# A NEW ROUTE DISCOVERY ALGORITHM FOR MANETS WITH CHASE PACKETS

MZNAH A. AL-RODHAAN, LEWIS MACKENZIE, MOHAMED OULD-KHAOUA

*Department of Computing Science*
*University of Glasgow*
*Glasgow, UK G128QQ*
*Email: {rodhaan, lewis, mohamed}@dcs.gla.ac.uk*

**Abstract:** We introduce a new dual-tier route discovery algorithm for MANETs. The new algorithm uses chase packets to stop the fulfilled route requests in on-demand routing protocols. The algorithm works by including the most likely destinations for a given source node in a local neighbourhood and broadcast route requests with full speed within this region. However, outside such neighbourhood the route request has a very high chance of being fulfilled, so propagation of route requests in this region is deliberately delayed to provide the chase packets the opportunity to catch the route request and minimise network congestion without affecting the discovery process. The algorithm is adaptive and continuously updates the boundary of each source node's neighbourhood to optimize performance. Furthermore, we provide a detailed performance evaluation using mathematical and simulation modelling and compare our algorithm with existing algorithms, to demonstrate that it does indeed improve the average route request time required.

*Keywords*: MANETs, Routing protocols, Route Discovery, Chase Packets, Performance analysis.

## 1. INTRODUCTION

Traditional wired networks have been the main focus of research. However, they are useful but not suitable for mobile situations. When mobile devices such as notebooks and PDAs appeared, users wanted wireless connectivity, and this duly become a reality. Wireless networks may be infrastructure-oriented, such as the access point dependent networks [Murthy and B., 2004] or infrastructure-less such as Mobile Ad hoc Networks (MANETs) [Chlamtac, Conti, 2003; Murthy and B., 2004; Tanenbaum, 2003]. Some of the dominant initial motivations for MANET technology came from military applications in environments where there is no infrastructure. However, while such applications remain important, MANET research has diversified into areas such as disaster relief, sensors networks, and personal area networks [Tanenbaum, 2003].

The design of an efficient and reliable routing strategy is a very challenging problem due to the limited resources in MANETs [Murthy and B., 2004]. Many multi-hop routing protocols have been proposed and investigated in the literature [Adjih, T. Clausen,2003; Haas, Pearlman, 2002; Johnson, D. Maltz, 2003; Perkins, 2001; Perkins, E. Belding-Royer, 2003; Young-Bae and Nitin, 2000], and can be divided into three categories: *proactive*, *reactive*, and *hybrid* [Abolhasan, Wysocki, 2004]. In a proactive routing protocol (table-driven), the routes to all the destinations (or parts of the network) are determined statically at the start up, and maintained by using a periodic route update process, An example of this class of routing protocols is the Optimized Link State Routing Protocol (OLSR) [Adjih, T. Clausen, 2003]. However, in a reactive routing protocol (on-demand), routes are determined dynamically when they are required by the source using a route discovery process. Its routing overhead is lower than the proactive routing protocols if the network size is relatively small [Das, Castaneda, 1998]. When a source needs to send messages to a destination in on-demand routing, it initiates a broadcast-based route discovery process to look for one or more possible paths to the destination. Examples from this class are the Dynamic Source Routing (DSR) [Johnson, D. Maltz, 2003] and Ad Hoc On Demand Distance Vector (AODV) [Perkins, 2001]. Finally, a hybrid routing protocol combines the basic properties of the first two classes of protocols. That is, they are both reactive and proactive in nature. The Zone Routing Protocol (ZRP) [Haas, Pearlman, 2002] is an example belonging to this class.

### 1.1. Route Discovery Algorithms

On-demand routing is known to use low bandwidth and low power consumption, since the nodes have no periodical tasks, which made it very appealing for MANETs scenarios [Abolhasan, Wysocki, 2004]. On-demand routing algorithms search for the desired route only when needed, as the name implies. When a source node needs to send packets to a destination, it initiates a route discovery process to look for a route, or several routes, to that destination using broadcasting techniques. After discovering the needed route(s), the source will start transmitting data packets using the route(s) discovered.

In MANETS, broadcasting is an essential part of on-demand routing protocols. It is used to discover a route or multiple routes. For example, DSR [Perkins, 2001] and AODV [Perkins, 2001] both use broadcasting. These protocols depend on a simple flooding as a form of broadcasting, where each node may receive multiple copies of a unique route request packet and retransmit it exactly once. Unfortunately, the flooding leads to a reproduction of the packet causing redundancy that will congest the network and increase the chances of collision: these combined are known as the *broadcast storm problem* [Yu-Chee, Sze-Yao, 2002]. Moreover, flooding consumes a lot of node resources such as bandwidth and power. These problems can be reduced to control the flooding by stopping the route request as soon as possible upon the discovery of the needed route.

In on-demand routing protocols, when a source node needs a route to a particular destination it first searches in its cached routes where any seen or overheard route is stored for future use; if this is unsuccessful, it starts a route discovery process whereby a route request packet is broadcasted from a node to its neighbours until it arrives at the destination or an intermediate relay node that has a route to the destination while other nodes broadcast it until the time to live (TTL) field reaches zero.

The route discovery process often floods the network with route request packets looking for routes throughout the network. Unfortunately, the route request will keep spreading even after a route has been found and that will congest the network and waste its resources. The route discovery protocols can be optimised by minimizing such overhead and reducing or stopping the unnecessary propagation of route request packets after the route has been discovered.

