Dynamic AODV Backup Routing in Dense Mobile Ad-Hoc Networks[†]

Wen-Tsuen Chen and Wei-Ting Lee

Department of Computer Science, National Tsing Hua University,
Hsin-Chu, Taiwan 300, ROC
Tel:+886-3-5742896 Fax:+886-3-5711484
wtchen@cs.nthu.edu.tw and leif@mercury.cs.nthu.edu.tw

Abstract. The frequent change of network topology in mobile ad-hoc network leads to the stability and reliability problems of routing. Many routing schemes such as multi-path routing and backup path routing are proposed to increase the link reliability. Multi-path routing protocols usually concentrate on load balancing or disjoint routing. However, the problem of packet loss caused by re-routing from the source to the destination is ignored. In this paper, we propose the Dynamic AODV Backup Routing Protocol (DABR) to enhance the Ad hoc on-Demand Distance Vector (AODV) routing in dense mobile ad-hoc networks. The DABR follows the route discovery mechanism of AODV and dynamically calculates the backup routes. Upon the failure of primary route, data packets can be salvaged by redirecting them to the backup routes. The simulation results show that the link reliability of DABR is higher than the conventional AODV while the overhead is controlled.

1 Introduction

In recent years, mobile ad-hoc networks are applied in more and more areas. The characteristics of ad-hoc networks such as infrastructureless and mobility make it easy to deploy in many areas including academia, business, and military. In mobile ad-hoc networks, nodes are considered to be routers which can forward packets and can move freely within the coverage of network. The movement of node results in the change of network topology and the change of routing. The function of routing protocol is to maintain the correct routes even if the topology changes frequently. Besides the problem of changed topology, ad-hoc routings suffer from many strict problems. In ad-hoc networks, low bandwidth (compared to the bandwidth of wired networks), limited battery life, variable nodal density, and potentially large number of nodes make the routing protocol difficult to design.

Many routing schemes are proposed for ad-hoc networks. They can be classified into two catalogs, on-demand and proactive routing. The former includes AODV [1], DSR [2], ABR [6], and etc. The latter includes DSDV [5], OLSR [4], FSR [3] and etc.

[†] This work was supported by the Ministry of Education, Taiwan, R.O.C. under Grant 89-E-FA04-1-4.

In the fashion of on-demand scheme, the source node discovers a route to the destination node only when the route is needed. Mobile nodes usually follow the Route Request (RREQ) / Route Reply (RREP) mechanism to discover a route dynamically. For example, when the source node wants to communicate with the destination node but the route is unknown, the source will broadcast a RREQ message to the networks. The RREQ will be propagated throughout the network until it is received by the destination or is intercepted by an intermediate node which knows a route to the destination. Then a RREP message is replied to the source in the form of unicast and the route is established. The behavior described above is called the route discovery. The main advantage of on-demand routing protocol is that it won't incur any control overhead when there are not any communications in the network. Hence, the change of topology only affects the active routes.

In the other hand, the proactive routing protocols calculate the routes to every node proactively based on the global network information. Each node should periodically or triggered broadcast its routing information (e.g., link-state or distance vector) through the entire network. According to the collected routing information, the node produces a routing table which contains routes to every reachable node. The advantages of proactive routing are low latency of routing setup, good resilience of re-routing, and high capability of route status monitoring. However, the proactive routing suffers from many problems. If the number of nodes increases dramatically, the exchange of routing information will incur very large overhead. And the size of routing table is proportioned to the number of nodes in the network. That is, the demand of both storage and computation capability will increase as the network scale grows.

In order to provide more route reliability onto the on-demand routing, many approaches are proposed to find multiple paths [12], [13], [14] rather than just one shortest path. Multi-path routing schemes allocate multiple paths at the phase of route discovery and deliver data packets among these paths to balance the load of traffic. If one of the paths fails, the source node can use the other path(s) to deliver data packets. Although the reliability of path is increased and the delay of reconstructing a new route is eliminated, the data packets which are sent onto the failure path are missing. Packet loss is not handled in MAC or IP layer but is expected to be recovered in higher layer such as TCP or application layer. Even though all the missing packets can be recovered, the end-to-end delay is produced.

