Auction-based Admission Control for Self-Aware Networks

Georgia Sakellari[1], Timothy Leung[2], and Erol Gelenbe[3]

[1] University of East London, School of Computing, Information Technology and Engineering, E16 2RD London, UK, g.sakellari@uel.ac.uk
[2] Imperial College London, AESOP Group, Department of Computing, SW7 2AZ London, UK, timothy.leung08@imperial.ac.uk
[3] Imperial College London, ISN Group, Electrical & Electronic Engineering Department, SW7 2BT London, UK, e.gelenbe@imperial.ac.uk

Abstract. In the future, networks will be increasingly Quality-of-Service (QoS) oriented, and admission control will play a vital role in ensuring that relevant QoS requirements of flows are satisfactorily met. Recent research viewing the network as a chargeable resource, associating a pricing mechanism for different types of data, is also becoming increasingly prevalent. In this paper, we introduce the idea of using auctions to coordinate the distributed admission control system for self-aware networks we have developed in the past. The proposed system employs the use of quota at each of the nodes in order to regulate congestion in the network. The current level of quota at a node indicates the amount of additional burden the node is permitted to introduce into the network through the acceptance of flows. If a node does not have enough quota to admit an incoming flow, it can request additional top-up quota whose allocation is controlled by an auctioning mechanism.

1 Introduction

Data transmission requests by QoS-oriented services, or flows, can be parameterised by a set of QoS requirements and a source-destination pair. A network will only be able to admit and satisfy a certain number of such flows before resource and topology limitations of the network mean that flow QoS requirements cannot be met. An admission control (AC) system governs the admission of each flow given the current congestive state of the network and the flow's QoS requirements, such as required bandwidth, delay, jitter and packet loss. AC systems can either be centralised or distributed; limitations of having a centralised system are such that it acts as a single point of failure and it bottlenecks the system by limiting flow admission requests to the rate at which it can process them. Distributed systems, on the other hand, base their decisions on limited knowledge and might not be very accurate. Here, we propose a novel auction-based, measurement-based admission control system that tries to overcome the limitations of both a centralised and a completely distributed system. It uses a quota system, derived from the burden of a flow on the network, which input nodes

use to restrict flow admission. Our scheme is implemented atop the Cognitive Packet Network (CPN) [**?**,**?**,**?**], since it already employs a measurement-based infrastructure, and atop the distributed AC algorithm presented in [**?**,**?**], which is also measurement-based and allows different users to have different and multiple QoS criteria when entering the network.

This paper's purpose is to provide a proof of concept of how auctions could help improve the performance of the distributed AC presented in [**?**,**?**]. We do not elaborate on how to dynamically calculate quota in each machine. This is something that we are currently working on.

2 Our proposed algorithm

Our auction-based distributed admission control system employs the use of quota at each of the nodes in order to regulate congestion in the network. The current level of quota at a node indicates the amount of additional burden it is permitted to introduce into the network through the acceptance of flows. If the distributed admission control (DAC) algorithm described in [**?**,**?**] decides that a new flow requesting to enter the network can be accommodated without affecting the existing flows of that node and the node has enough quota, then the flow will be accepted. If the DAC decides a new flow can be accepted but the node does not have enough quota, it can request additional top-up quota whose allocation is controlled by an auctioning mechanism. An advantage of auction-based DAC is that it is able to exercise a degree of coordination and control over the admission of flows into the network only when the network's available capacity becomes scarce.

In order to be admitted, an incoming flow will follow these steps:

1. Given that q is a measure of network burden, the flow is assigned a value of Δq, which can be interpreted as the extra burden the additional flow will place on the network. It can be computed through the probing process used by the DAC. By also applying the DAC we can have an estimation on whether the network can accommodate the new flow or not. If the flow cannot be accommodated in the network, it will be rejected outright.
2. Each input node has pre-allocated quota and is permitted to grant immediate entry to any flow for which it currently has sufficient available quota. If the incoming flow is admitted, the amount of available quota at the input node is decremented accordingly, and only when the flow frees up the connection is the used quota restored. If, however, the input node has insufficient quota, the admission request from the incoming flow is turned over to the auctioning mechanism.
3. The auctioning mechanism allocates top-up quota, denoted by δq, to input nodes in order to cover the difference between the amount required for the input node to admit a flow and the amount that the input node has available. The auctioning mechanism may be viewed as a reverse auction that will, over time, receive decreasing bids for top-up quota from input nodes. Depending on the auction algorithm, the auction will terminate with an appropriately

low bid being accepted and the amount of quota corresponding to that bid will be lent to the input node that submitted the bid in order to accept a flow; the top-up quota is to be returned when *any* connections terminate from that input node.

