CASCON 2019 Proceedings

Sponsored By
IBM Centre for Advanced Studies
IBM Canada Lab

Edited By
Tima Pakfetrat - IBM Canada Ltd.
Guy-Vincent Jourdan – University of Ottawa
Kostas Kontogiannis - Western University
Robert Enenkel - IBM Canada Ltd.

Hilton Suites Toronto/Markham Conference Centre & Spa, Markham, Ontario, Canada November 04 - November 06, 2019

Full papers are reproduced here from camera-ready copies prepared by the authors. Permission has been granted to IBM Canada Ltd. and its related companies, and the Association for Computing Machinery, in each case without charge, to reproduce, distribute and publish in any medium or distribution technology

Table of Contents

Full Papers

Position Papers

Experience Report

Workshops

AI & Blockchain & Robotics

Cloud Computing

Compilers, Languages, Runtimes

Systems & Innovations

Message from the Conference Chair

CASCON x EVOKE 2019

Message from the Conference Chair

CASCON x EVOKE 2019

Welcome to CASCON 2019!

For the past 29 years, IBM Centre for Advanced Studies (CAS) has been hosting the Annual International Conference on Computer Science and Software Engineering. This is a testimony to our commitment to Academia and to the Applied Research in Canada. CASCON is well established yearly international meeting of minds that is unique in nature by being a purely academic conference sponsored by industry. The quality of papers, workshops, expo posters, and presentations showcased at CASCON is proof of the hard work of many academics and IBMers.

CASCON's theme follows the latest trends in Computer Science and Software Engineering, and this year is INTERDISCIPLINARY. INTERCONNECTED. INTEGRATED. These do sum-up the main challenges that technology faces these days. We see great success in industry and academia when technology is applied across fields and disciplines.

This year we are bringing forward a partnership with EVOKE, a successful developer conference, as a testimony to our commitment not only to academia and IBM development, but also to other developers that work across Canada and the Greater Toronto (GTA) Area. Through this partnership we are able to bring together the worlds of academia, research, development and every industry for a three-day marathon to discuss about research and technology, about interesting challenges but also about achievements and success stories. Indeed, CASCON attendees will experience a different CASCON, the same being true for EVOKE attendees who will be able to learn about technology trends, present papers, participate in workshops, and exhibit prototypes and solutions.

We have a full 3-day agenda with 10 parallel tracks where academic papers, developer talks, workshops, panels, and fireside chats will be presented to our combined audiences. The tracks are:

- AI, Blockchain & Robotics,
- Cloud Computing,
- Compilers, Languages & Runtimes
- Everything Data
- Extended Reality & Gaming
- IoT, Mobile & Smart Cities
- Privacy & Security
- Quantum Computing
- Software Engineering
- Systems & Innovation

In addition, we have 75 academic and industry posters in the expo area, where deep technical conversations happen. Finally, the main tent consists of a 3-day packed schedule that includes 6 keynotes, one competition, many panels and presentations by well recognized names in both industry and academia.

Overall, the CASCON x EVOKE 2019 event has a unique content consisting of 39 academic papers, over 71 developer talks, 30 workshops and 75 poster presentations. As with previous years, the proceedings, which include the technical papers, position papers, and detailed workshop abstracts, are also available online in the ACM Digital Library.

None of these would be possible without our dedicated community of Academics and IBMers and partners. As CASCON conference chair, I am fortunate to be immersed in an exceptional team of professionals that make a vision come true. I would like to start by thanking the Canada Lab Director, Steven Astorino for his thought leadership and his aspiration to create one of Canada's best developer and academic conferences. Because of him, the partnership with the Evoke event was possible, and as a result this has significantly increased our developer focused content as part of our event. As in the past years, special thanks to the Head of IBM CAS Canada, Mr. Marcellus Mindel, who's dedication to IBM, research and development is stronger than ever.

A strong academic conference has is also a reflection of a strong steering committee that guides the path that the conference is taking and provides critical feedback on the changes that the conference naturally has from year to year. Big thank you to the CASCON Steering Committee members (Prof. Guy-Vincent Jourdan, Prof. Hausi Müller, Mr. Joe Wigglesworth, Prof. Kelly Lyons, Prof. Ken Wong, Prof. Kenneth Kent, Mr. Marcellus Mindel, Prof. Marin Litoiu, Mrs. Tinny Ng, and Prof. Ying Zou).

An academic conference is not possible without the dedication that the Program Chairs commit prior to the conference day and is always reflected in the quality of papers that are accepted. Dr. Robert Enenkel from IBM and Prof. Kostas Kontogiannis from Western University have made a positive impact on the content this year, working tirelessly to orchestrate the paper submissions, revisions, and paper awards for our conference. Big thank you for the 62 Program Committee members who diligently peer-reviewed the papers and selected the top candidates; for the Best Paper Selection Committee and for the Most Influential Paper Award Committee. I would like to also thank Prof. Guy-Vincent Jourdan (Publication Chair) and Ms. Tima Pakfetrat (Conference Proceedings Editor) for taking care of our proceedings, and for ensuring that all content was filtered, approved, and published in the ACM Library.

