On the Performance of Traffic Locality Oriented Route Discovery Algorithm with Chase Packets

Mznah A. Al-Rodhaan[1], Lewis Mackenzie[2], Mohamed Ould-Khaoua[3]

***Abstract*—A new traffic locality oriented route discovery algorithm with chase packets, referred to as TLRDA-C, is introduced. It improves route discovery process in on-demand routing protocols for MANETs running applications that exhibit traffic locality. The algorithm defines a neighbourhood region that includes the most likely destinations for each given source node. TLRDA-C broadcasts any route request travelling within their source node's neighbourhood region according to the routing algorithm used. However, propagation of the route request is deliberately delayed within the beyond-neighbourhood region to provide the associated chase packet with an opportunity to stop the fulfilled route request and minimise network congestion. The algorithm is adaptive and continuously updates the boundary of each source node's neighbourhood to improve performance. We provide detailed performance evaluation using simulation modelling and compare our algorithm with AODV, Limited Broadcasting, and Blocking-ERS. Our result shows that TLRDA-C improves the performance by minimizing the average end-to-end delay as well as the network overhead and congestion level.**

***Index Terms*-Chase Packets, MANETs, On-demand routing protocols, Route discovery process.**

I. Introduction

Users dreamed of wireless connectivity with the appearance of mobile devices such as notebooks and PDAs, so wireless networks appeared and this duly become a reality. Wireless networks might be infrastructure-oriented, such as the access point dependent networks [17] or infrastructure-less such as Mobile Ad hoc NETworks (MANETs) [17], [22]. 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 [22].

The design of an efficient and reliable routing strategy is a very challenging problem due to the limited resources in MANETs [17]. Many multi-hop routing protocols have been proposed and investigated in the literature as in [2], [11], [13], [20], [21], [23]. MANETs routing protocols is divided into three categories: proactive, reactive, and hybrid [1], [20]. In proactive routing protocols (table-driven), the routes to all destinations (or parts of the network) are determined statically at the start up then maintained using a periodic route update process, An example of this class is the Optimized Link State Routing Protocol (OLSR) [2]. However, in reactive routing protocols (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]. When a source node needs to send messages to an unseen 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 of this class are the Dynamic Source Routing (DSR) [13] and Ad Hoc On Demand Distance Vector (AODV) [21]. Finally, hybrid routing protocols combine the basic properties of the first two classes of protocols; so they are both reactive and proactive in nature as in Zone Routing Protocol (ZRP) [11].

When a source node needs to send packets to an unseen destination in on-demand routing algorithms, it initiates a route discovery process looking 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 discovered route.

DSR [13] and AODV [21] both use broadcasting for route discovery process. These protocols may 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. Source node first searches its routing table where any seen or overheard route might have been stored for future use; if not found, a route discovery process is started using any form of broadcasting. In the simple flooding, route requests keep propagating until the time to live (TTL) field reaches zero or the whole connected network is covered. Unfortunately, the flooding leads to redundancy that highly congest the network and increase the chances of collision: these combined are known as the broadcast storm problem [24]. Moreover, simple flooding consumes a lot of network resources such as bandwidth and power which can be reduced by stopping the route request as soon as possible upon the discovery of the needed route as a way of controlling the flooding.

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 process can be improved by minimizing such overhead and reducing or better stopping the unnecessary propagation of route request packets after discovering the route. The new algorithm in this study defines a local neighbourhood that includes the most likely destinations for each given source node and broadcasts route requests originated from that source node according to the routing algorithm in used. However, outside such neighbourhood propagation of the route requests is deliberately delayed to

[1]Department of Computing Science, University of Glasgow, Glasgow, UK G12 8QQ, Email: rodhaan@dcs.gla.ac.uk. Sponsored by King Saud University, Riyadh, Saudi Arabia.

[2] Department of Computing Science, University of Glasgow, Glasgow, UK G12 8QQ, Email: lewis@dcs.gla.ac.uk.

[3]Department of Electrical & Computer Engineering, Sultan Qaboos University, Al-Khodh, 123, Muscat, Oman, Email: mok@squ.edu.om.

provide the chasing mechanism with an opportunity to stop the propagation of the fulfilled route request to minimise network congestion.

The rest of the paper is organised as follows: Section 2 presents the related work while section 3 presents our newly proposed algorithm; evaluates the performance, describes the simulation environment and observation after conducting a comparative study with AODV [21], Limited Broadcasting [10], and Blocking-ERS [19]. Finally, Section 4 concludes this study.

II. Related Work

An algorithm for route discovery optimization, Limited Broadcasting (L-B for short), that eliminates the need for historical or location information has been proposed in [10]. It achieves this by employing chase packets approach. Chase packet is a control packet that is broadcasted after finding the desired route to stop the fulfilled route request from further propagation. The algorithm broadcasts 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 using virtual channels. The main purpose of broadcasting a 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 always favours the chase over the route requests. For instance, if there is a route request ready to be broadcasted in any node it will be given a ¼ of the time to be send. Doing so would delay all route requests for the source node in hand as well as other source nodes that might be trying to broadcast route requests yet getting low priority due to time division between route requests, chase, and route reply packets.

In the L-B algorithm, 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 Limited-hop Broadcast Algorithm (LHBA) [25]. 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 finder of the route to K hop neighbours to free this part of the network from the fulfilled route request. It 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.

