Enhanced Blocking Expanding Ring Search in Mobile Ad Hoc Networks

Ida M Pu
Goldsmiths, University of London
London SE14 6NW, UK
Email: i.pu@gold.ac.uk

Yuji Shen
University of Birmingham
Birmingham B15 2TT, UK
Email: y.shen.1@bham.ac.uk

***Abstract*—We introduce BERS*, an enhanced Blocking Expanding Ring Search (BERS) protocol for route discovery in mobile ad hoc networks (MANETs). BERS is an energy efficient alternative that was developed recently based on the Expanding Ring Search (ERS). ERS is widely applied in reactive routing protocols such as DSR and AODV. BERS* is a faster and more energy-time efficient version of BERS. It reduces the route discovery latency of BERS by nearly half while maintainning a similar level of energy saving. Our results show that, among the three protocol schemes (BERS*, BERS and ERS), BERS* incurs the least search latency when the hop number of the route nodes is greater than 3, and has achieved the best performance in terms of energy-time efficiency when the hop number of the route nodes is greater than 7. We have also discovered the conditions that allow collective optimisation of BERS* and ERS.**

***Index Terms*—algorithm, efficiency, energy-time, latency, MANET, BERS*, BERS, ERS**

I. Introduction

Energy efficiency is an important issue in Mobile Ad Hoc Networks (MANETs). Nodes in MANETs rely on limited power and computation resources, yet are required to cooperate in all sorts of fundamental network activities including routing. Routing can consume a relatively large amount of limited resources due to the dynamic and cooperative nature of MANETs. To reduce the overhead size, reactive routing protocols have been proposed and become popular such as DSR [1] and AODV [2]. One main characteristics of the reactive routing protocols is that they act on demands only, for example, a route is to be established only when a source node requires data packets to be sent to a destination.

Reactive routing protocols in MANETs are often supported by an Expanding Ring Search (ERS) [1], [2], [3]. ERS is a controlled flooding technique. To avoid flooding in a larger area than necessary, an ascending incremental TTL sequence is often used to define a series of maximum flooding radius. This may, however, lead to waste of energy in a number of failed search attempts by flooding in smaller areas before a successful search in the final round of the floodings.

Blocking Expanding Ring Search (BERS) [4] is an energy efficient alternative that was developed recently based on ERS. It identified the energy inefficiency of ERS and has achieved a substantial amount of energy saving. Although BERS is efficient in terms of energy saving for a route discovery process, the increased latency restricts its applications in certain dynamic and time-constraint environment where low latency is also important. In this study, we have addressed the weakness in time inefficiency of BERS and developed BERS*, an enhanced BERS, to reduce the route discovery latency while maintainning a similar level of energy saving. The BERS* can reduce the latency by nearly half that of BERS, and has shown a significant improvement to overall performance in terms of energy-time efficiency [5].

In the rest of the paper, we briefly describe the related work in Sections II, including the TTL-based ERS and BERS. We summarise the results on energy consumption and latency of ERS in Sections II-A, and outline the energy and time mechanisms of BERS in Sections II-B. We then introduce BERS*, a new energy-time efficient approach in Sections III, providing the analysis of its searching heuristics and algorithms, and conductting the performance evaluation and a comparison between ERS, BERS and BERS*. In Section IV, we discuss our simulation settings and analytical results. Finally, in Section V, we conclude our results and findings.

II. Related work

A. Expanding ring search (ERS)

The expanding ring search is an effective way of finding a route between two distinct nodes (S, D) in a MANET, where S represents a source node; and D represents a destination, or a *route node* that can offer the route information to the destination. There may be more than one route between S and D, and the ERS aims to find one with least effort.

ERS conducts a breath-first like search (in terms of the hop-number, flooding from the source) via rebroadcasting by intermediate nodes from one level to the next level in a continuous and relay fashion. Typical control messages include RREQ (Route REQuest) and RREP (Route REPly). Each of them contains some essential information for cooperation, for example, the *source* and *destination* addresses, initial *hop count*, and *time-to-live* value (*TTL*).

