*Proceedings of the 2008 Winter Simulation Conference*
*S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.*

# A NEW POLICY FOR THE SERVICE REQUEST ASSIGNMENT PROBLEM WITH MULTIPLE SEVERITY LEVEL, DUE DATE AND SLA PENALTY SERVICE REQUESTS

Anshul Sheopuri
Sai Zeng
Chitra Dorai

IBM T.J. Watson Research Center
19 Skyline Drive
Hawthorne, NY 10532, U.S.A.

## ABSTRACT

We study the problem of assigning multiple severity level service requests to agents in an agent pool. Each severity level is associated with a due date and a penalty, which is incurred if the service request is not resolved by the due date. Motivated by Van Meighem (2003), who shows the asymptotic optimality of the Generalized Longest Queue policy for the problem of minimizing the due date dependent expected delay costs when there is a single agent, we develop a class of Index-based policies that is a generalization of the Priority First-Come-First-Serve, Weighted Shortest Expected Processing Time and Generalized Longest Queue policy. In our simulation study of an assignment system of a large technology firm, the Index-based policy shows an improvement of 0-20 % over the Priority First-Come-First-Serve policy depending upon the load conditions.

## 1 INTRODUCTION

Several large companies employ Service Request Assignment Systems (SS) to support hardware and software issues faced by users. Such services are typically outsourced to IT service providers. The performance of the system is governed by a mutually agreed contract between the buyer and the service provider.

In our setting, the penalties espoused in the contract are as follows: If the request is not resolved by its due date, a (fixed) Service Level Agreement (SLA) penalty cost is incurred. Otherwise, no cost is incurred. The due date and penalty cost for a service request are determined by the severity level assigned to the request by a Subject Matter Expert (SME), who reviews it prior to the assignment decision.

We model the problem of service request assignment as a continuous time optimization problem. Since the problem is intractable, we develop near-optimal policies. Specifically, the policy answers the following two questions dynamically.

- If at least one agent is free and there are multiple service requests waiting to be served, which request should be assigned to one of the free agents?
- If all agents are busy and there is a waiting service request, which service request that is being served should be preempted, if at all?

The remainder of the section is organized as follows. Section 1.1 discusses the relevant literature. We then introduce the notation and formulate the problem in Section 1.2. In Section 1.3, we summarize our main results.

### 1.1 Literature Survey

The literature on modeling the performance of (SS) has mainly focused on addressing two questions: (1) staffing, i.e., how many agents should be staffed on a particular shift, and, (2) assignment, i.e., what policy should be followed to assign service requests to agents? Several researchers have addressed the question of staffing and assignment jointly (for example, (Gurvich et al. 2007), and (Bassamboo et al. 2004)). However, oftentimes, staffing decisions are tactical and cannot be implemented simultaneously with assignment decisions. Consequently, the bulk of the literature examines the staffing and assignment decisions independently. For example, a well-cited rule for staffing is the "Square root safety rule", which suggests keeping a square root of workload safety stock of agents, analogous to classical inventory models (Tijms 2003). Below, we focus on the literature related to assignment only as it is most relevant to our work.

Among the policies suggested for assignment, the First-Come-First-Serve (FCFS) is most common. This intuitive policy suggests that requests be assigned in the order in which they are received. In systems with service requests of multiple severity levels, Priority FCFS is a natural extension to the FCFS policy. In this policy, requests are assigned in the order that they are received, but with strict preference given to higher severity requests. In the sequel, we use FCFS and Priority FCFS interchangeably to mean severity level preference based assignment.

While the FCFS policy is intuitive, it does not consider the penalty costs, due dates, etc. that are seen in practice. Recently, researchers have developed policies with the objective of minimizing the costs stipulated in the contract. Van Meighem (1994) shows the asymptotic optimality of the Generalized cμ rule for convex delay costs and a single agent. According to this policy, service requests are assigned dynamically based on the product of the service rate and marginal cost at the age (or time in system) of service request.

Van Meighem (2003) studies costs which are a function of whether the job has resided in the system longer than its due date. He observes that this cost structure is intractable, and thus considers a sequence of continuous cost functions, which is discontinuous in the limit. He uses the Generalized cμ rule analysis to show that the Generalized Longest Queue (GLQ) policy is asymptotically optimal for this cost structure when there is a single agent. We will review this policy in more detail in the Section 2.

