Alcom-FT Technical Report Series
ALCOMFT-TR-03-164

On the Effect of User Mobility and Density on the Performance of Protocols for Ad-hoc Mobile Networks *

Ioannis Chatzigiannakis[†] Elena Kaltsa[‡] Sotiris Nikoletseas[†‡]

June 29, 2002

Abstract

In this paper we demonstrate the significant impact of (a) the mobility rate and (b) the user density on the performance of routing protocols in ad-hoc mobile networks.

In particular, we study the effect of these parameters on two different approaches for designing routing protocols: (a) the route creation and maintenance approach and (b) the "support" approach, that forces few hosts to move acting as "helpers" for message delivery. We study one representative protocol for each approach, i.e. AODV for the first approach and RUNNERS for the second.

We have implemented the two protocols and performed a large scale and detailed simulation study of their performance. The main findings are: the AODV protocol behaves well in networks of high user density and low mobility rate, while its performance drops for sparse networks of highly mobile users. On the other hand, the RUNNERS protocol seems to tolerate well (and in fact benefit from) high mobility rates and low densities.

1 Introduction and Previous Research

The problem of communication among remote mobile hosts is one of the most fundamental problems in ad-hoc mobile networks and is at the core of many algorithms, such as for counting the number of hosts, electing a leader, data processing etc. In ad-hoc mobile networks, a node MH_s that wants to communicate with MH_r might not be within communication range but could communicate if other hosts lying "in between" them are willing to forward packets for them.

The simplest way to establish communication in such networks is to perform flooding. By flooding we mean a distributed process of broadcasting a packet to all nodes within the transmission radius of the initiating sender and so on, until all nodes in the sender's "connected component" receive the packet. It is evident that such an approach will waste a large portion of the available bandwidth and thus is inefficient.

Towards avoiding flooding several protocols have been proposed in the related literature. A possible categorization of these protocols is based on the approach taken for establishing communication. One approach tries to construct and dynamically update paths between the mobile hosts. The second approach instead avoids path creation and maintenance by rather taking advantage of the hosts movement and accidental meetings in the network area.

*This work has been partially supported by the IST Programme of the European Union under contract numbers IST-1999-14186 (ALCOM-FT) and IST-2001-33116 (FLAGS).

[†]Computer Technology Institute, P.O. Box 1122, 26110 Patras, Greece, Emails: {ichatz,nikole}@cti.gr

[‡]Dept of Computer Engineering and Informatics, University of Patras, 26500 Patras, Greece. Email: kaltsa@ceid.upatras.gr

1.1 The Path Construction and Maintenance Approach

In an attempt to overcome the inefficiency of flooding, some algorithms have been presented that try to construct a connectivity related dynamic data structure (such as a dynamic-graph) that represents the topology of the network by a series of message passes. By taking advantage of such a data structure, messages can be transmitted over a limited number of intermediate hosts that interconnect MH_s with MH_r, also known as *routing paths* or *routes*. Indeed, this approach is common in ad-hoc mobile networks that either cover a relatively small space (i.e. the temporary network has a small diameter in terms of the transmission range), or are dense (i.e. comprised of thousands of wireless hosts). In such cases, since almost all "intermediate" locations are occupied by some hosts, communication is possible and broadcasting can be efficiently accomplished [1].

[2, 23] proposes the Dynamic Source Routing (DSR) protocol, which uses *on-demand route discovery*. There exist many variations of the DSR protocol that try to optimize the route discovery overhead. [24] presents the AODV (Ad Hoc On Demand Distance Vector routing) protocol that also uses a demand-driven route establishment procedure. More recently TORA (Temporally-Ordered Routing Algorithm, [19, 20]) is designed to minimize reaction to topological changes by localizing routing-related messages to a small set of nodes near the change. [12, 21] attempt to combine proactive and reactive approaches in the Zone Routing Protocol (ZRP), by initiating route discovery phase on-demand, but limit the scope of the proactive procedure only to the initiator's local neighborhood. [15] propose the Location-Aided Routing (LAR) protocol that uses a global positioning system to provide location information that is used to improve the performance of the protocol by limiting the search for a new route in a smaller "request zone".

1.2 The "Support" Approach

In contrast to all such methods, in [3, 7–9] a new framework is introduced for protocols designed for highly dynamic movement of the mobile users. The idea of the approach is to take advantage of the mobile hosts natural movement by exchanging information whenever mobile hosts meet incidentally. Specifically, the protocol framework proposes the idea of forcing a small subset of the deployed hosts to move as per the needs of the protocol. This set of hosts is called the "support" of the network (denoted by Σ). Assuming the availability of such hosts, the designer can use them suitably *by specifying their motion* in certain times that the algorithm dictates.

This approach may be useful in cases of rapid deployment of a number of mobile hosts, where it is possible to have a small team of fast moving and versatile vehicles, to implement the support. These vehicles can be cars, jeeps, motorcycles or helicopters. We interestingly note that this small team of fast moving vehicles can also be a collection of independently controlled mobile modules, i.e. robots. This specific approach is inspired by [26] that studies the problem of motion co-ordination in distributed systems consisting of such robots, which can connect, disconnect and move around. [26] deals with metamorphic systems where (as is also the case in our approach) all modules are identical. Note that our approach of having the support moving in a coordinated way, *i.e. as a chain of nodes*, has some similarities to [26].

Definition 1 *The subset of the mobile hosts of an ad-hoc mobile network whose motion is determined by a network protocol P is called the* support Σ *of $\mathcal{P}$. The part of P which indicates the way that members of Σ* move and communicate *is called the* support management subprotocol M_Σ *of $\mathcal{P}$.*

The scheme follows the general design principle of mobile networks (with a fixed subnetwork however) called the "two tier principle" [14], stated for mobile networks with a fixed subnetwork however, which says that any protocol should try to move communication and computation to the fixed part of the network. The assumed set of hosts that are coordinated by the protocol *simulates* such a (skeleton) network; the difference is that the simulation actually constructs a coordinated *moving* set.

