Assessment of DSACC and QPART Algorithms in Ad Hoc Networks

Seyed Hossein Hosseini Nazhad Ghazani
Department of Computer Engineering, Ahar Branch, Islamic Azad University, Ahar, Iran
Email: S.HosseiniNazhad@iau-ahar.ac.ir

ABSTRACT
The rapid advancement in wireless over wired has augmented the need for improving the Quality of Service (QoS) over such wireless links. However, the wireless ad hoc networks have too low bandwidth, and establishing a QoS in these networks is a difficult issue. So, support of quality of service in ad hoc networks is the topical issue among the network science researchers. In this research we are going to evaluate the performances of DSACC (Distributed Scheduling algorithm with Collision Control) and QPART (QoS protocol for Ad hoc Real Time Traffic) algorithms in different conditions. These two algorithms are able to support quality of service in ad hoc networks. It should be noted that we have used ns-2 simulator software to compare these two algorithms.

KEYWORDS: *Ad Hoc, Quality of Service, Scheduling, Distributed.*

1. INTRODUCTION

A mobile ad hoc network (MANET) is a collection of mobile nodes that can communicate with each other without using any fixed infrastructure. A multi hop ad hoc network is an ad hoc network in which the packets of a traffic flow are relayed by one or more intermediate nodes before they reach the destination. We can create ad hoc networks easily. So, regarding the ease of the installation of these networks, it is estimated that this type of networks will be used commonly. To support different types of real time applications, providing various qualities of service (QoS) guarantees for multi-hop flows is an important issue in wireless ad hoc networks. As we know Best-effort services cannot render services for this type of applications. The Best-effort service characterizes a service in which the network does not provide any patronages and guarantees that packets are delivered and a quality of service be done. However, real-time flows related to multimedia applications are more important than other flows and should be served specifically.

There are many algorithms for attention to QoS parameters in ad hoc networks [1, 2, 3, 4, 5]. Most of existent algorithms do not control the rate reception of flows. These algorithms do not note to this fact that accept of the new flows may decrease the bandwidth of the available nodes in the scope of the node transmission. Recently, other algorithms have been prepared to support of service differentiation in Ad hoc networks. Many of them specifically target

IEEE 802.11 [6]. For example, studies in [7, 8, 9, 10] propose to control the contention window sizes or the inter-frame spacing values to improve network throughput, while studies in [11, 12] propose priority-based scheduling to provide service differentiation. Most of these works utilizes a static mode such as different back-off mechanisms, different DIFS lengths, or different maximum frame lengths, based on the priority of the traffic to provide the service differentiation.

In continue and before the assessment of the efficiency of the two algorithms of QPART and DSACC, we will have a brief look at their functions.

2. QPART ALGORITHM

The QPART [13] algorithm is considered to be one of the best algorithms to establish quality of service in ad hoc wireless networks. This algorithm could support QoS with the least overload into the network. This algorithm groups the flows into three groups of flows sensitive to delay, sensitive to bandwidth, and best-effort services. Then, it uses the queue structure to control the back-off amount of flows. In each node, one queue for n number of best-effort flows and n queues of real-time flows crossing the node are created. The creation of these queues is locally carried out and there would be no overload for the network. Below some formulas used by this algorithm to control CW related to flows will be investigated.

- The QPART algorithm uses the following formula to control the CW of delay-sensitive flows :

$$CW^{(n+1)} = CW^{(n)} * (1 + a\frac{d/_m - D^{(n)}}{d/_m}) \quad (1)$$

Where the superscript n represents the n^{th} update iteration, D denotes the actual peak packet delaan at the node during a update period and α is a small p.sitive constant (α=0.1).

- The QPART algorithm uses the following formula to control the CW of bandwidth-sensitive flows :

$$CW^{(n+1)} = CW^{(n)} + \beta(q - Q^{(n)}) \quad (2)$$

Where q is a threshold value of the queue length that is smaller than the maximum capacity of the queue, Q represents the actual queue length and β is a positive constant ($\beta=1$).

