A Reliability Aware Flooding Algorithm (RAFA) in Wireless Multi-hop Networks

Youndo Lee, Yoonbo Shim, Yanghee Choi, and Taekyoung Kwon
School of Computer Science and Engineering
Seoul National University, Seoul, Korea
{ydlee,ybshim}@mmlab.snu.ac.kr, {yhchoi,tkkwon}@snu.ac.kr

Abstract— **Flooding is a mechanism that distributes packets to every node of the network. The flooding mechanism is frequently used in many operations in wireless multi-hop networks. Since flooding exploits hop-by-hop broadcasting that suffers from unreliable transmission and fading, it is hard to achieve the reliability in flooding. As unreliable flooding may lead to a coverage hole, it will have a negative effect upon upper layer protocols. In this paper, we introduce a Reliability Aware Flooding Algorithm (RAFA), which estimates the expected reliability using two-hop topology knowledge. The estimated reliability is used for deciding whether or not to retransmit a packet. Using NS-2 [19] simulator, we show that RAFA achieves the higher reliability than RBP [1] by adjusting the number of retransmissions considering the network topology, regardless of the network topologies, the node density or the number of bottlenecks.**

I. Introduction

Flooding is a mechanism that propagates a packet throughout network. Due to its viability, there is a plenty of flooding-based protocols in wireless networks. In fact, most of routing protocols leverage the flooding mechanism. For example, DSR [7] and AODV [8] use a flooding message for discovery, maintenance and update of routes. In Directed Diffusion [9], flooding is used for disseminating interests to sensors. Overall, the flooding mechanism is exploited in sensor networks, MANETs, and vehicular networks, etc. Almost all the above protocols assume that flooding can propagate a packet to every node in a network. However, since flooding commonly exploits hop-by-hop broadcasting that suffers from unreliable link quality, collision and fading, it is hard to achieve the sufficiently high reliability. As a matter of fact, because there is a frequent transmission failure due to the above reasons, when flooding needs to achieve higher reliability, it should be augmented by some mechanism. For this reason, many researchers have proposed a lot of schemes that cope with the collision and/or the link error. For the reduction of collision, PHY-layer capture, MAC-layer TDMA, random slot selection, and application-layer jitter schemes are used. Although these approaches do not guarantee collision-free, it may help reduce collisions. To deal with the link error, there are some studies how to exploit the retransmission mechanism. When the transmission of a packet fails, these schemes increase the reliability by retransmitting at the MAC, network, or application layer. This retransmission mechanism leverages ACKs to figure out whether the transmission of the packet is successful or not. There are two kinds of ACKs, i.e., *explicit* and *implicit* ACKs.

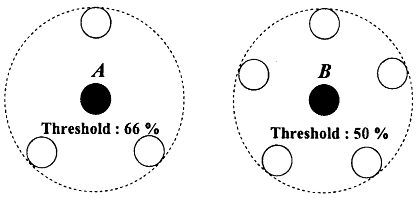

Fig. 1. An example of RBP operation

An *explicit* ACK refers to an ACK packet transmitted by a receiver to confirm the success of transmission, *directly*. While An *implicit* ACK happens as follows. When node A sends a data packet to node B, and overhears B's forwarding the packet to another node. In this way, A confirms that the packet is successfully received by B. The cost of a flooding scheme highly depends on how to combine these two kinds of ACKs, which will be detailed later. To the best of our knowledge, RBP [1] is a state-of-the-art protocol on the retransmission-based flooding mechanism. RBP improves the reliability of flooding using the knowledge about the node density and bottleneck link. A node's retransmission policy is to retransmit the packet only if the ratio of the received ACKs from its neighbors is less than a certain threshold. This threshold is adjusted by the neighborhood density and whether the link with its neighbor is bottleneck or not bottleneck link. The bottleneck link represents the link which uniquely connects two nodes each other and may largely affect the reliability of the network. RBP assumes that the nodes in the network are deployed uniformly and it considers only one source of packets to flood. We propose a Reliability Aware Flooding Algorithm (RAFA), which guarantees the required reliability of flooding. The remainder of this paper is organized as follows. We first state some preliminaries and motivations. After that, we describe RAFA in Section III. In Section IV, we show the performance evaluation. Finally, we conclude our work and discuss the future directions.

