Communication Networks

AC ALGORITHMS IN AQUILA QOS IP NETWORK

CHRISTOF BRANDAUER
Salzburg Research, Jakob Haringer Strasse 5/III, 5020 Salzburg, Austria
brandauer@salzburgresearch.at

WOJCIECH BURAKOWSKI, MAREK DABROWSKI, HALINA TARASIUK
Warsaw University of Technology, ul. Nowowiejska 15/19, 00-665 Warsaw, Poland
{wojtek, mdabrow5, halina}@tele.pw.edu.pl

BERTHOLD KOCH
PMC, Johann-Keller-Weg 8a, D-86919 Utting, Germany
bert.koch@t-online.de

Abstract. This paper presents the admission control (AC) algorithms that are implemented in the prototype multi-service AQUILA QoS IP network [4]. The discussed algorithms were developed for regulating traffic submitted to network services (NS) dedicated for handling: (1) real-time streaming traffic of variable bit rate (VBR) type, and (2) elastic traffic, produced by greedy TCP sources. For the former NS, named Premium VBR, the measurement based AC (MBAC) scheme is proposed. The applied approach employs the well known Hoeffding bound formula [2] for calculating required link capacity to assure packet loss ratio at a predefined level. Moreover, this method is supported by the declarations about the peak bit rate as well as by the measurements of mean bit rate on aggregate flow level. For the latter NS, named Premium Multimedia, two alternative AC algorithms are investigated, both designed for assuring requested TCP throughput. While the first of them follows the Token Bucket Marking (TBM) concept [12], the second one adjusts the advertised TCP window size to enable an ideal TCP behaviour (i.e. lossless packet transfer). The simulation results are included for illustrating the advantages of the discussed algorithms.

1 INTRODUCTION

The DiffServ architecture [1] is extensively investigated as a promising solution for providing Quality of Service (QoS) in IP-based networks. One of the approaches for such a network is the network concept developed inside the AQUILA IST European project [3]. It defines an enhancement of a generic DiffServ architecture by adding new functionalities, among others for admission control and resource management as well as by defining a new set of network services.

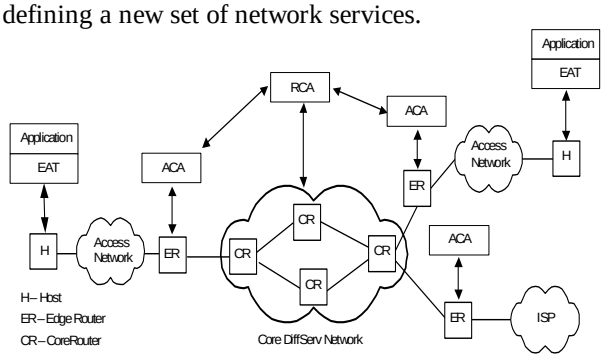

Figure 1: Architecture of AQUILA QoS IP network.

Figure 1 depicts the AQUILA architecture. The traffic submitted to the core network is controlled by the AC agents (ACA) associated with edge routers (ER). The call handling scenario is the following. For establishing connection, a user is supported by the end-user application toolkit (EAT). The EAT sends a request containing traffic contract parameters to the ACA. The ACAs (on ingress and egress sides) can admit this call when available resources (bandwidth, buffer size) are sufficient. The Resource Control Agent (RCA), situated on the top of the network, follows the network load and updates the volume of allocated bandwidth for particular ACAs.

For now, four types of packet flows requiring QoS guarantees have been recognized as typical in the Internet. They are the following: (1) streaming constant bit rate (e.g. VoIP), (2) streaming variable bit rate (e.g. video applications), (3) elastic, produced by greedy TCP or TCP-like sources (e.g. FTP), and (4) elastic, emitted by non-greedy TCP sources (e.g. home banking). In this spirit, four QoS network services (NS) have been defined and implemented in AQUILA: Premium CBR (PCBR) for traffic (1), Premium VBR (PVBR) for traffic (2),

Premium Multimedia (PMM) for traffic (3), and Premium Mission Critical (PMC) for traffic (4). Each network service is optimised for handling its specific type of packet flows with guaranteed QoS. For this purpose, each network service is supported by appropriate traffic handling mechanisms, including admission control. In addition, Standard Service for best effort traffic is also provided.