TTL sequence-based mechanism is generally adopted to minimise flooding in ERS. The TTL number may increase with a specified value [6], a fixed value of 1 [7] or 2 [8], [9], or a random value [10]. An optimal set of TTL values was introduced later to solve the generic minimal cost flooding search problem. However, it has been shown that there is no significant advantage of using the optimal TTL sequence compared to the basic ones [10]. In addition, the optimal TTL sequence-based discovery causes longer delay than the basic route discovery mechanism [10]. Figure 1 shows how a set of flooding regions are controlled by a sequence of predefined TTL values of $1, 2, 3, \cdots$, and n.

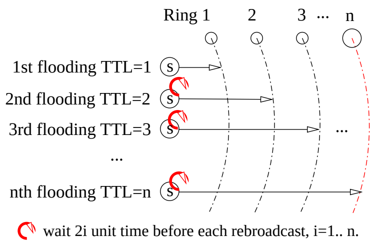

Fig. 1. TTL sequence-based ERS

The TTL sequence-based ERS suffers from energy inefficiency. As it can be seen from Figure 1, if no RREP is received, the source node reinitialises a ring search by re-broadcasting a RREQ with an increased TTL number. The route discovery procedure repeats until a route node is found, or the source node abandons the search. Otherwise, it is possible that the entire network is repeatedly flooded because the source node does not necessarily have sufficient global knowledge about the network. This can overload the network and exhaust the valuable energy resources of individual nodes. It is especially expensive when searching is required in a large area of the network.

We define the latency as the searching period required, startting from the time when the source node sends the first RREQ until the time by which the flooding ceases. Assume that TTL increment is 1, and it takes 1 unit of time for a message to be transmitted from one node to its one hop neighbour. The total amount of energy and of the latency for the search process can be calculated [4] as summarised in the table in section III-C.

ERS wastes energy by re-broadcasting RREQs redundantly. A flooding analysis shows that re-broadcasting could provide at most 60% additional coverage and only 41% on average over that already covered by the previous attempt [11].

B. Blocking expanding ring search (BERS)

The BERS is an alternative energy efficient ERS scheme [4]. The source node S in BERS, unlike that in ERS, issues a RREQ once only. It does not resume an incomplete route search procedure even when a re-flooding is required. The re-flooding can be initialised by any appropriate intermediate nodes. These intermediate nodes may take over a re-flooding process on behalf of the source node and act as an *agent* node. In addition to fulfilling their normal duties as a relay node (for example, examining if they are a route node themselves), they rebroadcast after $2H$ 'waiting time' if they are not a route node, where H is their hop number. The source node S is, however, still responsible for terminating the route discovery process, and issuing a termination control packet END (the 'stop_instruction' in [4]) upon receipt of a RREP. The automatic flooding continuous until a END message reaches all the nodes on the last flooding ring H_r, i.e. where a route node was found.

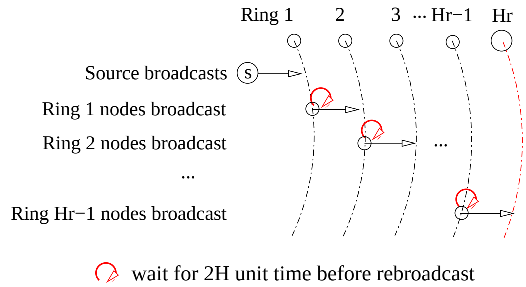

Fig. 2. BERS

Figure 2 shows an example of the BERS approach in which the first round re-flooding, the second re-flooding, $\cdots$, the last re-flooding are initialised by the relay nodes in Ring $1, 2 \cdots, H_r - 1$ respectively.

Two signals are used in BERS to control flooding. One is the RREP, which can be sent to the source node S by any route node reporting the route information. The other is called END which can only be sent by the source node S. The RREP informs the source node that 'a route node has been found', while the END is an instruction to everyone involved in the flooding to terminate the route discovery process.

We define again the search latency as the time required for a period from the source sending the first RREQ to the time by which the flooding ceases. The total amount of energy consumed for one route discovery and the search latency can be estimated [4] as summarised in the table in section III-C.