The backup path routing is another type of multiple path routings. Multiple short backup routes are attached to the active primary route [7]-[9]. Data packets are delivered along the primary route rather than distributed them among the backup routes. In general, mobile nodes in the primary route should exchange the routing information with their neighbor nodes [8], [9]. Therefore, the scope of backup path is limited to the vicinity and the length of back path is also restricted. When the primary route is disconnected (due to the absence of relay nodes or radio shadowing), the data packets which are on transmission can be salvaged by re-directing them into the backup route without any delay.

An ideal backup routing protocol in on-demand fashion should achieve the goals as follows.

(1) High delivery rate and low loss rate of data packets
(2) Transparent to the source node

(3) Correct and loop-free backup routes establishing

(4) Precise lifetime (which is the period between the creation and the destruction) of backup routes

(5) Low overhead of maintaining the backup routes

In this paper, we investigate the issue of backup routing which is based on the on-demand routing protocol in the environment of dense mobile ad-hoc networks. We propose a Dynamic AODV Backup Routing (DABR) protocol to dynamically build the backup routes with low control overhead. The DABR follows the standard RREQ/RREP messages of AODV and introduces two new message types: *Alternative Route Request (AREQ)* and *Alternative Route Reply (AREP)*. In DABR, the finding of backup route is initiated after the establishment of primary route and is invisible to the originating node. In order to avoid incurring too much overhead, the length of backup routes is limited.

The rest of this paper is organized as follows. Section 2 states the previous works of backup routing protocols. Section 3 describes the details of DABR protocol. Section 4 shows and explains the simulation results and Section 5 concludes the paper.

2 Related works

Several previous works has been proposed to enhance the link reliability and lower the packet loss in the network layer. Both AODV and DSR have their own mechanisms to salvage the data packets. And several derived approaches have also been proposed.

The local repair mechanism of AODV in [1] is defined as an option. If a link breaks, the upstream node of the broken link can repair the link by initiating a RREQ for the destination and waiting for a RREP. The flooding range of the RREQ is restricted by setting the TTL and the value must be shorter than the formal one. The limited TTL prevents the node from selecting a backup path which is too long. During the local repair, data packets are buffered and thus the end-to-end delay occurs here. If the node receives a RREP before the timeout, the alternate route is set and the data packets are sent to the route. Otherwise, the buffered data packets will be discarded and the RERR message is delivered for the destination. Note that there may be not any overlap between the alternate path and the original path. That is, the backup path may quite different from the original path.

The broken link can be repaired proactively before the incoming packets suffer the transmission error. If the MAC layer could provide the notification of link error to the network layer, the route can be repaired earlier. As soon as the link fails, the incoming packets can be forwarded to the backup route without any delay. However, that the routes which are no longer in active still may be repaired will consume the bandwidth of network.

The AODV-BR [7] is proposed to provide AODV with backup routing without producing any control messages. The alternative paths are set during the propagation of RREPs. Every node must overhear the RREPs which are sent to its neighbors and store the senders in the alternate routing table. After the propagation of RREPs from

the destination to the source, the primary and alternate routes will form a fish bone structure, which is illustrated in Figure 1. If one link in the primary route fails, the upstream node of the broken link must broadcast the data packets to its neighbors and issue a RERR message to the originator. This behavior causes that the source node stars the RREQ/RREP mechanism again to discover a new route for the destination. In the same time, however the broadcasted packets may be salvaged if the neighbors know the alternate route to the destination.

The AODV-BR is based on two characters. One character is that every node can overhear the RREPs. To do this, the node must receive all packets regardless of the destination. The other character is that the node must broadcast the data packets to salvage them if the local link breaks. However, all the nodes which know the alternate route to the destination will forward the data packets. Multiple copies of the data packets will consume the bandwidth and the destination node will receive the redundant packets. Although the AODV-BR claims that it doesn't produce any additional control overhead, the two characters make the protocol difficult to implement in reality. Additionally, the behavior of AODV-BR will be in vain if the topology of neighbors changes after the setup of primary route.