Many auction mechanisms can be used. The simplest one would be to have a constant amount of time that the auctioneer waits for bids before awarding the top-up quota to the lower bidder. The auction mechanism described in [**?**] can also be used. The goal of the AC is to admit as many flows as possible as fast as possible into the network, while satisfying all QoS requirements for admitted flows. Quota at each input node governs how much autonomy the input node has over flow admission and increases the speed at which the AC can make admission decisions. Too low a level of quota at an input node will tend to increase the likelihood of invoking the auctioning mechanism which acts as a centralised admission control system by serialising admission requests, while a high level of quota will reduce the rate of traffic directed towards the auctioning mechanism and allow input nodes to admit flows that require a disproportionate amount of quota by constituting a larger burden to the network than other flows vying for quota awaiting admission at other input nodes.

The goal of the auctioning mechanism includes (i) to minimise top-up quota expenditure and conserve its resources so it can continue to run auctions, and (ii) to try to accept as many bids as possible and as soon as possible since each new bid means rejecting or stalling the flow that submitted the previous bid. The balance between these two goals can be pressurised by the amount of top-up quota the auctioning mechanism has at its disposal. We are currently investigating algorithms to allocate quota to both the input nodes and the auctioneer. In the experiments conducted for this paper though, each input node has pre-allocated quota and is permitted to grant immediate entry to any flow for which it currently has sufficient available quota. The quota values we have chosen are explained in Section **??**.

3 Configuration of the experiments

In order to evaluate our mechanisms we conducted our experiments on the real testbed located at Imperial College London (same as in [**?**]). There are three sets of experiments, with different amount of users constantly making requests to send traffic; in the first set we had 5 users corresponding to 5 Source-Destination (S-D) pairs, in the second we had 9 users, and finally in the third and more demanding set of experiments, we had 17 users independently making requests to send traffic into the network. We set a random waiting time W between requests, chosen to be uniformly distributed in the range of $[0, 15]$ seconds. We set the probing rate at 40% of the user's rate and the probing duration at $2s$. In our experiments all links have $10\,Mbits/s$ capacity and all users have the same QoS requirements: round-trip $delay \leq 50\,ms$ and $jitter \leq 2.5\,ms$. When a call is accepted, the source generates UDP traffic of $300kbps$ constant bit rate that lasts for a duration which is uniformly distributed in the range of

values $[10, 200]$ seconds. Also, we have introduced constant background traffic of $3.2Mbps$. Each experiment lasts for $5min$ ($300s$) and is conducted 5 times. The results presented here are the average values of those runs. At a specific input node, in order to avoid having more than one users requesting to enter the network at the same time each user request enters a queue ("request queue"). In the centralised version there is one such queue and it is located in the data collection point, so that all users from all input nodes will queue there in order to enter the network.

Each set of experiments covers six cases: (i) The centralised AC, proposed in [**?**] (CAC), (ii) The AC in each input node simply allows everybody to enter the network (DAC-All), (iii) The DAC proposed in [**?**,**?**] (DAC), (iv) The DAC is enabled and the nodes are fully coordinated (DAC-Full) as explained in [**?**,**?**], (v) The DAC is enabled and the nodes are randomly coordinated (DAC-Rand) as in [**?**,**?**], and (vi) The DAC is enabled and our proposed auction mechanism controls the admittance when a node does not have enough quota (DAC-Auction).

Each input node has pre-allocated quota and is permitted to grant immediate entry to any flow for which it currently has sufficient available quota. Because all users in our experiments have the same QoS criteria, the quota each user needs in order to be accepted into the network depends on (equals) the duration they will occupy the network for. Based on experimental trials we have decided that the total amount of quota to be distributed among the source (input) nodes of the network is 5000. So, the auction quota at each input node when we have 2 users is 2500, when we have 5 users is 1000, and for 13 users making requests into the network the value drops to 385 per input node. Also, the auction quota when we have 2 users is 7500, when we have 5 users is 3000, and for 13 users making requests into the network the auctioneer's quota is 1150. The auctioneer's total quota also depends on the number of source nodes in the networks so that a source node will not be allocated more quota that it can handle. Another way of assuring this will not happen is by having a maximum amount of quota a node can have. As the auction mechanism is concerned, we have chosen a basic auction where the auctioneer waits for a constant amount of time, in this case $1s$, before deciding the winner.