In this paper we present the AC algorithms associated with PVBR and PMM services. For PVBR, the measurement based AC (MBAC) scheme is proposed. The applied approach employs the well known Hoeffding bound formula [2] for calculating the required link capacity to assure a packet loss ratio at a predefined level. Moreover, this method is supported by the declarations about the peak bit rate as well as by the measurements of mean bit rate on aggregate flow level. For PMM, two alternative AC algorithms are investigated, both designed for assuring requested TCP throughput. While the first of them follows the Token Bucket Marking (TBM) concept [12], the second one adjusts the advertised TCP window size to enable an ideal TCP behaviour (i.e. lossless packet transfer).

The structure of the paper is the following. Section 2 gives an overview (including numerical results) of the applied MBAC method for PVBR service. The AC algorithms for PMM service are described in section 3. The numerical results illustrating their effectiveness are also included. Finally, section 4 concludes the paper.

2 MEASUREMENT BASED AC FOR PVBR SERVICE

The PVBR service is designed for handling streaming VBR traffic with target QoS objectives defined as low packet loss ratio and low packet delay. For example, a candidate application for using this service is real-time video, like video-conferencing or live streaming video. For meeting the above QoS requirements, the traffic submitted inside PVBR should be served with relatively high priority and low delay. As a consequence, this leads to handling PVBR traffic in a separate way in the routers by submitting it to a dedicated buffer of limited size. In addition, relatively high priority for such a queue should be assigned in the scheduler governing access to the outgoing link.

One can find some similarities between the PVBR service in the AQUILA QoS IP network and the rt-VBR service in ATM. The recognised solution for admission control in the case of rt-VBR is to use the Rate Envelope Multiplexing (REM) scheme and to apply a Declaration Based AC (DBAC) approach to control the submitted traffic. Notice that DBAC assumes using the declarations referring to the peak (PR) and sustained (SR) bit rates for calculating the required link capacity, expressed in the form of effective bandwidth. A list of methods for calculating effective bandwidth can be found e.g. in [2]. However, this approach gives satisfactory results only when the declared value of SR is close to the mean rate of the flow, m. The DBAC method based on the Lindberger formula [2] for calculating effective bandwidth was tested at the beginning phase of the AQUILA project [3]. Unfortunately, it turned out that it is rather difficult for a user to precisely specify a priori the proper value of SR close to m. Remark, that this value is policed and incorrect declarations could cause undesirable packet dropping. In addition, even if it is possible to make correct declarations in the case of, e.g. stored video, it is not possible in the case of, e.g. live video. As a consequence, to avoid possible packet losses, a user rather over-declares the SR comparing to m, which usually leads to a non-effective bandwidth utilisation. This motivated the replacement of the DBAC by an MBAC approach. By applying the MBAC algorithm, one can expect the following profits:

- To simplify the traffic declarations; usually it is difficult for the user to specify accurate parameters other than the PR;
- To achieve better network utilization (finally, leading to more accepted flows) by taking into account in the AC the real submitted traffic to the network, which is usually lower than the declared;
- To capture the stochastic nature of the user traffic more accurately than it is possible with deterministic parameters assumed by the DBAC methods.

The MBAC method selected for the PVBR service is based on the Hoeffding bound [6, 15]. This method requires only the measurement of the mean bit rate on aggregate flow level, so it was relatively easy to implement. A special measurement module was implemented as a part of the ACA agent. The number of transferred bytes on the outgoing link was collected by polling the router in predefined fixed periods. The method implemented for estimating the mean bit rate was based on the sliding window algorithm. The detailed discussion of MBAC implementation in AQUILA trial network can be found in [18].

Consider a system of capacity C with N_{PVBR} running flows. Each flow is characterized by declared peak bit rate PR_i (i=1,…,N_{PVBR}). For this system we measure the aggregate mean bit rate, say M. The applied MBAC approach assumes, that new flow with declared peak rate PR_{new} is admitted only if:

$$PR_{new} + M + \sqrt{\frac{-\ln P_{loss}}{2}\left(\sum_{i=1}^{N_{PVBR}} PR_i^2 + PR_{new}^2\right)} \le C\,, \quad (1)$$

where P_{loss} is the target packet loss probability, e.g. 10^{-4}.

