

COMPUTING SCIENCE

Energy-aware management of customer streams

Kanat Aidarov, Paul Ezhilchelvan and Isi Mitrani

TECHNICAL REPORT SERIES

No. CS-TR-1330

April 2012

Energy-aware management of customer streams

K. Aidarov, P. Ezhilchelvan and I. Mitrani

Abstract

Customers submit streams of jobs of different types for execution at a service centre. The number of jobs in each stream and the rate of their submission are specified. A service level agreement indicates the charge paid by the customer, the quality of service promised by the provider and the penalty to be paid by the latter if the QoS requirement is not met. To save energy, servers may be powered up and down dynamically. The objective is to maximize the revenues received while minimizing the penalties paid and the energy consumption costs of the servers used. To that end, heuristic policies are proposed for making decisions about stream admissions and server activation and deactivation. Those policies are motivated by queueing models. The results of several simulation experiments are described.

Bibliographical details

AIDAROV, K., EZHILCHELVAN, P., MITRANI, I.

Energy-aware management of customer streams
[By] K. Aidarov, P. Ezhilchelvan, I. Mitrani

Newcastle upon Tyne: Newcastle University: Computing Science, 2012.

(Newcastle University, Computing Science, Technical Report Series, No. CS-TR-1330)

Added entries

NEWCASTLE UNIVERSITY

Computing Science. Technical Report Series. CS-TR-1330

Abstract

Customers submit streams of jobs of different types for execution at a service center. The number of jobs in each stream and the rate of their submission are specified. A service level agreement indicates the charge paid by the customer, the quality of service promised by the provider and the penalty to be paid by the latter if the QoS requirement is not met. To save energy, servers may be powered up and down dynamically. The objective is to maximize the revenues received while minimizing the penalties paid and the energy consumption costs of the servers used. To that end, heuristic policies are proposed for making decisions about stream admissions and server activation and deactivation. Those policies are motivated by queueing models. The results of several simulation experiments are described.

About the authors

Kanat Aidarov is a member of teaching staff at Al-Farabi Kazakh National University, Almaty, Kazakhstan. He was a visiting member of staff at Newcastle in Feb-Apr 2012. His research interests include event processing, distributed systems and algorithms, performance evaluation and optimization; applied mathematics: numerical methods, Markov chains, control theory.

Paul Devadoss Ezhilchelvan received Ph.D. degree in computer science in 1989 from the University of Newcastle upon Tyne, United Kingdom. He received the Bachelor of Engineering degree in 1981 from the University of Madras, India, and the Master of Engineering degree in 1983 from the Indian Institute of Science, Bangalore. He joined the School of Computing Science of the University of Newcastle upon Tyne in 1983 where he is currently a Reader in Distributed Computing.

Isi Mitrani studied Mathematics at the Universities of Sofia and Moscow (Diploma, 1965), and Operations Research at the Technion, Haifa (MSc, 1967). He joined the University of Newcastle in 1968 as a Programming Advisor, became a Lecturer in 1969 (PhD, 1973), Reader in 1986 and Professor in 1991. He has held several visiting positions, including sabbatical years at INRIA (Le Chesnay) and Bell Laboratories (Murray Hill). His research interests are in the areas of Probabilistic Modelling, Performance Evaluation and Optimization. Publications include 4 authored books, 3 edited books, more than 100 journal and conference papers and one patent.

Suggested keywords

SERVICE PROVISIONING
ENERGY SAVING
CUSTOMER STREAMS
PROFIT MAXIMIZATION
QUEUEING MODELS

Energy-aware management of customer streams

K. Aidarov
Kazakh National University
Kazakhstan
kanat.aydarov@kaznu.kz

P. Ezhilchelvan, I. Mitrani
Newcastle University
Newcastle upon Tyne, UK
paul.ezhilchelvan@ncl.ac.uk

ABSTRACT

Customers submit streams of jobs of different types for execution at a service center. The number of jobs in each stream and the rate of their submission are specified. A service level agreement indicates the charge paid by the customer, the quality of service promised by the provider and the penalty to be paid by the latter if the QoS requirement is not met. To save energy, servers may be powered up and down dynamically. The objective is to maximize the revenues received while minimizing the penalties paid and the energy consumption costs of the servers used. To that end, heuristic policies are proposed for making decisions about stream admissions and server activation and deactivation. Those policies are motivated by queueing models. The results of several simulation experiments are described.

