

AODVjr, AODV Simplified

Ian D. Chakeres^a

idc@engineering.ucsb.edu

^aElectrical and Computer Engineering Department, University of California, Santa Barbara, USA

^bNational Institute of Standards and Technology, Gaithersburg, Maryland, USA

Luke Klein-Berndt^b

luke.klein-berndt@nist.gov

In this paper AODVjr, a simplified version of the AODV protocol, is described. AODVjr is compared in simulation to a full featured AODV implementation. The results show that AODVjr performs as well as AODV and describes other positive effects of a smaller protocol specification.

I. Introduction

The Ad hoc On-demand Distance Vector (AODV) routing protocol [4] provides a method of routing in mobile ad hoc networks. AODV has many great features:

- Built for mobile networks
- Creates route on-demand
- Loop free with quick convergence
- Scales well
- Fits easily into the existing protocol stack

For these reasons, AODV is currently the easiest and most widely implemented MANET protocol.

As authors of AODV implementations we found that though AODV is simple in comparison to other MANET routing protocols, the specification still contains many sections prone to erroneous programming. AODVjr is a trimmed down AODV specification which removes all but the essential elements of AODV. This paper shows that AODVjr has nearly the same performance as AODV.

II. AODVjr Description

AODVjr removes the following items from the AODV specification [3].

- Sequence Numbers
- Gratuitous RREP
- Hop Count
- Hello Messages
- RERR
- precursor lists

To accomplish this AODVjr requires slightly different operation when compared to AODV. Removing sequence numbers requires the destination to respond to RREQ (see figure 1a); no intermediate nodes may respond. This also eliminates the need for Gratuitous RREP since all routes will be bidirectional. Since the destination will only respond to the first RREQ it receives the “best” (fastest) route is always chosen regardless of the number of hops.

To perform route maintenance route lifetimes are only updated by the reception of packets and not the sending of packets. This requires the destination to occasionally send a packet to the source. If data traffic is unidirectional periodic messages (*connect*) are sent to maintain the route (see figure 1b). If data communications are bidirectional,

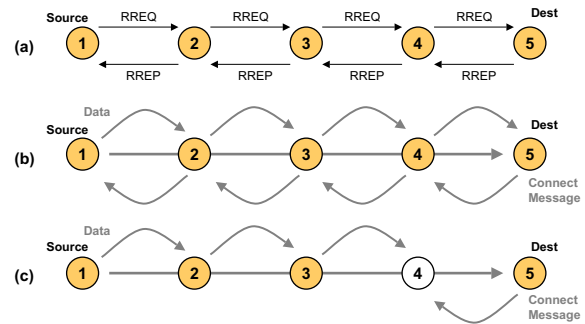


Figure 1: AODVjr Operations.

no additional overhead is needed. Using this end-to-end strategy, hello messages, RERR and precursor lists are not needed.

When a break in the route occurs, the source will stop receiving messages from the destination. In figure 1c, node 4 leaves the route. After a period of time node 1 detect the route is broken because it has not received a message from the destination and will reinitiate route discovery if the route is still needed.

III. Simulation and Results

To examine the performance of AODVjr, the ns-2 [2] simulator was utilized. The base AODV code was a version updated by Marina Mahesh. The AODVjr code was developed for this study. IEEE 802.11 at 2 Mbps was used as the physical, data link and MAC layer protocols. The mobility model uses the random waypoint model in a square field. Multiple size fields were used - 500 m x 500 m field with 25 nodes (default), 750 m x 750 m field with 50 nodes and 1000 m x 1000 m field with 100 nodes. Simulations were run with a maximum speed of five meters per second (unless otherwise noted). Node speeds are uniformly distributed between zero and the maximum speed. For each simulation there are ten sources. Each source sends five 512-byte data packets per second. Each data point represents an average of five runs with varying initial positions and movement, but identical data traffic. Three protocols are simulated AODVjr, AODV with hello messages (AODV) and AODV with link layer feedback (AODV LL).

The first set of simulations examines how AODVjr compares with AODV while varying mobility, network size and

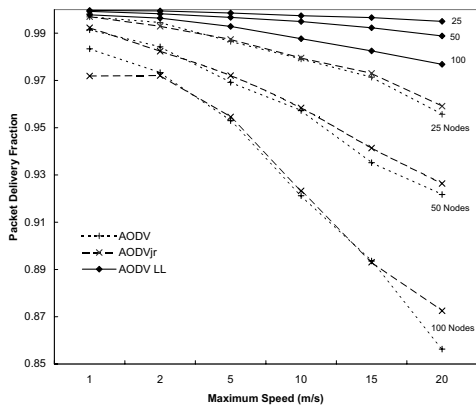


Figure 2: Packet delivery fraction.

