Considering User Behaviours in Active Servers Queue Management

A. Djafari Marbini, L. E. Sacks

Department of Electronic and Electrical Engineering

University College London

Telecommunications Systems Group

{a.marbini@ee.ucl.ac.uk}

Abstract: This paper is an initial study of the impact of the queuing length on overall system optimisation in the context of telecommunication services engineering. The queue lengths are adjusted using Pareto distribution. It is assumed that the user behaviour in sending requests will follow Pareto distribution patterns; in the rate of arrival, in the size of jobs and, indeed in their value. This queuing system is to be implemented in a Swarm model for Active Networks. Considering user behaviour will allow the operators to introduce more effective queues to the active servers so that the system will operate more efficiently.

1 Introduction.

The distribution of user demands varies over several decades. For example, the number of users who have access to SMS and e-mail services are orders of magnitude larger than those with access to video conferencing facilities. Distribution of exposable income, in a given sample of population exhibits Pareto distribution. We are motivated to believe that many attributes of job demands such as the rate of requests and size of requests follow that frequency. Active Networks allow users to add customized programmes into active nodes, so new services can be introduced without having to introduce lengthy standardisation procedures. The increased diversity in services makes the presence of a distributed management system with optimising parameters for CPU usage and data storage more important.

2. The Problem.

Currently server performances vary widely. Specifically, the performance of HTTP cache servers seem to be independent of size and network topology [2], thus suggesting that the quality difference lies upon the user community. It can be assumed that a similar type of behaviour can be observed in other types of severs. Most system allocation of resources at the moment do not make use of user behaviour patterns. The user behaviour is not very easily predictable, because the services are no longer homogeneous. If the operators were to make use of user behaviour in offering the services and providing demands, then the quality of service and user satisfaction could be improved. The two systems (users and service platforms) seem to be working antagonistically, since either the service platform will be overloaded with very long queues or the number of rejected or lost jobs will be very high. A middle ground needs to be established where by the system is not overloaded, but at the same time the number of rejected jobs is kept to a minimum, as delay and throughput are both very important entities in terms of user satisfaction.

An analogy from economics model could be applied to users' request behaviour. It was shown by the economist, Vilfredo Pareto, that for a constant amount of money in a society, the spread of incomes follows a log-log distribution. A society could only tolerate a few millionaires for every million paupers! The number of middle classes would be higher and those with little money would be higher still. If we were to plot the amount of money against the portion in the society on a logarithmic scale, one would see a straight line; this has the name 'Pareto distribution'. Considering that the money would remain constant in the society, if the number of people with a specific portion of money increases, a proportional decrease would be seen in the other categories and vice versa.

If we were to apply a similar model to user demand patterns then a better resource allocation can be applied. It can be assumed that, if we were to define job requests in terms of duration, the number of

requests with smaller durations e.g. e-mail and SMS, is much larger than those with relatively longer durations e.g. video conferencing and on line gaming. This is simply because a larger percentage of users have access to e-mail than they do to video conferencing. If the user demands were to be defined into a number of categories in terms of job durations, a Pareto distribution over a bounded limit could be defined. The network demand patterns can be predicted by looking at the history of requests received previously. When job durations follow Pareto distribution, the operator can make use of this observation and allocate specific queues for requests of particular durations. This would mean that each queue would only accept specific type of jobs, and it would reject all others. The system will implicitly assume, that although a particular queue is empty, but rejecting an immediate incoming job would allow, a larger request that is expected to come, to fill a larger portion of the queue. This would achieve an overall better performance by the system.