1. INTRODUCTION

This paper addresses an optimization problem arising in the market of computer services. A service provider employs a cluster of servers in order to offer a number of different services to a community of users. The users pay for having their jobs run, but demand in turn a certain quality of service. More precisely, a user wishes to submit a specified number of jobs of a given type, at a specified rate (jobs per second); such a collection is referred to as a 'stream'. There is a charge for running a stream (which may depend on the type), and a QoS guarantee on the part of the provider: the average waiting time of all jobs in the stream will not exceed a given bound. If that obligation is not met, the provider pays a specified penalty to the user.

In addition, there is an energy consumption cost associated with each running server. The provider thus faces a difficult trade-off: energy costs are minimized by keeping the number of servers powered up as low as possible, while revenues are maximized by using as many servers as possible in order to be able to accept incoming streams and avoid paying penalties. It is therefore desirable to employ dynamic policies for deciding when to power servers on and off and whether to

admit incoming streams or not. Moreover, those policies should react appropriately to changes in demand.

We design and evaluate stream admission and server allocation heuristics that aim to maximize the average profit obtained (revenues minus costs) per unit time. They are based on queueing models of system behaviour. Under reasonably realistic assumptions, those models are intractable and it is necessary to use approximations. That approach is justifiable because the policies that emerge yield significantly larger profits than the default policy of 'keep all available servers powered up and admit all streams'. The proposed heuristics are readily implementable and can be used in real systems.

There is an extensive literature on both server allocation and energy-saving topics. However, the particular problem considered here does not appear to have been studied before. Perhaps the most closely related work is by Mazzucco et al [9, 10, 11] and Mitrani [13]. The concepts of charge, obligation and penalty were defined in [9] and were applied to individual jobs (rather than to streams). The notion of a user stream was introduced in [10]. That paper examined the allocation of servers between different types of users, but did not consider the costs of providing the servers and the desirability of dynamically powering them up and down. The latter aspects were studied in [13] and [11], without addressing the economics of user streams and the admission policies associated with them.

More distantly related are works by Chase et al [3], Villela et al [14], Levy et al [7], and Liu et al [8], who consider various server allocation policies. Chandra et al [4], Kanodia and Knightly [6], Bennani and Menascé [2] and Chen et al [5] examine certain aspects of resource allocation and admission control in systems where the QoS criterion is related to waiting or response time. Those studies do not consider the issues associated with dynamic policies and profit maximization.

The system model and the associated service-level agreements are described in section 2. The mathematical analysis and the resulting heuristic policies for stream admissions and server activation/deactivation and are presented in section 3. Some simulation experiments where the heuristics are evaluated and compared to the default policy are reported in section 4. A summary and comments on future research directions are given in the conclusion.

2. THE MODEL

The provider has a cluster of N servers which can be used to serve user jobs. A user request is referred to as a ‘stream’; it consists of a number of jobs, submitted at a given rate over a period of time. These streams may be of m different types, with different demand characteristics. More precisely, a stream of type i ($i = 1, 2, \dots, m$) consists of k_i jobs, submitted at the rate λ_i jobs per second. If a stream is accepted, all jobs in it will be executed. The service times of type i jobs are i.i.d. random variables with mean and second moment b_i and $M_{2,i}$ respectively.

Streams of type i arrive at the rate of γ_i (the arrival instant of a stream is the moment when the first of its jobs is submitted). Hence, if there is no admissions policy and all incoming streams are accepted, the total offered load would be equal to

$$\rho = \sum_{i=1}^m \gamma_i k_i b_i . \quad (1)$$

In that case, the system would be stable if $\rho < N$. Of course, the presence of an admissions policy invalidates that requirement.

The *quality of service* experienced by an accepted stream of type i is measured by the observed average waiting time, W_i :

$$W_i = \frac{1}{k_i} \sum_{j=1}^{k_i} w_j , \quad (2)$$

where w_j is the waiting time of the j th job in the stream (the interval between its submission and the start of its service). One could also decide to measure the quality of service by the observed average response time, taking the job lengths into account.

N. B. It is worth emphasizing that the right-hand side of (2) is a random variable; its value depends on every job that belongs to the stream. Hence, even if all interarrival and service times are distributed exponentially, one would have to include quite a lot of past history into the state descriptor in order to make the process Markov. This remark explains why some of the approximations that follow are really unavoidable.

