearer for higher query rates, where the cost of maintenance is amortized over a large number of queries. For low mobility, in figures 17(a) and (b), the maintenance overhead is low and the advantages of using contacts are the clearest (46–85% savings for low query rates $q = 0.005$ query/m, and 86–94% savings for high query rates $q = 0.05$ to 0.5 query/m).

For high mobility, in figures 18(a), (b) the savings are less than low mobility scenarios, nonetheless they are still significant for moderate to high query rates (22–75% savings for $q = 0.05$ query/m, 79–93% savings for $q = 0.5$ query/m over flooding or bordercast). For low query rates and high mobility however, e.g., for 20 m/s and $q = 0.005$ query/m, CARD and bordercasting perform worse than flooding, where maintenance overhead dominates and only very few queries are triggered (an unlikely scenario in mobile ad hoc networks). For high mobility, large-scale, high query rates (1000 nodes, 20 m/s, 0.5 query/m), we get savings between 79% (vs. bordercasting) and 87% (vs. flooding).

To further understand the effect of query rate and mobility on the total overhead we investigate the *overhead ratio* (OR) metric for CARD over the total overhead of bordercast and flooding. This metric enables us to have a more comprehensive view of the operating conditions under which CARD is favorable. Let OR(C/B) be the overhead ratio for CARD over bordercast, and OR(C/F) and OR(C/S) be the overhead ratio of CARD over flooding and smart flooding. Let CSM be the *contact selection* and *maintenance* overhead, and let ZO be the *zone* (*or vicinity*) *maintenance* overhead, both in packets per node per m/s. Also, let CQO be the CARD query overhead in packets per query, hence $q \cdot \text{CQO}$ is the overhead in packets per node per m/s. Define BQO as the query overhead for bordercast. Hence, we get

$$\text{OR(C/B)} = \frac{\text{CSM} + \text{ZO} + q \cdot \text{CQO}}{\text{ZO} + q \cdot \text{BQO}}.$$

Similarly, we have

$$\text{OR(C/F)} = \frac{\text{CSM} + \text{ZO} + q \cdot \text{CQO}}{q \cdot \text{FQO}} \quad \text{and}$$

$$\text{OR(C/S)} = \frac{\text{CSM} + \text{ZO} + q \cdot \text{CQO}}{q \cdot \text{SQO}},$$

where FQO and SQO is the flooding and smart flooding overhead in packets per query, respectively.

OR(C/B) and OR(C/F) were evaluated for $q = 0.01$ to 100 query/sec/(m/s) per node. Figure 19 shows results for OR(C/B) and figure 20 shows results for OR(C/F). From the figures we note that, in general, when q is quite small (e.g.,

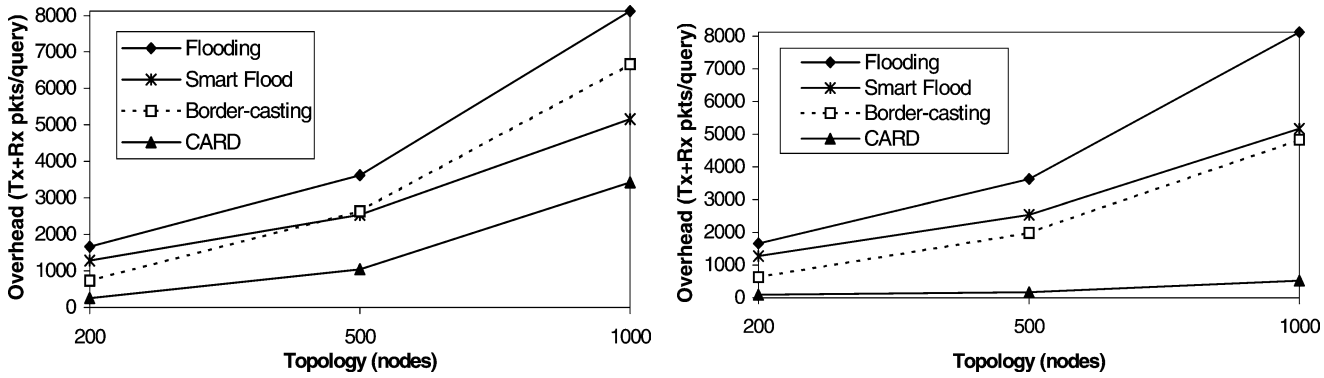


Figure 17. Total overhead for low mobility and different query rates. (a) $V_{\max} = 1$ m/s, CMR $q = 0.005$ query/m. (b) $V_{\max} = 1$ m/s, CMR $q = 0.05$ to 0.5 query/m.

