A scalable quorum based location update scheme for routing in ad hoc wireless networks

Ivan Stojmenović[1,2] *and Pedro Eduardo Villaneuva Peña*[2]
[1]SITE, University of Ottawa, Ottawa, Ontario K1N 6N5, Canada Ivan@site.uottawa.ca
2DISCA, IIMAS, UNAM, Ciudad Universitaria, Coyoacan, Mexico D.F. 04510, Mexico

Abstract

Most previously proposed position based routing algorithms for wireless ad hoc networks were based on forwarding the actual message along multiple paths toward an area where destination is hopefully located. The significant communication overhead can be avoided if the routing strategy is changed. We first propose to apply a variant of well-known route discovery scheme. The source node issues several search 'tickets' (each ticket is a 'short' message containing sender's id and location, destination's best known location and time that location is reported, and constant amount of additional information) that will look for the exact position of destination node. When the first ticket arrives at the destination node *D, D* will report back to source with brief message containing its exact location, and possibly creating a route for the source. The source node then sends full data message ('long' message) toward exact location of destination. We then propose a quorum based location update scheme, in which nodes report their new positions to their neighbors whenever a link is broken or created. After certain number of such link changes, nodes forward their new position to all nodes located in its 'column', that is, to the north and south of their current location with certain 'thickness' of reporting. The destination search then begins with two tickets being sent in the east and west direction, with certain 'thickness', looking for the most up to date information of destination's position. When the tickets reach each end of current 'row', the search is continued toward best reported destination position, with corrections along the path as better information becomes available closer to destination. One of tickets can be sent from the source directly toward destination, to take advantage of possibly correct information on destination location. The intersection of 'row' and 'column' can be guaranteed by adding outer face of the ad hoc network both of them. We show through simulation that the proposed routing and location update schemes provide scalable routing scheme with high success rates and reasonable communication overhead. The proposed method is also applicable in scenarios when all nodes move out of original region, preserving connectivity.

1. Introduction

Mobile ad hoc networks [G] consist of wireless hosts that communicate with each other in the absence of a fixed infrastructure. Routes between two hosts in network may consist of hops through other hosts in the network. The task of finding and maintaining routes in the network is nontrivial since host mobility causes frequent unpredictable topological changes. A number of protocols for achieving efficient routing have been recently proposed. They differ in the approach used for searching a new route and/or modifying a known route, when hosts move. The surveys of these protocols, that do not use geographic location in the routing decisions, are given in [BMJHJ, RS]. In this article we will discuss only location based approaches.

Macker and Corson [MC] listed qualitative and quantitative independent metrics for judging the performance of routing protocols. Desirable qualitative properties include: distributed operation, loop-freedom (to avoid a worst case scenario of a small fraction of packets spinning around in the network), demand-based operation, and 'sleep' period operation (when some nodes become temporarily inactive). Some quantitative metrics that are appropriate for assessing the performance of any routing protocol include [MC]: end-to-end data delay, average number of data bits (or control bits) transmitted per data bits delivered. The quantitative metrics include [BMJHJ, SL] (each of them is an average value): *hop count* (the number of edges, i.e. transmissions on the path from source to destination), *delivery rate* (the ratio of numbers of messages received by destination and sent by senders), and *flooding rate* (the ratio of the number of message transmissions and the shortest possible hop count between two nodes). Each transmission in multiple routes is counted, and message can be sent to all neighbors with one transmission. These metrics were considered when proposing the quorum based location update scheme in this paper.

Ad hoc networks are best modeled by *unit* graphs constructed in the following way. Two nodes *A* and *B* in the network are neighbors (and thus joined by an edge) if the Euclidean distance between their coordinates in the network is less than the transmission radius *R*, which is equal for all nodes. The unit graph is valid model when there are no obstacles in the signal path (e.g. a building). Ad hoc networks with obstacles can be modeled by subgraphs of unit graphs.

Ad hoc networks consist of autonomous nodes that run their routines in asynchronous fashion. The communication algorithms between nodes are therefore all distributed. However, we will use a subclass of distributed algorithms, called *localized* algorithms, which are distributed algorithms where simple local node behavior achieves a desired global objective. Localized algorithms therefore resemble the class of greedy sequential algorithms. In a localized routing algorithm, each node makes the decision to which neighbor(s) to forward the message based solely on the location of itself, its neighboring nodes, and destination. Localized algorithms reduce communication overhead for maintaining full networks information, including position of all nodes and activity status of all nodes, including static ones. More detailed discussion is given in two survey papers, one on location updates for efficient routing [S2], and one on position based routing schemes in ad hoc networks [GSB].

This paper deals solely with localized location based routing algorithms. Location information is available to nodes either through low power low cost GPS receivers or by measuring signal strengths and calculating relative coordinates [CHH]. The main goal of this paper is to describe a scalable location update scheme, by controlling the overhead of each routing task to $O(\sqrt{n})$, where n is the number of nodes in the network. Another goal is to present a method that does not require nodes to remain in a fixed region, which is the case with other known methods. In other words, the proposed method is applicable when all nodes move in more or less same direction (e.g. rescue team or soldiers). This feature is lacking in another scalable, home-agent based scheme, recently proposed independently in five articles [BBCGHL, MJKLD, PG, S3, WS].

This paper is organized as follows. Section 2 gives a review of localized routing algorithms and location update techniques in ad hoc networks. Section 3 proposes a routing strategy that mimics the route discovery process [BMJHJ], which separates the location update, destination search, path creation and data traffic phases in a routing process. Sections 4 and 5 propose new quorum based location update and destination search schemes. Section 6 deals with simulation and experimental results that confirm the efficiency of proposed methods. Future work is discussed in section 7.

2. Literature review

There exist three basic localized position based routing algorithms. Source or any intermediate node *S* will select one of its neighbors *A* according to a criterion. In the *DIR* method, derived from [BCSW, KV, KSU], the best neighbor has the closest direction (that is, angle) toward the destination. That is, neighbor with minimum angular distance from the imaginary line joining the current node and the destination is selected. *GEDIR* method [F, SL] selects neighbor *A* that is closest to the destination *D*. In *MFR* method [TK], the best neighbor *A* will minimize the dot product $DA \cdot DS$. Alternatively, one can maximize the dot product $SD \cdot SA$. Each method stops forwarding the message at a node for which the best choice is to return the message back to previous node. *GEDIR* and *MFR* methods are loop-free, while *DIR* method may create loops, unless past traffic is memorized or a timestamp is enforced [SL]. A routing algorithm that guarantees delivery in unit graphs is described in [BMSU]. It has been generalized in

[BFNO] for wider ranges of graphs. Power aware routing is discussed in [SL2]. The concept of internal nodes is introduced in [WL] and it significantly improves delivery rates and hop counts in routing algorithms [SSW]. Detailed survey on position based routing schemes in given in [GSB].