Blocking-ERS (B-ERS for short) [18], [19] is another algorithm that aims to improve the energy consumption. This algorithm uses chase packets to optimize the route request process. This approach modifies the Expanding Ring Search (ERS) that can be used to rebroadcast the route request in AODV or DSR. ERS works by searching successively larger areas to improve the process of broadcasting the route request utilizing the route cache stored within intermediate nodes. B-ERS works by introducing a delay equals to 2hop-count at each node from the start. After this delay, the intermediate node may receive a chase packet called “stop_instruction” from the source node. Stop_instruction is broadcasted to cover the current ring only. In case of receiving the chase packet, the intermediate node will discard both the route request and the chase packet else it will rebroadcast the route request to cover a larger ring where rings are increased sequentially.

III. The New Algorithm

In this study, we propose a new route discovery algorithm with chase packets that uses a dual-tier approach [3] and utilizes chase packets for networks that exhibit traffic locality. Earlier version of this idea was published in [4].

Due to the scarce resources in MANETs, we kept our algorithm simple so we avoid collecting or manipulating large amount of data. Our algorithm works by establishing a neighbourhood region that includes most of the likely destinations for each source node. Neighbourhood, τ_1, and beyond-neighbourhood, τ_2, regions form a disjoint dual-tier partition of the network where $\tau_1 \cap \tau_2 = \emptyset$. The idea is to process the route requests quickly 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 the route reply, it broadcasts a chase packet as an attempt to terminate the propagation of the fulfilled route request since the chase packet travels faster than the route request within the beyond-neighbourhood region.

Each node has a locality parameter LP where $LP \in \mathbb{N}^*$ which corresponds to the current estimated depth of its neighbourhood calculated as in Equation 1. To illustrate the neighbourhood adjustment process, Let $s \in N$ be a source node in a network of N nodes and define the function, $h_s : N \to \mathbb{Z}^+ \cup \{0\}$ where $h_s(u)$ is the hop count between s and some other node $u \in N$ and $h_s(s) = 0$. Also, let us consider the source node *s* after it has completed its start-up phase. The source node *s* communicates with the node *f* that is $h_s(f)$ hops away. The finder of a route *f* is the first node that finds the route in its cache table whether it is the destination or an intermediate node. The number of previous route requests that already been sent by *s* is denoted by *y* where the source node needs to store the number of its previous route requests (y). Equation 1 is used by *s* to update its *LP*:

$$LP_{new} = \begin{cases} \lceil \alpha \times LP_{old} + (1-\alpha) \times h_s(d) \rceil & h_s(d) \geq LP_{old} \\ \lfloor \alpha \times LP_{old} + (1-\alpha) \times h_s(d) \rfloor & h_s(d) < LP_{old} \end{cases} \quad \alpha = \frac{y}{(y+1)} \quad (1)$$

In TLRDA-C algorithm, source node broadcasts route requests after adding the value of its *LP* to the route request packet so intermediate nodes can decide if the route request is within its source node’s neighbourhood or not, to avoid ambiguity we will use LP_r to refer to the *LP* stored in the route request. Moreover, the delay added in the beyond-neighbourhood region in TLRDA-C is equal to *2LPr*.

A node *x* is considered to be part of the neighbourhood of a source node *s* if $h_s(x) \leq LP$. Considering a source node *s*, any node $v \in \tau_1$ satisfies the condition $h_s(v) \leq LP_r$ and any node $u \in \tau_2$ should satisfy the condition $h_s(u) > LP_r$.

If a route reply is not received within an estimated period of time called NETwork Traversal Time (*NETTT*), the source node will try again to discover the route. TLRDA-C calculates this estimated time as in (2) assuming the worst-case scenario where Node Traversal Time (*NTT*) follows the on-demand routing algorithm used and D is the network diameter:

$$NETTT = 2\{(LP * NTT) + (D - LP)(NTT + 2LP)\} \quad (2)$$

Fig. 1 shows the steps that will be performed by each node upon receiving a route request while Fig. 2 shows the performed steps when a route reply received and Fig. 3 shows the algorithm that will be performed upon receiving the chase message.

The chase packet format includes the route request ID and the source IP address to uniquely identify a particular route request that is associated with the chase packet. It also includes a broadcast ID that is used with the source IP address to identify a redundant chase packet.

The source node 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 where broadcast storm problem of chase packets may congest the network. The price to pay here is an extra amount of delay equal to the time needed to travel $h_s(f)$ distance is added. On the other hand, in the case of multi-path routing protocols such as on-demand multipath distance vector protocol (AOMDV) [16] and multi-path dynamic source routing protocol (MP-DSR) [15] 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 packet as soon as it receives enough route replies.

Each node needs a warm-up period upon joining the network where this node should broadcast according to the routing algorithm in used until the neighbourhood is reasonably established. Moreover, our algorithm is assuming that $2h_s(f)$ is not located near the boundary of τ_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.