Neighborhood Aware Source Routing (NSR) [8] is proposed to improve the capability of backup routing in DSR. In NSR, both the nodes in the primary route and their 1-hop neighbors should broadcast their 1-hop link-state information. Therefore, every node in the primary route maintains the network topology within two hops. Consequently, the backup routes or shortcuts are computed by the partial topology dynamically by comparing the source routes in the Route Cache and the partial topology. If a link breaks, the upstream node of the broken link must replace the original source route on the packet with the backup route which has been calculated proactively. NSR utilizes the node id to lower the size of control messages and simplify the computation of backup routes. However, the Route Error message should be delivered to notify the other nodes. The reason is that multiple portions of backup routes make the long path. The long path results in the long round trip time. Therefore, the source node should discover the new route for the destination even if there are backup routes.

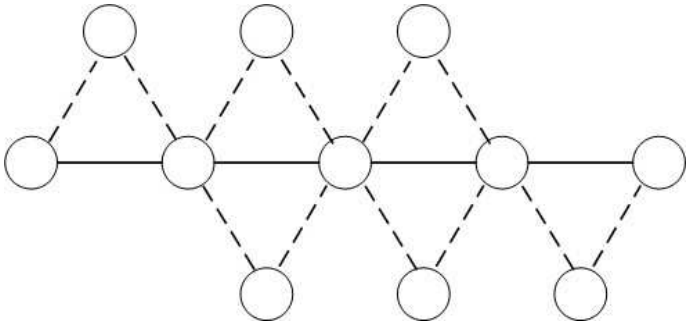

Fig. 1. The primary path and the backup paths

3 Dynamic AODV backup routing (DABR) protocol

The DABR protocol focuses on dynamically finding the backup routes for the existing primary route, which is built by the route discovery mechanism of AODV. Besides the normal routing table, every node must maintain an additional one, called the backup routing table. The backup routes can not exist without the primary route. In fact, the life of backup routes begins at the establishment of the primary route and ends when the given lifetime is up.

Figure 2 shows the finite state machine of backup routing. When the MAC layer detects the occurrence of link failure, the state is changed from normal route to error route. The RERR is immediately delivered to announce that the link breaks here and every node should not use this link anymore. If the backup route exists, the state transfers to the backup route. In this state, the incoming data will be salvaged by redirecting them into the backup route rather than discarded or buffered. However, once the new route has learned from the receiving of RREP, the state enters the normal route. In the state of normal route, every node uses the normal route to deliver and forward packets.

Nodes in the primary route must notify its neighbors that the backup routes are needed by broadcasting the *AREQ* messages containing the routing information, named vector here. The propagation range of AREQ is limited to control the overhead. According to the collected vectors in the *AREQs* from other nodes, the neighbors can determine whether a backup route exists. If there are backup routes, the neighbor will reply *AREPs* to the nodes in the primary route.

The DABR uses three types of message to discover backup routes. They are listed and described in Table 1.

3.1 Message formats

In this section, we define the format of messages used in DABR, including *AREQ*, *AREP* and *AERR*. The format of *AREQ* is defined in the following.

AREQ = **<Source Address (srcAddr), AR Sequence (arSeq), Previous Address (preAddr), Hop Count, Vector>**

The first field is the address of the node which originates the message. The **arSeq** is the sequence number binding to the *AREQ* message. The **preAddr** is the address of the previous node which initiates or forwards the *AREQ* to the receiving node. The hop count from the source to the node receiving the *AREQ* will be saved in the **Hop Count** field. The last field **Vector** contains the routing information of the source node. The backup routes are computed by the collected **Vectors**. Its format is defined in the following.

