A new Call Admission Control algorithm for IEEE802.16 networks

Conference Paper · July 2010

DOI: 10.1109/ICCDA.2010.5541082 · Source: IEEE Xplore

CITATIONS
4

READS
50

4 authors, including:

Samaneh Rashidi
University of Münster
13 PUBLICATIONS **43** CITATIONS
SEE PROFILE

Hassan Taheri
Amirkabir University of Technology
92 PUBLICATIONS **566** CITATIONS
SEE PROFILE

Mohammad Fathi
University of Kurdistan
38 PUBLICATIONS **391** CITATIONS
SEE PROFILE

A New Call Admission Control Algorithm for IEEE802.16 Networks

Samaneh Rashidi.B
Department of Computer Engineering
Azad university, branch of Arak, Iran
Samane.rashidi@gmail.com

Hasan Taheri
Department of Electrical Engineering
Amirkabir university
Tehran, Iran
htaheri@aut.ac.ir

Masoud Sabaghi
Department of Electrical Engineering
Azad University branch of Ashteyan, Iran
msabaghi@gmail.com

Mohammad Fathi
Department of Electrical Engineering
Amirkabir university
Iran, Tehran
mfathi@aut.ac.ir

Abstract— In this paper, we present a new Call Admission Control (CAC) for IEEE 802.16 broadband wireless networks considering Quality of Service (QoS) constraint. With the assumption that there is a scheduling algorithm to allocate total bandwidths, our proposed CAC algorithm accepts new request for some slot times temporary. In this algorithm, during certain time slots, departure rate for ongoing connections will be decreased and for new connection will increase. After this period, if QoS for ongoing connections and new request is satisfied, new request will be accepted permanently and otherwise it will be rejected. We study the results in two phases, first we show that our algorithm works well with some routine scheduling algorithms and second we show the performance of our algorithm in comparison to some other CAC algorithms. We simulate our algorithm with Matlab tools, and the results show proposed algorithm is quite efficient.

Keywords-call admission control; scheduling; IEEE 802.16

I.INTRODUCTION

IEEE 802.16 standard has been proposed to provide high speed broadband wireless connectivity with QoS guarantee for connections. This standard has a base station (BS) and one or more subscriber station (SS). BS provides a gateway between SSs and Internet, and SSs provide access to BS for local users.

Providing QoS is very important in IEEE802.16, and because of the different traffic classes in the network, IEEE 802.16 introduces four types of services: UGS for real time data stream with fixed size data packet on a periodic basis (like VOIP). rtPS for real time data stream with variable sized packets on a periodic basis (like video MPEG), nrtPS for non real time polling service (like FTP), BE for data stream without any QoS guarantee (like HTTP). This standard determine signaling for providing QoS, but some main blocks like call admission control (CAC) and scheduling was left for researchers.

The aim of CAC is to achieve a certain QoS for ongoing and accepted requests with limiting connections in the system. So, when there is a new request for system, CAC must evaluate system QoS, departure rate for non real-time connections and delay for real-time connections, and make a decision to accept or reject the request. If new request is accepted, QoS must be guarantee for new connection as well [1].

There is some research in CAC, and some algorithms were presented. A good CAC algorithm must have four traits: 1) minimize handoff failed; 2) maximize accepted request; 3) provide bandwidth efficiency and 4) reduce delay [2].

In [1], a CAC algorithm based on a conservative approach was presented called conservative and quality of service (CAQoS). In [2], a joint strategy of bandwidth allocation and admission control for elastic users competing for a downlink data channel in cellular networks was considered. In [3], a multiclass CAC based on reservation-pool for handoff requests was presented. In [4], a dynamic CAC algorithm for UGS connections was proposed.

Most of these algorithms satisfy one or two parameters mentioned for a good CAC algorithm. The aim of this paper is to introduce a new CAC algorithm that satisfy all four parameters, provide suitable QoS for service classes (reduced delay for real time service classes and increased departure rate for non-real time service classes) and reduce blocking for new requests and hand off connections.

To achieve this goal, we make decision during T slot time. With this strategy, we make a better decision than other CAC algorithms and there are fewer faults.

Usually a CAC algorithm is evaluated with a scheduling algorithm, so we evaluate our proposed algorithm with three scheduling algorithms and will show that it works well with them. Also, we will compare our proposed algorithm with three other CAC algorithms and will prove that our algorithm out performs them.

The rest of this article is organized as follow. In section 2 we present CAC algorithms, at first subsection we introduce our proposed algorithm, then we have an overview of some scheduling algorithms at second subsection and we describe different CAC algorithms at third subsection. In section 3, we provide the result of simulation of proposed CAC algorithm with different scheduling algorithms at first subsection and compare different CAC algorithms at the second subsection.