Each service-level agreement includes the following three clauses:

1. Charge: For each accepted stream of type i , the user shall pay a charge of r_i (this would normally depend on the number of jobs in the stream, k_i , their average service time b_i and submission rate, λ_i).
2. Obligation: The observed average waiting time, W_i , of an accepted stream of type i shall not exceed q_i .
3. Penalty: For each accepted stream of type i whose W_i exceeds q_i , the provider shall pay to the user a penalty of p_i .

So, in addition to their traffic characteristics, streams of type i have ‘economic parameters’, namely the triple (r_i, q_i, p_i) , $i = 1, 2, \dots, m$.

All jobs submitted by all active streams join a common FIFO queue and can be served by any of the currently operative servers. Each of the latter incurs a cost of c per unit time. The provider must make dynamic decisions about whether or not to accept incoming streams and how many servers to employ. More precisely, if at a stream arrival instant there are n operative servers, the following actions may be considered: reject the new stream; accept the new stream but do not power up any new servers; accept the new stream and power up one new server; \dots ; accept the new stream and power up $N - n$ new servers.

These possibilities will be labeled $-1, 0, 1, \dots, N - n$, respectively.

If, at a stream completion instant, there are n operative servers, the actions are: leave the servers as they are; power down one server; power down two servers; \dots ; power down n servers.

These possibilities are labeled $0, 1, \dots, n$, respectively.

The times taken to power servers up or down are assumed to be small compared to the lifetime of a stream and will be neglected. Also, it is assumed that the intervals between consecutive policy decision instants are large compared to individual job interarrival and service times. That is, enough jobs arrive and are served during such an interval to enable the system to be treated as having reached steady state.

The performance of the system is measured by the average profit, R , received per unit time. This is given by

$$R = \sum_{i=1}^m a_i [r_i - p_i P(W_i > q_i)] - cS , \quad (3)$$

where a_i is the average number of type i streams that are accepted into the system per unit time, $P(W_i > q_i)$ is the probability that the observed average waiting time of a type i stream, (2), exceeds the obligation q_i , and S is the average number of operative servers. The objective of the management policy is to maximize the value of R .

3. POLICIES

Consider the system state when a stream of type i arrives. Suppose that L_j streams of type j are currently active ($j = 1, 2, \dots, m$), and n servers are operative. If decision -1 is taken (i.e., the new stream is rejected), then nothing changes to affect the system’s future, so that decision can be assigned value 0.

On the other hand, if decision s is taken (i.e., the new stream is accepted and s new servers are powered up, $s = 0, 1, \dots, N - n$), then the following changes will occur: (a) L_i will increase by 1; (b) a revenue of r_i will be received; (c) a potential penalty of p_i may be payable; (d) the new servers will incur running costs. The value of (d) can be estimated by remarking that, in the absence of long waiting times, the lifetime of a stream of type i is roughly k_i/λ_i . Hence, the running costs of the new servers are approximately sck_i/λ_i .

To assess the value of (c), we need to estimate the probability that the average waiting time of the jobs in the new stream will exceed the obligation q_i , given that there will be $n + s$

operative servers. To do that, we model the system as a $GI/G/n + s$ queue with arrival rate, average service time and offered load given by

$$\lambda = \sum_{j=1}^m L_j \lambda_j, \quad b = \frac{1}{\lambda} \sum_{j=1}^m L_j \lambda_j b_j, \quad \rho = \lambda b \quad (4)$$

(remember that L_i has increased by 1). Note that the arrival rate and offered load appearing in (4) depend on the currently active streams. This is different from the long-term offered load in (1).

Although there is no exact solution for the average waiting time, β , in the $GI/G/n + s$ queue, an acceptable approximation is provided by an appropriate scaling of the corresponding $M/M/n + s$ result (see Whitt, [15]):

$$\beta = \frac{ca^2 + cs^2}{2} w_{M/M/n+s}, \quad (5)$$

where $w_{M/M/n+s}$ is the average waiting time in the Markovian $M/M/n + s$ queue with the above parameters, and ca_i^2 and cs_i^2 are the squared coefficients of variation of the inter-arrival intervals and service times, respectively. We shall approximate cs^2 by saying that a job starting service is of type i with probability λ_i/λ and averaging over the job types:

$$cs^2 = \frac{1}{\lambda b^2} \sum_{i=1}^m L_i \lambda_i M_{2,i} - 1. \quad (6)$$

The value of ca^2 will be taken as 1 (i.e., assume that the arrival processes of both streams and jobs within a stream are reasonably close to Poisson). That assumption is not essential, but if it is not made, some mechanism of estimating ca^2 would have to be provided.