In a recent paper ([16]), Q.Li and D.Rus present a model which has some similarities to our approach. The authors give an interesting, yet different, protocol to send messages, which forces *all the mobile hosts to slightly deviate (for a short period of time) from their predefined, deterministic routes, in order to propagate the messages.* Their protocol is, thus, *compulsory* for any host and it works only for deterministic host routes. Moreover, their protocol considers the propagation of only one message (end to end) each time, in order to be correct. In contrast, the support scheme allows for simultaneous processing of many communication pairs. In their setting, [16] shows optimality of message transmission times.

1.3 Our Contribution

The performance and correctness of protocols of the two approaches seems to depend heavily on the mobility rate of hosts as well as on the density of hosts in the network area.

In this work, we study the particular effect of these two parameters on the efficiency of representative protocols following the two approaches: the AODV protocol and the RUNNERS protocol.

We implement the two protocols and carry out a large scale simulation and detailed experimental evaluation of various important measures of their performance. We focus on the impact of two important network parameters, i.e. user density and mobility, on the performance measures studied. The detailed investigation basically highlights the advantages and disadvantages of each approach and its suitability for a certain network and user profile. In particular, our main findings are: (a) AODV behaves very well in dense networks, as shown by the high delivery rate and the small delay it achieves in this case. In sparse networks however, the RUNNERS protocol is more suitable, (b) AODV operates satisfactory in networks of low mobility rate of hosts, while RUNNERS performance is less dependent on mobility rate (in fact it seems to benefit from high mobility rates).

These findings validate experimentally, for the first time, our conjecture stated in [8] according to which in cases of high mobility rate and/or low density of user spreading, any algorithm that tries to maintain a global structure with respect to the temporary network will even become erroneous if the mobility rate is faster than the rate of updates of the algorithm.

We note the following important characteristics of our simulation experimental work: (a) we are able to measure very large sizes (e.g. network area size, number of users, number of messages generated, time period simulated etc.), (b) for the first time (to the best of our knowledge) AODV's performance is evaluated in a 3D network area setting. This shortage of 3D evaluation in relevant research applies also to other major protocols. Note that the 3D case is very practical and important, (c) we investigate the protocols' behavior for a plethora of network parameters (e.g. mobility rate, density of users, message generation rate, network are shape and size, etc.).

2 The Model of Motions

Throughout the paper, by *mobile hosts* (or mobile users) we shall refer to those hosts (or users) that do not belong to the support Σ. Let u be the number of mobile hosts and let the size of the support be $|\Sigma| = k$.

Our model has two parts: one part models the motion space (i.e. the network area), while the other part models the motions the mobile hosts perform in the motion space.

2.1 The Motion Space

To model the motion space, we map the three-dimensional space $\mathcal{S}$ (with possible obstacles) to a *motion graph* $\mathcal{G} = (V, E)$ (which will be defined shortly). Let $|V| = n$. The area where the stations move is abstracted by a graph by neglecting the detailed geometric characteristics of the motion. We first assume that each mobile host has a transmission range represented by a sphere tr having the mobile host at its center. This means that any other host inside tr can receive any message broadcast by this host. We approximate this sphere by a cube tc with volume $\mathcal{V}(tc)$, where $\mathcal{V}(tc) \simeq \mathcal{V}(tr)$. The size of tc can be chosen in such a way that its volume $\mathcal{V}(tc)$ is the maximum integral value that preserves $\mathcal{V}(tc) < \mathcal{V}(tr)$. To capture mobility and transmission range of hosts in $\mathcal{S}$, the space $\mathcal{S}$ is quantized into cubes of volume $\mathcal{V}(tc)$.

The *motion graph* $\mathcal{G} = (V, E)$, which corresponds to the above quantization of $\mathcal{S}$, is constructed in the following way. There is a vertex $v \in V$ for each cube of volume $\mathcal{V}(tc)$. Two vertices v and w are connected by an edge $(v, w) \in E$ iff the corresponding cubes are adjacent. Let $\mathcal{V}(S)$ be the volume of space $\mathcal{S}$. Clearly, $n = \mathcal{O}\left(\mathcal{V}(S)/\mathcal{V}(tc)\right)$ and consequently n is a rough approximation of the ratio $\mathcal{V}(S)/\mathcal{V}(tr)$, i.e. n measures area size in terms of transmission range.

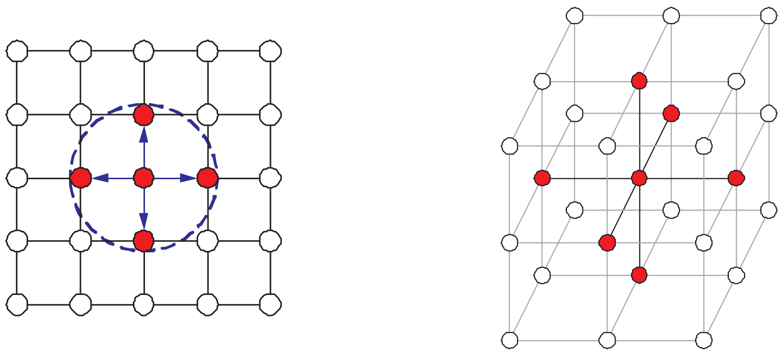


Figure 1: Transmission in 2D and 3D grid-motion graphs for $\beta = 2$

In this work we use a variation of this basic motion graph model and instead of dividing the original motion space $\mathcal{S}$ into consecutive cubes of volume $\mathcal{V}(tc)$, we introduce a variable $\beta \geq 1$ so that $\mathcal{S}$ is divided into consecutive cubes of volume $\mathcal{V}(tc)/\beta$. When $\beta = 1$ we get the basic model, however for $\beta > 1$, if a mobile host that resides on node u broadcasts a message, this message is received by any other host i that resides on node v_i such that $d(u, v_i) \leq \beta$. We do this generalization to allow path construction between hosts located in adjacent cubes. For a graphical representation of a transmission in the case when $\beta = 2$ in 2D and 3D grid motion graphs, see Fig. 1.