Vector = {Terminus Address (tmAddr), Terminus Sequence (tmSeq), Hop Count to Terminus (HC2T), Source Address (srcAddr), Lifetime}

The **Vector** contains the routing information of the source node which initiates the *AREQ*. The **tmAddr** is the address of the destination node which the backup routes for it are requested. The **tmSeq** is the sequence number binding to the terminus node. The **HC2T** is the hop count from the source node to the terminus node. The hop count represents the position of node in the primary route. **HC2T** is the main criterion to determine the backup routes. Additionally, the **Lifetime** of vector is given to control the lifetime of vectors. If the lifetime is expired, the vector will be deleted. Next, we define the format of *AREP*.

AREP = **<Terminus Address (tmAddr), Hop Count to Terminus (HC2T), Previous Address (preAddr), Lifetime>**

The **tmAddr** of *AREP* is the address of the destination node. The **HC2T** is the distance from the node sending the *AREP* to the terminus node. The node in the primary route uses the value to determine a shortest backup route if receiving multiple *AREPs*. The **preAddr** is the address of the node which initiates or forwards the *AREP*. The **Lifetime** field is also required to limit the lifetime of backup routes. When the lifetime is up, the backup route is removed from the backup routing table.

AERR = **<Terminus Address (tmAddr), Terminus Sequence (tmSeq), HC2T>**

The *AERR* is defined in the above. The meaning of each field is the same as that have mentioned in the previous paragraphs.

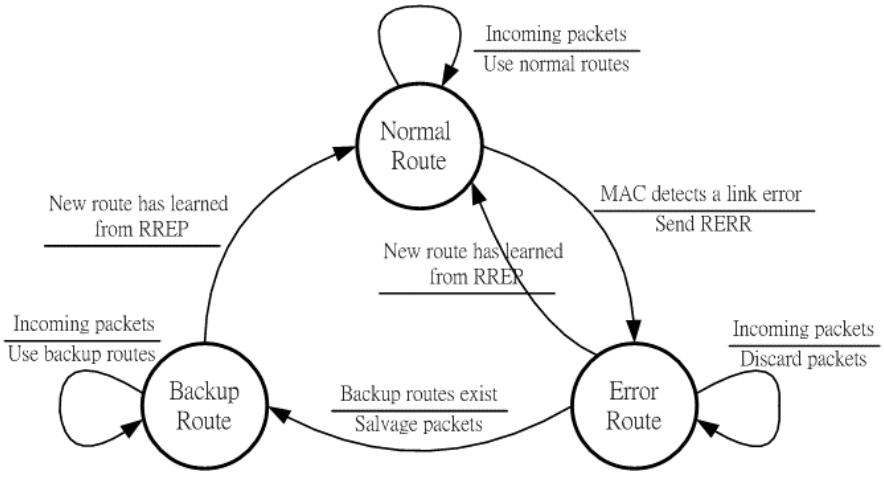

Fig. 2. The finite state machine of backup routing

Table 1. The message types of DABR

Message	Abbreviation	Description
Alternative Route Request	AREQ	Nodes in the primary route broadcast the *AREQs* to notify their neighbors that the backup routes are needed.
Alternative Route Reply	AREP	The neighbors of the primary route send *AREPs* to show the existing of the backup routes
Alternative Route Error	AERR	The node which encounters the broken link from the backup route should send *AERRs* to notify other nodes.

3.2 Operations of DABR

In AODV routing protocol, both RREQ and RREP messages have the field of hop count. It is easy to record the number of hops from the source to the node which receives the message. Therefore, in the phase of route discovery, the node can obtain the information of hop count to the two termini. Thus, the **Vector** can be formed. The receiving of RREP implies that the primary route is established. Subsequently, the node starts broadcasting *AREQs* to its neighbors periodically. The **HC2T** in the sending *AREQ* is copied from the local saved **HC2T**. When initiating a new *AREQ*, the node must increase its own **arSeq**. The pair of **arSeq** and **srcAddr** can determine a unique *AREQ*.