The average waiting time in the $M/M/n+s$ queue is given by the well-known Erlang-C formula (or Erlang delay formula, e.g., see [12]).

When the system is heavily loaded, the waiting time in the $GI/G/n_i$ queue is approximately exponentially distributed (see [15]). Since the variance of the exponential distribution is equal to the mean, the waiting time variance can also be approximated by (5). Hence, the observed average waiting time of a stream of type i , which according to (2) involves the sum of k_i waiting times, can be treated as being approximately normally distributed with mean β and variance β/k_i . That approximation appeals to the central limit theorem and ignores the dependencies between individual waiting times.

Based on the normal approximation, the probability that the observed average waiting time, W_i , exceeds the obligation, q_i , can be estimated as

$$P(W_i > q_i) = 1 - \Phi\left(\frac{q_i - \beta}{\sqrt{\frac{\beta}{k_i}}}\right), \quad (7)$$

where β is given by (5), and $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution (mean 0 and variance 1). That function can be computed very accurately by means of a rational approximation (see [1]).

If $\rho \geq n + s$ (violating the stability condition), then it is natural to set $\beta = \infty$ and $P(W_i > q_i) = 1$.

The quality of the approximation (7) will depend on how well the implied assumptions are satisfied, namely the load is heavy and there is a large number of jobs per stream (the second of these conditions also ensures that any dependencies between the waiting times within a stream can be neglected). On the other hand, if the system is lightly loaded, then it is not so important to come up with a clever admission policy; all incoming streams would be admitted.

Thus, the value of decision s , $v(s)$, can be assessed as

$$v(s) = r_i - p_i P(W_i > q_i) - \frac{sck_i}{\lambda_i}. \quad (8)$$

The policy we propose to apply at arrival instances is to evaluate $v(s)$ for $s = 0, 1, \dots, N - n$, and if any of those values are positive, choose the decision that yields the largest value. If all are negative, choose decision -1. That policy will be referred to as the ‘Current State’ heuristic.

When a stream completes, a sensible policy is to power down as many servers as were powered up when that stream arrived.

The performance of the Current State heuristic will be compared against the ‘default’ policy which keeps all servers permanently powered up and accepts all incoming streams.

Although the computations involved in applying the Current State heuristic are by no means excessive, in some circumstances it may be desirable to avoid them. Therefore, we include in the comparisons a ‘Simple’ heuristic which ignores the possible penalties and just ensures that the number of active servers exceeds the current total offered load (including that of the new stream). In other words, the simple heuristic accepts the incoming stream if there is a non-negative integer s such that $n + s > \rho$; the number of new servers powered up is equal to the smallest such s . If that is impossible, the stream is rejected.

4. SIMULATION RESULTS

To compare the performance of the above policies, a number of simulation experiments were carried out. Lack of space allows us to show only two sets of results. In both cases, the profits achieved by the three policies in a system where the total number of servers is $N = 40$, are plotted against the stream arrival rate. Each point in the graphs corresponds to a simulation run where more than a million jobs arrive and are served. Each run is divided into 10 portions for the purpose of computing confidence intervals.

In the first experiment, all streams are of the same type. Each consists of 100 jobs, arriving at the rate of $\lambda = 0.9$ and requiring an average service time of $b = 1$, with squared coefficient of variation $cs^2 = 1$. The power consumption cost for one server is $c = 0.5$ per second. The revenue and penalty per stream are $r = p = 200$, while the obligation is $q = 1$ (in other words, if the average waiting time over the 100 jobs is larger than the average service time, the user will get his money back).