A. Performance Analysis

Simulations have been conducted to evaluate our algorithm, TLRDA-C, against AODV, L-B, and B-ERS algorithms using NS2 simulator version 2.29 [9]. TLRDA-C, L-B, and B-ERS were implemented as a modification to the existing AODV implementation.

```
Steps preformed by each node upon receiving a route request in TLRDA-
1:   If route request is a duplicate
2:       Discard the route request
3:   Else
4:       If chase packet has been received then
5:           Store route request information
6:           Discard the route request
7:       Else
8:           If hop_count > $LP_r$ then
9:               Wait ($2LP_r$) unit time
10:          End if
11:          Process the route request
12:      End if
13:  End if
```

Fig. 1. Route request processing at each node for TLRDA-C.

```
Steps performed by each node upon receiving a reply packet in TLRDA-
1:   If current node is the sender then
2:       Create a chase packet
3:       Broadcast the chase packet
4:       Start transmitting the data
5:   End if
6:   Process the route reply
```

Fig. 2. Route reply processing at each node for TLRDA-C.

```
Steps performed by each node upon receiving the chase packet in TLRDA-C
1:   If the chase packet is a duplicate then
2:       Discard it.
3:   Else
4:       Store chase information
5:       If the route request received then
6:           If the route request broadcasted then
7:               Broadcast the chase packet.
8:           Else
9:               Discard both packets.
10:          End if
11:      Else
12:          Discard the chase packet.
13:      End if
14:  End if
```

Fig. 3. Chase packets processing at each node for TLRDA-C.

Modelling movements is not obvious in MANETs. In order to simulate a new protocol, it is necessary to use a mobility model that reasonably represents the movements of a typical node [7]. With the lack of real traces, an accurate synthetic mobility models should be chosen carefully to determine whether the proposed protocol would be useful when implemented in practice.

In MANETs, the entity mobility models typically represent nodes whose movements are completely independent of each other in un-cooperative fashion, e.g. the Random Way Point (RWP) model [14]. In contrast, 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 [6], [12].

B. Simulation Environment

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 shown below and a 95% confidence interval is produced (shown as standard error bars in the relevant figures).

Table 1 provides a summary of the chosen simulation parameter values.

TABLE 1
SIMULATION PARAMETERS

Parameter	Value
Transmission range	100m
Topology size	1000x1000m
Simulation time	900s
Packet size	512bytes
Packet rate	4pkt/s
Data sessions	5,10, …,35
Traffic type	CBR(UDP)
Routing protocol	AODV
Number of Nodes	20,30,..,100
Number of runs/point	30
Antenna type	Omni Antenna
MAC protocol	IEEE 802.11
Maximum speed	2,5,7,10,13,15m/s
Minimum speed	1m/s
Mobility Model	RPGM model
SDR, ADR	0.5
Propagation model	Two-Ray Ground model

A traffic generator was used to simulate constant bit rate

(CBR) with payload of 512 bytes. Moreover, data sessions between different source and destination pairs in groups of ten nodes were simulated to simulate traffic in a network that exhibit traffic locality. Data packets are transmitted at a rate of four packets per second, assuming nodes are identical so the transmission range is 100m to approximately simulate networks with a maximum hop count of 10 hops, links are bidirectional, and mobile nodes operate in a flat arena.

The RPGM mobility generator [5] was used to generate mobility scenarios for 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 15m/s. Each group contains 10 nodes with Speed Deviation Ratio (*SDR*) and Angle Deviation Ratio (*ADR*) = 0.5. The minimum speed is 1 with 50s as pause time.

In our simulation, we concentrate on three major parameters in three different analyses by varying one parameter while keeping the other two constant as explained blow:

<u>Network size analysis</u>: 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 (small size network) to 100 (moderate size network) with traffic load of 10 data sessions and a maximum speed of 15m/s.

<u>Traffic load analysis</u>: Traffic load of sizes 5, 10… 35 data sessions were used in some simulations with network of size seventy nodes and maximum speed of 15m/s. A reasonably incremented amount of traffic was used to test our algorithm meanwhile avoiding saturation. We used 5 communication sessions as light traffic and 35 communication sessions as heavy traffic.

<u>Mobility analysis</u>: The value of the maximum speed can be 2, 5, 7, 10, 13, or 15m/s with networks of 70 nodes and traffic load of 10 data sessions. Slow speed networks have maximum speed of 2m/s while fast speed networks have 15m/s.

The comparison metrics include:

- Network coverage (nodes) is the number of receiving nodes per route request where the node is counted as one if it receives one or more copies of the same route request. This metric provides an indication of the success rate of the chasing mechanism where each algorithm is compared to AODV because it uses simple flooding which gives complete coverage.
- End-to-end delay (ms) is the total delay for the application data packet while transmitted from source to destination plus the route discovery time which is the round trip time from sending a route request until receiving the route reply.
- Route request latency (ms) is the average of delays per hop among all route requests in a single run. Latency of one route request is the average delay experienced by the route request between intermediate nodes at each hop from the time it was sent by a source node until the time it was discarded.
- Packet loss (packets) is the number of dropped packets in a single run.
- Routing overhead (packets) is measured by the number of received route requests plus the number of received chase packets, if there is any, in the whole network.

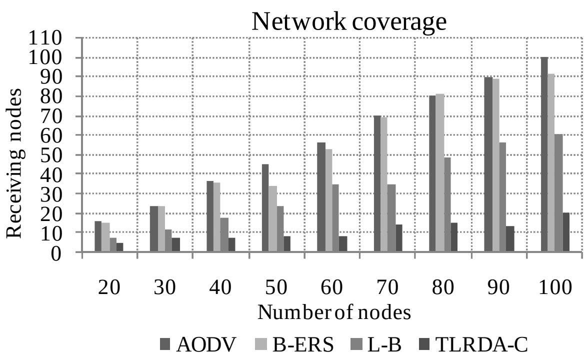

Fig. 4. Network coverage versus network size with 10 communication session and 15m/s as maximum speed.