For the second year in a row, we have the privilege to have Mrs. Tinny Ng and Prof. Hausi Müller acting as Workshop Co-Chairs. I would like to thank them both for preparing a rich program consisting of top workshops. I extend this thank you note to the Workshop Selection Committee members for reviewing the materials and making sure that the best workshops are accepted.

CASCON Technology Expo is the collaboration hub of the conference. With 75 technical exhibits and new content that is changing daily. This task was done under the leadership of our Expo co-chairs Prof. Gerhard Dueck and Mr. Mark Stoodley.

Special thanks to our IBM CAS Canada Team, Mr. Dennis Buttera, Mrs. Jennifer Collins, Ms. Maria Gallaher, Mrs. Tinny Ng, and for our exceptional group of interns Ms. Maxine Arbez Cheung, Mr. Sandy Bagga, Mr. Abdul El-Rahwan, Mr. Eric Lacey, Mr. Alexander Mah, and Ms. Tima Pakfetrat for all the heavy lifting that goes behind the scenes and often is unnoticed but without which nothing is possible. Thank you to Mr. Rodney D'Silva and Mr. Alan Heighway for taking care once more of all the CASCON network related tasks.

I would like to thank all the CASCON volunteers and Prof. Marin Litoiu (Volunteer Chair) and Ms. Gillian Cai (Volunteer Coordinator) for all the help during the conference.

The Evoke event is powered by a strong team that helped bring lots of attention and content to our joint event from GTA area developers. Many thanks to Mr. Patrick Kasebzarif (Executive Producer, Plastic Havas), Mr. Manreet Bains (Marketing Lead, Plastic Havas), Ms. Rovina Sigamoney (Creative Lead), Ms Shabnam Gharib (Program Lead, Plastic Havas) and Ms. Kathleen Bittle (VP Operations, Plastic Havas).

Finally, I would personally like to thank all the persons that submitted content to our conference and all our CAS Collaborators for promoting and contributing to this event, and last but not least, **a big thank you to all CASCON x EVOKE participants** for all the idea exchanges and brilliant discussions that happen during the conference.

I wish you all a wonderful and productive time at CASCON x EVOKE 2019!

Iosif-Viorel (Vio) Onut, Ph.D.,

Iosif Viorel Onut

Conference Chair | CASCON 2019
Principal R&D Strategist | Centre for Advanced Studies | IBM Canada
Lab Adjunct Professor | University of Ottawa

Message from the Conference Co-Chairs

CASCON x EVOKE 2019

Message from the Conference Co-Chairs

CASCON x EVOKE 2019

Welcome to CASCON x EVOKE 2019, the 29th Annual International Conference on Computer Science and Software Engineering hosted by the IBM Centre for Advanced Studies (CAS) and Plastic Havas!

The theme of CASCON x EVOKE 2019 is INTERDISCIPLINARY – INTERCONNECTED – INTEGRATED**.** This year we explore research challenges as well as economic and societal impacts of a wide variety of subject areas through 6 thought-provoking keynote presentations, 29 research paper and 10 position paper presentations, 30 workshops, and 75 poster and demo exhibits.

Our keynote presenters, who will enlighten the audience on different topics, are Ann Cavoukian, Distinguished Expert-in-Residence, Privacy & Data Analytics, Ryerson University; Bruce Croxon, Managing Partner, Round13 Capital; Robert Herjavec, Founder and CEO, Herjavec Group; Brad Micklea, VP and GM, Red Hat; Amber Simpson, Associate Professor, Queen's University; and Rob Thomas, GM, IBM Data & AI, IBM.

This year we received a total of 117 paper submissions, an increase of 29% from last year, from many different countries in North and South America, Europe, South and East Asia, and Africa. We accepted 29 full papers and 10 position papers, for an overall acceptance rate of 33%. Each paper was rigorously reviewed by three members of the 66-member Program Committee, ensuring a high-quality program. This year, the program is organized into 7 tracks: AI, Blockchain & Robotics; Cloud Computing; Everything Data; IoT, Mobile & Smart Cities; Privacy & Security; Software Engineering; and Systems & Innovations. As in previous years of CASCON, the CASCON x EVOKE 2019 proceedings are archived in the ACM Digital Library for ease of access.

The Technology Expo provides an excellent opportunity to experience emerging research results, leading-edge products, and developing product areas through 75 poster and demo exhibits.

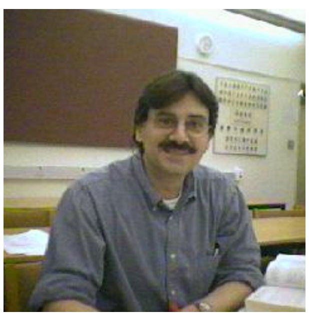
The 30 workshops are wonderful forums for presenting, discussing, and debating issues, problems, ideas, technology gaps, work-in-progress, and gaining hands-on experience with new technology and product directions.