2.2 The Motion of the Mobile Hosts

In order to model the motion of the mobile hosts we use a *random motion model* which is a natural starting point aiming at studying the average-case efficiency of protocols for basic communication. To this end, the motion of a mobile host is modelled by concurrent and independent random walks. This type of motion was used in [5, 6], assumes only local knowledge of $\mathcal{G}$ and hosts move using at most one edge in a single step. We note here that movement following independent random walks approximates quite accurately dense motions of many users, since it captures the characteristics of motions. In fact, the assumption that the mobile hosts move randomly, either according to uniformly distributed changes in their directions and velocities, or according to the random waypoint mobility model by picking random destinations, has also been used in previous research (see e.g., [11, 13]).

We assume that the random motion is performed *independently* of the motion of the support, i.e., we exclude the case where some of the mobile hosts are deliberately trying to avoid the members of the support during the whole execution of the protocol. This is a pragmatic assumption usually followed by application protocols.

3 The Protocols

3.1 The Runners Support Routing Protocol (RUNNERS)

The runners support routing protocol is presented in [5, 8] and the main idea is as follows. First, we assume that there is a *setup phase* in the network, during which the support Σ of the network is formed. This can be done either by a randomized process, that randomly selects a number of mobile users or alternatively, the implementor may provide a specific number of mobile hosts (with specialized specifications, such as fast moving and versatile vehicles) that will form the support Σ.

The idea is to force this small team of $k = |\Sigma|$ mobile hosts to move fast enough so that they cover (in sufficiently short time) the entire network area (the move of the rest is not affected by the protocol). In particular, support members move performing concurrent and independent random walks in the network area. Each mobile host that wants to send some information to another host sends its message to any member σ of Σ whenever σ comes within its transmission range. In addition, during this communication, it can receive any messages that have designated it as the receiver. Hence, the support serves as a set of mobile *relay* hosts, that temporarily buffer messages until they get delivered. The expectation is that as this small team of mobile hosts moves randomly around the area of the ad-hoc network, it will cover the entire area in sufficiently short time, thus servicing effectively all messages.

When two runners meet, they exchange any information given to them by senders encountered using a *synchronization subprotocol*, a distributed algorithm that is partially based on the *two-phase commit* algorithm as presented in details in [17].

The runners protocol is presented also in [4] in a more formal and complete way. In that paper mechanisms to reduce the memory (buffer) requirements for the support's members are also discussed.

3.2 Ad-Hoc On-Demand Distance Vector Routing (AODV)

The Ad-Hoc On-Demand Distance Vector (AODV) routing protocol described in [24] builds on the DSDV algorithm ([22, 23]). AODV is an improvement on DSDV because it typically minimizes the number of required broadcasts by creating routes on a demand basis, as opposed to maintaining a complete list of routes as in the DSDV algorithm. AODV is a pure on-demand route acquisition system, since nodes that are not on a selected path do not maintain routing information or participate in routing table exchanges ([24]).

When a source node desires to send a message to some destination node and does not already have a valid route to that destination, it initiates a path discovery process to locate the other node. It broadcasts a route request (`RREQ`) packet to its neighbors, which then forward the request to their neighbors, and so on, until either the destination or an intermediate node with a "fresh enough" route to the destination is located. AODV utilizes destination sequence numbers to ensure that all routes are loop-free and contain the most recent route information. Each node maintains its own sequence number, as well as a broadcast `ID`. The broadcast `ID` is incremented for every `RREQ` the node initiates, and together with the node's `IP` address, uniquely identifies a `RREQ`. Along with its own sequence number and the broadcast `ID`, the source node includes in the `RREQ` the most recent sequence number it has for the destination. Intermediate nodes can reply to the `RREQ` only if they have a route to the destination whose corresponding destination sequence number is greater than or equal to that contained in the RREQ.

During the process of forwarding the RREQ, intermediate nodes record in their route tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path. If additional copies of the same RREQ are later received, these packets are discarded. Once the RREQ reaches the destination or an intermediate node with a fresh enough route, the destination/intermediate node responds by unicasting a route reply (RREP) packet back to the neighbor from which it first received the RREQ. As the RREP is routed back along the reverse path, nodes along this path set up forward route entries in their route tables which point to the node from which the RREP came. These forward route entries indicate the active forward route. Associated with each route entry is a route timer which will cause the deletion of the entry if it is not used within the specified lifetime. Because the RREP is forwarded along the path established by the RREQ, AODV only supports the use of symmetric links.

Routes are maintained as follows. If a source node moves, it is able to reinitiate the route discovery protocol to find a new route to the destination. If a node along the route moves, its upstream neighbor notices the move and propagates a link failure notification message (an RREP with infinite metric) to each of its active upstream neighbors to inform them of the erasure of that part of the route [23]. These nodes in turn propagate the link failure notification to their upstream neighbors, and so on until the source node is reached. The source node may then choose to reinitiate route discovery for that destination if a route is still desired.

4 Implementation Details

All of our implementations follow closely the protocols described above. They have been implemented as C++ classes using several advanced data types and structures of LEDA [18]. We used the simulation environment developed in [5, 6] (*adhocsim*) where each class is installed in an environment that allows to read graphs from files as well as to perform a network simulation for a given number of rounds, a fixed number of mobile hosts and certain communication and mobility patterns. After the execution of the simulation, the environment stores the results again on files so that the measurements can be represented in a graphical way.

Note that in this work, we are not interested in examining the behavior of protocols in the presence of failures of hosts or message losses due to communication failures, thus we have not implemented any special code to handle node failures and restarts.

Implementation details for AODV. The protocol defines a set of parameters that need to be fine tuned in order to maximize its performance. More specifically, the lifetime of a route constructed by the protocols as well as the time periods that the protocol will wait until a route is discovered are *closely related to the diameter of the network* ([25]). In order to set the proper values to these parameters we measured the diameter of the motion graph used in each experiment. Recall that the time is measured in simulation rounds.

Note that in table 1 the RREQ_HOPS value is extracted from the RREQ packet received and signifies the number of hops that the packet has traversed in order to reach the node.

Finally, to implement the increment and update of the *Sequence Numbers* (SN), that are used to avoid the formation of cycles in the routing paths and is therefore *crucial for the correct operation of the protocol*, we have followed very closely the rules defined in [25].