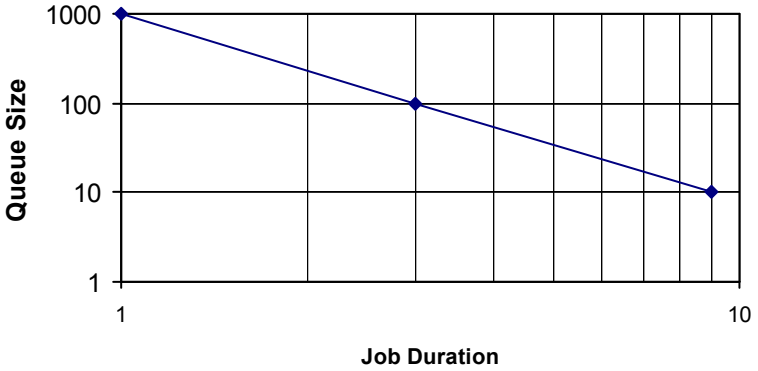
This model uses the experience of traffic jam management. What would be the effect on overall traffic throughput, if traffic lights were added to a long road with no cross streets [3]. Although one might assume that this would reduce the throughput, but in fact it was found, that in particular situations additional traffic lights, improve the overall throughput. In a two-lane road if one lane slows down a lot of people will switch to the other lane. The result will be that after a while the first lane starts going and the second one slows than. Unless a system is put in place to stop the cars from moving from one lane to the other, the traffic will oscillate from one lane to the other, thus reducing the overall throughput. If speed limits and speed cameras were introduced, so that the cars would slow down, there would be a small jam behind a camera. The cars will speed away after the camera. They will form a queue of roughly equal distances between the cars. As the cars move away from the camera one by one they accelerate smoothly until they reach the speed limit.

The speed limits in specific lanes are comparable with having queues, which only accept jobs of particular size and reject all others. This might cause an immediate traffic of job requests but overall it would allow the system to work more efficiently.

3. The Model.

In the model used in this paper, a single server is receiving different types of job requests from a collection of users. The types of jobs are defined by the size of the duration. They are divided into three different categories of small, large and medium sized jobs. There are three classes of queues within the server, specific to each category of job. The job durations were decided to have constant values. The queue sizes were decided as such that, they followed a Pareto distribution pattern.

Figure 1: Queue Size against Job Duration (Logarithmic Scale)



In a particular time interval the users request, a random number of randomly sized jobs. This is implying that the requests are being generated with Gaussian distribution. The distribution is defined between 0 and system capacity, which is the total number of jobs that can be accepted by the server, before it starts overflowing. In the particular example that can be seen in figure 1, a queue of size

1000 is assigned to jobs with duration 1, a queue of size 100 is assigned to jobs with duration 3, and similarly a queue of size 10 is assigned to jobs with duration 9. The assumption being, that the users are 100 times more likely to request a job with smallest duration than a job with largest duration. This queue size pattern was chosen, in order to follow the user request pattern. Over the duration of the whole experiment the users' request rates will follow Pareto distribution.

3. Results

The simulation of the model described in the pervious section was ran several times. Each time the simulation was ran for duration of 6000 time units (Epochs). This was done so that various combinations of system queues could be tested and the resulting delays and job rejections were recorded for several cases. The different tests were done by changing the number of jobs in each queue, but at the same time retaining the integral of graph (Figure 1) a constant value. The other condition that had to be satisfied was that, the graph (Figure 1) had to maintain its Pareto distribution format. The delays were measured by, finding the difference between job request arrival time and the time that the job request left the queue. The values plotted on Figures 2 and 3 were the values after the system reached a steady state. The graphs labelled figures 2 and 3 show the simulation results of three test scenarios. All test conditions remained the same, other than the queue sizes. The queue sizes were as follows in each test

Test 1: $Q_1 = 1000$ $Q_2 = 100$ $Q_3 = 10$.
Test 2: $Q_1 = 850$ $Q_2 = 100$ $Q_3 = 13$.
Test 3: $Q_1 = 500$ $Q_2 = 100$ $Q_3 = 20$.

The plots in figures 2 and 3 show that, test 1 had the most number of jobs rejected and the in test 3 the delays were the highest, in all three queues. As expected the number of rejected jobs increases, as the delay in the queue decreases.