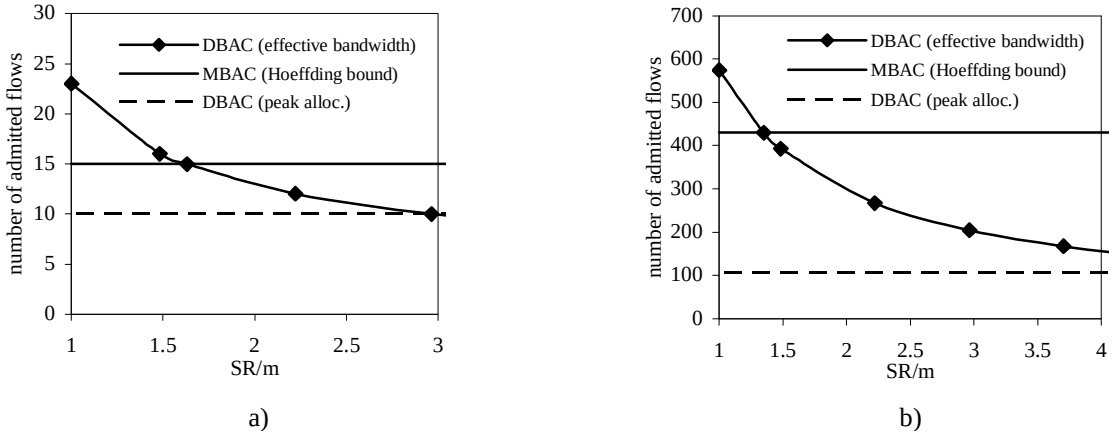

Figure 2: Comparison of efficiency of MBAC and DBAC schemes on a link with capacity a) 10Mbps and b) 100Mbps.

Figure 2 shows a comparison between the MBAC and DBAC methods. It shows the number of admitted flows as a function of an over-declaration factor defined as ratio of SR to m. These characteristics were collected assuming that the PR value is declared correctly. Two DBAC methods were considered: one based on effective bandwidth estimation using the Lindberger formula and the second assuming the peak rate allocation [2]. The tested traffic patterns are MPEG4 video traces with the peak and mean rates equal to 0.94 and 0.135 Mbps [16].

As it was expected, MBAC outperforms DBAC only when the SR is essentially higher than m (say more than 1.5 times). Better results could be obtained by using MBAC method with per flow measurements but this is hard in the AQUILA architecture, which assumes AC at the network ingress and egress. While it can easily be done on the ingress side, per-flow measurements on the egress side require classification of flows leaving from the network and this is not supported in the AQUILA architecture.

3 AC FOR PMM SERVICE

The PMM service is aimed at the efficient handling of greedy TCP flows requiring guarantees with respect to the target requested rate (R_{req}). The example PMM application is a large file transfer (FTP), where a user wants to transfer a file within a satisfactory time interval.

The TCP traffic control algorithms employ a sliding window based mechanism. Basically, TCP transmission is controlled through a send window which limits the maximum amount of data that may be outstanding, i.e. sent but not acknowledged. The size of the send window changes over time due to the flow and congestion control algorithms. The average rate of a TCP sender (R_{avg}) is given by $R_{avg} = W_{avg} / RTT_{avg}$, where W_{avg} is the average size of the send window and RTT_{avg} is the average round-trip-time (RTT) for that TCP connection. W_{avg} depends on the packet drop probability and thus on the degree of congestion. RTT_{avg} depends on several delay components (transmission, propagation, queuing, processing). Due to the volume based control and the fact that the R_{avg} depends on varying factors that are hard to estimate, the goal of guaranteeing TCP rates is not easily achieved.