3.2.1 Operation of receiving an AREQ

When a node receives the first *AREQ*, it does not react immediately but waits for a timeout. If the *AREQ* comes from the immediate upstream or downstream node, the message should be discarded. During the timeout, the node collects the *AREQs* and caches the **Vectors** contained in the messages. After the timeout, the node finds the nodes which **HC2T** are smaller than the node itself according to the **Vectors**. These nodes with smaller **HC2T** are candidates of the backup next hop. In order to keep the maintenance simple, each node only maintains one backup next hop for each destination. Therefore, the backup next hop is set to the node with smallest **HC2T** among the candidates. Next, the node should send *AREPs* to notify its possible immediate upstream nodes that there is a backup route to the destination. The sending *AREP* carries the **HC2T** of the backup next hop plus one.

As shown in Figure 3, for example, the path *s* -> *a* -> *b* -> *c* -> *d* is the primary route and the two termini are *s* and *d*. In this case, node *f* can hear the *AREQs* from node *a*, *b* and *c*. The corresponding **Vectors** are <7, a>, <6, b> and <5, c> (the first field represents the **HC2T** to the terminus *d* and the second field is the id of nodes). Since node *f* is not in the primary route, it doesn't know the **HC2T** to *d*. Node *f* just

select node *c* as the backup next hop because the **HC2T** of *c* is the smallest one. Then, *f* sends *AREPs* to the other neighbors which have larger **HC2T** (i.e., node *a* and *b*).

If the topology of the primary route changed because of the moving of nodes, the potential shortcuts may exist. In Figure 4, node *a* receives the *AREQ* from *c* and the **HC2T** of *c* is smaller than node *a* itself. Therefore, node *a* should select *c* as the backup next hop for the terminus *d*. Node *a* should not send any *AREPs* if there are not any other neighbors having larger **HC2T** than *a*.

Multiple primary routes could share some portion of common routes and neighbors. The common routes result from that the RREQ is replied by the intermediate node rather than the destination node. Figure 5 shows that the two primary paths *s -> a -> b -> c -> d* and *e -> f -> g -> c -> d* have the common route *c -> d* and the common neighbor *h* and *i*. Namely, *h* can receive *AREQs* for the same terminus *d* from node *a*, *b*, *f* and *g* while *i* can hear *AREQs* from *b* and *g*. In the view of *h*, both *b* and *g* have the smallest **HC2T**s but only one should be selected to be the backup next hop. What node should *h* choose depends on the order of receiving the *AREQs*. Suppose that the *AREQ* from *b* comes more early than *g*, node *h* chooses *b* as the backup next hop. Next, *h* should decide what upstream nodes should be notified by the *AREPs*. Now, the **HC2T** of backup next hop is 5 and therefore *h* should reply the *AREPs* to *a* and *f* because their **HC2T**s are larger than 5.

The routing loop may be produced when the link *b -> c* and *g -> c* break at the same time. If *h* sends *AREP* to *g*, and *i* sends *AREP* to *b*, the routing loop *g -> h -> b -> i -> g* is formed. In order to avoid the problem, *h* must not send *AREP* to *g*. Similarly, node *i* should not send any *AREPs* because the **HC2T**s of *b* and *g* are tied. The policy is that the node should not send any *AREPs* to the nodes which have the same **HC2T**s as the backup next hop.

3.2.2 Operation of receiving an AREP

When a node receives an *AREP* message, it first checks whether there is already a backup route to the **tmAddr**. If there are not any backup routes, the node assigns the node which initiates the *AREP* to be the backup next hop. Otherwise, the node will choose the shorter route by comparing the **HC2T**s in the backup routing table and the *AREP*.

For example, when node *a* in Figure 3 receives the *AREP* from *e*, node *a* should set *e* as the backup next hop for the terminus *d*. Subsequently, if node *a* hears the *AREP* from *f*, node *f* will substitute for node *e* as the backup next hop because the **HC2T** of *f* is smaller than *e*. The situation of node *b* is somewhat different form *a*. Node *b* hears the *AREPs* from *f* and *g* and they have the same **HC2Ts**. If the *AREP* from *f* comes more early than *g*, node *b* will select *f* as the backup next hop and discard the *AREP* from *g*.