II. Preliminaries and Motivations

A. Connectivity between neighbors

RBP [1] improves the reliability of flooding, by using the retransmission mechanism. The retransmission policy of RBP is to perform retransmissions only if the received ACK ratio is less than a certain threshold. The ACK ratio is the number

978-1-4244-2703-1/08/$25.00 ©2008 IEEE

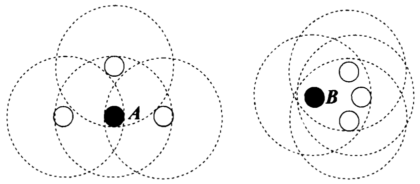

Fig. 2. Node A and B both have four neighbors. However, their reliability will be different each other. The reliability of A will be lower than that of B

of the received ACKs to the number of neighbor nodes. This threshold of each node is determined by the neighborhood density. That is, when the neighborhood density is low, RBP sets the threshold high and when the neighborhood density is high, it sets the threshold low. The intuition behind the above adaptive threshold is that the higher the density of neighborhood becomes, the higher reliability will flooding achieve. See Fig. 1 for an illustration of the RBP retransmission policy. The retransmission threshold of node A, whose neighbor number is 3, is 66% while that of node B, whose neighbor number is 5, is 50%. By adjusting the retransmission threshold according to the number of neighbors, RBP reduces the number of unnecessary retransmissions without having a negative effect on the reliability. As stated above, the intuition behind RBP is that the reliability of the flooding is proportional to the number of neighbors. This intuition seems reasonable but, unless nodes are uniformly distributed, the number of neighbors cannot directly indicate the reliability of flooding. For example, in Fig. 2, nodes A and B have the same number of neighbors, four. However, the reliability of flooding on each topology is not same at all. The reliability of the right topology is much higher than that of the left one because neighbors of B are neighbors of one another. On the other hand, neighbors of A have no neighbors except A. In other words, reliability is also affected by connectivity between neighbors as well as the number of neighbors. In RBP, the retransmission threshold of A and B will be same which leads to either perform unnecessary retransmissions or decrease the reliability.

B. Bottleneck link effect

Another issue is the effect of bottleneck links. In Fig. 3, there is a bottleneck link, i.e., the link between A and B. Success of transmission on that link largely affects the reliability of the network-wide flooding because all of six nodes located on the right side of node B cannot receive the flooding packet if the transmission from A to B fails. The problem is that the bottleneck link may exist irrespective of the number of neighbors. When we consider the number of neighbors only, the bottleneck link may not be detected. Therefore, if the number of neighbors is considered in the retransmission policy (like RBP), the reliability of flooding may be severely poor.

The authors of RBP were aware of the importance of the bottleneck link with respect to the reliability of the flooding. So, they proposed a simple mechanism that finds out the bottleneck link. The proposed mechanism is to make every node record the first sender of the flooding packet. If most of flooding packets arriving first are sent by a particular node, then the link from the node is regarded as the bottleneck link. The effectiveness of this mechanism is affected by the distribution of source nodes of the flooding packets. If there is only one source node or source nodes are gathered together, this mechanism will work well. On the other hand, if source nodes are distributed uniformly, the above mechanism will not be effective in detecting the bottleneck link. This is the severe constraint because any node in the network may be the source of the flooding in many protocols such as AODV, DSR, etc.

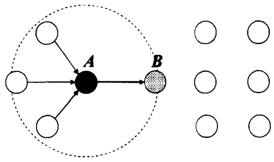

Fig. 3. Topology with a bottleneck link.