An AC scheme for TCP flows that has been investigated by many authors, e.g. in [8, 14], assumes to reject a new connection by dropping SYN and ACK SYN segments in the case of network congestion. For instance, congestion can be identified when the number of waiting packets in the queue exceeds a predefined threshold value. It was shown that this approach could guarantee a fair share of link capacity between running TCP connections but without the possibility of rate differentiation between the flows. For elastic traffic, a list of proposed AC algorithms based on some declarations can be found in [15]. However, none of these proposals is targeted for guaranteeing a requested rate to TCP.

For the PMM service two AC algorithms are implemented. Each of them operates per TCP flow and is of declaration based type. They assume that a user, before establishing TCP connection, submits its request to the network. The traffic contract specifies the target requested bit rate (R_{req}). Furthermore, on the basis of the R_{req} and information about RTT of the TCP connection, the user declarations are mapped into the form of single token bucket parameters, say rate (R) and bucket size (BS), constituting input parameters for the AC decision. The first of the two proposed AC is based on the token bucket marking (TBM); the second one enables an ideal TCP behaviour by setting an appropriate value for the advertised window size.

3.1 TOKEN BUCKET MARKING AND DIFFERENTIAL DROPPING

One approach to the task of assuring TCP rates is to employ a TBM mechanism at the network ingress and to differentially (depending on their marking) drop packets

inside the network. We investigate here the feasibility of such an approach under the assumption that the token bucket is configured with a static parameter set, which is computed from a flow's requested rate.

It is well known that the aggressiveness of TCP flows is indirectly proportional to both RTT_{avg} and W_{avg}. Aggressiveness denotes the ability of a flow to grab bandwidth from the network. In the PMM service class where flows may request different rates and potentially have different RTTs there will thus generally be a competition between flows of different aggressiveness. The fundamental idea of an approach with TBM/differential dropping is to counteract the inequalities of TCP aggressiveness and to trim each flow to its requested rate. It is essential that a flow does not exceed that rate because the extra amount of bandwidth taken by that flow would be missed by the other concurrent flows inside that service class. The TBM is able to push-back aggressive flows and to support the weak flows by marking less/more packets as in-profile.

In the literature there is an accurate model of TCP sending behaviour in a network with token bucket marking at the ingress and differential dropping in the core network [12]. This model looks very promising as it provides closed-loop formulae for the computation of the required token bucket rate/size in order to achieve a target sending rate. We assume here an under-subscribed scenario where – according to the model - it's always possible to achieve the requested rate. The model provides a theoretical solution for assuring TCP rates and we thus evaluate its practical applicability.

The model in [12] requires (amongst other parameters) knowledge of the packet drop probability for out-of-profile packets (p_2) and the flow's average round-trip-time RTT_{avg}. The token bucket rate R that is required to achieve a desired sending rate R_{req} is computed as [12]:

$$R = \begin{cases} R_{req} - \dfrac{3}{2R_{req}\,p_2\,RTT_{avg}^{\,2}} & R_{req} \le \dfrac{3W}{2RTT_{avg}} \\ \dfrac{4}{3}\left(R_{req} - \dfrac{3}{2RTT_{avg}\sqrt{2}}\sqrt{Z + \dfrac{1}{p_2}} \right) & R_{req} > \dfrac{3W}{2RTT_{avg}} \end{cases} \quad (2)$$

where $W = \sqrt{2(BS + 1/p_2)} + 2\sqrt{2BS}$, and *BS* is the token bucket size.

After the computation of the token rate R the amount of resources needed for that request can be determined. For those flows, where $R > R_{req}$, the flow requires at least an available bandwidth of R in order to obtain R_{req}. For those flows, where $R < R_{req}$, an amount of R_{req} must be available.

On this basis we can deduce a simple admission control rule that can be considered as a special case of peak rate allocation. The resources required by a single flow are expressed by the greater value of the token rate R and the requested rate R_{req}. The inequality in (3) ensures that the bandwidth required by the aggregate stream submitted to PMM is smaller than ρ_{PMM} times the capacity C reserved for this service, where ρ_{PMM} is an over-allocation factor to keep some safety margin.

$$\sum_{i=1}^{N_{PMM}} \max(R_i, R_{req,i}) \le \rho_{PMM} \cdot C \quad (3)$$

The number of flows in the PMM (including the new one if being admitted in this service) is denoted by N_{PMM}.

