Improvement to Blocking Expanding Ring Search for MANETs
Mznah A. Al-Rodhaan¹, Lewis Mackenzie¹, Mohamed Ould-Khaoua²

¹Department of Computing Science
University of Glasgow
Glasgow, UK G12 8QQ
Email: {rodhaan, lewis}@dcs.gla.ac.uk

²Department of Electrical & Computer Engineering
Sultan Qaboos University
Al-Khodh, 123, Muscat, Oman
Email: mok@squ.edu.om

Abstract
Blocking-ERS is an energy efficient route discovery algorithm for MANETs. It uses chase packets to improve the route request process. In Blocking-ERS, most of the time the fulfilled route request manages to escape from the chase packet with the help of mobility. So its success rate in the catching process is low due to the immature discard of chase packets.

In this report, we are proposing an improved algorithm named Blocking-ERS-Plus to overcome this deficiency in Blocking-ERS by allowing chase packets be broadcasted until the catching is insured to maximise the success rate. We provide detailed performance evaluation using simulation modelling and compare Blocking-ERS-Plus with both Blocking-ERS and AODV. Our findings show that Blocking-ERS-Plus outperforms both Blocking-ERS and AODV by reducing end-to-end delay due to the reduction in network congestion and by improving route request latency, routing overhead, and packet loss due to the higher success rate in the catching process which has positive impact on the network performance.

Keywords: MANETs, On-demand routing protocols, Route Discovery, Chase Packets, Performance analysis, ERS, Blocking-ERS.

1. Introduction
Wireless connectivity was a dream but with the evolution of mobile devices such as notebooks and PDAs, wireless networks appeared and this duly become a reality. Wireless networks might be infrastructure-oriented, such as the access point dependent networks [14] or infrastructure-less such as Mobile Ad hoc NETworks (MANETs) [6, 14, 20]. Some of the dominant initial motivations for MANET technology came from military applications in environments lacking infrastructure. However, while such applications remain important, MANET research has diversified into areas such as disaster relief, sensors networks, and personal area networks [20].

The design of an efficient and reliable routing strategy is a very challenging problem due to the limited resources in MANETs [14]. Many multi-hop routing protocols have been proposed and investigated in the literature as in [2, 10, 12, 17, 18, 21]. MANETs routing protocols are divided into three categories [1, 17]: proactive, reactive, and hybrid. 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 Optimised 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 [7]. In on-demand routing, when a source node

² Mohamed Ould-Khaoua was with Department of Computing Science, University of Glasgow, G12 8RZ UK. He is now with the Department of Electrical & Computer Engineering, Sultan Qaboos University.
needs to send messages to an unseen destination; 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) [12] and Ad Hoc On Demand Distance Vector (AODV) [18]. 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) [10].

On-demand routing algorithms search for the desired route only when needed. Since the nodes have no periodical tasks that covers the network, on-demand routing protocols are known to use low bandwidth and consume less power which makes them appealing for MANETs scenarios [1]. When a source node needs to send packets to an unseen destination, 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 route(s) discovered.

DSR and AODV algorithms 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. In simple flooding, when a source node needs a route to a particular destination it 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 simple flooding leads to redundancy that will highly congest the network and increases the chances of collision: these combined are known as the broadcast storm problem [22]. Moreover, flooding consumes a lot of network resources such as bandwidth and power which can be reduced by stopping the route request as soon as the needed route discovered 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 which will congest the network and waste its resources. The route discovery process can be improved by minimising such overhead and reducing or better stopping the unnecessary propagation of route request packets after discovering the route.

In this report, a new route discovery algorithm, Blocking-ERS-Plus, based on Blocking-ERS [16] will be proposed. It works by broadcasting a control packet called chase packets by the source node after receiving a route reply. Chase packet is a control packet that is broadcasted after finding the desired route to stop the fulfilled route request from further propagation. The chase packet is allowed to cover as much as it needs. Propagation of the route requests is deliberately delayed to provide the chasing mechanism with an opportunity to stop the propagation of the fulfilled route request to minimise network congestion.

The rest of this report is organised as follows: Section 2 presents the related work and section 3 sheds some light on Blocking-ERS. Section 4 presents our newly proposed algorithm, Blocking-ERS-Plus, based on Blocking-ERS, evaluates the performance, conducts a comparative study with Blocking-ERS and AODV [18], and describes the simulation environment and observation. Finally, Section 5 concludes this study.