The rest of the paper is organised as follows: Section 2 presents the related work in the literature while section 3 discusses the limitation of existing techniques, Section 4 introduces some of the notation and definitions while section 5 presents our newly proposed algorithm; then in section 6 evaluates the performance and conducts a comparative study demonstrating the superiority of our algorithm in terms of reducing network latency over others in the literature for example [Gargano, Hammar, 2004] and [Park, Kim, 2006]; section 7 describe the simulation environment and observation. Finally, Section 8 concludes this study.

## 2. RELATED WORK

An algorithm for route discovery optimization that eliminates the need for historical or location information has been proposed in [Gargano, Hammar, 2004]. It achieves this by employing chase packets to stop the propagation of the fulfilled route requests. The algorithm works by broadcasting a route request using only ¼ of the channel time while the rest of the channel time is dedicated to transmit the route reply and broadcast the chase packet after finding the route as illustrated in Figure 1. The main purpose of broadcasting the chase packet is basically to stop the route request packet from further propagation after finding the needed route. The main deficiency of this algorithm is that it favours the chase over the route requests. For instance, as depicted in Figure 1, if there is a route request between any two nodes it will be sent with a lower speed due to the fact that it uses ¼ of the channel speed. Doing so would delay the route discovery for the source node in hand as well as the other source nodes that might be trying to route discover yet getting low priority due to the chase messages sent by other nodes.

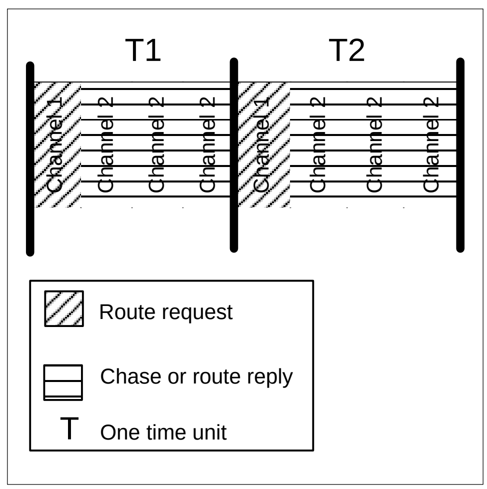


Figure 1. Time division between chase massage, route reply and route request.

Blocking-ERS [Park, Kim, 2006] is another algorithm that aims to improve the energy consumption. This algorithm uses chase packets to optimize the route request process. This approach utilizes the Expanding Ring Search (ERS) which can be used to rebroadcast the Route Request (RREQ) in AODV. ERS works by searching successively larger areas to optimize the process of broadcasting the RREQ utilizing the route cache stored within intermediate nodes while

Blocking-ERS works by introducing a delay equal to 2 * the hop count in each ring. After this delay the intermediate node may receive a chase packet called “stop_instruction” from the source node. In case of receiving the chase packet, the intermediate node will discard RREQ else it will rebroadcast the RREQ to cover a larger area.

Limited-hop Broadcast Algorithm (LHBA) was proposed in [Zhang and Jiang, 2005]. It uses chase packets to optimize the route request by reducing the redundancy of the route request in an effort to alleviate the broadcast storm problem. In this algorithm the chase packet will be broadcasted by the route finders to K hop neighbours to free this part of the network from the fulfilled route request.

In the algorithm of [Gargano, Hammar, 2004], the sender is responsible for initiating the chase packet which may then experience an extra delay in catching up with the route request. This shortcoming has been addressed in [Zhang and Jiang, 2005] which solves this problem by initiating the chase request by any node that discovers a route. However, this algorithm may congest the network with traffic causing a storm problem of chase packets also it is unsuitable for multi-path discovery.

## 3. LIMITATIONS

Despite the fact a lot of effort that has been devoted to minimising the overhead of the route discovery process, most existing strategies that have been proposed in the literature [Das, Castaneda, 1998; Gargano, Hammar, 2004; Minematsu, Saito, 2002; Park, Kim, 2006;Young-Bae and Nitin, 2000] have one or more of the following deficiencies:

- Need special resources such as location determining technology (e.g. GPS) which might be costly or hard to implement. An example of such strategies is presented in [Young-Bae and Nitin, 2000].

- Require storage of a large amount of historical data in the nodes which requires additional memory and drains power. For some examples see [Castaeda, Das, 2002; Das, Castaneda, 1998; Minematsu, Saito, 2002].

- Depend on information that has a high chance of being stale as is the case with some examples in [Castaeda, Das, 2002; Minematsu, Saito, 2002].

- Do not improve the caching and overhearing capabilities, i.e. information gathering, of new routes. Some of such strategies are presented in [Castaeda, Das,2002; Chang and Liu, 2004; Koutsonikolas, Das, 2005; Minematsu, Saito, 2002; Young-Bae and Nitin, 2000].

- Do not utilize the resources available. Use a small portion of the channel time for broadcasting route requests which will delay the discovery process and/or reduce chances of finding new routes. Such delays in route discovery may also increase the possibility of losing these routes after their discovery. Likewise, such lack of utilization may hinder the chasing process. For more details and examples see [Gargano, Hammar, 2004] or [Park, Kim, 2006].

- Flood the network with broadcast messages causing a broadcast storm problem that waste both bandwidth and power as discussed in [Katia, Kumar, 2001].

In the rest of the paper, we will propose a new route discovery algorithm that overcomes some of the afore-mentioned deficiencies. Our algorithm uses chase packets and dual-tier approach to make the route discovery process more efficient by reducing its latency.

