Wireless Netw
DOI 10.1007/s11276-009-0203-0

A highly topology adaptable ad hoc routing protocol with complementary preemptive link breaking avoidance and path shortening mechanisms

Rei-Heng Cheng · Tung-Kuang Wu · Chang Wu Yu

Abstract Ad-hoc on-demand distance vector routing (AODV) is a well-known routing protocol for mobile ad hoc networks. The original AODV protocol works in a semi-dynamic fashion, by establishing a route on demand and using that route until it breaks. However, to suit the changing network topology of ad hoc networks, more aggressive and adaptable routing strategies are required. A number of researches have proposed improving AODV performance by locally repairing broken links, predicting and replacing potentially vulnerable links, or shortening a link through removing redundant nodes from the transmission path. Although local repair may relieve some problems, it usually results in longer paths and thus a considerable performance drop in heavy traffic conditions. There are also issues regarding packet loss and communication delay due to route rebuilding once the link is broken. Predicting and replacing potentially vulnerable links may require special hardware, additional tables to maintain, or other extra overhead. Finally, path shortening may result in shorter and more efficient routes, but there is no guarantee that the new paths will be robust. This paper proposes integrating preemptive link breaking avoidance and path shortening mechanisms into a modified AODV protocol. However, the difficult issue lies in determining the right timing to initiate the two independent mechanisms so that the two dynamically and complementarily operating mechanisms can work together to improve the routing performance. Through numerical analysis and simulation, we have arranged a simple parameter setting for controlling the activation of each mechanism at the appropriate time. The proposed combination is a highly dynamic ad hoc routing protocol that is capable of adapting itself to the changing network topology and achieving extremely good performance in various routing performance metrics. Extensive simulations show that each of the two schemes alone improves AODV performance. More importantly, the integrated protocol performs even better in terms of data delivery rate, average delay time, and network overhead. To be more specific, in the best cases our protocol can reduce up to 82% in control overhead and 66% in delay time, while achieving 12% more in data delivery rate comparing to AODV.

Keywords Ad hoc networks · Dynamic routing · Dynamic link breaking avoidance · Dynamic path shortening · AODV

R.-H. Cheng
Department of Information Management, Hsuan Chuang University, Hsin-Chu, Taiwan, ROC
e-mail: rhc@hcu.edu.tw

T.-K. Wu (✉)
Department of Information Management, National Changhua University of Education, No. 2, Shi-Da Rd, Changhua City, Taiwan, ROC
e-mail: tkwu@im.ncue.edu.tw

C. W. Yu
Department of Computer Science and Information Engineering, Chung Hua University, Hsin-Chu, Taiwan, ROC
e-mail: cwyu@chu.edu.tw

1 Introduction

Mobile ad hoc networks are wireless networks with no fixed infrastructure. They can be characterized by dynamic topology due to node mobility, limited battery power and limited bandwidth. Each mobile node in ad hoc networks may move arbitrarily and act as both a router and a host. Many routing protocols for ad hoc networks have been

Journal : Large 11276 | Dispatch : 11-8-2009 | Pages : 23
Article No. : 203 | ☐ LE | ☐ TYPESET
MS Code : WINET-683 | ☑ CP | ☑ DISK

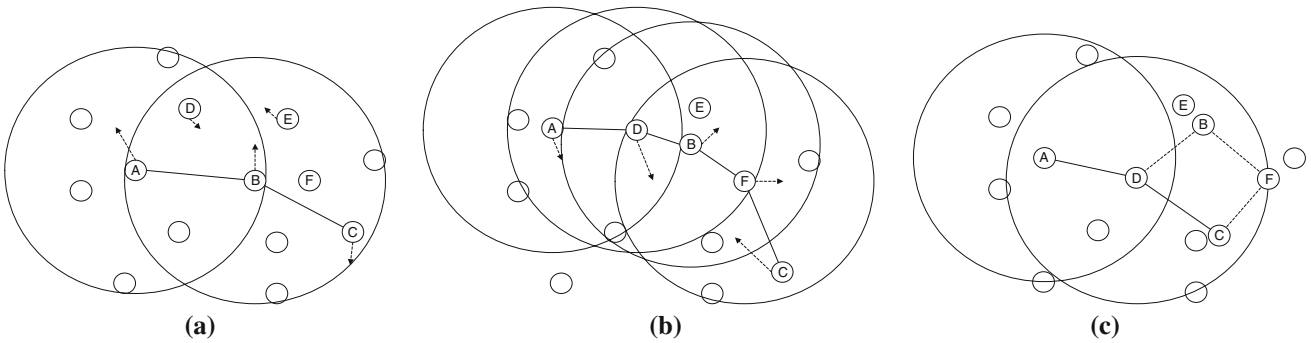


Fig. 1 An adaptable routing protocol that may dynamically adjust itself to the changing network topology (the *dashed arrows* indicate the moving directions of related nodes). (**a**) A route *A-B-C* is established on-demand, (**b**) node *D*, *F* are added in between link *A-B* and *B-C* to keep the route from breaking, (**c**) a route *A-D-B-F-C* is replaced (*shortened*) by a new route *A-D-C* with less intermediate nodes to increase communication efficiency

developed, such as AODV [1], DSR [2, 3], TORA [4] and DSDV [5]. These routing protocols are usually categorized into two types: table-driven and source-initiated on-demand protocols [6]. The table-driven protocols, such as DSDV, always try to maintain the routes in the network, so that a transmission route can be established immediately for data packet delivery. However, protocols in this category suffer from heavy overhead resulting from frequent routing table updates. The source-initiated on-demand protocols initiate a route request only when there is a need for transmission. They save overhead due to periodic routing information refreshing, but suffer from longer latency before the actual data delivery can start.

Among all routing protocols, AODV has been known for its outstanding performance [7, 8]. However, routing paths established with AODV remain unchanged during data transmission unless the links fail. Due to the frequently changing network topology of an ad hoc network, two possible scenarios may occur as a result. First, some intermediate nodes from an earlier established path may later become redundant, resulting in packets routing through unnecessarily longer paths. Secondly, a path established at one time may later be broken due to some intermediate nodes from a path moving out of range.

With respect to the first scenario, some researches worked on shortening the routing path by reducing the number of intermediate nodes in between the source and destination [9–11]. However, this also implies that the distance between two adjacent routing nodes increases, which may also result in a link that is more vulnerable to breaking. Accordingly, there were also studies that worked on developing link breaking avoidance protocols [12, 13] or searching for backup routes for the broken links [14–17]. Unfortunately, doing so usually means that packets must go through more hops via the repaired route, which may also induce congestion issues since some intermediate nodes could be involved in many concurrent communications [18].

Figure 1a presents a scenario from the above description in which link *A-B* is on the verge of breaking. In such a case, applying the link shortening mechanism may not stop potential link breaking. Preemptive link breaking avoidance may save the link, but in the long run could result in a route with too many intermediate nodes and thus increase collisions and node loading which again reduces the communication efficiency (see Fig. 1b). Apparently, a more appropriate mechanism would be one that is capable of adapting the main routes to the changing topology. To be more specific, a good ad hoc routing protocol should add intermediate nodes before links break (preemptive link breaking avoidance) to increase route stability, while also dynamically adjusting route lengths (path shortening) to improve communication efficiency. Figure 1b and c show such an idea.

The above two mechanisms, path shortening and preemptive link breaking avoidance, work in two quite opposite ways, yet may be complementary to each other. It would be interesting to combine the two ideas and see how they interact and perform. To the best of our knowledge, there is no related study devoted to finding this answer. In this work, we propose integrating the dynamic link breaking avoidance and path shortening features by arranging a pair of parameters for controlling the activation of each mechanism using theoretical analysis and a few modifications to the original AODV. These modifications include nodes turning on their overhearing functions and maintaining some extra information. The resulting new protocol is capable of adapting to the changing network topology and then adjusting the communication path accordingly.

Extensive simulations show that each of the two schemes alone improves AODV performance. More importantly, the new integrated protocol that we developed

Journal : Large 11276 | Dispatch : 11-8-2009 | Pages : 23
Article No. : 203 | LE | TYPESET
MS Code : WINET-683 | CP | DISK

performs even better than other major protocols in terms of data delivery rate, average delay time and network overhead. Note that a preliminary idea for this work was appeared in [19].

The rest of this paper is organized as follows. Section 2 describes previous researches that are related to our work. Section 3 presents the details of the proposed scheme with the theoretical analysis and simulation results shown in Sects. 4 and 5, respectively. Finally, Sect. 6 concludes this work and lists future research directions.

2 Related work

Ad hoc on-demand distance vector (AODV) [1] is a well-known on-demand routing protocol, which establishes a route only when a source requires to send messages to some destination without requiring periodic update of routing information. In case of link broken scenario, a broken link notification is sent, which triggers source node to restart the route discovery process.

Dynamic source routing (DSR) [2, 3] uses source routing, with each packet to be routed carrying in its header the complete, ordered list of nodes that the packet must go through. If a link fails, the upstream node of this failed link sends a route error packet to the source node. When a route error is received, the hop in error is removed from this host's route cache and all routes that contain this hop must be truncated at that point. A new route discovery process must be initiated by the source. A salvaging technique proposed by Johnson et al. [3] uses some alternate routes from the cache when a data packet reaches a failed link on its source route.

As we have seen in the above descriptions, the two major on-demand protocols (AODV and DSR) do not respond quickly enough to link failure. They usually suffer from the risk of flooding the whole network for new route discovery. Consequently, many researches proposed maintaining backup routes in advance so that route switching can be fast and thus delay can be minimized. Lee and Gerla [14] proposed a so called AODV-BR protocol that uses nodes that are one hop away from the main route to form the backup routes. Lai, Hsiao and Lin [20] modified the AODV-BR algorithm [14] and proposed two improved algorithms, AODV-ABR and AODV-ABL, to enhance the adaptability of AODV protocol to the topology-changing environment. Overhearing function also needs to be turned on so that nodes around the main route may be able to learn whichever nodes on the main routes that are within their power range. In case a link A-B were broken, the upstream node A would send a one-hop BRRQ (Backup Route ReQuest) and await for neighboring nodes' replies (BRRP: Backup Route RePly) regarding their hop counts to the destination. The upstream node can then select an appropriate detouring node with the least hop count. However, the protocols do not work preemptively. In addition, the average hop count of routes in the two protocols may add up as the protocols proceed. Chung et al. [15] proposed the ad hoc backup node setup routing protocol (ABRP) that is similar to the DSR. ABRP saves backup route information in certain on-the-route node. Similarly, ABRP does not update its backup route information to reflect the network topology change. Agarwal and Jain [16] proposed a modified AODV protocol in which a node records information from its two upstream nodes. After some evaluation, one of the upstream nodes will be included in the main route with the other being used for the backup route. Chen and Lee [17] proposed a 2HBR protocol that extends the idea in [14] further by including nodes that are two hops from the main route as the potential backup nodes. However, it is questionable whether the pre-established backup routes would ever be used, not to mention that these backup routes may not be available when they are actually needed due to the mobile nature of nodes in ad hoc networks. Sengul and Kravets [21] proposed a local recovery algorithm and combined that with the DSR protocol. In case of link breakage and no alternative route in the route cache, the protocol issues a local recovery request looking for potential node for path detour. If any packet successfully reaches the destination through the newly discovered route, the destination node would then notify the source node for such a route change by returning a notification message.

There were also a number of route repairing mechanisms proposed. These protocols try to use local repair to avoid possible RREQ flooding in rebuilding a new route, which can be effective in avoiding a broadcast storm and serious communication delay, especially in a large-scale network. Protocols that fall into this category are given below. Lee, Royer and Perkins proposed to repair a link error by requiring upstream node of the broken link broadcast a RREQ to the destination. The source does not need to be informed and the data delivery would not be interrupted. Unfortunately, the process may result in flooding RREQ messages to the entire network. [22]. Castañeda et al. [23] proposed two heuristics that utilize prior routing histories to localize the query floods to a limited region of the old routes and the dynamically collected information to repair the broken route. Yu et al. [18] proposed a dynamic route repairing protocol that repairs a broken route using information provided by nodes overhearing the main route communication. When links go down, their protocol intelligently replaces these failed links or nodes with backup ones that are adjacent to the main route. Soliman and Al-Otaibi [24] proposed an algorithm to enhance the local repair (LR) phase of AODV by using a preemptive mechanism to detect potential link failures and

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

to find in advance some alternative links. By eliminating the use of regular and costly control massages such as RREQ, RREP, and RERR, their protocol is able to reduce substantially the control overhead generated by the repair process. However, to achieve that, their algorithm needs to maintain a Neighbor's Activity Table (NAT) recording information located two-hop away. The maintenance of such a NAT depends on nodes of an active route monitoring activities of the other active routes, which makes their protocol sensitive to the number of conversations. To be more specific, in cases of fewer conversation-pairs, the NAT table may not be able to populate enough alternative links for future path detouring. On the other hand, the protocol may perform well as compared to similar protocols in case of higher loading condition. While local repair may have relieved some problems, there are still issues regarding to packet loss and communication delay due to route rebuilding once the link is broken, or performance drop in heavy traffic condition [18].