III. Enhanced BERS (BERS*)

BERS*, an enhanced version of BERS, works in a similar way as BERS except that it requires intermediate nodes to wait for only half amount of waiting time on each round. Instead of waiting for $2H$ units of time before resuming the route discovery process, intermediate nodes in BERS* wait for H units of time only. This speeds up the overall route discovery process by nearly a two-folds compared to BERS. As the waiting time is shortened to half, BERS* is required to flood one ring beyond the H_r. In other words, the flooding ceases at ring $H_r + 1$.

Like BERS, the source node S in BERS* issues a RREQ (including a hop number H) only once to initialise the route discovery process. Intermediate nodes on each subsequent ring take over the responsibility of rebroadcast. They wait, if not a route node, for H units of time before re-broadcasting a RREQ.

A. Algorithms

To develop these ideas further, we have derived four algorithms for BERS*. Algorithm 1 is for the source node. Algorithm 2, 3 and 4 are for the intermediate, and route nodes.

Algorithm 1 covers the actions of a source node for the route discovery process, with the life time $(1 + 2.5MAX_H + 0.5MAX_H^2)$. This includes initialising a route discovery process by first sending a RREQ (line 1), sending a END instruction after a RREP is received (line 4) and handling the route information in RREPs (line 5, 6).

Algorithm 1 Source node

1: broadcast RREQ, including $H = 1$ and MAX_H
2: wait until a RREP is received or the life time runs out
3: **if** receives a RREP, while waiting **then**
4: broadcast the END (including H_r) to everyone within the H_r ring
5: use the 1st RREP as the route for data packets and save the 2nd RREP as a backup
6: drop any later RREPs
7: **end if**

Algorithm 2 Intermediate node

1: **repeat**
2: listen to RREQ
3: **until** RREQ is received
4: **if** 1st RREQ is received **then**
5: call procedure_rreq
6: **end if**
7: **repeat**
8: listen to RREP
9: **if** 1st RREP is received **then**
10: forward RREP by unicast
11: **end if**
12: listen to END
13: **until** END is received
14: call procedure_end

Algorithm 3 procedure_rreq

1: **if** RREQ.$H > MAX_H$ **then**
2: drop the RREQ and any other related messages
3: erase the (S,D) pair in route cache, and terminate
4: **else**
5: **if** route information is in the cache **then**
6: send a RREP (including H_r) to the source node
7: **else**
8: wait for a period of ‘waiting time’ (H for BERS*)
9: **while** waiting **do**
10: **if** END is received **then**
11: call procedure_end
12: **else if** RREP is received **then**
13: forward RREP by unicast
14: **end if**
15: **end while**
16: **if** no END, nor RREP is received during waiting **then**
17: update RREQ.H and rebroadcast RREQ
18: **end if**
19: **end if**
20: return
21: **end if**

Algorithm 4 procedure_end

1: **if** END.$H \leq H_r$ **then**
2: forward END
3: **else**
4: drop END
5: **end if**
6: erase the source-destination pair in the route cache
7: terminate

Similarly, Algorithm 2 summarises the actions taken by intermediate nodes depending on which of the three messages (RREQ, RREP, END) is received. Algorithm 3 and 4 are two procedures describing actions of the intermediate nodes when a RREQ and END are received respectively.

In Algorithm 3, once a route node is identified, a RREP will be sent with the current hop number (i.e. H_r) to the source node (line 5–6). Other intermediate nodes need to wait for a period of H ‘waiting time’ (line 8) and start flooding if no END instruction is received (line 16–17). During the ‘waiting time’ period, the intermediate nodes need to forward a END (line 10–11, calling the procedure_end in Algorithm 4) or the RREP (line 13) because there might be the 2nd RREP for the source node as a backup.

*B. Energy and time efficiency of BERS**

Let H_r be the hop number of a route node, and n_i be the number of broadcasting nodes in ring i, where $i = 1, 2, \cdots, H_r$. Assume each broadcast or unicast consumes 1 unit of energy. The total amount of energy consumption can be computed for one route discovery process as follows, where n_{H_r} represents the number of nodes on Ring H_r, and n_r the number of route nodes:

$$E_{BERS*} = E_{BERS} + 2n_{H_r} - n_r \text{ (UnitEnergy)}$$

The total amount of time taken includes the time for three stages: (a) searching for the route node, (b) returning the RREP, and (c) broadcasting the END instruction.