## 4. PRELIMINARIES

Let $N = \{n_1, n_2, \cdots, n_l\}$ be the set of nodes in some network of diameter, *d*, where the diameter of MANET is the path with the smallest number of hops between the furthest two arbitrary nodes in the network [Kurkowski, Camp, 2005]. Let $s \in N$ be a source node and define a function, $h_s : N \to \mathbb{Z}^+ \cup \{0\}$ where $h_s(v)$ is the hop count between $s$ and some other node $v \in N$ and $0 < h_s(v) < d$. Let *m* and *n* be two positive integers with $0 \le m \le n \le d$. The subset of all nodes, $v \in N$, for which $m < h_s(v) \le n$ will be called a *tier* of the network with respect to *s*. A sequence of positive integers $\alpha_i$ where $0 < i \le k$, $\exists\ \alpha_k = d$ and $\alpha_{i-1} < \alpha_i \forall i,$ $1 < i \le k$ defines a set of disjoint tiers of *N* with respect to *s*. In general, the tier $\tau_i$ is the subset of all nodes, $v \in N$, for which $\alpha_{i-1} < h_s(v) \le \alpha_i$. The width of $\tau_i$ is equal to $\alpha_i - \alpha_{i-1}$, $\forall i, 1 < i \le k$.

## 5. THE NEW ALGORITHM

We propose a new route discovery algorithm with chase packets that uses a dual-tier approach. Due to the scarce resources in MANETs, we aim at keeping our algorithm simple so we avoid collecting or manipulating large amount of data. Our algorithm works by establishing a *neighbourhood* that includes most of the likely destinations for the source node on hand and which, with its complement, the "*beyond neighbourhood*" forms a dual-tier partition of the network. The idea is to process the route requests using the full channel time during the first phase, within the neighbourhood. However, a slight delay is introduced to slowdown the propagation of route requests within the beyond neighbourhood region. As soon as the source receives a *route reply* message, it will broadcast a chase packet without imposing any extra delays, in an attempt to terminate propagation of the route request. This chase packet travels faster than the route request message within the beyond neighbourhood region.

Formally, let us consider the tier-partition $\{\tau_1, \tau_2\}$ where $\tau_1$ and $\tau_2$ are the neighbourhood and beyond neighbourhood respectively. It is obvious that the tiers are disjoint sets so $\tau_1 \cap \tau_2 = \phi$. Let us consider a source node $s$ , any node $v \in \tau_1$ satisfies the condition $h_s(v) \le \alpha_1$ and any node $u \in \tau_2$ should satisfy the condition $\alpha_1 < h_s(u)$. $\alpha_1$ is continuously tuned to adapt to the current situation using the values of $h_s(f)$ for all $f$ with respect to $s$, whether $f$ is an intermediate node or the destination itself. This can be achieved by means of average or weighted average of these values. Figure 2 shows the steps that will be performed by each node upon receiving a route request.

```
Steps preformed by each node upon
receiving a route request in 2-tier approach
If chase packet has been received then
    Discard the route request
Else
    If hop_count <= α1 then
        Broadcast the route request.
    Else
        Wait t time
        Broadcast the route request
    End if
End if
```

Figure 2. Route request processing at each node

When a route reply received, the receiving node will perform the steps in Figure 3 and upon receiving the chase message, the algorithm in Figure 4 will be performed.

```
Steps performed by each node
upon receiving a reply message
If current node is the sender  then
      Create a chase packet
      Broadcast the chase packet
      Start transmitting the data
Else
      Route the route reply to sender.
End if
```

Figure 3. Route reply processing at each node

This algorithm will implement a mechanism for updating its neighbourhood boundary using the most recent routes discovered for that source node so the boundary will be dynamically changing as the network status changes. If the destination is way beyond the boundary it will be eventually discovered without the need for any boundary immediate expansion strategy because the route request still travels outside its boundary but with slower speed.

```
Steps performed by each node
upon receiving the chase packet

If the chase packet is a duplicate then
    Discard it.
Else
    If the route request received then
        If the route request broadcasted then
            Broadcast the chase packet.
        Else
            Discard both packets.
        End if
    Else
        Discard the chase packet.
    End if
End if
```

Figure 4. Chase packets processing at each node

The source node *s* is the initiator of the chase message even if the routing algorithm is a uni-path routing protocols, as in DSR or AODV, to avoid many chase packets running in the network to catch the same route request. The price to pay here is an extra amount of delay equal to the time needed to travel $h_s(f)$ distance is added but it is worth the trouble. On the other hand, in the case of multi-path routing protocols such as on-demand multipath distance vector protocol (AOMDV) [Marina and Das, 2001] and multi-path dynamic source routing protocol (MP-DSR) [Leung, Jilei, 2001] the sender needs to discover additional routes, so the sender is the only node that observes the discovery of all the needed routes. As a result, the sender will initiate the chase message as soon as it knows that such routes are discovered this will happened immediately upon receiving the route reply.
Each node needs a warm-up period upon joining the network where this node should use the original broadcast algorithm i.e. simple flooding. Once a neighbourhood is established it will be used for the algorithm. Moreover, our algorithm is assuming that $2h_s(f)$ is not located near the boundary of $\tau_2$, which is mostly the case; otherwise the chase packet may be unable to catch the route request. In this particular situation the overhead overcomes the benefits.

## 6. MATHEMATICAL FORMULATION

We assume that most of the destinations are within $\tau_1$, all nodes are identical, the links are bidirectional and, in the interests of simplicity, we ignore the role of mobility and the influence of the MAC layer. In this algorithm, soon as a node $f$ , at distance $h_s(f)$ from the source $s$, finds a route; it will send a route reply to the source, in the reverse direction and $f$ will then stop propagating the route request. However, other nodes may continue to propagate the route request throughout the network since they may not be aware of the successful route discovery by node $f$ .

