Improving Time-Efficiency in Blocking Expanding Ring Search for Mobile Ad Hoc Networks

Ida M Pu[a], Daniel Stamate[a], Yuji Shen[b]

[a]*Department of Computing, Goldsmiths, University of London*
London SE14 6NW, United Kingdom
[b]*Brain Research Imaging Centre, University of Edinburgh*
Edinburgh, United Kingdom

Abstract

We propose a new strategy for reducing the amount of latency and energy consumption in Blocking Expanding Ring Search (BERS) and enhanced Blocking Expanding Ring Search (BERS*) for mobile ad hoc networks (MANETs). BERS and BERS* are respectively energy and energy-time efficient route discovery protocols for MANETs as compared to conventional Expanding Ring Search (ERS). In this study, we identify unnecessary waiting time caused by a STOP/END instruction in BERS/BERS* and explore the potential of further improvement of their time efficiency. This leads to tBERS and tBERS*, the improved BERS and BERS* respectively. In tBERS/tBERS*, a route node may also issue the STOP/END instruction to terminate flooding. We implement this idea in algorithms, conduct analysis, and achieve further latency reduction in both tBERS and tBERS* as well as the energy saving in tBERS*.

Keywords: algorithm, expanding ring search, ad hoc network, energy time efficient, routing, ERS, BERS, BERS*, tBERS, tBERS*, MANETS

1. Introduction

Mobile ad hoc networks (MANETs) rely upon cooperation of mobile nodes to establish communication routes dynamically. Since the nodes are generally powered by batteries, both energy and time efficiency are particularly important for MANETs. Many energy and/or time efficient routing protocols have been developed for MANETs. The Blocking Expanding Ring Search (BERS) [1] and the Enhanced Blocking Expanding Ring Search (BERS*) [2] are two recently developed protocols of route discovery for MANETs.

Both BERS and BERS* aimed at improving the energy efficiency of a widely used search strategy, namely, the Expanding Ring Search (ERS) in popular reactive routing protocols, such as Dynamic Source Routing (DSR) [3] and Ad hoc On-Demand Distance Vector Routing (AODV) [4, 10]. BERS was proposed as a new flooding control method to remove an energy-waste act in ERS at the cost of some increased discovery latency (latency in short). To focus on protocols performance, we define the latency as the time required from the moment when a RREQ (Route REQuest) is sent to the moment when the flood is terminated. To improve the overall performance in both energy and time efficiency, the BERS* was developed to reduce the latency in BERS by nearly half with a slight increase of the energy consumption [2]. Among ERS, BERS and BERS*, BERS* achieved the best overall performance in terms of the energy-time efficiency [2].

Both BERS and BERS* use a chase packet, referred to as STOP in BERS and END in BERS*, to terminate the flooding after a route node is discovered. For ease of discussion we refer them to as STOP/END instruction to mean STOP in BERS or END in BERS*. The STOP/END instruction is issued only by the source node in BERS and BERS*. The source node cannot issue the STOP/END instruction until the first RREP (Route REPly) arrives. In other words, the STOP/END instruction is not issued without a delay caused by waiting for RREPs.

In this paper, we address this problem and propose a new approach. We will demonstrate that such a STOP/END instruction can be sent out as soon as the route node is discovered. Instead of waiting for the source node to issue a STOP/END instruction, the route node may also take part in issuing the STOP/END command to terminate the flooding. We will demonstrate how the route discovery latency of both BERS and BERS* can be reduced without any increase of energy consumption.

We will further demonstrate that the energy consumption for BERS* can also be reduced slightly when applying this new strategy. In the rest of the paper, we first describe briefly the related work in Section 2, including the time-to-live(TTL)-based ERS, BERS and BERS*. We then introduce in Section 3 two new time-efficient protocols referred to as tBERS and tBERS* (corresponding to BERS and BERS* respectively), providing the algorithms for tBERS and tBERS* and comparisons with the original BERS and BERS*. We present the simulation results in Section 5. Finally, we summarise our conclusions in Section 6.