C. *Simulation Analysis*

Considering *network size analysis*, Figs. 4-8 display the performance results of comparing TLRDA-C against AODV, B-ERS, and L-B algorithms using networks with different sizes. The number of nodes is multiple of 10 starting from 20 until 100 with a minimum speed of 1m/s and a maximum speed of 15m/s. The number of communication sessions is ten.

Fig.4 shows that TLRDA-C achieves a better success rate for the catching process than the other algorithms: AODV, B-ERS, and L-B. The rate of success for the catching process is determined by the amount of coverage. The optimal success rate is when the coverage equals to $h_s(f)$ but this cannot be obtained efficiently without the use of external resources. When the network is covered completely by a route request, while the algorithm uses the chasing technique, the rate of the success in the chasing process is zero; i.e. less coverage means higher success rate. In AODV, where simple flooding is used, there are no chase packets so the network is almost covered by default where the coverage is 100% most of the time. In B-ERS, the coverage is nearly equal to AODV because the discard of the chase packet before catching the associated route request makes the fulfilled route requests cover the whole network most of the time. B-ERS's coverage is less than that of AODV by 6% in small size and 1% in moderate size network. This little improvement might be due to low catching or packet loss especially when contention is high as in moderate size networks. L-B succeeds in the catching process to some extent and its coverage is less than that of AODV by 55% in small size and 37% in moderate size network due to the small amount of delay added to route requests compared to B-ERS and TLRDA-C which makes the fulfilled route requests propagates further in the network. TLRDA-C achieves the best success rate among all the four algorithms. Its coverage is less by 69% in small size network and by 85% in moderate size network compared to AODV and less than B-ERS by 67% to 85% while it is less by 31% to 76% compared to L-B coverage.

Fig. 5 explores the end-to-end delay for TLRDA-C, L-B, B-ERS, and AODV. TLRDA-C has low average of route discovery time which reduces the end-to-end delay metric because the discovery time is included in the end-to-end delay. Thus, it reduces the average end-to-end delay more than L-B, B-ERS, and AODV. The end-to-end delay increases with the network size for all four algorithms because when the network size increases the hop count of a path increases which in turn increases the discovery time.

TLRDA-C achieves a lower end-to-end delay due to the faster propagation of the route request within its neighbourhood region remembering that TLRDA-C broadcast with less contention. The reason behind the increment in the average end-to-end delay in both B-ERS and L-B is the delaying of route requests from start and before discovering the required route. TLRDA-C's improvement of the average end-to-end delay ranges from 58% to 67% over AODV, between 62% and 70% over B-ERS, and from 51% to 68% over L-B. If the route discovery process is fast, the reply will reach the source node earlier which gives the source node the opportunity to broadcast the chase packet and the application data earlier. Application data is stored within the source node until a valid route is found. This affects the end-to-end delay so if the discovery is a quick process, the data are stored for less time which reduces network latency.

Fig. 6 shows the superiority of TLRDA-C in minimising the average of route request latency. The average route request latency of TLRDA-C is reduced due to the better success rate in the catching process for TLRDA-C as shown above in Fig. 4.

The route requests in B-ERS reside in the network more than AODV; the reasons behind this phenomenon are: i) the large delay added always to route requests, ii) low success rate of catching fulfilled route requests. TLRDA-C improves the average of route request latency by 46% to 57% over AODV, 64% to 83% over B-ERS, and 35% to 50% over L-B.

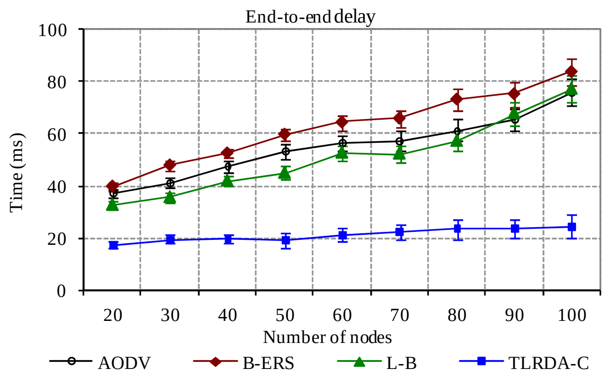

Fig. 5. End-to-end delay versus network size with 10 communication session and 15m/s as maximum speed.

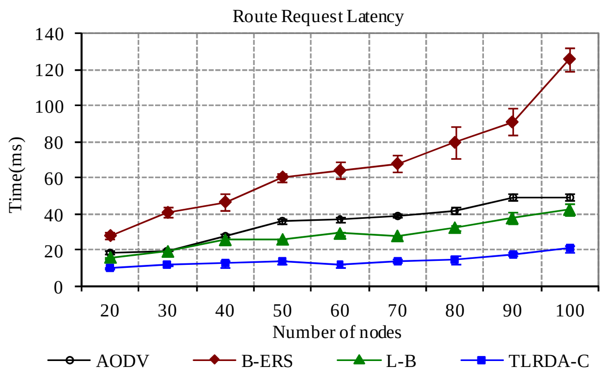

Fig. 6. Route request latency versus network size with 10 communication session and 15m/s as maximum speed.

