TL-ERS: A Traffic Locality Based Expanding Ring Search for MANETs

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 approach to traffic locality then use it to develop a new route discovery algorithm. The algorithm, we named Traffic Locality-Expanding Ring Search (TL-ERS), improves the route discovery process for MANETs that exhibit traffic locality by establishing a neighbourhood that includes the most likely destinations for a particular source, then broadcasting route requests using this neighbourhood as a first ring, in which to search for the target. If route discovery in this ring proves unsuccessful, then the algorithm establishes a second ring, double the size of the first, if route discovery here also fails the algorithm finally resorts to flooding. TL-ERS is adaptive and continuously updates the boundary of the source node’s neighbourhood to optimize performance. Furthermore, we provide a detailed performance evaluation using simulation to demonstrate its advantages over both the AODV and the Expanding Ring Search (ERS).

Keywords: MANETs, On-demand Routing Protocols, Route Discovery Optimization, Traffic Locality, ERS.

I. INTRODUCTION

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 as in access point dependent networks [20] or infrastructure-less such as Mobile Ad hoc Networks (MANETs) [7, 20, 24]. Some of the dominant initial motivations for MANET technology came from military applications in environments that lack infrastructure. However, MANET research subsequently diversified into areas such as disaster relief, sensors networks, and personal area networks [24].

The design of an efficient routing strategy is a very challenging issue due to the limited resources in MANETs [20]. Multi-hop routing protocols can be divided into three categories: proactive, reactive, and hybrid [1]. 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) [2]. 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 [8]. In on-demand routing, when a source needs to send messages to a destination it initiates a broadcast-based route discovery process to look for one or more possible paths to the destination. Examples from this class are Dynamic Source Routing (DSR) [14] and Ad Hoc On Demand Distance Vector (AODV) [21]. 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. Zone Routing Protocol (ZRP) [12] is an example belonging to this class.

In MANETs, broadcasting is an essential part of the discovery process in on-demand routing protocols. For example, DSR and AODV depend on simple flooding as a form of broadcasting but simple flooding leads to the broadcast storm problem [23]. So flooding consumes a lot of node resources such as bandwidth and power.

The route discovery process often floods the network with route request packets looking for routes throughout the network. Unfortunately, the route request will keep propagating even after a route has been found and this will, of course, contribute to congestion and wastage of 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 [3, 11]. In this paper, we will show how exploiting the concept of traffic locality can help to reduce such overhead and improve the efficiency of the route discovery process of on-demand routing protocols such as AODV in networks that exhibits traffic locality.

The rest of the paper is organised as follows: Section II presents the related work in the literature while section III discusses locality in MANETs. Section IV describes our proposed algorithm, TL-ERS, that utilizes traffic locality; section V describes the simulation environment and evaluates the performance by conducting a comparative study that demonstrates the superiority of our algorithm over both AODV and Expanding Ring Search (ERS) [21] in terms of reducing route request overhead. Finally, Section VI concludes this study.

II. RELATED WORK

In on-demand routing protocols, the broadcasting of the route request used in route discovery dominates most of the routing overhead. Several approaches have been proposed to reduce this overhead by using variations of limited broadcasting so that the overhead of the route discovery process is reduced by limiting the route request broadcast.
The broadcast of the route request can be controlled using the TTL field in the IP header of the route request packet. Expanding Ring Search (ERS) is one of the route request optimization techniques that lower the overhead cost for DSR as presented in [15], and for AODV as proposed in [22]. In ERS the source node searches for the target in a multi-ring scheme instead of one-to-all scheme. This is achieved by increasing the TTL value from an initial value to a predefined threshold to expand the radius of the search linearly. Reference [25] tried to find the best initial value for TTL theoretically. They found that the pessimistic search, the worst case where the initial ring may contain the needed route, gives the best performance if destination’s speed is known to the source node but the absence of this information makes it a challenging task. Another study [5] proposed two approaches. The first assumes the probability distribution of the destination is known prior to the discovery process while the second one assumes the probability distribution of the destination is not known, fitting more with the unpredictability of MANETs. The second approach uses a sequence of random TTL values to minimize the worst-case search cost. Approaches in [5] was later investigated in [16] on DSR and was observed that when caching of previous routes is taking into consideration the route discovery has similar overhead but higher latency compared to the basic route discovery in DSR.

Hop-Wise Limited broadcast (HoWL) [18] is another approach that limits a route request by predicting the destination location from old routes. It sends the route request packet with a TTL equal to the average of old routes, maybe stale, to that particular destination if it targeted that destination before; otherwise it uses the simple flooding.