2. Related Work

2.1. Expanding ring search (ERS)

The ERS is an effective way of finding a route between a source node and a route node. A route node is referred to as either the destination or the node that can offer the route information to the destination. As a controlled flooding technique, ERS is used frequently in reactive routing protocols. Starting typically in a predefined small searching area, the ERS conducts a new search from the source node in an enlarged searching region each time if no route node is found. This incremental searching process continues until a route node is found or the maximum searching area is reached. The search in ERS is performed with flooding involving rebroadcasts through intermediate nodes in a continuous and relay fashion, and it is like a search zone expanding gradually, ring by ring, from a small ring to a large one.

The source node in ERS initializes the flooding and controls the searching area in each extended flooding as well as the maximum searching area. Two control signals, RREQ (Route REQuest) and RREP (Route REPly), are used in ERS to effectively control the flooding. To minimise flooding, a time-to-live (TTL) mechanism is adopted by ERS. The TTL sequence determines the order of flooding search, and it may be assigned with an increment at a specified value [5], a fixed value of 1 [6] or 2 [7], [8], or a random value [9]. 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.

The TTL-based ERS suffers from an energy inefficiency. As it can be seen from Figure 1, if no RREP is received by the source node, the source node will rebroadcast a RREQ with an increased TTL value. The rebroadcasting of a new RREQ from the source node each time

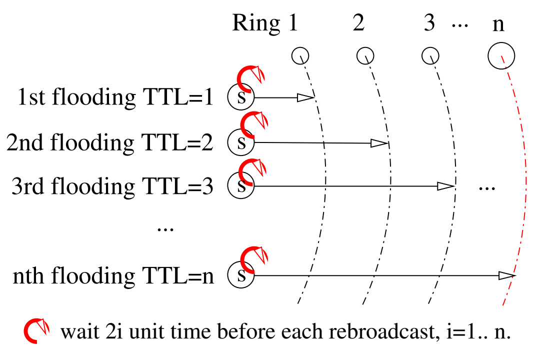


Figure 1: ERS

introduces energy waste. A redundancy happens in the area overlapped with the previously covered searching area. This may happen many times before a route node is found or an entire network is searched.

2.2. Blocking expanding ring search (BERS)

The BERS [1] can be viewed as an energy-efficient ERS. BERS adopts a strategy that the source node passes on the right of re-flooding to the intermediate nodes. The source node in BERS issues a RREQ once only. An intermediate node, acting as an agent, performs rebroadcasting on behalf of the source node. To fulfil this strategy, an extended $2H$ units of waiting time is implemented in BERS, where H is the hop number. The source node, however, is still responsible for the termination of the route discovery process. The source node issues a STOP instruction upon receiving a RREP to terminate flooding. The flooding continues until the chase packet, a STOP instruction, reaches all the nodes on the last flooding ring H_r, where a route node was found.

Figure 2 demonstrates how the BERS works in the searching propagation from one ring to the next ring. In BERS, the source node issues a RREQ first and waits for RREP. If no route node is found in the first ring, the nodes in the first ring rebroadcast the RREQ with an increased hop number. This process continues until a RREP is received by the source node. The source node then broadcasts a STOP message to terminate the search. The STOP instruction can only be sent by the source node. The nodes involved in the flooding will receive a STOP instruction. The RREP can be sent by any route node to the source node. Note that an extended $2H$ units

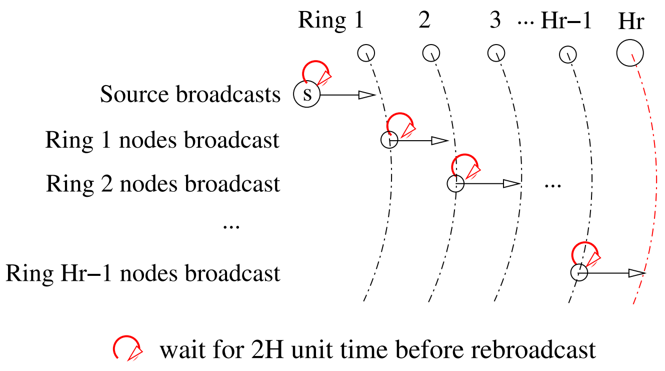


Figure 2: BERS

of waiting time is required for BERS on each round of flooding in order to effectively terminate further flooding after a route node is found.

