Admission Control in Self Aware Networks

Georgia Sakellari, Maurizio D'Arienzo, Erol Gelenbe
Intelligent Systems and Networks Group
Electrical & Electronic Engineering Dept.
Imperial College
London SW7 2BT
{g.sakellari,m.darienzo,e.gelenbe}@imperial.ac.uk

***Abstract*—The worldwide growth in broadband access and multimedia traffic has led to an increasing need for Quality-of-Service (QoS) in networks. Real time network applications require a stable, reliable, and predictable network that will guarantee packet delivery under QoS constraints. Network self-awareness through on-line measurement and adaptivity in response to user needs is one way to advance user QoS when overall network conditions can change, while admission control (AC) is an approach that has been commonly used to reduce traffic congestion and to satisfy users' QoS requests. The purpose of this paper is to describe a novel measurement-based admission control algorithm which bases its decision on different QoS metrics that users can specify. The self-observation and self-awareness capabilities of the network are exploited to collect data that allows an AC algorithm to decide whether to admit users based on their QoS needs, and the QoS impact they will have on other users. The approach we propose finds whether feasible paths exist for the projected incoming traffic, and estimates the impact that the newly accepted traffic will have on the QoS of pre-existing connections. The AC decision is then taken based on the outcome of this analysis.**

I. Introduction

The worldwide growth in broadband access and multimedia traffic has led to an increasing need for Quality-of-Service (QoS) in networks. Real time applications, such as Internet telephony (Voice-over-IP), video conferencing, remote teaching, remote medical diagnosis and treatment, and online trading systems, require stable, reliable, and predictable networks that will guarantee packet delivery under QoS constraints. This is not secured by the current "best-effort" Internet architecture. A promising solution, capable of satisfying those demands, is Self-Aware Networks [1]. These are packet networks that use the CPN [2] or other adaptive packet routing protocols and address QoS by using adaptive techniques based on on-line measurement. CPN is designed to perform Self-Improvement by learning from the experience of smart packets and by using neural networks and genetic algorithms. In Self-Aware Networks, as with all types of networks, congestion needs to be carefully examined. A way to control traffic congestion and satisfy users' Quality of Service requests without overprovisioning, is to implement admission control (AC), which is the purpose of the work presented in this paper.

AC is one of the classical problems in networks which has been addressed since the early days of telephony. In its simplest and broadest form, the problem is to determine whether some user u who requests some service S and is willing to pay a price Π for a level of Service Q should be admitted into the network. The decision should obviously be taken in terms of the network's ability to satisfy the user, the reward that the network expects to obtain by offering this service to the user, and the risk that it might incur if it does accept the user but is unable to provide this user and other pre-existing users with a satisfactory service. Factors involved in the AC decision are:

- The measurement of up to date QoS metrics that relate to the request that is made by user u;
- The estimation of the impact of the new flow on the QoS experienced by current users;
- The cost benefit analysis of the acceptance or rejection decision with respect to overall economic or utility considerations for the network.

In a Self-Aware network we propose to establish a connection for user u to a service S with the following elements:

- The AC constantly gathers data on the ongoing users QoS in the network by exploiting the network's Self Awareness.
- u turns to the AC system and indicates that it wants a service S at QoS value Q for a price Π.
- The AC executes an algorithm to discover at the feasible paths that may satisfy u's requirements while other existing flows' QoS needs are also satisfied.
- All such paths are probed to see whether the addition of the new user will preserve the preceding conditions.
- If a satisfactory path is found then u is accepted. Otherwise it is rejected.

In this paper we will detail this approach, describe an implementation in our network test-bed, and report on some experimental results.

The paper is organised as follows: the next section reviews the literature on AC and Section IV introduces a multiple criterion AC algorithm based on a probing technique. In Section III we present an existing architecture that implements the Self Aware Network concept in the framework of the CPN systems [1]. Section V reports preliminary results of the AC algorithm we describe, using our existing CPN testbed. Concluding remarks can be found in Section VI.

II. AC: Related work

There is abundant literature about AC in networks and we will only point to some of the relevant work. **Rate or**

Simple Sum is the simplest parameter-based algorithm, and the most widely implemented by switch and router vendors [3]. It ensures that the sum of requested resources does not exceed the link capacity. The Simple Sum algorithm does not take into account QoS metrics other than bandwidth, and it assumes that every user will use all of its reserved bandwidth. A measurement-based version of Rate Sum is **Measured Sum** [3] which tries to increase the network utilization by measuring the actual network load and substitutes the reserved rates of the existing users with the measured load . Again the admission decision is based only on the availability of bandwidth and does not consider any other QoS metric.