For stage (a), the time taken consists of the time for broadcasting and waiting time. It takes H_r units of time for broadcasting. As the waiting time for ring i is i, where $i = 1, 2, \cdots, H_r - 1$, the total amount of waiting time is $\sum_{i=1}^{H_r-1} i$. For (b), it takes H_r units of time for S to receive a $RREP$, i.e. $T_{RREP} = H_r$. For (c), it takes another $H_r + 1$ units of time for the END instruction to be received by the nodes on the last ring $H_r + 1$.

We define the search latency as the time required for a period from the source sending the first RREQ to the time by which the flooding ceases. Assume it takes 1 unit of time for a message to be transmitted from one node to its one hop neighbour. The total amount of time for the process is:

$$T_{BERS*} = 1 + 3H_r + \sum_{i=1}^{H_r-1} i = 1 + \frac{5H_r + H_r^2}{2} \text{ (UnitTime)}$$

*C. Comparison of energy and latency between ERS, BERS and BERS**

We summarise the energy consumption and the time taken by the three approaches in the table below, where n_r is the number of route nodes on Ring H_r:

Scheme	Energy Consumption	Latency
ERS	$(n_r + 1)H_r + \sum_{i=1}^{H_r-1} \sum_{j=1}^{i} n_j$	$H_r + H_r^2$
BERS	$2(1 + \sum_{i=1}^{H_r-1} n_i) + n_r H_r$	$2H_r + H_r^2$
BERS*	$E_{BERS} + 2n_{H_r} - n_r$	$1 + 2.5H_r + 0.5H_r^2$

As we expected, both the level of energy consumption and the amount of time taken depend on the distance between the source node and its nearest route node in terms of H_r, while the amount of the energy consumption depends also on the node distribution, i.e. the number of nodes on each ring, within the area defined by H_r.

IV. Simulation and results

We have conduct a number of analytical simulation based on the above theoretical results and implemented in IDL 6.0 (Research Systems, Boulder, CO, USA). Our main goal is to investigate the difference between the performance of BERS*, BERS and ERS in terms of energy efficiency, latency and energy-time efficiency. In order to gain the insight of the performance of three schemes, we conduct a series of experiments on the three sets of the searching algorithms, and investigate their behaviours under a uniform node distribution as follows: we assume a total of 1000 nodes are placed uniformly in a geographic area covering a region of $H_r = 10$.

A. Energy efficiency or latency separately

We first measure the performance of BERS*, BERS and ERS in terms of energy efficiency or latency incurred.

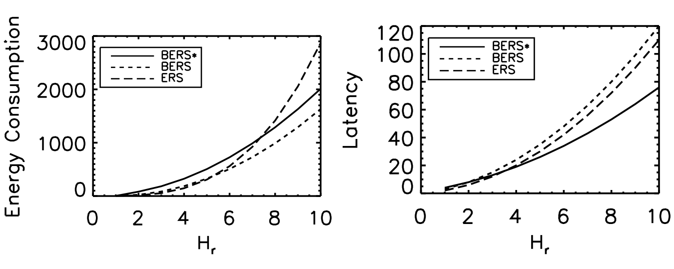

Fig. 3. Energy consumption or latency

Figure 3 (left) shows a plot of the energy consumption against H_r. As we can see, while the amount of energy consumption increases as the number of rings increases, BERS is the most energy efficient of the three. When $Hr \leq 7$ approximately, ERS is more energy efficient than BERS*, but when $Hr > 7$, BERS* is more energy efficient than ERS.

Figure 3 (right) shows the time delay required for the three schemes against H_r. As the H_r increases, the latency also increases for all the three schemes. When $Hr \leq 3$, as we can see, BERS* and ERS incur a similar amount of time delay. When $Hr > 3$, BERS* is the most time efficient, the ERS is the next, and BERS is the least time efficient.

These two figures in Figure 3 suggest a trade-off between the energy saving and searching latency.

1) Energy efficiency: As we can see from Figure 3 (left), when $H_r \leq 7$, BERS and ERS are more energy efficient than BERS*, and when $H_r > 7$, BERS and BERS* are more energy efficient than ERS. To show the detailed energy saving achieved by BERS, we compare further between BERS and ERS, and between BERS and BERS*.