Motivated by the above observations, we propose a Reliability Aware Flooding Algorithm, dubbed RAFA, that reflects the network topology better than RBP. Furthermore RAFA estimates the expected reliability using two-hop topology knowledge. Estimated reliability is used for deciding whether or not to retransmit a packet. The details of RAFA is described in the next section.

III. Algorithm Description

In RAFA, both kinds of ACKs are used to enhance reliability. To reduce the number of ACK transmissions, RAFA first exploits *implicit* ACK (like RBP). When the sender learns that a certain neighbor rebroadcasts the flooding packet by overhearing, the sender concludes that the neighbor has received the flodding packet successfully. At the receiver side, the receiver sends back the ACK packet only when receives a duplicate flooding packet from the same sender, *explicitly*.

In addition to ACK scheme, RAFA adopts a retransmission mechanism for reliability. In RAFA, whether or not to retransmit a packet is decided by an *expected reliability*, which is a probability that neighbors that do not send ACK (we call them unconfirmed neighbors) receive the packet by other nodes' flooding. Since a receiver rebroadcasts the packet, unconfirmed neighbors may receive the lost packet on another path, even though the sender does not retransmit the packet. RAFA employs an algorithm that estimates the *expected reliability*. In the following subsections, the proposed algorithm is detailed and the optimization of the proposed algorithm is described.

A. Basic Algorithm

The *expected reliability* is determined by the network topology and link quality. To calculate the *expected reliability* more exactly in a distributed fashion, total network topology and qualities of all of the links in the network must be known to all the nodes. However, in wireless multi-hop networks, the above requirements may not be effective of feasible. Therefore, in RAFA, nodes calculate the approximate values of the *expected reliability*.

The *expected reliability* is estimated in a distributed fashion using the knowledge of the two-hop topology and the quality of all of the links on the two-hop topology. Every node knows this information by exchanging its own neighbor list with its neighbors' lists. Nodes measure the quality of links to their neighbors in a similar way as [2] and [3]; by exchanging sequence-number-stamped packets with their neighbors.

Algorithm 1 Calculate Expected Reliability

- L_{ab} = Quality of link from a to b
- ER_i = Expected Reliability for neighbor node i
- NL_i = Neighbor List of node i
- S = A set of all the nodes that will be excelueded in relaying the flooding packet to node i
- n = A node estimating its neighbor nodes' reliability
- fp = A failure probability that a target node cannot receive the flooding packet

1: ER_i = calculateER(i, $\{n\}$)
2:
3: calculateER(node i, S)
4: **if** node i's reception is acknowledged **then**
5:   return 1
6: **else**
7:   fp = 1
8:   **for all** $a \in (NL_i - S)$ **do**
9:     fp $*= (1 - L_{ai}\cdot$ calculateER$(a, S \cup \{i\}))$
10:   **end for**
11:   return (1−fr)
12: **end if**

Now, we describe the algorithm that estimates the *expected reliability*. In Alg. 1, the *expected reliability* for a particular neighbor node i is estimated by node n as shown in line 1. Node n is the one that is running this algorithm, and the set S whose initial value is $\{n\}$ contains all the nodes that will be excluded in relaying the flooding packet to node i. The *calculateER* function returns 1 if node i's reception is acknowledged (line 5). Otherwise, the return value is calculated by calling *calculateER* for node i's each neighbor recursively. Since the *expected reliability* of node i is equal to the probability that at least one of node i's neighbors delivers the flooding packet to node i, it can be calculated as follows.

$$ER_i = 1 - \prod_{\forall a \in NL_i} (1 - L_{ai} \cdot ER_a) \quad (1)$$

Lines 7-11 realize this equation. In line 8, a neighbor node is excluded from the calculation if it is in the set S, which includes the identifiers of nodes who have called the *calculateER* function. By this condition, a loop is prevented.

This algorithm is the basis for the retransmission policy. In RAFA, the retransmission is triggered when the minimum of all the unconfirmed neighbors' *expected reliability* is less than the target reliability. Alg. 2 shows the retransmission algorithm.