Acceptance Region schemes decide whether to admit a new flow based on the current state of the system and whether the state lies within the "acceptance" or "rejection" region. The acceptance region is calculated in order to maximize the line utilization for a nominal packet loss, given a set of flows with a given declaration of peak and mean rates. The calculation in both [4] and [5] assumes that calls arrive according to a Poisson process, that the calls admitted are independent and stay in the network for exponentially distributed time, that they have identical bandwidth requirement statistics and they are described by a continuous-time ergodic Markov fluid process. Though the acceptance region algorithms are quite simple, this simplicity comes as a result of simplifications of the network model, which results in limitations in such algorithms. For instance, they perform quite poorly at low link capacity. Another limitation is related to the often made assumption of homogeneous on-off sources. Thus it may not be clear whether such algorithms are still applicable when the traffic sources do not fit this model.

Measurement Based Admission Control with Delay and Bandwidth Constraint schemes do both delay and bandwidth checking and are used with predictive service for "tolerant" applications which allow a certain degree of QoS violations. When a new flow requests service, the network must characterize its traffic. The algorithm described in [6], is a measurement based admission control algorithm for predictive service which approximates the maximum delay of predictive flows by replacing the worst-case parameters in the analytical models with measured quantities. The computation of worst-case queueing delay is different for guaranteed and predictive services. Such algorithms cannot be used in networks with strict QoS requirements and are only applicable in networks with predictive service. Also the schemes tend to exceed the needed bandwidth reservation, since they use worst-case delays, which is rarely the case since multiple sources will rarely simultaneously transmit packets at peak rate.

Equivalent Bandwidth, is the minimum bandwidth that is needed to carry the traffic that is generated by a source in isolation, without violating the QoS requirements. In this approach, each source is assigned an equivalent bandwidth and a new call is accepted if the sum of these equivalent bandwidths is less than the capacity of the links. In [7] and [8] a fluid-flow model for the source and a bandwidth allocation process is used to calculate the equivalent bandwidth either by taking into account the impact of source characteristics (the duration of the burst period) when the impact of individual connection characteristics is critical, or when the effect of statistical multiplexing is of significance. Equivalent bandwidth schemes are characterised by their simplicity, since determining whether a given set of traffic sources can be accommodated without any QoS violation comes down to comparing the sum of the equivalent bandwidths of individual sources to the link capacity. This approach does not consider the effect of buffering which increases the effective capacity of a system. Moreover, since Equivalent Capacity AC algorithms reserve the resources that are specified by the source traffic description, users could request more resources than they require leading to low network utilization.

The Diffusion based statistical AC uses statistical bandwidth based on closed-form expressions that use diffusion approximation models [9]. It exploits information about buffer sizes and the multiplexed traffic that shares a common link to obtain a diffusion approximation based cell loss estimate and assumes that traffic follows an "On-Off" behaviour. A new connection is admitted if the statistical bandwidth on every intermediate link along the selected path is less than the link capacity. The use of diffusion-based techniques has been shown to be conservative with respect to cell loss, but more economical in bandwidth allocation.

Endpoint Admission Control [10], [11], [12], [13], [14] is a measurement-based scheme in which the end host (endpoint) probes the network by sending probe packets at the data rate it would like to reserve and records the resulting level of packet losses, specially marked packets, or other QoS criteria. The host then admits the flow only if the loss (or marking) percentage is below some threshold value. Endpoint admission control requires no explicit support from the routers that do not need to keep a per-flow state or process reservation requests. This approach simply uses the fact that routers may drop or mark packets in a normal manner. In some cases the probe packets are treated equally with the data packets and in others they are sent at a different priority level. This approach suffers from the shortcomings of any measurement based scheme where estimates may not be in line with what will be observed when the real traffic is sent instead of the probe traffic.

III. Self Aware Networks

Self Aware Networks (SAN) is a proposal of QoS enabled networks with enhanced monitoring and self improvement capabilities. A promising SAN architecture is Cognitive Packet Network (CPN). CPN [1],[2],[15],[16],[17], a packet routing protocol which addresses QoS using adaptive techniques based on on-line measurements. In CPN, users declare their QoS requirements (QoS Goals) such as minimum delay, maximum bandwidth, minimum cost, etc.