Some researchers thus proposed various preventive mechanisms that would switch to a new backup route prior to possible route breaking. Among these protocols, S. Crisostomo et al. [12] proposed to use nodes' position and mobility information, and apply such information to predict vulnerable links and potential backup nodes. Special hardware like global positioning system (GPS) is required to assist the implementation of such protocol. Srinath et al. [13], on the other hand, proposed to maintain the so called neighbor power list and power difference table so as to initiate backup route switching mechanism when signal strength between two adjacent nodes in an active link is weakening. However, maintaining two tables requires periodical HELLO messages interchange, and thus imposes a considerable burden on the network. Tsai et al. [25] proposed keeping track of the signal to noise ratio (SNR) of links so that a new route request would be initiated once the SNR is below some pre-defined threshold to reduce the possibility of link breaking. However, in mistaken prediction cases, their method may result in additional route re-establishment. Even if the prediction is correct, local route repair may well be sufficient instead of complete new route construction. Two independently proposed protocols [26, 27] make use of link status information of routes that is piggybacked in the RREP packets to notify the source so that it can plan for the right timing to route reconstruction. In the meantime, nodes on the route also need to perform link quality monitoring and confirm that with PING-PONG process for possible link breakage. As soon as a potential link breakage is identified, the source will be warned so that it can preemptively initiate route reconstruction procedure. Note that there always exists the possibility of false identifying link breakage. In their cases, it can result in unnecessary route reconstruction and may possibly cause the broadcast storm in heavy traffic condition.

In addition to the route backup/repair or link breaking avoidance protocols, there are also some researches that focusing on path shortening. DSR, for example, stores the full routing path in the source node. When a node overhears a packet carrying a source route, it examines the unexpended portion of that source route. If this node is not the intended next-hop destination for the packet, and it is named in the later unexpended portion of the packet's source route, it can then infer that the intermediate nodes before itself in the source route are no longer needed in the route [3, 8]. This mechanism is based on the extra routing path information piggybacked in the transmitted packets, and the node has to reply a so-called *gratuitous* RREP to the original sender of the packet for shortening routes. However, it is not efficient because the reply may take many hops to reach the source node. Saito et al. [9] presented a proximity-based dynamic path shortening scheme. In this scheme, each node in the active route monitors its local link quality and estimates whether to enter the proximity of its neighboring node to shorten the active route.

Both of the methods mentioned above are only concerned with whether some of the intermediate nodes should be removed from the active route to reduce the number of routing hops. However, they do not consider the possibility of replacing two (or more) intermediate nodes with a node that may happen to be in the right position.

Roy [11] presented a source-tree on-demand adaptive routing protocol (SOAR). In SOAR, wireless routers exchange minimal source trees consisting of the state of the links that are in the path used by the router to reach active destinations. Based on this information, SOAR can find the shorter route to replace the active paths. Just like the source route information in DSR, the minimal source tree information in SOAR puts an extra transmission burden on the network.

Gui and Mohapatra [10] proposed a so called SHORT protocol that would look for possible shortcut by piggybacking hop counts information (from the source node) within packets and making use of the overhearing mechanism using nodes around the active route. Except for three special cases, their protocol can successfully achieve the goal of shortening the communication path. However, the protocol lacks the capability of avoiding possible route breaking as a result of route shortening. In addition, up to three special cases were left unsolved with their protocol.

Finally, M. G. Zapata [28] proposed a protocol that adopts shortcut finding and route repairing mechanisms. However, the two mechanisms work independently and lack the complementary feature. In addition, their route repairing mechanism is initiated after a link is broken,

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

which may induce additional communication delay and packet loss as stated earlier. Instead of using packet overhearing, their shortcut finding mechanism requires nodes on the active route to issue periodic shortcut RREQ and thus imposes extra burden on the network.

3 The proposed scheme

In this work, we propose a protocol that dynamically activates link breaking avoidance procedure, which we call *Dynamic Link Breaking Avoidance* (*DLBA*), by inserting intermediate nodes to potential vulnerable links, and initiates path shortening mechanism, which we call *Dynamic Path Shortening* (*DPS*), if there is some shortcut available for the active route. However, for the two independent and complementary mechanisms to cooperate seamlessly together so as to improve the routing performance, the difficult issue would be determining the right timing to initiate each of the two. Fortunately, with careful design, the right timing can simply be controlled using preset threshold values for the two parameters. On the other hand, it may also be questionable whether the gain in routing performance is worthwhile with all these efforts. In fact, our simulations indicate that the proposed protocol, in almost all cases, achieves the best performance in terms of data delivery rate and communication delay, but with much lower control overhead compared to the other conventional ad hoc routing protocols. We will have an in-depth discussion of the above issues in Sects. 4 and 5. In the remaining portion of this section we will first present details of DLBA, DPS, and some implementation issues.

Note that our proposed dynamic and adaptive ad hoc routing mechanisms operate on top of a modified AODV protocol. In AODV, the hop counts to the destinations are kept in mobile nodes of the active routes. Since a mobile node located near a data transmission path may overhear packet in transmission, it should be able to determine if it can act as a detour node for the original path and create a shorter path by piggybacking the hop count information within the data packets. In case a node does find itself a suitable detour node, it can contact the related nodes, its potential upstream and downstream nodes, to direct/receive traffic to/from it. To reduce the number of control packets (to be discussed later in Sect. 3.3), the hop count in our protocol is defined as the distance from the source, as opposed to AODV's convention. In other words, when a packet is first sent from the source node, its hop count value would be set to zero and incremented by one whenever it goes through an intermediate node. To avoid any confusion, we use the term *altitude* to represent the so called "hop count" hereafter. Accordingly, $altitude_X$ represents the altitude of node X, which also means that node X is the $altitude_X$-*th* node from the source (without counting the source). Figure 2 illustrates the definition of altitude in our protocol.

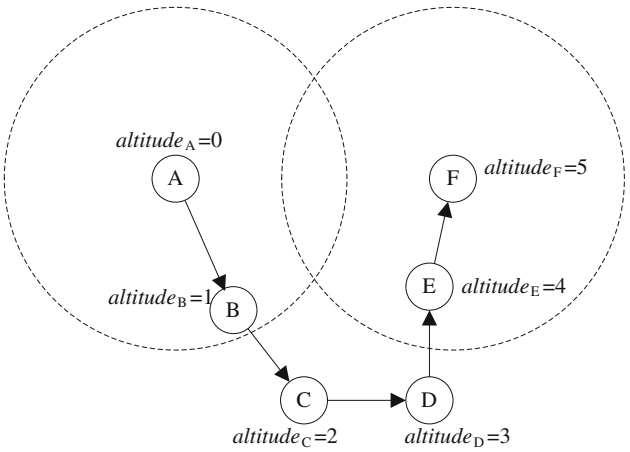


Fig. 2 An illustration of the definition of altitude (node *A* and *F* represent the source and destination, respectively)

3.1 Dynamic link breaking avoidance (DLBA)

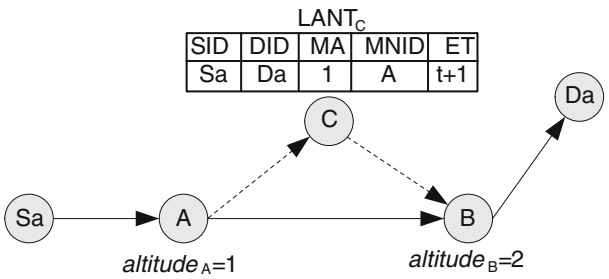
The DLBA mechanism operates by monitoring the quality of communication links and inserts an intermediate node in between the two end nodes of some breaking link. For example, the breaking link *A*-*B* in Fig. 3a is replaced by two links *A*-*C* and *C*-*B*. Link quality is determined by signal strength upon receiving a packet. In case the received signal power level is weaker than some predetermined threshold, which indicates that the distance between the two end nodes may be longer than some threshold, the DLBA procedure is then activated. Here we assume that the signal strength can be used to estimate the relative node distance [29].

With the proposed DLBA, each node needs to record the minimal altitude that it overheard, which is maintained in an extra data structure called the *Lowest Altitude Neighbor Table* (LANT). The structure of each row in the table is depicted as shown in Fig. 3b.

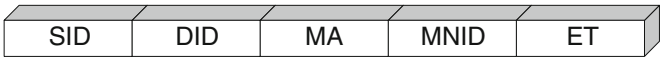
When a node receives or overhears a packet, it extracts the *Source ID*, *Destination ID*, *Sender ID* and Sender's altitude information from the packet. In case there is a valid record in its *LANT* with the <*SID*, *DID*> pair matches the one just received, a check is performed to see if the related fields (*MA*, *MNID*, and *ET*) should be updated. Otherwise, the received information is inserted into the *LANT* as a new record. The Expiration Time, *ET*, is set to the current time plus some period of time fixed to 1-s in our simulation.

For example, let refer to Fig. 3a and suppose at some specific time *t* node *A* forwards a packet, which was originated from node *Sa* and destined for node *Da*, to node *B*. In case node *C* overhears such a packet and adds to its

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK



(a)



SID: Source ID
DID: Destination ID
MA: Minimal Altitude
MNID: The ID of the node with minimal altitude
ET: The Expiration Time of this record

(b)

Fig. 3 (**a**) The dynamic link breaking avoidance mechanism operates by inserting a new node (e.g., node *C*) into the path and replacing the breaking link *A*-*B* with new links *A*-*C* and *C*-*B*. (**b**) The data structure used by the dynamic link breaking avoidance mechanism for each node to record the minimal altitude it overheard

LANT table an entry as specified in Fig. 3a. In case the downstream node *B* receives a data packet from its upstream node *A* and notices that the signal strength is weaker than the predefined threshold value, it informs neighboring nodes to help node *A* build a new route by broadcasting a *help* packet with its altitude (e.g., 2) and a zero time-to-live value. A *help* packet is a newly defined control packet in our protocol that is similar to an ordinary AODV control packet (such as RREQ), except the packet type is changed to *HELP*. In case node *C* receives such a *help* packet from node *B*, it calculated the difference of altitude between that carried in the *help* packet and the minimal altitude information it keeps. If the difference is greater or equal to one, which indicates node *C* itself is a good detouring candidate for the link *A*-*B*, it may then activate the dynamic route detouring procedure. Node *C* then set node *B* as the next hop to destination *Da* and notifies node *A* to change its next hop to itself. The detouring procedure is thus successfully complete with node *C* inserted in between node *A* and *B*. The detailed DLBA algorithm is presented in Algorithm 1 .

```
1. When node x receives or overhears a data packet or help packet P,
2. {
3.     If the receiving signal strength is lower than a preset threshold value
4.     {
5.         If the packet is directed to node x, broadcast a help packet to node x's
               neighbor
6.         Else discard the overheard packet
7.         Return
8.     }
9.     Extract the following information from the packet :
           <SourceID, DestinationID, SenderID, Altitude>
10.    Find the pair <SourceID, DestinationID> in LANT.
11.    If no matched entry found, store the related information into LANT.
12.    Else
13.    {
14.        Let the found entry be <SID_f, DID_f, MA_f, MNID_f, ET_f>
15.        If (Altitude-MA_f ≧ 1 and P is a help packet)
16.            Update node x's next hop to node SenderID
17.            Notify the upstream node MNID_f to update its next hop to node x
18.        Else if (Altitude < MA_f ) or (Sender ID == MNID_f)
19.            update the related fields of the found entry
20.    }
21. }
```

Link quality is the primary concern when determining whether a route should be detoured. The DLBA mechanism deals with such a quality issue by (1) monitoring the stability of an active link and taking proper measure whenever necessary, i.e., sending out *help* packet in case the received signal strength is weaker than a pre-determined threshold, and (2) carefully choosing the detouring nodes. To be more specific, if a mobile node overhears a data packet with weak signal strength, it indicates that the receiving node may not be close enough to or may be getting out of the power range of the path, and may not be a good candidate for a detour node. Accordingly, in our proposed scheme, a mobile node discards the overheard data packets with signal strengths lower than a predefined threshold value. If more than one node responds to the *help* packet, the upstream node chooses the one that arrives first and ignores the other respondents.

3.2 Dynamic path shortening (DPS)

The DPS mechanism operates by recognizing potential redundant intermediate nodes and either removes or replaces the nodes from the established path. There are two types of path shortening scenarios, type I and type II, considered in our DPS protocol:

1. With Type I scenario, some redundant nodes are directly removed from the active route (refer to Fig. 4a). Type I operation occurs when a downstream node in an active route (e.g., node *D*) can overhear the packets sent from its upstream nodes that are more than one hop away (e.g., node *B*). In such case, the node(s) in between the upstream and downstream nodes (e.g., node *C*) can be removed from the active route.
2. With Type II scenario, some redundant nodes in the active route is replaced with a node that is not on the active route (refer to Fig. 4b). Type II operation occurs when a node (e.g., node *F*) can overhear packets sent from two nodes (e.g., node *B* and *E*) in an active route

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

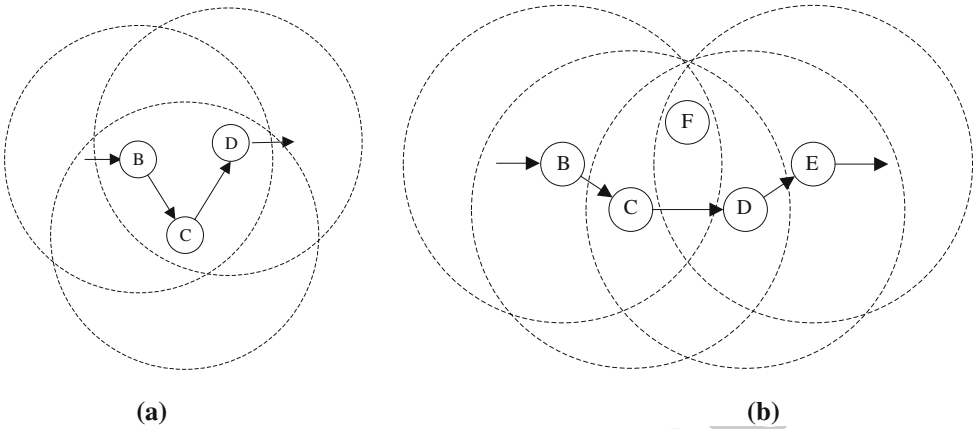


Fig. 4 (**a**) Node *D* can overhear packets from node *B* and *C*, (**b**) node *F* can overhear data packets coming from node *B*, *C*, *D* and *E*

and the altitude difference between these two nodes is greater than two.

