AMO - Advanced Modeling and Optimization, Volume 15, Number 2, 2013

Limits of responsiveness for geographically remote knowledge bases: an operational analysis

Georgios Constantine Pentzaropoulos[§]
Mathematics and Information Technology Unit,
Department of Economics, University of Athens,
8 Pesmazoglou Street, 105 59 Athens, Greece.

Abstract

The purpose of this work is the analysis of responsiveness of remote knowledge bases (KBs). Responsiveness, *R(s)*, is seen here as a measure of service quality that deserves optimization. The chosen method is operational analysis, i.e. a variant of classical stochastic theory relying upon measurements over finite observation periods. The analysis incorporates service ability, elapsed time, and throughput. From these metrics, estimates of *R(s)* are derived analytically. Critical points indicating congestion are obtained, and a condition for efficient flow balance is also stated. A numerical example is the basis for discussing performance improvement. The results of this study should be of interest to experts responsible for managing knowledge resources across the Internet.

Keywords: Knowledge bases; workload; responsiveness; operational analysis; flow balance; decision rules; optimal resource management.

MSC (2010) Classification: 60K20, 68M20, 68T35.

1. Introduction

Knowledge bases are key elements of today's information economy and society [1]. The efficiency of knowledge bases (KBs) relies largely on the capabilities of high-speed telecommunications networks. Key features include broadband infrastructures, intelligent access forms, and improved quality of service. Both state authorities and Internet providers offer a wide range of services in a fully competitive environment. Such services include on-line searching in digital libraries, transfer of files and video, and forms of collaborative work such as multicast conferencing and decision-making across national borders. Developments have been strongly supported by regulation and standardization in many countries. In the European Union such developments are now part of EU's Digital Agenda [2]. Similar activities are taking place among most of the OECD member states [3].

[§] *Email: gcpent@econ.uoa.gr*
AMO - Advanced Modeling and Optimization. ISSN: 1841-4311

KBs can be classified as either machine-readable or human-readable. The former contain rules used for automated reasoning under variable conditions. The latter, which are examined here, are used for retrieval of knowledge via specific rules. Human-readable KBs are widely distributed across the world wide web and are typically accessed by search engines. These engines are able to mine data in KBs. Modern knowledge-based systems incorporate artificial intelligence techniques that aid in the decision-making procedures [4].

Knowledge bases are integral parts of today's web-based systems, which support a wide range of human activities. Of special interest today are web-learning support systems with advanced designs, frameworks and functions [5], as well as systems for automatic information discovery within the invisible world wide web [6]. The latter include advanced query routing mechanisms for locating the required information, extracting/integrating files and other forms of media, and then downloading the entire hidden web content.

KB-offered services vary widely. Yet, there are some common aspects which are often seen as problems from the users' viewpoint. Complaints are linked with poor performance as exemplified by long waiting times, connection failures, and so on. Thus, performance is often described as inadequate or unacceptable. With respect to KB management, typical problems are slow system response times, low productivity, and sometimes system saturation.

The purpose of this paper is the of study responsiveness when information is requested from distant KB hosts. The method consists of three elements: (i) an operational view of user-system communication, (ii) the introduction of a new performance measure called "*responsiveness to service requests"*, and (iii) the performance analysis of workflow. Responsiveness is obtained in closed form and its evaluation requires a few input data. Critical points indicating congestion are also obtained analytically. Finally, an illustrative example serves as a framework for discussing performance improvement.

2. Definitions

Let us consider Figure 1. The link shown contains a finite number of servers, denoted by S_1, S_2, S_3, ... , S_K, the last server being the KB host. Users occupy M workstations from which they generate their requests. At any time, a portion of M users, denoted by N, will be active while the rest M-N users will be in a thinking state.

The routing of requests follows the direction of the arrows. The communication system is assumed capable of servicing N requests with N constant during some period of usage. This assumption is realistic when the above is a period of *heavy demand;* then a backlog of requests is formed in the users' area. When a request completes service, the next in-line request is admitted via the input/output port: this action keeps N constant. Periods of heavy system usage are especially interesting from a management perspective.

From the above, it follows that the traffic at the I/O port should always be regulated on-line, and this is true with all protocols in use today. Such active regulation keeps *N* practically constant at a number called the *window size*. Later we explore a range of values for *N* and then find the maximum allowable value, when the system becomes congested and thus liable to performance saturation. Further, with reference to Figure 1, we define the following performance metrics:

(i) *Total service ability of system, σ(s)*: the sum of service times (s_i) of its servers.
(ii) *Elapsed time, E(N)*: time required for service completion including queueing delays at the nodes.
(iii) *Think time, T(u)*: user time measured from the instant of service completion to the issue of next request.