A detailed literature review on location update schemes for efficient routing in ad hoc networks is given in recent survey [S2]. The survey includes the DREAM algorithm [BCSW], LAR schemes [KV], ant based location updates [CL], Voronoi diagram and convex hull based approaches [S1], and doubling circles scheme [APL], among others. We will restrict the review only to results directly related to this paper.

Given a set *S* of *n* servers, a quorum system is a set of mutually disjoint subsets of S whose union is S. When one of servers require information from the other, it suffices to query one server from each quorum. It is possible to form quorums of size approximately $n^{1/2}$ [M]. For example, 25 servers can be organized into 5 rows and 5 columns. Each column serves as a quorum. Thus each node *(i,j)* (located in *i*-th row and *j*-th column) replicated its data to all servers *(i',j)* in its column. To extract the information from server *(i,j)*, server *(i', j')* may inquire within its *i'*-th row, and the server *(i',j)* will provide requested information. Variations of this scheme are used in fixed networks, where the set of queried servers is bound to contain at least one server that belonged to the quorum that received the latest update. Hence, each query returns the latest value of the queried data. Such a query and update strategies has been previously employed for location management in cellular networks [KAS, PHS, PS].

Karumanchi, Muralidharan and Prakash [KMP] discussed information dissemination in partitionable mobile ad hoc networks. They study the problem of getting the location of some other node in the network and the surroundings of that node (e.g. firefighter) without the need to route any message to that node. Thus their performance evaluation is limited in measuring the accuracy of the obtained information (i.e. the distance between found and exact location of other node). They use a well known approach for information dissemination by replicating information at multiple nodes acting as repositories, and employing quorum based strategies to update and query information. In [KMP], *n* nodes are divided into $n^{1/2}$ groups with $n^{1/2}$ nodes in each in two ways and preserve such quorums while nodes move. They also discussed the question when to update location, and argued that distance-based updates (based on absolute distance traveled since last update) and movement-based updates (based on the velocities of nodes) may have limited usefulness in ad-hoc networks (such location updates are used in [BCSW, KV]). For instance, nodes may move within a small circle, causing unnecessary location updates. Karumanchi, Muralidharan and Prakash [KMP] concluded experimentally that the best strategy is to update when a certain pre-specified number of links incident on a node have been established or broken since the last update. We decided to apply this strategy in our paper. However, the quorum organization in our paper differs from one in [KMP]. The quorums applied in this paper refer to nodes currently located on a north-south or east-west route form a source/destination, while quorums from [KMP] consist of very same nodes, whose movement makes them difficult to find by an algorithm more efficient than flooding.

Modifications of quorum based strategy for use in dynamic or partitionable servers has been considered in [ESC, H, KMP, HL1, HL2, LH]. The main idea in [ESC, H, KMP] is that each server (or node) selects one of quorums at random, to increase the chance of obtaining relatively up to date information in several 'columns'. Haas and Liang [HL1, HL2, LH] proposed another variant of quorum based distributed mobility management scheme. First, virtual backbone is initiated and maintained [LH]. Each node in the network is either in the virtual backbone or is at most *r* hops away from a virtual backbone node. Thus communication between two backbone nodes may go through both backbone and non-backbone nodes. Thus two ‘neighboring’ backbone nodes may be up to $2r+1$ hops away. A distributed scheme for initiating is based on selecting, repeatedly, a node with maximal number of unassigned *r* hop neighbors as backbone nodes, and assigning all its *r* hop neighbors to that node. We observe that this is, in fact, a well-known clustering algorithm [LG] generalized to *r* hops. The maintenance of cluster structure is known to require significant communication overhead (for instance, local changes may cause global updates by chain effect) [GKP, WL]. A significantly better backbone structure, one that does not require any communication overhead and provides connectivity between nodes, is described in [WL], and can be also generalized to *r*-hop case. Nodes in virtual backbone are database servers for location information.

They define a quorum system, that is a set of subsets such that any two subset intersect in small number of preferably constant number t of databases. Each subset then has the same size k. In [HL1] the choice of subsets is uniform and is performed by applying a centralized balanced incomplete block design algorithm. The selection of these subsets at random was discussed in [HL2]. When a node moves, it updates its location with one subset containing the nearest backbone node. Each source node then queries the subset containing its nearest backbone for the location of destination, and uses that location to route the message. The routing algorithm is not discussed in [HL1, HL2, LH]. It can be easily observed that location updates, destination searches are not local, and that they involve routing between backbone nodes. Thus backbone nodes must exchange their location information in order to perform their duties. It is not clear whether subset querying or backbone node communication can be than more efficiently than by flooding all nodes. Taking the entire overhead into account, it is not clear whether the whole routing algorithm will perform better than a simple flooding algorithm, with redundant retransmissions eliminated (roughly half nodes suffice to retransmit to achieve reliable flooding) [SSZ]. The flooding based routing algorithm does not require any location updates, requires no quorum based structure for its maintenance (which, in turn, relies on flooding anyway), and does not require communication overhead for its own backbone structure, which is dominating set as defined in [SSZ, WL].

Location update technique that is most relevant to this paper is the home agent based scheme, which has been independently proposed in five papers [BBCGHL, MJKLD, PG, S3, WS]. In this scheme, each node selects a home region, known to other nodes, and sends location updates to nodes currently located in its home region. In order to locate destination, nodes send destination search message (or full message in some variants) toward home agent region of destination, and redirect search or message from the region to current location of destination. The main advantage of proposed quorum based scheme over home agent based one is its flexibility when all nodes move to new region (in this case, all home regions become empty, and significant overhead is needed to handle such node movements).

3. A routing strategy for ad hoc networks

The problem of routing in mobile networks is apparently very difficult one, and so far no complete and satisfactory (in terms of almost guaranteed delivery without significant communication overhead) scheme was proposed. We propose to clearly divide the problem into several components, and study each of them separately. The approach is similar to source-initiated on-demand routing [BMJHJ] which initiates destination search by flooding the network, and then uses the created path (which is memorized in the process) to destination for routing. In [HOTV] it was argued that such flooding is the best routing method for very high mobility rates. In our approach, all the message traffic related to routing tasks is divided into four components as follows.

1) *Location update* messages are initiated by each node, which acts on its movement. Location updates are required by some other tasks as well (e.g. clustering, broadcasting, etc.).
2) *Destination search* messages, initiated by a source node, when it wants to route a message toward destination. This idea is similar to the dynamic source routing (e.g. [JM]) applied in ad hoc networks that do not use location in their routing decisions. However, it also differs significantly from these methods since they suggest full flooding as a means of searching, while we propose an intelligent search based on location of nodes. Also, we do not need to maintain any kind of routing tables, and believe that position information suffices to make intelligent decisions without them.
3) *Path creation* messages, initiated by destination upon receiving the first copy of a search message. The destination learns the location of sender from the search message and is able to find the best path accurately. Since the transmission speed is far greater than node movement speed, the path creation phase in a localized routing may, to a large extent, be considered as the operation performed on a static network. Thus routing algorithms for static networks (with known location of destination), such as those proposed in [KSU, SL, BMSU, SL2, SSW, TK], may be applied for the path creation phase. This

assumption is justified since each node mountains the list of neighbors and learns the exact location of destination, which is the only information needed for making a routing decision at each node.