The literature on scheduling is related to our problem. For a review of the literature on scheduling, we refer the readers to Pinedo (1995). Here, we state one result that we will refer to later in the paper. For the problem of scheduling requests to minimize the weighted flow time with a single agent, Smith (1959) shows the optimality of the Weighted Shortest Processing Time (WSPT) policy. According to this policy, each service request is assigned a number, given by the product of the weight assigned to the request and the inverse of the processing time. The requests are then scheduled for service in descending order of the numbers assigned to them. An extension to this policy which schedules requests in the product of the weights and inverse of mean processing times (Weighted Shortest Expected Processing Times (WSEPT)) is known to be the optimal for the case of stochastic processing times for a single agent (Rothkopf 1966).

## 1.2 Notation and Problem Definition

Let the service requests comprise n levels of severity, denoted by 1, 2, …, n. We will use k to denote the severity level of a generic service request. Let $\mu_k$ be the average service rate, i.e., the inverse of the mean processing time of a severity k request. Let $D_k$ be the due date of severity k service requests. Let $c_k$ be the penalty cost incurred if a service request is not resolved within its due date $D_k$, i.e., the penalty cost incurred is zero if a severity k request is resolved in time $D_k$ and $c_k$ if it takes longer to resolve the request.

The contract between the buyer and service provider stipulate higher SLA cost penalties for the more important severity levels. Requests with higher penalty costs are also more difficult to solve and their mean service rates are thus lower. Consequently, without loss of generality, we assume that the penalty costs and service rates are ordered in severity levels as follows:

$$c_1 > c_2 > \ldots > c_n \text{ and } \mu_1 < \mu_2 < \ldots < \mu_n .$$

Before we formulate the problem, we formally define what we mean by a policy. A policy π is a dynamic decision rule to determine which service requests to assign to agents and which service requests to preempt, given the state of the system. In order to write the objective function, we need some additional notation. Let $S_k$ be the set of severity k service requests. Let $\tau_{ik}$ be the sojourn time of the $i^{th}$ request in the set $S_k$. The problem is to find the policy π that minimizes the penalty cost, i.e.,

$$\sum_{k \in \{1,2,\ldots,n\}} c_k \sum_{i \in S_k} 1\{D_k > \tau_{ik}\} .$$

## 1.3 Our Main Results

We develop a class of policies, which we call Index-based policies that is a generalization of three well-know policies: First-Come-First-Serve, Weighted Shortest Expected Processing Time and Generalized Longest Queue policy. Each of these polices is either optimal for a special case of our problem or a related problem.

We benchmark the performance of our policies through a simulation study. We simulate the service request assignment process of a large technology firm to evaluate the performance of our policies. The main insights of our study are:

- The cost of the best Index-based policy, within the class of index-based policies compared to the FCFS policy shows an improvement of 0-20% for the base case of problem instances in the dataset.
- Allowing preemption can add significant value to the business process. However, for certain load regimes, preempting service requests without considering elapsed service time can lead to poorer performance.
- The best Index-based policy perform better than the FCFS policy on a customer satisfaction metric that we develop.

The remainder of the paper is structured as follows. In Section 2, we derive our policy class, which we refer to as Index-based policies. In Section 3, we perform an extensive numerical investigation. We briefly discuss the implementation in Section 4 and discuss the scope for future research in Section 5.

## 2 INDEX-BASED POLICIES

Recall that any policy answers the following two questions dynamically.

- If at least one agent is free and there are multiple service requests waiting to be served, which request should be assigned to one of the free agents?
- If all agents are busy and there is a waiting service request, which service request that is being served should be preempted, if at all?

In the sequel, we propose a class of index-based policies to make the assignment decision. We then discuss the various preemption schemes that we consider.

To motivate our class of index-based policies, we first examine Van Meghem (2003) in detail. As stated earlier, he considers a (SS) with requests of multiple severity levels, due dates, SLA penalty costs and a single agent. Using results from Van Meighem (1994), he shows that a dynamic priority rule, which he refers to as the Generalized Longest Queue (GLQ) policy, is asymptotically optimal. The GLQ policy is FCFS within severity level and prioritizes the severity level with highest index at time t, defined as

$$I_k(t) = \frac{N_k(t)}{\lambda_k D_k},$$

where $N_k(t)$ is the number of severity k requests in the system at time t.