The main problem of the above sketched approach lies in the estimation of input parameters needed for the TBM, especially the average round-trip-time RTT_{avg} and the drop probability for out-profile packets p_2. We leave aside the discussion of finding a reasonable estimation for RTT_{avg}. Instead we focus on p_2 because it seems impossible to find an accurate estimation for this parameter. It would require a constant drop probability, independent of the number of congested routers in the flow's path and the level of congestion. However, p_2 depends on the portion of out-profile packets in relation to the total number of packets arriving at a router ("out-share"). This out-share in turn depends on (i) the size of requested rates and (ii) on what portion of the capacity that is reserved for the PMM service, is currently allocated to other flows.

Dependence (i) is due to the fact that flows with lower bandwidth requests produce more out-of-profile packets than flows with higher bandwidth requests; this general effect is reinforced if the token bucket rate is computed according to (2). Dependence (ii) is due to the fact that greedy TCP flows will fully utilize the available capacity - independently of the requested rate.

Interestingly, it is not possible to make a worst-case estimation of p_2. In the one case, if p_2 is estimated too low, the sending rates of the TCP flows are over-estimated and the resulting token rates are too small. For those requests where $R > R_{req}$, the $\max(R,R_{req})$ is smaller than the amount of bandwidth that is really needed by that flow. Thus, in general, too many flows would be admitted.

In the other case, if p_2 is estimated too high, the resulting sending rates of the TCP flows are under-estimated and the computed token rates are too high. This again leads in general to a situation where too many flows are admitted because the real TCP rates are higher than the ones used for the admission control algorithm.

Consequently, the difficulty of estimating the parameter p_2 leads to the problem of incorrect admission control. It is thus very questionable if providing TCP rate guarantees is feasible on the basis of TBM with a *static* configuration and differential queue management. This motivates the investigation of adaptive markers. One such approach has been published quite recently [17].

To investigate the impact of incorrect parameter estimations, a large simulation study is conducted in [21]. In this work, a broad spectrum of the input parameter space is explored to identify inter-parameter dependencies and to find a feasible service class configuration.

Although ideal traffic handling is not feasible with a static marking profile the simulation results encourage the practicability of a real-world implementation. Despite simulations were run at a very high service class utilization (and thus an unrealistically high service request blocking probability) a set of configuration parameters that enables a successful operation could be identified.

The QoS objectives can be more easily reached under a lower service utilization where more unallocated resources are available. Please see [21] for the details.

3.2 AC ALGORITHM WITH ADVERTISED TCP WINDOW SETTING

The second AC algorithm for PMM takes into account the following factors: the TCP behaviour, the token bucket mechanism features, and the impact of the RTT. We argue that the above elements have significant impact on TCP throughput guarantees. More specifically, we aim at guaranteeing the TCP throughput to be stable and close to the requested bit rate (R_{req}), even if the admitted connections differ in R_{req} as well as minimum RTT values.

For understanding the proposed AC we start with short discussion about the desirable traffic profile produced by a TCP source. We state that such a profile is achieved when TCP sends packets with maximum rate keeping lossless packet transfer. When a packet is lost the TCP mechanisms, like congestions avoidance, fast retransmit/fast recovery or slow start, are activated leading to reducing TCP send window and, as a consequence, TCP throughput. Therefore, we argue that for meeting the QoS objectives of PMM we need an ideal (or almost ideal) TCP behaviour, as discussed in [9].

The shape of the TCP send window size evolution in the case of ideal TCP behaviour is as shown on Figure 3. In this scenario no packet losses are observed as long as the size of the send window does not exceed a predefined threshold value, say W_{req}. However, such an ideal TCP behaviour is only feasible by setting a proper value for the advertised window size in the receiver, which should be equal to W_{req}.