II.THE CAC ALGORITHMS

In this section, we first introduce proposed CAC algorithm in the first subsection, and then, in the second subsection, we introduce scheduling algorithms that we use in our simulations. In third subsection, we introduce some other CAC algorithms.

A. PROPOSED ALGORITHM

In this subsection, we introduce our proposed algorithm for making decision to accept or reject the requests. Table I shows parameters that are used in the pseudo code.

In our algorithm, handoff and UGS connections have highest priority in the system. Therefore, in each slot the algorithm calculates the amount of bandwidth required for handoff request; if it is less than the available bandwidth, it will be accepted and otherwise it will be rejected. Then, we survey UGS requests, If there is enough bandwidth available and ongoing connection are in a good condition, the request will be accepted and otherwise it will be rejected.

We assume that the remaining bandwidth ($C_{total} - C_{handoff} - C_{UGS}$) is allocated to the ongoing connections in the rtPS, nrtPS and BE class of services. Our CAC algorithm makes decision as follow. For response to the new requests, we study rtPS, nrtPS and BE classes respectively. Let us assume that NC is the new connection request. Making decision to accept or reject the new request is shown in the algorithm 1. We assume that departure rate at the beginning is more than the threshold amount. This algorithm tries to decrease departure rate of other connections and increase departure rate for the new connection. If departure rate for ongoing connections and the new request stay above the threshold, it will be accepted and otherwise it will be rejected. In this algorithm $\alpha_1, \alpha_2, \alpha_3$ are constant amounts that must be small, until the algorithm synchronies during T slot time.

B. SCHEDULING ALGORITHMS

In this subsection we review the scheduling algorithms that CAC algorithm use.

1) Round Robin (RR) algorithm: This algorithm is a simple algorithm that divides bandwidth equally between queues.

ALGORITHM I. PROPOSED CAC ALGORITHM

1. Upon arrival of any new connection NC
2. for t=1 to T
3. $\Delta\mu^t_{rtps} = \alpha_1(\mu^{TH}_{rtps} - \mu^t_{rtps})$
4. $\Delta\mu^t_{nrtps} = \alpha_2(\mu^{TH}_{nrtps} - \mu^t_{nrtps})$
5. $\Delta\mu^t_{BE} = \alpha_3(\mu^{TH}_{BE} - \mu^t_{rtps})$
6. $\mu^{t+1}_{rtps} = \mu^t_{rtps} - \Delta\mu^t_{rtps}$
7. $\mu^{t+1}_{nrtps} = \mu^t_{nrtps} - \Delta\mu^t_{nrtps}$
8. $\mu^{t+1}_{BE} = \mu^t_{BE} - \Delta\mu^t_{BE}$
9. $\mu^{t+1}_{NC} = \mu^t_{NC} + \Delta\mu^t_{rtps} + \Delta\mu^t_{nrtps} + \Delta\mu^t_{BE}$
10. end
11. if ($\mu^{T+1}_{rtps} > \mu^{TH}_{rtps}$ and $\mu^{T+1}_{nrtps} > \mu^{TH}_{nrtps}$ and $\mu^{T+1}_{BE} > \mu^{TH}_{BE}$ and $\mu^{T+1}_{NC} > \mu^{TH}_{NC}$)
12. admit the arriving connection *NC*.
13. else
14. block *NC*.
15. end

2) Modified Largest Weighted Delay (MLWDF) algorithm: This algorithm is one of the best algorithms that exist. MLWDF guarantee delay and throughput. In this algorithm, at each time slot, queue j will be served as follows

$$p_j(t) = \gamma_j w_j(t) r_j(t) \quad (1)$$

Where $p_j(t)$ is the bandwidth allocated to queue j, $w_j(t)$ is the head-of-the-line packet delay for queue j, $r_j(t)$ is the channel capacity with respect to flow j, and γ_j are arbitrary positive constants. For better result we can replace $w_j(t)$ with the length of the queue for some services [5].

3) EDF + WFQ + FIFO hybrid algorithm: This algorithm is a combination of Earliest Deadline First (EDF) for rtPS class of services, Waited Fair Queue (WFQ) for nrtPS class of services and FIFO for BE class of services.

In EDF algorithm, packets are sort according to their deadline. So, packets with earliest deadline will be given service sooner than others. In WFQ algorithm each connection has a weight, and packets are sort according to their weight and packets with highest weight will be given service sooner than others [6].

C. CAC ALGORITHMS

We chose MLWDF as a scheduling algorithm for the following CAC algorithms.