3.3 Maintenance of backup routes

When the backup route fails, the upstream node of the broken link should send *AERR* to claim that the other nodes should not use the broken link anymore. Actually, the function of *AERR* is the same as the RERR message. However, unlike the

broadcasting of RERR, the TTL of *AERR* is limited to one or two hops (the value is corresponding to the TTL in *AREQs*).

To avoid the heavy control overhead, the maintenance of backup routing table is based on the lifetime control. Both **Vectors** and backup routing entries have lifetimes. The expired **Vectors** are deleted from the cache and the expired backup routes are marked as inactivated.

3.4 Range extension of AREQ

The propagation range of *AREQs* can be extended to two hops by controlling the **Hop Count** field. Every node in the primary route broadcasts *AREQs* to at most 2-hop away. The **HC2T** plus the **Hop Count** in the **Vectors** is used to determine the shorter route. For example in Figure 6, node *f* hears *AREQs* from *a*, *b* and *c*. Node *f* will select *c* as the backup next hop because the value of **HC2T** plus **Hop Count** in the **Vector** is the smallest one. Subsequently, it will send *AREPs* to *e* and *b* since their **HC2T** plus **Hop Count** are larger than *c*. The range extension of *AREQ* will broaden the length of backup routes. However, if the total length of the backup routes is too large, the delay of data packets will increase.

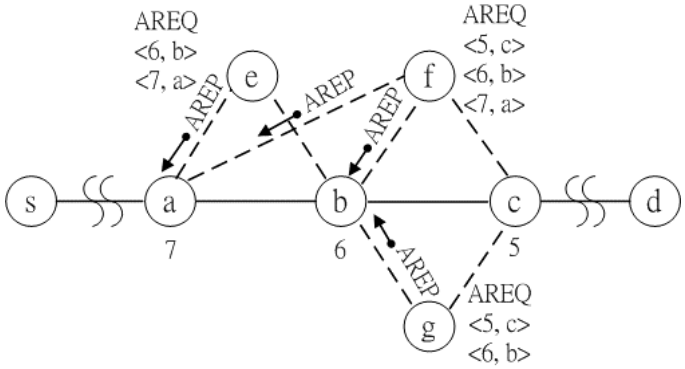

Fig. 3. The primary route and backup routes in the DARR

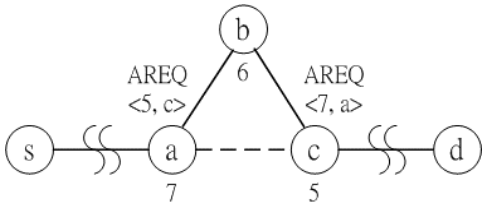

Fig. 4. The shortcut in the primary route

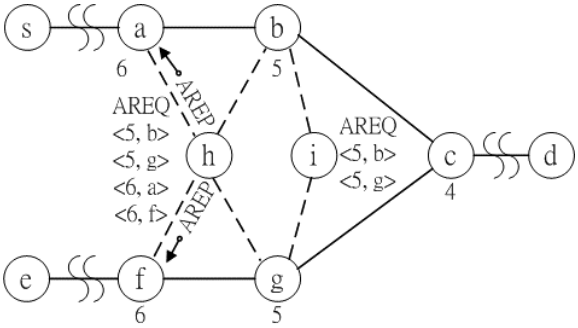

Fig. 5. The overlapping neighbor nodes between multiple primary routes

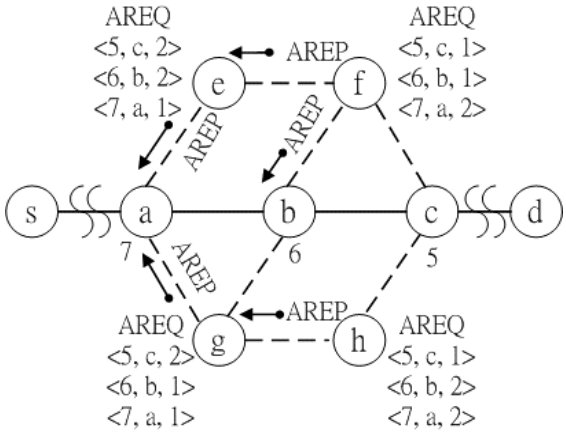

Fig. 6. Backup Routes with 2-hop AREQs

4 Performance evaluation

In order to evaluate the improvement of performance made by the DABR protocol, we compare the AODV and the DABR via simulation.