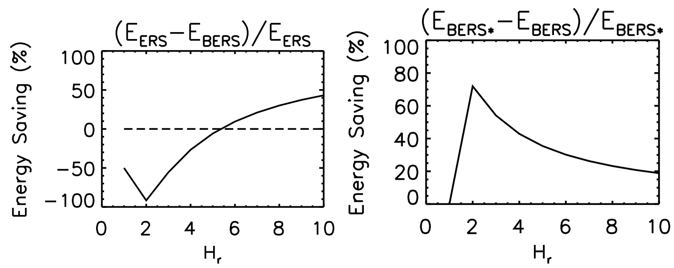

Fig. 4. Energy saving percentages

Figure 4 shows the energy savings in percentage of BERS vs. ERS and of BERS vs. BERS*, and the tendency of the savings as H_r increases. Figure 4 (left) shows clearly that BERS does not save any energy until $H_r > 5$. This suggests that ERS should be used when $H_r \leq 5$. Figure 4 (right) tells us that, although BERS is more energy efficient than BERS*, the saving percentage of BERS drops from 72% to nearly 19% when H_r increases from 2 to 10. It suggests that BERS makes no more than 20% energy savings than BERS* for a larger H_r, for example, when $H_r \geq 10$.

2) Latency: Similarly, we investigate further the searching latency of ERS vs. BERS*, and BERS vs. BERS*, using the mathematical expressions from our analysis:

$$(T_{ESR} - T_{BESR*})/T_{ERS} = \frac{H_r^2 - 3H_r - 2}{2(H_r^2 + H_r)}$$

$$(T_{BESR} - T_{BESR*})/T_{BERS} = \frac{H_r^2 - H_r - 2}{2(H_r^2 + 2H_r)}$$

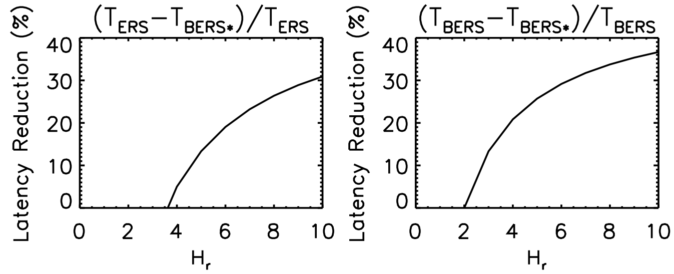

Fig. 5. Latency reduction

Figure 5 (left) shows that when $H_r > 3$, BERS* improves increasingly the time efficiency in comparison with that of ERS by as much as 31% when $H_r \geq 10$. Figure 5 (right) shows that, on the other hand, BERS* improves the time efficiency in comparison with that of BERS even more, by as much as 37% when $H_r \geq 10$. Although BERS* consumes slightly more energy than that of BERS, the time efficiency of BERS* makes it more attractive.

B. Energy efficiency and latency together

Most research on energy efficient algorithms settles on the algorithm that is the most energy efficient, or that is relatively more energy efficient than another. We feel strongly, however, that energy efficiency issues cannot be discussed in isolation. It is insufficient to consider energy efficiency alone without investigation on the cost since there is often a trade-off between a gain of energy saving and loss in the time delay. Our findings on the performances of BERS*, BERS and ERS demonstrate a strong correlation between the energy consumption and the searching latency. The research on the energy efficiency and latency separately for the BERS*, BERS and ERS in the previous sections motivate further investigations on which one is more energy-time efficient.

We consider the overall performance of BERS*, BERS and ERS, applying the product model [5] to measure energy-time efficiency. Having taken into consideration both energy consumption and incurred latency, we derive the following results, highlighted in figures 6 and 7 .

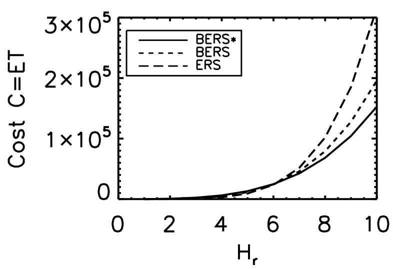

Fig. 6. Energy and latency

Figure 6 shows plots of the overall energy-time efficiency of BERS*, BERS and ERS against H_r. As we can see, BERS* outperforms BERS and ERS in terms of the energy-time efficiency when $H_r \geq 7$. At $H_r = 10$, BERS* improves the energy-time efficiency by 50% compared to ERS, and by 21% compared to BERS.