2.3. Enhanced blocking expanding ring search (BERS)*

The BERS* is based on BERS and aimed at reduction of the latency for BERS to achieve the best overall performance in terms of the energy-time efficiency [2].

BERS* works in a similar way as BERS except that it reduces the waiting time to half on each round of flooding. The intermediate nodes in BERS* take over the responsibility of rebroadcast on each subsequent ring search after the source node issues a RREQ. These intermediate nodes, if not a route node, wait for H units of time before re-broadcasting a RREQ.

In contrast to the $2H$ units of waiting time in BERS, this speeds up the overall route discovery process by nearly two-folds and improves the overall performance of the route discovery in terms of the energy-time efficiency. As it takes $2H_r$ units of time for the nodes at ring H_r to receive the END flooding signal, the flooding in BERS* may terminate at ring $H_r + 1$, one ring beyond the ring where the route node was found.

Figure 3 shows that only one extra ring is required due to concurrent node actions between the waiting and the propagation of RREP and END instructions. As we can see, first, it takes H_r units of time for RREP to transmit from the route node R to the source node S (arrowed lines in blue) before concurrent actions begin. Secondly, during the next 1 unit of time, the END instruction is broadcast to ring 1 while nodes a and b continuing the flooding from ring H_r to $H_r + 1$ (arrowed lines in red). Third, it takes the next H_r units of time for the END packet to catch up with node c during its $H_r + 1$ units of waiting time (dashed arrowed lines in purple from

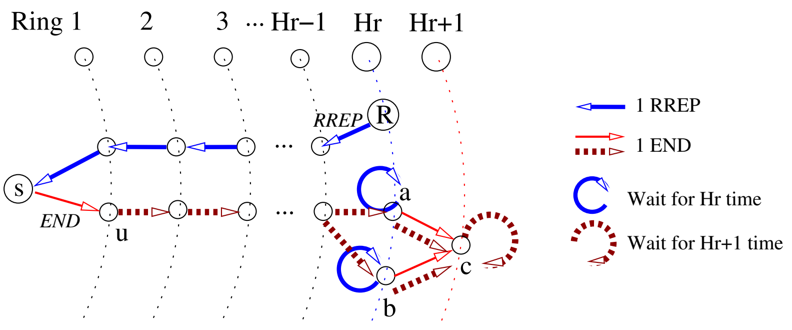


Figure 3: BERS*

u at Ring 1).

The study has shown that the BERS* can achieve the best overall performance in consideration of both the energy and time efficiency [2].

3. Time-Efficient BERS (tBERS) and BERS* (tBERS*)

The tBERS and tBERS* are referred to as the enhanced BERS and BERS* respectively for reducing the latency without increase of any energy consumption. We adopt the names of the chase packets, i.e. STOP for tBERS and END for tBERS*, and STOP/END for tBERS and/or tBERS*. The tBERS works the same way as BERS, and tBERS* the same as BERS*, except that both tBERS and tBERS* allow the route node to issue a STOP/END instruction. By passing-on the right of issuing the termination instruction to the route nodes, tBERS and tBERS* achieve latency reduction in comparison to BERS and BERS* respectively. The amount of the reduced latency equals to H_r, the transmission time for a RREP to reach the source node.

In BERS or BERS*, the STOP/END is only issued by the source node to terminate flooding. In tBERS or tBERS*, upon receipt of the RREP, the source node issues a STOP/END instruction, too, just as that in BERS or BERS*. However, the purpose of this is to terminate the rest of the flooding that was uncovered by the STOP/END issued earlier by the route node. This part of the flooding is caused by those nodes on ring H_r but did not receive the STOP/END from the route node due to their unreachable geographic location during the RREP unicast. In addition, in tBERS or tBERS*, the data packets are ready to be sent soon after the receipt of the RREP by the source node.