While the GLQ policy proposed by Van Meighem (2003) is asymptotically optimal for the case of a single agent, our setting is of multiple agents. We propose a new policy class with two parameters, x and y, which we call the index-based class of policies. For a given x and y, the policy is a modification of the GLQ policy with a SLA penalty cost and service rate term considered multiplicatively:

$$In_k(t) = (c_k)^x (\mu_k)^y \frac{N_k(t)}{\lambda_k D_k}.$$

We use $\pi(x,y)$ to denote an Index-based policy with parameters x and y. The index based policy is sensitive to SLA penalty costs and service rates, which allows us to show that this class encompasses the FCFS policy, the WSEPT policy and the GLQ policy. We state this result rigorously next.

**Theorem 1** *Suppose that the number of arrivals in [0, T] is bounded above by $M(T) < \infty$. Then, for the class of Index-based policies $\pi(x,y)$ operated during [0,T], the following statements hold:*

*1) For a given y, there exists $m_1(T) < \infty$ such that for $x > m_1(T)$, $\pi(x,y)$ is FCFS.*
*2) There exists $m_2(T) < \infty$ such that for $x = y > m_2(T)$, $\pi(x,y)$ is WSEPT.*
*3) For $x = y = 0$, $\pi(x,y)$ is GLQ.*

Proof: Omitted.

The best policy, within the class of Index-based policies, is computed by means of a search over the parameters x and y. We refer to this as the best Index-based policy. Since the index-based policy class contains the FCFS policy, the cost of the best Index-based policy is guaranteed to be better than the FCFS policy.

We next address the question of when to preempt a service request in case all agents are busy. We consider three preemption rules:

(P1) No preemption: Never preempt requests being serviced.
(P2) Partial preemption: Preempt the lowest index service request that is being served that has been served for less than the mean of the service time of its severity level if a higher index service request is waiting.
(P3) Full preemption: Preempt the lowest index service request that is being served if a higher index service request is waiting.

In case of preemption, we allow for work-saving, i.e., if a service request is preempted, an agent who is assigned this request later learns about the prior resolution attempts. This assumption is not unreasonable as agents document solutions that have been attempted. However, note that the insights of our results carry over to the no work-saving case as well.

## 3 COMPUTATIONAL RESULTS

To simulate a (SS), we used data from a large service provider. The dataset contains information about arrival times, service times, severity levels and due dates regarding 297 service requests. Since we do not know the optimal policy or the optimal cost, we benchmark the performance of our policy against the FCFS policy.

We first make an interesting observation. A histogram of the empirical distribution of the inter-arrival times of

Severity 1 level tickets clearly shows that the tail is long (histogram excluded for confidentiality reasons). Consequently, we tested whether the distribution of the inter-arrival times is sub-exponential Weibull using a chi-squared test. The null hypothesis that the distribution is Weibull with parameters k = 0.74 and λ = 54.61 was not rejected at 95 % statistical significance. The reader may note that this observation is in contrast with the assumption of Poisson arrivals that is common in the queueing literature. This underscores the value of a simulation-based approach to our problem.

In out computational experiments, we compare the best Index-based policy with the FCFS policy. The reader may note that the optimal value of the parameters x and y is computed by means of search. Further, as stated before, since the FCFS policy belongs to the Index-based policy class, the best Index-based policy is guaranteed to perform better than the FCFS policy on an expect cost penalty measure. Of course, evaluating the best Index-based policy through a search on the parameter space of x and y means that the best Index-based policy is computationally more intensive than the FCFS policy. However, it is worth adding that the computational effort required to determine the best Index-based policy may be controlled through coarseness of the search over the parameter space.

The remainder of the section is structured as follows. We first discuss the performance of the Index-based class of policies for the base-case of parameters in the dataset (The parameters have been normalized due to confidentiality concerns) in Section 3.1. We then discuss the value of preemption in the business process in Section 3.2. Section 3.3 discusses sensitivity analysis with respect to the service rates and the penalty costs. Finally, in Section 3.4, we discuss the performance of our policies on a Customer Satisfaction score that we develop.

## 3.1 Performance of Policies

We first test the performance of our policy for the problem instances of SLA penalty costs and mean service times in the data for various load conditions (or number of agents). The values of penalty costs and service rates for the base case are $c_1 = 100, c_2 = 5, c_3 = 2, c_4 = 1$ (USD) and $\mu_1 = 16000, \mu_2 = 4000, \mu_3 = 1000, \mu_4 = 5$ (in service requests / min). We benchmark the performance of our policy by comparing the cost that can be "affected" with that of the FCFS policy. We define this formally next.