Implementation details for RUNNERS. A basic parameter that affects the performance of the RUNNERS protocol is the size of the support (k). The proper selection of a value for k is studied

Parameter	Value
NET_DIAMETER	$\mathrm{diam}(\mathcal{G})$
NET_TRAVERSAL_TIME	NET_DIAMETER
PATH_DISCOVERY_TIME	$2 \cdot$ NET_TRAVERSAL_TIME
ACTIVE_ROUTE_TIMEOUT	$2 \cdot$ NET_TRAVERSAL_TIME
MY_ROUTE_TIMEOUT	$2 \cdot$ ACTIVE_ROUTE_TIMEOUT
TIME_TO_RETRY	PATH_DISCOVERY_TIME
MAX_TRIES	3
REVERSE_TIMEOUT	$\max\{\texttt{ROUTE\_LIFETIME}, 2 \cdot \texttt{NET\_TRAVERSAL\_TIME} - 2 \cdot \texttt{RREQ\_HOPS}\}$

Table 1: AODV Parameters

Graph Type	Graph Size	Support Size (k)
2D grid	121	10
2D grid	529	15
2D grid	1024	20
3D grid	125	10
3D grid	512	15
3D grid	1000	20

Table 2: RUNNERS Support size parameter

thoroughly in [5]. Based on that work, the values that we selected for the size of the support are listed in table 2.

In addition the mobile hosts that belong to the support have motion rate $\mu = 1$, i.e. they move continuously in the area covered by the network.

5 Efficiency Measures

In order to evaluate the performance of the above protocols we investigate a set of efficiency measures. We start with the message delivery rate and the message delay achieved by a protocol.

Definition 2 Let M_{total} be the total number of messages send out and M_{succ} the number of messages that were successfully delivered to their destination. Let $\mathcal{D} = \frac{M_{succ}}{M_{total}}$ be the *success rate* (or delivery rate) of a communication protocol.

Definition 3 Let $\mathcal{T}$ be the *total time* (or message delay) required for a *data message* to reach its final destination.

Regarding the AODV protocol, we define two additional efficiency measures in order to evaluate the ability of the protocol to construct and maintain routing paths. Since the RUNNERS protocol is not based on the *path construction and maintenance approach*, these two metrics are not applicable.

Definition 4 Let $\mathcal{H}$ be the *number of "hops"* required to propagate a *data message*, i.e. the length of the path of intermediate hosts.

Definition 5 Let $M_{control}$ be the total number of control messages transmitted by the protocol. Let $\mathcal{C} = \frac{M_{control}}{M_{total}}$ be the ratio of control messages transmitted per message that needs to be transmitted.

6 Experimental Evaluation

6.1 Experimental Setup

A large number of experiments were carried out modelling the different possible situations regarding the geographic area covered by an ad-hoc mobile network and the motion of hosts within it. Each experiment is characterized by a tuple of parameters $\langle \mathcal{P}, k, u, \mu, \lambda, \mathcal{G}, n \rangle$, where $\mathcal{P}$ is the protocol used, $k = |\Sigma|$ (applicable only in the $\mathsf{RUNNERS}$ case), u is the total number of mobile hosts h ($h \notin \Sigma$, in the $\mathsf{RUNNERS}$ case), μ is their rate of movement, λ is the message generation rate by each host per round, $\mathcal{G} = (V, E)$ is the motion graph, and $n = |V|$.

Each experiment proceeds in lockstep rounds called *simulation rounds* (i.e. we measure simulation time in rounds). Each mobile host h is considered identical in computing and communication capability and resides initially in a particular vertex v_h of the motion graph $\mathcal{G}$. Two mobile hosts i, j communicate by sending messages if they reside on the same vertex of the motion graph (i.e. $v_i = v_j$) or when they are within transmission range of each other (i.e. $d(v_i = v_j) \leq \beta$). We assume that in this case message exchange takes negligible time (i.e., the messages are short packets and the wireless transmission is very fast).

In a simulation round each mobile host h may move to a vertex of the motion graph $\mathcal{G}$ and performs some computation. We assume that a mobile host can receive several messages and send several messages at the same round. If $h \in \Sigma$, its move and local computation are determined by the protocols described in Section 3. If $h \notin \Sigma$, then with probability μ it moves to a vertex of the motion graph $\mathcal{G}$ (e.g., if $\mu = 0.05$ it moves roughly every 20 rounds) and with probability λ the host h generates a new message (e.g., if $\lambda = 0.01$ a new message is generated roughly every 100 rounds) by picking a random destination host.

For each motion graph $\mathcal{G}$ considered, we used various numbers of mobile hosts u, namely $u \in [10, 100]$, which were injected at random positions of $\mathcal{G}$ and generated message exchanges with various rates $\lambda \in [0.05, 1.0]$. The selection of this range for λ is based on the assumption that the mobile hosts do not execute a real-time application that requires continuous exchange of messages, but can tolerate delays in the order of fractions of seconds to several seconds. Our focus is on applications that are asynchronous in nature so that they can tolerate reasonable end-to-end delays. Examples of such applications include electronic mail, database synchronization between a mobile terminal and a central database, and certain types of event notification (similar assumptions are made in [10]).

Finally, regarding motion graphs, we considered 2D and 3D grid graphs to model different geographic areas. The former case is the simplest pattern of motion one can consider (e.g. mobile hosts that move on a plane surface). We used three different $\sqrt{n} \times \sqrt{n}$ grid graphs with $n \in \{121, 529, 1024\}$. The latter case helps in evaluating the performance of the protocols when the mobile hosts move in the 3D space. We considered three different 3D grid graphs ($n^{1/3} \times n^{1/3} \times n^{1/3}$) with $n \in \{125, 512, 1000\}$.

6.2 Experimental Results and Discussion

We carried out each experiment for a very large number of rounds (at least $10,000$) during which a very large number of messages (at least $200,000$) were delivered to the designated receivers.