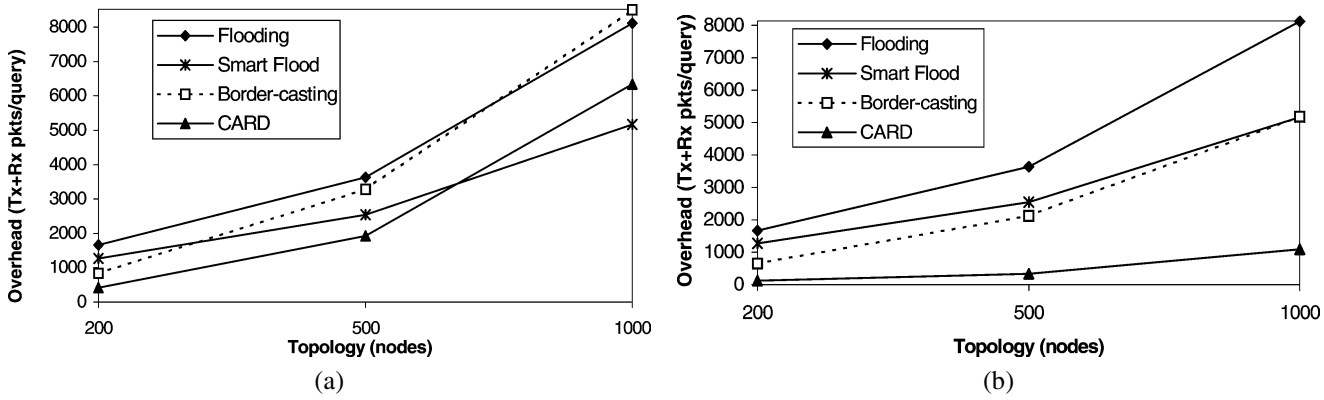


Figure 18. Total overhead for high mobility and different query rates. (a) $V_{\max} = 20$ m/s, CMR $q = 0.05$ query/m. (b) $V_{\max} = 20$ m/s, CMR $q = 0.5$ query/m.

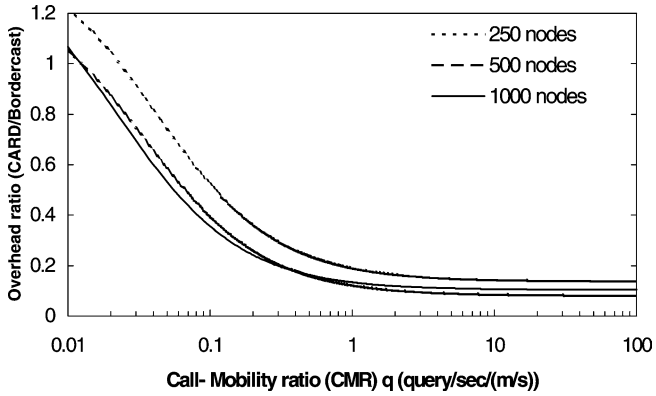


Figure 19. OR(C/B): the overhead ratio for CARD over bordercast for various values of q.