Similar to the DLBA mechanism, upon receiving or overhearing a packet, a node under DPS mechanism extracts the *Source ID*, *Destination ID*, *Sender ID* and Sender's altitude information from the packet, and checks if the path shortening mechanism should be activated. The detailed DPS algorithm is presented in Algorithm 2, with conditions of type I and II scenarios checked in line 13 and 15, respectively .

```
1. When node x receives or overhears a data packet,
2. {
3.      If the receiving signal strength is lower than a preset threshold value
4.      {
5.          Discard the packet
6.          Return
7.      }
8.      Extract the following information from the packet:
          <SourceID, DestinationID, SenderID, Altitude>
9.      Find the pair <SourceID, DestinationID> in LANT.
10.     If no any match entry found, store the related information into LANT.
11.     Else
12.     {  Let the found entry be <SIDf, DIDf, MAf, MNIDf, ETf>
13.        If  (Altitude–MAf ≧ 1) and (the packet is directed to node x)
              //A Type I shortcut is found
14.           Notify the upstream node MNIDf  to update its next hop to node x
15.        Else if (Altitude-MAf > 2)
              //A Type II shortcut is found
16.           Update node x's next hop to node SenderID
17.           Notify the upstream node MNIDf  to update its next hop to node x
18.        Else if (Altitude < MAf ) or (SenderID == MNIDf)
19.           update the related fields of the found entry
20.     }
21.}
```

With type II detouring, the decision is made according to whether the signal quality of the overheard packets is above some predefined threshold and the difference in altitude value contained within the packets meets the criteria. In case both conditions are satisfied, the substituting node can go ahead and update its next routing node information. However, both of the type I and II detouring cases require notifying the upstream node to change its next hop to the substituting node. In the meantime, the upstream node would determine whether to accept the detouring request by evaluating the signal quality of the received request packet.

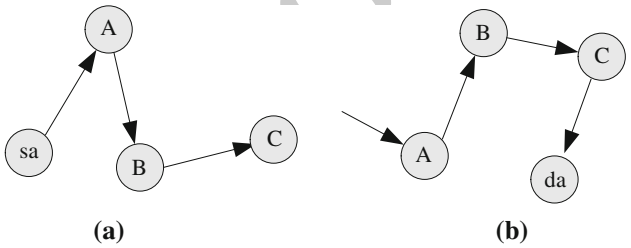


Fig. 5 Two special cases with shortcut detection involves: (**a**) the source node, (**b**) the destination node, that were not taken care of in [10]

Figure 5 illustrates two special cases that involve either the source or destination node of an active route. Both cases cannot be handled by the shortcut detection method proposed in [10]. The two cases apparently match the type I scenario and can be successfully dealt with using our method. Take Fig. 5a as an example, as node *B* is within the power range of both source node (*sa*) and node *A*, it can overhear packets from both nodes. Node *B* would first record the altitude value of *sa*, which is zero. When node *A* redirects the packet, now with altitude 1, to node *B*, which makes the difference in altitude contained in the packet from node *A* and that recorded in node *B* to be 1. The scenario meets the conditional test specified in Line 13 of Algorithm 2 and the shortcut that goes directly from source node *sa* to node *B* is successfully recognized. Similarly, in Fig 5b, if the destination node *da* is able to overhear packets from both node *A* and *C*, the shortcut between node *A* and *da* can also be detected.

3.3 Implementation issues

In this section, we will discuss three implementation issues of our proposed protocol, which include altitude information maintenance, race conditions and routing loops that may occur in distributed environment like ad hoc networks.

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

3.3.1 Altitude information maintenance

The path shortening may be achieved by removing some redundant nodes from the route (Fig. 4a) or replacing two or more nodes in the active route with a substituting node (Fig. 4b). In both cases, the altitude information of all affected nodes along the route has to be recalculated. In our proposed protocol, a node calculates its altitude by simply adding one to the altitude value piggybacked in the received data packets originated from its upstream node. For example, once node *B* in Fig. 6 sends data packets to node *F* after the path was reorganized and shortened, node *F* finds out the altitude of *B*, which is two in this case, from the received data packets, and then knows that three would be the new altitude piggybacked in the data packets to its direct downstream node. The altitude of node *E*, and all subsequent downstream nodes, will adjust their altitudes accordingly. As we can see, this process is simple and straightforward. On the other hand, if the hop count information is recorded with respect to the destination as AODV does, the adjusted hop count information of the nodes in a newly established path cannot be determined until the reply control packets sent back from the destination node. This can be easily seen in Fig. 7, in which partial route *B-C-D-E* is replaced with *B-F-E* and the hop counts (to the destination) of all the nodes in the route (node *sa*, *A, B, F* and *E*) have to be determined and re-assigned according to the reply messages from node *da*.

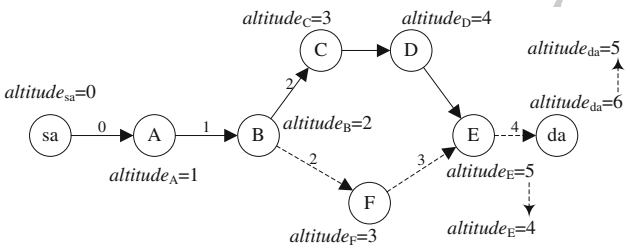


Fig. 6 The altitude information updating sequence (depicted in *dashed lines*) after route *B-C-D-E* is changed to *B-F-E*

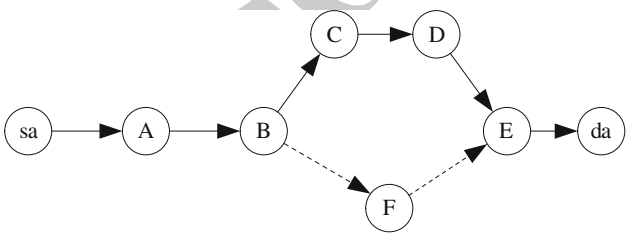


Fig. 7 The hop counts of all the nodes in the route (*sa*, *A, B, F* and *E*) have to be re-assigned in case the hop count is defined with respect to the destination instead of from the source

3.3.2 Race condition

The other concern one may have is the race condition, which results from multiple mobile nodes finding new shorter paths almost at the same time and asking their upstream nodes to detour simultaneously. In this subsection, we give a formal discussion explaining why there always exists a directed path from the source to destination node under multiple DPS activations with the following two assumptions:

Assumption (1): We assume that, when multiple DPSs are applied, no link in the main route and detour route breaks due to node mobility.
Assumption (2): The altitude weights of nodes on the main route are updated without delay. When a node is removed from the main route, it can clear all the altitude information stored in it immediately.

Note that the above two assumptions may fail in case when nodes move very quickly. However, what we concern here is whether the breakage of an active route from the source to destination node is a result of multiple activations of DPS. For example, an activation of DPS may replace a route with one or two links, which may be robust at the moment the decision is made but soon be broken during the route switching process due to some intermediate node moving away. It is evident that DPS itself does not cause the breaking of the new substitute link.

In the following we will first describe the intuitive idea by illustrating scenarios (Fig. 8) when two mobile nodes finding new shorter paths at the same time, followed by a more formal discussion.

Suppose the starting and end nodes of the two potential partial routes to be replaced by the new shorter paths are (*A, B*) and (*C, D*), respectively. The relationship between the two partial routes, as shown in Fig. 8, can be one of four following scenarios:

a. The two partial routes, e.g., (*A, B*) and (*C, D*), are completely independent to each other (see Fig. 8a).
b. One partial route, e.g., (*C, D*), is contained within the other partial route, e.g., (*A, B*) (see Fig. 8b).
c. The two partial routes are exactly the same, e.g., $A = C$ and $B = D$, so eventually there is only one partial route replaced (see Fig. 8c).
d. The two partial routes are partly intersected, e.g., part of the partial route (*C, D*) is within the other partial route (*A, B*) (see Fig. 8d).

Apparently, the former two scenarios do not cause any problem in term of race condition. The third scenario occurs when two downstream nodes ask the same upstream node for detouring, e.g., both node *F* and *G* ask node *A* to direct traffic to them as illustrated in Fig. 8c. Depending on

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

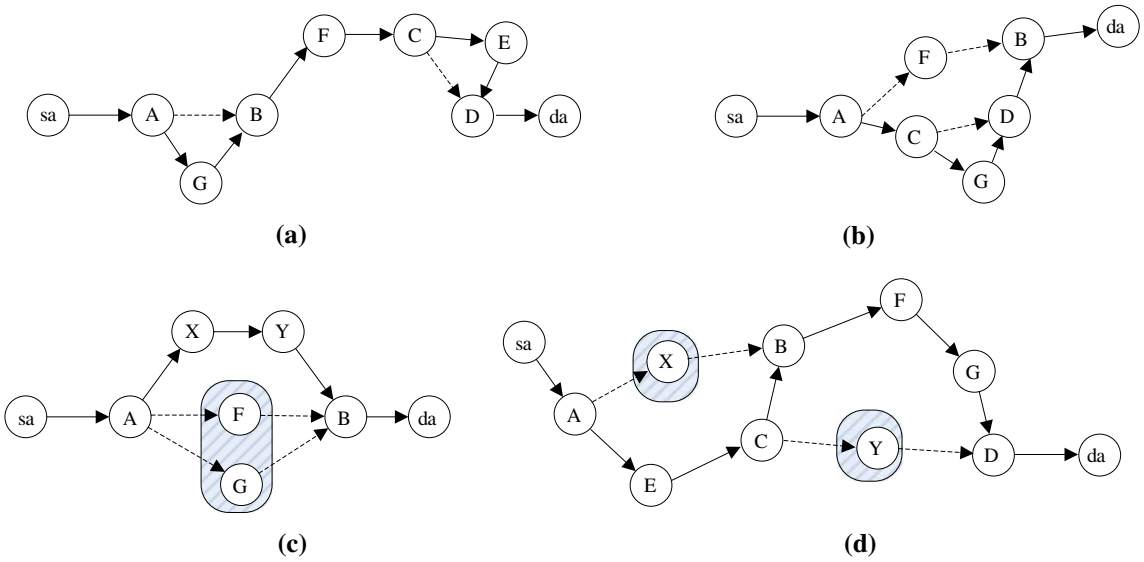


Fig. 8 Illustrations of the relationship between two partial routes which could be replaced by DPS at almost the same time (the original paths and new short cuts are represented by *solid and dashed lines*, respectively), (**a**) partial route *A*(-*G*-)*B* and *C*(-*E*-)*D* are disjoint, (**b**) partial route *C*(-*G*-)*D* is contained within partial route *A*(-*C*-*G*-*D*-)*B*, (**c**) only one partial route *A*(-*X*-*Y*-)*B* is replaced (two nodes inform the same upstream node to detour), (**d**) partial route *A*(-*E*-*C*-)*B* and *C*(-*B*–*F*-*G*-)*D* are intersected (two nodes inform two different upstream nodes to detour)

which node initiates the request first, the final route would either be *A*-*G*-*B* or *A*-*F*-*B*. In either case, a shortened path is established without any problem, either. The fourth scenario occurs when two potential downstream nodes ask two different upstream nodes for detouring, as depicted in Fig. 8d, where node *X* asks node *A* while node *Y* asks node *C* to detour, respectively. Since packets would go directly through node *X* instead of the new link that consists of node *Y*, the latter detouring has no effect at all. As a result, the other newly established route, *sa*-*A*-*X*-*B*-*F*-*G*-*D*-*da*, would be a valid one. In other words, even with various race conditions, our proposed path shortening scheme can still operate correctly.

A formal discussion of the above examples is given below. Suppose that the current main route P from source p_0 to destination p_k is a route with length k; that is, $P = p_0, p_1, p_2 \ldots p_k$ where p_i is upstream node of p_{i+1} for $0 \leq i \leq k-1$. The route can be represented by a directed acyclic graph G with vertex set $\{p_0, p_1, p_2, \ldots, p_k\}$ and arc set $\{[p_0, p_1], [p_1, p_2], \ldots, [p_{k-1}, p_k]\}$. Note that for $0 \leq i \leq k-1$, there exists a directed path from p_i to the destination p_k.

Theorem 1 *Under the above two assumptions, there always exists a directed path from p_0 to p_k even when multiple DPSs are applied simultaneously.*

Proof Initially, there exists a directed path from each node in the main route to the destination node. When multiple DPSs are applied simultaneously, it is sufficient to show that there always exists a directed path from p_0 to p_k by tracing the downstream nodes from the source p_0, even though sometime the resulting routes form a directed tree toward the destination node after multiple DPSs has been applied.