Define the *Sunk cost* as the cost of service request violations corresponding to requests whose service time exceeds the due date. Intuitively, this cost corresponds to SLA penalty violations which are unavoidable, i.e., the cost incurred irrespective of the number of the agents that are staffed.

Since any policy that we propose cannot affect the sunk cost associated with SLA penalty violations, we compute the difference of the cost of SLA penalty violations and Sunk cost as the metric of performance of a policy. We define this to be the *Operating cost* of the policy. To benchmark the performance of the index policy, we compute the percentage improvement of the Operating cost of the index policy over the Operating cost of the FCFS policy.

We first summarize our results. For each problem instance of the number of agents, we compare the cost of the best preemption scheme with the Index-based policy and the best preemption scheme with the FCFS policy. We note that the percentage improvement is 0-20% depending upon the number of agents. The detailed results are provided in Table 1 in the Appendix.

Next, we discuss the performance of our policy compared to the FCFS policy as the number of agents is varied. We first note that the performance of both policies is similar when the number of agents is either small or large. The intuition behind this observation is as follows: When the number of agents is small, both policies primarily target reductions in severity 1 service requests. When the number of agents is large, some agents are "always free" and thus the policy need not be intelligent. This can be seen in Figure 1, which shows the percentage improvement in Operating Cost of the Index based policy against the FCFS policy as the number of agents is varied. Thus, we recommend using Index-based policies in medium load conditions.

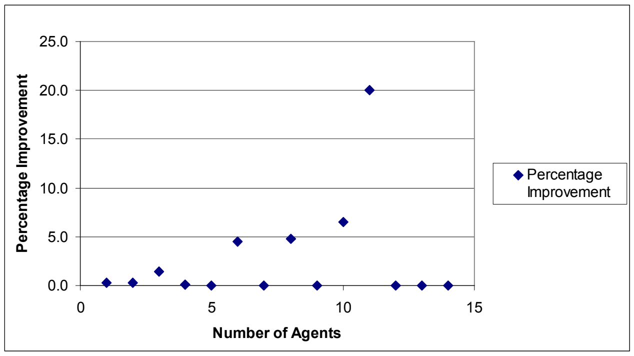

Figure 1: Comparison of Index-based policy with the FCFS policy

## 3.2 Value of Preemption

We now investigate the value of preemption in the business process. We compare the performance of the three preemption schemes that we consider: (1) No preemption, (2) Partial preemption and (3) Full preemption.

We make the following observations from Figure 2, which shows the performance of the Index-based policies for the three cases of preemption. Our first observation is that none of the policies dominate the other two for all load conditions. Secondly, when the number of agents is large, the performance under the three preemption schemes is identical.

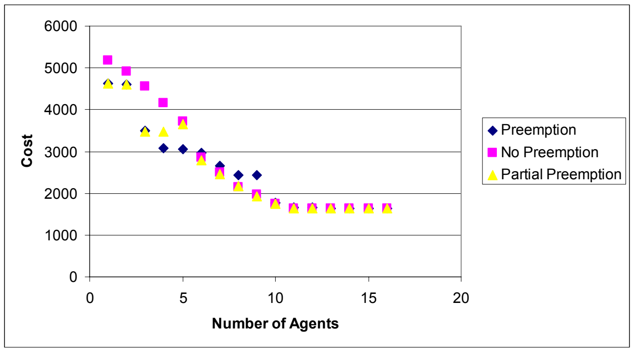

Figure 2: Value of preemption for Index-based policies

We summarize our recommendations of the preemption scheme that performs well for Index-based policies for different number of agents (see Table 2). The entries in the table may be interpreted as follows: A "X" in the ith row and jth column denotes that the jth preemption scheme performs well for the ith load condition.

Table 2 Recommendation for the different preemption schemes

| # Agents | No | Partial | Full |
|---|---|---|---|
| Low | | X | X |
| Medium | | | X |
| High | X | X | |
| Very High | X | X | X |

## 3.3 Sensitivity Analysis

Next, we consider the performance of the best Index-based policy against the FCFS policy when the penalty costs and service rates are varied. We expect that greater benefit can be derived from using the Index-based policy against the FCFS policy if the penalty costs are "similar" or the service rates are "dissimilar" across different severity levels, because in either case, there is a greater incentive to give assignment preference to a lower severity level. We make this notion rigorous below.