III. TRAFFIC LOCALITY IN MANETS

Locality is observed in networking through the fact that devices within the same geographical area tend to communicate more often than those that are further apart so those devices exhibit temporal and/or spatial locality [9]. This has motivated the concepts of network clusters and workgroups. The working set of a particular node in a network normally refers to the set of nodes which the node is mostly communicating with, not necessarily neighbours, along with intermediate nodes on routes to those targets, during some time interval of interest. In MANETs, locality is observed through the fact that neighbours, nodes in the same geographical area, tend to receive communication from the same sources, highlighting the spatial locality. Also, nodes communicated with the near past have high probability of re-communicating in the near future leading to temporal locality. Looking at the behaviour of MANETs overall, we observe that the traffic follows a certain pattern, not purely spatial or temporal, in which the source node tends to communicate with a certain set of nodes more than others. This observation has motivated us to introduce another form of locality in MANETs, referred to henceforth as traffic locality.

Traffic locality is based on the working set concept, identifying the set of nodes that a given source is mostly communicating with. These nodes are not necessarily identified by spatial or temporal locality but rather by intensity of traffic within the working set over some time interval. Moreover, if a source exhibits traffic locality with a certain destination, the intermediate node comprising the route in question will also be a member of the source node’s working set until one of them moves away.

The reasons as to why MANETs exhibit traffic locality are related to the communication requirements of the users carrying and operating the nodes. One common application that exhibits traffic locality in MANETs is a communication group ad hoc network [19] where a group of nodes communicate with each other regardless of their location.

We use the traffic locality concept to optimize the route discovery process. This can be achieved by an adaptive route discovery algorithm that will gradually build up the node neighbourhood as a region centred at the source node and expected to contain most of the members of its working set. Establishing this neighbourhood is a challenging task as it must adapt by expanding and shrinking according to traffic in an effort to reflect the working set. Each node needs a start-up period upon joining the network within which uses simple flooding. Once a locality neighbourhood is reasonably estimated, the algorithm broadcasts any route request with the full capacity of the channel to all nodes within that neighbourhood, in an attempt to minimise the route discovery time. Due to the scarce resources in MANETs, the algorithm must be kept simple by avoiding the collection or manipulation of large amount of data. Each source node has a locality parameter $L_P$ which reflects the current estimated depth of its neighbourhood as it may be defined by the weighted average of hop counts between that source node and route finders of previous routes: the finder of a route is the first node that finds the route in its cache table whether it is the destination or an intermediate node. The source node may also keep a counter of its previous route requests whether a weighted or standard average is used. A node, $x$, is considered to be part of the neighbourhood set of a source node, $s$, if the hop count between $s$ and $x$ is less than or equal to $L_P$.

The algorithm is adaptive and adjusts its neighbourhood. If a route finder is outside the neighbourhood then this requires that the neighbourhood be altered via some adaptation strategy. One such strategy is as follows: $L_P$ is adjusted by taking the weighted average of the current value of $L_P$ and the new hop count between the source and the route finder. Alternatively a doubling strategy [11] could be used but this lacks a countervailing shrinking ability and so will not be considered. Fig. 1 shows how the algorithm would shrink or expand the neighbourhood of the source node.

To illustrate the neighbourhood adjustment process, let us consider the source node $s$ after it has completed its start-up phase and let $N = \{n_1, n_2, \ldots, n_i\}$ 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 two furthest apart nodes in the network [17]. Let $s \in N$ be a source node and define a function $h_s : N \rightarrow \mathbb{Z^+} \cup \{0\}$ where $h_s(v)$ is the hop count between $s$ and some other
node \( v \in \mathbb{N} \) and \( 0 < h_s(v) < d \). When the source node \( s \) communicates with any node \( f \) that is \( h_s(f) \) hops away and \( y \) is the number of previous route requests that already been sent by \( s \), the following formula is used by \( s \) to update its LP:

\[
LP_{\text{new}} = \begin{cases} 
\alpha \times LP_{\text{old}} + (1 - \alpha) \times h_s(f) & h_s(f) \geq LP_{\text{old}} \\
\alpha \times LP_{\text{old}} + (1 - \alpha) \times h_s(f) & h_s(f) < LP_{\text{old}} 
\end{cases}
\]

where \( \alpha = \frac{y}{y + 1} \). It is clear that if \( h_s(f) \geq LP_{\text{old}} \) then the neighbourhood of \( s \) will expand and it will shrink if \( h_s(f) < LP_{\text{old}} \).