When the source receives the route reply, it will initiate and broadcast the chase packet while the route request would still be propagating throughout the network at almost twice the distance from the source i.e. $2h_s(f)$. Moreover, by the time the chase packet is $2h_s(f)$ distance from the source, the route request would have propagated further and the chase packet would still be chasing it.

Let us consider the velocity of the chase packet $v_2$ and the velocity of the route request to be $v_2$ within $\tau_1$ and $v_{12}$ within $\tau_2$. Within the 1st tier, $\tau_1$, the route request and the route reply as well as the chase packet will all be travelling using the full channel time, i.e. $v_2 = 1$. On the other hand, within the 2nd tier, $\tau_2$, the algorithm will slowdown the propagation of the route request from $v_2$ to $v_2\beta$ , where $\beta = \alpha_1 / d$ , to allow the chase packet to catch the route request. So the velocity $v_{12}$ will be:

$$v_{12} = v_2\beta \qquad (1)$$

When the chase packet is initiated we have a distance of $2\,h_s(f)$ between the route request and the chase packet. Moreover, the chase packet will always catch the route request in $\tau_2$ where $v_{12} < v_2$.

Below we will calculate the chase time, $t_c$ , and the total time, *T*, considering the following two possibilities:

- Case 1: The route request will be in the 1st tier, $\tau_1$, when the source initiates and broadcasts the chase packet i.e. $h_s(f) \le \alpha_1 / 2$.

- Case 2: The route request will be in the 2nd tier, $\tau_2$, when the source initiates and broadcast the chase packet i.e. $h_s(f) > \alpha_1 / 2$.

### 6.1. Calculating the Chase Time $t_c$

The time $t_c$ that is needed for a particular chase packet to catch the route request that associated with it can be calculated from the following formula:

$$v_2 t_c = \begin{cases} \alpha_1 - ((\alpha_1 - 2h_s(f))/v_2)v_{12} + v_{12}\, t_c & h_s(f) \le \alpha_1/2 \\ 2h_s(f) + v_{12}\, t_c & h_s(f) > \alpha_1/2 \end{cases} \quad (2)$$

Let us consider the Case 1. We can use (1) and (2) to calculate the value of chase time $t_c$ from the following formula:

$$v_2\, t_c = \alpha_1 - \alpha_1\beta + 2h_s(f)\beta + v_2 t_c \beta \quad (3)$$

Giving us the value for the chase time $t_c$ as follows:

$$t_c = (\alpha_1 + \beta 2h_s(f) - \beta\alpha_1))/(v_2(1-\beta))$$

$$t_c = \alpha_1(1-\beta)/v_2 + \beta 2h_s(f)/(v_2(1-\beta)) \quad (4)$$

Now let us consider Case 2. From (1) and (2) we can conclude the following formula:

$$v_2 t_c = 2h_s(f) + v_2 t_c \beta \quad (5)$$

Then the chase time $t_c$ can be calculated as follows:

$$t_c = 2h_s(f)/(v_2(1-\beta)) \quad (6)$$

### 6.2. Calculating the Total Broadcast Time *T*

The total broadcast time, *T*, is the time from sending a particular route request until the chase packet catches such route request. It can be calculated as:

$$T = 2h_s(f)/v + t_c \quad (7)$$

Where $v$ stands for the route request velocity. In the dual-tier approach the value of $v$ differs to be $v_2$ or $v_{12}$ depending on whether the route request is currently broadcasted within the tier $\tau_1$ or $\tau_2$ respectively.

For the Case 1, the total broadcast time, *T*, will be calculated as:

$$T = 2h_s(f)/v_2 + t_c \quad (8)$$

By taking the value of $t_c$ from (4) and simplifying the formula we get:

$$T = (\alpha_1 + 2h_s(f) - \alpha_1\beta)/(v_2(1-\beta)) \quad (9)$$

For the Case 2, the total broadcast time, *T*, can be calculated as follows:

$$T = \alpha_1/v_2 + (2h_s(f) - \alpha_1)/v_{12} + t_c \quad (10)$$

Using (1) in (10), *T* can be simplified as:

$$T = (\alpha_1\beta + 2h_s(f) - \alpha_1)/(v_2\beta) + t_c \quad (11)$$

By taking the value of $t_c$ from (6) we get:

$$T = (\alpha_1\beta + 2h_s(f) - \alpha_1)/(v_2\beta) + 2h_s(f)/(v_2(1-\beta)) \quad (12)$$

### 6.3. Performance Comparison

In this section we conduct a comparison between our algorithm and both of Blocking-ERS in [Park, Kim, 2006] and Limited Broadcasting in [Gargano, Hammar, 2004]. Such comparison aims at studying the behaviour of these three algorithms in a typical network as well as a large scale network, a scalable network. This study concentrates on evaluating the growth of the chase time, $t_c$, and the total broadcast time, *T*. It uses various values for the hop count from the source to the route finder, $h_s(f)$, for different routes under the same environment. For a typical network with a diameter of 20 hops, Figure 5 shows the chase time, $t_c$ , and Figure 6 shows the total broadcast time, *T*, using various hop count values for the first network . Both figures are drawn using Table 1 and Table 2 respectively. Limited Broadcasting algorithm uses a delay of 0.75 thus we have chosen the values for $\alpha_1$ and $d$ that yield the same delay for fair compression. Blocking-ERS did not mention a formula for calculating $t_c$ so we are assuming that chase time for Blocking-ERS is $t_c$ = 2 x hop count.