On the contrary, suppose that p_0 is not connected to p_k. Only two possibilities need to be discussed:

1. Trap in a loop when tracing from p_0: Since every detour route is acyclic and the main route is also acyclic, the loop exists among detour routes and the main route. Since every detour route connects two nodes in the main routes, the loop must contain at least two nodes p_i and p_j with $i < j$ in the main routes. In the loop, node p_j must connect to p_i in a directed path via detour routes. However, it is impossible since all detour routes are disjoint from the main route, and connects from p_i to p_j with $j > i$, which is guaranteed by DPS. In DPS, the upstream node p_i will always replace a link $[p_i, p_{i+1}]$ with a directed acyclic path P' which is disjoint from the main route except the end node p_j with $j > i$ located in P; that is, P' contains arc set $\{[p_i, q_1], [q_1, q_2], \ldots, [q_l, p_j]\}$ where $\{q_1, q_2, \ldots, q_l\} \cap \{p_0, p_1, p_2, \ldots, p_k\} = \emptyset$.
2. Meet a node $R \neq p_k$ without a downstream node when tracing from p_0: there are two possibilities for node R. (1) R is in $\{p_0, p_1, p_2, \ldots, p_{k-1}\}$ or (2) R is in $\{q_1, q_2, \ldots, q_l\}$, which are nodes in detouring routes but not in $\{p_0, p_1, p_2, \ldots, p_{k-1}\}$. The second case is impossible since it violates the assumption that no link in the main route and detour route breaks due to node

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

mobility. For the remaining case, let R be p_i where $0 \le i \le k-1$. If p_i is not a detouring node, it must keep a downstream node p_{i+1} in the original route and a contradiction occurs. Otherwise, p_i is a detouring node. According to our assumption, there exists a directed acyclic path P' which is disjoint from the original main route and connecting to a node p_j with $j > i$ located in the main route, which also indicates R has a downstream node. A contradiction occurs. Finally, we obtain the desired result. ■

3.3.3 *Routing loop issue*

It can be seen from the deduction in previous section that neither race nor loop condition occurs with multiple activations of DPS. We can also prove that it is loop-free under Assumption (2) (listed in Sect. 3.3.2) when multiple DLBAs are applied.

Theorem 2 *DLBA is loop-free under Assumption (2)*

Proof Suppose that the initial main route is represented by a directed path $P = p_0, p_1, p_2 \ldots p_k$. At the start, we have that P is loop-free due to the design of AODV. Consequently, we have altitude $(p_i) <$ altitude (p_{i+1}) for $0 \le i \le k-1$, where altitude (p_i) indicates the altitude information of node p_i.

According to the design of DLBA, p_i (the downstream node of p_{i-1}) will issue a *HELP* packet when it receives a packet from p_{i-1} and finds the signal strength lower than the threshold value. In case a node q, which is not on the main route, overhears the *HELP* packet (indicating node q is located within the communication range of p_i), it initiates the route detouring procedure by marking itself on the path, finding an appropriate node p_r located in the main routing path (with altitude $(p_r) <$ altitude (p_i)), and constructing a path consisting of arcs $[p_r, q]$ and $[q, p_i]$ to replace the partial route from p_r to p_i (including the breaking link p_{i-1} to p_i). After that, lock node p_r to prevent additional detour for a while. All the nodes in the replaced partial route from p_{r+1} to p_{i-1} clear their altitude information stored in them immediately to indicate its absence in the main route hereafter according to Assumption (2).

Since q is not on the node set of the main route (i.e., $\{p_0, p_1, p_2, \ldots, p_k\}$) and $[p_r, q]$ and $[q, p_i]$ (with altitude $(p_r) <$ altitude (p_i)) are used to replace a partial route including link $[p_{i-1}, p_i]$, we have that the main route combined with newly added directed paths always connect from a small-altitude node to a high-altitude node under Assumption (2), even when multiple DLBAs are activated simultaneously. Note that the altitude (q) should be updated to altitude $(p_r) + 1$ immediately based on Assumption (2). When a link $[p_i, p_j]$ constructed originally in the main route or a directed path from p_i to p_j added by DLBA here can be viewed as a binary relation R such that $\{p_i \; R \; p_j |$ there is a route from p_i to p_j with altitude $(p_i) <$ altitude (p_j), where $i, j \in \{p_0, p_1, p_2, \ldots, p_k\}\}$. Theoretically, the whole constructed routing paths can be regarded as a partially ordering relation R (owing reflexive, anti-symmetric, and transitive relations) on node set $\{p_0, p_1, p_2, \ldots, p_k\}$. The fact prohibits any loop existed in the constructed routing paths by DLBA. ■

As shown above, our proposed protocol (DPS and DLBA) can be free of routing loops in cases where the altitude information is maintained up to date all the time. However, since altitude information maintenance relies on overhearing packets that are sent from mobile nodes, inconsistency in altitude information may occur in some cases. For example, change of route right after the activation of DPS or DLBA may be one of such cases. To be more specific, as can be seen in Fig. 9a, node X overhears packet originates from node E, which has an altitude of 4. Node X then set the minimal altitude it overheard to 4. Suppose links A-B, B-C, C-D, D-E are all on the verge of breaking, activations of DLBA may add nodes R, S, T, and U into the original path as shown in Fig. 9b. If node X happens to move closer to node U and overhears a packet (which has altitude value of 7) from node U to E, node X then activates DPS according to the criteria depicted in

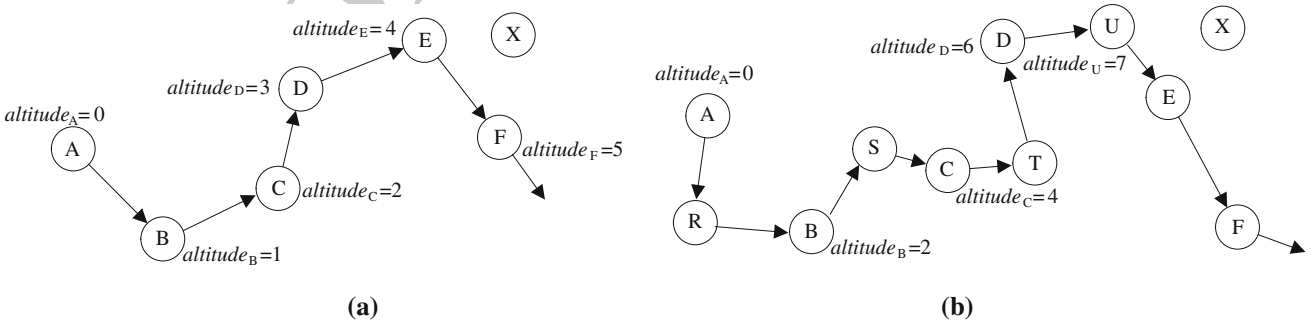


Fig. 9 Scenarios showing the occurrence of routing loop, with (**a**) and (**b**) being the communication path prior to and after the activation of DLBA

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

Algorithm 2 by setting node U as its next-hop node and asking node E to change its next hop to node X. As we can see, a routing loop occurs.

It has been an essential issue in avoiding looping conditions in routing protocol design. However, as routing loop rarely occurs in our simulation, only 3.78 occurrences per simulation run in average to be more specific, we thus concentrates on developing a loop detection algorithm instead. Once a looping condition is identified, node in our protocol would invalidate corresponding route in its routing table and send a RERR packet to the source node asking for a route reconstruction.

Our loop detection algorithm is quite simple and is based upon altitude information since nodes on a path would update their altitudes according to those contained in the packets. Once a node finds out its altitude value smaller than that contained in the packet, it indicates a possible looping condition. To further confirm potential looping condition, this node will insert the identification information about this packet into its own checking list CH_LIST. If routing loop indeed exists, the same packet would loop back and the node can be sure of a routing loop occurrence by comparing the packet's identification information with that stored in the CH_LIST. Detailed loop detection algorithm is presented in Algorithm 3.

Algorithm 3. Loop Detection Algorithm

- Whenever a node received a packet, it checks if the information tuple (source ID of the packet, destination ID of the packet, packet ID) of this packet have been stored in its CH_LIST. If the searching result is positive, a loop is detected. By comparing the time stamp field of each entry in the CH_LIST with current time, all the expired entries in the CH_LIST are removed.
- Whenever a node finds the altitude value contained in receiving packet is higher than itself, it will store (source ID of the packet, destination ID of the packet, packet ID, current time + valid time period) into its own checking list CH_LIST. (We set the valid time period of each record to 5-s in our simulations)

4 Performance analysis and evaluation

Due to the constant moving nature of nodes and thus the changing network topology, one may be skeptical if dynamically adjusting communication path could have positive effect to the routing performance other than increasing network loading. In addition, the fact that DLBA may increase path hop count suggests that it should be activated only when necessary. To avoid triggering DLBA mechanism right after DPS operation, the distance of the any adjacent nodes of a new shorten link should not be too long. It appears that there exist some rules or guidelines governing the proper timing of activating both DLBA and DPS so that their operations may result in good routing performance. In the following two subsections, we will first try to resolve skepticism regarding to the necessity of DLBA and DPS operations, and then explore the appropriate parameter settings so that DLBA and DPS can cooperate together to achieve (hopefully) the optimal performance (in terms of various routing metrics) through brief analysis of the protocol, follows by presenting the experimental simulation results.

4.1 Cost effectiveness in adapting routes to the changing network topology

Let us first focus on the DLBA mechanism. In case the distance of two adjacent nodes, say node A and B as shown in Fig. 10, on a path exceeds some pre-determined threshold, the DLBA mechanism would be activated to insert an intermediate node between them. Let node B be the downstream node and has found link A-B being on the verge of breaking, it then sends a notification packet to its upstream node A. For each node within the neighborhood of link A-B that hears such a packet, it would evaluate whether itself should involve and resolve such potential link breaking incident. If the evaluation turns out to be positive, the node then sends a packet to the upstream node A suggesting a possible detouring. Consequently, to analyze the overhead of activating DLBA, it is necessary to calculate how many nodes involve in such an activity. Let us define the distance threshold value mentioned above as follows.

$Threshold_{DLBA}$: The parameter defines the threshold distance (in meters) between two adjacent nodes of a link, above which the DLBA mechanism is triggered. For briefness we denote this threshold distance using T_D hereafter.

Note, with scenario depicted in Fig. 10, nodes that are within the downstream node B's power range may hear the notification packet, but only nodes that locate within the

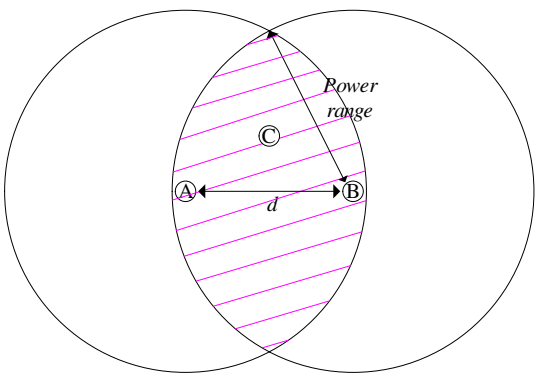


Fig. 10 Nodes within the *line-shaded area*, such as node C, are potential candidates that may involve in the DLBA process

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

power range of both upstream node A and downstream node B are qualified as the detour nodes. For example, nodes located within the line-shaded area in Fig. 10 are potential candidates that may be chosen by the DLBA mechanism to insert in between link A-B. Accordingly, if the area of the line-shaded area can be derived, we are able to estimate the number of potential detouring nodes provided that nodes are distributed uniformly in the network area. Once the number of potential detouring nodes is available, the number of control packets that may be triggered by the DLBA mechanism can then be calculated.

Theorem 3 *Upon activation of DLBA mechanism, all nodes that are involved in a DLBA activation locate within a specific area, and the area, S_{area}, of such specific area is less than or equal to $2 \times r^2 \cos^{-1}\frac{T_D}{2r} - T_D \times \sqrt{r^2 - \frac{T_D^2}{4}}$, where r is the maximum power range.*

Proof Let d denote the distance between the centers of two neighboring nodes. The line-shaded area in Fig. 10 can be calculated with the following equations.

$$\begin{aligned} S_{\text{area}}(d) &= 4 \times \int_{\frac{d}{2}}^{r} \sqrt{r^2 - x^2}dx \\ &= 4 \times \left(\frac{r^2}{2}\cos^{-1}\frac{d}{2r} - \frac{1}{2} \times \frac{d}{2} \times \sqrt{r^2 - \frac{d^2}{4}}\right) \end{aligned} \quad (1)$$

Since the protocol we proposed in this paper relies on estimating the distance between nodes through the strength of received signal. When the link distance d goes beyond a pre-determined threshold value T_D, the downstream node would be aware of such situation during packet receiving and trigger the DLBA mechanism. Thus, d should be greater than or equal to T_D. From (1), we get the following result.

$$S_{\text{area}} = 2 \times r^2 \cos^{-1}\frac{T_D}{2r} - T_D \times \sqrt{r^2 - \frac{T_D^2}{4}} \quad (2)$$

■

Assume that there are n nodes distributed uniformly on an area $Area$. By multiplying S_{area} as derived in Theorem 3 with node density, which is $n/Area$, we can derive Theorem 4.

Theorem 4 *In case there are n nodes distributed uniformly on an area with size Area. The expected number of nodes qualified as the detouring node is less than or equal $S_{area} \times n/Area$. And the expected number of control packets generated as a result of a DLBA activation is less than or equal $((S_{area} \times n/Area) + 1) \times \alpha$, where α is the expected number of packets that goes through the breaking link and triggers the DLBA mechanism.*

Proof When a downstream node finds out its distance to the upstream node exceeds the preset threshold value T_D, it sends out (one-hop broadcast) a *help* packet. Any node that is qualified as the detouring node would send a CREQ request to the upstream node. Accordingly, the expected number of control packets is equal to (the expected number of the potential detouring node plus one (the *help* packet)) $\times$ α. By Theorem 3 with given node density, we can obtain that the expected number of nodes qualified as the detouring node is less than or equal to $S_{\text{area}} \times n/Area$. ■

DLBA activation is a consequence of some link being breaking. However, since nodes are moving randomly, the breaking links may not be broken eventually. Accordingly, some DLBA activations may turn out to be unnecessary, which results in superfluous control packets. On the other hand, if the link breaking prediction is correct, a communication route may be saved from breaking and a route reconstruction process is thus avoided. Let assume p be the probability of DLBA activation that correctly and successfully saves a particular link from breaking. In a connected network with n nodes, an AODV route reconstruction procedure may result in n flooding RREQ packets. The number of reply packets to RREQ, RREP, would be related to the path length which is usually far smaller than n and may be neglected for simplicity. Accordingly, the expected number of control packets saved as a result of DLBA activation would be $p \times n$. Therefore, as long as the control packet reduction, $p \times n$, is greater than the overhead induced by DLBA, which is $((S_{\text{area}} \times n/Area) + 1) \times \alpha$ as illustrated in Theorem 4, the inclusion of DLBA would be justified in term of control packet overhead. The above descriptions lead to Theorem 5 as follows.