Fig. 2 shows the steps of updating the locality parameter \( LP \) by the source node after receiving the route reply so the neighbourhood region will be ready for next route request. For clarity, the function Ceiling will return the smallest integer greater than or equal to its parameter while the function Floor will return the greatest integer less than or equal to its parameter. To prevent \( \alpha \) from approaching too close to 1 as \( y \) getting bigger due to \( \lim_{y \to \infty} \alpha = 1 \) , where only the function Ceiling or Floor will affect the value of \( LP \), we need to reset \( y \) to an initial value, Initial\(_y\), when \( y \) reaches some threshold, Threshold\(_y\).

**Algorithm preformed by source node upon receiving a route reply**

\[
\text{If } y > \text{Threshold, then} \\
\quad y = \text{Initial}\(_y\) \\
\text{End if} \\
\alpha = \frac{y}{y + 1} \\
LP_{\text{new}} = \alpha \times LP_{\text{old}} + (1 - \alpha) \times LP_{\text{old}} \\
\text{If hop count} < LP_{\text{old} \text{ consecutive}} \text{ then} \\
\quad LP_{\text{new}} = \text{Floor}(LP_{\text{new}}) \\
\text{Else} \\
\quad LP_{\text{new}} = \text{Ceiling}(LP_{\text{new}}) \\
\text{End if} \\
LP_{\text{old}} = LP_{\text{new}} \\
\text{y} = y + 1
\]

IV. **TRAFFIC LOCALITY-EXPANDING RING SEARCH**

Network-wide flooding is a very expensive process. A process such this should be avoided in a resource-limited environment such as a MANET. One way to search for a route without covering the whole network is to use Expanding Ring Search (ERS) [22]. It works by searching successively larger areas centred around the source node, until the required route is located. The basic idea behind ERS is to find a local node with a valid route to the destination and avoid flooding the entire network in search of such a route. Therefore, the source node starts the search by broadcasting a route request with TTL= TTL\(_{-}\)START to flood the first ring. Each time the source times-out without receiving a reply it re-initiates the route request with TTL incremented by TTL\(_{-}\)INCREMENT. This process continues until a TTL\(_{-}\)THRESHOLD is reached. If no route has been located by this time, flooding is used with TTL = network diameter (\( d \)). All nodes in a connected network use the same fixed predefined values for TTL\(_{-}\)START, TTL\(_{-}\)INCREMENT and TTL\(_{-}\)THRESHOLD. For instance, when ERS is used to optimize AODV protocol described in [21] it employs TTL parameters as in Table 1 and follows Fig. 3. Moreover, the route request time (RRT) is the time from the first initiation of a route request until it is discarded. RRT for ERS is:

\[
RRT_{ERS} = \sum_{i=1}^{T} (2i - 1) \quad h_s(f) \leq T \\
\sum_{i=1}^{T} (2i - 1) + d \quad h_s(f) > T
\]

Where \( h_s(f) \leq T \) means success in ring \( r \), \( h_s(f) > T \) means last attempt which is whole network coverage, \( T = \text{TTL-THRESHOLD} \) and \( r \) is the ring that contains \( h_s(f) \).

To obtain a search result that as close as possible to the optimal search result whilst costing less, the search strategy has to be set to suit the application scenario and system configuration. ERS is not necessarily better than simple flooding if the ERS parameters are not selected properly [5, 6, 16, 25]. Selecting the initial TTL value for the first search ring is an important step towards a more effective search [25].

Traffic Locality-Expanding Ring Search (TL-ERS) is a modification of ERS using the traffic locality concept. The main difference between ERS and TL-ERS is in the TTL parameters. ERS uses a fixed radius for all nodes in the network depending on the search ring but TL-ERS uses adaptive values. TL-ERS uses the value of \( LP \) as the radius of the first ring. \( LP \) differs from source node to source node and always updated to reflect the current environment. It uses the parameters listed in Table 1 and follows Fig. 4. Moreover, TL-ERS limits the number of rings in the worst case to three rings to achieve lower cost broadcasting as discussed in [6].

The pseudo code for initiating or reinitiating a route request with the correct TTL is described in Fig. 5.

<table>
<thead>
<tr>
<th>TABLE 1: ERS PARAMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL parameter</td>
</tr>
<tr>
<td>TTL(_{-})START</td>
</tr>
<tr>
<td>TTL(_{-})INCREMENT</td>
</tr>
<tr>
<td>TTL(_{-})THRESHOLD</td>
</tr>
</tbody>
</table>
Fig. 3. Successive rings in ERS

Algorithm performed by the source node upon sending a route request in TL-ERS approach for initiating or reinitiating route request.