CPN makes use of three types of packets: smart packets (SP) for discovery, source routed dumb packets (DP) to carry payload, and acknowledgements (ACK) to bring back information that has been discovered by SPs which are used in nodes to train neural networks. Conventional IP packets may tunnel through CPN to seamlessly operate mixed IP and CPN networks. SPs are generated either by a user requesting to create a path to some CPN node, or by a user requesting to discover parts of network state, including location of certain fixed or mobile nodes, power levels at nodes, topology, paths and their QoS metrics.

SPs discover routes by using random neural networks (RNN) [18] with reinforcement learning (RL). RL is carried out using a QoS Goal (such as packet delay, loss, hop count, jitter, etc) which is defined by the user who generated a request for the connection, or by the network itself. The decisional weights of a RNN are increased or decreased based on the observed success or failure of subsequent SPs to achieve the Goal. Thus RL will tend to prefer better routing schemes, more reliable access paths to data objects, and better QoS.

When a Smart Packet arrives to its destination, an ACK is generated and heads back to the source of the request, following the reversed path of the SP. It updates mailboxes (MBs) in the CPN nodes it visits with the information which has discovered, and provides the source node with the successful path to the node. All packets have a life-time constraint based on the number of nodes visited, to avoid overburdening the system with unsuccessful requests or packets which are in effect lost. A node in the CPN acts as a storage area for packets and mailboxes (MBs). It also stores and executes the code used to route smart packets. It has an input buffer for packets arriving from the input links, a set of mailboxes, and a set of output buffers which are associated with output links. The route brought back by an ACK is used as a source route by subsequent DPs of the same QoS class having the same destination, until a newer and/or better route is brought back by another ACK. ACK messages also contain timestamp information that can be used to monitor the QoS metrics on a single link and/or partial or complete paths.

IV. A MULTIPLE CRITERION AC ALGORITHM

The AC algorithm we propose is based on the ability of CPN to collect QoS information on all links and paths that the SPs have explored and on all paths that any user is using in the network. Furthermore, since it is the users that determine the QoS metrics that interest them, CPN collects data for the different QoS metrics that are relevant to the users themselves.

Thus, collectively, users may be concerned with m distinct QoS metrics $q_v \in R$, $v = 1, ...m$ that are specified in terms of QoS constraints $[q_v \in C_v(u)$ for each user u, where $C_v(u) \subset R]$ is typically an interval of acceptable values of the QoS metric v for user u. We will detail the AC algorithm in terms of forwarding packets from some source s to a destination d. However this approach can be generalised to the case where u is requesting some service S.

The starting point of our approach is that every QoS metric can be considered as a value which increases as the "traffic load" increases. Obviously, the addition of a new connection will increase the load of the paths it may be using, and therefore it is assumed that the value taken by the QoS metrics will increase. For example, delay increases as the network traffic load increases. Now consider some path π, and assume that a small increase x in the load that is obtained in a controlled manner, e.g. by sending probe packets at rate x, yields an estimate of the manner in which the QoS metric q varies around the current load point Y:

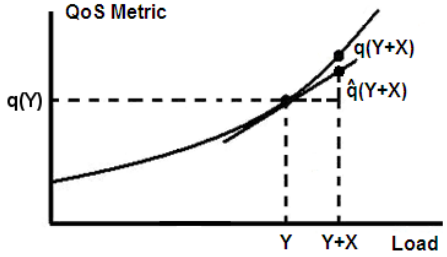

Fig. 1. Illustration of the variation of a QoS metric W as a function of load

$$\hat{q}' = \frac{q(Y+x) - q(Y)}{x}. \tag{1}$$

We use this estimate to evaluate the impact of a new flow with total traffic rate X by using the measured derivative from (1):

$$\hat{q}(Y+X) = q(Y) + \hat{q}'X, \tag{2}$$

without having to know the initial load Y. This estimate may be optimistic or pessimistic. However it is likely that the path that we will select because it provides the most favorable impact on current flows and because it satisfies the QoS needs of the new flow, is also likely to also be the best path in terms of actual observed QoS after the new user's full traffic is inserted.

Now consider the network graph $G(N, E)$ with nodes N, $n = |N|$, and the set E of directional links (i, j), where $i, j \in N$. The CPN algorithm explores $G(N, E)$ and collects QoS data about the parts of the network that are being currently used, or which have been explored by SPs. We assume that this data is available in one or more locations in the form of $n \times n$ *link QoS matrices* Q_v with elements:

- $Q_v(i, j) = \ r$ where $r \geq 0$ is a real number representing the QoS of link (i, j) which has been measured at some recent enough time, and
- $Q_v(i, j) = \ unknown$ if a SP has not explored the link for QoS metric v or if this happened so long ago that the value could be inaccurate.