Theorem 5 *In case there are n nodes distributed uniformly on an area Area and the probability of a given DLBA activation that correctly and successfully saves a link from breaking is p. As long as $p > (S_{area}/Area + 1/n) \times \alpha$, the DLBA mechanism can be expected to reduce the number of control packets.*

To get a rough idea of what Theorem 5 may indicate, we provide a practical example as follows. With 100 nodes distributed uniformly in a 2,200 m $\times$ 600 m rectangular area, the maximum power range set to 250 m, the value of T_D set to 240, and α (the expected number of packets that goes through the breaking link and triggers the DLBA mechanism) set to one, the right hand side of the equation in Theorem 5 amount to 7.15%. In other words, as long as there are more than 8 successful activations of DLBA out of every 100 activations, DLBA may reduce the number of control packets as compared to scenario that is without DLBA. Note that the network environment given above is typical in many ad hoc network simulations, which is also what we adopt in ours. By successful DLBA activation, we

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

mean without that particular DLBA activation, a breaking link would be broken as predicted. In addition, in our subsequent simulations, we also found that DLBA indeed reduces the number of overall control packets in almost all cases.

Since DLBA mechanism operates by replacing a breaking link with two new ones, it is essential that we set some criteria to the new links so that they will not trigger another immediate DLBA activation. Accordingly, we define the threshold value to ensure the quality of the newly selected links as follows.

Threshold$_{quality}$: The parameter defines the threshold distance, at which any two adjacent nodes of a newly established path by DLBA mechanism must not exceed. The primary purpose of setting such restriction is to ensure the newly constructed link will not trigger another immediate DLBA activation. In other words, this parameter is used to assure path quality when new path is introduced through DLBA mechanism. For briefness we denote this threshold distance by *Tq* hereafter.

The relationship of T_D, T_q and maximum power range is depicted in Fig. 11.

To reflect the introduction of T_D and T_q, Fig. 10 is modified and the new scenario is shown in Fig. 12. Although the line-shaded area is shrunk, it does not affect the correctness of Theorem 3 $\sim$ 5 since only upper and lower bound are of interest in all the three cases. In addition, if the exact values of T_D and T_q can be acquired, it is possible to derive more accurate bounds using similar procedure. However, we will not go any further in this aspect.

Increasing *Tq* value may result in a greater line-shaded area, which means the DLBA mechanism would more likely be successful. In other words, DLBA may have a higher probability in finding a detour node. However, it may also result in links that are more vulnerable to

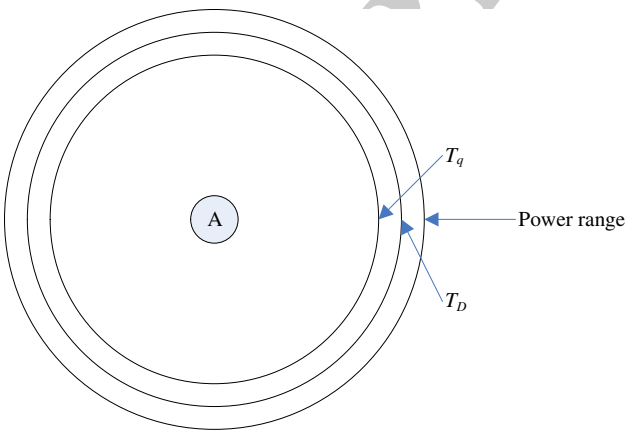


Fig. 11 Relationship between two threshold distances: T_D and T_q

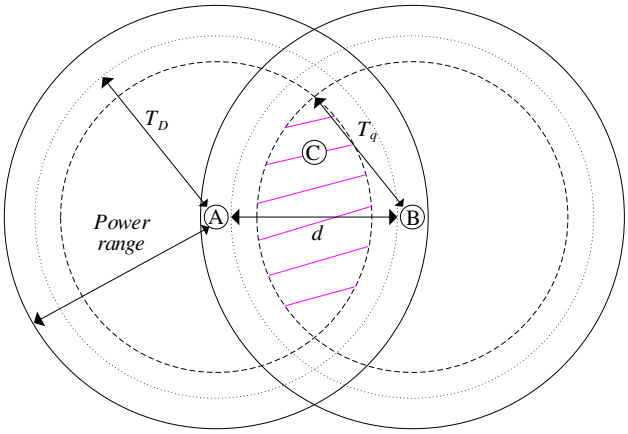


Fig. 12 Node *C* can be used as a detouring node and a new partial route *A-C-B* is formed to replace the original bad quality link *A-B*

breaking, which again triggers another DLBA mechanism activation. Frequent DLBA activation induces more network overhead and increases chance of packet collision, thus causing more packet loss. On the other hand, a lower *Tq* value indicates a stricter requirement to the newly generated alternative DLBA links. However, it also means that there is less chance that DLBA will take action. The determination of an optimal *Tq* value thus requires further investigation. To simplify the following analysis, we assume the probability of a link failure to be P_b and is proportional to the length, *len*, of the link. We then have:

$$P_b(link_{ab}) = 1/r \times len(link_{ab}) \tag{3}$$

where r is the maximum power range.

Accordingly, we can derive the following theorem.

Theorem 6 *Assume that DLBA is activated when the distance of a link is exactly T_D, and the probability of a link failure is proportional to the length of the link, i.e. $P_b(link_{ab}) = 1/r \times len(link_{ab})$. In case the probability of link breaking of the new partial route constructed by DLBA is less than the original link, then the length of each link in the newly constructed partial route, T_q, should meet the following equation: $T_q < r \times (1 - \sqrt{1 - \frac{T_D}{r}})$.*

Proof Suppose the DLBA mechanism replaces a breaking link $link_{ab}$ with $link_{ac}$ and $link_{cb}$. We would expect the probability of at least one of the replacing links fails to be lower than the replaced link, as follows.

$$P_b(link_{ab}) > P_b(link_{ac}) + P_b(link_{cb}) - P_b(link_{ac}) \times P_b(link_{cb}) \tag{4}$$

Combining (3) and (4), we derive:

$$1/r \times len(link_{ab}) > 1/r \times len(link_{ac}) + 1/r \times len(link_{cb}) - (1/r)^2 \times len(link_{ac}) \times len(link_{cb})$$

Journal : Large 11276 | Dispatch : 11-8-2009 | Pages : 23
Article No. : 203 | LE | TYPESET
MS Code : WINET-683 | CP | DISK

The above equation can be further simplified as follows:

$$len(link_{ab}) > len(link_{ac}) + len(link_{cb}) - 1/r \times len(link_{ac}) \times len(link_{cb}) \quad (5)$$

According to Fig. 12, the worse case in finding a detouring node, say C, would be that distance between node C and node A, B are both T_q. In addition, for $len(link_{ab}) = T_D$, (5) can then be represented as: (Note that it is not likely that link distance is eactly T_D every time the DLBA activated. The exact distance depends on node mobility and packet transmission speed, however, for the sake of simplicity, the details are not taken into consideration here.)

$$T_D > T_q + T_q - 1/r \times T_q^2$$

The solutions of the above equation are:

$$T_q > r \times \left(1 + \sqrt{1 - \frac{T_D}{r}}\right) \quad \text{or}$$

$$T_q < r \times \left(1 - \sqrt{1 - \frac{T_D}{r}}\right)$$

Since $T_q \leqq$ maximum power range $= r$, which make (6) below the only solution.

$$T_q < r \times \left(1 - \sqrt{1 - \frac{T_D}{r}}\right) \quad (6)$$

■

According to Fig. 12, and T_q set to $r \times \left(1 - \sqrt{1 - \frac{T_D}{r}}\right)$, we can find that the larger the T_D value the larger the line-shaded area and thus the higher probability a node can be found to avoid a potential link breaking occurrence.

The purpose of DPS is to improve communication efficiency by eliminating potential redundant node(s) from paths, which means that it tends to increase node distance of links. However, longer link distance may likely trigger the activation of DLBA very shortly and thus offset the effect of DPS. In order to avoid such interference, our protocol requires the distance of new link constructed by DPS also not exceeding T_q.

Regarding to the number of control packets resulted from DPS, the two types of path shortening should be considered separately. For simpler type I scenario (Fig. 4a), there is only one control packet generated. Accordingly, as long as there are more than one data packet going through the newly shortened path, the control packet overhead associated with DPS can be justified. On the other hand, the case with Type II path shortening (Fig. 4b) is much more complicated because it must take into consideration various scenarios involving shortening a multi-hop to a single hop and the race conditions mentioned in Sect. 3.3.2. However, if assisted by DLBA so that shortened paths can be sustained for a longer time and then allows more data packets flowing through such a shortened path, the control packet overhead by type II DPS can still be justified.

4.2 Performance evaluation

As mentioned in the previous subsection, the value of T_D should be as large as possible. However, the T_D value should also be less than the maximum power range so that DLBA has enough time to find an alternative route before the link is broken. Since the power range in most simulations is set to 250 m and with T_D being set more conservatively to 240, we derive through (6) that upper bound of T_q equals 200.

To roughly derive appropriate values for T_D and T_q, we conducted a simulation with ns2. We set the simulation environment to a 2,200 × 600 m rectangular area with 100 nodes inside, limited the number of source-destination communications to a single pair, and varied the T_D value between 210 and 250 and the T_q value between 190 and 250. The results are shown in Fig. 13. It appears that the best performance occurs when $T_D = 240$ and $T_q = 210$ or 220. As a result, we will set the T_D and T_q values to 240 and 210, respectively. Note that under the same T_D and T_q setting from the previous simulation, link breaking occurrences can be reduced by 94% with simultaneous application of both DLBA and DPS.

5 Simulation results

We used ns2 [30] simulator (version 2.1b8a) running on a personal computer with Microsoft Windows XP Professional operating system to evaluate the performance of the proposed protocols. The radio coverage region for each mobile node is assumed to be a circular area 250 m in diameter (maximum power range). We assume the source node sends three data packets every second (Constant Bit Rate), and uses UDP as the transport protocol. The data packet size was set to 64 bytes. Each node has a queue that is capable of holding up to 50 packets awaiting transmission.

In most cases, the simulation environment is a 2200 by 600 m rectangular region with 100 mobile nodes moving around. Nodes are placed randomly inside the region. Once the simulation begins, each node moves toward a randomly selected direction with a random speed ranges from 0 to 20 m per second. Upon reaching some randomly determined location, the node pauses for a fixed time, *pause time*, and proceeds again in a similar manner. Transmissions between pairs of source and destination nodes, called *conversions*, are selected arbitrarily from nodes within the

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

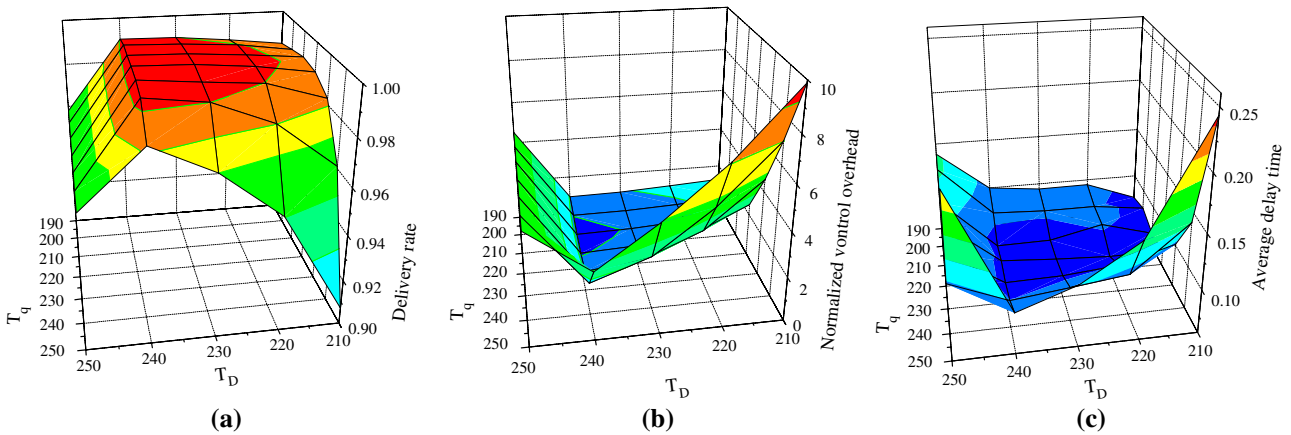


Fig. 13 Effects of varying T_D and T_q to the three performance metrics, (**a**) data delivery rate, (**b**) control overhead, and (3) delay time

Table 1 Parameters setting in our simulation

Parameter	Value
Power range (transmission rage)	250 m
Channel capacity	2 M bits/s
Simulation time	500 s
Number of nodes	100
Number of comm. pairs	10
Topology size	2,200 × 600 m^2
Traffic type	CBR: constant bit rate
Packet rate	3 packets/s
Packet size	64 bytes
Mobile speed	0–20 m/s
Path loss model	Two-ray ground
MAC protocol	802.11 DCF
Mobility model	Random waypoint
Interface queue type	DropTail/PriQueue

simulation area. The simulation time limit is set to 500 s with each simulation scenario representing an average of 30 randomly generated test sample runs. Table 1 lists all the essential parameters and their values used in our simulation (Unless specified otherwise, all parameters use values listed in Table 1).