In Figure 5, we can see that our algorithm needs less chase time than Limited Broadcasting to catch the route request message. Moreover, as the value of the hop count increases the time our algorithm requires to catch the route request will be much less as compared to the time required by Limited Broadcasting. For instant, when the hop count, $h_s(f)$ , was given the value 4 and 8 the chase time needed for our algorithm was less by 5.33 and 10.67 time unit as compared to the required chase time by Limited Broadcasting. Furthermore, as the hop count increases our algorithm is superior nonlinearly with respect to the chase time. However, Blocking-ERS gives the lowest chase time this due to the fact that the route requests get queued for longer time which makes the catching process quicker but will eventually increase the total broadcast time as shown in Figure 6.

| $h_s(f)$ | Chase Time $t_c$ | | |
|---|---|---|---|
| | our algorithm | Limited Broadcasting | Blocking-ERS |
| 2 | 5.33 | 8 | 4 |
| 3 | 8.00 | 12 | 6 |
| 4 | 10.67 | 16 | 8 |
| 5 | 13.33 | 20 | 10 |
| 6 | 16.00 | 24 | 12 |
| 7 | 18.67 | 28 | 14 |
| 8 | 21.33 | 32 | 16 |

Table 1. Chase time ($t_c$) for typical network.

For the total broadcast time, $T$, Figure 6 shows the performance of our algorithm compared to both Limited Broadcasting and Blocking-ERS using different hop count values. We can clearly see that when the hop count equal 4 our algorithm requires total time less than Limited Broadcasting and Blocking-ERS by 20.33 and 140.33 respectively. Furthermore, as the hop count increases our algorithm outperform the other two algorithms nonlinearly with respect to the total broadcast time.

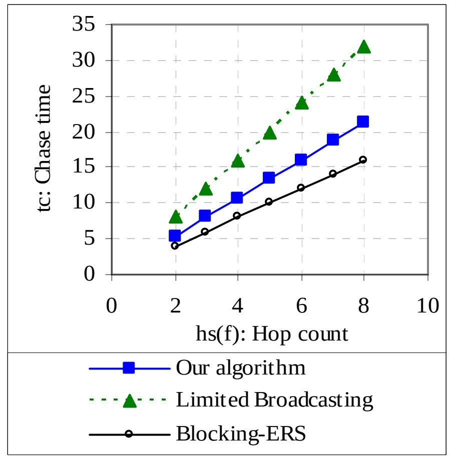


Figure 5. Chase time ($t_c$) for typical network.

| $h_s(f)$ | Total Time $T$ | | |
|---|---|---|---|
| | our algorithm | Limited Broadcasting | Blocking-ERS |
| 2 | 6.33 | 24 | 78 |
| 3 | 17.00 | 36 | 119 |
| 4 | 27.67 | 48 | 168 |
| 5 | 38.33 | 60 | 225 |
| 6 | 49.00 | 72 | 290 |
| 7 | 59.67 | 84 | 363 |
| 8 | 70.33 | 96 | 444 |

Table 2. Total broadcast time ($T$) for typical network.

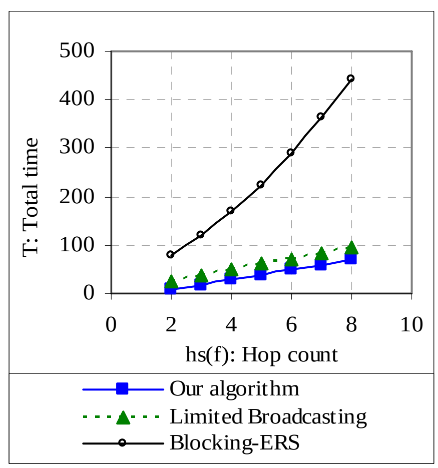


Figure 6. Total broadcast time ($T$) for typical network.

For the scalable network with a diameter of 60 hops, Figure 7 and Figure 8 also show the chase time, $t_c$ , and the total broadcast time, $T$, respectively using various hop count. These figures are drawn from the Table 3 and Table 4 respectively using the same delay of 0.75 for both our algorithm and Limited broadcasting.

| $h_s(f)$ | Chase Time $t_c$ | | |
|---|---|---|---|
| | our algorithm | Limited Broadcasting | Blocking-ERS |
| 8 | 21.33 | 32 | 16 |
| 10 | 26.67 | 40 | 20 |
| 12 | 32.00 | 48 | 24 |
| 14 | 37.33 | 56 | 28 |
| 16 | 42.67 | 64 | 32 |
| 18 | 48.00 | 72 | 36 |
| 20 | 53.33 | 80 | 40 |
| 22 | 58.67 | 88 | 44 |
| 24 | 64.00 | 96 | 48 |
| 26 | 69.33 | 104 | 52 |
| 28 | 74.67 | 112 | 56 |

Table 3. The chase time ($t_c$) for scalable network.

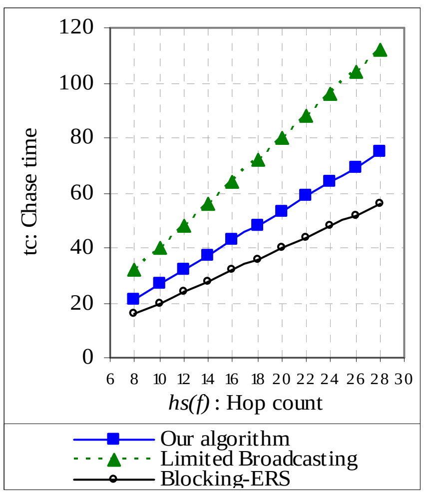


Figure 7. Chase time ($t_c$) for scalable network.