1) ALGORITHM 1

This algorithm is based on token bucket and acts as follows:

Step 1: Compute remaining bandwidths according to:

$$B_{remain} = C_{total} - \mu^t_{UGS} - \mu^t_{rtps} - \mu^t_{nrtps} - \mu^t_{BE} \quad (2)$$

Step2: Compares remaining bandwidths with new request. If there are enough bandwidths, the new request will be accepted, else follow to step 3.

Step 3: if lower class flows (BE and nrtPS) have bandwidth more than their threshold ($\mu^{TH}_{BE}, \mu^{TH}_{nrtps}$), allocate less bandwidth to them, otherwise follow to step 4.

Step4: if higher class flow has bandwidth more than its threshold (μ^{TH}_{rtps}), decrease departure rate of it to r_rtPS, else go to step 5.

Step 5: reject the new request [7].

2) ALGORITHM 2

This algorithm is proposed in [6]. The algorithm assumes that:

- There are N rtPS connections with r_i token bucket (packet per slot) and b_i token bucket size (packet) and delay (d_i)

TABLE I. PARAMETER USED IN THE PSEUDO CODE

Parameter	Description	Parameter	Description
C_{total}	Total available bandwidth	μ_{rtps}^{TH}	rtPS delay threshold
$C_{handoff}$	Request bandwidth for hand off connection	μ_{BE}^{t}	Departure rate for BE queue at slot t
C_{UGS}	Request bandwidth for UGS service	μ_{nrtps}^{TH}	nrtPS departure rate threshold
μ_{rtps}^{t}	Departure rate for rtPS queue at slot t	μ_{BE}^{TH}	BE departure rate threshold
μ_{nrtps}^{t}	Departure rate for nrtPS queue at slot t	α_i	Positive constant

- $C_{rtPS} = \sum r_i$
- $m_i = \frac{d_i}{f}$, that m_i must be integer and f is session duration
- To guarantee delay in connection i with rtPS flow, they use

$$b_i \le [(m_i - 1)(1 + \frac{C_{NRT}}{\mu_{rtps}^{t}}) - 1] r_i f \qquad (3)$$

And the algorithm policy is as follow

Step1: for rtPS connection:

a) If there is enough bandwidth ($B_{remaind} \le C_{total} - \mu_{UGS}^{t} - \mu_{rtps}^{t}$), accept new request, else reject it.
b) Check delay guarantee (equation 2), if it satisfies guarantee delay for new connection and accept it, else reject it.
c) Check delay for ongoing connections (equation 2). If it does not satisfy for even one connection, reject new request.
d) Update rtPS connection.

Step 2: for nrtPS connection, if there is enough bandwidth ($B_{remaind} = C_{total} - \mu_{UGS}^{t} - \mu_{rtps}^{t} - \mu_{nrtps}^{t}$) accept it and update nrtPS connections.

Step 3: accept BE request without any guarantee.

3) ALGORITHM 3

In [4], the author focuses on nrtPS service. If there is enough bandwidth available (there is few connections in the system), nrtPS flow could have high departure rate. When amount of request for connecting to the system increases, it decreases departure rate for ongoing nrtPS flows until the amount of UGS, rtPS and nrtPS connections increase in the system. They call this action, degradation.

This algorithm acts as follow:

Step 1: when there is new rtPS request, if current available bandwidth is enough for ongoing connections and new request, accept new request, otherwise decrease departure rate for nrtPS connections step by step. If degradation receives to maximum amount and there is not enough bandwidth, reject the new request.

Step 2: when there is a new nrtPS request, if there is enough bandwidth accept it, else decrease departure rate for ongoing nrtPS connections until it can be accepted. If there is not enough bandwidth, the new request will be rejected.

Step 3: when there is new BE request, if another service do not need bandwidth, BE request could have bandwidth.

III.SIMULATION

We show the results of simulation in two subsections. In subsection one we show the results of simulation for our proposed CAC algorithm with three scheduling algorithms that we introduced. In the second subsection, we show the results of simulation for comparing our algorithm with other CAC algorithms that we mentioned.

We consider a scenario with 15 requests, 5 requests for each service. After time slot 200, new requests reach to the system one by one after 100 time slot. Thus rtPS, nrtPS and BE requests enter to the system respectively.

A. PROPOSED ALGORITHM WITH DIFFERENT SCHEDULING ALGORITHM

Figures 1-6 show throughput and delay for mentioned algorithms in section 2, subsection 2. As you can see, departure rate for BE and nrtPS remain higher than the threshold amount ($\mu_{rtps}^{TH} = 200, \mu_{BE}^{TH} = 400$) and delay for rtPS remain less than rtPS threshold ($\mu_{rtps}^{TH} = 8$), and also in all of them, all new requests are accepted. The time of request arrivals are marked, rtPS requests are marked with r, nrtPS requests are marked with nr and BE requests are marked with BE.