Both tBERS and tBERS* take the advantages of the concurrent activities. The first type of concurrent activities takes place between the RREP unicast and the STOP/END multicast from the route nodes. Although the STOP/END issued by the route node may not be able to catch all the nodes at ring H_r to terminate flooding, this approach in tBERS and tBERS* aims at stopping some of the nodes at ring H_r from further flooding, earlier than it would otherwise in BERS and BERS*. The second type of concurrent activities takes place between the data packets and the STOP/END instruction issued by the source node. This allows data packets to be transmitted earlier than those in BERS and BERS*.

As we shall see later, it can be demonstrated that the level of the energy consumption for tBERS is the same as that of BERS. However, the energy saving can be achieved as a bonus for tBERS*. In BERS*, the flooding is terminated at $H_r + 1$, one ring beyond the ring where the route node was found. In contrast, only part of the flooding in tBERS* is terminated on average at this one extra ring while the rest of the flooding may be terminated earlier at H_r, the ring where the route node was found. This is because some of the nodes in ring H_r may receive the END instruction during their H_r units of waiting time.

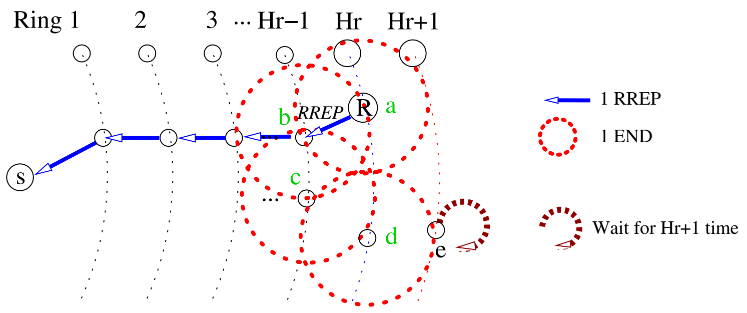


Figure 4: An example in tBERS*

Figure 4 shows an example of tBERS* where node d stops flooding early despite it is more than one-hop away from route node R. Here node d is at ring H_r but beyond one-hop distance from the route node R. However, node d would receive an END signal from route node R via nodes b and c. In contrast, if it were in BERS*, node d would have broadcast one more ring after H_r units of its waiting time and the flooding would not have been terminated until ring $H_r + 1$.

In case of the nodes in the uniform distribution, about one third of nodes in the ring H_r may receive the END message during their waiting time, as shown in Figure 5, where the nodes marked in red involving termination of the flooding before RREPs reach the source node S.

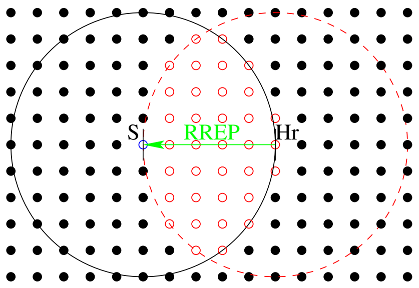


Figure 5: tBERS* for uniform distribution of nodes

The uniform distribution described above represents the best case for tBERS*. In the worst case, however, it is possible that none of the non-route nodes at ring H_r receives the END instruction from the route node due to its geographic position. In this case, tBERS* would then work just as the same as the BERS*, i.e. receiving the END instructions from the source node during the second H_r period.

The route node in tBERS or tBERS* is responsible for issuing both the RREP and STOP/END instructions. There are two ways to realise the ideas of sending out the RREP and STOP/END instructions simultaneously. One is to combine both RREP and STOP/END together and the other is to send out the RREP instruction first and the STOP/END instruction next. Note that RREP is sent in unicast whereas the STOP/END is in broadcast. As an example, we highlight our ideas in algorithms for tBERS* taking the second approach.