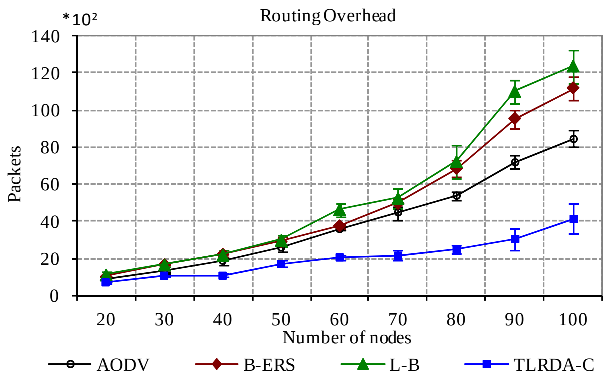

Fig. 7. Network overhead versus network size with 10 communication session and 15m/s as maximum speed.

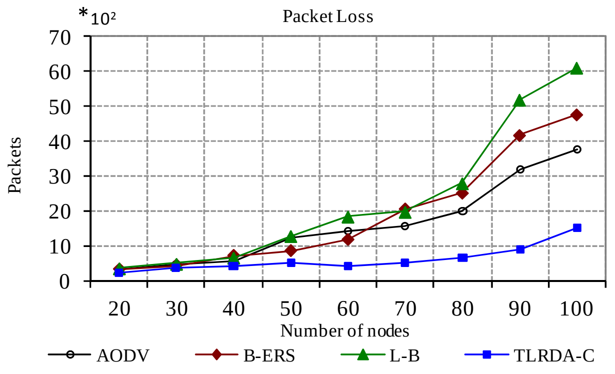

Fig. 8. Packet loss versus network size with 10 communication session and 15m/s as maximum speed.

TLRDA-C sends the chase packet earlier without adding any extra delay to the chase packet propagation which makes the chasing process quicker than L-B. Since TLRDA-C achieves the lowest end-to-end delay among all four algorithms, the chase packets starts earlier in TLRDA-C which give chase packets a better opportunity to succeed.

Fig. 7 shows the routing overhead for all four algorithms where TLRDA-C reduces routing overhead more than AODV, B-ERS, and L-B. This improvement increases in moderate network which intern improves power consumption and gives better bandwidth utilisation. TLRDA-C's improvement of the routing overhead is 17% to 51% over AODV, 32% to 63% over B-ERS, and 38% to 72% over L-B. The number of received chase packets in B-ERS is less than TLRDA-C by 3% to 20% but because of higher number of received route requests in B-ERS, routing overhead is lower in TLRDA-C.

TLRDA-C reduces packet loss in the whole network compared to AODV, B-ERS, and L-B as shown below in Fig. 8 because the network in TLRDA-C is less congested which saves more packets especially with moderate size network environment. TLRDA-C improves packet loss by 21% to 59% over AODV, 13% and 68% over B-ERS, and 22% to 75% over L-B. Simple flooding is very costly in moderate size networks in terms of overhead because increasing number of nodes will increase the number of hops for any single packet. This increases the channel contention and congests the network leading to increment in packet loss. However, in TLRDA-C the success of freeing the network from unwanted route requests saves more important packets from being dropped while needed. Therefore, the network performance is improved for TLRDA-C by reducing latency and overhead due to the higher success rate of the catching process. So the quick broadcasting in a less congested environment such as

TLRDA-C improves the network performance in terms of latency and overhead. This improvement increases with moderate networks as revealed from Figs. 4 to 8.

For *traffic load analysis*, Figs. 9-13 display the results of running our algorithm, TLRDA-C, against AODV, L-B, and B-ERS for 900 seconds of simulation time using networks of size 70 nodes with a random speed range between 1 and 15m/s. The amount of traffic ranged from 5 to 35 sessions incremented by five.

Fig. 9 demonstrates how much the network is covered. From this figure we can compare the success rate of the chasing technique in stopping the fulfilled route requests in all the algorithms that use chase packets technique. AODV covers the network completely as expected from simple flooding but when the network is injected with heavy traffic as in 35 communication sessions the number of receiving nodes was almost double the network size which means that some of the route requests were reinitiated more than once by the source node due to the high congestion and contention.

B-ERS success rate is lower than both TLRDA-C and L-B. TLRDA-C has the best success rate among all four algorithms. TLRDA-C's coverage is less by 90% in light traffic and 94% in heavy traffic compared to AODV and less than B-ERS by 83% to 84% while it is less by 53% to 60% compared to L-B coverage. So the success rate of TLRDA-C improves more with heavy traffic load.

Fig. 10 shows that TLRDA-C improves the end-to-end delay over AODV, B-ERS, and L-B. This improvement due to the quicker broadcasting while the required route is not discovered yet compared to B-ERS and L-B. TLRDA-C works in a less congested environment compared to AODV. TLRDA-C's improvement ranges from 55% to 65% over AODV, 59% to 63% over B-ERS, and 51% to 56% over L-B. When the traffic load increases, channel contention increases which increases the end-to-end delay in all four algorithms.

Fig. 11 reveals the superiority of TLRDA-C among the four algorithms in terms of the average of route request latency because it achieves higher success rate in the catching process and avoid delaying route request before discovering the required route. TLRDA-C improves the average of route request latency by 53% to 67% over AODV, 67% and 72% over B-ERS, and 36% to 50% over L-B.