AODV, with or without the dynamic link breaking avoidance and/or dynamic path shortening features, result in four possible routing protocols: AODV, AODV with link breaking avoidance (denoted by *DLBA* hereafter), AODV with dynamic path shortening (denoted by *DPS* hereafter), and AODV with both link breaking avoidance and dynamic path shortening (denoted by *DLBA* + *DPS* hereafter). In addition, DSR is also included in our simulations as a comparison. Four performance metrics, as listed below, are adopted to evaluate how the five protocols perform.

- Data delivery rate—The ratio of the number of successfully delivered data packets to the total number of data packets sent.
- Delay time—The time a successfully delivered data packet takes from the source to the destination.
- Normalized control overhead—The total number of control packets sent out by all nodes divided by the total number of successfully delivered data packets.
- Average hop counts—The average number of hops a data packet needs to traverse to reach its destination.

Four simulation setups, which include changing (1) node mobility, (2) traffic loading, (3) node density, and (4) route hop count (reflected by changing simulation geometry), are conducted to explore how the five protocols perform under various environmental parameters as described above.

In the following, we will present and interpret the simulation results.

5.1 Effects of node mobility

The objective of this simulation setting is to explore how the node mobility affects the performance of these protocols. Node mobility is reflected with varying pause time, while simulation region is fixed at 2200 by 600 m with 100 mobile nodes and 10 concurrent conversations. A larger pause time implies that nodes stay at some specific locations longer, which again indicates a more stable network topology. The results are shown in Fig. 14.

Both AODV and DSR are on-demand based routing protocols, which establish routes upon request. However, once the routes are constructed, no effort is done to adjust the routes according to the changing topology. As a result, both protocols are more sensitive to node mobility and are less adaptive to network topology changes. Figure 14a and b reflects quite well the characteristic of the two protocols. As we can see, both AODV and DSR do not perform well

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

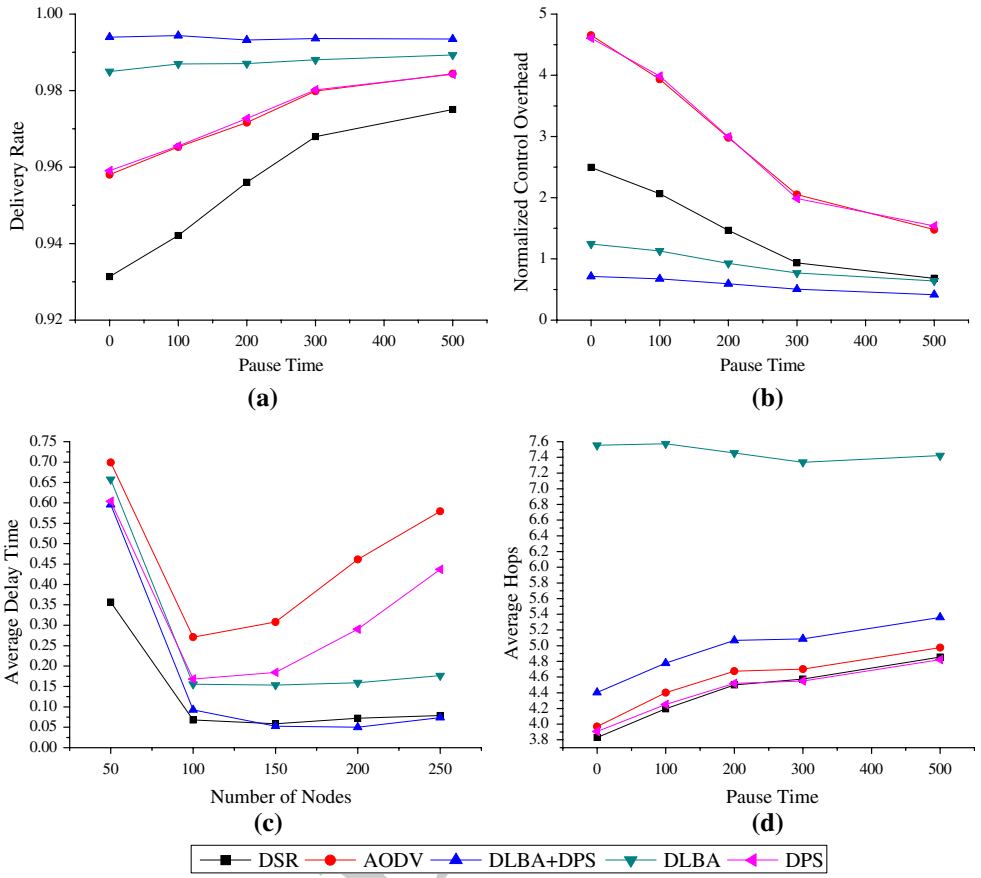


Fig. 14 Simulation results on effects of node mobility to routing performance

in terms of data delivery rate and control overhead in the cases with smaller pause time (equivalent to higher node mobility). When comparing the two protocols, AODV apparently has higher control overhead and longer delay time, which is a direct consequence of its attempt to find a new route upon link breaking. On the other hand, the comparatively lower control overhead and delay time of DSR may be attributed to the fact that DSR establishes multiple paths from a given source to its destination, which are then cached in nodes at the route construction stage. Consequently, DSR would first try to pick an alternative route from those available in the cache prior to initiating a new route request. However, more data packets would be lost if all previous established routes are out of date and not available, which explains why the successful transmission rate of DSR is lower than AODV. For the same reason, the performance gap in delivery rate appears to widen in more dynamic network environment (with smaller pause time).

There seems to be virtually no improvement when embedding the DPS mechanism in AODV, which is reasonable because AODV usually adopts the shorter path. DPS thus has little chance for action. On the other hand, embedding the DLBA mechanism to AODV does have significant impacts on improving delivery rate, lowering control overhead and reducing delay time. It is evident that DLBA indeed prevents quite a lot of potential links from breaking. However, since DLBA operates by inserting intermediate nodes into an existing path, the average hop counts in the route also increase. A longer path (which contains more intermediate nodes) implies that the route is more vulnerable as the probability of any given node failure or any given node moving out of range also increases. In this case, the effect of DPS begins to surface, which, according to Fig. 14d, consistently reduces the average hop counts by 2.06–3.15 when comparing to the case that DLBA is adopted alone. The other three performance metrics are also improved considerably. In the most dynamic condition (pause time = 0), the combination of DLBA and DPS increases the delivery rate by 3.75%, reduces the control overhead and delay time for up to 84.67 and 66.91% as compared to the original AODV. Even in an extremely static case (pause time = 500 s), DLBA + DPS still edges AODV in the three performance metrics by 0.92, 71.81, and 31.15%, respectively. In summary, as long as the network topology does change, the DLBA/DPS combination is capable of improving AODV performance. Note that, even though we set pause time to 500 s, which is equal to our simulation time period, the network topology does change since nodes still move prior to pausing. A highly dynamic network topology certainly poses a higher challenge to the

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

routing protocols; accordingly, all subsequent simulations assume and set the pause time to zero.

5.2 Effects of traffic loading

The objective of this simulation setting is to explore how these protocols react to various traffic loading conditions. The simulation region was fixed at 2200 by 600 m with 100 mobile nodes and zero pause time. The number of conversations was varied between 10 and 50. Note that 50 pairs of concurrent conversations implies that virtually every node within the region is either transmitting or receiving and represents an extremely heavy traffic scenario. The results are presented in Fig. 15.

Similar to the results given in other researches, DSR performs better than AODV in terms of delay time and control overhead. However, as we have explained in previous sections, DSR performs less well in delivery rate under a highly dynamic network environment (pause time = 0).

Note that DLBA also achieves higher data delivery rate than AODV until the number of conversations exceed 30. The reason for such a turnover is most likely a consequence of packet collisions. To be more specific, the DLBA mechanism results in longer communication paths, which indicates that a node may very likely participate in more than one communication stream. In the case of an extremely busy network condition, concurrent data packets from different streams may have higher probability in contending for resource at the same node. Data delivery rate thus drop as a result of such data packet contention. Furthermore, as fewer data packets make it through longer paths, data packets that are successfully delivered are usually via some shorter routes. The fact is reflected in Fig. 15d, in which average hop count decreases with heavier traffic loading.

On the other hand, the combination of DLBA and DPS has proven again to be the best performer among others under all traffic load conditions in our simulation. We can see from Fig. 15d that when DPS and DLBA are working together, a mere one-hop reduction in communication path may impact the overall performance. In addition, the paths established with DLBA + DPS appear to be more stable. The data delivery rate curve, Fig. 15a, remains flat for a conversation size as great as 40, and it also drops less sharply beyond that point than the other protocols. The same pattern can also be observed in the other three metrics as shown in Fig. 15b, c and d. In particular, in the busiest traffic condition, DLBA + DPS maintain a respectable delivery rate of 93.33%, which is 9.68% higher than that

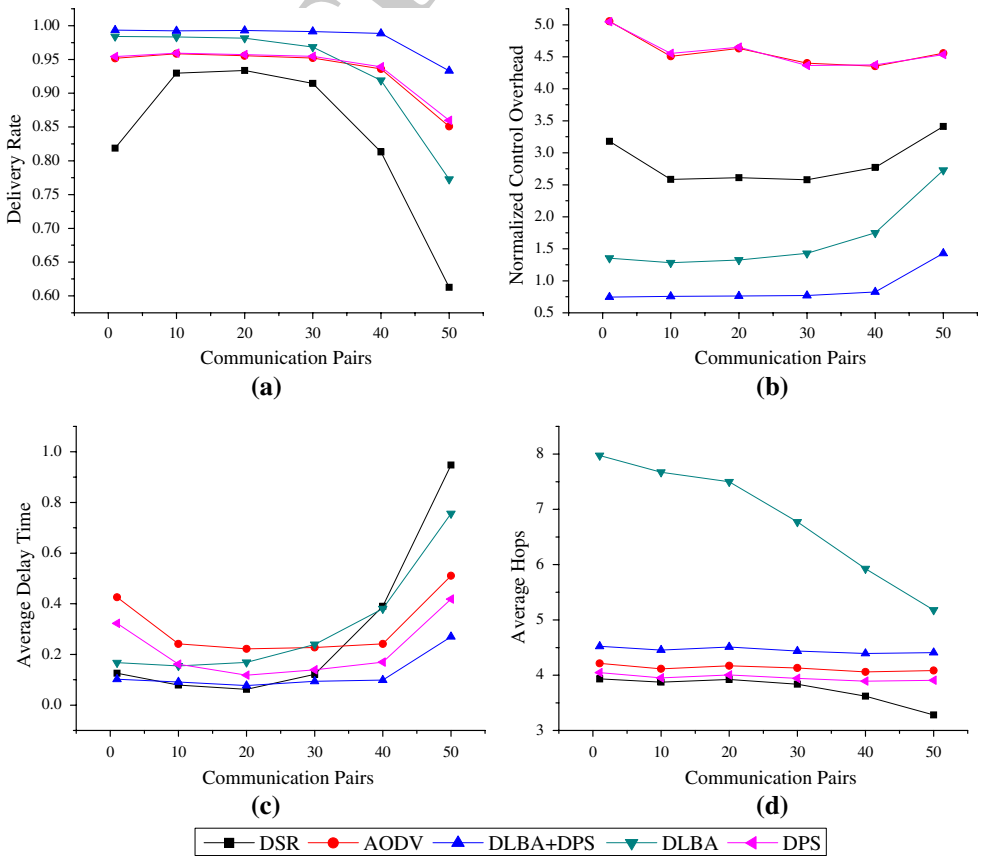


Fig. 15 Simulation results on effects of traffic loading to routing performance

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

achieved by AODV. Better yet, DLBA + DPS does that with 68.66% less in control overhead and 47.16% less in transmission delay (as compared to AODV).

5.3 Effects of node density

The objective of this simulation setting is to evaluate how the protocols perform under various node density conditions. Simulation region is again fixed at 2200 by 600 m and the number of conversations is fixed at 10, while the number of nodes within the simulation region is set to 50, 100,150, 200 and 250, respectively. The results are presented in Fig. 16.

According to Fig. 16a, we can see that data delivery rate of each protocol is comparatively lower in the cases of lower node density (e.g., number of nodes = 50). Apparently, in lower node density condition, there is less chance in successfully finding an available route. It also takes longer to establish a potential route when required, which results in longer delay time as can be seen in Fig. 16c. On the other hand, as node density increases, the cost (control overhead) of RREQ broadcast packets as a result of link breaking also increases (Fig. 16b), which adds more packets to the queue of each node and increases the average waiting time of packets in each node (Fig. 16c). In addition, higher node density seems to have negative effect to data delivery rate, which is particularly true for DSR. However, for AODV-based protocols, the effect is not as obvious. The possible explanation for this may be that routes discovered by DSR usually share many common nodes, or may pass through nodes that are quite close to each other, which again results in routes, either in use or in cache, that have very high similarity. The effect would be more significant under a higher node density condition that makes these routes more likely to be broken at the same time and thus reduces the data delivery rate.

The evidence that DLBA can effectively prevent links from breaking (and thus link re-establishment occurrences) can be seen in Fig. 16b. The much flatter curves for DLBA and DLBA + DPS, as compared to AODV and DPS, show that node density has less impact on the DLBA-embedded protocols, which is apparently a direct consequence of much fewer RREQ broadcasts.