$q < 0.01$) then CARD incurs more overhead than flooding and bordercasting. This is due to the fact that CARD expends communication overhead to select and maintain contacts, as well as vicinities. If the nodes are relatively idle, resulting in very small q, then there is not enough query to amortize the cost of the maintenance overhead. This scenario is unlikely though, as we expect idle nodes to transit into *sleep* mode (to conserve energy) and not participate in periodic activities (such as vicinity and contact maintenance) while idle. From that perspective, one may consider q to be the call-to-mobility ratio during active periods. Hence, it is unlikely that q will become too small for most practical purposes. As q becomes moderate (around $q = 0.01$ query/m) we start noticing the advantage of CARD in overhead savings. In figure 18 we see that OR(C/B) becomes less than 1 (the cross over point) for $q \sim 0.01$–0.025 query/m. Also, OR(C/B) becomes less than 0.2 (i.e., 80% overhead savings) for $q \sim 0.295$–0.315 query/m for 500 and 1000 nodes and $q = 0.810$ query/m for 250 nodes. For $q \sim 10$ query/m OR(C/B) approaches 0.11 for 500 and 1000 nodes and 0.18 for 250 nodes; i.e., over 80% saving in overhead.

In figure 20 we observe that OR(C/F) becomes less than 1 for $q \sim 0.015$–0.02 query/m. Furthermore, OR(C/F) < 0.2 for $q \sim 0.14$–0.155 query/m, and OR(C/F) < 0.1 for $q \sim 0.43$–0.51 query/m. For $q \sim 10$ query/m, the overhead ratio OR(C/F) approaches 0.066; i.e., over 93% saving in overhead. In figure 21 the overhead ratio with respect to smart flooding is shown. In addition to achieving better success rate than smart flooding, CARD also achieves less total overhead for all values of $q > 0.035$ query/m. The ratio OR(C/S) goes below 0.2 for $q \sim 0.22$–0.3 query/m, and goes below 0.1 for $q \sim 0.92$–9.8 query/m, approaching 6.6–9.9% (i.e. more than 90% in overhead savings) when q approaches 10 query/m.

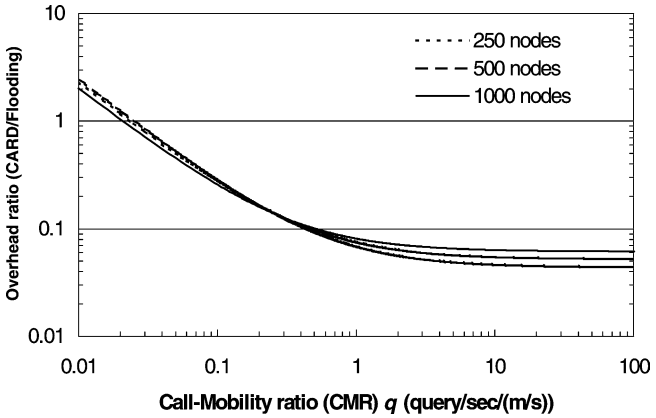


Figure 20. OR(C/F): the overhead ratio for CARD over flooding for various values of q.

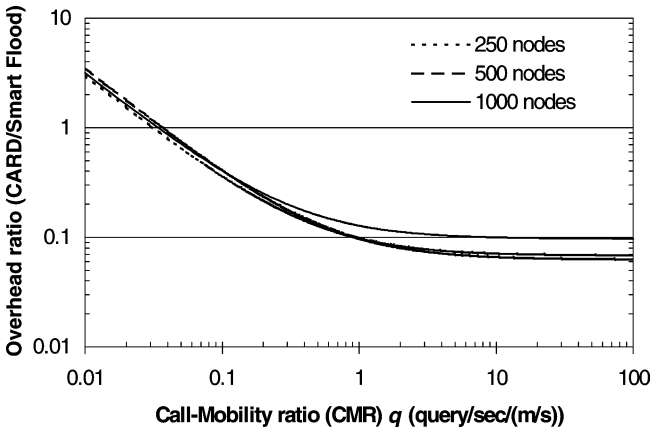


Figure 21. OR(C/S): the overhead ratio for CARD over smart flooding for various values of q.