4.1. Simulation environment

The simulation is based on the GloMoSim [10] which is a network simulation library built by the PARSEC [11]. The PARSEC is a language for parallel execution of discrete-event simulation.

Initially, mobile nodes are uniform distributed within the simulated terrain of 1000 * 600 meter^2. The 802.11 MAC protocol is utilized and the transmission range of nodes is about 180 meters. The mobility model is the random way-point [2], i.e., every node selects a random target location and moves to the target with a fixed

speed. After arriving at the target, the node pauses for a period of time and randomly selects the next target to move to. In this simulation, the pause time of nodes in all experiments is set to zero.

In order to focus on the effects of routing protocols in network layer, the traffic pattern of application layer in simulation is CBR (Constant Bit Rate) under UDP. When the simulation starts, CBR clients and CBR servers are randomly assigned and will not change through the experiment. The connections will last till the end of simulation. The item size of CBR is 512 Bytes and the time intervals between the items to be sent are from 0.5 to 0.9 second. The delivery rate of IP packets is measured by counting the sending and receiving items. The ratio of all received packets in destination nodes over all sending packets of source nodes is the delivery rate of data packets. We also measure the control overhead by counting the number of control packets which are delivered from all nodes in the network. Besides the control packets of AODV (i.e., RREQ/RREP/RERR), control packets of DABR include AREQ and AREP. The propagation range of both AREQ and AREP is 1-hop. The simulated time in all experiments is 10 minutes.

4.2 Simulation results

The simulation results in Figure 7 show that the delivery rate of data packets in DABR is higher than AODV. The IP packets will be dropped if there are neither normal routes nor backup routes. The different speed of nodes is specified in the x-axis. When the nodal mobility is high (i.e., the speed of nodes is high), the improvement of DABR is excellent.

The control overhead is the number of control packets sent by nodes in the network. As shown in the Figure 8, the DABR incurs more control overhead than AODV, however, the additional overhead is almost constant. Although the data packets which encounter the broken link are salvaged by redirecting them to the backup routes, the RERR messages should be sent back to the source node in the same time. The source node should discover the newest routes to avoid the large length of backup routes.

5 Conclusion

In this paper, we develop the DABR protocol to enable backup routing in the AODV. The simulation results show that the additional overhead of DABR is almost constant. However, the gain of packet delivery rate grows with the nodal mobility. We conclude that the DABR is a simple and overhead controlled backup routing protocol which outperforms the AODV in link reliability, especially in the network with high nodal mobility.

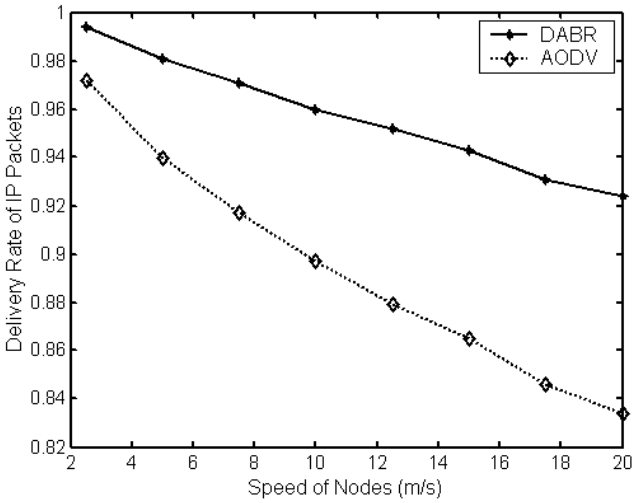

Fig. 7. The delivery rate of data packets (5 connections in 60 nodes)

