Routing Protocol Performance for Mobility in IoT Nodes

By Saily Ghodke and Sui Kun Guan

For Professor Melody Moh
Computer Communication Systems

San Jose State University
October 2016

Table of Contents

Acronyms

IoT	Internet of Things
AODV	Ad-hoc On-demand Distance Vector Routing
DSR	Dynamic Source Routing
PDR	Packet Delivery Ratio

1.Introduction

IoT is the face of the new world of technology and is exponentially growing in terms of its size and popularity. There are thousands of millions of IoT devices in use today, that make life faster, easier and connected. All these devices can be visualized to be a complex network of nodes, that could either be infrastructure based or be self-organizing, dynamic and without any central access point. These nodes move within the network, from one location to the other, with different velocity, consuming different bandwidth, carrying different data and routing across different paths. The nodes communicate between each other. Maintaining connectivity and reliability among the nodes, is a major challenge in IoT. Determining the optimal route, that has the fastest delivery of data, highest throughput, highest PDR and the lowest cost in terms of bandwidth, time etc. is a crucial part to achieve efficient mobility in IoT nodes. The following sections describe two routing protocols - AODV and DSR and compare their performance in terms of throughput and PDR.

Section II describes the existing solution. Section III gives the system architecture in order to implement the solution described in section I. Section IV describes the project design and section V provides the performance analysis and result. Section VI describes an enhanced DSR with a variation of Nagle’s algorithm.

2. Existing Literature

2.1 Ad-hoc On-demand Distance Vector Routing

AODV belongs to the reactive class of routing protocols. The nodes operating on AODV do not maintain a routing table that contains information about the network topology. As its name suggests, AODV is an on-demand routing protocol, suited to routing in ad hoc networks. In this protocol, every node calculates its next destination node based on the current network topology and traffic. What distinguishes AODV from the other reactive routing protocols is the inclusion of a destination sequence number, which is useful to maintaining fresh routes while routing a packet. The routing is done on a per-flow basis, where the source node and the intermediate nodes carry the information of only the next neighbor. The nodes discover their neighbor by sending a “hello” message. When the neighbors are discovered, a RREQ message, consisting of source identifier, destination identifier, source sequence number, destination sequence number, broadcast identifier and TTL value, is sent to all the neighbors. The TTL value is incremented with every subsequent packet. Every node maintains its own sequence number and updates its information only if the current destination sequence number is greater than the previous destination sequence number stored for that particular node [1].

2.2 Dynamic Source Routing

Similar to AODV (section 2.1), DSR is a reactive routing protocol i.e. the nodes do not maintain a routing table containing the entire network topology and the routing between the nodes is based on the current traffic and network topology. It operates on the principles of routing and caching. The DSR protocol consists of two phases - Route discovery and Route maintenance. In the Route discovery phase, the source nodes flood the network with a RREQ packet. Each node that receives this packet can take either of the following three actions: If the receiving node is the destination node, it receives the packet; If the receiving node has received the packet before, it drops the packet; If the receiving node is not the destination node, it simply forwards the packet to the next node. The Route maintenance phase operates by sending RRPL packet, which are transmitted along the same path that was taken by the RREQ packet. The data packet carries the complete path to be traversed [1].

2.3 AODV v/s DSR

The following sections describe the experiment that compares AODV and DSR and analyzes its results.

3. System Architecture- simulation

3.1 Software used

In this simulation, ns-3 network simulator will be used to evaluate the performances of AODV and DSR routing protocol under large number of IoT nodes.