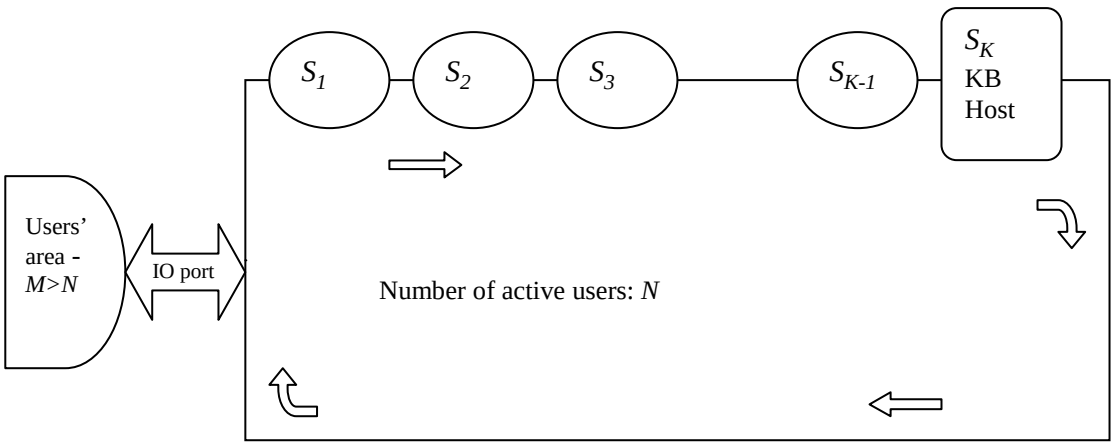

Figure 1: Cyclic queueing model of a communication system including a KB host.

The total service ability of the system, *σ(s)*, is by definition as follows:

$$\sigma(s) = (s_1 + s_2 + \cdots + s_K) \qquad (1)$$

From metrics (i), (ii) and (iii) defined above we introduce in this work the following composite performance measure: *Responsiveness to service requests, R(s) = {total service ability of the system} / {this quantity augmented by the corresponding mean elapsed time}*, i.e.:

$$R(s) = \sigma(s) / \{\sigma(s) + E(N)\} \qquad (2)$$

All measurements across the network of Figure 1 are assumed to take place during finite observation periods. The basic performance measures typically include event counters and timers. Time stamps are often used to indicate time intervals from requests originating from user terminals and arrivals at the KB host. More complex measures are derived from these, and are typically called operational quantities. Invariant relations among operational quantities that hold in any observation period of interest are called operational laws.

3. Operational analysis

A performance model has a central role in system evaluation: it gives estimates about the system's operation, and it can also provide management with performance predictions under alternative choices. When a capacity planning exercise or system redesign is due, such predictions can be of significant value.

3.1. *Modelling alternatives*

Several types of performance models have been developed over the years. Earlier models were mainly empirical, based on methods from statistics such as regression. Simulation has also been a favourite technique amongst many performance analysts. Analytical models based on *stochastic analysis* (queueing theory) were later thought to be a better alternative. They became widely acceptable, and remain so, because of the following reasons: (a) they are easier to construct and validate, (b) they can be realized using standard programming languages, and (c) results are given in closed-form expressions with clearer interpretations.

An interesting departure from classical stochastic theory is *operational analysis*. This method retains the basic form of stochastic analysis, including the advantages listed above. However, it is based upon a group of so-called operational rather than probabilistic laws. Operational analysis does not seek to replace analysis based on queueing theory, but to interpret previously known results using measured data. Many of the steady-state limit theorems of stochastic analysis have their equivalent operational laws. The operational interpretation was introduced roughly thirty-five years ago, when performance analysts realized that the formulas of stochastic queueing systems worked very well in real computer networks even though the traditional Markovian assumptions did not seem to hold in reality.

In their seminal paper, Denning and Buzen laid the foundations of operational analysis by describing the laws of this method [7]; these are as follows:

(i) All quantities should be defined so as to be *precicely measurable,* and all assumptions stated so as to be *directly testable.*

(ii) The system must be *flow-balanced,* i.e. the number of arrivals at a given device must be (almost) the same as the number of departures from that device during the observation period.