We start our experimentation by considering the effect of the number of users that participate in the network on the performance of the two protocols. The results of the experiments reported are shown in Figures 2, 3, 4 and 5. In these graphs, we have included the results for 2D grid motion graphs of size $n = 121$ (similar results hold for other motion graphs considered). For this set of experiments we have used a fixed rate of motion ($\mu = 0.01$) and a fixed message generation rate ($\lambda = 0.02$). Note that since the area of the network remains fixed (i.e. the size of the motion graph $\mathcal{G}$ is fixed) increasing the number of users essentially increases the density of the users in the network.

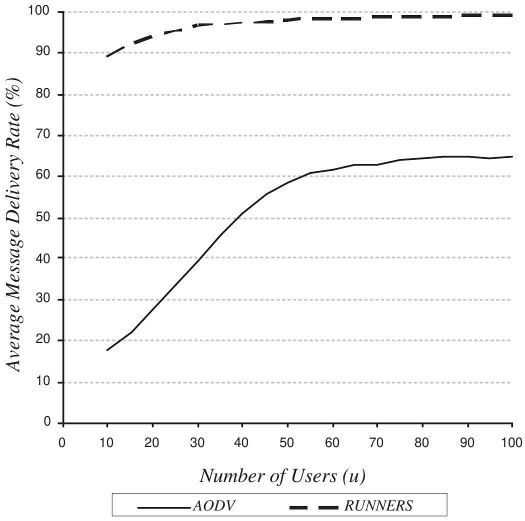


Figure 2: Average Message Delivery Rate ($\mathcal{D}$) for 2D grid graphs ($n = 121$, $\mu = 0.01$, $\lambda = 0.02$, $u \in [5, 100]$).

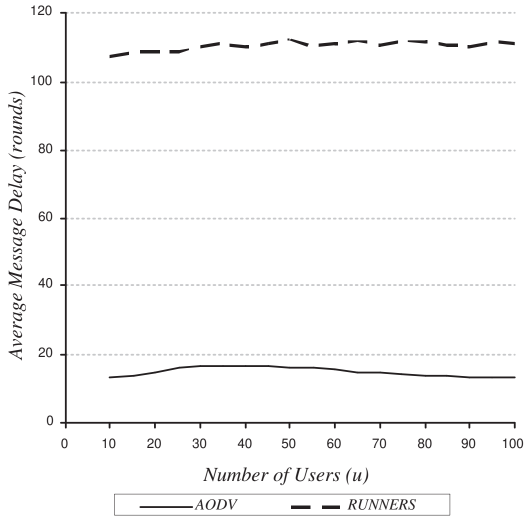


Figure 3: Average Message Delay ($\mathcal{T}$) for 2D grid graphs ($n = 121$, $\mu = 0.01$, $\lambda = 0.02$, $u \in [5, 100]$).

In fig. 2 we observe that the number of users has significant impact on the performance of AODV regarding the delivery rate. Actually, for low user population the delivery rate is very low, while increases fast with u until a certain number of users is reached ($u = 45$) when no further improvement is observed. On the other hand, although the RUNNERS protocols is also affected by the number of users, it is on a much smaller scale. In all cases considered, the RUNNERS protocol achieves a very high delivery rate compared to AODV.

Figure 3 depicts the effect of the number of users on the average message delay. First of all we observe that AODV is much more efficient (regarding this metric) than the RUNNERS protocol, while both protocols' efficiency seems to be unaffected by the density of the users. This however should be evaluated in combination with the results in fig. 2, since AODV exhibits low delivery rates for low densities. We here measure delays for delivered messages only.

Figures 4, 5 show the average number of hops and the ratio of control messages (per data message) sent by AODV. Recall that these two metrics are not applicable for the RUNNERS protocol since no control messages are used and no paths are constructed. The experiments show that when the network is comprised of few mobile hosts, the average number of hops is about 2, implying that most of the times the routing pats are very short and data messages are delivered when the sender and receiver are within each others transmission range. Again note that we here measure only over messages that manage to reach their destination. As said earlier as well, many messages are not making it to their destination in AODV for low user densities. As the number of users increases, the routing paths become longer reaching a maximum value of 4.

However, in the effort to construct these paths, AODV uses a large number of control messages. In fact, the experiments suggest that the ratio of control messages per data message sent is linear to the number of users that participate in the network. This implies a significant protocol overhead in AODV, especially for large user populations.

A different way to evaluate the performance of the network under different densities of users

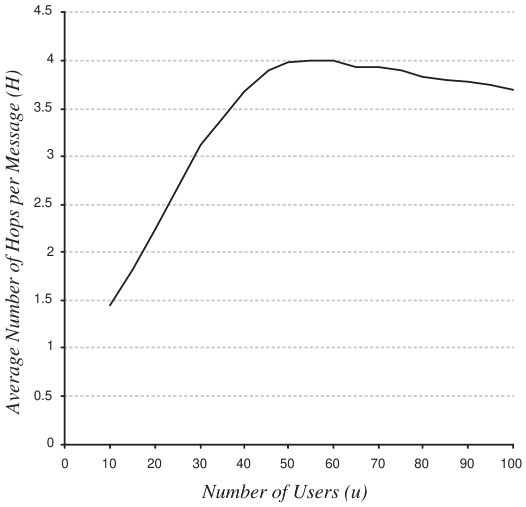


Figure 4: Average number of hops ($\mathcal{H}$) per Message for 2D grid graphs ($n = 121$, $\mu = 0.01$, $\lambda = 0.02$, $u \in [5, 100]$).