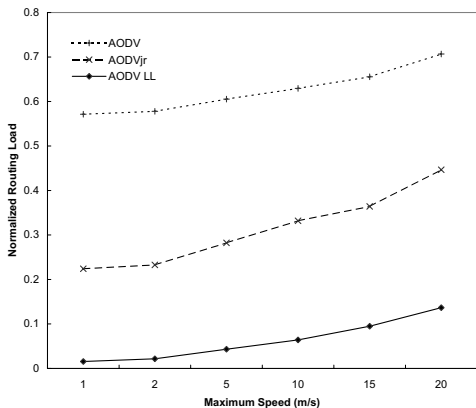


Figure 3: Normalized routing load.

number of nodes. Figure 2 shows the packet delivery fraction. AODV LL performs excellently, but is not realizable because current hardware does not support link layer feedback. AODVjr and AODV have near identical packet delivery fractions for all speeds. Though AODVjr outperforms AODV when the network size is large and the mobility is high.

Figure 3 shows the normalized routing load for a network of 25 nodes. AODV LL has the lowest overhead, since there is no periodic messaging needed for route maintenance. AODVjr has more routing overhead (due to *connect* messages), but is still nearly half that of AODV. The high normalized routing load in AODV is due to the broadcast of hello messages by every node. The real-life impact of hello messages is even larger than that displayed in this graph, because in 802.11 broadcast messages must be processed by each receiving node. While *connect* messages are unicast and therefore only processed by nodes participating in data routing.

To further highlight the savings from not using hello messages figure 4 shows the number of control packets with varying network size. AODV using hello messages, the total control overhead increases linearly with the network size. While using *connect* messages, the total control overhead is related to the number of active unidirectional

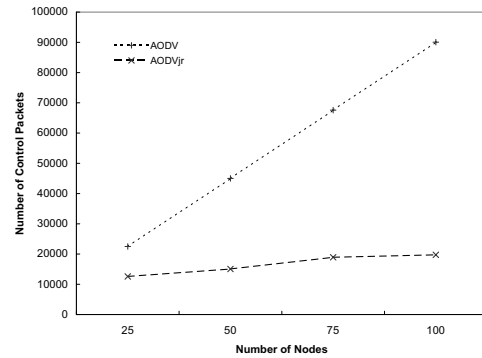


Figure 4: Control packet overhead.

traffic sessions. Additionally as described in section II, if the data traffic is bidirectional there is no associated overhead for maintaining the routes in AODVjr.

IV. Conclusion

In this paper it has been shown that AODVjr has nearly the same performance as AODV. This motivates examination of the AODV protocol, as many of the protocol semantics provide little added benefit, under the conditions examined in this paper.

AODVjr could easily be extended to incorporate optimizations that are standard in AODV with ease, for example sequence numbers, RERR, link layer feedback. Another significant feature of AODVjr is the control packets contain no mutable fields, this allows security to be added easily. An approach similar to ARIN [1] could be applied to secure routing.

One immeasurable quality of AODVjr is simplicity. As authors of AODV implementations we predict AODVjr would take less than half the time to program and debug when compared with a full AODV implementation. Making implementation easy is necessary for wide deployment of the AODV protocol.

References

- [1] Bridget Dahill, Kimaya Sanzgiri, Brian Levine, Elizabeth Belding-Royer, and Clay Shields. A Secure Routing Protocol for Ad Hoc Networks. 2002. (Submitted for publication).
- [2] Kevin Fall and Kannan Varadhan. ns Manual. <http://www.isi.edu/nsnam/ns/doc/>. The VINT Project.
- [3] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. Ad hoc On-Demand Distance Vector (AODV) Routing Protocol. *IETF Internet Draft, draft-ietf-manet-aodv-10.txt*, January 2002. (Work in Progress).
- [4] Charles E. Perkins and Elizabeth M. Royer. The Ad hoc On-Demand Distance Vector Protocol. In Charles E. Perkins, editor, *Ad hoc Networking*, pages 173–219. Addison-Wesley, 2000.