For the total broadcast time, $T$, Figure 8 shows the performance comparing the same algorithms using different hop counts. We can clearly see that our algorithm needed less total times.

| $h_s(f)$ | Total Time $T$ | | |
|---|---|---|---|
| | our algorithm | Limited Broadcasting | Blocking-ERS |
| 8 | 40.33 | 96 | 96 |
| 10 | 61.67 | 120 | 140 |
| 12 | 83.00 | 144 | 192 |
| 14 | 104.33 | 168 | 252 |
| 16 | 125.67 | 192 | 320 |
| 18 | 147.00 | 216 | 396 |
| 20 | 168.33 | 240 | 480 |
| 22 | 189.67 | 264 | 572 |
| 24 | 211.00 | 288 | 672 |
| 26 | 232.33 | 312 | 780 |
| 28 | 253.67 | 336 | 896 |

Table 4. The total broadcast time ($T$) for scalable network.

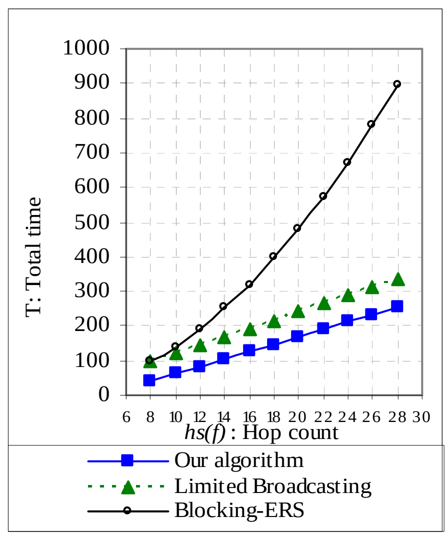


Figure 8. Total broadcast time ($T$) for scalable network.

Our main gaol is to reduce the route request latency represented by the total time $T$. Considering both the typical network and scalable network, let us compare the total time T for our algorithm with the two algorithms as shown in Figure 6 and Figure 8 we can clearly see the superiority of our algorithm i.e. less latency. Furthermore, looking at the asymptotic behaviour, our algorithm will further outperform the two existing algorithms with further growth of the hop count regardless of network size.

## 7. SIMULATION

A simulation has been conducted to evaluate our new algorithm against both Limited Broadcasting and Blocking-ERS. The three algorithms were implemented in the AODV protocol using NS2 version 2.29 [Fall, 2000]. The comparison metrics include the chase time and the total broadcast time to study the route request latency.

### 7.1. Simulation Environment

We model a typical and scalable network with network diameter of 20 and 60 respectively. Mobile nodes were placed within a squared simulation area of 1000 meter for the first network and 3000 meter for the second. Table 5 provides a summary of the chosen parameter values for both networks. Moreover, the simulation area for the scalable network was increased to keep the node density approximately constant in all scenarios. The node density was held constant throughout the simulation runs due to the fact, that increasing network density causes more network congestion not closely related to the routing protocol in use [Sung-Ju, Belding-Royer, 2003].

| Parameter | Value |
|---|---|
| Transmission range | 100m |
| Topology size | 1000x1000 and 3000x3000 |
| Simulation time | 200-900secs |
| Packet size | 512bytes |
| Packet rate | 4pkt/sec |
| Traffic type | CBR(UDP) |
| Routing protocol | AODV |
| Network diameter | 20-60 |
| Antenna type | Omni Antenna |
| MAC protocol | IEEE 802.11 |

Table 5. Simulation parameters.

The radio propagation range for each node is 100 meter and the channel capacity is 1Mbps. the first network was simulated for 200 seconds of simulation time and the second network was executed for 900 seconds. For each hop count, ten runs were performed changing the sender and destination pair for each run. The results of these runs were averaged together to produce the resulting graphs in section 7.2.

Our equations in the previous section, mathematical formulation, were derived under the assumption that the network operates under light traffic thus with minimum message blocking. Such assumption would provide ideal conditions and enable the model to yield good predictions as the blocking component in our model is negligible. In our simulation runs, one data session was injected in each run to provide light traffic.

A traffic generator was used to simulate constant bit rate sources with a data payload of 512 bytes. Moreover, to reduce channel contention and packet collision as much as possible, one data session with a selected source and destination was simulated in each run. The source transmits data packets at a rate of 4 packets per sec. To be able to compare the simulation results with the mathematical model, section 6, we compute the time of chase packet as well as the total broadcast time for the same various hop counts.

### 7.2. Simulation Analysis

Figure 9, Figure 10 and Figure 11 display the results of running our algorithm against both Limited Broadcasting and Blocking-ERS for 200 seconds using a typical network with 20 node diameter in 1000 square meter. On the other hand, Figure 12, Figure 13 and Figure 14 display the results of running same algorithms for 900 seconds but in a larger network with a diameter of 60 node in 3000 square meter.

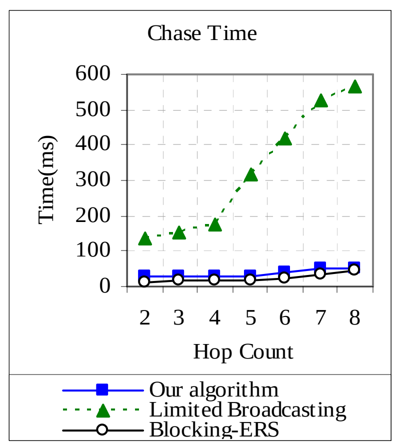


Figure 9. Chase time for typical network.