Algorithm 2 Basic retransmission algorithm of RAFA

- ER_{min} = Minimum expected reliability
- TR = Target reliability

$ER_{min} = 1$
for all unconfirmed neighbor u **do**
  ER_u = calculateER(u, $\{n\}$)
  if $ER_{min} > ER_u$ **then**
    $ER_{min} = ER_u$
  end if
end for

if $ER_{min} < TR$ **then**
  Retransmit the packet.
end if

In the above algorithm, the target reliability, TR, can be less than the required reliability by applications. For example, if .99 reliability is required, TR can be set to a certain value less than .99. Since the *expected reliability* is estimated by using only the two-hop topology information, RAFA's *actual reliability* will be higher than the TR. This difference generally increases as the node density becomes higher due to the availability of more alternate paths which are not included in the two-hop topology. Therefore, for the same required probability, we can decrease the TR as the node density increases. The analytic study on the relation between the TR and the required reliability is our future work.

B. Simplified Algorithm

The proposed retransmission algorithm is relatively simple but its computational cost can be very high with high node density. In order to alleviate this problem, we simplify the proposed retransmission algorithm. In the basic retransmission algorithm (Alg. 2), the sender of a flooding packet estimates the *expected reliability* for all the unconfirmed neighbors because the sender needs to know the minimum of the *expected reliability* of unconfirmed neighbors for retransmission decision. We streamline this part by inferring the minimum of *expected reliability* without calculations for all of the unconfirmed neighbors.

The idea is to use the number of confirmed common neighbors to reduce the computation overhead of estimating the reliability. The confirmed common neighbors for an unconfirmed neighbor u are nodes whose receiving the packet is acknowledged and are common neighbors of both nodes n and u. We denote the number of confirmed common neighbors by NC, and NC for a certain neighbor u by NC_u. We consider that NC_u is roughly proportional to the *expected reliability* of u. Thus, the minimum of the *expected reliability* is approximated by the *expected reliability* of a neighbor whose NC is the minimum. Alg. 3 shows the modified retransmission algorithm. We denote this modified version by RAFA-NC. Since RAFA-NC performs calculateER only once,

Algorithm 3 Simplified retransmission algorithm of RAFA

- NC_i = Number of confirmed neighbors among the common neighbors of nodes n and i
- NC_{min} = Minimum of NC_i for every neighbor i
- N_{NC} = Identifier of a node who has NC_{min}

NC_{min} = number of neighbors of the sender + 1
for all unconfirmed neighbor u **do**
  if $NC_{min} > NC_u$ **then**
    $NC_{min} = NC_u$
    $N_{NC} = u$
  end if
end for

if calculateER($N_{NC}, \{n\}$) $< TR$ **then**
  Retransmit the packet.
end if

the computational cost is significantly reduced. Note that the computational cost of deriving NCs is much lower than that of estimating the *expected reliability*.

IV. Performance Evaluation

In this section, we evaluate the performance of RAFA and its simplified version, called RAFA-NC by simulations using NS-2 [19]. We compared RAFA and RAFA-NC with classic flooding and RBP but results of classic flooding have been omitted in this paper due to its poor performance.

Simulations consist of two scenarios. In the first scenario, we examine the effect of bottleneck links on flooding performance. Fig. 4 shows topologies used for the first scenario. Simulations have been performed for each topology. In each simulation, a randomly chosen node floods a packet to the whole network. The interval between two consecutive floods is 20 seconds and the number of floods is 200. The node that floods a packet is randomly chosen for each flood. There are no other control or application traffic except flooding packets.

In the second scenario, we evaluate the performance with random topologies. The simulation area is an 1000m x 1000m rectangle space. The number of nodes are 10, 15, 20, 25 or 30, which are randomly distributed in the simulation area. 10 topologies are randomly generated for each number of nodes and simulations have been performed for each topology. Like the first scenario, a randomly chosen node floods a packet to the whole network in each simulation. The interval between floods is 20 seconds and the number of floods is 50 times. There is only flooding traffic in the network.