Let $\mathbf{x^i}$, i = 1, 2, 3 be n-dimensional vectors and let

$$\mathbf{x^i} = (x_1^i, x_2^i, \dots, x_n^i).$$

Definition: Let $2 \le k \le n$.

Let $x_j^1 = x_j^2 \forall j \in \{1,2,3,\dots,n\} \setminus k$. We say that the components of $\mathbf{x^2}$ are *less similar* than those of $\mathbf{x^1}$ (the components of $\mathbf{x^1}$ are *more similar* than those of $\mathbf{x^2}$ ) if

$$| x_k^1 - x_{k+1}^1 | \le | x_k^2 - x_{k+1}^2 |$$

The similarity of the components of a vector, when only component k is varied, is defined with respect to component k+1 since the Index-based policy would outperform the FCFS only by reducing the number of lower severity level violations. This intuition that the benefit from Index-based policies is higher when the SLA penalty costs are more similar is confirmed in the figure below (see Figure 3). A similar result may be obtained for less similar service rates (see Figure 4). Both Figures 3 and 4 are for the case of Full preemption, keeping all other parameters of the base case fixed.

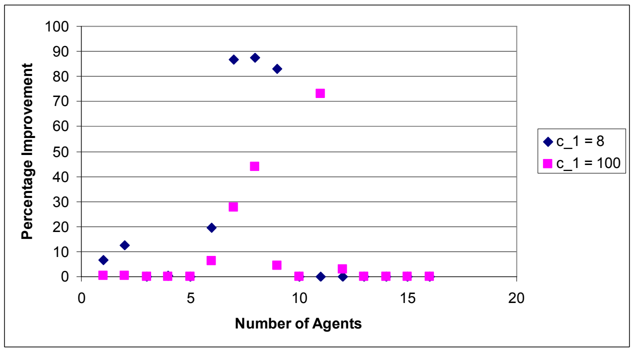

Figure 3: Sensitivity with Penalty Cost

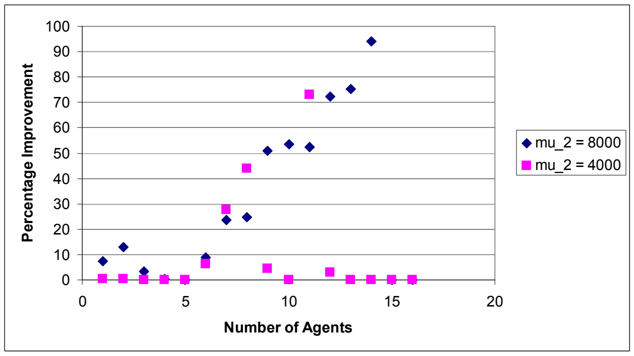

Figure 4: Sensitivity with Service Rates

## 3.4 Customer Satisfaction

The reader may note that the structure of the penalty function results in the following intuitive property of an optimal policy: "If a service request is not resolved by its due-date, it is optimal to never resolve it". This is a consequence of the fact that the contract does not penalize non-resolution of service requests. However, in several practical scenarios, there is a good-will cost associated with either not resolving a request or resolving it too late. Consequently, our policy does not have this property. Instead, we measure the impact of the goodwill cost by evaluating a customer-satisfaction metric.

We use the following Customer Satisfaction metric,

$$\sum_{k\in\{1,2,\ldots,n\}} c_k \sum_{i\in S_k} 1\{(1+\beta)D_k > \tau_{ik}\},$$

the total cost of SLA violations when the due date of each request is extended by a factor of $\beta > 0$. Clearly, one can develop other customer satisfaction metrics as well.

Our results show that the best Index-based policy has a significantly greater customer satisfaction metric that the FCFS policy for any number of agents for the case of $\beta = 0.25$ (see Figure 5). Consequently, we conclude that the Index-based policies do not result in a poorer customer satisfaction than the FCFS policy.