Figure 9 and Figure 12 show the chase time for all three algorithms. Blocking-ERS has a slightly less chase time due to the fact that the route requests get queued for longer time which makes the catching process quicker but will eventually increase the total broadcast time as shown in Figure 10 as well as Figure 13.

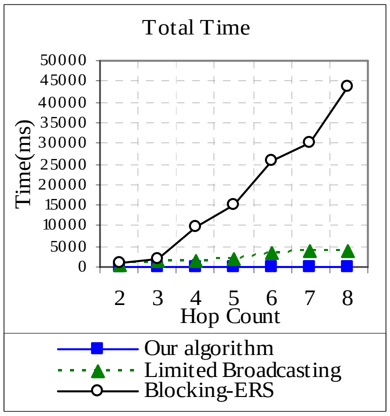


Figure 10. Total broadcast time for typical network.

Looking at the total broadcast time, our algorithm outperforms both Limited Broadcasting and Blocking-ERS regardless of network size. This fact was observed from Figure 10 and Figure 13 as well as Figure 6 and Figure 8 from the mathematical model. This is due to the fact that Limited Broadcasting and Blocking-ERS imposes more delays than our algorithm on the network. To highlight the difference in total broadcast time between our algorithm and Limited Broadcasting we extracted Figure 11 and Figure 14 from Figure 10 and Figure 13 respectively scaling both graphs to show more detail.

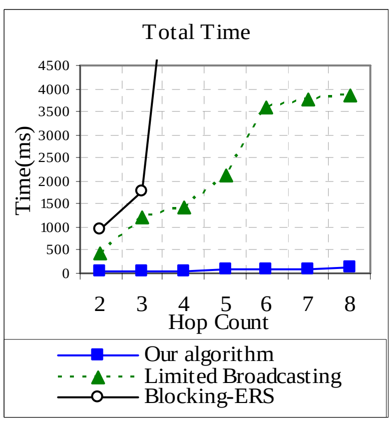


Figure 11. The first 4500 ms of Figure 10.

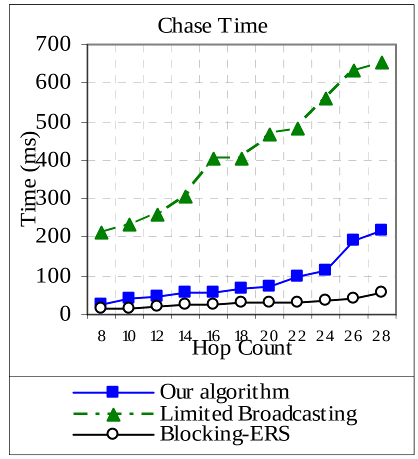


Figure 12. Chase time for scalable network.

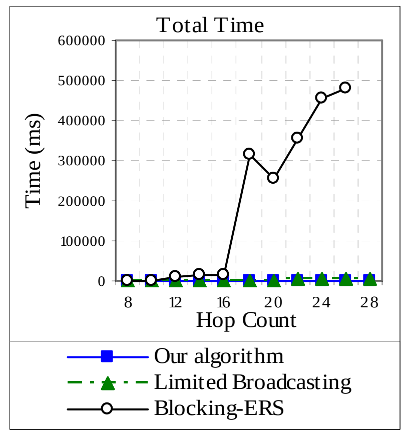


Figure 13. Total broadcast time for scalable network.

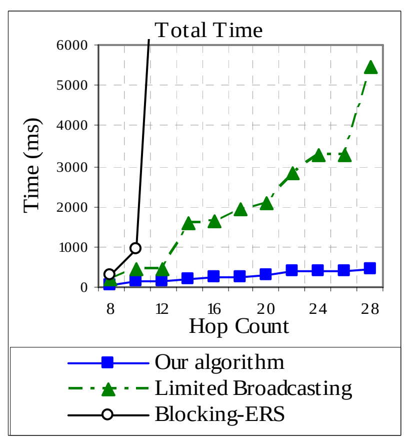


Figure 14. The first 6000 ms of Figure 13.

The simulation results were found to be inline with the mathematical model. As the graphs in both cases demonstrate the same trend keeping in mind that the simulation environment was subject to some overhead i.e. channel contention and packet collision when running the simulation.

## 8. CONCLUSION AND FUTURE WORK

In this paper, we have developed a new dual-tier route discovery algorithm for MANETs with chase packets that works by including most of the likely destinations for the source node on hand in the first tier and broadcasting the route requests with full channel time within such tier. In order to provide a much better chance for the chase packets to catch the route requests, the algorithm delays the propagation of the route requests within the second tier which will in turn minimise the network congestion. The algorithm is also adaptive and continuously updates the boundary between the two tiers to provide the best performance. Moreover, we have provided a detailed performance evaluation for our new algorithm and compared it with two of the existing algorithms in the literature [Gargano, Hammar,2004] and [Park, Kim,2006] using mathematical model and simulation. Our evaluation showed that our algorithm requires less total broadcast time which makes it superior and gets better asymptotically with large networks.

Route request latency in our algorithm is lower than the other two algorithms which will have a good impact on the network performance since the transmission of data packets will start earlier in our algorithm that will improve the network latency. As part of our future work, we are currently working towards extending our simulation model to work in the presence of heavy traffic.

## REFERENCES

Abolhasan, M., T. Wysocki, and E. Dutkiewicz,2004 "*A review of routing protocols for mobile ad hoc networks."* Ad Hoc Networks. **2**(1): p. 1-22.