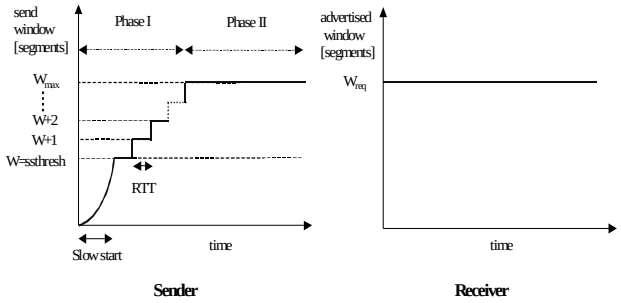

Figure 3: Ideal TCP behaviour.

One can distinguish between two phases of the TCP send window evolution as depicted on Figure 3 and corresponding to: (I) the beginning of the connection when slow start and congestion avoidance mechanisms govern the TCP behaviour, and (II) the time, when the send window size is stabilised at the value W_{max} equal to advertised window (set in the receiver to the W_{req}). Notice that the R_{req} corresponds to the phase II.

Furthermore, we assume that the traffic emitted by a single TCP source is described in the form of the token bucket parameters, in addition allowing us for the in-profile traffic policing. The starting point for the token bucket dimensioning are the TCP connection demands expressed by R_{req} and minimum RTT, RTT_{min}. Next, on the basis of R_{req} and RTT_{min}, we can calculate parameters of the associated token bucket, (R, BS), and, additionally, the advertised window size W_{req}. The R value depends on R_{req} while the BS on the required maximum send window size (equal to W_{req}). For calculating W_{req}, we take into account that the R value corresponds to the worst case traffic, which occurs at a minimum RTT. The expression for W_{req} is:

$$W_{req} = R * RTT_{\min} \tag{4}$$

In addition, one can write the following relation for phase II of the assumed ideal TCP behaviour:

$$R_{req} = \frac{W_{req}}{RTT_{avg}} < R \tag{5}$$

Substituting (4) to (5), we receive the relation between R and R_{req}, which is:

$$R = R_{req} \frac{RTT_{avg}}{RTT_{\min}} \tag{6}$$

For the purpose of the RTT_{avg} calculation, we make the simplified assumption that the variable part of this parameter, say RTT_{var}, caused by buffering packets in

routers, is mainly the packet waiting times in the edge routers. As a consequence we can write:

$$RTT_{avg} = RTT_{\min} + RTT_{\text{var}} \quad (7)$$

The proposed approximation for RTT_{var}, which gives accurate results, is obtained by assuming a M/D/1 queuing system with $\rho=R_{req}/R$ and the constant packet service time equal to MTU/R (MTU – maximum transfer unit).

Taking into account the above approximation, the final formula for R is the following:

$$R = R_{req} + \sqrt{A/(R_{req} + \sqrt{A/(R_{req} + \sqrt{A/R_{req}})})} \quad (8)$$

where

$$A = \frac{R_{req}^2 MTU}{2RTT_{\min}}. \quad (9)$$

The BS (in bits) is calculated in straightforward way from the token bucket features:

$$BS = \frac{W_{req} * (L_{in} - R)}{L_{in}} \quad (10)$$

where the L_{in} denotes the bit rate of the link connecting the host to the edge router (see Figure 1).

The proposed AC rules are the following. Let us assume that the link capacity C with the associated buffer of size B is allocated for handling PMM traffic. The AC mechanism admits a new flow, which now is described by (R_{new}, BS_{new}), if the following conditions are satisfied:

$$\sum_{i=1}^{N_{PMM}} R_i + R_{new} \le C \quad (11)$$

$$\sum_{i=1}^{N_{PMM}} BS_i + BS_{new} \le B \quad (12)$$

where N_{PMM} denotes the number of running TCP flows, each described by (R_i, BS_i), i=1,..,N_{PMM}.

For applying the considered AC algorithm in an effective way, an additional functionality is required from TCP applications. The TCP application has to inter-work with the AC mechanism to allow for setting the advertised TCP window size. For instance, this can be achieved as follows. A user declares R_{req} value with the aid of the end user application toolkit EAT. For the purpose of the admission decision the ACA calculates W_{req} and the token bucket parameters. Next, the W_{req} value is sent jointly with the admission decision to the calling part. The received value of W_{req} is used by the TCP application to set value of the receive buffer size (=W_{req}) in both the calling and called parts. This is possible to be done by some socket API mechanisms during the TCP connection set-up phase. In addition, a simple mechanism have to be implemented for sending W_{req} from the ACA to the TCP application.