2. Related Work

Limited Broadcasting [9] aims at improving the route discovery process without the need for historical or location information. It achieves this by employing chase packets approach. The algorithm broadcasts a route request using only \( \frac{1}{4} \) 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. 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 by any node it will be given a \( \frac{1}{4} \) of the time to be sent. Doing so would delay all route requests for the source node in hand as well as other source nodes that might be trying to send route request yet getting low priority due to time division between route requests, chase, and reply packets.

In the Limited Broadcasting 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) [23]; it solves this problem by initiating the chase packets 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. It uses chase packets to optimise 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 is broadcasted to K hop neighbours to free this part of the network from the fulfilled route request.

The broadcast of the route request can be controlled using the TTL field in the route request packet. Expanding Ring Search (ERS) [13, 19] is one of the route request improvements techniques that lower the overhead cost when succeeded. It is presented first for DSR then proposed for AODV. In ERS the source node searches for the target in 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. Blocking-ERS [15, 16] has later been introduced to reduce the energy consumption in ERS. B-ERS uses chase packets to optimise the route request process; more explanation in the following subsection.

3. **Blocking-ERS**

Blocking-ERS (B-ERS for short) [15, 16] improves energy consumption by stopping the fulfilled route requests. It uses chase packets to stop the propagation of route requests after discovering the needed route. It is an improvement of ERS where each new ring starts from the previous ring instead of starting from the source node as in ERS. Blocking-ERS works by introducing a delay equal to 2hop-count*NTT at each ring where rings are increased sequentially and Node Traversal Time (NTT) follows the on-demand routing algorithm used. 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. If no chase packet is received within 2hop-count*NTT time, it will rebroadcast the route request to cover a larger ring. Chase packet is broadcasted up to the ring where the finder of the route resides at maximum to cover only that ring. The source node needs to know how many hops away does the finder of the route reside, thus route reply packet should be extended by one byte to carry the value of the hop-count.

In the presence of mobility, B-ERS suffers from performance degradation due to the immature discard of chase packets where most of the time the fulfilled route request managed to escape with the help of mobility from the associated chase packet.

In this report, we are proposing a new algorithm called Blocking-ERS-Plus (B-ERS+ for short) to overcome this deficiency in B-ERS. It works by continuing to broadcast chase packets until the catching is insured to maximise the success rate of the catching mechanism.

4. **Blocking-ERS-Plus**

Blocking-ERS-Plus (B-ERS+ for short) is an improvement of B-ERS to increase the success rate of the catching process which improves network performance in terms of latency and overhead for MANETs. These two algorithms differ only in the processing of the chase packets. In B-ERS, the chase packet is allowed to be broadcasted only in the ring where the finder of the route resides. However, in B-ERS+ it is allowed to be broadcasted beyond that in an effort to catch the fulfilled route request even after the intermediate node dealing with that route request moves away from its place when the route request received. B-ERS+ uses the chase module, as in Figure 1, for processing chase packets.

Upon receiving the chase packet, the steps in Figure 1 are performed at each node. If the chase packet is a duplicate, it is discarded (line 2). Otherwise, the needed information is stored (line 4) where each node keeps track of all received route requests and chase packets by storing the needed information i.e. their broadcast ID and originator IP address. If the route request and the chase packet information are stored in the same table, a bit flag is needed to distinguish between route requests and chase packet records. If the matching route request is broadcasted already then the chase packet is broadcasted as well (line 7) but if the route request is waiting to be broadcasted then both the route request and its matching chase packet are discarded (line 9). If the route request is not received yet, the chase packet is discarded (line 12) after storing the needed information (line 4).
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. Unlike B-ERS, B-ERS+ does not need to extend the format of the route reply packet because the source node broadcasts the chase packet without restricting it to the ring where the finder of the route resides.

---

**Steps performed by each node upon receiving the chase packet in B-ERS+**

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

---

Figure 1: Steps performed by each node to process the chase packets in B-ERS+.

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.

5. Simulation and Performance Analysis

Simulation runs have been conducted to evaluate B-ERS+ and compare it with B-ERS using ns2 simulator version 2.29 [8]. B-ERS+ and B-ERS were implemented as a modification to the existing AODV implementation. The comparison metrics include:

- **Network coverage** 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.