A highlight of the conference planning process is the selection of the Best Paper, Best Student Paper, Most Influential Paper, and, new this year, New Ideas and Emerging Research Paper Award. The CASCON x EVOKE 2019 Best Paper Award goes to authors James You, Qi Zhang, Curtis D'Alves, Bill O'Farrell, and Christopher Anand for their paper, *Using z14 Fused Multiply-Add Instructions to Accelerate Elliptic Curve Cryptography*. The Best Student Paper Award goes to student authors Mouna Abidi and Manel Grichi for their paper, *Behind the Scenes: Developers' Perception of Multi-language Practices*, co-authored with their supervisor Foutse Khomh. The Most Influential Paper is awarded to authors Ye Hu, Johnny Wong, Gabriel Iszlai, and Marin Litoiu for their paper, *Resource Provisioning for Cloud Computing*, originally published in CASCON 2009. Finally, the New Ideas and Emerging Research Paper Award goes to authors Luis F.Rivera, Miguel Jiménez, Prashanti Angara, Norha M.Villegas, Gabriel Tamura, and Hausi A.Müller for their paper, *Towards Continuous Monitoring in Personalized Healthcare through Digital Twins*.

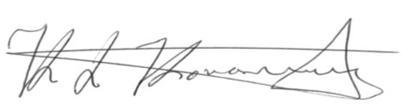
We are immensely grateful to the many people who helped and supported us in organizing CASCON x EVOKE 2019. We thank all the authors of technical papers, workshop proposals, and technology expo submissions. We thank the hard-working members of the Program Committee for their dedication to excellence in completing the reviews and engaging in online discussion of the papers. We also recognize the workshop and technology showcase committees, and the awards committees, for their important work. We thank the entire CASCON x EVOKE 2019 organizing team, including Hausi Müller and Tinny Ng, who coordinated the workshop selection and program; Gerhard Dueck and Mark Stoodley, who orchestrated the technology expo selection and program; Guy-Vincent Jourdan and Tima Pakfetrat, who assembled the proceedings; and Tinny Ng, who kept the conference website up to-date. Finally, we would like to thank the CASCON Steering Committee for their valuable support towards compiling this year's program.

We wish you a wonderful experience at CASCON x EVOKE 2019 and hope you will find time to enjoy the opportunities for networking in the stimulating social events.

Welcome to CASCON x EVOKE 2019!



Kostas Kontogiannis
Western University
CASCON x EVOKE 2019 Program Co-Chair



Robert Enenkel
IBM Canada
CASCON x EVOKE 2019 Program Co-Chair

Message from the Most Influential Paper of 2009 Award Committee Co-Chairs

CASCON x EVOKE 2019

Message from the Most influential Paper of 2009 Award Committee Co-Chairs

CASCON x EVOKE 2019

Since 2010, CASCON has presented a “Most Influential Paper” (MIP) Award to a paper published a decade earlier at CASCON in order to recognize the lasting contributions and impact of such paper to theory and practice.

Selecting the Most Influential Paper is a process that takes into account several factors. These factors include the impact the paper and its corresponding research had in the subject area, the evolution and significance of the topics discussed in the paper during the past decade, the consequent work spawned by the paper, and of course the number of citations as an indication of how the community perceived and used the work as a springboard for new initiating new research activities.

The CASCON x EVOKE 2019 Most Influential Paper of 2009 was selected by the MIP Selection Committee, consisting of Ettore Merlo, École Polytechnique de Montréal, Bill O’Farrell, IBM Canada, Joe Wigglesworth, IBM Canada, Robert Enenkel, IBM Canada (co-chair) and Kostas Kontogiannis, Western University (co-chair).

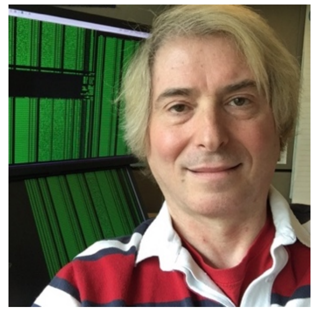
The committee followed a selection process similar to that established in previous years of CASCON. First, all CASCON 2009 papers were collected. Citation counts, types of citations, related work conducted during the past decade, and evolution and significance of the areas each paper related to, were parameters which were considered. The consequent analysis yielded a short list of papers which were considered by the Most Influential Paper review committee. Each member reviewed the short-listed papers and their associated bibliographic data according to criteria adapted from the ACM SIGSoft Impact Project and the Journal of the American Society for Information Science (JASIST) published by Wiley. Once the members of the committee analyzed the papers in the short list, they conferred to discuss and debate each candidate paper. It is to our pleasure to report that the selection of the Most Influential Paper to be presented in CASCON 2019 was unanimous.

This year’s Most Influential Paper Award goes to the paper titled “*Resource Provisioning for Cloud Computing*” by Ye Hu, Johnny Wong, Gabriel Iszlai and Marin Litoiu. The paper was the result of collaborative work between the University of Waterloo, York University, and IBM.

The winning paper addresses the important issue of how resources may best be allocated to an application mix in cloud computing, while meeting service level agreements. More specifically, it introduced an algorithm that determines an allocation strategy that suggests the smallest number of resources required, for applications to meet their Service Level Agreement requirements. Overall, the paper not only provided an innovative approach to this subject in 2009, but its approach is still timely and relevant even though the underlying virtualization technology evolved dramatically over the past decade. In this respect, we consider that the paper was visionary and paved the way for other researchers to work in the area of resource provision.

We congratulate the authors for their outstanding contribution, and we thank the MIP Award Committee for their reviews and deliberations.