(iii) The devices must be *homogeneous*, i.e. the routing of jobs must be independent of local queue lengths at the devices, and the mean times between service completions must not depend on the queue lenghts of other devices.

These principles lead to mathematical equations that are equivalent to the traditional Markovian assumptions. Algorithms for computing performance measures of interest such as queue lengths, throughputs, and response times can be found in several books and edited volumes [8,9,10]. In this article, we introduce a form of operational analysis that allows for the study of user-system interaction while requiring minimal computational effort. This form is also useful for the examination of system transient behaviour during peak usage.

3.2. *Evaluation of responsiveness*

The total service ability of the system, $\sigma(s)$, can easily be evaluated from known server specifications. The main task is to estimate $E(N)$ in Equation (2). An exact estimation is possible using operational analysis; however, this procedure is iterative with each value of $E(N)$ depending on previous values. An acceptable approximation in closed-form might be preferable, as this would give a direct estimate of $E(N)$ for any value of N. Such an approximation can be obtained by taking into account the systems' maximum throughput, as explained below.

Let ρ_i be the utilization of server S_i and γ_i its mean throughput. In operational analysis, this utilization can be expressed as the product of the throughput and the corresponding service time, i.e. $\rho_i = \gamma_i \cdot s_i$. As the system takes larger values of N, the queues of requests in front of the servers will become longer. Eventually, there will be some instant when the slowest server will have to complete work at its maximum: at that instant, its utilization will have reached 100%. This saturated server then becomes a limiting factor or *bottleneck* of the system.

Let us denote by S_{max} the bottleneck server, by ρ_{max} its utilization and by γ_{max} the corresponding throughput. S_{max} is the slowest server because it has the largest mean service time, which we denote by:

$$s_{max} = \max \{ s_1, s_2, \ldots , s_K \} \qquad (3)$$

Then, as in the case of the mean values: $\rho_{max} = \gamma_{max} \cdot s_{max} = 1$. Therefore, $\gamma_{max} = 1/s_{max}$. Let $\gamma(N)$ be the mean system throughput, i.e. the mean rate (in seconds) at which requests leave server S_K with N requests present. Since γ_{max} is the maximum throughput anywhere in the system, it follows that the maximum value of $\gamma(N)$, denoted by γ^*, could be closely approximated by γ_{max}, i.e.:

$$\gamma^* \approx \gamma_{max} = 1/s_{max} \qquad (4)$$

A well-known property of operational analysis states that in a closed queueing system the number of "customers" being served is equal to the product of the system's throughput and the waiting time (with all quantities as means). This expression is an adaptation of Little's law from classical stochastic analysis.

In our notation, mean number of "customers" are the user requests N, the mean system throughput is $\gamma(N)$, and the mean waiting time is $E(N)$. Little's law will also apply when $\gamma(N)$ reaches its maximum, in which case: $N = \gamma^* \cdot E(N)$. Substituting γ^* as $\gamma^* \approx 1/s_{max}$ and solving for $E(N)$ this expression gives the following approximation:

$$E(N) \approx N \cdot s_{max} \tag{5}$$

After the above result, Equation (2) takes the following form:

$$R(s) \approx \sigma(s) / \{\sigma(s) + N \cdot s_{max}\} \tag{6}$$

Note that the equal sign in Equation (2) has become a near equal sign due to the approximation introduced previously. When the system is empty, i.e. $N = 0$, $R(s) \approx 1$: this is an ideal case (i.e. 100% responsiveness), which clearly cannot be expected in an operational system. When user activity increases from $N = 1$ onwards, then $R(s)$ decreases proportionately as in Equation (6). In the limiting case, i.e. when $N \to \infty$, which practically means that N has exceeded some critical point (to be determined), it follows that $R(s) \to 0$. This is seen by the users as inability to communicate with the distant KB host.

3.3. *Congestion and saturation*

When N=1, there is no contention. Therefore, $E(1)$ reduces to the sum $\sigma(s)$. Little's law implies that $\gamma(1) \cdot E(1) = 1$ or else $\gamma(1) = 1/\sigma(s)$. This is the minimum value of throughput. The corresponding maximum value was found approximately $1/s_{max}$. Therefore, the mean throughput is constrained as follows: $1/\sigma(s) \leq \gamma(N) \leq 1/s_{max}$. Thus, $\sigma(s) \cdot \gamma(N) \leq \sigma(s)/s_{max}$.