5. Conclusions

In this paper we presented the CARD architecture for resource discovery and small transfers in large-scale ad hoc networks. The main contributions of this paper include the introduction of a contact-based architecture that explicitly trades-off route optimality (as in shortest path routes) for communication and energy efficiency, along with a proactive contact selection scheme to reduce vicinity overlap. Unlike existing routing protocols, instead of expending significant overhead to discover shortest path routes, CARD explicitly focuses on route discovery or query delivery with the least overhead, even if the routes used are suboptimal. We believe such trade-off is appropriate for our target applications, mainly resource discovery and small transfers. Salient features of our architecture include its ability to operate without requiring any location information or any complex coordination. In our architecture, each node proactively discovers resources within its vicinity. Based on small world concepts, we have utilized the notion of *contacts* to serve as short cuts that increase reachability beyond the vicinity. Two protocols for contact selection were introduced and evaluated: (a) probabilistic method and (b) edge method. The edge method was found to result in more reachability and less overhead during selection due to reduced backtracking, and was thoroughly analyzed over the various dimensions of the parameter space (including R, r, D, NoC, and network size). We further compared our approach to flooding and bordercasting. The overall overhead experienced by CARD was found to be significantly lower than the other approaches. Overhead savings are function of the query rate, reaching over 93% (vs. flooding and smart flooding) and over 80% (vs. bordercasting) in communication saving for high query rates; a drastic improvement in performance.

These results show a lot of promise for the contact-based approach to support short transfers in many applications of ad hoc networks. One possible future research direction to investigate is to integrate CARD with other routing protocols (e.g., ZRP), where CARD may be used as the resource discovery (and transaction routing) protocol. Similarly, we plan to investigate the integration of CARD in other data dissemination protocols for sensor networks, such as directed diffusion [12]. Instead of using flooding, CARD maybe use for efficient resource discovery.

References

[1] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu and H. Yu, Advances in network simulation, IEEE Computer (May 2000).

[2] T.-W. Chen and M. Gerla, Global state routing: A new routing scheme for ad-hoc wireless networks, in: *Proc. of the IEEE Internat. Conf. on Communications (ICC)* (1998).

[3] C.-C. Chiang, Routing in clustered multihop, mobile wireless networks with fading channel, in: *Proc. of IEEE SICON'97* (April 1997).

[4] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum and L. Viennot, Optimized link state routing protocol, in: *Proc. of IEEE INMIC* (2001).

[5] Z. Haas and M. Pearlman, The zone routing protocol (ZRP) for ad hoc networks, IETF Internet draft for the Manet group (June 1999).

[6] W. Heinzelman, J. Kulik and H. Balakrishnan, Adaptive protocols for information dissemination in wireless sensor networks, in: *The ACM MOBICOM Conf.*, Seattle, WA (August 1999).

[7] A. Helmy, Architectural framework for large-scale multicast in mobile ad hoc networks, *IEEE Internat. Conf. on Communications (ICC)*, Vol. 4, New York (April 2002) pp. 2036–2042.

[8] A. Helmy, Small worlds in wireless networks, IEEE Communications Letters 7(10) (2003) 490–492.

[9] A. Helmy, Mobility-assisted resolution of queries in large-scale mobile sensor networks (MARQ), Computer Networks (Special Issue on Wireless Sensor Networks) 43(4) (2003) 437–458.

[10] A. Helmy, TRANSFER: Transactions routing for ad-hoc networks with efficient energy, in: *IEEE Global Communications Conf. (GLOBECOM)* (December 2003).

[11] A. Helmy, S. Garg, P. Pamu and N. Nahata, Contact-based architecture for resource discovery (CARD) in large scale MANets, in: *IEEE/ACM IPDPS Internat. Workshop on Wireless, Mobile and Ad Hoc Networks (WMAN)* (April 2003) pp. 219–227.

[12] C. Intanagonwiwat, R. Govindan and D. Estrin, Directed diffusion: A scalable and robust communication paradigm for sensor networks, in: *ACM MobiCOM Conf.* (August 2000).

[13] D.B. Johnson and D.A. Maltz, The dynamic source routing protocol for mobile ad hoc networks, IETF Internet draft (October 1999).

[14] J. Li, J. Jannotti, D. Couto, D. Karger and R. Morris, A scalable location service for geographic ad hoc routing, in: *The ACM MOBICOM Conf.* (2000).