Adjih, C., et al. 2003 "*Optimized Link State Routing Protocol*." October [cited.

Castaeda, R., S. Das, and M. Marina,2002 "*Query localization techniques for on-demand routing protocols in ad hoc networks."* Wireless Networks. **8**(2/3): p. 137-151.

Chang, N. and M. Liu, 2004 "*Revisiting the TTL-based controlled flooding search: optimality and randomization*." in *Proceedings of the 10th annual international conference on Mobile computing and networking*. ACM Press: Philadelphia, PA, USA.

Chlamtac, I., M. Conti, and J.J.N. Liu,2003 "*Mobile ad hoc networking: imperatives and challenges."* Ad Hoc Networks. **1**(1): p. 13-64.

Das, S.R., et al. 1998. "*Comparative performance evaluation of routing protocols for mobile, ad hoc networks*".

Fall, K.,2000 " *ns Notes and Documentation."* The VINT Project.

Gargano, L., M. Hammar, and A. Pagh. 2004. "*Limiting flooding expenses in on-demand source-initiated protocols for mobile wireless networks*". in *18th International Parallel and Distributed Processing Symposium*.

Haas, Z.J., M.R. Pearlman, and P. Samar, 2002 "*The Zone Routing Protocol (ZRP) for Ad Hoc Networks*." IETF MANET Working Group.

Johnson, D., D. Maltz, and Y.-C. Hu, 2003 "*The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*." The Internet Engineering Task Force, IETF, draft-ietf-manet-dsr-09.txt.

Katia, O., V. Kumar, and T. Gene,2001 "*Flooding for reliable multicast in multi-hop ad hoc networks."* Wireless Networks. **7**(6): p. 627-634.

Koutsonikolas, D., et al. 2005. "*On optimal TTL sequence-based route discovery in MANETs*". in *Second International Workshop on Wireless Ad Hoc Networking*.

Kurkowski, S., T. Camp, and M. Colagrosso,2005 "*MANET simulation studies: the incredibles."* SIGMOBILE Mob. Comput. Commun. Rev. **9**(4): p. 50-61.

Leung, R., et al. 2001. "*MP-DSR: a QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks*".

Marina, M.K. and S.R. Das. 2001. "*On-demand multipath distance vector routing in ad hoc networks*".

Minematsu, M., et al. 2002. "*HOWL: an efficient route discovery scheme using routing history in ad hoc networks*". in *27th Annual IEEE International Conference on Local Computer Networks (LCN'02)*

Murthy, S. and M. B., 2004 *Ad Hoc Wireless Networks: Architectures and Protocols*: Prentice Hall.

Park, I., J. Kim, and I. Pu. 2006. " *Blocking Expanding Ring Search Algorithm for Efficient Energy Consumption in Mobile Ad Hoc Networks*". in *Third Annual Conference on Wireless On-demand Network Systems and Services*. Les Menuires, France.

Perkins, C., 2001 *Introduction to Ad hoc networking*: Addison Wesley.

Perkins, C., E. Belding-Royer, and S. Das, 2003 "*AODV Ad Hoc On-Demand Distance Vector Routing*." The Internet Engineering Task Force, IETF, RFC 3561.

Sung-Ju, L., E. Belding-Royer, and C. Perkins,2003 "*Scalability study of the ad hoc on-demand distance vector routing protocol."* Int. J. Netw. Manag. **13**(2): p. 97-114.

Tanenbaum, A., 2003 *Computer Networks*: Pearson Education.

Young-Bae, K. and H.V. Nitin,2000 "*Location-aided routing (LAR) in mobile ad hoc networks."* Wireless Networks. **6**(4): p. 307-321.

Yu-Chee, T., et al.,2002 "*The broadcast storm problem in a mobile ad hoc network."* Wireless Networks. **8**(2/3): p. 153-167.

Zhang, H. and Z.P. Jiang. 2005. "*On reducing broadcast expenses in ad hoc route discovery*". in *Second International Workshop on Wireless Ad Hoc Networking (WWAN) ICDCSW'05).*

## AUTHORS BIOGRAPHY

**Mznah A. Al-Rodhaan** received her B.S. (Hons) and MSc in Computer Science from King Saud University, Riyadh, Saudi Arabia in 1999 and 2004 respectively. Currently, she is a Ph.D. candidate in the Department of Computing Science at University of Glasgow, UK. Her current research interests include computer networks, wireless networks, mobile ad hoc networks, routing, and broadcasting.

**Lewis M. Mackenzie** graduated with a B.Sc. in Mathematics and Natural Philosophy from the University of Glasgow, U.K., in 1980. He was awarded the PhD in 1984, also at the University of Glasgow, for his work in the development of multicomputers for use in nuclear physics. He now lectures at the Department of Computing Science at the University of Glasgow, which he joined in 1985. His current research interests include multicomputers, high-performance networks, and simulation.

**Mohamed Ould-Khaoua** received his B.Sc. degree from the University of Algiers, Algeria, in 1986, and the M.App.Sci. and Ph.D. degrees in Computer Science from the University of Glasgow, U.K., in 1990 and 1994 respectively. He is currently a Reader in the Department of Computing Science at the University of Glasgow, U.K. His research focuses on applying theoretical results from stochastic processes and queuing theory to the quantitative study of hardware and software architectures. He is the founding co-chair of the international workshop series on performance modelling, evaluation, and optimization of parallel and distributed systems (PMEO-PDS). He has served on the program committee of many international conferences and workshops. His current research interests are performance modelling/evaluation of parallel and distributed systems, parallel algorithms, and wired and wireless networks.