From the link matrices Q_v we can compute:

- The set of known (explored) paths $P(i, j)$ from i to j, and
- The *path QoS matrices* K_v, where $K_v(i, j)$ is the *known best value* of the QoS metric v for any *path* going from i to j if such a path exists and if the links on the path have known entries in the link QoS matrices. Other entries in K_v are set to the value "unknown".

By "best value" we mean that several paths may exist for the source-destination pair (i, j), but $K_v(i, j)$ will store, for instance, the smallest known delay for all paths going from i

to j if q_v is the delay metric. We will discuss later how the path QoS matrices are computed from the link matrices.

The proposed AC algorithm proceeds as follows. Let u be a new user requesting admission for a connection from source s to destination d carrying a traffic rate X and with QoS constraint $q_v(u)$. Suppose also that the network is currently carrying other users z, any one of which will be generically represented by some QoS constraint $q_w(z)$.

- Find the set $P(s,d)$. If it is empty, send SPs to discover paths. If unsuccessful, reject the request. Otherwise find the path which corresponds to the best value of the QoS metric $K_v(s,d)$ and send probe traffic at rate x along that path.
- Use the probe traffic to obtain $\hat{q}'_w(i,j)$ for each QoS metric w of interest, including $w = v$, and for all concerned links (i,j). Note that most links will not be concerned by the probe traffic which only travels along a particular path so that for the unconcerned links we take $\hat{q}'_w(i,j) = 0$.
- Then compute

  $$\hat{Q}_w(i,j) = Q_w(i,j) + X\hat{q}'_w(i,j) \tag{3}$$

  for all concerned links and all QoS metrics. For unconcerned links we take $\hat{Q}_w(i,j) = Q_w(i,j)$.
- Compute $\hat{K}_w$ from $\hat{Q}_w$ in the same manner (to be detailed below) as we compute K_w from Q_w for all the QoS metrics of interest, including v.
- If $\hat{K}_v(s,d) \in C_v(u)$ AND $\hat{K}_w(s',d') \in C_w(z)$ for all other current users z with source-destination pair (s',d') and QoS metric q_v, then accept u; else reject the request.

A. Computing the QoS matrices

The well known "Warshall's algorithm" [19] determines for each $i,j \in N$ whether there is a path from node i to node j by computing the Boolean matrix P, the transitive closure of the graph's adjacency matrix A, in less than n^3 Boolean operations. Floyd's algorithm [20] extends Warshall's algorithm to obtain the cost of the "smallest cost path" between any pair of vertices in the form of a real-valued matrix.

The $n \times n$ QoS matrix Q_v has elements $Q_v(i,j)$ which either contain the currently known value of the $v-th$ QoS metric for link (i,j), or the value "unknown". An unknown value can be handled as a non-existent link until some new information is obtained. If the QoS metric q_v is additive, i.e. if the value of the metric over a path is the sum of the metrics over the links which compose the path, then we can construct the matrix K_v using the Floyd-Warshall algorithm so that $K_v(i,j)$ is the smallest value of the QoS metric among all known paths from i to j. Thus well known techniques can be used to construct K_v from Q_v, and hence $\hat{K}_v$ from $\hat{Q}_v$ if the metrics of interest are additive. Note that delay, and the variance of delay, are both additive metrics. Although loss rate is not additive (it is sub-additive in the sense that the path loss rate is smaller than the loss rate of individual links in the path), the number of lost packets is an additive metric. For non-additive metrics we have developed a generalisation of the Floyd-Warshall algorithm which we will not detail in this paper because of space limitations.

V. Experimental results

In this section we report some experimental results obtained with the AC algorithm that we propose using the test-bed of Figure 2. We have developed a two-tier user level application to test the AC algorithm, where the lower layer exploits the existing CPN capabilities in order to gather real time QoS metrics related to each link. The current implementation repeatedly collects values of the following QoS metrics: average delay, jitter, loss rate, and available bandwidth.