3.1. Algorithms

We only give four algorithms for tBERS* below as the algorithms for tBERS are similar. Note an END instruction is used in the algorithms to terminate the flooding. Algorithm 1 is for the source node. Algorithms 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. This includes initialising a route discovery process by first sending a RREQ (line 1), and handling the route information in RREPs (lines 4, 5). To avoid the potential collision between the END instruction and the data packets, we introduce one unit time lag for the data packets, i.e. wait for 1 unit of time before calling the procedure_data_packets (line 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:     use the 1st RREP as the route for data packets and save the 2nd RREP as a backup
5:     drop any later RREPs
6:     wait for 1 unit of time
       before calling procedure_data_packets (excluded in this paper)
7: end if
```

Similarly, Algorithm 2 summarises the actions taken by intermediate nodes depending on which of the three messages (RREQ, RREP, END) is received.

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
```

Algorithms 3 and 4 are two procedures describing actions of the intermediate nodes when a RREQ and END are received respectively. If it is a route node, it will initialise and broadcast the END instruction (line 8, Algorithm 3).

Algorithm 3 procedure_rreq

```
1: if RREQ.$H > MAX_H$ then
2:     drop the RREQ and any other related messages
3:     erase the Source-Destination (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:         wait for 1 unit of time
8:         broadcast the ‘END’ (including $H_r$) to nodes within the $H_r$ ring
9:     else
10:        wait for a period of $H$ units of time
11:        while waiting do
12:            if ‘END’ is received then
13:                call procedure_end
14:            else if RREP is received then
15:                forward RREP by unicast
16:            end if
17:        end while
18:        if neither END nor RREP is received during waiting then
19:            update RREQ.H and rebroadcast RREQ
20:        end if
21:    end if
22:    return
23: 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
```

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 (lines 5–8). Other intermediate nodes need to wait for a period of H units of time (line 10) and start flooding if no END instruction is received (lines 18–19). During the ‘waiting time’ period, the intermediate nodes need to forward an END (lines 12–13, calling the procedure_end in Algorithm 4) or the RREP (line 15) because there might be a 2nd RREP for the source node as a backup.

4. Comparisons of the results

In this section, we highlight and compare the energy consumption and the latency for ERS, BERS, tBERS, BERS* and tBERS* in the table below. We have excluded the mathematical details for quantifying the energy consumption and the discovery latency here. The interested reader can find the detailed calculations for ERS, BERS and BERS* in [1] and [2], whereas the calculation of the energy consumption and the latency for tBERS and tBERS* follow those of BERS and BERS* respectively. Notations can be found in the Appendix.

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	$2(1+\sum_{i=1}^{H_r-1} n_i) + n_r H_r$	$2H_r + H_r^2$
tBERS	$2(1+\sum_{i=1}^{H_r-1} n_i) + n_r H_r$	$H_r + H_r^2$
BERS*	$E_{BERS} + 2n_{H_r} - n_r$	$1 + 2.5H_r + 0.5H_r^2$
tBERS*	$\leq E_{BERS} + 2n_{H_r} - n_r$	$1 + 1.5H_r + 0.5H_r^2$

Note that the latency for tBERS is $H_r + H_r^2$ and the latency for tBERS* is $1 + 1.5H_r + 0.5H_r^2$. While the amounts of the energy consumption for BERS and tBERS are the same, the amount of the energy consumption for tBERS* is less than that of BERS*, which depends on the node distribution.

5. Simulation and results

We have conducted a number of simulations based on the above analytical results and implemented in IDL6.0 (Research Systems, Boulder, CO, USA). Our main goal is to investigate the improvements made by our new strategy applied to the routing discovery protocols of BERS and BERS*. In order to gain the insight of the performance characteristics of the new schemes, we have conducted a series of experiments on ERS, BERS, tBERS, BERS* and tBERS* under a uniform node distribution as follows. We assume that a total of 1000 nodes are placed uniformly in a geographic area covering a region of $H_r = 10$. We compare the difference between the performance of these protocols in terms of time efficiency, energy efficiency and energy-time efficiency under the above assumption for the node distribution.

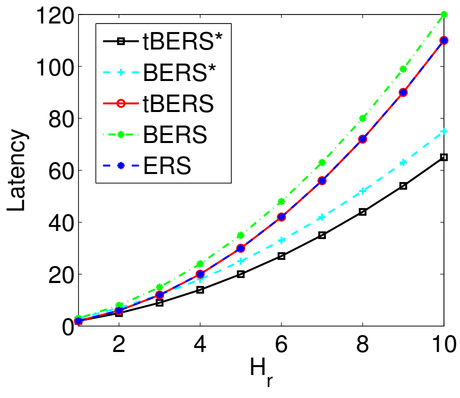
5.1. Latency comparison

The results in comparison of the time efficiency can be summarised in Figure 6.