- **Route request latency** is the average delay per hop among all route requests in a single run.

- **End-to-end delay** is the total delay for the actual transmitted data plus the route discovery time which is the round trip time from sending a route request until receiving the route reply.

- **Packet loss** is the number of dropped packets in a single run.

- **Routing overhead** is measured by the number of received route requests plus number of received chase packets in the whole network.

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 [5]. 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 movements, e.g. the Random Way Point (RWP) model [13]. 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 [4, 11].
5.1 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 until it becomes stable. For each topology, 30 runs were performed. The results of these runs were averaged to produce the graphs shown below, Figures 2-16, and a 95% confidence interval is produced which is shown as standard error bars in the relevant figures. Table 1 provides a summary of the chosen simulation parameter values.

The simulation area is kept constant in all scenarios to study the algorithms performance in small to moderate size environments, since we are interested in knowing the behaviour of the algorithms in both environments. 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. Data packets are transmitted at a rate of four packets per second, assuming nodes are identical so the transmission range is fixed to 100m in all nodes. This approximately simulates networks with a minimum hop count of 10 hops between two border nodes one on opposite sides of the other in a connected network. Links are bidirectional, and mobile nodes operate in a flat arena.

The RPGM mobility generator was used [3] to generate mobility scenarios for all of our simulation runs 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 with speeds between 1 and 15m/s. Speed Deviation Ratio (SDR) and the Angle Deviation Ratio (ADR) are used to control the deviation of the velocity (speed and direction) of group members from their leader’s velocity. Moreover, nodes move randomly within their group where SDR = ADR = 0.5. Each group contains 10 nodes. The minimum speed is 1 with 50s as pause time.

<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>512bytes</td>
</tr>
<tr>
<td>Packet rate</td>
<td>4pkt/s</td>
</tr>
<tr>
<td>Data sessions</td>
<td>5, 10, ... , 35</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>MAC protocol</td>
<td>IEEE 802.11</td>
</tr>
<tr>
<td>Maximum speed</td>
<td>2.5, 7, 10, 13, 15m/s</td>
</tr>
<tr>
<td>Minimum speed</td>
<td>1m/s</td>
</tr>
<tr>
<td>Mobility Model</td>
<td>RPGM model</td>
</tr>
<tr>
<td>SDR, ADR</td>
<td>0.5</td>
</tr>
<tr>
<td>Propagation model</td>
<td>Two-Ray Ground model</td>
</tr>
</tbody>
</table>

5.2 Simulation Analysis

The simulation analysis considers all the three analysis: network size, traffic load, and mobility analyses as stated in Table 2.

<table>
<thead>
<tr>
<th>Analyses</th>
<th>Simulation parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Network Size</td>
</tr>
<tr>
<td>Network Size</td>
<td>20, 30, ... , 100 nodes</td>
</tr>
<tr>
<td>Traffic Load</td>
<td>70 nodes</td>
</tr>
<tr>
<td>Mobility</td>
<td>70 nodes</td>
</tr>
</tbody>
</table>
Network Size Analysis

Figures 2 to 6 display the results of running B-ERS+ against both B-ERS and AODV for 900 seconds using networks with different number of nodes, from 20 to 100 node in an area of 1000m x 1000m with a minimum speed of 1m/s and a maximum speed of 15m/s. The number of data sessions is ten.

Figure 2 shows that the success rate of the catching process has been improved dramatically for B-ERS+ compared to B-ERS and AODV. The rate of success in the catching process is determined by the amount of coverage. The optimal success rate is when the coverage equals the number of hops between source node and the finder of the route but this is impossible to obtain without the use of external resources. When the network is covered completely by a route request while the algorithm uses chasing technique, the rate of the success in the chasing process is zero so 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. However, B-ERS coverage is less than AODV by 6% and 1% in small and moderate size network respectively. This little improvement might be due to a rare catching or pocket loss especially when contention is high as in moderate network. On the other hand, the coverage in B-ERS+ is improved by 76% to 80% compared to AODV and by 74% and 78% over B-ERS.

![Network coverage graph](image)

Figure 2: Network coverage for different number of nodes for networks of 10 data sessions and 15m/s as maximum speed.