4) *Data traffic* messages, initiated by source upon receiving reply from destination containing its exact location, possibly together with the path toward destination (e.g. in a form of next hop routing information at intermediate nodes [PR]). Alternatively, the source may attempt to create another path, knowing destination location accurately, by applying any localized routing algorithm defined on static networks.

In this routing scheme, we may also divide all messages into short and long ones. Short messages do not have the real information (to be forwarded to destination) as part of message (unless it is a very brief message, e.g. alarm), and therefore has much lesser number of bits than the message that contain the real information. Location update, destination search, and path creation messages are short messages. Location update messages are generated independently on routing request, as a preparation for successful destination search. Destination search and path creation messages are generated by routing requests. They are still communication overheads. When the real message, containing data to be forwarded to destination, is relatively ‘long’, this routing scheme is justified.

Note that, with this general routing scheme, the routing problem is divided into two components that may be investigated separately, as follows.

Component 1: *Location update and destination search schemes.*

Component 2: *Routing to a destination whose position is known* (includes path creation from destination to the source, and data traffic from source to destination).

Satisfactory localized solutions for path creation and data traffic phases are already proposed in [BFNO, BMSU, SL, SSW, SL2, S1]. Because of drawbacks of existing solutions for the location updates and destination search schemes, we shall concentrate on these two components in this paper. We shall propose new solutions for them in the next two sections.

The main difference between our proposed location update and destination search strategies and previously proposed analogous solutions (including non-position based route discoveries and route maintenance) is that full flooding was previously used as regular technique to construct the route, maintain the route or update the location in many cases. For example, when destination moves extensively and away from the source, no solution other than full flooding was suggested. We propose to deal with such movement pattern by reducing full flooding to row and column paths of certain thickness.

4. Location update

In this paper, we shall adopt the quorum-based idea to enable efficient routing in mobile ad hoc networks. Clearly, nodes in ad hoc network do not stay in the same 'column', and the distributed information may easily disperse due to node movement. Moreover, it is not clear what the 'column' is, and how all the nodes in a column, once defined, will receive latest updates. Nevertheless, we believe that this idea is worth pursuing, and much more promising than the one proposed in [HL1, HL2, LH].

The basic update procedure is performed by each moving node whenever it observes that, due to its movement, an existing edge will be broken (that is, the distance between two nodes becomes $>R$). Similarly, the same action may be taken when a new neighbor is detected (that is, in response to a message arriving from a new neighbor). The availability of geographic position information enables nodes to estimate the connection time with other nodes, as proposed in [SRV, SLG]. The connection time is defined as the estimated duration of a connection between two neighboring nodes. Neighboring nodes frequently update their location to each other, and this information may be used to estimate the direction and speed of their movements. In turn, this suffices to estimate the connection time. Let A and B be the two neighboring nodes which move at speeds a and b, respectively. Here, A and B are position vectors while a and b are directional vectors. At time t, they move to new positions $A'= A+at$ and $B'= B+bt$. They will loose their connection when the distance between them becomes $>R$, where R is the radius of corresponding unit graph. The time t when the connection will be lost can be estimated by solving quadratic equation

$|A'B'|=|B\text{-}A+(b\text{-}a)t|=R$ [SRV,SLG]. Note that this adopted criterion allows nodes to move with same speed and in the same direction (e.g. a road) without generating any location update message.

In response to detected broken link, each node *A* will broadcast a message containing its new location information to all neighbors that are at most *t* hops away from *A*. Thus, for $t>1$, the information will be retransmitted by neighboring nodes. These re-transmissions may cause broadcast storm problem [NTCS], and therefore its efficiency for larger *t* is doubtful. This method was adopted in our experiments, in accordance with unit graph definition. Alternatively, the location information may be transmitted to all nodes at distance $\leq tR$ with one transmission, if nodes may adjust transmission power, and spend more energy for short messages. Lin and Liu [LL] discussed this difference and even proposed an extreme difference in radii for short and long messages. Nodes in [LL] are able to send their new location to all other nodes in network with a single broadcast (single-hop network for location updates). However, when sending exact data, the network is treated as multi-hop one. We believe that a single-hop network is unrealistic if scalability of algorithm is one of main goals. However, for location updates, transmissions with radius *tR* can be considered, where *t* is network parameter. In both cases, location updates beyond immediate neighbors may facilitate destination search. However, in response to newly detected neighbor (only direct neighbors will respond), nodes will reply using transmission radius *R*, and other nodes within radius *R* will also learn updated position of that node. This is preferred solution since normally these nodes do not change their neighborhood, and new neighbor is the only one in real need of location information. Note that nodes do not transmit their location in response to location update received to already existing neighbors, in order to control the overhead, and restrict updates to actions caused by moving nodes.

The main location update method is to forward the new location information (and node's identifier) within a 'column' in the network, in the following way. Each node uses a counter to count the number of previously made changes in edge existence (the number of created or broken edges). When the counter reaches a fixed threshold value *e*, location information is forwarded along the 'column', and *e* is reset to 0. Node *A* that reaches threshold value will initiate location update within column as follows. *A* transmits update information to all its neighbors (within transmission radius *R*). The same message also includes location or *ID* of the northernmost neighbor *B* of *A* (and similarly southernmost neighbor *C*). The selected neighbor *B* retransmits the message in the same fashion (but in the north direction only). This search continues until a node *B* has no neighbor that is more to the north than *B*. The initiator of location update *A* sends the update in both directions, while other nodes follow only one of directions. The main location update route, in north-south direction, is created in the process. It consists of all nodes on the route and all of their neighbors. This is refereed to as the column with thickness $p=1$. Neighboring nodes may retransmit to enlarge the thickness to $p=2$ (some of nodes may not to retransmit if an optimization [SSZ] is applied). The process may continue until arbitrarily selected thickness *p* is reached. This method is used in our experiments. Note that an alternate solution may apply transmission radii *pR* instead, similarly as commented above for update in node's neighborhood.

The frequent problem with the scheme is that the northernmost node as determined by the northward update may be only locally northernmost. A 'horizontal' destination search can miss such a node, which can remain 'below' it. To overcome this problem, each locally northernmost node may switch to *FACE* mode [BMSU] until another node, more northern, is found on a face. Such nodes may return back to regular upward move. This switch can be repeated few times. The *FACE* algorithm can be improved by applying a short-cut scheme [DSW]. The final result will be that all nodes at the outer face of the network will receive location update. This method guarantees that 'horizontal' destination search and 'vertical' location update will intersect at one of nodes on the outer face. The drawback is that these nodes will have more traffic demands.