From this simulation, in proper node density conditions DLBA + DPS can achieve a significant 4.58% improvement in data delivery rate when compared to AODV. In addition, the greater the node density, the greater the control overhead and delay time savings with DLBA and

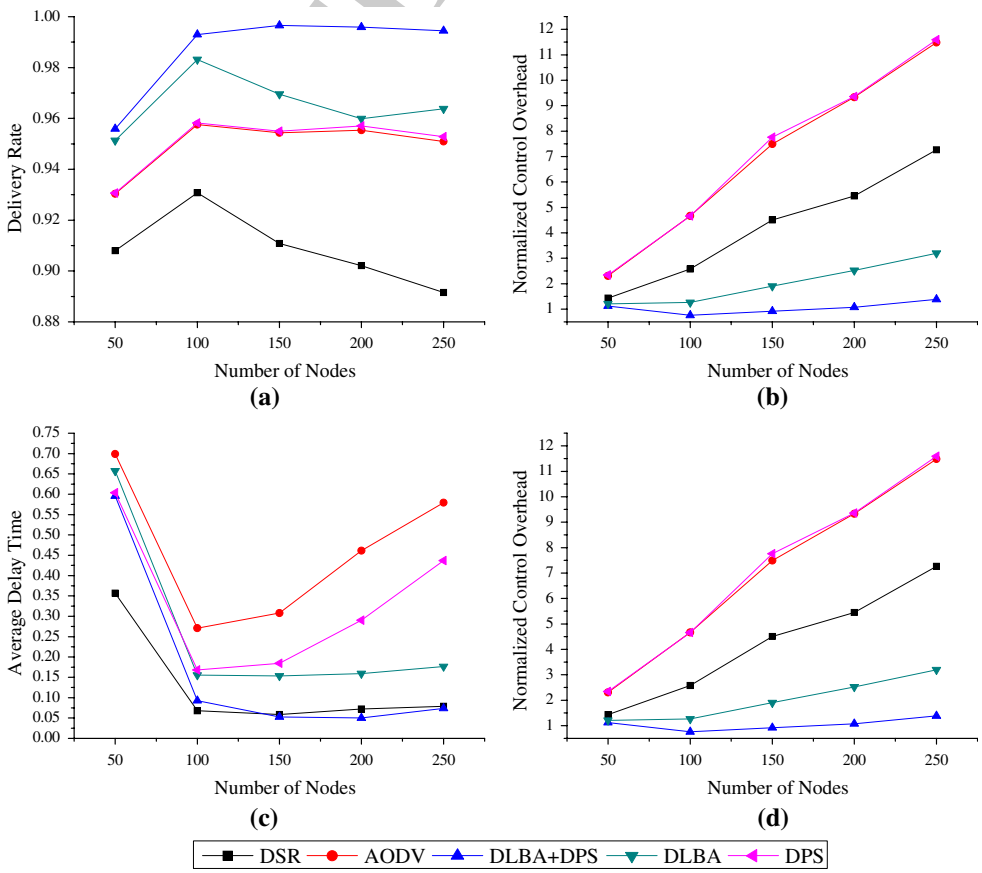


Fig. 16 Simulation results on effects of node density to routing performance

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

DPS adoption. In the case of the highest node density (number of nodes = 250), DLBA + DPS can save up to 87.95% in control overhead and 87.29% in delay time.

5.4 Effects of route hop counts

The objective of this simulation setting is to explore how the geometric area (and thus hop count of routes) affects the performance of these protocols. The width of the simulation region is fixed at 600 m, while changing the length to 1,100, 2,200, 3,300, 4,400, and 5,500 m. Node density is kept constant by placing appropriate number of nodes (50, 100, 150, 200, and 250, respectively) within the simulation areas. The number of conversations is fixed at 10 and pause time is set to zero. The results are presented in Fig. 17.

First of all, note that according to Fig. 17d the adjustment of geometric area indeed affects the average route hop count overall. Longer path length increases the probability of link breaking and thus route reconstruction occurrences, which then result in a lower data delivery rate and higher control overhead.

Again, DLBA achieve quite well in data delivery rate and control overhead due to its ability in reducing link breaking occurrences. However, the comparatively longer path length makes DLBA perform only slightly better than AODV. On the other hand, the excessively longer path shortens the distance between two adjacent intermediate nodes, which provides room for effective DPS operation and thus reduces the DLBA side effect. According to Fig. 17d, the reduction in average hop count is 2.99 with DLBA and DPS working together. The improvement in delay time is also quite obvious, as shown in Fig. 17c. For longer distance communication environments, DLBA + DPS may also outperform DSR (and the other protocols) in terms of data delivery rate, control overhead, and delay time. In the most extreme case in this simulation (length of simulation = 5,500), DLBA + DPS can reduce up to 82.47% in control overhead and 66.79% in delay time, while achieving 12.52% more in data delivery rate than AODV.

5.5 Discussion

We have conducted extensive simulations to explore how our proposed mechanisms, DLBA and DPS, perform when embedded in the original AODV protocol under various node mobility, network traffic loads, node density, and network geometric environments. In almost all cases, integrating the DLBA mechanism to AODV can achieve much better performance than original AODV in terms of data delivery rate and control overhead, which may be

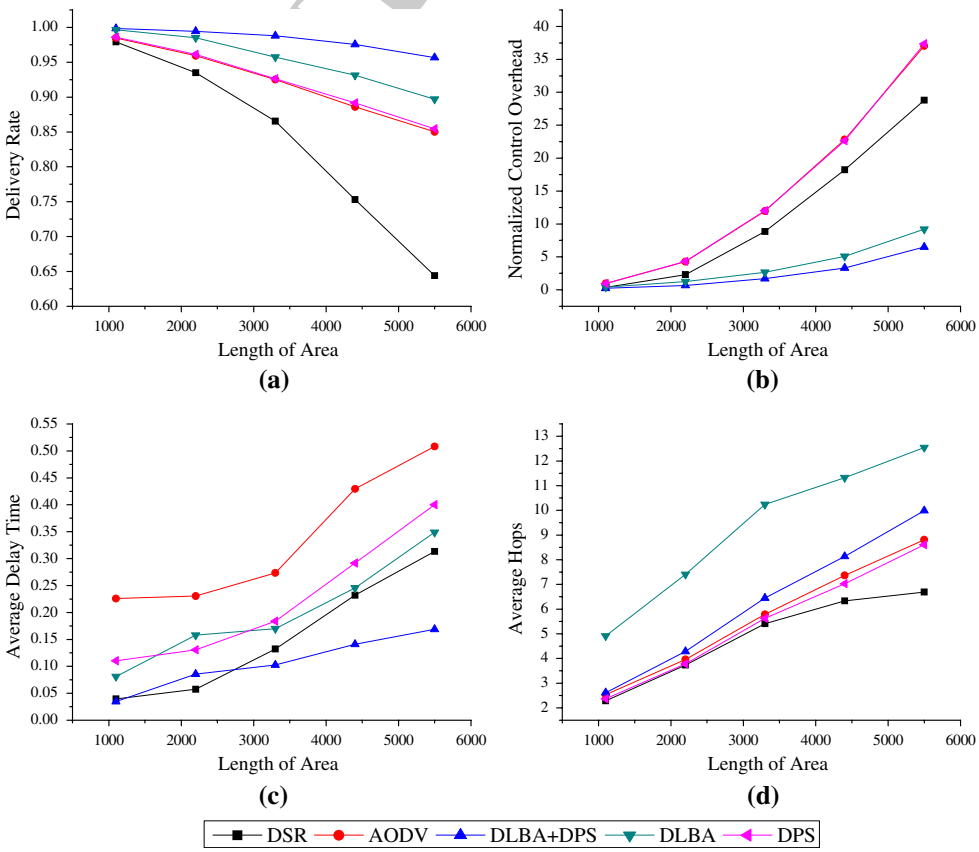


Fig. 17 Simulation results on effects of path length to routing performance

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

primarily attributed to the capability of DLBA in effectively reducing link breaking occurrences. However, since DLBA works by inserting intermediate nodes to link that is on the verge of breaking, it thus results in paths that are usually longer than those constructed by AODV. Consequently, in terms of communication delay, DLBA does not perform as well as it does in the other two routing performance metrics. In cases that network traffic is extremely heavy (number of conversations = 50), data delivery rate with DLBA drops somewhat more sharply than the other protocols and even performs worse than AODV (see Fig. 15a).

Contrary to the cases with DLBA, embedding DPS into AODV shows virtually no improvement. It appears that routes established with AODV under static network topology environment are usually short and robust enough, which leaves little room for DPS to operate. Even under a rather dynamic network condition (with high node mobility), DPS does not seem to contribute much, either. The potential cause can be the following. First, for similar reason stated above, there is not much DPS can do when routes first constructed with AODV. Second, as nodes start fast moving around, a path in operation may have been broken before DPS has any chance to do its job.

On the other hand, the combination of both DPS and DLBA achieves the best performance in all scenarios (varying node mobility, traffic loading, node density, and network geometry) of our simulations in terms of data delivery rate and control overhead. It also performs comparatively well or even better than DSR in term of communication delay. Apparently, DLBA application results in longer yet more robust communication links, which provide perfect starting points for the timely and efficient DPS mechanism. In other words, the two mechanisms cooperate together and preemptively adapt the working communication routes to the changing network topology. The outcome of application of such an adaptable routing protocol would be routes that are both stable and efficient.

Figure 18, which shows normalized number of link breaking occurrences and RREQ packets, gives qualitative evidences to the above speculations. Figure 18a and b show that embedding DLBA mechanism to AODV has reduced link breaking occurrences by 49.89–88.54% and cut the RREQ packets by 48.20–84.57%. With both DLBA and DPS cooperating together, those figures add up to 80.62–96.56 and 77.34–92.80%, respectively. On the other hand, in cases that DPS is working alone with AODV, improves AODV's performance slightly, reducing link

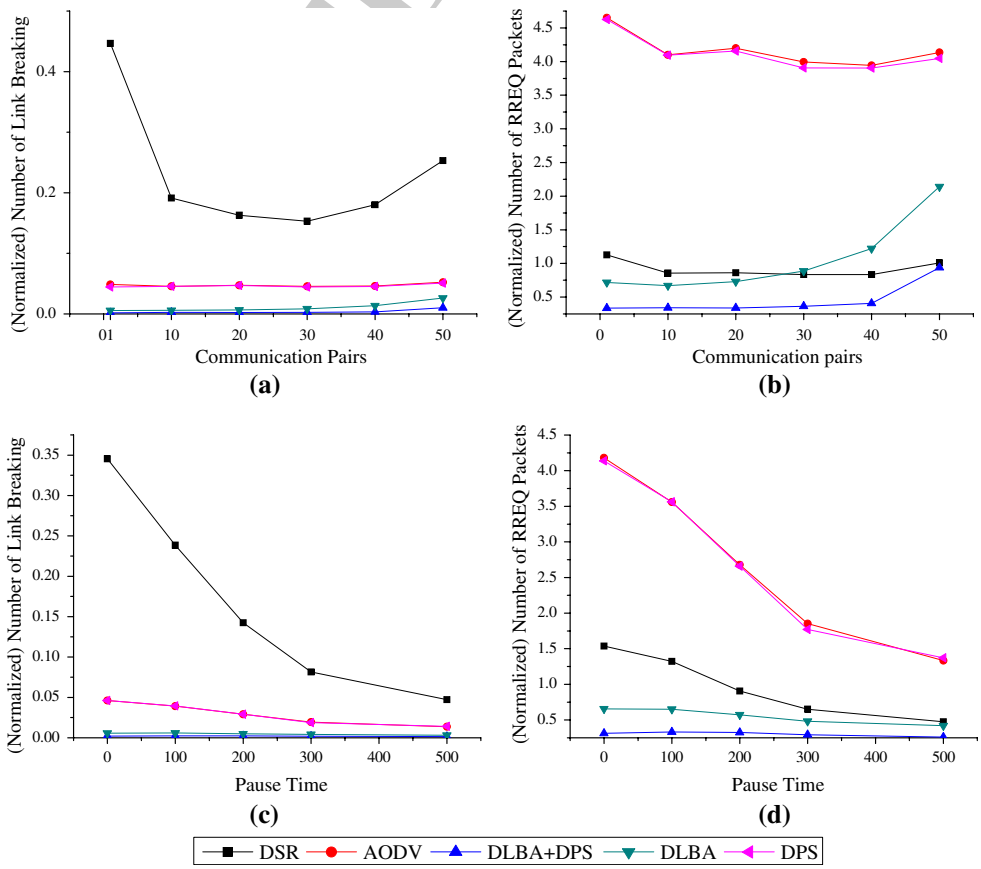


Fig. 18 Simulation results showing the normalized number of link breaking occurrence and RREQ packets of various routing protocols under varying traffic loading and node mobility conditions

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

breaking occurrences and RREQ packets by 0.22–2.82 and 0.14–0.2.23%.

Also shown in Fig. 18a–d, the much higher link breaking occurrences and lower numbers of RREQ DSR packets, as compared to AODV, provide evidence for our earlier arguments regarding DSR's tendency to use routes that are stored in cache. In addition, Fig. 18c and d also reveal that the discrepancy is wider as the network topology changes more rapidly.