Figure 3 explores the end-to-end delay for B-ERS+, B-ERS, and AODV. B-ERS+ reduces the average end-to-end delay more than both B-ERS and AODV because the network in B-ERS+ is less congested. In fact, B-ERS+ improves end-to-end delay by 9% to 25% over B-ERS and by 4% to 16% over AODV.

Figure 4 shows that the average route request latency of B-ERS+ is less than that of B-ERS which means that the catching process was more successful in B-ERS+. It improves the average route request delay by 44% to 60% over B-ERS.

Figure 5 shows that B-ERS+ reduces the number of received route requests but increases the number of chase packets received compared to B-ERS. Nevertheless, the routing overhead in B-ERS+ is improved by 45% to 55% over B-ERS. B-ERS+ improves the routing overhead by 33% to 40% over AODV. This improvement increases with the increment of network size leading to an improvement in power consumption and bandwidth utilisation as well. The overhead increases with increment of network density regardless of the algorithm used because the average number of route request received may increase more when the route request spreads deeper in the network when it is broadcasted in all direction and may reach more nodes each time it propagates further.

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 which increases line contention and congest
the network leading to increment in packet loss. Figure 6 shows that B-ERS+ reduces the packet loss over
B-ERS by 22% to 67% and by 23% to 58% over AODV which means that B-ERS+ improves network
congestion. B-ERS+ improvement increases with the increment of network size.

Figure 3: End-to-end delay for different number of nodes for networks of 10 data sessions and 15m/s as maximum speed.

Figure 4: Average route request latency for different number of nodes for networks of 10 sessions and 15m/s as maximum speed.

Figure 5: Routing overhead for different number of nodes for networks of 10 sessions and 15m/s as maximum speed.
Therefore, the network performance is improved for B-ERS+ compared to B-ERS by reducing latency and overhead due to the higher success rate of the catching process in B-ERS+. This improvement increases with moderate networks.

Traffic Analysis

Figures 7 to 11 display the results of running B-ERS+ against both AODV and B-ERS for 900 seconds using networks of size 70 nodes in an area of 1000m x 1000m with a random speed between 1m/s and 15m/s. The amount of traffic ranges from 5 to 35 data sessions incremented by five.

Figure 7 demonstrates the network coverage for AODV, B-ERS, and B-ERS+. AODV covers the network completely as expected from simple flooding but when the network is injected with heavy traffic, as in 35 data sessions, the number of receiving nodes is almost double the network size which means that some of the route requests are reinitiated more than once by the source node due to the high congestion and contention. At 30 data sessions, B-ERS succeeded in some of the chasing process but still its success rate is lower than B-ERS+. B-ERS+ improves the success rate over B-ERS dramatically by 85% to 87%. B-ERS+ catches more route requests compared to B-ERS because it broadcasts chase packets to cover a larger area enabling each chase packet to reach the associated route request. B-ERS+ improves the success rate over AODV by 84% to 95%.

Figure 8 shows that B-ERS+ improves the end-to-end delay over both B-ERS and AODV because B-ERS+ frees the network from unneeded route requests which reduces the network congestion. This is the case despite the fact that both B-ERS AND B-ERS+ algorithms add the same amount of delay to route requests at each node. This improvement is 39% to 44% over B-ERS and 26% to 42% over AODV.

Figure 9 reveals the superiority of B-ERS+ over AODV and B-ERS in terms of the average of route request latency because of the higher success rate in the catching process which reduces network congestion. The route request latency increases with traffic load due to the increase of the number of packets in the network which adds more contention and may result in more collision. B-ERS+ improves route request latency over B-ERS by 30% to 42% and 17% to 24% over AODV.

Figure 10 depicts the routing overhead for all three algorithms. B-ERS+ incurs lower routing overhead than B-ERS due to the higher success rate of the catching process. B-ERS+ improvement increases with traffic load over B-ERS and reaches 62% in heavy load and reaches 82% over AODV.

B-ERS+ incurs less packet loss in the whole network compared to B-ERS as shown in Figure 11 because the network in B-ERS+ is less congested. The packet loss increases with the increment of traffic load for all three algorithms. B-ERS+ improves packet loss by 29% to 71% over B-ERS and by 38% to 70% over AODV.
Figure 7: Network coverage for different amount of traffic load in a network of 70 nodes and 15m/s as maximum speed.

Figure 8: End-to-end delay for different amount of traffic load in a network of 70 nodes and 15m/s as maximum speed.