Kostas Kontogiannis
Western University
CASCON x EVOKE 2019 Program Co-Chair



Robert Enenkel
IBM Canada
CASCON x EVOKE 2019 Program Co-Chair

Organizing Committee

CASCON x EVOKE 2019

Organizing Committee

Conference Chair

Iosif Viorel Onut	IBM Canada Ltd.

Conference Program Co-Chair

Kostas Kontogiannis	Western University
Robert Enenkel	IBM Canada Ltd.

Workshop Co-Chair

Hausi Müller	University of Victoria
Tinny Ng	IBM Canada Ltd.

Expo Co-Chair

Gerhard Dueck	University of New Brunswick
Mark Stoodley	IBM Canada Ltd.

Finance and Registration Chair

Marcellus Mindel	IBM Canada Ltd.

Website Chair

Tinny Ng	IBM Canada Ltd.

IT and Network Chair

Alan Heighway	IBM Canada Ltd.
Rodney D'Silvia	IBM Canada Ltd.

Volunteer Chair

Marin Litoiu	York University

Publication Chair

Guy-Vincent Jourdan	University of Ottawa

Conference Proceedings Editor

Tima Pakfetrat	IBM Canada Ltd.

Executive Producer

Patrick Kasebzarif	Plastic Havas

VP, Operations

Kathleen Bittle	Plastic Havas

Program Lead

Shabnam Gharib	Evoke Canada

Marketing Lead

Manreet Bains	Evoke Canada

Creative Lead

Rovina Sigamoney	Plastic Havas

Steering Committee

Marin Litoiu	York University
Kelly Lyons	University of Toronto
Marcellus Mindel	IBM Canada Ltd.
Hausi Müller	University of Victoria
Tinny Ng	IBM Canada Ltd.
Iosif Viorel Onut	IBM Canada Ltd.
Joe Wigglesworth	IBM Canada Ltd.
Ken Wong	University of Alberta
Ying Zou	Queen's University
Guy-Vincent Jourdan	University of Ottawa
Kenneth Kent	University of New Brunswick

Program Committee

Akihiro Hayashi	Rice University
Alexander Chatzigeorgiou	University of Macedonia
Alexei Lapouchnian	University of Toronto

Andrew Craik	IBM Canada Ltd.
Andriy Miranskyy	Ryerson University
Arie Gurfinkel	University of Waterloo
Bill O'Farrell	IBM Canada Ltd.
Biruk Habtemariam	IBM Canada Ltd.
Calisto Zuzarte	IBM Canada Ltd.
Chanchal Roy	University of Saskatchewan
Christopher Anand	McMaster University
Diwakar Krishnamurthy	University of Calgary
Ebrahim Bagheri	Ryerson University
Ettore Merlo	École Polytechnique de Montréal
Farhana Zulkernine	Queen's University
Foutse Khomh	École Polytechnique de Montréal
Frank Dehne	Carleton University
Fred Popowich	Simon Fraser University
Gabriel Silberman	Barcelona Institute of Science and Technology
Gerhard Dueck	University of New Brunswick
Ghizlane El Boussaidi	École de technologie supérieure
Grace Lewis	Carnegie Mellon Software Engineering Institute
Guy-Vincent Jourdan	University of Ottawa
Hadi Hemmati	University of Calgary
Hanan Lutfiyya	University of Western Ontario
Hausi Müller	University of Victoria
Herna Viktor	University of Ottawa
Hugh Leather	The University of Edinburgh
James Green Jeremy	Carleton University
Bradbury	University of Ontario Institute of Technology
Jin Li	PointClickCare
Joe Wigglesworth Jose	IBM Canada Ltd.
Nelson Amaral	University of Alberta
Juergen Dingel	Queen's University
Juergen Rilling	Concordia University

Kelly Lyon	University of Toronto
Kenneth Kent	University of New Brunswick
Ladan Tahvildari	University of Waterloo
Maleknaz Nayebi	École Polytechnique de Montréal
Manos Papagelis	York University
Marc Moreno Maza	University of Western Ontario
Marin Litoiu	York University
Marios Fokaefs	École Polytechnique de Montréal
Mark Chignell	University of Toronto
Mark Stoodley	IBM Canada Ltd.
Michael Smit	Dalhousie University
Michalis Famelis	Université de Montréal
Mohammed Sayagh	École Polytechnique de Montréal
Morteza Zihayat	Ryerson University
Norha Villegas	Universidad Icesi
Panos Patros	University of Waikato
Parke Godfrey	York University
Paul Ward	University of Waterloo
Ramiro Liscano	IBM Canada Ltd.
Renato De Mori	McGill University
Sudhakar Ganti	University of Victoria
Suprio Ray	University of New Brunswick
Tse-Hsun Peter Chen	Concordia University
Ulrike Stege	University of Victoria
Weiyi Shang	Concordia University
Wolfram Kahl	McMaster University
Ying Zou	Queen's University

Workshop Selection Committee

Calisto Zuzarte	IBM Canada Ltd.
Dorina Petriu	Carleton University
Gennady Pekhimenko	University of Toronto
Hausi Müller	University of Victoria
Joanna Kubasta	IBM Canada Ltd.
Kishor Patil	IBM Canada Ltd.
Marin Litoiu	York University
Suprio Ray	University of New Brunswick

Tinny Ng	IBM Canada Ltd.
Vince Pasquantonio	IBM Canada Ltd.
Ying Zou	Queen's University