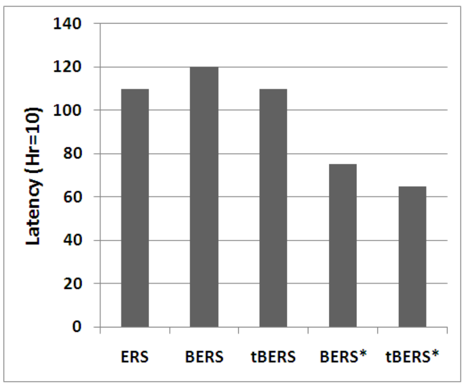
Figure 6a shows the time delay required for the five schemes against H_r. As we can see, the latency increases for all five schemes as H_r increases. However, the latency for tBERS and tBERS* are smaller than the corresponding values in BERS and BERS*. tBERS* has the smallest time delay, demonstrating that the tBERS* is the most time efficient scheme among the five schemes.

Figure 6b highlights the latency of the five schemes when $H_r = 10$. As we can see, tBERS outperforms BERS and tBERS* outperforms BERS*. The most time efficient scheme is tBERS*, followed by BERS*, tBERS and ERS, and finally BERS. When $H_r = 10$, tBERS* improves the time efficiency of BERS* by about 12%.

Similarly, when $H_r = 10$, tBERS can reduce the latency for BERS by about 7.5%.



(a) Comparison



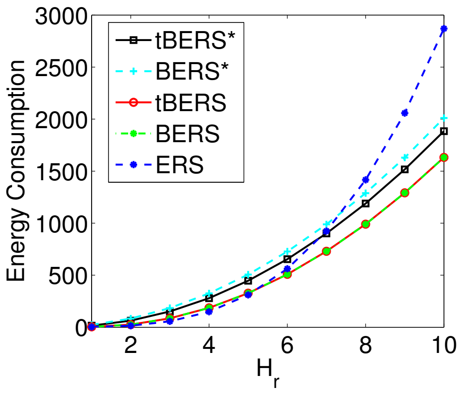
(b) Comparison when $H_r = 10$

Figure 6: Latency

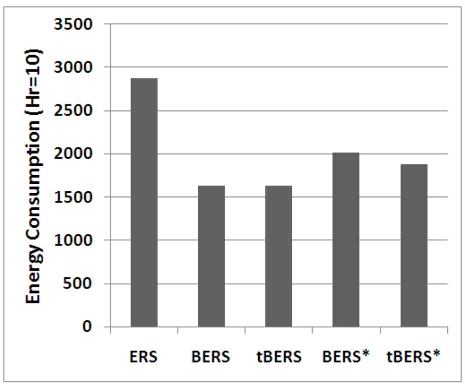
5.2. *Energy consumption comparison*

The results in comparison of energy consumption can be summarised in Figure 7.

Figure 7a shows a plot of the energy consumption against H_r for the five schemes. As it is shown, the amount of energy consumption increases as the H_r increases. BERS and tBERS incur the same amount of energy consumption. These two schemes are the most energy efficient. The level of the energy consumption for tBERS* is lower than BERS*. In comparison to BERS*, when $H_r = 10$, this gives tBERS* an energy saving of about 6%.



(a) Comparison



(b) Comparison when $H_r = 10$

Figure 7: Energy consumption

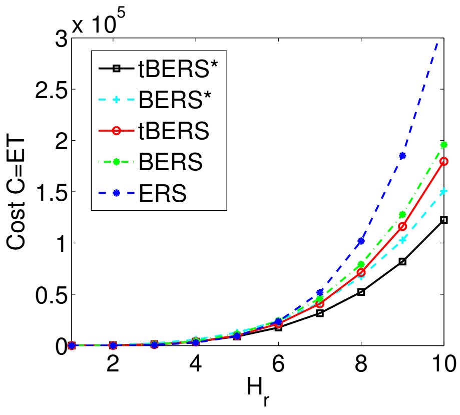
Figure 7b highlights the energy consumption of the five schemes when $H_r = 10$. It is clearly shown that BERS and tBERS are the most energy efficient. Compared to BERS*, tBERS* consumes slightly lower energy.