- The QPART algorithm uses the following formula to control the CW of best effort flows :

$$CW^{(n+1)} = CW^{(n)} \times (1 + \gamma(f - F^{(n)})) \quad (3)$$

Where f is a *congestion threshold* for an idle channel time, F is the actual idle channel time and γ is a positive constant ($\gamma=0.1$).

QPART algorithm uses above formulas to regulate CW of flows and it tries to improve the parameters related to quality of service of the flows existing in the network with the same formulas. It should be noted that this algorithm uses the formula proposed in IEEE 802.11 to calculate the amount of back-off of each of the flows.

3. DSACC ALGORITHM

The problem of QPART algorithm is that to establish flows' quality of service, the status of the network has not been considered carefully. Each of the flows uses the formulas and related queues to calculate its CW and tries to gain a communicative medium and then send data packets. Meanwhile, due to the high acceptance rate of flows, the probability of the existence of overpopulation and chaos in some areas in the network is inevitable. In this case, the status of the network should be taken into consideration when we calculate the

amount back-off of a flow. In other words, when a node is trying to send data packets and faces number of collisions, this means that the network has heavy traffic. If any of the nodes faces this situation, it should be noted that the network is in busy status and consider this in calculating CW and choose a higher amount for the CW and avoid the quarrel to gain channels. This should continue until the channel is vacant and the collisions reduce. The goal of DSACC [14] algorithm is to present an intelligent framework to send flows intelligently in wireless ad hoc networks. Regarding the fact that using an ad hoc network is increasingly developing it is possible that the broadness and as a result the number of users who use these networks in the same environment is enhanced. As it is known, the management of all networks and specifically ad hoc network management become complicated by increasing the number of users and the quality of services posed decreases. In our proposed algorithm we take into consideration the status of the network to resist against this problem and control the existing flows of the network better. In addition to paying attention to the network's status in calculating CW, it is possible to use fixed wireless routers or slow moving wireless routers during path found. In DSACC, we have used the following formula to control the status of network besides using the formulas proposed by QPART algorithms.

$$Back\text{-}off = Rand[0,(2^{r}+R_{col})*CW_{\min}]*Slot\_Time \quad (4)$$

In above formula R_{col} shows the collision rate between the two successful frame transmissions of a station and r is a positive number.

The pseudo-code of Receive, Send and Back-off functions are as follows:

```
Receive Sensitive Packet ( P as packet)
{
   If (Create_Time(p)>ReT(p)) then
       Reject (P)
   Static Waiting_Time=0 ;
   Waiting_Time++;
}

Send Sensitive Packet ( P as packet)
{
  Create_Time(p) = Create_Time(p) + Waiting_Time(p);
}

Back-off Time
{
     Get minimum CW( CW_min ) from network layer.
     Calculate Back-off time according to:
Back-off = Rand[0,(2^r + R_col)*CW_min ]*Slot_Time
}
```

4. MODEL VALIDATION

In this section, we use the NS-2 program and evaluate the performance of DSACC and QPART algorithms in supporting QoS. NS (network simulator) is a name for series of discrete event network simulators. All of them are discrete-event network simulator, primarily used in research and teaching. NS is free software, publicly available under the GNU GPLv2 license for research, development, and use. In our simulation the AODV routing algorithm in a network with 11Mb bandwidth is used. The used parameters are shown in Table 1:

Table.1. Parameters of simulation

a	0.1	f	1 ms	q	5 Packets
β	١	r	0.1	CW update Interval	0.1

To prove the packet scheduling capability in DSACC algorithm, we have supposed the network with 1000 m * 1000 m size,

and evaluated their operations in multi-hop status. In this simulation there are 8 flows for each delay, bandwidth sensitive and best effort flows (24 flows). The hop counts of flows are 1 to 7. The delay sensitive flows should reach to destination node within 100 ms. These flows generate 50 packets per second, that the size of each packet is 512 bytes. Each bandwidth sensitive flow generates 50 packets per second, that the size of each packet is 512 bytes. The size of best effort packets is 512 bytes. The average delay of delay-sensitive flows has shown in figure 1. The DSACC and QPART algorithms maintain the delay of delay-sensitive flows below of their requirements (100 ms). It means that both of them could support quality of service in this situation. But as it can be seen in the figure, due to the use of fixed nodes and also considering the present status of the network, DSACC algorithm has had a better function than QPART algorithm and the reached QoS by the DSACC is better than reached QoS by the QPART. It seems that the DSACC algorithm is able to manage networks with a lot of nodes. In our next experiment, we will increase the number of nodes and will study the performances of the algorithms.

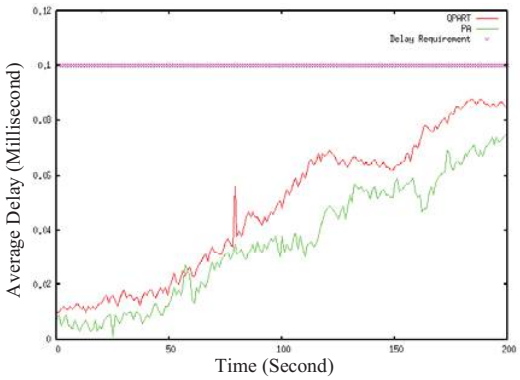

Fig.1. Average delay of delay-sensitive flows

In the following, we evaluated the behavior of algorithms in a network in which there are many nodes in this network. We considered the size of the network to be 2000 m*2000 m in which 180 nodes were scattered randomly in the network environment that 10 nodes of them have moved slowly. This network entails both real-time and best-effort flows. We consider the number of real-time flows to be 32 and best-effort flows 50. The number of real-time flows in this test is double related to previous test. Also the number of all flows is fourfold compared to previous evaluation. Randomly some nodes are chosen and then start communication with each other by sending some flows. Regarding that our aim of this experiment is to study the behavior of the algorithms in networks having a great deal of nodes, we took into consideration the size and production rate of real-time and non- real-time packets as the previous test. In the first 50 seconds of the test, 8 real-time flows were transmitted among the nodes of the network. As it can be seen in figure 2, both algorithms mentioned send their flows with a little less amount of delay compared to the needed delay. The rest of real-time flows are sent in 50^{th} second. As you can see, QPART algorithm loses its management and control over the flows in the network and the delay amount of the flows increases more than what is needed. In other words, this algorithm cannot realize the service quality of the flows in this situation. However, the DSACC algorithm by considering the status of the network and taking advantage of the less moving nodes could manage the acceptance rate and realizes the needed quality of service of flows. In 80^{th} second 16 flows of real-time flows have finished, then, QPART

algorithm gains the control of itself gradually on the flows present in the network and realizes their service quality. But as it can be observed in the figure, the time needed to return to the usual status in QPART algorithm is more than the DSACC algorithm. In other words the DSACC algorithm could adapt itself with more situations, so, we could say this is quick, lightweight and better than QPART.