MIP Selection Committee

Ettore Merlo	École Polytechnique de Montréal
Bill O'Farrell	IBM Canada Ltd.
Joe Wigglesworth	IBM Canada Ltd.
Robert Enenkel	IBM Canada Ltd.
Kostas Kontogiannis	Western University

Most Influential Paper of 2009

CASCON x EVOKE 2019

Resource Provisioning for Cloud Computing

Ye Hu[1], Johnny Wong[1], Gabriel Iszlai[2] and Marin Litoiu[3]

[1]University of Waterloo, [2]IBM Toronto Lab, [3]York University

Abstract

In resource provisioning for cloud computing, an important issue is how resources may be allocated to an application mix such that the service level agreements (SLAs) of all applications are met. A performance model with two interactive job classes is used to determine the smallest number of servers required to meet the SLAs of both classes. For each class, the SLA is specified by the relationship: Prob [response time $\leq x$] $\geq y$. Two server allocation strategies are considered: shared allocation (SA) and dedicated allocation (DA). For the case of FCFS scheduling, analytic results for response time distribution are used to develop a heuristic algorithm that determines an allocation strategy (SA or DA) that requires the smallest number of servers. The effectiveness of this algorithm is evaluated over a range of operating conditions. The performance of SA with non-FCFS scheduling is also investigated. Among the scheduling disciplines considered, a new discipline called probability dependent priority is found to have the best performance in terms of requiring the smallest number of servers.

1 Introduction

To meet the increasing demand for computing resources, the size and complexity of today's data centers are growing rapidly. At the same time, cloud computing infrastructures are becoming more popular. An immediate question is how the resources in a cloud computing infrastructure may be managed in a cost-effective manner. Static resource allocation based on peak demand is not cost-effective because of poor resource utilization during off-peak periods. In contrast, autonomic resource management could lead to efficient resource utilization and fast response in the presence changing workloads. This paper is concerned with resource allocation strategies that are relevant to autonomic resource management.

The two-level resource management architecture presented in [1] provides a framework for our investigation. At the lower level, there are multiple application environments (AEs). Each AE consists of a set of computing resources that are shared by one or more applications. At the higher level, a global arbiter performs resource allocation across AEs.

In this paper, we consider the processing of interactive jobs only. These jobs generally have small processing requirements and require good response time performance. The SLAs under consideration are based on the probability distribution of response time, namely, Prob [response time $\leq x$] $\geq y$ where x is a threshold value and y is the target probability. Our approach is to use performance models to obtain results that can be used to guide resource allocation decisions.

In our investigation, the computing resources at each AE are modeled by servers. When the global arbiter makes resource allocation decisions, information on the number of servers that should be allocated to each AE would be very helpful. This corresponds to the smallest number of servers required to meet the SLAs of all applications that are assigned to the AE.

Jobs processed by an AE are classified according to their workloads and SLAs. One or more applications may be included in the same class. The number of servers required is affected

Copyright © 2009 Ye Hu, Johnny Wong, Marin Litoiu and IBM Canada Ltd. Permission to copy is hereby granted provided the original copyright notice is reproduced in copies made.

by the resource allocation strategy and job scheduling discipline within the AE. The allocation strategies under consideration are shared allocation (SA) and dedicated allocation (DA). In SA, the servers are shared by all job classes. DA, on the other hand, allocates to each job class a fixed number of servers; these servers are not available to the other classes. As to job scheduling, the disciplines considered include first-come first-served (FCFS) and two priority disciplines where job classes with more demanding SLAs are given higher priority.

In [2], a multi-server queueing model was used to show that SA is superior to DA with respect to mean response time over all jobs. However, the issue of SLA was not included in the investigation. When SLAs are considered, SA may not be the better strategy under all operating conditions.

In general, a cloud computing infrastructure [3, 4] may provide service to a large number of job classes. Results on the performance difference between SA and DA for an arbitrary number of classes are difficult to obtain. This is because of the potentially large number of possible allocation strategies that need to be evaluated. Additional complexity is introduced when the impact of scheduling discipline is included in the investigation. To keep the complexity at a manageable level, we consider the special case of two job classes. In spite of this simplification, our results are directly applicable when the global arbiter, taking into consideration issues such as application isolation, management and security, decides to use a divide-and-conquer approach in which an AE contains at most two job classes. In addition, our results provide valuable insights into the performance of alternative resource allocation strategies and job scheduling disciplines, and can be used to develop heuristic methods for resource allocation when more than two classes are assigned to an AE [5].

Our investigation includes (i) a comparative evaluation of SA and DA under FCFS scheduling; (ii) a heuristic algorithm that determines a resource allocation strategy (SA or DA) that results in the smallest number of servers required to meet the SLA of both classes; and (iii) a comparative evaluation of FCFS, head-of-the-line priority (HOL) [2] and a new scheduling discipline called probability dependent priority (PDP).

The remainder of this paper is organized as follows. Our performance model is described in Section 2. Section 3 presents results on the merits of SA and DA under FCFS. A heuristic algorithm to select the preferred resource allocation strategy under FCFS is also developed and evaluated. The impact of priority scheduling on performance is investigated in Section 4. Related work is discussed in Section 5. Finally, Section 6 contains some concluding remarks.