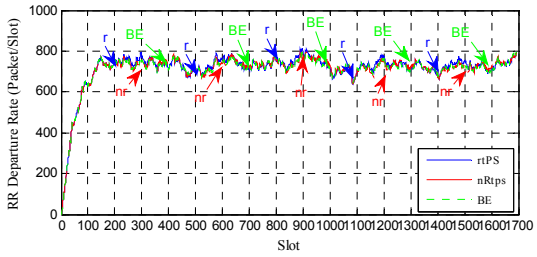

Figure 1. Departure rate with RR scheduling algorithm

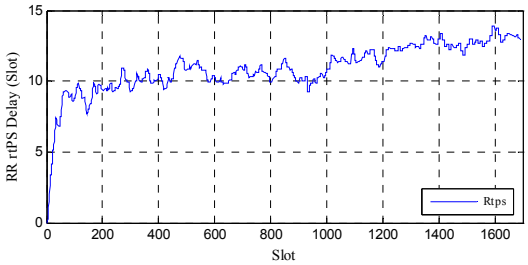

Figure 2. Delay for rttPS flows with RR scheduling algorithm

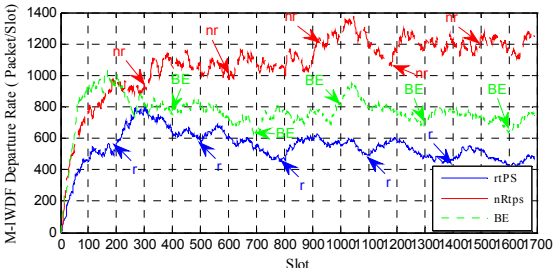

Figure 3. Departure rate with MLWDF scheduling algorithm

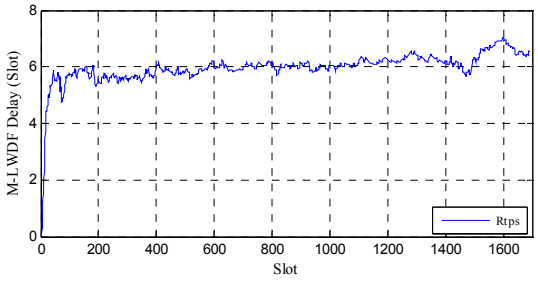

Figure 4. Delay for rttPS flows with MLWDF scheduling algorithm

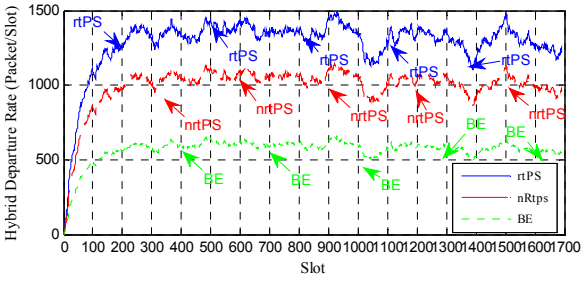

Figure 5. Departure rate with hybrid scheduling algorithm

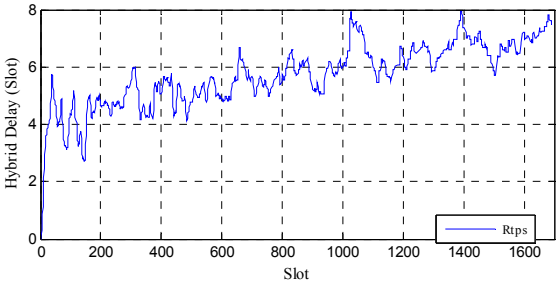

Figure 6. Delay for rttPS flows with hybrid scheduling algorithm

B. *COMPARING CAC ALGORITHMS*

To compare these CAC algorithms, we chose MLWDF scheduling algorithm.

Figures 7-9 show the result of departure rate for nrtPS and BE with different CAC algorithms and variable bit rate, figure 9 shows delay for rtPS algorithm.

The mentioned CAC algorithms in section 3 made decision on admitting or blocking new request as soon as it has been arrived, whilst our proposed algorithm take some times (T slot) to make this decision. (Figure 7 and Figure 8)

Some parameters like arrival traffic and channel state information in a wireless network are non deterministic, i.e. vary over time. So making a decision at the moment might not be the right one, while our algorithm makes decision during a time period, so it has more comprehensive information about the system to make a better decision.