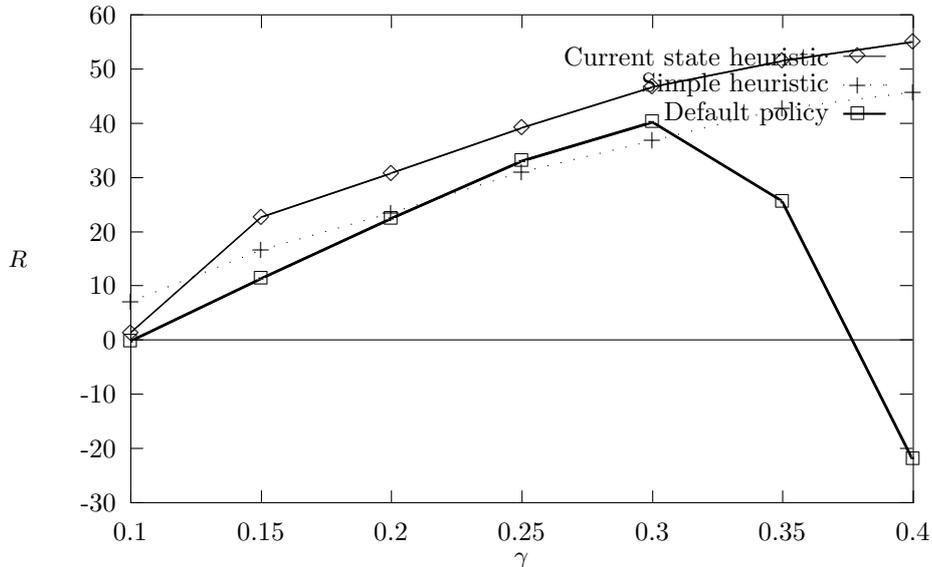


Figure 1: Policy comparison: single stream type

Note that, if all streams are accepted, the average offered load would be $100\gamma b$, so the system would saturate at $\gamma = 40/100 = 0.4$. We vary γ in the range (0.1,0.4).

Figure 1 shows that, in this case, the Current State heuristic produces consistently higher profits than the other policies. The simple heuristic also performs quite well, roughly keeping up with the Current State heuristic but yielding approximately 20% lower profits. The important point to note is that the profits of both these heuristics grow steadily with the offered load. On the other hand, the default policy yields negative profits at very low and very high loads. This is because in those situations the costs of running the servers and paying penalties are greater than the revenues received.

The confidence intervals for all three policies are largest at low traffic rates, where fewer jobs go through the system. However, for the vast majority of points, the half-width of the 90% confidence interval is less than 10% of the observed mean value; in many cases it is less 5%.

In the second experiment, the same 40 servers are available to serve two stream types, with the following parameters:

	Type 1	Type 2
k	100	100
λ	0.9	0.4
b	1	4
r	200	600
q	1	4
p	400	1200

Thus, jobs of type 2 are longer but are submitted less frequently than those of type 1. In both cases, the waiting time obligation is equal to the average service time. Type 1 streams pay 3 times higher charges; this time both penalties are twice as large as the corresponding charges. The running

cost for one server is again $c = 0.5$ per second.

The squared coefficient of variation, cs^2 , is now computed according to (6), with $M_{2,1} = 2b_1^2$ and $M_{2,2} = 2b_2^2$.

The arrival rates of both stream types are increased in a fixed proportion: 65% of all incoming streams are of type 1 and 35% are of type 2. If the overall stream arrival rate is γ , the offered load is $65\gamma + 35 \cdot 4\gamma = 205\gamma$. Hence, if all streams are accepted, the saturation point would be $\gamma = 40/205$.

Figure 2 compares the profits of the two heuristics and the default policy when γ is varied in the range (0.025,0.2).

Again we observe that both the Current State and the simple heuristic yield profits that increase with the offered load, while the default policy can produce highly negative profits at low and high loads. The profits of the simple heuristic are now within 15% or less of those of Current State.

5. CONCLUSIONS

We have examined the problem of maximizing profits in a service provisioning environment where demand comes in the form of customer streams and where QoS requirements and energy costs are conflicting factors. An approximate queueing analysis has enabled us to propose easily implementable heuristic policies for making intelligent decisions about stream admissions and server activations and deactivations. The Current State heuristic, in particular, yields significantly higher profits than the default policy of keeping all servers powered on and accepting all incoming streams. The simple heuristic is much easier to implement and also performs well.

It is clearly necessary to carry out a more extensive programme of experimentation, exploring the behaviour of the heuristics under different demand conditions and economic regimes. It would also be desirable to implement these