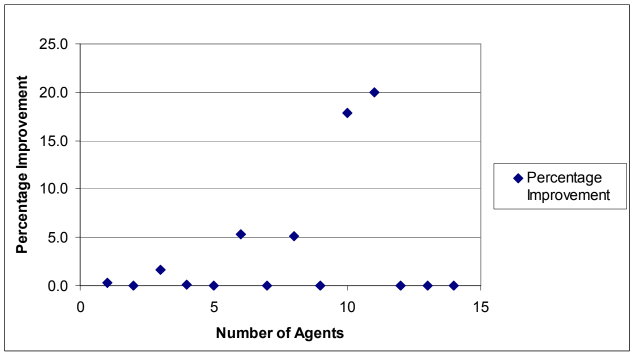

Figure 5: Customer Satisfaction of Index-based policy versus FCFS policy

## 4 IMPLEMENTATION

We now discuss an implementation of (SS). The implementation has two modules. The first module performs batch or static analysis to compute the optimal parameters of the Index-based policy at the end of every review period, say everyday. The second module takes these parameters as input from the static module to enable the dynamic real-time assignment decision during the next period, for example, the next day.

We now examine the implementation in detail. The historical service request information is maintained in the Operational Database 1, from where is extracted periodically using SQL scripts 2 (say everyday) to determine the optimal policy, within the class of index-based policies. The optimal values of the parameters x and y are fed into the assignment system 7, which is used to assign requests extracted using SQL scripts 6 extracted from the database 5 which is populated with daily arrivals at every small time increment. The assignment decision is then fed into database 8, which is then extracted by the agents (see Figure 6).

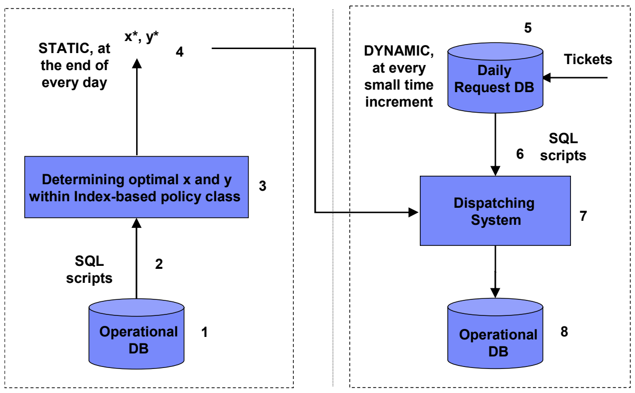
STATIC, at the end of every day
x*, y*
4
Determining optimal x and y within Index-based policy class
3
SQL scripts
2
Operational DB
1
DYNAMIC, at every small time increment
5
Daily Request DB
Tickets
SQL scripts
6
Dispatching System
7
Operational DB
8

Figure 6: An illustrative implementation.

## 5 CONCLUSION AND FUTURE RESEARCH

In this paper, we propose an algorithm to assign service requests to agents. The class of policies that we propose, which we refer to as index-based policies, is motivated by the GLQ policy proposed by Van Meighem (2003). The policy class contains the FCFS, WSEPT and GLQ policy. The average improvement of the Index-based policy over the FCFS policy is 0-20%, depending upon the load condition.

One drawback of our analysis is that we are unable to offer guidance on the optimality gap of our policies. Understanding the optimality gap would enable us to determine the scope for improvement from using more sophisticated policies than those proposed in this paper.

Though our model captures several dimensions of (SS), it has certain limitations as well. In particular, it would be interesting to capture the skill level of agents and use this information in the decision of assigning service requests to agents. Developing new policies that incorporate this information would definitely enrich the model and be of practical interest.

## ACKNOWLEDGEMENTS

The authors would like to thank Vijay Iyengar and Abhijit Bose, IBM T.J. Watson Research Center for several helpful suggestions relating to benchmarking and process insights respectively.