Each node receiving new location information for *A*, by means of any of transmissions, will record it, together with date of update. Moreover, each node transmitting any kind of message will include its own location with the message, thus providing an update on its own position to its neighbor within the selected transmission radius.

5. Destination search

The search for destination is performed in the following way. The source *S* broadcasts search request among neighbors that are located at most *q* hops away, where *q* is another network parameter. If destination is *q+1* hops away, it will be located (the destination could be neighbor of a node *s* hops away). The search messages always carry the location of sender (and time), and nodes receiving the information will update sender's location in their tables. If there is no reply, the search continues in east-west direction, in similar way as for the location update (that is, easternmost and westernmost neighbors will extend the search in corresponding directions). The search message includes time of last available information, and other nodes are requested to provide more up to date information, if they have. This process goes as follows. Let *A* be a node of east-west search route. All nodes that are at most *s* hops away (where *s* is another network parameter) receive the search message using a variant of broadcasting scheme [SSZ]. Each node *C*, which has more up to date information about *D* than *A* (or a neighbor that previously sent similar message), responds by sending location information about *D*, and time location is recorded, to one of its neighbors that is closest to *A* (to guaranty delivery, algorithm [BMSU] can be used instead). In order to synchronize the process and avoid too many collisions, a MAC variant of IEEE 802.11 with waiting times inversely proportional to $|AC|$ can be used, so that the process starts from furthest neighbors and continues toward *A*. If node *C* receives one or few such messages from nodes that are further from *A* than *C* is, it will choose the most up to date among them (including its own information), and forward it to its neighbor that is closest to *A*. If *C* receives similar message from one of its neighbors, containing same or more up to date information, it will stop its own forwarding process. Also, if *C* receives no message, and has no better information, it remains silent. Eventually, node *A* may receive better information about location of *D*. The best information is then forwarded in given direction (eastward or westward) to furthest possible neighbor, and forwarded similarly until a node is reached that has no neighbor in followed east or west direction.

When easternmost and westernmost nodes are reached, the search strategy changes. The message search is then oriented toward the destination, using latest available information for each search message. There are three searches initiated. The first one originates at sender node *S*, using the best information collected within neighbors at most *q* hops away. This search does not need to wait for the result of searches in the east and west directions. The other two searches are initiated by easternmost and westernmost nodes in a given 'row'. Each of three search tasks follows a path toward destination, using *GEDIR* strategy [SL], or *GFG* [BMSU] (to guaranty process continuation). That is, at each step, the neighbor closest to destination is selected to forward the message. Since any of messages between two nodes can be heard by all nodes within radius *R*, if any of them has a better information about destination, it will respond, and the new information will take precedence over the former one in forwarding decision. Note that the location of source node is updated at each node that hears any of destination search or path creation messages.

Note that variants of destination search messages using variable transmission radii at nodes may also be considered, similarly as in the case of location updates.

Figure 1 illustrates location update and destination search schemes, where we assume $t=p=s=1$. Destination node *D* moves (other nodes in Fig. 1 are assumed to be static) from positions *D1* to *D2* and finally to *D3*, and causes some edges to brake or to be created (indicated by arrows, with nodes *A1*, *A2*, *A3* and *A4*). At position *D3*, it decides to send location update in its current 'column' by sending its position in the north direction (it does so in the south direction as well, but there is no neighbor in south direction in Fig. 1). The main update path is indicated in bold lines, and bold long dashed lined indicate some other nodes that hear the update message. Source node *S* initiates destination search in the east-west direction (in Fig. 1 it has only neighbors in west direction). The search path is indicated in bold line, and bold short dashed lines connect some other nodes that observe the search. The location update column and destination search row intersect in Fig. 1 in seven nodes. The easternmost and westernmost points in the row, and node *S* then turn the search toward learned position of *D*. In Fig. 1, point *W* learned the up to date information, and may find the destination by applying any localized routing algorithm for static nodes (in

Fig. 1, *GEDIR* algorithm would produce path *W-B-C-A3-D3*). Figure 2 gives another illustration for vertical location updates and horizontal destination search.

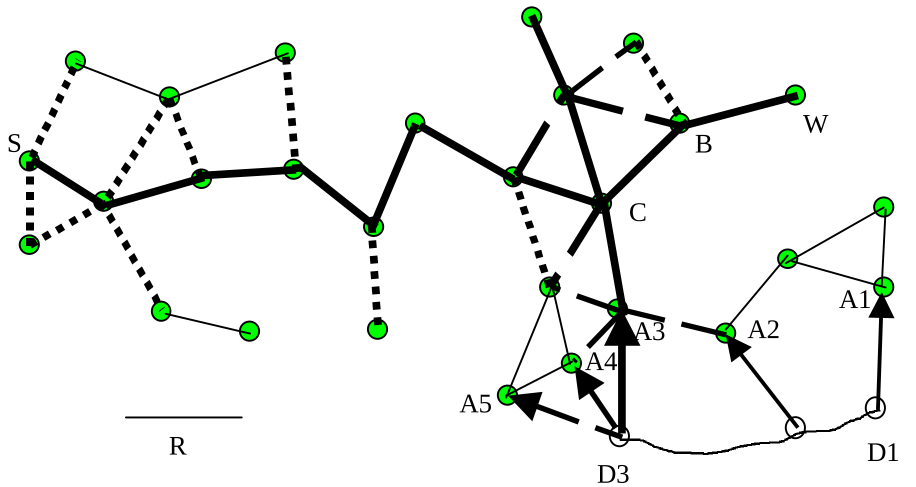

Figure 1. Location update from D3 and destination search from S

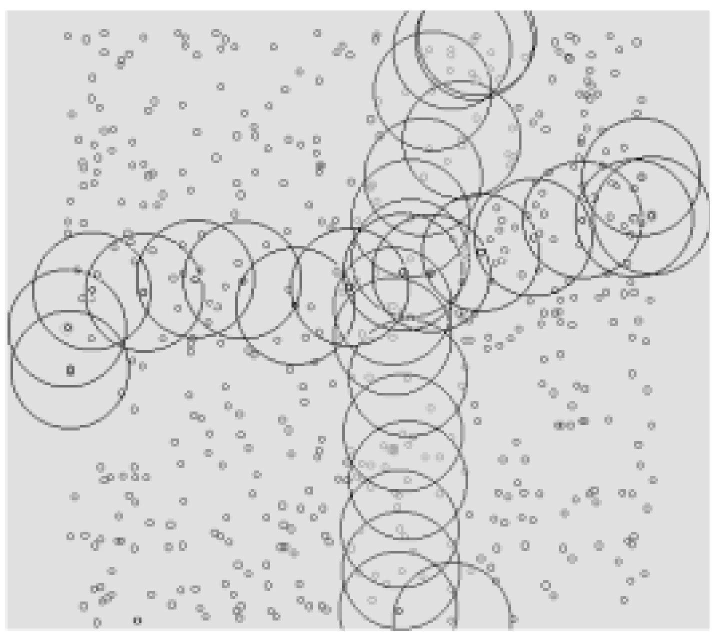

Figure 2. Horizontal destination search and vertical location update intersection

