Using Connection Expansion to Reduce Control Traffic in MANETs

Zeki Bilgin
Department of Computer Science
The Graduate Center, CUNY
New York, 10016
zbilgin@gc.cuny.edu

Bilal Khan
Department of Math.& Computer Science
John Jay College, CUNY
New York, 10016
bkhan@jjay.cuny.edu

Ala Al-Fuqaha
Department of Computer Science
Western Michigan University
alfuqaha@cs.wmich.edu

ABSTRACT

We consider the problem of control traffic overhead in MANETs with long-lived connections, operating under a reactive routing protocol (e.g. AODV). In such settings, control traffic overhead origins can be traced principally to connection link failures, which trigger expensive global route discoveries. In this paper, we introduce a route maintenance scheme developed with the objective of reducing global route discoveries in such settings. The proposed scheme decrease the expected number of route discovery attempts by taking preemptive action to counteract impending link disconnections due to *node movement*. The proposed scheme was implemented as an extension of AODV in ns2, and compared with the standard AODV under different network regimes. Through the analysis of data derived from extensive simulations, we demonstrate that the proposed scheme significantly decreases overall control traffic while maintaining comparable packet delivery rates, at the cost of only very minor degradation in path optimality.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: -Networks Protocols–*Routing Protocols*

General Terms

Algorithms, Design, Performance

Keywords

MANET, mobility, route maintenance, link expansion

1. INTRODUCTION

The problem of routing has been the subject to extensive research efforts, both in static settings, where nodes are fixed, and dynamic settings, in which they are mobile. In static networks, the problem of routing is relatively easy, because the problem needs to be solved infrequently. Once a route is established for a pair of source and destination nodes, it is likely to be valid a long time of period because link failures and link formation is rare in static networks. However, this is not the case for dynamic networks. In these settings, routing is a more challenging problem since in the presence of mobility, node movement induces frequent network topology changes, causing both the *formation* of new links and the *disconnection* of existing ones.

Disconnection of links generally entails extra work for routing protocols, since existing connection paths must be repaired. More precisely, when a link on an active connection is broken, the routing protocol generally reinitiates a local or global route discovery operation, which usually requires flooding–an expensive operation in terms of control traffic incurred. Strangely, the dual event–i.e., node mobility causing the creation of new links resulting in the *connection* of previously disconnected nodes–presents a resource for the routing protocol, that is, for the most part, ignored. In this work, we use this resource. Specifically, we develop a route maintenance scheme that reduces overall control traffic overhead by pre-emptively and surgically replacing the weak links within a connection on which failure is immanent.

2. RELATED WORK

One of the earliest studies in this field is the work of Park and Voorst [3], who presented an algorithm called "Anticipated route maintenance" to predict whether a link between two nodes will break within a predefined time interval. The authors use the node locations and velocities, as determined via GPS, to derive their likelihood estimates. Park and Voorst's algorithm consist of two phases: Expand and Shrink. The Expand routine prevents the route from being broken by inserting bridge nodes into a weak link. The Shrink routine eliminates unnecessary hops and shortens the path, thereby preventing it from being unnecessarily long. The implementation of the scheme and performance studies of their scheme were published subsequently in [1]. Unfortunately, their scheme requires GPS, and its implementation requires nodes along a connection to exchange routing table information.

In contrast to Park and Voorst's GPS-based approach, Qin et al. [4] propose a link breakage prediction algorithm based on the change in the signal strength of consecutive received packets. When a node estimates that there is an incident link which is likely to be broken soon, it initiates a "Broken route message" and requests the source node to find