If ring = 1 then
  TTL=LP
  broadcast the route request
Else
  If ring = 2 then
    TTL= 2LP
    broadcast the route request
  Else
    If ring=3 and 2LP < d then
      TTL= d
      broadcast the route request
    Else
      destination not found.
    End if
  End if
End if

Fig. 5. TTL initialization steps for initiating or reinitiating a route request in TL-ERS.

TL-ERS works by initialising the TTL field with the value of LP for the first search ring. If the source node times out without receiving a route reply, it reissues the route request with TTL equal to twice LP. If it times out again it will flood the whole network by assigning TTL to be the network diameter. The RRT of TL-ERS equals:

\[
RRT_{TL-ERS} = \begin{cases} 
LP & h_x(f) \leq LP \\
LP + 2LP & LP < h_x(f) \leq 2LP \\
LP + 2LP + d & 2LP < h_x(f) \leq d 
\end{cases}
\]

Where \( h_x(f) \leq LP \) means success in the first ring, \( LP < h_x(f) \leq 2LP \) means success in the second ring and \( 2LP < h_x(f) \leq d \) means whole network coverage.

V. SIMULATION

Simulations have been conducted to evaluate our algorithm, TL-ERS, against both AODV and AODV with ERS [21]. ERS for short, algorithms. The three algorithms were implemented using NS2 simulator version 2.29 [10] conducting extensive experiments to evaluate the performance of TL-ERS. The comparison metrics include the route request (RREQ) delay and the total number of route requests to study the route request latency and overhead. The metrics also include end-to-end delay to study the network behaviour in different scenarios.

Since nodes are mobile in MANETs, modelling these movements is not obvious. In order to simulate a new protocol such as TL-ERS, it is necessary to use a mobility model that reasonably represents the movements of a typical node [4]. Accurate mobility models should be chosen carefully to determine whether the proposed protocol will be useful when implemented or not. Moreover, one of the main characteristics of mobility in MANETs is the maximum speed of nodes because the average speed of nodes determines the rate of breaking links which increase the overhead in on-demand protocols.

In MANETs, the entity mobility models typically represent nodes whose movements are completely independent of each other, e.g. the Random Way Point (RWP) model [15]. On the other hand, a group mobility model may be used to simulate a cooperative characteristic such as working together to accomplish a common goal. Such a model reflects the behaviour of nodes in a group as the group moves together, e.g. Reference Point Group Mobility (RPGM) model [13].

A. Simulation Environment

We assume all nodes are identical, all links are bidirectional and no selfishness in the network. Mobile nodes are assumed to operate in a squared simulation area of side 1000m. The transmission range is 100m and is fixed in all nodes to approximately simulate networks with a maximum hop count of 10 hops. Each run was simulated for 900 seconds of simulation time, ignoring the first 30 seconds as a start-up period for the whole network. For each topology, 30 runs were performed. The results of these runs were averaged to produce the graphs, Fig. 6-13, shown below. TABLE 2 provides a summary of the chosen simulation parameter values.

The simulation area is kept constant in all scenarios to study TL-ERS performance in both scarce and dense environments, since we are interested in knowing the behaviour of our algorithm in both kinds of environments.

A traffic generator was used to simulate constant bit rate (CBR) with a packet data payload of 512 bytes. Moreover, data sessions, flows, between different source and destination pairs
in groups of ten nodes were simulated to simulate traffic in a network that exhibit traffic locality. The source transmits data packets at a rate of four packets per second.

### TABLE 2: SIMULATION PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission range</td>
<td>100m</td>
</tr>
<tr>
<td>Topology size</td>
<td>1000x1000m</td>
</tr>
<tr>
<td>Simulation time</td>
<td>900s</td>
</tr>
<tr>
<td>Packet size</td>
<td>512 bytes</td>
</tr>
<tr>
<td>Packet rate</td>
<td>4 pkt/s</td>
</tr>
<tr>
<td>Data sessions</td>
<td>10,15,…,50</td>
</tr>
<tr>
<td>Traffic type</td>
<td>CBR(UDP)</td>
</tr>
<tr>
<td>Routing protocol</td>
<td>AODV</td>
</tr>
<tr>
<td>Number of Nodes</td>
<td>20,30,..,100</td>
</tr>
<tr>
<td>Number of runs/point</td>
<td>30</td>
</tr>
<tr>
<td>Antenna type</td>
<td>Omni Antenna</td>
</tr>
<tr>
<td>Propagation</td>
<td>Two Ray Ground</td>
</tr>
<tr>
<td>MAC protocol</td>
<td>IEEE 802.11</td>
</tr>
<tr>
<td>Maximum speed</td>
<td>5,10,…,30m/s</td>
</tr>
<tr>
<td>Minimum speed</td>
<td>1m/s</td>
</tr>
<tr>
<td>Mobility Model</td>
<td>RGPM model</td>
</tr>
<tr>
<td>y-threshold</td>
<td>10</td>
</tr>
<tr>
<td>y-init</td>
<td>1</td>
</tr>
</tbody>
</table>