We shall now illustrate the variant that guarantees the intersection of ‘row’ and ‘column’ in Fig. 3. The ‘Row’ *SUV-SW* constructed by destination search from *S,* and ‘column’ *DABC-DEF*, constructed by location update by *D*, do not intersect. However, they both can extend their search and update on the outer boundary of the set (indicated by ‘heavy’ bold lines), which is constructed using planar Gabriel graph (indicated by ‘normal’ bold lines). The unit graph in Figure 3 consists of all indicated edges. An edge *UV* belongs to Gabriel graph if an only if $|PM|>|UV|/2$ for every other node *P*, where *M* is midpoint of *UV* (common neighbors are the only nodes to be checked, and this can be done without any communication initiated for that purpose). The planarity and connectedness of Gabrial graph is discussed in [BMSU].

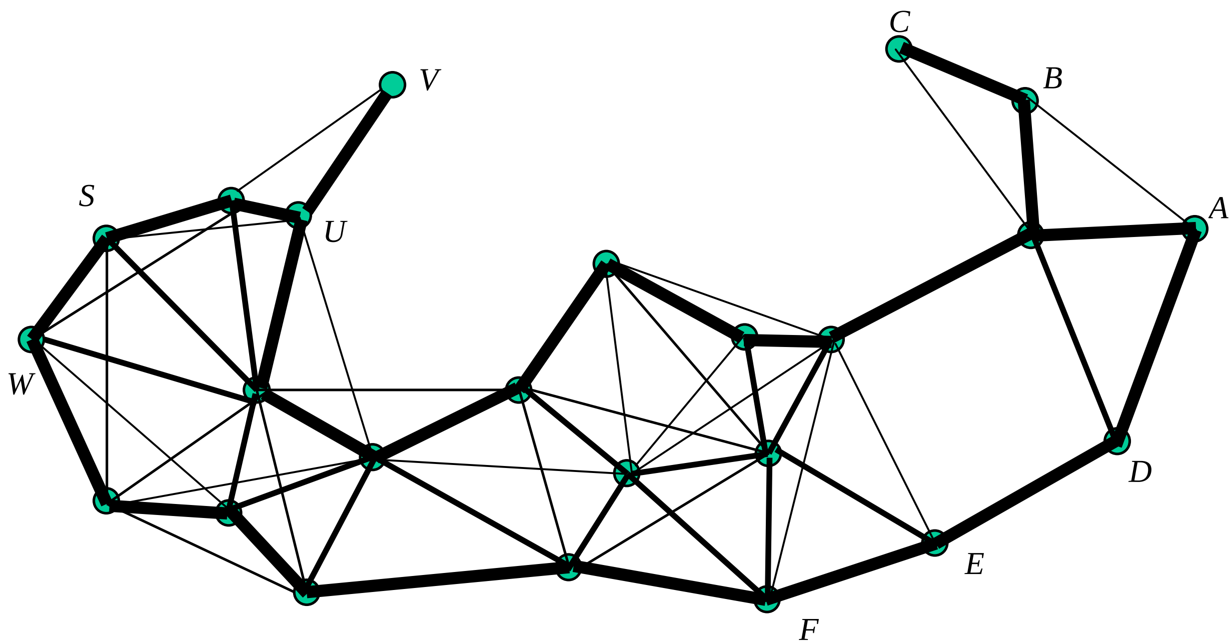

Figure 3. ‘Row’ *SUV-SW* and ‘column’ *DABC-DEF*, and outer boundary in ‘heavy’ bold lines

In order to follow the outer boundary from *D* or *S*, the routing scheme from [BMSU] needs minor modification. It is sufficient to ignore the position of destination, that is to assume that the face holding packet never changes, with starting point being one of locally extreme points in north, south, east or west direction. Assume also that clockwise direction is followed. Then each node, currently with the packet, will forward it to the neighbor (only edges belonging to Gabriel graph are eligible) that has closest direction to the direction of incoming packet, measured clockwise. For instance, let assume that destination or update packet arrived from node *D* to node *E*. With respect to direction *ED*, node *E* has two neighbors with edges in Gabriel graph, and node F has smaller clockwise angle with respect to *ED* between the two, thus message is forwarded to *F*. The problem here is that nodes cannot locally determine whether they belong to the outer boundary. To overcome this problem, messages may include the northernmost (similarly for other three directions) node ‘visited’ along the location update (and similarly destination search), and stop following outer boundary upon return to the same node. This is, in fact, the method for determining whether destination is disconnected from source.

The nodes on the outer boundary may be interpreted as a kind of location servers for the network. In fact, it suffices that only these nodes memorize the locations of all nodes from the network. In order to identify the location server nodes, one more pass around outer boundary, upon returning to the northernmost node, is needed (that is, between first and second return to the northernmost node, message will visit all boundary nodes). The overhead can be reduced if only one of destination searches or location update follows the outer boundary, while the other will stop when it reaches the first such node. In order to propose reactive algorithm, we suggest that location update messages stop at the first outer boundary node, without following the boundary. It further suffices that the updates are performed in only one direction, say northward. Since nodes on the boundary may move and thus loose their status, a mechanism is needed to transfer the location database to one of remaining outer boundary nodes when moving nodes loose such status. This can be achieved by transmitting location database upon moving out of boundary, with information being picked up by all neighboring nodes. The thickness of the boundary may also be increased to increase method reliability.

Finally, the protocol needs also to handle node mobility between two location updates. The destination search will reach the area where node was located when it last initiated ‘column’ location update. Therefore moving node needs to maintain ‘connection’ with nodes remaining in that area. The best method may depend on the size *e* of ‘breakage’ counter. Reactive protocols being preferred, we suggest that the node closest to reported destination position initiates geocasting up to distance of *e* hops. That is, all nodes up to distance of *e* hops from it will receive the destination search message. The parameter can be increased for better reliability.