PHY/MAC layer environments are common for both scenarios. IEEE 802.11 is used for MAC protocol. The bandwidth is 1Mbps and the transmission range is 250m. The packet error model is the uniform error model, and the error rate is set to 0.4.

We evaluate three metrics: *reliability*, *normalized number of transmitted packets (NNP)*, a *reliability cost metric (RCM)* and the overhead of ACK traffic. *Reliability* means the percentage of nodes that receive a flooding packet. *NNP* is the *total number of transmitted packets / number of nodes / number of floods*. This metric indicates the average packet transmission cost for each flood per node. *RCM* [1] is defined as the number of packet transmissions per node to achieve the near-perfect (99%) reliability. *RCM* is the product of *NNP* and the number of floods, F, that is requited to achieve the near-perfect reliability. Given the *reliability* R, F is derived from the following equation.

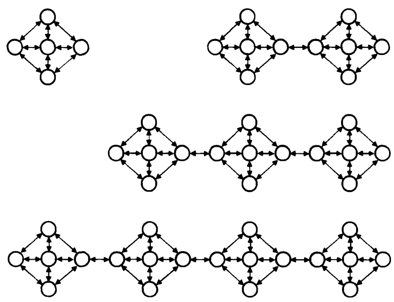

Fig. 4. Topologies with zero or more bottleneck links

$$.01 = (1 - R)^F \tag{2}$$

Note that we treat F as a real value average for fine-grained comparison, although the number of floods is an integer.

A. Effect of Bottleneck Links

Figs. 5-8 show the results of the first scenario. In all figures, the performance results of RAFA and RAFA-NC are almost same. Therefore, we mention RAFA only. Fig. 5 shows the *reliability* with respect to the number of bottleneck links as shown in Fig. 4. RAFA achieves the higher *reliability* than RBP irrespective of the number of bottleneck links and the gain increases rapidly as the number of bottleneck links increases. The *reliability* of RAFA remains almost constant with respect to the number of bottleneck links. Since RAFA calculates the *expected reliability* by exploiting the two-hop topology, the existence of bottleneck links doesn't affect the *reliability* of RAFA. On the contrary, the *reliability* of RBP is degraded rapidly as the number of bottleneck links increases. This is because the source node of flooding is determined randomly for each flood. As stated in Section II.B, RBP's identifying mechanism of bottleneck links is ineffective where source nodes are distributed.

Fig. 6 depicts *NNP* with the varying number of bottleneck links. *NNP* of RAFA is higher than that of RBP in all cases. This means that RAFA triggers more retransmissions than RBP. In such cases, the important question is whether it is the necessary retransmission or not. The answer can be inferred from Fig. 5; the *reliability* of RAFA is higher than that of RBP at all times. This means that extra retransmissions triggered by RAFA are needed to achieve the required reliability.

RCM with the varying number of bottleneck links is plotted in Fig. 7. *RCM* of RAFA is always lower than that of RBP,