2 Performance Model

In our performance model, computing resources at each AE are modeled by servers. There are two job classes; each has its own workload and SLA. With two job classes, the number of AEs is either 1 or 2 and the corresponding resource allocation strategies are SA or DA. Our models for SA and DA are shown in Figures 1 and 2. For SA, job arrivals from the two classes are combined into a single stream and served by a pool of m servers. For DA, each job class has its own dedicated pool of servers, and we use m_1 and m_2 to denote to number of servers allocated to class 1 and class 2, respectively.

We assume that for class i (i = 1, 2), the job arrival process is Poisson with rate λ_i and the service time distribution of both classes is exponential with mean $1/\mu$. As mentioned earlier, the SLA is based on the relationship Prob [response time $\leq x$] $\geq y$. We use SLA(x, y) to denote such an SLA.

We assume that for DA, jobs are processed in FCFS order. A number of scheduling disciplines are considered for SA, namely FCFS, HOL, and PDP.

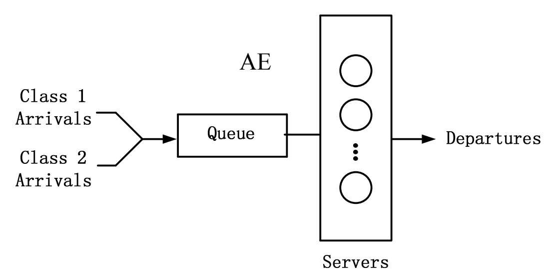


Figure 1: Shared Allocation

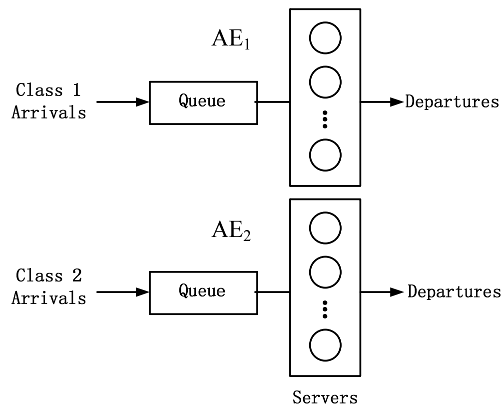


Figure 2: Dedicated Allocation

3 Resource Provisioning

3.1 Analytic Results for FCFS

Under DA, the model for each job class can be viewed as an M/M/m model with FCFS scheduling. The same model is also applicable when FCFS is used in SA. For this model, analytic results for the response time distribution are available in [6]. Let $F(x)$ be the cumulative distribution function (CDF) of response time, i.e., $F(x)$ = Prob [response time $\le x$]. In [6], it was shown that:

$$F(x) = P(0)\left[\sum_{n=0}^{m-1}(1-e^{-\mu x})\frac{\rho^n}{n!} + \frac{m\rho^m}{m!(1-m+\rho)}\left(\frac{1-e^{-(m-\rho)\mu x}}{m-\rho} - 1 + e^{-\mu x}\right)\right] \quad (1)$$

where $P(0) = (\sum_{n=0}^{m-1}\frac{\rho^n}{n!} + \frac{m\rho^m}{m!(m-\rho)})^{-1}$

is the probability that the system is empty, $\rho = \lambda/\mu$ is the traffic intensity, and λ and μ are the arrival rate and service rate, respectively. Note that $m > \rho$, otherwise the system does not have sufficient capacity to handle the load.

3.2 Allocation Strategies

Consider first DA. The results in Equation (1) can be used to determine m_{D1} and m_{D2}, the smallest number of servers required to meet the SLA of class 1 and class 2, respectively. For m_{Di} (i = 1, 2), the value of ρ in Equation (1) is given by λ_i/μ.

An algorithm that determines the smallest number of servers required is included as Algorithm 1 below. This algorithm starts with $m = \lfloor\rho\rfloor + 1$ and increases m until the target probability y is achieved. Let SLA_i be the SLA of class i (i = 1, 2). m_{Di} can be obtained by setting the arrival rate to λ_i, the service rate to μ, and SLA(x, y) to SLA_i.

Let m_D be the smallest number of servers required under DA to meet the SLA of both classes. m_D is given by:

$$m_D = m_{D1} + m_{D2} \quad (2)$$

Algorithm 1

Input:	σ	// Arrival rate
	μ	// Service rate
	SLA(x, y)	//Service level agreement
Output:	m	// Minimum number of Servers required

```
1:  m = ⌊σ/μ + 1⌋
2:  while F(x) < y, m++
3:  return m
```

Consider next SA. Under FCFS, the CDF of response time can be obtained by extending the work in [7] to the case of multiple servers. The resulting CDF is the same as that for the M/M/m – FCFS model with arrival rate equal to $\lambda_1 + \lambda_2$, i.e., the aggregated rate of the two classes. Furthermore, both classes have the same CDF of response time which is given by Equation (1) with $\rho = (\lambda_1 + \lambda_2)/\mu$.