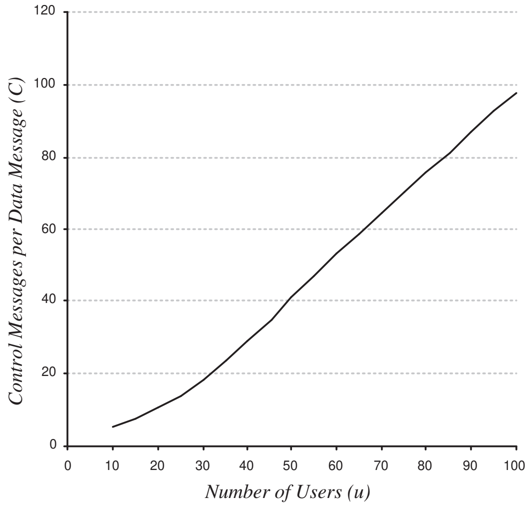


Figure 5: Control Messages per Data Message ($\mathcal{C}$) for 2D grid graphs ($n = 121$, $\mu = 0.01$, $\lambda = 0.02$, $u \in [5, 100]$)

is to keep the number of users fixed (i.e. $u = 50$) and vary the size of the motion graph used (i.e. variate n). Note that protocols' performance might depend on both number of users and area size separately (not only on their ratio, i.e. the density). This means that two networks of same user density should be evaluated separately for the two parameters above. Based on this approach, we conducted an additional number of experiments only for the AODV protocol in three motion graphs of different sizes (see fig. 6 and 7). Remark that this kind of experiments that investigate the impact of the size of the motion graph on the RUNNERS protocol are reported in [5].

The results of these "reverse" experiments behave in the same way as their "original" version; that is, as the size of the motion graph increases (i.e. when the user density decreases) the performance of the network drops, both in terms of delivery rate and average message delay.

In addition, these experiments indicate that modifying the message generation rate (λ) does not affect the performance of the protocol. This is based on the higher-level application type we have chosen to simulate, which picks a random destination for each new message generated. The selection of this value for λ is based on the assumption that the mobile hosts do not execute a real-time application that requires continuous exchange of messages, but can tolerate delays in the order of fractions of seconds to several seconds. Our focus is on applications that are asynchronous in nature so that they can tolerate reasonable end-to-end delays. Examples of such applications include electronic mail, database synchronization between a mobile terminal and a central database, and certain types of event notification (similar assumptions are made in [10]).

Based on the above observations, it is evident that in dense ad-hoc networks, the path construction approach of the AODV protocol works very well by achieving high delivery rates (i.e. the majority of the messages are delivered to their final destination) while the average message delay is low. On the other hand, the performance of AODV drops when the user density decreases. In fact, in sparse networks, broadcasting may become even impractical, as two distant peers may even not be reached by any broadcast since hosts may not occupy all "intermediate" locations. Even if a (sufficiently long, thus not efficient) communication path is established, single link failures happening when a small number of hosts that were part of the communication path move in a way such

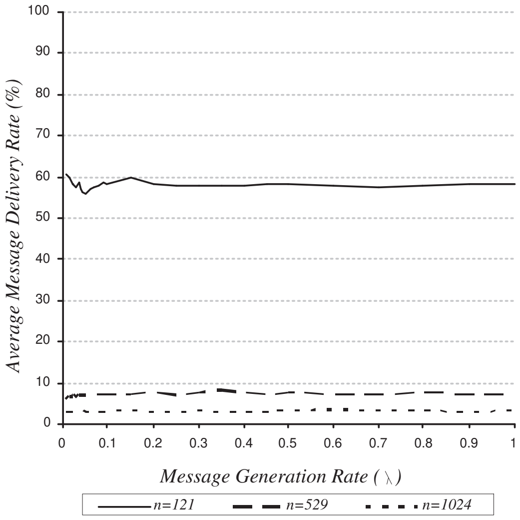


Figure 6: Average Message Delivery Rate ($\mathcal{D}$) for 2D grid graphs ($n \in \{121, 529, 1024\}$, $\mu = 0.01$, $\lambda \in [0.01, 1]$, $u = 50$).

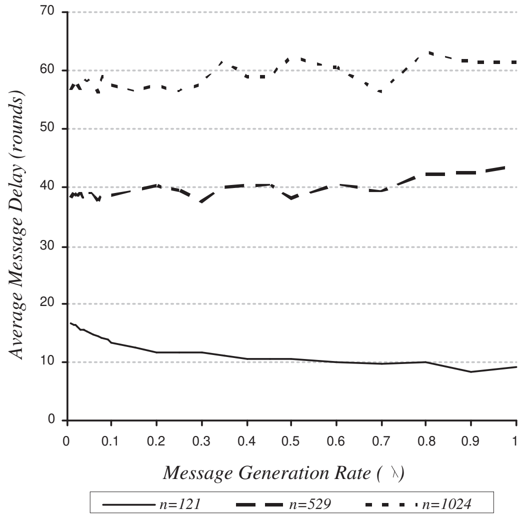


Figure 7: Average Message Delay ($\mathcal{T}$) for 2D grid graphs ($n \in \{121, 529, 1024\}$, $\mu = 0.01$, $\lambda \in [0.01, 1]$, $u = 50$).

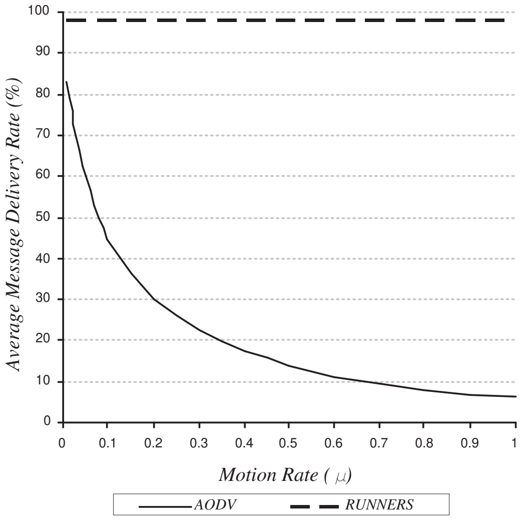
that they are no longer within the transmission range of each other, will make this path invalid. This is reflected in fig. 4 where the average path length is less than 2, i.e. most of the times the messages are transmitted directly to the receiver host.

This observation leads to the conclusion that in sparse networks (i.e. with small number of users relatively to the area covered by the network) where the network tends to be disconnected, the RUNNERS protocol is more suitable than AODV. Thus, even if the users are spread in remote areas and they do not move beyond these areas, it is now possible for information to reach them, as the RUNNERS protocol takes special care of such situations.