6. Performance evaluation

Our criteria for evaluating a location update scheme include loop-freedom, scalability, and guaranteed success of location update (providing ‘fresh’ information to destination search messages). It appears that the proposed method, in fact, has no competitor when all criteria are considered. The authors of previously proposed ‘quorum’ based location update scheme [HL1] never implemented it, and it appears that intelligent flooding as destination search, with zero location updates, will perform better. The request zone based methods [BCSW, KV] are not loop-free, not scalable, and do not guarantee search success. The improved version [S1] is loop-free and reduces communication overhead, but still uses flooding occasionally. The doubling circle method guarantees success of routing task, but the update occasionally may convert to flooding, thus again has limited scalability. The only scalable method appears to be home-agent based [BBCGHL, MJKLD, PG, S3, WS]. However, it will suffer when all nodes move out of original region. Moreover, even if the node movement is restricted inside a zone, the selected homes of many nodes may be empty. Our proposed quorum based method has no such deficiencies. The comparison between the two methods may choose various network settings, and the comparison will depend on the setting. In a truly random node movements, moderate or high density, we believe that home-agent based method may have smaller overhead. But reality may be different from the setting in many scenarios. In case of joint movement in the same direction, home-agent method will perform worse than even some other method listed here. We have therefore declared the proposed quorum based method as the ‘winner’ based on desired characteristics, and only implemented and measured its performance.

The parameters in the simulation are chosen as follows. The number *n* of nodes in the network was *n*=20, 50, 100, to address scalability issue. The average node degree *k* ranges between 3 and 14. The corresponding value for radius *R* is found as in [SL] (by sorting all possible edges and choosing *R* to provide exactly *nk/2* edges). For each graph, *n* nodes select their *x* and *y* coordinates at random from the interval [0,100]. Graphs are not checked for connectivity, since node movement may disconnect each graph and reconnect it frequently. Thus initial connectivity does not secure connectivity during node movements.

The parameter *t* for link distance of local updates was set to *t=1*. The parameters *s*, *p* for the column update and destination searches are tested for values 1, 2, and 3 each, with all possible 9 variations. The threshold counter rate *e* for triggering column update was fixed at *e*=3. The parameter *q* for local search was set to *q=2*.

The movement of nodes is similar to the one used in [CN, JM, MZ]. Each node generates a random number *wait* (rounded to nearest integer) in interval *[0..maxwait]*. The node does not move for *wait* seconds. This is called station time. When this time expires, node generates a random number *travel* in interval *[0,maxtravel]* (rounded to nearest integer), and generates a new position within the same square at random. Node then moves from old position to new position along the line segment joining them at equal speed for the duration of *travel* seconds. Upon arriving at new location, node again chooses waiting period etc. Mobility rate is given by formula *mobrate = maxtravel/(maxtravel+maxwait)*. Parameter *mobrate* received values 10%, 20%, 30%, 50%, 100%, while *maxtravel* was set to 100 seconds.

The following performance measurements are taken in the experiment. Each value is an average over 30 graphs and 100 source-destination pairs per graph. Only location updates and destination search messages were implemented, not the other two phases that are routing tasks with known destination. The actual routing tasks, including reporting back from destination to source, are simulated separately [BMSU, SL]. Thus success rate is measured only in terms of delivering destination search message to destination node.

1) Success rate, equal to the percentage of successful destination search requests. It was measured only for source destination pairs that were connected (this was tested using a shortest path algorithm).
2) Flooding rate, that is, the number of transmissions divided by *n*. Flooding rate is our selected measure of communication overhead.

As in [APL], no link details (such as MAC protocol, link errors, frame retransmissions etc.) are modeled here. We assume that no collisions occur in any of simultaneous message transmissions, and all transmitted messages are correctly received. Thus each node is assumed to be an infinite-buffer, store-and-forward queuing station. Error free transmission in any direction, or in all directions simultaneously, is also assumed.

We first studied the case of static networks, in order to verify the method effectiveness under ideal circumstances. If two nodes are connected, do location updates and destination search messages interest at a node, and at what cost? The number of message transmissions in location updates, and destination searches measures the cost. The success rate is measured by the percentage of connected source-routing pairs for which the corresponding row and column intersect.

	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12
CR p=1 s=1	0.307	0.542	0.7	0.739	0.881	0.955	0.962	0.989	0.988
CR p=s=2	0.741	0.905	0.958	0.987	0.991	0.996	0.998	1	1
CR p=s=3	0.92	0.984	0.995	0.999	1	1	1	1	1
CRCR p=s=1	0.405	0.666	0.834	0.912	0.96	0.994	0.99	0.994	0.995
CRCR p=s=2	0.878	0.957	0.986	0.999	0.998	1	1	1	1
CRCR p=s=3	0.969	0.994	1	1	1	1	1	1	1
RORO p=s=1	0.429	0.693	0.845	0.923	0.985	0.985	0.989	0.998	0.999
RORO p=s=2	0.868	0.956	0.988	0.998	0.999	1	1	1	1
RORO p=s=3	0.978	0.997	0.999	0.999	1	1	1	1	1
4L p=s=1	0.503	0.687	0.933	0.956	0.966	0.993	0.997	0.998	0.999
4L p=s=2	0.927	0.982	0.993	1	0.999	1	1	1	1
4L p=s=3	0.976	0.995	0.999	1	1	1	1	1	1
OUTER p=s=1	1	1	1	1	1	1	1	1	1

Table 1. Success rate for quorum based schemes for *n*=100 nodes

The initial experiments were performed with basic location update, called *CR* (Column for location updates, Row for destination search). The results were encouraging, but not fully satisfactory. We then experimented with three more variants, *CRCR* (location updates and destination searches are performed on both rows and columns), *RORO* (random direction and its orthogonal direction for both location updates and destination searches; the two directions are chosen independently), and *4L* (four lines, row, column, and two lines at angles $\pi/4$ between them, for both location updates and destination searches). Finally, outer boundary for location updates is added (*OUTER* method). The results are given in Tables 1 and 2. Several conclusions can be made from the obtained results. The simplest *CR* method already performs very well for dense networks (success rate over 95% for average degrees $k\geq 9$, with moderate flooding rate, below 22%). Doubling density ($p=s=2$) increases significantly success rate for sparse networks. CRCR method increases success rate over 10% (for degrees up to 8) with respect to *CR*, with flooding rate below 55% ($p=s=1$). *RORO* method did not differ significantly from *CRCR* one. *4L* method increased success rate, but flooding rate was too high. Finally, *OUTER* method guaranteed success while performing stable with respect to flooding rate, below 45%, since the underlying Gabriel graph is not affected too much by changes in graph density. Overall, it appears that simplest method suffices for dense networks, while *OUTER* one works well for all densities.