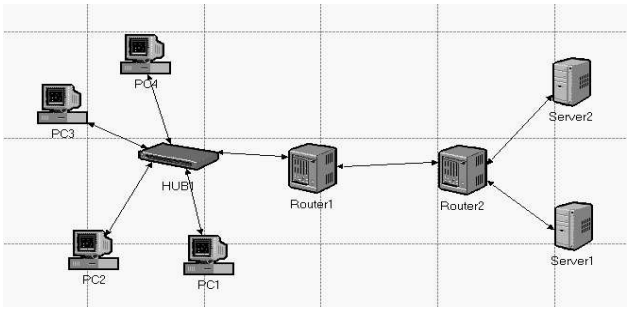

Figure 4: Network topology.

The capabilities of the proposed AC for guaranteeing QoS are illustrated by the simulation results corresponding to the TCP connections differing in the requested bit rates. A single bottleneck network topology with a link capacity of 2 Mbps and 4 running TCP connections was assumed as shown on Figure 4. The TCP connections are of greedy type, sending packets of constant size, MTU=1500 bytes. The RTT_{min} is the same for each connection and equals 0.1 sec.

Table 1: Received TCP throughput for each of TCP connections; C=2 Mbps, B=16 packets, RTT_{min}=0.1 sec.

*1	R_{req} /R (kbps)	W_{req} (Bytes)	BS (Bytes)	RTT_{avg} (appr/ sim) *2 (sec)	Thr*3 (kbps)
#1	288/400	5000	4800	0.14/0.12	333–356
#2	288/400				
#3	456/600	7500	7050	0.11/0.12	503–528
#4	456/600				

*1 TCP Connections (#1: PC1-Server1; #2: PC2-Server1; #3: PC3-Server2; #4: PC4-Server2)

*2 RTT_{avg} (appr): average value of round trip time obtained from approximate formula (7); RTT_{avg} (sim): average value of round trip time received from simulation

*3 Thr: TCP Throughput

Table 1 shows the results of the received throughput (with confidence interval 95%) for each of 4 TCP connections. The results were obtained by simulation in OPNET with TCP Reno implementation, with the proper settings of the advertised window size. One can observe that the received throughput is a bit greater than the requested rate. This result is satisfactory since we can guarantee for a user the TCP throughput close to the R_{req}. Notice, that in this case no packet losses occurred, as it was expected.

More results confirming the effectiveness of the investigated AC algorithm with advertised window setting we

presented in [19] and [20]. In [19], we show simulation results for the case with TCP connections differing in R_{req} and RTT_{min}. while in [20] we include measurement results from the AQUILA pilot network for both presented AC algorithms.

4 SUMMARY

In the paper we present the AC algorithms implemented in the AQUILA QoS IP network. They correspond to variable bit rate streaming and TCP-controlled traffic (of greedy type). For streaming traffic, a measurement-based approach is investigated. For TCP-controlled traffic, two types of AC are investigated, one based on a token bucket marking approach and the second one leading to the ideal behaviour of TCP (no packet losses) thanks to the appropriate setting of the advertised TCP window size. Different aspects of the applicability of these algorithms are discussed. Exemplary simulation results confirming the expectations are included.

Manuscript received on …

REFERENCES

[1] S. Blake et al., An Architecture for Differentiated Services, RFC 2475, December 1998.

[2] J. Roberts, U. Mocci, J. Virtamo (Eds.) , Final Report COST 242, Broadband network teletraffic: Performance evaluation and design of broadband multiservice networks, *Lectures Notes in Computer Science 1155*, Springer, 1996.

[3] A. Bak, W. Burakowski, F. Ricciato, S. Salsano, H. Tarasiuk, Traffic handling in AQUILA QoS IP network, Quality of Future Internet Services, *Lecture Notes in Computer Science 2156*, Springer 2001.

[4] AQUILA Project Consortium, Deliverable D1302, Specification of traffic handling for the second trial, http//www.ist-aquila.org, December 2001.