[15] J. Liu, Q. Zhang, W. Zhu, J. Zhang and B. Li, A novel framework for QoS-aware resource discovery in MANets, in: *IEEE Internat. Conf. on Communications (ICC)* (May 2002).
[16] W. Lou and J. Wu, On reducing broadcast redundancy in ad hoc wireless networks, IEEE Transactions on Mobile Computing 1(2) (2002).
[17] M. Mitzenmacher, Compressed bloom filters, in: *The Twentieth ACM Symposium on Principles of Distributed Computing (PODC)* (August 2001).
[18] S. Murthy and J.J. Garcia-Luna-Aceves, An efficient routing protocol for wireless networks, Mobile Networks and Applications (Special Issue on Routing in Mobile Communication Networks) (October 1996).
[19] S. Ni, Y. Tseng, Y. Chen and J. Sheu, The broadcast Storm problem in a mobile ad hoc network, in: *Proc. of the ACM MOBICOM Conf.* (August 1999) pp. 151–162.
[20] M. Pearlman and Z. Haas, Determining the optimal configuration for the zone routing protocol, IEEE Journal on Selected Areas in Communications 8 (1999) 1395–1414.
[21] C.E. Perkins and P. Bhagwat, Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers, ACM Computer Communications Review (October 1994) 234–244.
[22] C.E. Perkins, E.M. Royer and S.R. Das, Ad hoc on-demand distance vector routing, IETF Internet draft (October 1999).
[23] N. Sadagopan, B. Krishnamachari and A. Helmy, Active query forwarding in sensor networks (ACQUIRE), Ad Hoc Networks Journal (2004) to appear.
[24] N. Sadagopan, B. Krishnamachari and A. Helmy, The ACQUIRE mechanism for efficient querying in sensor networks, in: *First IEEE Internat. Workshop on Sensor Network Protocols and Applications (SNPA)*, in conjunction with IEEE ICC, Anchorage (May 2003) pp. 149–155.
[25] S. Wang and A. Helmy, Effects of small transfers and traffic patterns on performance and cache efficacy of ad hoc routing, (poster), in: *The ACM MOBICOM Conf. (The Ninth Annual Internat. Conf. on Mobile Computing and Networking)*, San Diego, CA (September 2003).
[26] D.J. Watts, The dynamics of networks between order and randomness, in: *Small Worlds* (Princeton Univ. Press, Princeton, 1999).
[27] D. Watts and S. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (4 June 1998).

Ahmed Helmy received his Ph.D. in computer science (1999), M.S. in electrical engineering (1995) from the University of Southern California, M.S. Eng. Math. (1994) and B.S. in electronics and communications engineering (1992) from Cairo University, Egypt. Since 1999, he has been an Assistant Professor of Electrical Engineering at the University of Southern California. In 2002, he received the National Science Foundation (NSF) CAREER Award. In 2000 he received the USC Zumberge Research Award, and in 2002 he received the best paper award from the IEEE/IFIP International Conference on Management of Multimedia Networks and Services (MMNS). In 2000, he founded – and is currently directing – the wireless networking laboratory at USC. His current research interests lie in the areas of protocol design and analysis for mobile ad hoc and sensor networks, mobility modeling, design and testing of multicast protocols, IP micro-mobility, and network simulation.
E-mail: helmy@usc.edu

Saurabh Garg received his M.S. in computer science from the University of Southern California in May 2003 and B.S. in engineering from University of Delhi, India. He is currently working as a Programmer for National Center for Ecological Analysis and Synthesis (NCEAS) affiliated with University of California, Santa Barbara. He has worked with researchers in fields of wireless networking, linguistics dialogue management and ecology. Before joining NCEAS he was working for Institute of Creative Technologies at the University of Southern California in the field of dialogue management. In wireless networking, he has worked on network simulation, protocol design and analysis.
E-mail: sgarg@usc.edu

Nitin Nahata received his M.S. in computer science from the University of Southern California in December 2002. He is currently working as a software engineer at Dynamix Technologies Inc.
E-mail: nnahata@usc.edu

Priyatham Pamu received his M.S. in computer science from the University of Southern California in December 2002.
E-mail: pamu@usc.edu