	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12
CR p=1 s=1	3.256	6.383	8.773	10.48	13.589	16.416	17.983	20.002	21.004
CR p=s=2	10.938	20.081	25.849	33.062	37.698	42.207	47.199	50.965	53.709
CR p=s=3	24.663	37.306	48.518	56.142	64.111	70.831	76.383	82.196	85.107
CRCR p=s=1	7.687	16.351	23.672	31.407	37.635	44.677	48.176	51.374	54.922
CRCR p=s=2	22.169	35.395	51.748	61.36	68.591	78.06	84.971	86.814	88.965
CRCR p=s=3	37.677	56.983	69.527	83.243	91.141	94.337	96.589	98.803	98.991
RORO p=s=1	8.54	17.424	26.259	32.437	42.013	47.478	50.602	26.194	60.871
RORO p=s=2	20.986	34.591	52.293	65.279	72.719	82.256	84.59	91.887	92.968
RORO p=s=3	42.378	56.885	75.922	84.548	93.205	95.843	97.838	98.49	99.569
4L p=s=1	8.54	17.424	26.259	32.437	42.013	47.478	50.602	26.194	60.871
4L p=s=2	20.986	34.591	52.293	65.279	72.719	82.256	84.59	91.887	92.968
4L p=s=3	42.378	56.885	75.922	84.548	93.205	95.843	97.838	98.49	99.569
OUTER p=s=1	40.12	41.2	41.81	42.22	42.89	43.15	43.94	44.43	44.83

Table 2. Flooding rate (in %) for quorum based schemes for n=100 nodes

The experiments with moving nodes were then performed. Message propagation speed was assumed to be significantly greater than movement speed, so that routes do not change while forwarding is in progress. The success rate did not drop significantly, while flooding rate was affected, and had to be measured differently, as the number of location updates in unit time divided by number of nodes. In order to normalize the measure, the unit time was the time between two routing tasks in the network. The actual data received were still proportional to nodes mobility speeds and rates, and thus their interpretation is difficult. The encouraging information is that *OUTER* method still guaranteed delivery, with rare exceptions of a nodes changing their speed and moving out of the area where they were supposed to connect other nodes. For instance, breakage counter e was used for local information, but occasionally the links between current and last updated location information of destination did not exist. Therefore some modifications and improvement are needed to address this issue. Nevertheless overall results were very positive.

7. Future work

This paper proposed location update and destination search algorithms. Once destination is found, an algorithm described in [BMSU] may be used for path creation and data traffic messages with guaranteed delivery. It is interesting to observe that this proposed scheme contains elements of all three basic routing algorithms: *GEDIR*, *DIR* and *MFR*. *MFR* algorithm is used for location update and first part of destination search, *DIR* algorithm is used in the *FACE* portion of *GFG* algorithm [BMSU] when *GEDIR* fails, and *GEDIR* is used otherwise.

Our location update scheme works well even if nodes move together to a new area (e.g. group of vehicles or soldiers). The columns and rows defined in our location update scheme are relative, not absolute. It also adapts well to synchronous node movements (such as vehicles on a highway), keeping mutual distances but moving at high speed. In this case, the edge disconnection is predicted based on estimated node position, using their reported speed and direction of movement.

Unsuccessful searches for destination may be converted into full flooding at termination nodes, if guaranteed delivery is required, and nodes have high mobility rates such that presented methods become unreliable. If this event occurs very rarely, it shall have no significant impact on communication overhead. In case of frequent failures, the quorum-based strategy may need further improvements. At very high speed, it may not be possible to do anything better than flooding, as observed in [HOTV]. We also believe that any location update method, no matter how clever is, will only work well up to a certain speed limits. Nevertheless it remains a challenge to push that limit as far as possible, with a loop-free and scalable method that does not resort to flooding too often.

The concept of internal nodes [SSW, WL] way be used to improve the performance of destination search, location update and path creation phases. Several definitions of internal nodes are proposed in [SSW, WL]. The simplest of them is defined as follows. A node is intermediate node if there exist two of its neighbors that are not directly connected [WL]. Thus intermediate node belongs to a shortest path between some pairs of nodes. Nodes which are not intermediate may stop forwarding the message, if it is sent to them in any of the phases (of course, such nodes may hear transmissions, but should not be selected as destination by any of the paths in any of the phases). The performance of any of the phase may be greatly improved by restricting the paths to internal nodes only. Longer and more successful path will be created that way.

We believe that quorum based idea for routing in ad hoc networks has the potential to be very efficient, in terms of small hop counts, almost guaranteed delivery, and small communication overhead, compared to other existing schemes. It is expected that the candidate methods will be compared in future with a common simulator and appropriate medium access layer.

References

[APL] K.N. Amouris, S. Papavassiliou, M. Li, A position based multi-zone routing protocol for wide area mobile ad-hoc networks, Proc. 49th IEEE Vehicular Technology Conference, 1999, 1365-1369.

[BBCGHL] L. Blazevic, L. Buttyan, S. Capkun, S. Giordano, J.-P. Hubaux and J-Y. Le Boudec, Self-organization in mobile ad hoc networks: the approach of terminodes, IEEE Communication Magazine, June 2001, 166-175.

[BCSW] S. Basagni, I. Chlamtac, V.R. Syrotiuk, B.A. Woodward, A distance routing effect algorithm for mobility (DREAM), Proc. MOBICOM, 1998, 76-84.

[BFNO] L. Barriere, P Fraigniaud, L. Narajanan, and J. Opatrny, Robust position based routing in wireless ad hoc networks with unstable transmission ranges, Proc. of 5th ACM Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications DIAL M01, 2001, 19-27.

[BMJHJ] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, Proc. MOBICOM, 1998, 85-97.

[BMSU] P. Bose, P. Morin, I. Stojmenovic and J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, 3rd int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Seattle, August 20, 1999, 48-55; ACM/Kluwer Wireless Networks, 7, 6, November 2001, 609-616.

[CHH] S. Capkun, M. Hamdi, J.P. Hubaux, GPS-free positioning in mobile ad-hoc networks, Proc. Hawaii Int. Conf. on System Sciences, January 2001.

[CL] D. Camara and A.F. Loureiro, A novel routing algorithm for ad hoc networks, Proc. HICSS, Hawaii, January 2000.

[CN] S. Chen and K. Nahrstedt, Distributed quality-of-service routing in ad hoc networks, IEEE J. Selected Areas in Communications, 17, 8, August 1999, 1488-1505.

[ESC] A. El Abbadi, D. Skeen, F. Cristian, An efficient fault-tolerant algorithm for replicated data management, Proc. 5th ACM SIGACT-SIGMOD Symp. Principles of Database Systems, 215-229, 1985.

[F] G.G. Finn, Routing and addressing problems in large metropolitan-scale internetworks, ISI Research Report ISU/RR-87-180, March 1987.

[G] S. Giordano, Mobile ad hoc networks, to appear in: Handbook of Wireless Networks and Mobile Computing (I. Stojmenovic, ed.), John Wiley & Sons, 2002.

[GKP] M. Gerla, T.J. Kwon and G. Pei, On demand routing in large ad hoc wireless networks with passive clustering, Proc. IEEE WCNC, September 2000.