3.2 Experimental setup for ns3

In the ns3 simulation, we will perform the simulation for 50, 100, 200 IoT nodes under a random walk mobility model for AODV and DSR routing protocol. The simulation parameters details are given in the table below:

Routing Protocol	**AODV vs. DSR**
Simulation Duration	100 seconds
Start-up Time	50 seconds
Number of Nodes	50, 100, 200
Mobility Model	Random way point
Mobility Speed	20 m/s
Mobility Region	300 x 1500 m^2
Pause Time	0
WiFi Mode	Ad-hoc
WiFi Layer 2 protocol	802.11b
WiFi rate	2 Mbps
WiFi loss model	Friis
Transmission Power	7.5 dBm
Number source/sink pairs	20
Transportation Layer protocol	UDP
Application data rate	2.048 Kbits/s (4 64-byte packets per second)
Application data sending start time	Randomly between the 50 and 51 second

Application data sending end time	Until the end of the simulation

Table 1

4. Project design

We utilized an existing ns-3 example “manet-routing-compare.cc” that comes from the ns-3 package (ns-allinone-3.20.tar.gz). In addition, we modified the example to use our experimental parameters, and run the following ns3 simulations independently. We modified the program command line arguments so that different simulation setups can be done through different sets of arguments:

1. AODV with 50 nodes
2. AODV with 100 nodes
3. AODV with 200 nodes
4. DSR with 50 nodes
5. DSR with 100 nodes
6. DSR with 200 nodes

5. Performance Evaluation Result

5.1 Throughput

Figure 1: Throughput comparison

From the throughput simulation result, we plot the calculated average throughput (kbps) vs number of nodes in the system. The graph reveals the following results:

1. As number of node increases, both AODV and DSR average throughput decreases.
2. In AODV, the throughput decreases in a much faster rate than in the DSR as number of node increases
3. For the same number nodes, DSR has better throughput than AODV. In addition, DSR provide approximately 10 times more throughput than AODV when number of nodes is large.

5.2 Packet Delivery Ratio

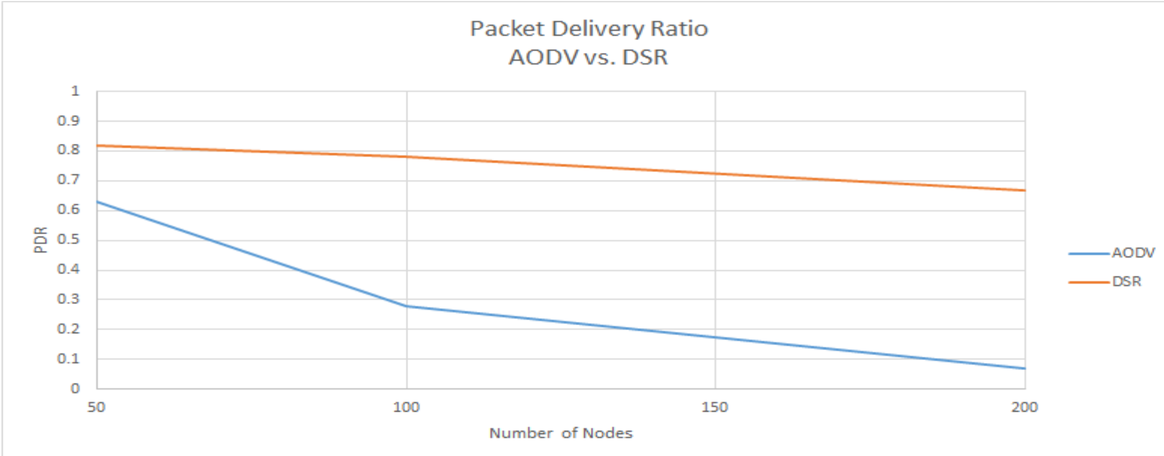

Figure 2: PDR Comparison

From the PDR simulation result, we plot the calculated PDR vs number of nodes in the system. The graph also reveals the consistent results with the throughput analysis:

1. PDR decreases as the number of nodes increases.
2. For DSR, even though number of nodes increase significantly from 50 to 200, the PDR is able to maintained between 0.9 to 0.7
3. For AODV, when the number of nodes increase significantly from 50 to 200, the PDR decreases drastically from around 0.65 to less than 0.1

6. DSR Enhancement

6.1 DSR Performance in high bandwidth, small packet size

In certain scenario, many real time applications often require continuously streaming with small packet size, but at a large bitrate. For example, in a real time health IoT application, IoT sensors may constantly monitoring user's' health information and transmit the data very frequently to the server. In this case, DSR routing performance may decrease because many small packets from different IoTs could flooding the network simultaneously, which can decrease the overall throughput and PDR.