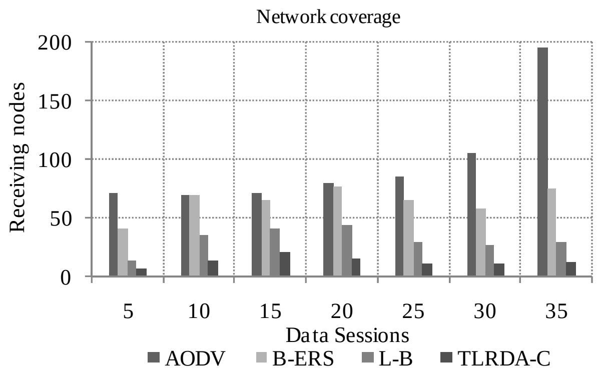

Fig. 9. Network coverage versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

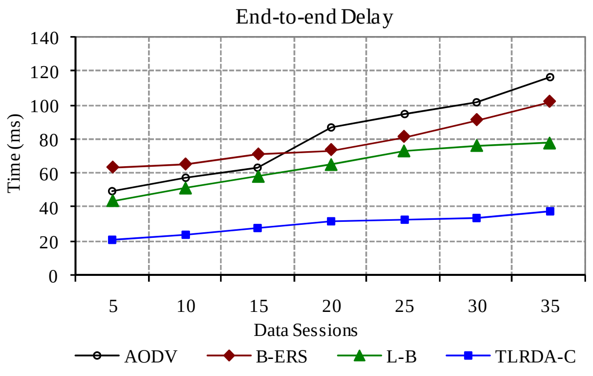

Fig. 10. End-to-end delay versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

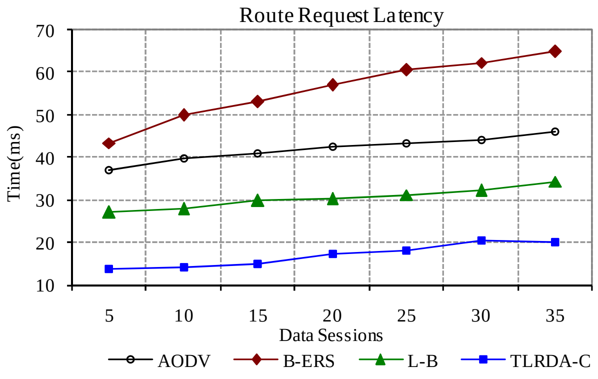

Fig. 11. Route request lifetime versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Fig. 12 expresses the routing overhead for all four algorithms and shows that TLRDA-C achieves lower routing overhead than AODV, B-ERS, and L-B. Such improvement increases with the increment of traffic load which should improve both power consumption and bandwidth utilisation. The improvement of the routing overhead in TLRDA-C is 51% to 81% over AODV, 1% to 60% over B-ERS, and 55% to 61% over L-B.

TLRDA-C incurs less packet loss in the whole network compared to AODV, B-ERS, and L-B as shown below in Fig. 13 because the network in TLRDA-C is less congested. The packet loss is increased with the increment of traffic load for all four algorithms. However, TLRDA-C improves packet loss by 32% to 67% over AODV, 22% to 68% over B-ERS, and 40% to 80% over L-B.

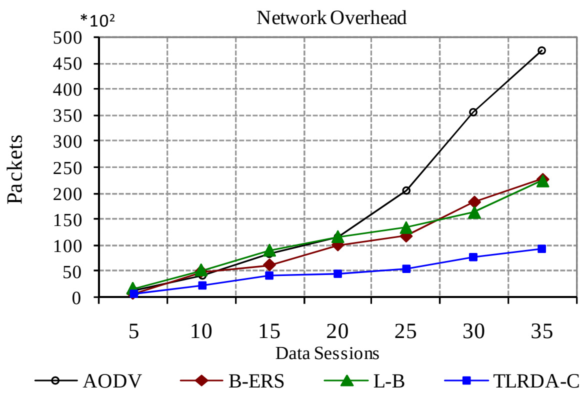

Fig. 12. Routing overhead versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

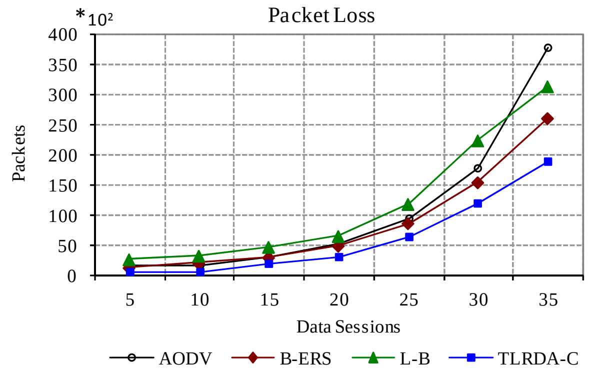

Fig. 13. Packet loss versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

For *mobility analysis*, Figs. 14-18 were extracted from simulation runs while increasing the maximum speed from 2 to 15m/s. The number of nodes was 70 and data sessions were 10.

Fig. 14 demonstrates network coverage as an indicator of the success rate of the catching process. AODV covers the network almost completely especially with fast networks because of the simple flooding. TLRDA-C has the best success rate among the four algorithms. TLRDA-C's coverage is less by 79% in slow networks and 86% in fast networks compared to AODV and less than B-ERS by 79% to 85% while it is less by 56% to 69% compared to L-B coverage. The average of end-to-end delay increases with fast networks regardless of the algorithm used. However, TLRDA-C offers better end-to-end delay over AODV, B-ERS, and L-B as shown in Fig. 15. This improvement is due to less congested environment among the four algorithms and/or quick broadcasting within the neighbourhood region compared to B-ERS and L-B. TLRDA-C's improvement ranges from 54% to 61% over AODV, 63% and 66% over B-ERS, and 41% to 52% over L-B.