In the second set of experiments we investigate the effect of user mobility on the performance of the two protocols. Essentially we vary the motion rate ($\mu \in [0.01, 1]$) of the mobile hosts and observe how it affects the efficiency of the protocols. Remark that the results reported in Figures 8, 9, 10 and 11 are based on a 3D grid motion graphs of size $n = 125$ (similar results hold for other motion graphs considered) with 50 mobile hosts (i.e. $u = 50$) and a fixed message generation rate ($\lambda = 0.02$).

Regarding the average delivery rate (see fig. 8) we observe that the motion rate of the users has a great impact on the performance of AODV while it seems to have no effect on the RUNNERS protocol. Actually, even for low rates of motion (e.g. $\mu = 0.2$) AODV manages to deliver only the 30% of the messages while for higher rates (e.g. $\mu \geq 0.6$) the delivery rate drops below 10%. On the other hand, the RUNNERS protocol achieves a very high delivery rate for all “frequencies” of motion.

Figure 9 depicts the effect of the motion rate on the average message delay. Again, the performance of the AODV protocol is degrading as the rate of motion increases, i.e. the average message delay increases as the users of network move more frequently. Interestingly enough, the RUNNERS protocol has a completely different behavior; that is, the average message delay drops as the rate of motion of the users increases. This can be explained by the fact that as the users are moving more frequently they will meet the nodes of the support more often. In this way the messages will be delivered to the support and to their final destination faster. Thus, the RUNNERS protocol is



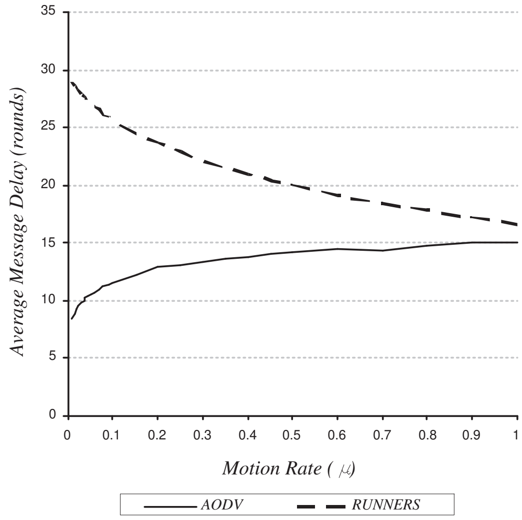
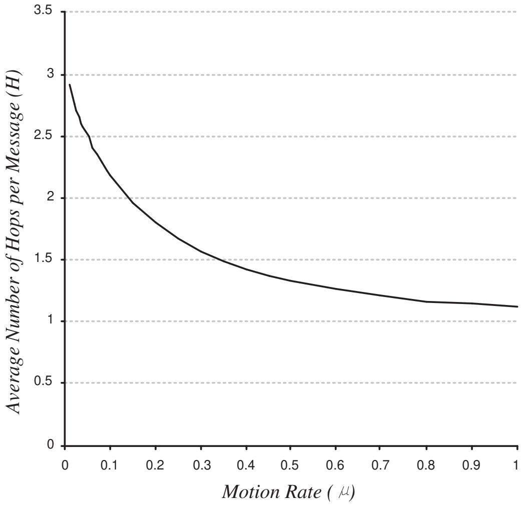


Figure 8: Average Message Delivery Rate ($\mathcal{D}$) for 3D grid graphs ($n = 125$, $\mu \in [0.01, 1]$, $\lambda = 0.02$, $u = 50$).

Figure 9: Average Message Delay ($\mathcal{T}$) for 3D grid graphs ($n = 125$, $\mu \in [0.01, 1]$, $\lambda = 0.02$, $u = 50$).



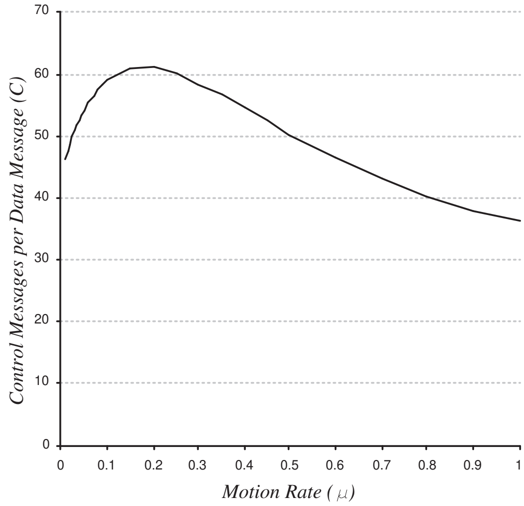


Figure 10: Average number of hops ($\mathcal{H}$) per Message for 3D grid graphs ($n = 125$, $\mu \in [0.01, 1]$, $\lambda = 0.02$, $u = 50$).

Figure 11: Control Messages per Data Message ($\mathcal{C}$) for 3D grid graphs ($n = 125$, $\mu \in [0.01, 1]$, $\lambda = 0.02$, $u = 50$)

particularly suitable in the case of high mobility rate, since increasing motion seems to positively affect its performance by accelerating meeting times.

It is evident that if all mobile hosts are static (for sufficiently long periods, e.g. $\mu = 0.01$) or quasi-static (they move very slowly, e.g. $\mu < 0.1$), establishing communication with the AODV protocol is efficient as long as the sender and the receiver are within the same connected component. In such a setting the protocol achieves a high success rate (e.g. $\mathcal{D} > 60\%$) and low message delays (e.g. $\mathcal{T} < 10$ rounds). However in the non-trivial and practical case in which the mobile hosts move and change their positions rapidly, some links will "fail" in the sense that some nodes will get "out of range" while it is possible for some other links to "form" as nodes will get "within range". Thus, the AODV protocol is overwhelmed by link failures/formations that are too frequent or too fast.

As shown in fig. 8, when the *rate* of topological changes is very high (i.e. frequent motion of the users) the AODV protocol fails to react fast enough and the messages are delivered only (in the rare case) when the sender meets the receiver (i.e. in a single hop transmission).