4 Experimental Results

In order to evaluate the performance of our algorithm in each set of experiments, four areas are covered. First, we measure the satisfaction of each user (see Figure **??**). We consider the satisfaction of a user as the percentage of time, over the whole lifetime of the user, where all the QoS requirements of that user are fulfilled. The results presented here are the average values for all users in the network. Secondly we measure the average time a user has to queue before being given a decision on whether to enter the network or not (see Figure **??**). Finally we count the total number of requests that were served during the duration of the experiments (see Figure **??**) and how many of them were actually accepted (see Figure **??**).

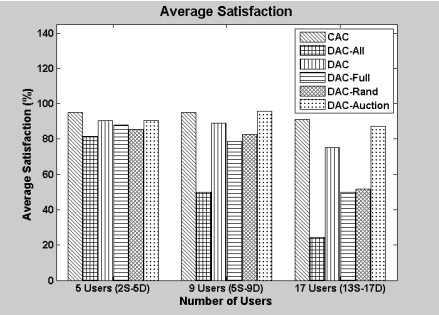

Fig. 1. Average Satisfaction of the Users

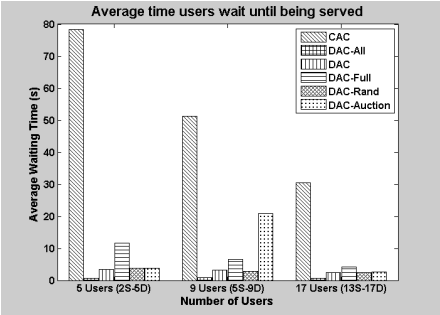

Fig. 2. Average time a user waits in the "request queue" before being served

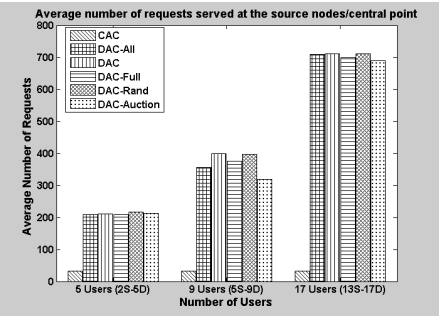

Fig. 3. Average Number of Requests

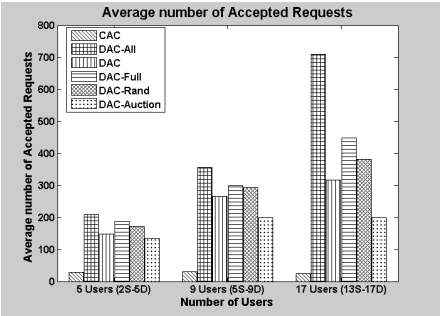

Fig. 4. Average Number of Accepted Requests

Figure **??** reveals that the shortest average time the users have to wait until being admitted is when no AC is used. This waiting time is close to zero but not zero, since in the cases where multiple users request to enter the network at the same input node there is still a queue at the input node. The longest waiting time is when the CAC is used since all users queue at the central point. When we apply the DAC, the time a user waits until being either admitted or rejected, is the shortest between all the distributed versions. The co-ordination slightly increases the waiting time due to the overhead caused by message exchanges. Finally, with the auction-based scheme, the waiting time is comparable to the DAC, case except when having 9 users in the network. In that scenario, the waiting time is quite high, which is also reflected in the total number of requests in Figure **??**. This could be due to the sub-optimality of the quota value.

Waiting for service affects the number of users that are actually accepted into the network. Figure **??** shows that centrally controlled admission of users results in small number of users served and accepted into the network, which

was expected based on the waiting times shown prior, while in all other cases the number of requests served is more or less the same. The fact that in some cases less requests were made when all users were accepted (DAC-all) is due to the randomness in the generation of the requests.

In practice, the centralised algorithm offers greater user satisfaction but fewer users (see figures **??** and **??**). The satisfaction of the DAC is lower probably, because multiple probes are present in the network and each input node has limited information, making the estimation of the algorithm less accurate. The use of coordination does not improve the DAC algorithm since it introduces the exchanging of messages which make the estimations less accurate. By allowing the DAC mechanism to work while the network traffic is low and serialising the admission mechanism when the network is close to congestion through the use of an auction process, we are able to achieve satisfaction rates close to the ones of the centralised AC but with considerably more users accepted into the network.

5 Conclusions

This paper constitutes a proof-of-concept that combines the advantages of both the distribution and centralised admission control systems, by way of an admission control system that implements an auction mechanism. Our future work will concentrate on how to calculate the quota and how to allocate these quota to the various input nodes and to the auction mechanism, in a way that reflects the congestive state of the network and the additional burden that nodes are permitted to inflict on the network.