[5] W. Burakowski et al, AQUILA network architecture: first trial experiments, Special Issue of *Journal of Telecommunications and Information Technology* No. 2/2002, Warsaw 2002.

[6] R. J. Gibbens, F. P. Kelly, Measurement-based connection admission control, *15th International Teletraffic Congress*, June 1997.

[7] W. Burakowski, M. Dabrowski, Multiservice QoS IP network: architecture and practical verification in pilot installation, *Telecommunication Review*, No. 5/2002 (in Polish), Warsaw 2002.

[8] M. Benameur, S. Ben Fredj, F. Delcoigne, S. Oueslati-Boulahia, and J. W. Roberts, Integrated Admission Control for Streaming and Elastic Traffic, Quality of Future Internet Services, *Lecture Notes in Computer Science 2156*, Springer 2001.

[9] H. ElAarag, M. Bassiouni, Performance evaluation of TCP connections in ideal and non-ideal network environments, *Computer Communications* 24 (2001), pp. 1769-1779.

[10] S. Floyd, V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking*, August 1993.

[11] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP Throughput: A Simple Model and its Empirical Validation, Proc. *ACM SIGCOM'98*, August 1998.

[12] S. Sahu, P. Nain, D. Towsley, C. Diot, V. Firoiu, On Achievable Service Differentiation with Token Bucket Marking for TCP, Proc. *ACM SIGMETRICS'00*, Santa Clara, CA, June 2000.

[13] T. Ziegler, C. Brandauer, S. Fdida, A quantitative Model for the Parameter Setting of RED with TCP traffic, The Ninth *International Workshop on Quality of Service* (IWQoS'2001), Karlsruhe, Germany, June 2001.

[14] R. Mortier, I. Pratt, C. Clark, and S. Crosby, Implicit Admission Control, *IEEE Journal on Selected Areas in Communications*, Vol. 18, No. 12, December 2000.

[15] F. Brichet, M. Mandjes, M. F. Sanchez-Canabate, Admission control in multiservice networks, Proceeding of the Mid-Term Seminar COST 257, Villamora, Portugal, 1999.

[16] F H. P. Fitzek, M. Reisslein, MPEG-4 and H.263 video traces for network performance evaluation, Technical report TKN-00-06, Technical University Berlin, Dept. of Electrical Eng., Germany, October 2000.

[17] Y. Chait, C.V. Hollot, V. Misra, D. Towsley, H. Zhang and C. S. Lui, Providing Throughput Differentiation for TCP Flows Using Adaptive Two-Color Marking and Two-Level AQM, Proceedings of the *INFOCOMM'2002* Conference, San Francisco 2002, USA.

[18] M. Dąbrowski, F. Strohmeier, "Measurement-based Admission Control in the AQUILA Network and Improvements by Passive Measurements", Architectures for the Quality of Service Internet, W. Burakowski, B. Koch, A. Beben (eds.), *Lecture Notes in Computer Science 2698*, Springer Verlag 2003, pp.189-202.

[19] W. Burakowski, H. Tarasiuk, "Admission Control for TCP Connections in QoS IP Network", Web and Communication Technologies and Internet-Related Social Issues, Chin-Wan Chung, et al (eds.), *Lecture Notes in Computer Science 2713*, Springer Verlag 2003, pp. 383-393.

[20] M. Dabrowski, G. Eichler, M. Fudala, D. Katzengruber, T. Kilkanen, N. Miettinen, H. Tarasiuk, M. Titze, „Evaluation of the AQUILA Architecture: Trial Results for Signalling Performance, Network Services and User Acceptance", Architectures for Quality of Service in the Internet, W. Burakowski, B. F. Koch, A. Beben (eds.), *Lecture Notes in Computer Science 2698*, Springer Verlag 2003, pp. 218-233.

[21] C. Brandauer and P. Dorfinger, An implementation of a service class providing assured TCP rates within the

AQUILA framework, Architectures for the Quality of Service Internet, W. Burakowski, B. Koch, A. Beben (eds.), *Lecture Notes in Computer Science 2698*, Springer Verlag 2003, pp. 203-217.