5.3. Comparison of energy-time efficiency

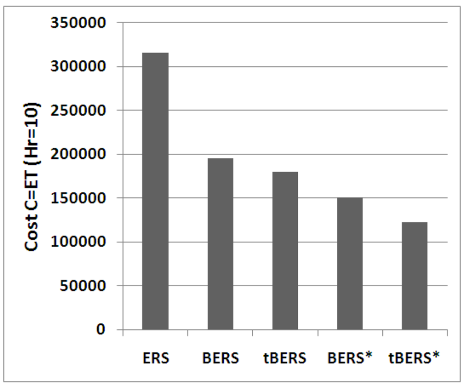
In order to compare the overall performance of the five schemes, we use the product model proposed in [11]. The overall cost is defined as the product of the amount of energy consumption and the amount of latency. The energy-time efficiency between the five schemes are then compared and summarised in Figure 8.

In Figure 8a, the energy-time efficiency is plotted against H_r for each of the five schemes. As we can see, tBERS* is the most energy-time efficient among the five schemes, and BERS* is the next energy-time efficient scheme. This means that the new strategies tBERS and tBERS* proposed in this paper have improved the existing protocols such as BERS and BERS* respectively. The overall cost for tBERS is lower than the corresponding cost of BERS.

Figure 8b highlights the comparison made when $H_r = 10$ where tBERS* improves the energy-time efficiency by nearly 19% compared to BERS*.



(a) Comparison



(b) Comparison when $H_r = 10$

Figure 8: Energy-time efficiency

6. Conclusions

We have introduced a new strategy initially aiming at further reduction of the latency of BERS and BERS*, two existing energy or energy-time efficient protocols. Our studies result in tBERS and tBERS*, two new more time and energy-time efficient routing protocols.

The results are interesting. First, by shifting the responsibility of terminating the flooding from the source node to the route node in tBERS and tBERS*, we have gained more time-efficiency than BERS and more time and energy efficiency than BERS* respectively. This suggests that there might be other opportunities of potentially beneficial redistribution of workload in a collective environment. Secondly, concurrency is found useful in our studies to improve the time efficiency. While latency and energy consumption shows a trade-off nature, concurrency allows improvement in both the time and energy efficiency. Thirdly, the energy-time efficiency is a useful metric for the investigation of an overall performance considering both the time efficiency and energy consumption. The results in time efficiency or in energy efficiency alone tend to be one-sided, incomplete and sometimes misleading. It is beneficial to consider both the time and energy consumption entities together.

References

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

[2] I. Pu, and Y. Shen, "Enhanced Blocking Expanding Ring Search in Mobile Ad Hoc Networks," in *Proceeding of the 3rd International Conference on New Technologies, Mobility and Security (NTMS 2009)*, pp. 451–455, Cairo, Egypt, 20–23 December 2009.

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

[4] 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)*, pp. 90–100, New Orleans, LA, February 1999.

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

[6] 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.

[7] 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.

[8] 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*, pp. 258–259, Mariana del Rey, CA, June 2002.

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

[10] 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.

[11] 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.

Appendix - Terms and notations

AODV	Ad hoc On-Demand Distance Vector Routing
BERS	blocking expanding ring search
BERS*	enhanced blocking expanding ring search
tBERS	time efficient BERS
tBERS*	enhanced tBERS
DSR	Dynamic Source Routing
STOP/END	flood termination instructions
ERS	expanding ring search
latency	time from RREQ sent to flood termination
TTL	time to live
MANET	mobile ad hoc network
RREQ	route request
RREP	route reply
d, D	destination
E_x	energy consumption for x
H	hop number
H_r	hop number of a route node
MAX_H	maximum number of rings
N_i, N_j	neighbour nodes
s, S	source
T_x	latency for x
n_i	number of nodes at ring i
n_r	number of route nodes