[GSB] S. Giordano, I. Stojmenovic, Lj. Blazevic, Position based routing algorithms for ad hoc networks: A taxonomy, July 2001, http://uxdea4.iimas.unam.mx/~ivan.

[H] M. Herlihy, Dynamic quorum adjustment for partitioned data, ACM Transactions on Database Systems, 12, 2, 170-194, June 1987.

[HL1] Z.J. Haas and B. Liang, Ad hoc mobility management with uniform quorum systems, ACM/IEEE Transactions on Networks, 7, 2, April 1999, 228-240.

[HL2] Z.J. Haas and B. Liang, Ad-hoc mobility management with randomized database groups, Proc. Of IEEE ICC, Vancouver, June 1999.

[HOTV] C. Ho, K. Obraczka, G. Tsudik, K. Viswanath, Flooding for reliable multicast in multi-hop ad hoc networks, Proc. MOBICOM, 1999, 64-71.

[JM] D. Johnson and D. Maltz, Dynamic source routing in ad hoc wireless networks, in Mobile Computing (T. Imielinski and H. Korth, eds), Norwell, MA: Kluwer, 1996.

[KAS] G. Krishnamurthy, A. Azizoglu, A.K. Somani, Optimal location management algorithms for mobile networks, Proc. 4th Annual ACM/IEEE Int. Conf. on Mobile Computing and Networking MOBICOM'98, 223-232, 1998.

[KMP] G. Karumanchi, S. Muralidharan and R. Prakash, Information dissemination in partitionable mobile ad hoc networks, Proc. IEEE Symp. On Reliable Distributed Systems, Lausanne, Oct. 1999.

[KSU] E. Kranakis, H. Singh and J. Urrutia, Compass routing on geometric networks, Proc. 11th Canadian Conf. on Computational Geometry, Vancouver, August, 1999.

[KV] Y.B. Ko and N.H. Vaidya, Location-aided routing (LAR) in mobile ad hoc networks, Proc. MOBICOM, 1998, 66-75.

[LG] C.R. Lin and M. Gerla, Adaptive clustering for mobile wireless networks, IEEE J. on Selected Areas in Communications, 15, 7, 1997, 1265-1275.

[LH] B. Liang and Z. J. Haas, Virtual backbone generation and maintenance in ad hoc network mobility management, INFOCOM, Israel, 2000.

[LL] C. R. Lin and J.S. Liu, QoS routing in ad hoc wireless networks, IEEE J. Selected Areas in Communications, 17, 8, Aug. 1999, 1426-1438.

[M] M. Maekawa, A $n^{1/2}$ algorithm for mutual exclusion in decentralized systems, ACM Transactions on Computer Systems, 14-159, May 1985.

[MC] J.P. Macker and M.S. Corson, Mobile ad hoc networking and the IETF, Mobile Computing and Communications Review, 2, 1, 1998, 9-14.

[MJKLD] R. Morris, J. Jannotti, F. Kaashoek, J. Li, D. Decouto, CarNet: A scalable ad hoc wireless network system, 9th ACM SIGOPS European Workshop, Kolding, Denmark, Sept. 2000.

[MZ] A.B. McDonald and T.F. Znati, A mobility-based framework for adaptive clustering in wireless ad hoc networks, IEEE J. Selected Areas in Communications, 17, 8, August 1999, 1466-1487.

[NTCS] S.Y. Ni, Y.C. Tseng, Y.S. Chen, J.P. Sheu, The broadcast storm problem in a mobile ad hoc network, Proc. MOBICOM, Seattle, Aug. 1999, 151-162.

[PR] C. Perkins and E.M. Royer, Ad hoc on demand distance vector (AODV) routing, internet draft, August 1998.

[PG] G. Pei and M. Gerla, Mobility management for hierarchical wireless networks, Mobile Networks and Applications, 6, 4, August 2001, 331-337.

[PHS] R. Prakash, Z.J. Haas, M. Singhal, Load balanced location management for mobile systems using dynamic hashing and quorums, TR-UTDCS-05-97, The University of Texas at Dallas, October 1997.

[PS] R. Prakash and M. Singhal, A dynamic approach to location management in mobile computing systems, Proc. 8th Int. Conf. on Software Eng. & Knowledge Eng. SEKE'96, 488-495, Lake Tahoe, Nevada, June 1996.

[RS] S. Ramanathan and M. Steenstrup, A survey of routing techniques for mobile communication networks, Mobile Networks and Applications, 1, 2, 1996, 89-104.

[SL] Ivan Stojmenovic and Xu Lin, Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks, IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 10, October 2001, 1023-1032.

[SL2] Ivan Stojmenovic and Xu Lin, Power-aware localized routing in wireless networks, IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 11, November 2001, 1122-1133.

[SRV] I. Stojmenovic, M. Russell, and B. Vukojevic, Depth first search and location based localized routing and QoS routing in wireless networks, IEEE International Conference on Parallel Processing, August 21-24, 2000, Toronto, 173-180.

[SSZ] Ivan Stojmenovic, Mahtab Seddigh and Jovisa Zunic, Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks, IEEE Transactions on Parallel and Distributed Systems, to appear.

[SLG] W. Su, S.J. Lee, M. Gerla, Mobility prediction in wireless networks, Proc. IEEE MILCOM, October 2000.

[DSW] S. Datta, I. Stojmenovic, J. Wu, Internal nodes and shortcut based routing with guaranteed delivery in wireless networks, Cluster Computing, to appear.

[S1] I. Stojmenovic, Voronoi diagram and convex hull based geocasting and routing in ad hoc wireless networks, Computer Science, SITE, University of Ottawa, TR-99-11, December 1999, www.site.uottawa.ca/~ivan .

[S2] Stojmenovic I., Location updates for efficient routing in wireless networks, in: Handbook on Wireless Networks and Mobile Computing, John Wiley & Sons, 2002, to appear, www.site.uottawa.ca/~ivan .

[S3] I. Stojmenovic, Home agent based location update and destination search schemes in ad hoc wireless networks, Computer science, SITE, University of Ottawa, TR-99-10, September 1999.

[TK] H. Takagi and L. Kleinrock, Optimal transmission ranges for randomly distributed packet radio terminals, IEEE Trans. on Communications, 32, 3, 1984, 246-257.

[WL] J. Wu and H. Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks, Proc. DIAL M, Seattle, Aug. 1999, 7-14.

[WS] S.C.M. Woo and S. Singh, Scalable routing in ad hoc networks, TR 00.001, March 2000, Dept. of Elec.& Comp. Eng., Oregon State University, Corvallis, OR, USA; Wireless Networks, 7, 5, September 2001, 513-529.

[WSCY] O. Wolfson, A.P. Sistla, S. Chamberlain and Y. Yesha, Updating and querying databases that track mobile units, Distributed and Parallel Databases Journal, 7, 3, 1999, 1-31.