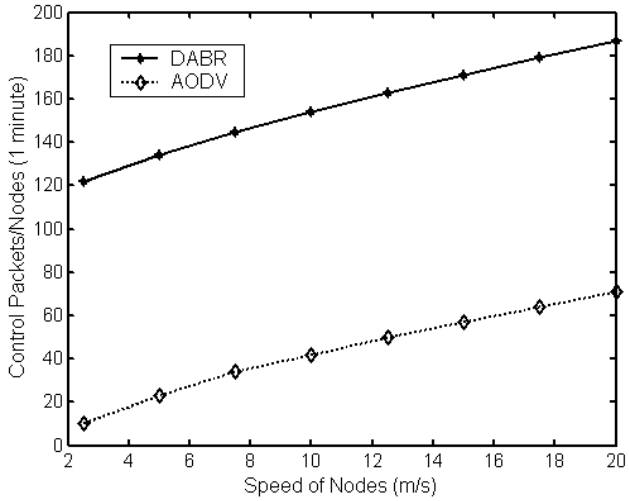

Fig. 8. The control overhead per nodes in one minute (5 connections in 60 nodes)

References

1. C. E. Perkins, E. M. Belding-Royer and S. R. Das, “Ad hoc on-Demand Distance Vector (AODV) Routing,” *Internet Draft* (work in progress), draft-ietf-manet-aovd-12.txt, 4 Nov. 2002.
2. D. B. Johnson, D. A. Maltz, Yih-Chun Hu and J. G. Jetcheva, “Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR),” *Internet Draft* (work in progress), draft-ietf-manet-dsr-07.txt, Feb 2002.
3. G. Pei, M. Gerla, and T.-W. Chen, ”Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks,” in *Proc. IEEE ICC 2000*, vol. 1, 2000, pp. 70-74.
4. T. Clausen et al, “Optimized Link State Routing Protocol,” *Internet Draft*, (work in progress) draft-ietf-manet-olsr-07.txt, Dec. 2002.
5. C.E. Perkins and P. Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers,” in *Proc. ACM SIGCOMM’94*, vol. 24, Oct. 1994, pp. 234-244.
6. C.-K. Toh, “Associativity-Based Routing For Ad Hoc Mobile Networks,” *Wireless Personal Communications Journal, Special Issue on Mobile Net-working and Computing Systems*, Kluwer Academic Publishers, vol. 4, no. 2, Mar. 1997, pp. 103-139.
7. S.-J Lee and M. Gerla, “AODV-BR: Backup Routing in Ad hoc network,” in *Proc. IEEE WCNC 2000*, vol. 3, 2000, pp. 1311-1316
8. M. Spohn and J. J. Garcia-Luna-Aceves, “Neighborhood Aware Source Routing,” in *Proc. ACM MobiHoc 2001*, 2001, pp. 11-21.
9. W.-P Chen and J. C. Hou, “Dynamic, Ad-hoc Source Routing with Connection-Aware Link-State Exchange and Differentiation,” in *Proc. IEEE Globecom’02*, vol. 1, 2002, pp.188-194.
10. X. Zeng, R. Bagrodia and M. Gerla, "GloMoSim: a Library for Parallel Simulation of Large-scale Wireless Networks," in *Proc. IEEE PADS 98*, May 1998, pp. 154-161.
11. R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and H.Y. Song, “PARSEC: A Parallel Simulation Environment for Complex Systems”, *IEEE Computer*, vol. 31, Oct. 1998, pp.77-85.
12. S. -J. Lee and M. Gerla, “Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks,” in *Proc. IEEE ICC 2001*, vol. 10, 2001, pp. 3201-3205.
13. M. R. Pearlman and et al, “On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Network,” in *Proc. ACM MobiHoc 2000*, 2000, pp. 3-10.
14. P. Pham and P. Perreau, “Multi-path Routing Protocol with Load Balancing Policy in Mobile Ad Hoc Network,” in *Proc. IEEE MWCN 2002*, 2002, pp. 48-52.