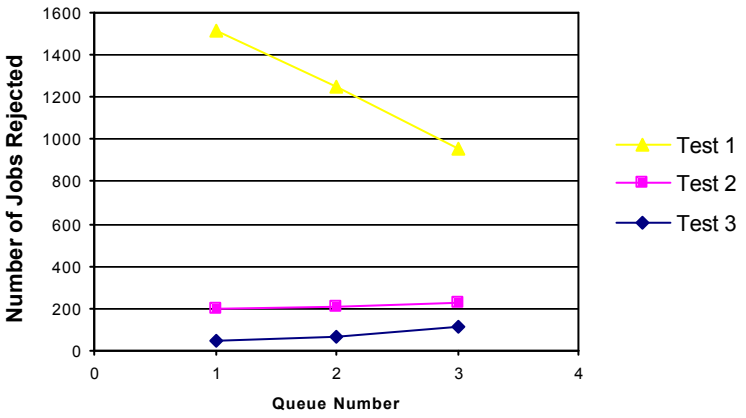


Figure 2. The number of jobs rejected in each queue is shown for three different tests.

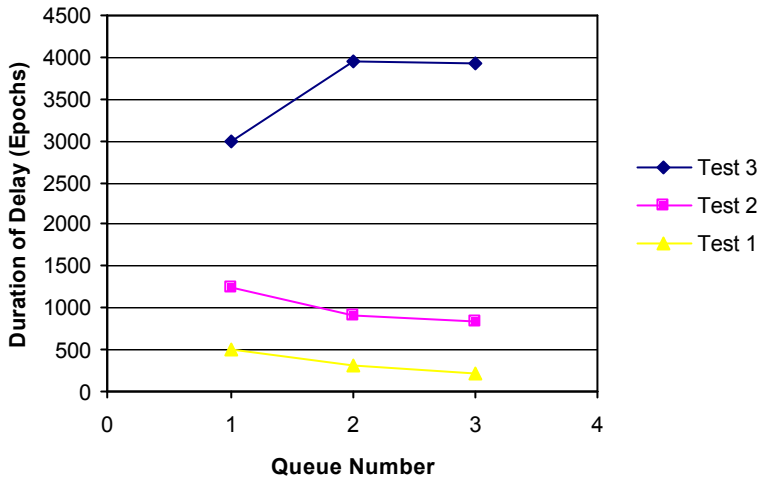


Figure 3 The duration of delay in each queue is shown for three different tests.

Scenario 2 gave the best overall outcome for all queues. It showed the overall best performance between achieving the minimum requests rejected and the minimum delay in the system. Although the decision about the best queue size combination would depend on what type of quality of service is expected by the users and the operators. For instance if users are requiring video conferencing services then very long delays will disrupt the service, but in the case of e-mail users, it would be more important for the job not to be lost rather than how long it would take it for it to arrive.

The job arrival rate is following Pareto distribution. The queue sizes are also following a Pareto distribution. Therefore, the closest the gradient of the Queue Size-Job Duration (On logarithmic scale) graph is to the gradient of the arrival rate, the better the system performance would be.

4. Conclusion.

In this simulation we have used the traffic jam analogy to efficiently manage server queues. Predicting the behaviour of the users, by looking at previous user request is useful. By manipulating user behaviour a better system performance and improved quality of service could be achieved. The queuing system described, will be implemented in an Active Network model [4].

References.

[1] I.W.Marshall, H.Gharib, J.Hardwicke and C.M.Roadknight. "A novel architecture for active service management" IM2001.
[2] I.W.Marshall, and C.M.Roadknight. "Linking Cache Performance to User Behaviour". Computer Networks and ISDN systems, 30 (1998) pp2123-2131
[3] M. Resnick. "Turtles, Termites, and Traffic Jams- Explorations in Massively Parallel Microworlds". MIT Press, (1999)
[4] A. L. M. Ching, S. L. Ling, L. E. Sacks. "Using Swarm to Model Network Complexities" LCS (2001)