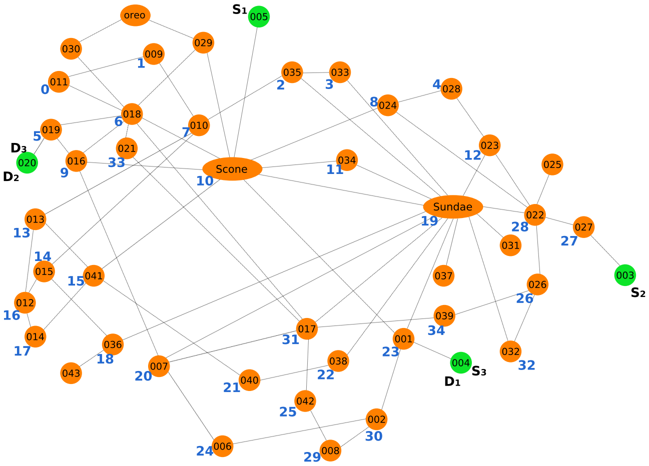

Fig. 2. The CPN testbed

Figure 2 shows a real testbed composed of 46 nodes with 100 Mbps links. Connection requests are made from three edge nodes, S_1, S_2, and S_3. The QoS requested by the users takes the form of the average delay which must not exceed 1.2 milliseconds, the delay's standard deviation which must not exceed 2.5 milliseconds, loss rate less than 5%, and path bandwidth that must exceed 10 Mbps. If a request is accepted by the AC algorithm, the source will generate a constant bit rate UDP traffic at 10 Mbps that lasts 150 seconds. After making a request, if the request is satisfied then the user will wait for a random time W and then make a request again. The same is true if its request is not satisfied. We set the random waiting time W among requests in order to have different rate for the arrivals. W is chosen to be uniformly distributed in a range of values which is $[0,120]$ seconds, $[0,90]$ seconds, and $[0,70]$ seconds, which coincides with an average arrival rate λ of 3, 4 or 5 requests per minute respectively. Thus the load on the system is increased in each of the successive experiments. Each experiment is repeated twice, first with the AC algorithm and then without. Figures 3,4,5 summarizes the experimental results. The left column contains the total acceptance rate for the connection requests, while the second column reports the satisfaction rate of a user (here user S_2) when the AC is enabled and disabled. By satisfaction of a user we mean that all four QoS requirements of that user are fulfilled. The acceptance rate when the AC is disabled is obviously 100%, but the consequence is seen in the second column. We observe that when the AC algorithm is enabled the percentage of times that the $S2$ user is satisfied is much higher with the AC than without.

VI. Conclusions

In this paper we have proposed an AC algorithm that uses measurements that are gathered by the CPN self-aware network architecture. The AC algorithm uses multiple QoS criteria and estimates whether a request for a connection should be accepted based on the incoming connection's QoS request and the impact it will have on ongoing connections. We report an implementation on a 46 node laboratory test-bed showing the effectiveness of AC with respect to user satisfaction.

Much further work can be done in this area. In a forthcoming paper we will present the algorithm in detail when non-additive QoS metrics are used and provide experimental results showing how the network's operation can be optimised from the viewpoint of its total traffic carrying capability in addition to the users' QoS needs. Further work is also planned concerning the application of this approach to peer-to-peer networks.

References

[1] E. Gelenbe, R. Lent, and A. Nunez, "Self-aware networks and QoS", *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1478-1489, Sep 2004.

[2] E. Gelenbe, Z. Xu, and E. Seref, "Cognitive packet networks", *Proceedings of the 11th International Conference on Tools with Artificial Intelligence*, pp. 47-54, Nov 1999.

[3] S. Jamin, S. J. Shenker, and P. B. Danzig, "Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service", *Proceedings of INFOCOM '97*, vol. 3, pp. 973-980, Apr 1997.

[4] R. Gibbens, F. Kelly, and P. Key, "A Decision-Theoretic Approach to Call Admission Control in ATM Networks", *IEEE Journal on Selected Areas of Communications*, vol. 13, no. 6, pp. 1101-1114, Aug 1995.

[5] D. Tse and M. Grossglauser, "Measurement-based Call Admission Control: Analysis and Simulation", *Proceedings of the INFOCOM '97*, pp. 981-989, Apr 1997.

[6] S. Jamin, P. B. Danzig, S. J. Shenker, and L. Zhang, "A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks", *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 56-70, Feb 1997.

[7] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-speed Networks", *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 7, pp. 968-981, Sep 1991.

[8] R. Guerin and L. Gun, "A Unified Approach to Bandwidth Allocation and Access control in Fast Packet-Switched Networks", *Proceeding of INFOCOM'92*, vol. 1, pp. 1-12, May 1992.

[9] E. Gelenbe, X. Mang, and R. Önvural, "Diffusion based statistical call admission control in ATM", *Performance Evaluation*, vol. 27-28, pp. 411-436, Oct 1996.