Figure 9: Route request delay for different amount of traffic load in a network of 70 nodes and 15m/s as maximum speed.
Mobility Analysis

Figures 12 to 16 were extracted from the simulation runs of AODV, B-ERS, and B-ERS+ algorithms for 900 seconds using networks of size 70 nodes in an area of 1000m x 1000m using six maximum speeds. The maximum speed takes one of the following values: 2, 5, 7, 10, 13, and 15m/s. The data sessions was fixed at 10.

Figure 12 demonstrates network coverage as an indicator of the success rate of the catching process. It shows that B-ERS+ improves the success rate over B-ERS regardless of speed by 80% to 86%.

Furthermore, B-ERS+ improves end-to-end delay over AODV and B-ERS as shown in Figure 13. This improvement is due to broadcasting in a less congested environment. B-ERS+ improves end-to-end delay by 41% to 52% over B-ERS and by 31% to 42% over AODV.

Moreover, route requests latency is lower for B-ERS+ than B-ERS and AODV regardless of speed which will improve the network performance as shown in Figure 14. As mentioned earlier, this improvement is due to the higher success rate of B-ERS+ in the catching process. B-ERS+ improves route request latency by 26% to 42% over B-ERS because when the route request propagates further in the network the hop count increases which will increase the amount of delay imposed. Moreover, B-ERS+ improves route request latency by 7% to 22% over AODV.

B-ERS+ incurs lower routing overhead than both B-ERS and AODV as shown in Figure 15. Routing overhead increases with the increment of speed regardless of the algorithm used. The improvement of the
routing overhead in B-ERS+ is 56% to 72% over B-ERS and by 44% to 60% over AODV. Moreover, B-ERS+ loses fewer packets compared to AODV and B-ERS as shown below in Figure 16 because the network is less congested. The packet loss is increased with the increment of maximum speed for all three algorithms. B-ERS+ improves packet loss by 66% to 86% over B-ERS while the improvement compared to AODV is 65% to 86%.

![Network coverage graph](image)

**Figure 12:** Network coverage for different maximum speeds in networks of 70 nodes and 10 data sessions.

![End-to-end Delay graph](image)

**Figure 13:** End-to-end delay for different maximum speeds in networks of 70 nodes and 10 data sessions.
Finally, B-ERS+ outperforms both AODV and B-ERS regardless of network size, traffic load, or maximum speed in all metrics used.

5.3 Simulation Results

The simulation analysis reveals clearly that B-ERS achieves low success rate of the catching process; for this reason B-ERS has higher route request latency and incurs higher routing overhead more than AODV for all the scenarios performed in our simulation runs. Therefore, B-ERS+ broadcasts chase packets until the catching is insured thus maximises the success rate compared to B-ERS. Simulation experiments and analysis were conducted to study the performance of B-ERS+ and compare its performance with that of both AODV and B-ERS while concentrating on the three-performance cases: network size, traffic load, and mobility. Almost the same behaviours were demonstrated by all the three analyses for all metrics used in this study. B-ERS+ outperforms both B-ERS and AODV by reducing end-to-end delay and route request latency which improves network latency. Moreover, B-ERS+ incurs lower routing overhead and reduces packet loss compared to B-ERS and AODV. In B-ERS+, the network is less congested due to the higher success rate in the catching process leading to network performance improvement from latency, congestion level and overhead prospective.

6. Conclusions

B-ERS has a low success rate where most of the time the fulfilled route request manages to escape with the help of mobility from its associated chase packet. Its success rate in the catching process is low due to
immature discard of chase packets. To that end, we proposed a new route discovery algorithm that we
called B-ERS+ as an improvement of the original B-ERS. It works by continuing to broadcast chase
packets until the catching process is insured to maximise the success rate of the catching mechanism.

In this report, we studied the performance of B-ERS, AODV, and the newly developed B-ERS+ using
simulation. B-ERS+ outperforms both B-ERS and AODV algorithms by reducing end-to-end delay due to
the reduction in network congestion and by improving route request latency, routing overhead, packet loss
due to the higher rate of success in the catching process. B-ERS+ succeeded in achieving high success rate
in the catching process which has positive impact on the performance in term of lowering congestion and
reducing routing overhead which saves network resources such as power and bandwidth.

7. References