As already stated, $\sigma(s) \cdot \gamma(N)$ is the mean number of "customers" N; therefore, $N \leq \sigma(s)/s_{max}$. Then, as N increases, which means that user transactions (n) also follow, there will be some point, denoted by N^*, for which this inequality will eventually become an equality. Then, for such an $N = N^*$:

$$N^* \approx \lceil \sigma(s)/s_{max} \rceil \cdot n \tag{7}$$

The near-equal sign in Equation (7) comes from the approximation $\gamma^* \approx 1/s_{max}$ and the square brackets denote augmentation of the fraction to the next integer value. The quantity N^* is called here the *"internal critical point"* of the system. Admission of requests N with $N > N^*$ will increase congestion and eventually lead to saturation.

As previously discussed, requests for service are generated in the users' area. The mean throughput generated from *(M-N)* users with an average think time *T(u)* will be definition be *(M-N)/T(u)*. Consequently, the ratio:

$$\lambda_{in} = (M-N)/(T(u)\cdot n) \qquad (8)$$

is the mean input rate into the system. When N reaches its critical point N^*, its active part M will also reach a corresponding point M^*in the users' area. Then, the system will have reached its maximum throughput as in Equation (4). The corresponding maximum input, denoted by λ^*_{in}, will be $\lambda^*_{in} = (M^*-N^*)/T(u)\cdot n \approx 1/s_{max}$. Solution for M^* implies that:

$$M^* \approx N^* + \left[T(u)\cdot n / s_{max}\right] \qquad (9)$$

The square brackets denote augmentation of the fraction to the next integer. M^* could be interpreted as follows. For an internal (i.e. inside the system) critical point N^* there is a corresponding *"external critical point"* M^*, which inicates the number of users when saturation appears. This will be evident at the slowest server. Therefore, the pair (N^*, M^*) is an index of the system's ability to accommodate effectively its population of users.

4. Limits of responsiveness

Information retrieval with K servers is a good example of user-host communication. A link between a group of local users and their host may then be formed as suggested by Figure 1. Servers S_1, S_2, ... , S_{K-1} are used for the relay of user queries while S_K houses the distant KB. Service at S_K includes scheduling of user queries, searching in system files, and internal communications. For illustration, we assume a moderate value K = 15. Mean service times s_i (seconds) are in the interval (0, 1), a choice reflecting a practical range of server speeds. Table 1 below shows the relevant data.

The slowest server is S_7 with $s_{max} = s_7 = 0.965$ seconds. The system's total service ability is $\sigma(s)$ = 10.140 seconds. Application of Equation (6) for several successive values of N has given an array of results for *R(s)* which are shown in Table 2. Then, assuming a mean of n = 10 number of transactions per session, application of Equation (7) gives $N^* = 110$. Also, assuming that the average user needs *T(u)* = 15 seconds of think time, application of Equation (9) gives the value $M^* = 270$.

Table 1: Service times for the *K* servers of Figure 1.

Service	*Value (seconds)*				
s_1 ..... s_5	0.546	0.467	0.847	0.325	0.645
s_6 ..... s_{10}	0.835	0.965	0.628	0.617	0.564
s_{11} .... s_{15}	0.873	0.674	0.694	0.726	0.734

$s_{max} = s_7 = 0.965$ $\sigma(s) = 10.140$

Finally, the pair (110, 270) indicates the critical points of this example system. It may be concluded that the KB host should be allowed to serve up to 270 users of which 110 should be active at any time.

Table 2: Responsiveness to service requests as *N* increases.

N (users)	*R(s) (%)*				
1 ..... 5:	91.3	84.0	77.8	72.4	67.8
6 ..... 10:	63.7	60.0	56.8	53.9	51.3
11 ..... 15:	48.9	46.7	44.7	42.9	41.2

$N^* = 110$ $M^* = 270$ $R^*(s) = 8.75$

From Table 2 we note that for $N = 11$, responsiveness stands at $R(s) = 48.9\%$, i.e. at less than half of its ideal value. If more requests are allowed into the system, $R(s)$ will continue to decline moderately, as shown in Table 2 for N from 12 to 15. Going further up to the critical point $N^* = 110$, we note that $R(s)$ declines sharply and now stands at $R^*(s) = 8.75\%$. In this limiting case, the maximum number of users are all active thus bringing the whole system to saturation.

4.1. *Workflow balance*

In the example, the slowest server (S_7) is placed about halfway through S_1 and the KB host. If one attempts to eliminate the bottleneck at S_7 by replacing this server by a faster one, then a new bottleneck could appear elsewhere. This will probably be at S_{11} as this server is the second slowest after S_7 (Table 1). In fact, any cyclic system that is not properly regulated may contain several bottlenecks. Therefore, a global strategy will be needed to keep workflow unrestricted on an end-to-end basis.