Let m_{si} be the smallest number of servers required under SA to meet SLA_i $(i = 1, 2)$. m_{si} can be obtained from Algorithm 1 by setting the arrival rate to $\lambda_1 + \lambda_2$, the service rate to μ, and SLA(x, y) to SLA_i. m_S, the smallest number of servers required to meet the SLA of both classes, is then given by:

$$m_S = \max(m_{S1}, m_{S2}) \quad (3)$$

3.3 SA and DA Comparison

In this section, we use numerical examples to evaluate the performance difference of DA and SA under FCFS scheduling. The input parameters considered are shown in Table 1, where λ_i is the arrival rate of class i, and x_i and y_i are parameters of SLA_i, representing the response time threshold and target probability, respectively. We restrict the values of λ_1 and λ_2 such that $\lambda_1 + \lambda_2 \le K = 40$. We feel that this represents a sufficiently wide range of workload. The service rate μ is set to 1.

λ_i	0.1, 0.2, … , 40.0
x_i	2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0
y_i	0.8, 0.85, 0.9, 0.96

Table 1: Parameter values

Our evaluation is based on the total number of servers required to meet the SLA of both classes, as given by m_D and m_S in Equations (2) and (3), respectively. For each combination of λ_i, x_i and y_i (i = 1, 2), SA (or DA) is superior if $m_s < m_D$ (or $m_D < m_S$). Our results show that when both classes have the same SLA, SA always performs better than, or has the same performance as, DA. However, when SLA_1 and SLA_2 are different, neither SA nor DA is superior for all combinations of parameter values. For example, the results for the two cases shown in Table 2 indicate that DA is superior for case 1, but SA is superior for case 2.

Case	λ_1	x_1, y_1	λ_2	x_2, y_2	m_D	m_S
1	0.6	3, 0.8	3.0	5, 0.95	5	6
2	0.6	3, 0.95	3.6	5, 0.8	8	7

Table 2: Two Example Cases

Our goal is to develop an efficient algorithm that determines the preferred allocation strategy (DA or SA) for given values of λ_i, x_i and y_i (i = 1, 2). This would facilitate resource management because the preferred strategy requires the smallest number of servers to meet the SLA of both classes.

3.4 Heuristic Algorithm

To develop our algorithm, we first remove the dependency of the preferred allocation strategy on SLA_1 and SLA_2 by using a metric that represents their difference. We then characterize, for a given value of the difference metric, the dependency of the preferred strategy on the arrival rates λ_1 and λ_2. The results are used to develop a heuristic algorithm that determines the preferred strategy.

3.4.1 SLA Difference

We note that for a given SLA, different arrival rates could result in different number of servers required. In Figure 3, we plot the smallest number of servers required m against the arrival rate λ for a pair of SLAs. We observe that the value of m for SLA(3, 0.95) is always larger than or equal to that for SLA(5, 0.8). Through extensive testing involving other SLA pairs, the following pattern is observed. Let $m(\lambda, SLA)$ be the smallest number of servers required for the given λ and SLA. For any pair of SLAs, either

$$m(\lambda_1, SLA_1) \geq m(\lambda_2, SLA_2) \quad \text{or}$$
$$m(\lambda_1, SLA_1) \leq m(\lambda_2, SLA_2)$$

over the range of values of λ considered (which is $0 < \lambda \leq 40$). This pattern led us to use a single metric to describe the difference in m for a pair of SLAs.

Let $G(SLA)$ be the average number of servers required to meet the given SLA over the range of arrival rates considered. $G(SLA)$ is given by:

$$G(SLA) = \frac{1}{K}\int_0^K m(x, SLA)dx \qquad (4)$$

where $K = 40$. $G(SLA)$ can be computed numerically. We define a metric called "SLA Difference" between SLA_1 and SLA_2 (denoted by D) as follows:

$$D = |G(SLA_1) - G(SLA_2)| \qquad (5)$$

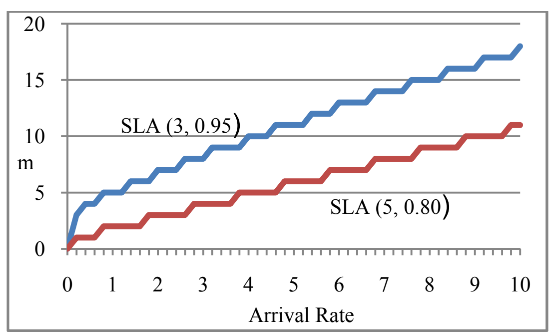


Figure 3: Smallest Number of Servers Required

3.4.2 Dependency on Arrival Rates

We now present results that show the impact of D, λ_1 and λ_2 on the merits of SA and DA. Consider the two scenarios summarized in Table 3. The SLA pair for scenario 1 is not the same as that for scenario 2, but the SLA difference of the two scenarios are almost the same (equal to 22.6). The results for these two scenarios are shown in

Figures 4 and 5, respectively. For each combination of λ_1 and λ_2, the corresponding intersection is grey if DA is the better strategy, and white if SA is better or as good as DA. We observe similar patterns of grey and white for both scenarios 1 and 2. Let f_D be the fraction of intersections that are grey (i.e., DA is better). Our results indicate that for both scenarios, f_D is approximately 5.2%.

A similar observation is made in Figures 6 and 7 where we consider two scenarios that have larger SLA differences (see Table 4). For these scenarios, the SLA difference D is 83.45 and the resulting f_D is increased to about 64%.