6.2 DSR Enhancement

In TCP, Nagle's Algorithm is a well-known algorithm to enhance the rate control in TCP. However, there is no rate control in UDP application that uses DSR routing. Therefore, we implemented a variance of the Nagle's algorithm in the DSR routing, and performed the simulation in NS3 to compare the DSR routing with and without Nagle's Algorithm.

6.3 Nagle's Algorithm

Nagle's Algorithm is a simple algorithm that try to accumulated small packet within certain period (~ order of milliseconds), and sent it as one large packet to save the overhead caused by these individual packets, with the trade off of small acceptable delay. The algorithm works as follow [2]:

> **if** there is new data to send
>   **if** the window size >= MSS **and** available data is >= MSS
>   send complete MSS segment now
>   **else**
>   **if** there is unconfirmed data still in the pipe
>   enqueue data in the buffer until an acknowledge is received
>   **else**
>   send data immediately
>   **end if**
>   **end if**
>   **end if**

6.4 Implementation of Nagle's Algorithm in DSR:

Since Nagle's Algorithm is used in the context of TCP, the version of Nagle's Algorithm does not need to concern with data acknowledgement. Our version of Nagle's algorithm can be defined as follow:

```
if there is new data to send
    if the (send buffer size + new data size) is >= MSS
    send complete MSS segment now
    else
    If the last sent time until now is < our threshold:
    Save the data to the send buffer.
    Schedule a send after our threshold
    else
    send data immediately
    Update the last sent time
    end if
    end if
    end if
```

6.5 Simulation Setup

We have made the following changes on top of the simulation setup described in section 3.2:

Simulation Duration	20 seconds
Start-up Time	10 seconds
Number of Nodes	10, 20, 30
Number source/sink pairs	4
Application data rate	~ 200 Kbits/s (400 64-byte packets per second)
Application data sending start time	Randomly between the 10 and 11 second