7 Closing Remarks

Our large scale simulation experimental study shows the significant impact of two very characteristic network parameters (that of mobility rate and user density) on protocols' performance.

Path maintenance algorithms seem to behave well in low mobility, dense networks, while their performance degrades in the case of highly mobile users and sparse networks. On the other hand, the semi-compulsory support approach seems to behave well in the case of high mobility rate (and always succeeds in delivering messages, even in low density networks).

Our findings suggest that protocol design should carefully take into consideration such vital parameters. It seems that (at least under the model we study) there may be no protocol that is suitable for all network cases. Thus, the network operator should in each case decide on which protocol/approach should be used.

References

[1] M. Adler and C. Scheideler, *Efficient communication strategies for ad-hoc wireless networks*, 10th Annual Symposium on Parallel Algorithms and Architectures, ACM, 1998, pp. 259–268.

[2] J. Broch, D.B. Johnson, and D.A. Maltz, *The dynamic source routing protocol for mobile ad-hoc networks*, Tech. report, IETF, Internet Draft, December 1998, draft-ietf-manet-dsr-01.txt.

[3] I. Chatzigiannakis, *Design and analysis of distributed algorithms for basic communication in ad-hoc mobile networks*, Ph.D. dissertation, Dept. of Computer Engineering and Informatics, University of Patras, Greece, May 2003.

[4] I. Chatzigiannakis and S. Nikoletseas, *An adaptive compulsory protocol for basic communication in highly changing ad-hoc mobile networks*, 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, IEEE/ACM, 2002, IPDPS Workshops.

[5] I. Chatzigiannakis, S. Nikoletseas, N. Paspalis, P. Spirakis, and C. Zaroliagis, *An experimental study of basic communication protocols in ad-hoc mobile networks*, 5th International Workshop on Algorithmic Engineering (WAE 2001), Springer-Verlag, 2001, Lecture Notes in Computer Science, LNCS 2141, pp. 159–171.

[6] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *Analysis and experimental evaluation of an innovative and efficient routing approach for ad-hoc mobile networks*, 4th International Workshop on Algorithmic Engineering (WAE 2000), Springer-Verlag, 2000, Lecture Notes in Computer Science, LNCS 1982, pp. 99–110.

[7] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *An efficient communication strategy for ad-hoc mobile networks*, 15th International Workshop on Distributed Algorithms (DISC 2001), Springer-Verlag, 2001, Lecture Notes in Computer Science, LNCS 2180, pp. 285–299.

[8] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *On the average and worst-case efficiency of some new distributed communication and control algorithms for ad-hoc mobile networks*, 1st International Annual Workshop on Principles of Mobile Computing (POMC 2001), ACM, 2001, pp. 1–19.

[9] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *Distributed communication algorithms for ad hoc mobile networks*, Journal of Parallel and Distributed Computing **63** (2003), no. 1, 58–74.

[10] M. Grossglauser and D. Tse, *Mobility increases the capacity of wireless adhoc networks*, 20th Annual International Conference on Computer Communications and Networking, IEEE, 2001, pp. 289–299.

[11] Z.J. Haas and M.R. Pearlman, *The performance of a new routing protocol for the reconfigurable wireless networks*, International Conference on Communications, IEEE, 1998.

[12] Z.J. Haas and M.R. Pearlman, *The zone routing protocol (ZRP) for ad-hoc networks*, Tech. report, IETF, Internet Draft, June 1999, draft-zone-routing-protocol-02.txt.

[13] G. Holland and N. Vaidya, *Analysis of TCP performance over mobile ad hoc networks (MOBICOM 1999)*, 5th Annual International Conference on Mobile Computing, IEEE/ACM, 1999, pp. 219–230.

[14] T. Imielinski and H.F. Korth, *Mobile computing*, Kluwer Academic Publishers, 1996.

[15] Y. Ko and N.H. Vaidya, *Location-aided routing (LAR) in mobile ad-hoc networks*, 4th Annual International Conference on Mobile Computing (MOBICOM 1998), IEEE/ACM, 1998, pp. 66–75.

[16] Q. Li and D. Rus, *Sending messages to mobile users in disconnected ad-hoc wireless networks*, 6th Annual International Conference on Mobile Computing (MOBICOM 2000), IEEE/ACM, 2000, pp. 44–55.

[17] N.A. Lynch, *Distributed algorithms*, Morgan Kaufmann Publishers Inc., 1996.

[18] K. Mehlhorn and S. Näher, *LEDA: A platform for combinatorial and geometric computing*, Cambridge University Press, 1999.

[19] V. Park and M. Corson, *A highly adaptive distributed routing algorithm for mobile wireless networks*, 16th Annual International Conference on Computer Communications and Networking, IEEE, 1997, pp. 1405–1413.

[20] V.D. Park and M.S. Corson, *Temporally-ordered routing algorithms (TORA) version 1 functional specification*, Tech. report, IETF, Internet Draft, October 1999, draft-ietf-manet-tora-spec-02.txt.

[21] M. Pearlman and Z. Haas, *Determining the optimal configuration for the zone routing protocol*, Journal on Selected Areas in Communications **17** (2003), no. 8, 1395–1414.

[22] C. Perkins, *Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers*, 18th Conference on Communications Architectures, Protocols and Applications (SIGCOMM 1994), ACM, 1994, pp. 234–244.

[23] C.E. Perkins, *Ad hoc networking*, Addison-Wesley, Boston, 2001.

[24] C.E. Perkins and E.M. Royer, *Ad-hoc on demand distance vector (AODV) routing*, 2nd Annual Workshop on Mobile Computing Systems and Applications, IEEE, 1999, pp. 90–100.

[25] C.E. Perkins, E.M. Royer, and S.R. Das, *Ad-hoc on demand distance vector (AODV) routing*, Tech. report, IETF, Internet Draft, June 2002, draft-ietf-manet-aodv-11.txt.

[26] J.E. Walter, J.L. Welch, and N.M. Amato, *Distributed reconfiguration of metamorphic robot chains*, 19th ACM Annual Symposium on Principles of Distributed Computing (PODC 2000), ACM, 2000, pp. 171–180.