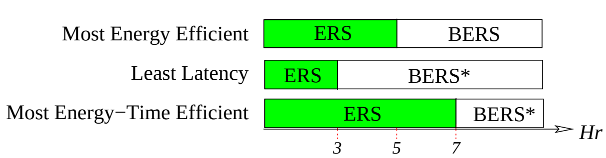

Fig. 7. Phrase diagram with thresholds

Figure 7 summarises overall performances of BERS*, BERS and ERS. The most energy efficient scheme is ERS when $H_r < 5$, or BERS when $H_r \geq 5$. The scheme with the least latency is ERS when $H_r \leq 3$, or BERS* when $H_r > 3$. The most energy-time efficient scheme is ERS when $H_r < 7$, or BERS* when $H_r \geq 7$.

V. Conclusions

We have introduced BERS*, an enhanced blocking expanding ring search scheme and analysed the performance of BERS*, BERS and ERS in terms of energy efficiency, search latency and energy-time efficiency. Our results show that, among the three schemes (BERS*, BERS and ERS), BERS* incurs the least latency when the hop number of the route nodes is greater than 3, and has achieved the best performance in terms of energy-time efficiency when the hop number of the route nodes is greater than 7.

The result suggests that BERS* can be potentially useful for large scale MANETs where the route node of a source is more than 7 hops away with a high probability. The findings are interesting and the analytical approach provides a way forward to gain the insight of complex systems such as a MANET when there are often more unknowns than knowns.

Our findings are valuable for practical applications. When more information is available about a MANET, for example, if the statistics about node distribution or the probabilities of the hop number of route nodes, or the size of the network are available, the threshold conditions can then be used to achieve further collective optimisation, e.g., switch between ERS and BERS* based on the thresholds.

References

[1] D.B. Johnson and D.A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, T. Imlellnski and H. Korth, Eds. Kluwer Academic, 1996, pp. 153–181.

[2] C.E. Perkins and E.M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of The 2nd Annual IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, New Orleans, LA, February 1999, pp. 90–100.

[3] S.-J. Lee, E.M. Belding-Royer and C.E. Perkins, "Scalability study of the ad hoc on-demand distance vector routing protocol," *Int. J. Network Mgmt*, vol. 13, no. 2, pp. 97–114, 2003.

[4] I. Park, J. Kim, and I. Pu, "Blocking expanding ring search algorithm for efficient energy consumption in mobile ad hoc networks," in *Proc. WONS 2006*, pp. 185–190, France, 2006.

[5] I. Pu, Y. Shen, and J. Kim, "Measuring energy-time efficiency of protocol performance in Mobile Ad Hoc Networks," in *ADHOC-NOW 2008*, D. Coudert et al, Eds. Springer-Verlag, Berlin Heidelberg, 2008, LNCS, vol. 5198, pp. 475–486. ISBN 978-3-540-85208-7.

[6] J. Hassan and S. Jha, "On the optimisation trade-offs of expanding ring search," in *Proc. of IWDC 2004*, December 2004, pp. 489–494.

[7] E.M. Royer, "Routing in Ad Hoc Mobile Networks: On Demand and Hierarchical Strategies," Phd thesis, University of California at Santa Barbara, USA, December 2000.

[8] C. Perkins, E. Royer, and S. Das. (2003, July) Ad hoc on-demand distance vector (aodv) routing. IETF Request for Comment. [Online]. Available: www.ietf.org/rfc/rfc3561.txt.

[9] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceeding of the ACM Sigmetrics Conference*, Mariana del Rey,CA, June 2002.

[10] D. Koutsonikolas, S.M. Das, H. Pucha, and Y.C. Hu, "On optimal TTL sequence-based route discovery in MANETs," in *In Proc. of the 2nd International Workshop on Wireless Ad Hoc Networking (WWAN 2005)*, Columbus, OH, June 2005.

[11] Y. Tseng, S. Ni, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, 2002, vol. 8, no. 2, pp. 153–167.