## APPENDIX

Table 1

| # Agents | Cost of Index-based Policy | | | Cost of FCFS Policy | | | Savings | % Savings |
|---|---|---|---|---|---|---|---|---|
| | No | Partial | Full | No | Partial | Full | | |
| 1 | 5186 | 4618 | 4618 | 5227 | 5127 | 4627 | 9 | 0.3 |
| 2 | 4912 | 4608 | 4608 | 4917 | 5117 | 4617 | 9 | 0.3 |
| 3 | 4561 | 3478 | 3503 | 4562 | 4464 | 3504 | 26 | 1.4 |
| 4 | 4153 | 3478 | 3073 | 4453 | 4464 | 3074 | 1 | 0.1 |
| 5 | 3723 | 3653 | 3057 | 3753 | 3653 | 3057 | 0 | 0.0 |
| 6 | 2851 | 2799 | 2967 | 3351 | 2851 | 3059 | 52 | 4.5 |
| 7 | 2512 | 2464 | 2662 | 2664 | 2464 | 3059 | 0 | 0.0 |
| 8 | 2153 | 2177 | 2432 | 2288 | 2178 | 3059 | 25 | 4.8 |
| 9 | 1974 | 1917 | 2432 | 2027 | 1917 | 2468 | 0 | 0.0 |
| 10 | 1754 | 1757 | 1762 | 1773 | 1765 | 1762 | 8 | 6.6 |
| 11 | 1642 | 1648 | 1667 | 1644 | 1765 | 1762 | 2 | 20.0 |
| 12 | 1632 | 1646 | 1667 | 1632 | 1646 | 1668 | 0 | 0.0 |
| 13 | 1632 | 1632 | 1632 | 1632 | 1632 | 1632 | 0 | 0.0 |
| 14 | 1632 | 1632 | 1632 | 1632 | 1632 | 1632 | 0 | 0.0 |

## REFERENCES

Basamboo J.M. Harrison and A. Zeevi 2004. Design and control of a large call center: Asymptotic analysis of an LP-based method" . Working Paper.

Gurvich I, Armony M., Mandelbaum A. 2008. Service Level Differentiation in Call Centers with Fully Flexible Servers. *Management Science*, forthcoming.

Van Meighem, J 1994. Dynamic scheduling with Convex delay Costs: The Generalized cm rule. *The Annals of Applied Probability*. **5**(3) 808-833.

Van Meighem, J. 2003. Due date Scheduling: Asymptotic Optimality of Generalized Longest Queue and Generalized Largest Delay Rules. *Operations Research* 51(1) 113-122.

Pinedo, M. 1995. Scheduling Theory, Algorithms and Systems. Prentice Hall.

Rothkopf, M. 1966. Scheduling with random service times. *Management Science* 12 703-713.

Smith, W.E. 1956. Various Optimizers for Single Stage Production. *Naval Research Logistics Quarterly* 3 59-66.

Henk C. Tijms. 2003. A First Course in Stochastic Models.

## AUTHOR BIOGRAPHIES

**ANSHUL SHEOPURI** is a Research Scientist with the Business Insight group at the IBM T.J. Watson Research Center. He is also an Assistant Adjunct Professor at the Leonard N. Stern School of Business, New York University. He has authored papers in conferences and journals such as M&SOM, Interfaces, etc. His current research interests include process optimization, inventory management and queueing theory. He received his PhD from the Leonard N. Stern School of Business, New York University in Operations Management and a B.Tech. in Mechanical and Industrial Engineering from the Indian Institute of Technology, Madras.

**SAI ZENG** is a Research Scientist at the IBM T.J. Watson Research Center. Her research interest is in Business Insight Generation, Product Quality Management, Systems Engineering, Product Lifecycle Management, Computer Aided Design and Computer Aided Engineering. More specifically, her expertise is focused on process improvement for Supplier Quality management, Process and System Collaboration and Integration Technologies, Predictive Analysis for Finance and Banking industry and associated Business Consulting Services offerings. Her research interests and past experience are also in reliability analysis modeling for product design, and product design-analysis integration. She has 20 journal/conference publications and 8 pending patents. She received the PhD degree in Mechanical Engineering, MS degree in Electrical and Computer Engineering from the Georgia Institute of Technology, MS and BS degrees in Mechanical Engineering from Huazhong University of Science and Technology, China.

**CHITRA DORAI** is a Senior Research Manager at the IBM T.J. Watson Research Center in New York. Her research interests are in the areas of cloud computing, distributed stream processing systems, e-learning media management, mobile imaging, multimedia content analysis, computer vision, pattern recognition, and machine learning. Chitra Dorai has published over 85 technical papers in premier IEEE and ACM conferences and journals, and has been granted multiple patents. She was the Editor of a book on Media Computing: Computational Media Aesthetics in 2002 and has contributed chapters to various books and edited collections in multimedia. She also served as the Associate Editor of IEEE Transactions on Multimedia and the Associate Editor of the Pattern Recognition Letters journal. She received the B. Tech. degree in Electrical Engineering from the Indian Institute of Technology, Madras, M.S. degree in Electrical Engineering from the Indian Institute of Science, Bangalore, and her Ph.D. from the Department of Computer Science at Michigan State University, with the Distinguished Academic Achievement Award. She is a senior member of the IEEE and a member of the ACM.