Table 2

6.6 Simulation Performance Evaluation: Throughput & Packet Delivery Ratio

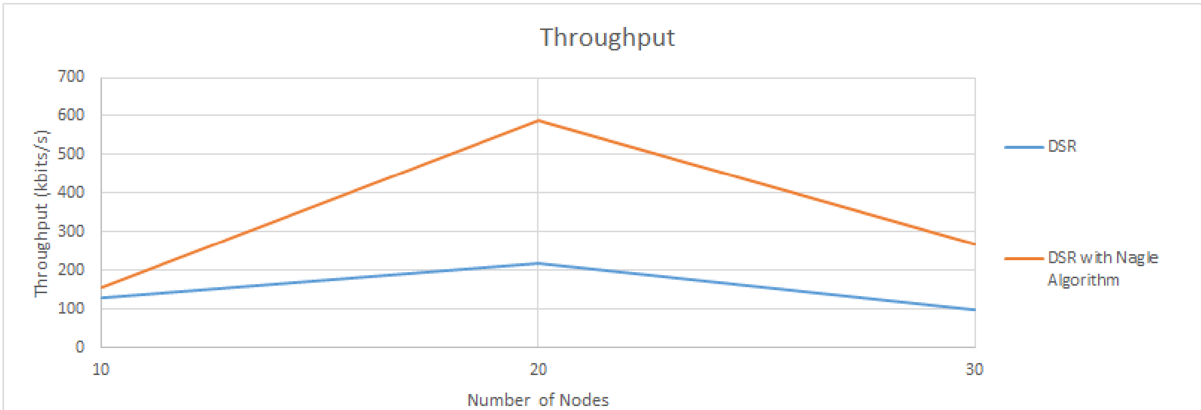

Figure 3: Throughput in Enhanced DSR

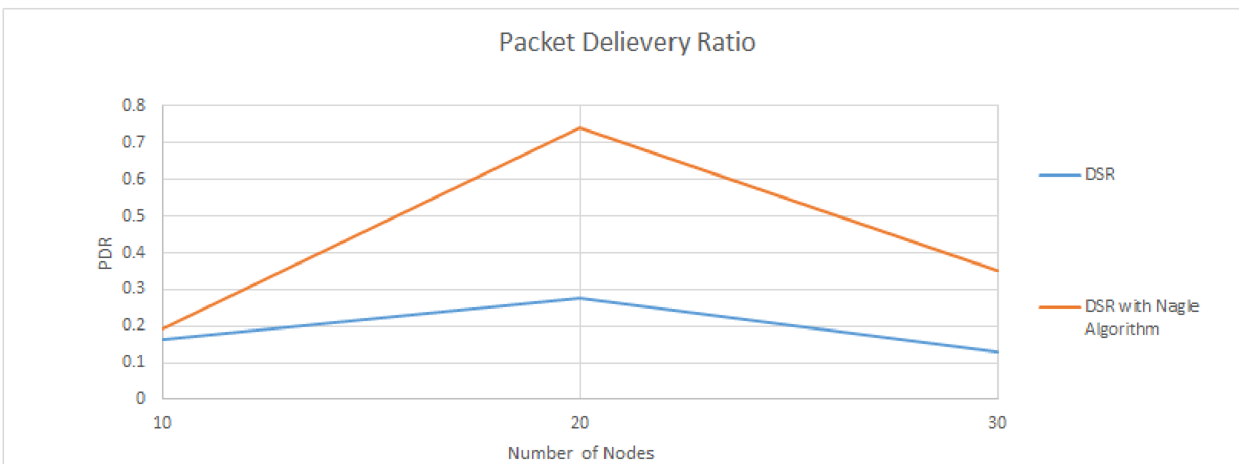

Figure 4: PDR in Enhanced DSR

From the simulation result, we plot the Throughput & DSR vs Number of nodes in both DSR and DSR with our added Nagle's Algorithm enhancement. The graphs show the following results:

1. Under the simulation setup, the added Nagle's Algorithm in DSR has better throughput and packet loss ratio than the regular DSR.
2. Under certain configurations (e.g., number of nodes = 20), the enhancement can be quite significant: approximately 2x.

7. Conclusion

We simulated mobile IoT using NS3 and implemented routing through AODV and DSR protocols. The performance for throughput was observed and noted after changing different simulation parameters. The performance evaluation revealed that the throughput decreases with the increase in the number of nodes, irrespective of the protocol. However, for the same number of nodes, DSR has a better throughput than AODV. The Packet Delivery Ratio also decreases as the number of nodes increases. We implemented an enhanced DSR algorithm using a variation of Nagle's algorithm. The following performance evaluation showed that has a better throughput and packet delivery ratio than the DSR without Nagle's algorithm under the high bandwidth with small packet size scenario. Also, the enhancement can be quite significant in case of certain configurations.

8. References

[1] Padmalaya Nayak and Pallavishree Sinha, “Analysis of Random Way Point and Random Walk Mobility Model for Reactive Routing Protocols for MANET Using NetSim Simulator ,” in *3rd International Conference on Artificial Intelligence, Modelling and Simulation (AIMS),* 2015, pp. 427 - 432.

[2] Nagle's algorithm, in *Wikipedia*, 31 October 2016, Retrieved November 27, 2016, from https://en.wikipedia.org/wiki/Nagle's_algorithm.

[3] Yufei Cheng, Egemen K. Çetinkaya, and James P. G. Sterbenz, “Dynamic source routing (DSR) protocol implementation in ns-3,” in *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques (SIMUTOOLS '12),* ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 2012, pp 367-374.