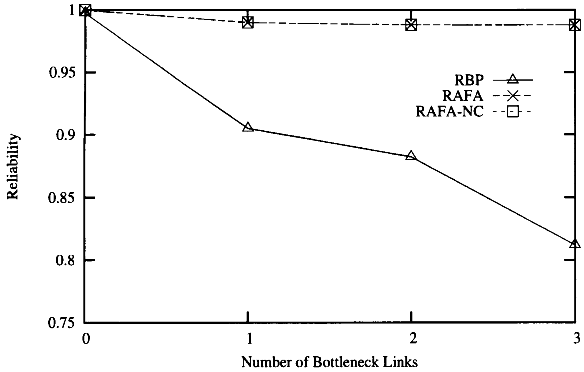

Fig. 5. Reliability, topologies with bottlenecks

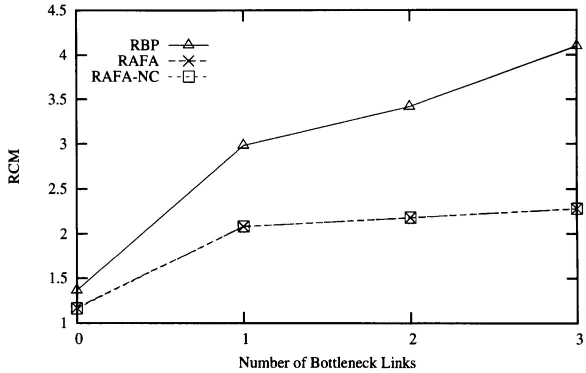

Fig. 7. RCM, topologies with bottlenecks

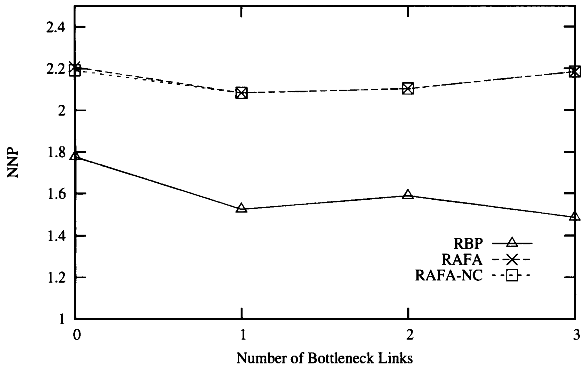

Fig. 6. NNP, topologies with bottlenecks

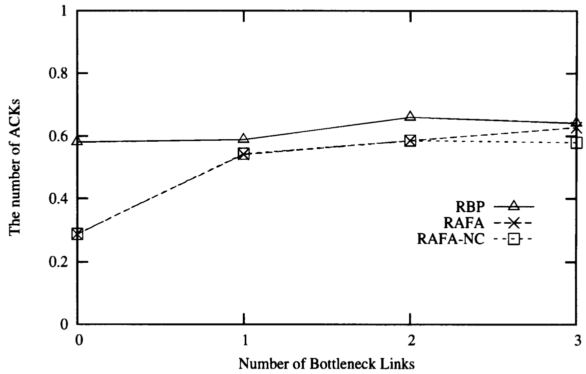

Fig. 8. The number of used explicit ACK per flooding of each node, topologies with bottlenecks

which means that RAFA achieves the near-perfect *reliability* with lower cost than RBP. The difference of RCM between RAFA and RBP increases as the number of bottleneck links increases. This is due to the ineffectiveness of RBP in detecting bottleneck links.

We plot the overhead of ACK traffic in Fig. 8, which is the number of explicit ACKs per flooding for each node. Although the ACK overhead of RAFA is growing with the more bottlenecks, it is necessary to satisfy the target reliability. Meanwhile, the reliability of RBP is decreased in the network that has many bottlenecks. Because RBP is used almost the same number of explicit ACKs regardless of the network topology, the reliability is dropped shown in Fig. 5.

B. Measurement on Random Topology

In this section, we show and discuss the results from the second scenario, i.e., simulations on the random topology through Figs. 9-12. The performance and the tendencies of RAFA and RAFA-NC are almost same, like the results of the first scenario; thus, we only mention RAFA. In Fig. 9, the reliability of RBP is lower than that of RAFA in relatively sparse topologies and becomes comparable to RAFA in dense environments. The reason is explained in Fig. 10 which shows that due to be the shortage of the needed retransmissions, RBP achieves in sparse topologies. In dense topologies, RBP raises the reliability at the cost of high retransmissions. Overall, RAFA achieves high reliability by adjusting the number of retransmissions, regardless of network density. In this context, Fig. 12 shows that the number of explicit ACKs of RBP remain high, while RAFA reduces the ACK overhead as the network becomes dense.