The RPGM model was utilized as a mobility model in all of our simulations since it models the random motion of groups of nodes and of individual nodes within the group. Group movements are based upon the movement of the group reference point following its direction and speed. Moreover, nodes move randomly within their group with speeds between 1 and 30m/s. Each group contains 10 nodes. The minimum speed is 1 with pause time of 50s.

In our simulation, we concentrate on three major parameters: network size, data sessions and maximum speed in three different cases by varying one parameter while keeping the other two constant as explained below:

- **Case 1:** Network size is the total number of nodes in the network. When the network size increases, the average hop length of routes also increases which may increase the error rate and/or increase network latency. Simulation has been performed using nine topologies with different number of nodes, multiples of 10, from 20 to 100 while other simulations use a fixed number of nodes i.e. constant value.
- **Case 2:** The data sessions of sizes 10, 15… 50 were used in some simulations with network of size seventy nodes and maximum speed of 30m/s. In the other hand, data sessions of size ten were used in other simulations for networks of variable sizes or variable maximum speed.
- **Case 3:** The maximum speed varied from 5, 10…30m/s in some simulation runs and fixed at 30m/s in others.

### B. Simulation Analysis

Considering case 1, variable network size, Figs 6, 7 and 8 display the results of running our algorithm, TL-ERS, against both AODV and ERS for 900 seconds using networks with different number of nodes using RGPM as the mobility model with a minimum speed 1m/s and a maximum speed of 30m/s. The number of data sessions is fixed to ten.

Fig. 6 shows the superiority of TL-ERS over both AODV and ERS by minimizing the route request overhead.

In Fig. 6(b), we have used a larger scale for TL-ERS vs. ERS to clearly compare their behaviour and we will do the same for other figures throughout this section. The RREQ overhead, measured by the number of RREQ received in the whole network, of TL-ERS is lower than ERS as Fig. 6(a) and significantly lower than AODV as shown in Fig. 6(a). In fact, the difference in RREQ overhead between TL-ERS and AODV increases with dense network. Nevertheless, the RREQ’s latency, measured by the average RRT, is almost the same as shown in Fig. 7(b) between TL-ERS and ERS. However, there is great reduction over AODV as in Fig. 7(a).

The end-to-end delay is defined to be the total delay for the actual transmitted data plus the RREQ delay. Fig. 8(a) and Fig. 8(b) show the end-to-end delay and demonstrate that TL-ERS gives less network latency than both AODV and ERS due to the reduction in number of RREQ. In other words, network performance improves due to the reduction in the number of rebroadcast route requests as presented in Fig. 6 with low latency which has a generally beneficial effect on the network performance due to the fact that the data can typically travel with less congestion.
The three algorithms give stable averages of data throughput, not shown here due to space, regardless of network size due to the fact that the communication always between nodes within the same group thus the movement follows the RGPM model.

For case 2, simulations with different number of data sessions, Fig. 9-11 display the results of running our algorithm, TL-ERS, against both AODV and ERS for 900 seconds using networks of size 70 nodes with a random speed range between 1 and 30m/s. The amount of traffic ranged from 10 to 50 flows incremented by five.

Fig. 9 and 10 show that TL-ERS improves the route request overhead over both AODV and ERS. This improvement increases with dense network. Fig. 11 shows that TL-ERS has lower network latency compare to both AODV and ERS due to the reduction in the number of rebroadcast route requests. The attractiveness of this stems from the fact the data can travel faster and with less congestion.

For case 3, Figs. 12-13 were extracted from simulating the three algorithms while increasing the maximum speed by five steps starting from 5 to 30m/s. The number of nodes was fixed at 70 and data sessions at 10. The overhead of the RREQs in AODV is higher than both TL-ERS and ERS as in Fig. 12(a) and the RREQ overhead of TL-ERS is better than ERS as shown in Fig. 12 (b). Fig 13(a) shows great reduction network latency due to reduction in RREQs latency in both ERS and TL-ERS over AODV so the success case is the dominant in both TL-ERS and ERS while Fig. 13(b) shows little improvement between TL-ERS and ERS regardless of speed which will improve the network latency as in Fig. 13.