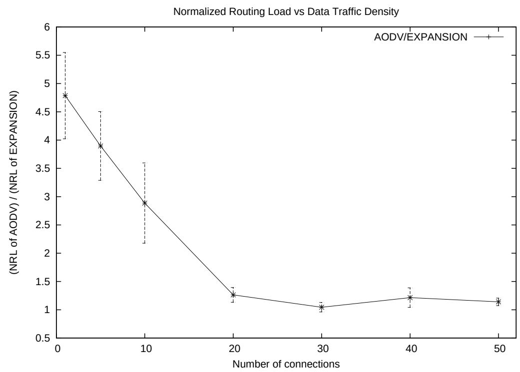

Figure 6: NRL vs. Traffic load

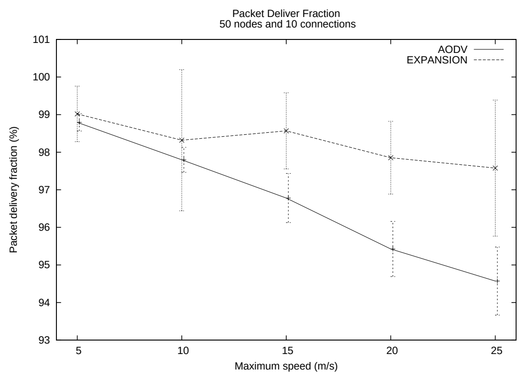

Figure 7: Packet Delivery Fraction vs. Mobility

overhead, without compromising on packet delivery fraction. The question is, at what price is this achieved? Clearly, since the scheme relies on path expansion, the scheme will be worse than pure AODV when it comes to path optimality.

Path Optimality: In this set of experiments, we calculated for each data packet delivered to its destination, the number of hops that the packet traveled in excess of the length of the optimal source-destination path (at the time of packet delivery). The histogram below shows the relative frequencies of the discrepancy from optimal, for both $AODV$ and $AODV + Expansion$ schemes. The experimental distribution was determined by averaging the histograms of 10 experiments, each having 10 random connections among 50 nodes moving at a maximum velocity of 5 m/s.

As expected, both AODV and AODV+Expansion make data packets travel on paths that are longer than optimal. The histogram makes apparent that $AODV + Expansion$ tends to make more packets travel on longer paths than pure $AODV$, yet the increase is quite modest. Interpreting the distribution in Figure 8 cumulatively, one can verify that $AODV$ delivers 92% of the data packets on paths whose length is at most 1 more than optimal. In contrast, $AODV + Expansion$ delivers 92% of the data packets on paths whose length is at most 3 more than optimal, and only 68% of the data packets on paths whose length is at most 1 more than optimal. The results are to be expected, since the Expansion scheme causes packets to travel on paths even further from optimal than those selected by AODV–this is because the *Expand* operation favors forming longer paths over incurring global route discovery when connection links fail. The *Shrink* operation balances the Expand operation to some extent, however, as is apparent from the fact that the distributions in Figure 8 are quite close.

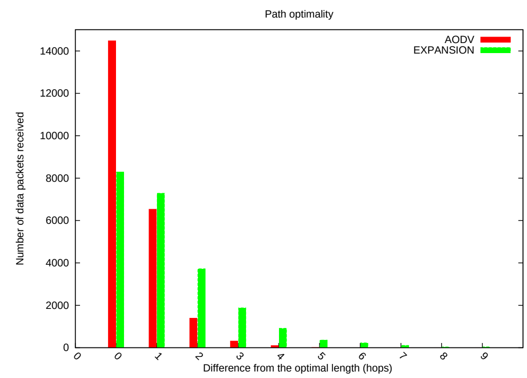

Figure 8: Distribution of path length discrepancy from optimal.

6. CONCLUSION

In this paper, we described a scheme called *Expansion* as an extension to AODV. The scheme tries to prevent link breaks on active connections by taking preemptive actions against them. The advantages of the proposed scheme are that it decreases overall control traffic overhead of the network while maintaining packet delivery fraction. The scheme pays for these improvement through a degradation in path optimality, which is seen to be quite modest. These conclusions were validated by extensive simulations using ns-2.

Acknowledgments

The first author thanks the Turkish National Police for supporting his graduate studies at City University of New York, of which this research is a part.

7. REFERENCES

[1] M. Al-Shurman, S.-M. Yoo, and S. Park. A Performance Simulation for Route Maintenance in Wireless Ad Hoc Networks. In *ACM-SE 42: Proceedings of the 42nd annual Southeast regional conference*, pages 25–30, New York, NY, USA, 2004. ACM.

[2] C.Perkins, E.B-Royer, and S.Das. Ad hoc on-demand distance vector (aodv) routing. Internet Engineering Task Force, 2003.

[3] S. Park and B. Voorst. Anticipated Route Maintenance (ARM) in Location-Aided Mobile Ad Hoc Networks. Globecom, San Antonio, TX, USA, 2001.

[4] L. Qin and T. Kunz. Pro-active Route Maintenance in DSR. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(3):79–89, 2002.

[5] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[6] M. F. Sjaugi, M. Othman, and M. F. A. Rasid. A New Distance Based Route Maintenance Strategy for Dynamic Source Routing Protocol. *Journal of Computer Science*, 4(3):172–180, 2008.