The above may be seen as a problem of assigning input flows to servers in proportion to their service ability. Such an assignment must be accomplished dynamically for all servers. This is a fairly complex problem in its general form. Solution may be obtained by optimization techniques; but, even in that case, the solution will have to be adjusted periodically as the system workflow changes dynamically following user behaviour. Such behaviour is never known *a priori* and it might exhibit fluctuations that produce a high delay variance.

We will not address this complex problem here, since our aim is to keep the analysis as simple as possible. Readers may see, for instance, solutions based on scheduling processes and multi-criteria methods [11,12].

4.2. *KB host as the slowest server*

From Table 1, the following sequence is evident:

$$s_7 > s_{11} > s_3 > s_6 > s_{15} \qquad (10)$$

Therefore, S_7 is the slowest server, and the system's throughput is dominated by it. When $N = N^* = 110$, the system throughput is at its maximum, i.e.: $\gamma^* \approx 1/s_{max} = 1/s_7 = 1/0.965 = 1.036$. Let also ρ_7 be the utilization (fraction of busy time) of server S_7. From operational analysis, this quantity is the product of throughput and its mean service time: thus, $\rho_7 = \gamma^* \cdot s_7 = 1$. This indicates that server S_7 is *saturated,* which was expected to be so, since N has reached its critical point. By comparison, $\rho_K = \rho_{15} = \gamma^* \cdot s_{15} = 1.036*0.734 = 0.769$. Therefore:

$$\rho_K < 1 \qquad (11)$$

which indicates that the host S_K is in *steady-state.*

Let us now assume that S_K becomes the slowest server. This can be done by interchanging the values of s_7 and s_{15} in Table 1. Then, $\rho_K = 1$, which confirms that the host is now the saturated server. Our goal here is to arrive at a new utilization factor for S_K, say $\rho_{K(\text{new})}$, which would bring S_K back to its steady-state. To achieved that aim we will need to study the host in (partial) isolation from the rest of servers. The isolation is achieved by a technique known as *decomposition*. This technique is best known from Econometrics, where it has been applied in the analysis of large-scale systems with many variables. Later, it was applied successfully in the study of computer and communication systems. For a closed queueing network with K servers which models a large-scale system with a KB host, the above principle of decomposition may be stated as follows.

Consider server S_K and then replace the entire network by a two-server system only consisting of S_K and one "composite", flow-quivalent server, say S_e. In our example, this server is *flow-equivalent* to S_1, S_2, ... , S_{K-1} in the sense that the throughput of the {*K-1* system equals the arrival rate at S_K with the same number N}.

Assume that the decomposition principle is applied to the network of Figure 1. Note that the throughput of server S_e, representing the first K-1 servers, equals the arrival rate at S_K (host). Calling the first quantity $\{\gamma_e|(K\text{-}1)\}$ and the second one λ_K this gives: $\lambda_K = \{\gamma_e|(K\text{-}1)\}$. The mean value of the last quantity is not known *a priori.* We may use the arguments of the previous section to estimate its maximum value, say $\{\gamma_e^*|(K\text{-}1)\}$. This is done by observing that the slowest server in the (K-1) system is S_{11}, with $s_{11} = 0.873$. Then, as in the case of the entire system studied previously, we see that $\{\gamma_e^*|(K\text{-}1)\} \approx 1/s_{max(K\text{-}1)} = 1/s_{11} = 1/0.873 = 1.145$.

From this result, it follows that $\{\gamma_e|(K\text{-}1)\} \leq 1.145$. Let us assume for illustration that this unknown mean is about three-quarters of its maximum, i.e.: $\{\gamma_e|(K\text{-}1)\} \approx (3/4)\cdot\{\gamma_e^*|(K\text{-}1)\} \approx (3/4)*1.145 \approx 0.859$. Note that s_K is now 0.965, because of the interchange between s_7 and s_K, which makes the host the slowest server. From these results, the new utilization of S_K, say ρ_K(new), may be obtained as $\rho_K(\text{new}) = \lambda_K \cdot s_K = \{\gamma_e|(K\text{-}1)\}\cdot s_K = 0.859*0.965 = 0.829$.