Fig. 11 shows the *RCM* of RAFA, RAFA-NC and RBP with respect to the number of nodes. *RCM* of RBP is higher than that of RAFA by about 28% when there are 10 nodes, while the difference of *RCM* between RBP and RAFA is reduced when the network becomes dense. The *RCM* of RBP is decreased by about 32% as the number of nodes becomes from 10 to 30. This result indicates that the performance of RBP is more affected by the node density than RAFA. RBP achieves higher performance in the dense network than in the sparse network. As the node density becomes higher, it is more likely that nodes are uniformly distributed. RBP performs best where nodes are uniformly distributed as stated in Section II.B. This is why the performance of RBP increases as the number of node increases. In case of RAFA, the node density affects the performance less than RBP. Although there is a little decrease of *RCM* in RAFA, in Fig. 11, as the number of nodes increases, it is due to the increase of chances receiving the lost packet through other paths without retransmissions.

V. Conclusion

In this paper, we present Reliability Aware Flooding Algorithm (RAFA) in wireless multi-hop networks. It decides whether to retransmit the flooding packet by estimating the expected reliability with only two-hop neighbor information. To

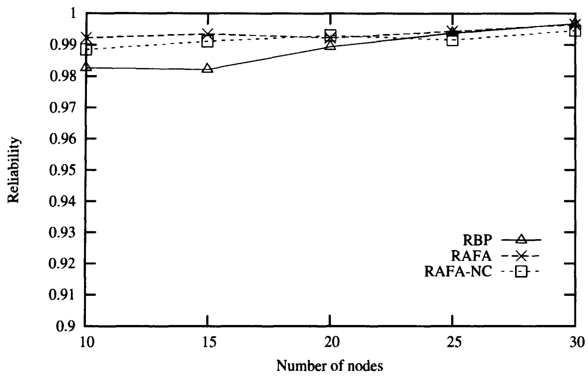

Fig. 9. Reliability, random topology

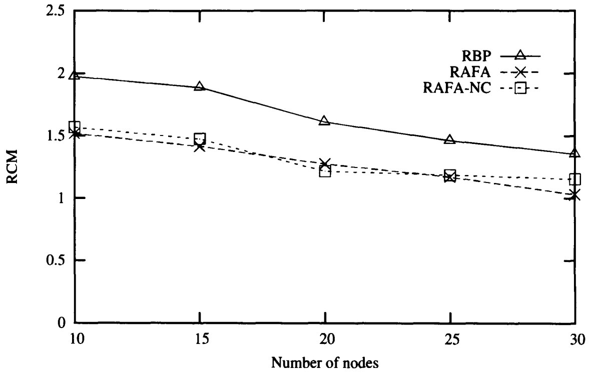

Fig. 11. RCM, random topology

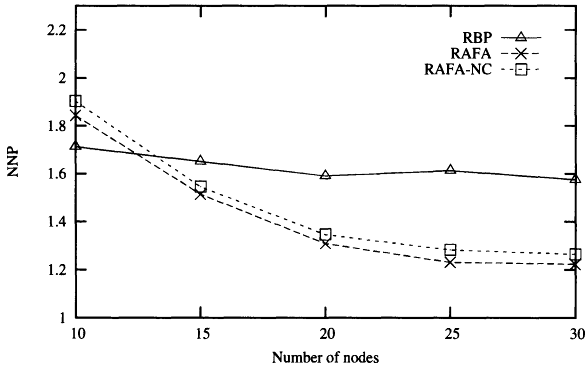

Fig. 10. NNP, random topology

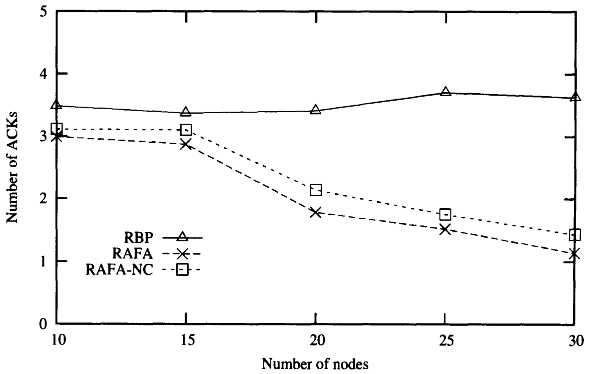

Fig. 12. The number of used explicit ACK per flooding of each node, random topology