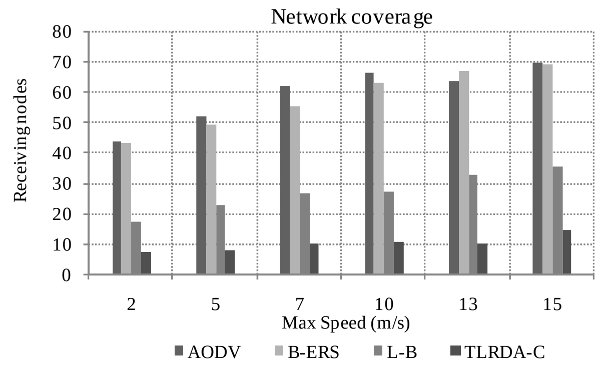

Fig. 14. Network coverage versus maximum speed in networks of 70 nodes and 10 communication sessions.

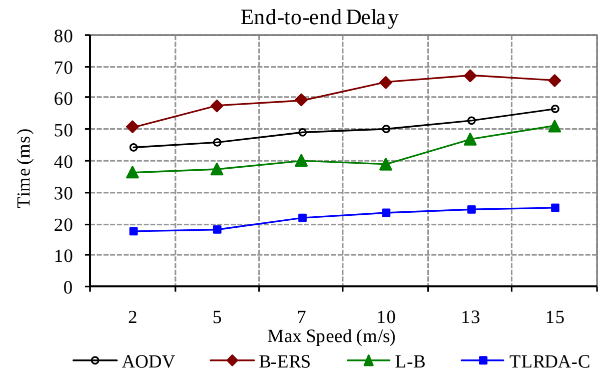

Fig. 15. End-to-end delay versus maximum speed in networks of 70 nodes and 10 communication sessions.

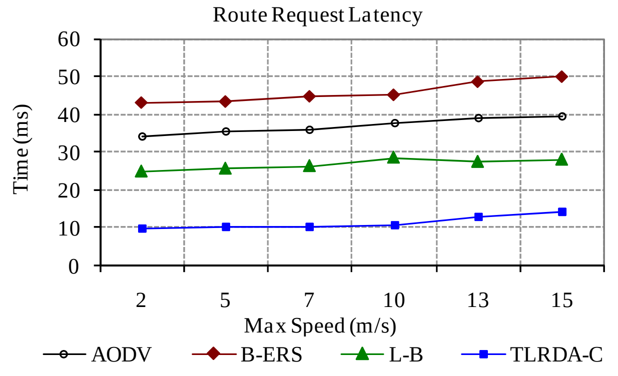

Fig. 16. Route request latency versus maximum speed in networks of 70 nodes and 10 communication sessions.

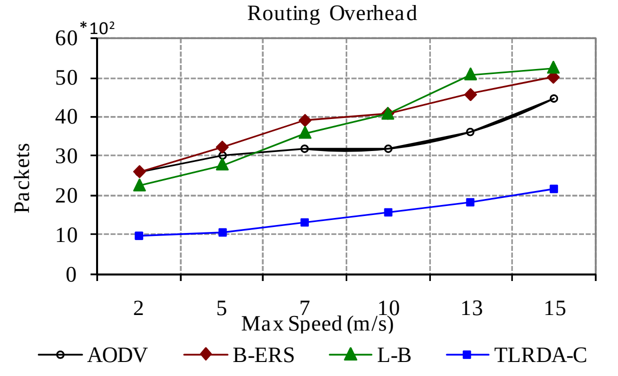

Fig. 17. Routing overhead versus maximum speed in networks of 70 nodes and 10 communication sessions.

Fig. 16 shows a great reduction in route requests latency for TLRDA-C over B-ERS, L-B, and AODV regardless of speed which will improve the network performance. The route request latency increase slightly with the increment of speed due to link breakage regardless of the algorithm used. However, TLRDA-C improves the average of route request latency by 64% to 71% over AODV, 72% to 77% over B-ERS, and 49% to 62% over L-B.

Routing overhead increases with fast networks with all the algorithms used. However, TLRDA-C incurs lower routing overhead than AODV, B-ERS, and L-B as shown in Fig. 17 which should improve both power consumption and bandwidth utilisation as mentioned before. The improvement of the routing overhead in TLRDA-C ranges from 49% to 62% over AODV, 57% to 67% over B-ERS, and 56% to 64% over L-B.

TLRDA-C loses fewer packets compared to AODV, B-ERS, and L-B as shown below in Fig. 18 because the network is less congested. The packet loss is increased with the increment of maximum speed for all four algorithms. However, TLRDA-C improves packet loss by 63% to 78% over AODV, 61% to 79% over B-ERS, and 63% to 82% over L-B.

Looking at the route request latency and end-to-end delay, our algorithm outperforms the three other algorithms. TLRDA-C reduces network overhead and packet loss as well. These two observations are true regardless of network size, traffic load, or maximum speed in all the tested scenarios which make network's improvement in TLRDA-C superior and gets better asymptotically with large networks. A low latency has a generally beneficial effect on the network performance. The attractiveness of TLRDA-C stems from the fact that the data can travel earlier and with less congestion.