5.6 Performance comparison to other major protocols

In this section, we compare our algorithm to other major algorithms that base on AODV or DSR. The protocols selected include those that are preemptive in nature like ours as well as those that use local repair strategy. All simulations use random way point mobility model with maximum speed set at 20 m per second. Since it is difficult to collect the codes that were developed in these protocols, we are unable to reproduce their results using exactly the same settings as ours. Accordingly, it may be easier for us to adapt our code to simulation environments listed in these papers. In cases that different simulator (GloMoSim) is adopted by the targets of comparison [20, 24], we also ported our algorithms to that specific simulator so that discrepancy due to differences in MAC and Physical layers, granularity of model, or mobility model can be minimized [31]. The detailed outcomes are listed in Table 2. Note that although we did our best to capture as accurately as possible their settings, there still exist a number of

Table 2 Performance comparison to other major protocols

	Variable		Delivery rate (%)		Delay (seconds)		Overhead	
			Ours	Target	Ours	Target	Ours	Target
[24][a]	Pause time	0	74.21 (69.56)	67.92	0.089 (1.365)	0.66	0.90[b] (2.63[b])	2.41[b]
	(1,500 × 300, 50 nodes, 40-pairs, 512B, 3 pkt/s)	300	72.76 (72.02)	71.30	0.101 (1.497)	1.07	0.89[b] (2.40[b])	1.44[b]
		600	73.32 (74.82)	71.93	0.111 (1.545)	0.98	1.05[b] (2.18[b])	1.32[b]
		900	72.20 (71.81)	70.86	0.116 (1.913)	1.70	1.13[b] (2.36[b])	1.42[b]
[20][a]	Comm. pairs	10	99.36 (99.68)	99.56	0.015 (0.051)	0.117	0.35[b] (0.205[b])	1.70[b]
	(1,500 × 300, 50 nodes, 512B, 4 pkt/s, 0-pause time)	20	98.02 (98.35)	97.68	0.032 (0.158)	—	0.49[b] (0.540[b])	—
	Traffic rate (pkt/s)	8	97.46 (97.36)	96.89	0.028 (0.119)	—	0.29[b] (0.382[b])	—
	(same as above)							
[18]	Comm. pairs	10	99.23	96.32	0.091	0.105	1,114	50,148
	(2,200 × 600, 100 nodes, 64B, 3 pkt/s, 0-pause time)	20	99.30	96.92	0.077	0.103	11,276	109,380
		30	99.14	94.97	0.094	0.210	22,727	171,948
		40	98.85	80.71	0.099	0.603	34,446	249,251
		50	93.33	45.24	0.270	1.508	100,186	298,119
[21]	Transmission rate	0.4 k	75.06	61.62	0.431	—	47.55[b]	69.07[b]
	(2,200 × 1,500, 150 nodes, 40-pairs, 128B, 60-pause time)	2.4 k	95.71	80.00	0.263	—	6.16[b]	14.26[b]
		4.4 k	96.10	71.66	0.286	—	3.84[b]	13.95[b]
[17]	Pause time	0	99.81	96.31	0.032	0.042	0.19[b]	1.81[b]
	(1,500 × 300, 50 nodes, 512B, 4 pkt/s, 5-pairs)							
[10]	Pause time	45	99.60	96.7	0.035	—	0.215[b]	3.9[b]
	(2,000 × 600, 100 nodes, 256B, 4 pkt/s, 16 pairs, 0–10 m/s)							
[23]	Comm. pairs	10	99.23	90.87	0.091	0.1	11,276	22,000
	(2,200 × 600, 100 nodes, 64B, 3 pkt/s, 0-pause time)	20	99.30	92.30	0.077	0.1	22,727	40,000
		30	99.14	92.67	0.094	0.08	34,446	110,000
		40	98.85	85.38	0.099	0.25	49,063	130,000
		50	93.33	69.36	0.270	0.61	100,186	158,000
[7]	Comm. pairs	10	99.85	96.26	0.030	—	6,386	63,598
	(1,500 × 300, 50 nodes, 64B, 4 pkt/s, 0-pause time)	20	99.82	96.13	0.026	—	12,656	124,097
		30	99.74	96.18	0.028	—	20,027	158,194

[a] Use GloMoSim, figures in parentheses are derived with ns2, [b]normalized control overhead

Journal : Large 11276 Dispatch : 11-8-2009 Pages : 23
Article No. : 203 LE TYPESET
MS Code : WINET-683 CP DISK

factors that may affect the simulation outcomes. For example, various versions of simulator may be using slightly modified MAC or physical layer parameters [30]. In addition, some of the papers do not list their detailed simulation parameters.

As we can see from Table 2, our algorithm performs consistently better than all other protocols in term of data delivery rate except in one scenario in [20] with 10 communication-pairs, which has only negligible 0.2% edge. But in an extremely heavy traffic environment, e.g., the number of communication pairs equals 50 in [18], the improvement with our protocol can be more than 100%. In terms of the other two metrics (delay and control overhead), our method is also better than all other protocols except in one case of [23], which performs particularly well in delay time when communication pairs equals 30.

When comparing to protocols using GloMoSim simulator, we also provide ns2 simulation results of our algorithms (figures in parentheses of Table 2). In other words, we have tested our algorithms in two different simulators with exactly same parameters setting. Based on the results, a couple of observations can be summarized as follows. First, it appears that both simulators somewhat agree with each other in term of data delivery rate. Second, when it comes to delay time or communication overhead, ns2 tends to give results with higher (worse) values. Note, although the above observations may be valid with our algorithms, they may not be generalized to all cases. However, the outcomes do indicate that direct comparison of simulation results derived from different simulators should be more careful or perhaps avoided.

6 Conclusions

The original AODV protocol works in a semi-dynamic fashion, which may establish a route on demand but continue using that route until it breaks. However, to suit the changing network topology of ad hoc networks a more aggressive and adaptable routing strategy is required. This paper proposes integrating the DLBA and DPS mechanisms into a modified AODV protocol by developing a pair of parameters to determine the timing for activating DLBA or DPS so that the two algorithms can work cooperatively and complementarily together. The new protocol can significantly improve the performance of the original AODV routing protocol and can also adapt itself well in a very dynamic network environments according to simulation results. Overall, the new protocol is capable of maintaining higher link quality, achieving higher data delivery rate with less average delay time, while incurring much less network overhead.

However, our proposed DPS scheme considers only the case that a node is used to replace two or more nodes in an established path. How can the protocol be modified so that it is capable of using more than one node to form a new shorter path will be our future work.

Note that the analysis in Sect. 4 does not consider the node mobility effects and packet transmitting speed in determining the threshold values. In future research, we will take into consideration these factors so that our protocol can become more adaptive to the changing environment.

The idea of topology adaptive routing can also be applied to other routing protocols. As a matter of fact, we are planning to integrate our proposed idea into the DSR protocol. Although DSR has a built-in path shortening mechanism, it has yet to rebuild routes beginning from the source node and then update the cache contents of all corresponding nodes. For our proposed mechanisms to be effective, the comparatively static characteristic of DSR needs to be modified, which would be our next primary research direction.

References

1. Perkins, C. E., & Royer, E. M. (1999.) Ad-hoc on-demand distance vector routing. In *Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications* (pp. 90–100).
2. Johnson, D. B., & Maltz, D. A. (1996). Dynamic source routing in ad-hoc wireless networks. *Mobile Computing, 15*, 3–181.
3. Johnson, D. B., Maltz, D. A., & Hu, Y. C. (2002). The dynamic source routing protocol for mobile ad hoc networks (DSR). In *IETF Internet-Draft, draft-ietf-manet-dsr-10.txt*.
4. Corson, M. S., & Park, V. D. (1997). Temporally ordered routing algorithm (TORA) version 1: Functional specification. In *Internet-Draft, draft-ietf-manet-tora-spec- 00.txt*.
5. Perkins, C. E., & Bhagwat, P. (1994). Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications* (pp. 234–244).
6. Royer, E. M., & Chai-Keong, T. (1999). A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications, 6*, 46–55.
7. Broch, J., Johnson, D. B., Maltz, D. A., Hu, Y.-C., & Jetcheva, J. (1998). A performance comparison of multi-hop wireless ad-hoc network routing protocols. In *ACM/IEEE International Conference on Mobile Computing and Networking* (October, pp. 85–97).
8. Das, S. R., Perkins, C. E., & Royer, E. M. (2000). Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies* (pp. 3–12).
9. Saito, M., Aida, H., Tobe, Y., & Tokuda, H. (2004). A proximity-based dynamic path shortening scheme for ubiquitous ad hoc networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems* (pp. 390–397).
10. Gui, C., & Mohapatra, P. (2003). SHORT: Self-healing and optimizing routing techniques for mobile ad hoc networks. In *Proceedings of the 4th ACM International Symposium on Mobile ad hoc Networking and Computing* (pp. 279–290).

Journal : Large 11276 | Dispatch : 11-8-2009 | Pages : 23
Article No. : 203 | ☐ LE | ☐ TYPESET
MS Code : WINET-683 | ☑ CP | ☑ DISK

11. Roy, S., & Garcia-Luna-Aceves, J. J. (2001). Using minimal source trees for on-demand routing in ad hoc networks. In *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies* (pp. 1172–1181).
12. Crisostomo, S., Sargento, S., Brandao, P., & Prior, R. (2004). Improving AODV with preemptive local route repair. In *IEEE International Workshop on Wireless Ad-Hoc Networks* (pp. 223–227).
13. Srinath, P., Abhilash, P., & Sridhar, I. (2002). Router handoff: A preemptive route repair strategy for AODV. In *IEEE International Conference on Personal Wireless Communications* (pp. 168–171).
14. Lee, S.-J., & Gerla, M. (2000). AODV-BR: Backup routing in ad hoc networks. In *IEEE Wireless Communications and Networking Conference (WCNC)* (September, pp. 23–28).
15. Chung, C. M., Wang, Y. H., & Chuang, C. C. (2001). Ad hoc on-demand backup node setup routing protocol. In *Proceedings of 15th IEEE International Conference on Information Networking* (pp. 933–937).
16. Agarwal, A., & Jain, B. (2004). QoS-based on-demand segmented backup routing in mobile ad hoc networks. In *Proceedings of the 12th IEEE International Conference on Networks (ICON)* (pp. 331–335).
17. Chen, H.-L., & Lee, C.-H. (2005). Two hops backup routing protocol in mobile ad hoc networks. In *Proceedings of the 11th International Conference on Parallel and Distributed Systems* (pp. 600–604).
18. Yu, C. W., Wu, T. K., & Cheng, R. H. (2007). A low overhead dynamic route repairing mechanism for mobile ad hoc networks. *Computer Communications, 30*, 1152–1163.
19. Cheng, R.-H., Wu, T.-K., Yu, C. W., & Kuo, C.-H. (2006). An altitude based dynamic routing scheme for ad hoc networks. *Lecture Notes in Computer Science, 4138*, 609–619.
20. Lai, W. K., Hsiao, S.-Y., & Lin, Y.-C. (2007). Adaptive backup routing for ad-hoc networks. *Computer Communications, 30*, 453–464.
21. Sengul, C., & Kravets, R. (2006). Bypass routing: An on-demand local recovery protocol for ad hoc networks. *Ad Hoc Networks, 4*(3), 380–397.
22. Lee, S. J., Royer, E., & Perkins, C. E. (2003). Scalability study of the ad hoc on demand distance vector routing protocol. *International Journal of Network Management, 13*(2), 97–114.
23. Castañeda, R., Das, S. R., & Marina, M. K. (2002). Query localization techniques for on-demand routing protocols in ad hoc networks. *Wireless Networks, 8*, 137–151.
24. Soliman, H., & Al-Otaibi, M. (2009). An efficient routing approach over mobile wireless ad-hoc networks. In *IEEE Consumer Communications and Networking Conference* (Jan., pp. 1–5).
25. Tsai, H.-M., Wisitpongphan, N., & Tonguz, O. K. (2006). Link-quality aware ad hoc on-demand distance vector routing protocol. In *1st International Symposium on Wireless Pervasive Computing*.
26. Boukerche, A., & Zhang, L. (2004). A performance evaluation of a pre-emptive on-demand distance vector routing protocol for mobile ad hoc networks. *Wireless Communications and Mobile Computing, 4*, 99–108.
27. Goff, T., Abu-ghazaleh, N. B., Phatak, D. S., & Kahvecioglu, R. (2003). Preemptive routing in ad hoc networks. *Journal of Parallel and Distributed Computing, 63*(2), 123–140.
28. Zapata, M. G. (2005). Shortcut detection and route repair in ad hoc networks. In *Third IEEE International Conference on Pervasive Computing and Communications Workshops* (pp. 237–242).
29. Wu, S. L., Tseng, Y. C., & Sheu, J. P. (2000). Intelligent medium access for mobile ad hoc networks with busy tones and power control. *IEEE Journal on Selected Areas in Communications, 18*(9), 1647–1657.
30. Fall, K., & Varadhan, K. (1999). Ns notes and documentation. The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, November, 1999.
31. Hogie, L., & Bouvry, P. (2006). An overview of MANETs simulation. *Electronic Notes in Theoretical Computer Science, 150*, 81–101.

Author Biographies

Rei-Heng Cheng received the M.S. and Ph.D. degrees in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, ROC in 1989 and 1995, respectively. Since 1999, he has been on the faculty of Hsuan Chuang University, HsinChu, Taiwan. He is currently an Associate Professor at the Department of Information Management. His research interests include wireless and sensor networks, image processing, and character recognition.

Tung-Kuang Wu received his Ph.D. degree in Computer Engineering from the Department of Computer Science & Engineering at Pennsylvania State University in 1995. He is currently a professor at the Department of Information Management at National Changhua University of Education, Changhua, Taiwan. His current research interests include wireless networks, parallel processing, special education technologies, and e-Learning.

Chang Wu Yu received the B.S. degree from Soochow University in 1985, M.S. degree from National Tsing Hua University in 1989, and Ph.D. degree from National Taiwan University in 1993, Taiwan, all in computer sciences. From 1995 to 1998, he was an Associate Professor at the Department of Information Management, Ming Hsin Institute of Technology. In 1999, he joined the Department of Computer Science & Information Engineering, Chung Hua University. His current research interests include graph algorithms, wireless networks, and distributed computing.

Journal : Large 11276 | Dispatch : 11-8-2009 | Pages : 23
Article No. : 203 | ☐ LE | ☐ TYPESET
MS Code : WINET-683 | ☑ CP | ☑ DISK