Another point that is necessary to consider, is that admitting a new connection and providing bandwidth for it at just one moment causes the allocated rate for existing connections to vary abruptly. This issue causes instability to the network that needs more signaling overhead to stabilize the network. In the proposed CAC, we vary the allocated rates gradually over time thus avoiding rate oscillations.

Table II list amounts of accepted requests in the system and you can see our proposed algorithm has more acceptances with QoS guarantee for all services.

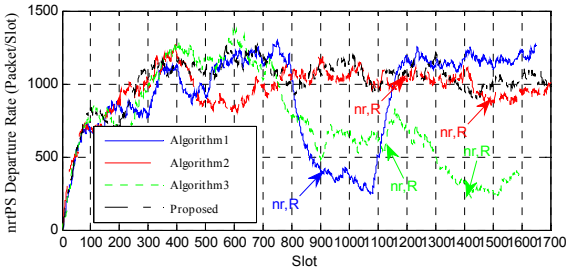

Figure 7. Departure rate for nrtPS with different CAC algorithm

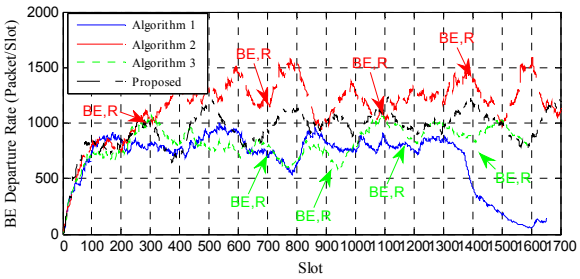

Figure 8. Departure rate for BE with different CAC algorithms

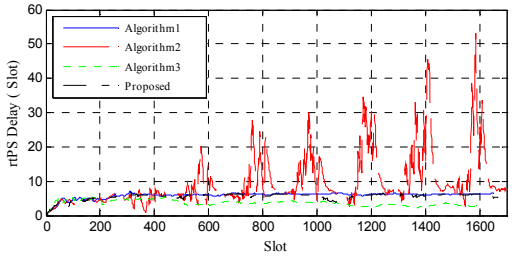

Figure 9. Delay for rtPS with different CAC algorithms

TABLE II. AMOUNT OF ACCEPT REQUEST IN DIFFERENT CAC ALGORITHMS

Algorithms	rtPs	nrtPS	BE
Algorithm1	4 request	3 request	3 request
Algorithm2	0 request	3 request	0 request
Algorithm3	4 request	4 request	1 request
Proposed	5request	5request	5 request

IV. CONCLUSION

In this paper, we introduced a new CAC algorithm, and we show that our algorithm throughput is good with different scheduling algorithms, and we showed that our algorithm works very well in the comparison to other CAC algorithms. It has less rejects for new requests and provide QoS for ongoing connections and the new connection. To achieve this goal, we use temporary accept policy at the system, then we evaluate system performance and make decision to accept or reject this request.

REFRENCES

[1] Pla, J.Virtamo, J.Martinez-Bauset, "Optimal robust policies for bandwidth allocation and admission control in wireless network," *Computer Networks*, vol. 52, no. 17, pp. 3258-3272, 2008.

[2] Y. C. Yee, K. N. Choong, L.Y.Low, S.W.Tan, S.F.Chien, "A conservative approach to adaptive call admission control for QOS provisioning in multimedia wireless networks," *Computer Communications,* vol. 30, no. 2, pp 249-260, 2007.

[3] F. Hu, N. K. Sharma, "multimedia call admission control in mobile network: a dynamical reservation-pool approach", *Computer Networks vol. 43* , pp 263-288, 2003.

[4] H. Wang, B. He, D. P. Agrawal, "Above packet level admission control and bandwidth allocation for IEEE 802.16 wireless MAN," *Simulation, Modeling, practice and Theory,* vo. *15*, pp. 366-382, 2007.

[5] M.Andrews, K.Kumaran, K.Ramanan, A. Stolyar, P.Whiting, "Providing Quality of Service over a Shared Wireless Link," *IEEE Communications Magazine* , vol. 32, no. 9, pp 150-154, 2001.

[6] K.Wongthavarawat, A.Ganaz, "Packet Scheduling for QoS Support in IEEE802.16 Broadband Wireless Access Systems," *International Journal of Communication Systems*, pp. 81-96, 2003.

[7] T. Tzu, H. J. Chi, Y. W. Chuang, "CAC and packet scheduling using token bucket for IEEE802.16 networks", *Journal of Communication, Vol. 1, No.2*, pp 30-37, 2006.