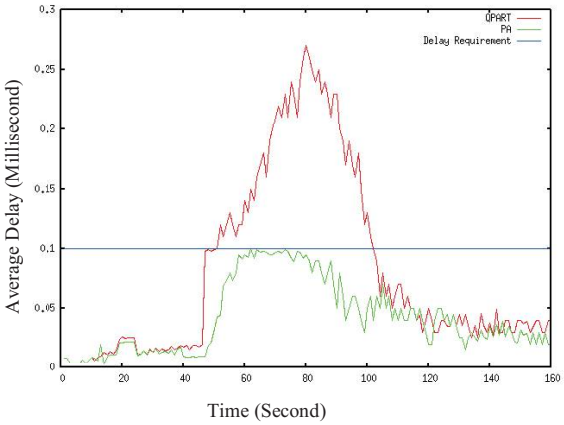

Fig.2. Average delay of real-time flows

5. CONCLUSIONS

We introduced some algorithms that could support the QoS in wireless ad hoc networks. Then, we compared two algorithms entitled DSACC (proposed by the authors of the present research) and QPART algorithm. Both algorithms support the QoS in ad hoc networks. The difference between them is that in calculating CW of flows, the QPART algorithm considers only the present status of the flows. But the DSACC consider to present status of the flows and network status. Thus, when the network is busy and the resources of the network are occupied, the DSACC algorithm performed better that QPART. By considering the result of this paper, we could claim that the DSACC algorithm is more intelligent than QPART algorithm.

REFERENCES

[1] S. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell. "INSIGNIA: An IP-Based Quality of Service Framework for Mobile Ad Hoc Networks" // Journal of Parallel and Distributed Computing, Special issue on Wireless and Mobile Computing and Communications, 2000, vol.60, pp. 374-406.

[2] G.-S. Ahn, A. Campbell, A. Veres, and L.-H. Sun. "Supporting Service Differentiation for Real-Time and Best-Effort Traffic in Stateless Wireless Ad Hoc Networks (SWAN)" // *IEEE Transactions on Mobile Computing*, 2002, vol.1, pp.192–207.

[3] S. Chen and K. Nahrstedt. "Distributed Quality-of-Service Routing in Ad-Hoc Networks" // *IEEE Journal of Selected Areas in Communications*, 1999, vol.17, pp. 1454-1465.

[4] T. Chen, M. Gerla, and J. Tsai. "QoS Routing Performance in a Multi-hop, Wireless Network" / In *Proceedings of the IEEE ICUPC'97*, San Diego, 1997, pp. 557-61.

[5] P. Sinha, R. Sivakumar, and V. Bharghavan. CEDAR: "A Core-Extraction Distributed Ad Hoc Routing Algorithm" / In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, New York, NY, 1999, pp. 202-209.

[6] http://www.csd.uoc.gr/~hy439/reading/802.11-1999.pdf.

[7] I. Ada and C. Castelluccia. "Differentiation Mechanisms for IEEE 802.11" / In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, Alaska, 2001, pp. 209-218.

[8] F. Cal´i, M. Conti, and E. Gregori. "Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit" // *IEEE/ACM Transactions on Networking*, 2000, vol.8, pp. 785-799.

[9] T. S. Ho and K. C. Chen. "Performance Evaluation and Enhancement of CSMA/CA MAC Protocol for 802.11

Wireless LANs" / In *Proceedings of the IEEE PIMRC*, Taiwan, 1996, pp. 535-547.

[10] H. Kim and J. C. Hou. "Improving Protocol Capacity with Model-based Frame Scheduling in IEEE 802.11-operated WLANs" / In *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (Mobi COM'03)*, San Diego, 2003, pp. 190-204.

[11] R. Rozovsky and P. Kumar. " A MAC Protocol for Ad Hoc Networks" / In Proceedings of the 2^{nd}-ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01), Long Beach, CA, 2001, pp. 67-75.

[12] A. Veres, A. T. Campbell, M. Barry, and L.-H. Sun. "Supporting Service Differentiation in Wireless Packet Networks Using Control" // *IEEE Journal of Selected Areas in Communications*, 2001, vol.19, pp. 2081-2093.

[13] Y. Yang, "Distributed QoS guarantees for real-time traffic in ad hoc networks". Sensor and Ad Hoc Communications and Networks, 2004, USA, pp. 118-127.

[14] S.H. Hosseini Nazhad, R.M.Alguliev, "Light Weight Distributed QoS Adapter in Large-Scale Ad hoc Networks" // Journal of American Science, 2011, vol.7, pp. 28-1251.