[10] L. Breslau, E. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint Admission Control: Architectural Issues and Performance", *Proceedings of ACM SIGCOMM 2000*, vol. 30, issue 4, pp. 57-69, Oct 2000.

[11] G. Bianchi, A. Capone, and C. Petrioli, "Throughput analysis of end-to-end measurement-based admission control in IP", *In Proceedings of IEEE INFOCOM 2000*, vol. 3, pp. 1461-1470, Mar. 2000.

[12] V. Elek, G. Karlsson, and R. Ronngren, "Admission control based on end-to-end measurements", *Proceedings of IEEE INFOCOM 2000*, vol. 2, pp. 623-630, Mar 2000.

[13] R. Gibbens and F. Kelly, "Distributed connection acceptance control for a connectionless network", *Proceedings of ITC 99*, vol. 2, pp. 941-52, Jun 1999.

[14] C. Cetinkaya and E. Knightly, "Egress admission control", *Proceedings of IEEE INFOCOM 2000*, vol. 1, pp. 1471-1480, Mar 2000.

[15] E. Gelenbe, R. Lent, and Z. Xu, "Design and performance of cognitive packet networks", *Performance Evaluation*, vol. 46, pp. 155-176, 2001.

[16] E. Gelenbe, R. Lent, and Z. Xu, "Measurement and performance of a cognitive packet network", *Journal of Computer Networks*, vol. 37, no. 6, pp. 691-701, Dec 2001.

[17] E. Gelenbe and P. Liu, "QoS and routing in the cognitive packet network", *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pp. 517-521, Jun 2005.

[18] E. Gelenbe, "Learning in the recurrent random neural network", *Neural Computation*, vol. 5(1), pp. 154164, 1993.

[19] S. Warshall, "A theorem on boolean matrices", *Journal of the ACM*, vol. 9, no. 1, pp. 11-12, 1962.

[20] R. W. Floyd, "Algorithm 97: Shortest path", *Communications of ACM*, vol. 5, no. 6, pp. 345, 1962.

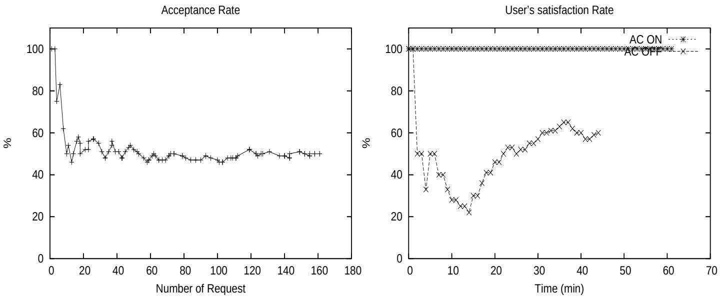

Fig. 3. Average waiting time W before new request is 60 secs (Average arrival rate $\lambda = 3\,requests/min$)

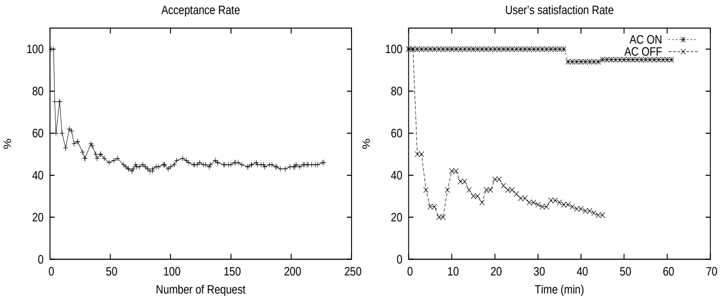

Fig. 4. Average waiting time W before new request is 45 secs (Average arrival rate $\lambda = 4\,requests/min$)

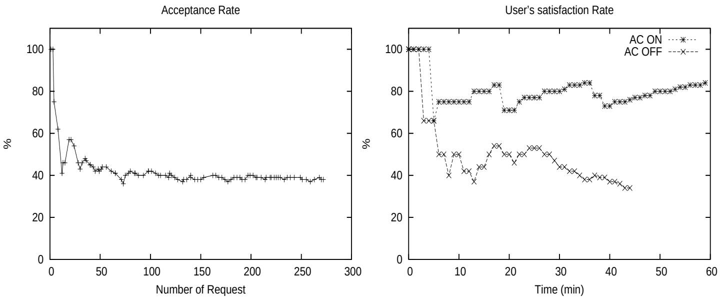

Fig. 5. Average waiting time W before new request is 35 secs (Average arrival rate $\lambda = 5\,requests/min$)