We easily see that $\rho_K(\text{new}) = 0.829 < 1$, which indicates that the host is brought back to its *steady-state.* Therefore, the equation:

$$\lambda_K = \{\gamma_e|(K\text{-}1)\} \qquad (12)$$

(flow-equivalent communicating system)

is a sufficient operational condition for achieving *flow balance* between the KB host and system of the rest (K-1) servers.

5. Concluding remarks

The purpose of this work was the study of responsiveness in user communications involving remote knowledge bases (KBs). Analysis was carried out by a cyclic queueing model based on operational analysis. The performance measure introduced was named *"responsiveness to service requests"*. This was obtained in a simple, closed-form expression: its evaluation required only a few input data, and the calculations were direct, i.e. without any iterations.

The type of network introduced in this study can be viewed as a collection of users and servers. Each active user generates transactions, which are processed by the intermediate servers, until ultimately a response is returned to the user.

Critical points (N^*,M^*), which indicate system congestion, were obtained analytically. The performance of the KB host was also studied by the application of decomposition and flow-equivalent aggregation. Equation (12) is then a sufficient condition for achieving flow balance between the KB host and the rest K-1 servers.

Responsiveness was considered here as a measure of service quality: as such, $R(s)$ may also be seen as an *index of user satisfaction.* When this index is within its normal limits, users should appreciate the benefits from using the services offered by their host. When $R(s)$ declines, so does the picture of the communication system as seen by the its users. Limits on the number of admissible requests and on the number of connected users were previously considered necessary in order to avoid system congestion and possible saturation.

The example given illustrates that bottleneck analysis is a central issue when trying to forecast values of throughput and response times. The operational laws incorporated in the present work can easily be coupled with bottleneck analysis to offer a simple but powerful method for performance analysis. Finally, the operational laws lead to the creation of flow-balanced networks.

Equating the flow out of S_e with the flow into S_K ensures that no one dominates the other. The above results suggest the following rule of good practice for managing the KB host (S_K) effectively:

"*If the host is the slowest server, its input rate should be lowered until it matches the throughput of a composite server (S_e), which is flow-equivalent to the system of the rest (K-1) request-relaying servers*".

Acknowledgements

This work has been partially financed by grant no.: 70/4/4733, awarded by the Research Committee of the University of Athens, Greece. An earlier version of this article is available at: http://arxiv.org/abs/1007.0542.

References

[1] N.H. Rothberg & G.S. Erickson, *From Knowledge to Intelligence: Creating Competitive Advantage in the Next Economy* (Sage Publications, London, 2005).

[2] European Commission, *Digital Agenda: Commission Outlines Action Plan to Boost Europe's Prosperity and Well-Being,* (EC/IP/10/581, Brussels, 2010).

[3] Organisation for Economic Co-Operation and Development, *Information Technology Outlook,* (OECD Publishing, Paris, 2010).

[4] H. Yoshizumi, K. Hori & K. Aihara, The dynamic construction of knowledge-based systems, in: C.T. Leondes (Ed.), *Knowledge-Based Systems: Techniques and Applications* (Academic Press, San Diego, California, 2000), pp. 560-604.

[5] L. Fan, Web-based learning support systems, in: J.T. Yao (Ed.), *Web-Based Support Systems* (Springer, London, 2010), pp. 81-94.

[6] E. Sweeney, K. Curran & E. Xie, Automating information discovery in the invisible web, in: J.T. Yao, (Ed.), *Web-Based Support Systems*, (Springer, London, 2010), pp. 167-180.

[7] P.J. Denning & J.P. Buzen, Operational analysis of queueing networks, *ACM Computing Surveys* **10** (3), (1978), pp. 225-261.

[8] E.D. Lazowksa, John Zahorjan, G.S. Graham & K.C. Sevcik, *Quantitative System Performance* (Prentice-Hall, New Jersey, 1984).

[9] D. Menascé, V. Almeida & L. Dowdy, *Capacity Planning and Performance Modeling* (Prentice-Hall, New Jersey, 1994).

[10] C.G. Cassandras & S. Lafortune, *Introduction to Discrete Event Systems, Second Edition*, (Springer, New York, 2008).

[11] R. Chen & S. Meyn, Value iteration and optimization of multiclass queueing networks, *Queueing Systems,* **32** (1-3), (1999), pp. 65-97.

[12] D.I. Giokas & G.C. Pentzaropoulos, Efficient storage allocation for processing in backlog-controled queueing networks using multicriteria techniques, *European Journal of Operational Rerearch,* **124**, (2000), pp. 539-549.