reduce the computational overhead of estimating the reliability, we also devise a simplified version, RAFA-NC, which takes into account the number of confirmed common neighbor for each unconfirmed neighbor. With extensive simulations using NS-2 , we validated RAFA, achieves the higher reliability than RBP by adjusting the number of retransmissions considering the network topology, regardless of the network topologies, the node density or the number of bottlenecks.

REFERENCES

[1] Fred Stann, John Heidemann, Rajesh Shroff, and Muhammad Zaki Murtaza, "RBP: Robust Broadcast Propagation in Wireless Networks," *Proc. ACM Sensys'06*, November 2006.

[2] Q. Cao, T. Abdelzaher, T. He, and R. Kravets, "Cluster-Based Forwarding for Reliable End-to-End Delivery in Wireless Sensor Networks," in *Proc. IEEE Infocom 2007.*

[3] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," *Proc. ACM Sensys'03*, 2003.

[4] I. Chlamtac, M. Coti, J J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges," *Ad Hoc Networks 1(2003)*, pp. 13 64, Elsevier, 2003.

[5] C. Perkins and P. Bhagwat. "Highly dynamic destination-sequenced distancevector routing (DSDV) for mobile computers," in *Proc. ACM SIGCOMM 94*, 1994.

[6] T. Clausen, P. Jaqcuet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol (OLSR)," RFC 3626, October 2003.

[7] D. B Johnson and D. A Maltz. "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[8] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-demand Distance Vector (AODV) Routing," *IETF Experimental RFC*, MANET working group, RFC 3561, July 2003

[9] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proc. ACM MOBICOM 2000*, August 2000.

[10] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *Proc. ACM MOBICOM 2000*, August 2000.

[11] S. Keshav, *An Engineering Approach to Computer Networking - ATM Networks, the Internet, and the Telephone Network,* Addison-Wesley, Reading, MA, 1997.

[12] F. Kuhn, R. Wattenhofer, Y. Zhang and A. Zollinger, "Geometric Routing: Of Theory and Practice," in *Proc. of the 22nd ACM Symposium on the Principles of Distributed Computing (PODC)*, 2003.

[13] J. Li, J. Jannotti, D. DeCouto, D. Karger and R. Morris, "A Scalable Location Service for Geographic Ad-hoc Routing," in *Proc. ACM MOBICOM 2000*, August 2000.

[14] Zhaomin Mo, Hao Zhu, Kia Makki and Niki Pissinou, "MURU: A Multi-Hop Routing Protocol for Urban Vehicular Ad Hoc Networks," in *Proc. MOBIQUITOUS 2006*, July 2006.

[15] Fabrizio Granelli, Giulia Boato and Dzmitry Kliazovich, "MORA: a Movement-Based Routing Algorithm for Vehicle Ad Hoc Networks," in *Proc. IEEE Workshop on Automotive Networking and Applications 2006*, December 2006.

[16] Tarik Taleb, Mitsuru Ochi, Abbas Jamalipour, Nei Kato and Yoshiaki Nemoto, "An Efficient Vehicle-Heading Based Routing Protocol for VANET Networks," in *Proc. IEEE WCNC 2006*, April 2006.

[17] Jing Zhao and Guohong Cao, "VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks," in *Proc. IEEE INFOCOM 2006*, April 2006.

[18] Hojin Lee, Youndo Lee, Taekyoung Kwon and Yanghee Choi, "Virtual Vertex Routing (VVR) for Course-Based Vehicular Ad Hoc Networks," in *Proc. IEEE WCNC 2007*, March 2007.

[19] "The Network Simulator - NS-2," http://www.isi.edu/nsnam/ns/, Online link.