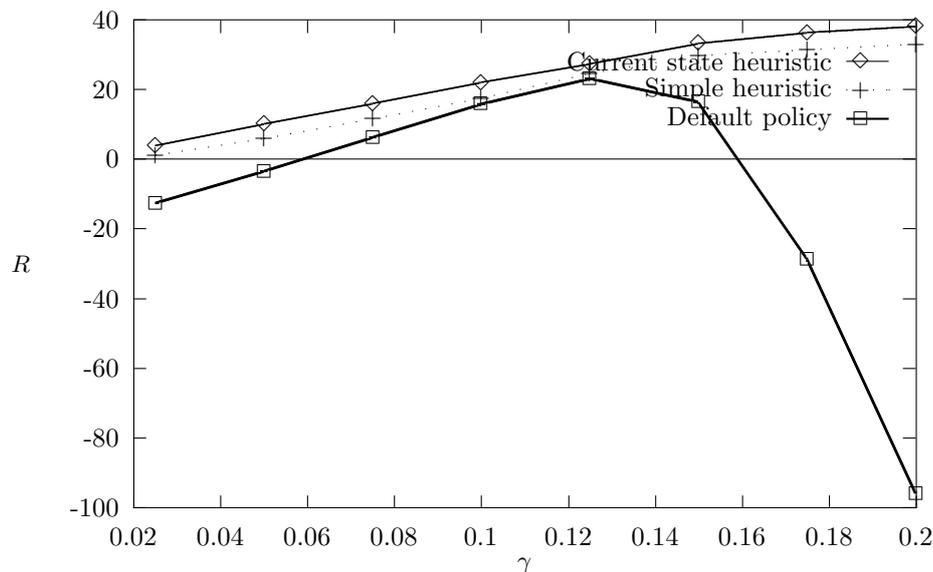


Figure 2: Policy comparison: two stream types

heuristics in a functioning system and observe their behaviour under real-life offered loads. In addition, one may wish to consider the unit costs of powering up servers.

A different operational model might dedicate servers to streams of a particular type. Then there would be a separate queue for each stream type. As well as deciding how many servers should be powered up, one would have to decide how many of the active servers should be allocated to each queue. That would also be an interesting topic of future research.

Acknowledgements

This work was carried out in the context of a collaboration between Kazakh National University and Newcastle University. Financial support was also provided by the RCUK project SIDE, which investigates the optimal use of computer clouds for stream processing.

6. REFERENCES

- [1] M. Abramowitz and I.A. Stegun, *Handbook of Mathematical Functions*, 10th printing, National Bureau of Standards, Washington, DC, 1972.
- [2] M.N. Bennani and D. Menascé, “Resource allocation for autonomic data centers using analytic performance methods”, Procs., 2nd IEEE Conf. on Autonomic Computing (ICAC-05), pp 229-240, 2005.
- [3] J.S. Chase, D.C. Anderson, P.N. Thakar, A.M. Vahdat and R.P. Doyle, “Managing energy and server resources in hosting centers”, Procs 18th ACM symposium on Operating systems principles, NY, 2001.
- [4] A. Chandra, W. Gong and P. Shenoy, “Dynamic resource allocation for shared data centers using online measurements”, Procs., 11th ACM/IEEE Int. Workshop on Quality of Service (IWQoS), pp 381-400, 2003.
- [5] X. Chen, P. Mohapatra and H. Chen, “An admission control scheme for predictable server response time for web accesses”, Procs., WWW10, Hong Kong, 2001.
- [6] V. Kanodia and E.W. Knightly, “Multi-class latency bounded web services”, Procs., 8th Int. Workshop on Quality of Service (IWQoS 2000), pp 231-239, 2000.
- [7] R. Levy, J. Nagarajarao, G. Pacifici, A. Spreitzer, A. Tantawi and A. Youssef, “Performance management for cluster based web services”, 8th Int. Symp. on Integrated Network Management, 2003.
- [8] Z. Liu, M. Squillante and J.L. Wolf, “On maximizing service-level-agreement profits”, Procs., EC’01, Tampa, 2001.
- [9] M. Mazzucco, I. Mitrani, J. Palmer, M. Fisher and P. McKee, “Web Service Hosting and Revenue Maximization”, Procs., 5th European Conf. on Web Services (ECOWS’07), Hale, 2007.
- [10] M. Mazzucco, I. Mitrani, M. Fisher and P. McKee, “Allocation and Admission Policies for Service Streams”, Procs. MASCOTS’08, Baltimore, pp 155-162, 2008.
- [11] M. Mazzucco, M. Vasar and M. Dumas, “Squeezing out the Cloud via Profit-Maximizing Resource Allocation Policies”, submitted for publication.
- [12] I. Mitrani, *Probabilistic Modelling*, Cambridge University Press, 1998.
- [13] I. Mitrani, “Managing Performance and Power Consumption in a Server Farm”, *Annals of Operations Research*, DOI:10.1007/s10479-011-0932-1, 2011.
- [14] D. Villela, P. Pradhan and D. Rubinstein, “Provisioning servers in the application tier for e-commerce systems”, *ACM Trans on Internet Technology*, Vol. 7, No. 1, 2007.
- [15] W. Whitt, “Approximations for the GI/G/m Queue”, *Production and Operations Management*, Vol. 2, No. 2, pp. 114-161, 1993.