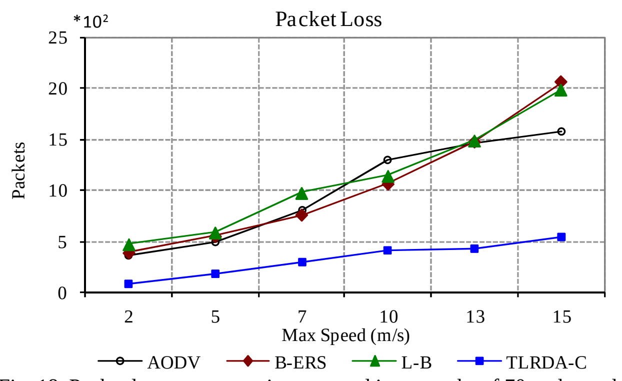

Fig. 18. Packet loss versus maximum speed in networks of 70 nodes and 10 communication sessions.

IV. CONCLUSION

We have developed a new traffic locality oriented route discovery algorithm for MANETs that uses chase packet; we called it TLRDA-C. In this algorithm, each source node maintains a neighbourhood region that includes most of the likely destinations and broadcasts the route requests according to the routing algorithm used. However, the algorithm delays the propagation of the route requests within the beyond-neighbourhood region by to minimise network congestion and provide the chase packets with an opportunity to stop the fulfilled route requests as early as possible. TLRDA-C is adaptive and continuously updates the neighbourhood boundary to provide a better performance. We have provided a detailed performance evaluation using simulation for our new algorithm and compared it with existing algorithms in the literature, AODV, Limited Broadcasting, and Blocking-ERS. The evaluation has shown that our algorithm has a better catching mechanism and gets even better asymptotically with large networks. TLRDA-C achieves lower end-to-end delay due to the reduction in network congestion and channel contention. Network overhead and packet loss were improved as well. TLRDA-C is superior and has a good impact on the network performance regardless of network density, traffic load, or speed. Moreover, the transmission of data packets in TLRDA-C starts earlier due to the reduction in the end-to-end delay.

REFERENCES

[1] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, pp. 1-22, 2004.

[2] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Minet, et al., "Optimized Link State Routing Protocol," *The Internet Engineering Task Force*, IETF,RFC 3626, 2003.

[3] M. Al-Rodhaan, L. Mackenzie, and M. Ould-Khaoua, "A new route discovery algorithm for MANETs with chase packets," *International Journal of Simulation Systems, Science & Technology, Special Issue on: Performance Modelling of Computer Networks, Systems and Services*, vol. 8, pp. 1-12, 2007.

[4] M. Al-Rodhaan, L. Mackenzie, and M. Ould-Khaoua, "A traffic locality oriented route discovery algorithm for MANETs," *Ubiquitous Computing and Communication Journal (UBICC)*, vol. 2, pp. 58-68, 2007.

[5] F. Bai, N. Sadagopan, and A. Helmy, "IMPORTANT: An evaluation framework to study the Impact of Mobility Patterns On RouTing in Ad-hoc NeTworks," University of Southern California, 2005.

[6] F. Bai, N. Sadagopan, B. Krishnamachari, and A. Helmy, "Modeling path duration distributions in MANETs and their impact on reactive routing protocols," *Selected Areas in Communications, IEEE Journal on*, vol. 22, pp. 1357-1373, 2004.

[7] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, pp. 483-502, 2002.

[8] S. R. Das, R. Castaneda, Y. Jiangtao, and R. Sengupta, "Comparative performance evaluation of routing protocols for mobile ad hoc networks," 1998.

[9] K. Fall, "NS Notes and Documentation," in *The VINT Project*, 2000.

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

[11] Z. J. Haas, M. R. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," IETF MANET Working Group, INTERNET-DRAFT, July, 2002.

[12] X. Hong, G. Mario, P. Guangyu, and C. Ching-Chuan, "A group mobility model for ad hoc wireless networks," presented at Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, Seattle, Washington, United States, 1999.

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

[14] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, vol. 353, I. a. Korth, Ed. Norwell, MA: Kluwer Academic Publishers, pp. 153-181, 1996.

[15] R. Leung, L. Jilei, E. Poon, A. L. C. Chan, and L. Baochun, "MP-DSR: a QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks," 2001.

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

[17] S. Murthy and B. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*, Prentice Hall, 2004.

[18] I. Park and P. Ida, "Energy Efficient Expanding Ring Search," presented at the First Asia International Conference on Modelling & Simulation, 2007.

[19] I. Park, J. Kim, and I. Pu, "Blocking Expanding Ring Search Algorithm for efficient energy consumption in mobile ad hoc networks," presented at the Third Annual Conference on Wireless On-demand Network Systems and Services, Les Menuires, France, 2006.

[20] C. Perkins, *Ad hoc networking*, Addison Wesley, 2001.

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

[22] A. Tanenbaum, *Computer Networks*, Pearson Education, 2003.

[23] K. Young-Bae and H. V. Nitin, "Location-aided routing (LAR) in mobile ad hoc networks," *Wireless Networks*, vol. 6, pp. 307-321, 2000.

[24] T. Yu-Chee, N. Sze-Yao, C. Yuh-Shyan, and S. Jang-Ping, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, pp. 153-167, 2002.

[25] H. Zhang and Z. P. Jiang, "On reducing broadcast expenses in ad hoc route discovery," presented at Second International Workshop on Wireless Ad Hoc Networking (WWAN), 2005.