Through extensive testing, it was found that the above observation is true for scenarios where the SLA differences are very close to each other. We also observe that f_D tends to increase with D. Based on these results, we conclude that SLA difference is potentially useful in our effort to develop a heuristic algorithm that determines the preferred strategy.

Scenario	SLA_1	SLA_2	D
1	SLA(4.5, 0.85)	SLA(2.5, 0.9)	22.58
2	SLA(4, 0.8)	SLA(2.5, 0.9)	22.60

Table 3: SLA pairs where $D = 22.6$

Scenario	SLA_1	SLA_2	D
3	SLA(5, 0.85)	SLA(3, 0.95)	83.45
4	SLA(4.5, 0.8)	SLA(3, 0.95)	83.48

Table 4: SLA pairs where $D = 83.5$

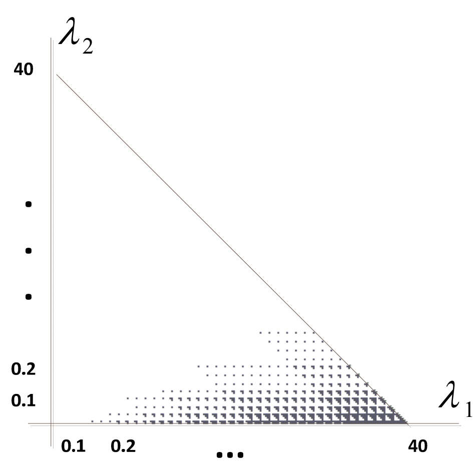


Figure 4: Scenario 1

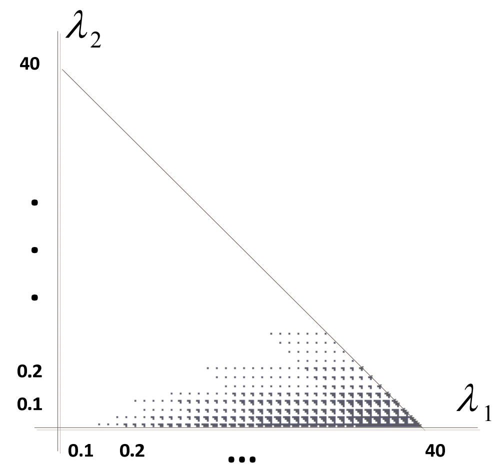


Figure 5: Scenario 2

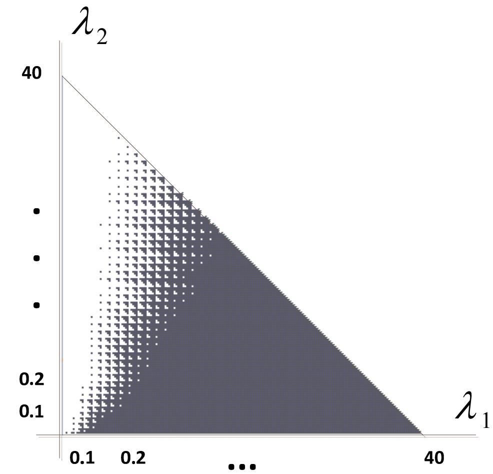


Figure 6: Scenario 3

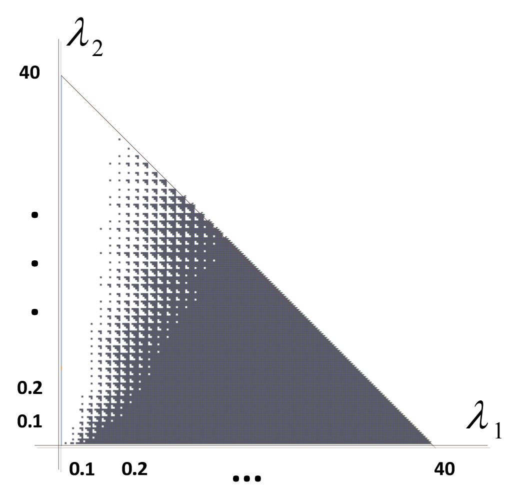


Figure 7: Scenario 4

3.4.3 Description of Algorithm

We observe from the results in Figures 4 to 7 that there are well-defined regions where DA or SA is likely the preferred strategy. These regions are separated by a straight line, as illustrated in Figure 8. Based on this observation, we define, for a given SLA difference, an angle α such that at least q% of the intersections in region 2 indicate that DA is the preferred strategy. In our investigation, we use $q = 90$. Using numerical examples, a plot of the angle α against SLA difference is shown in Figure 9.

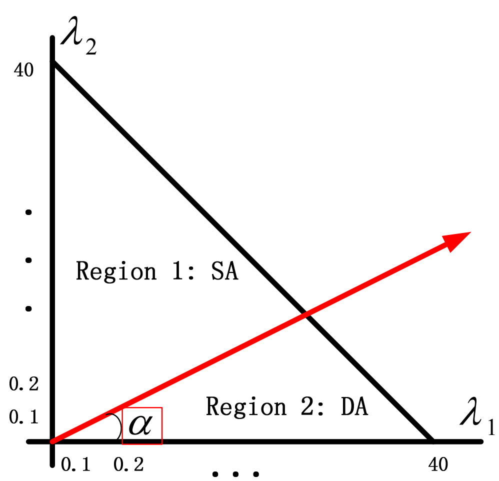


Figure 8: Heuristic method

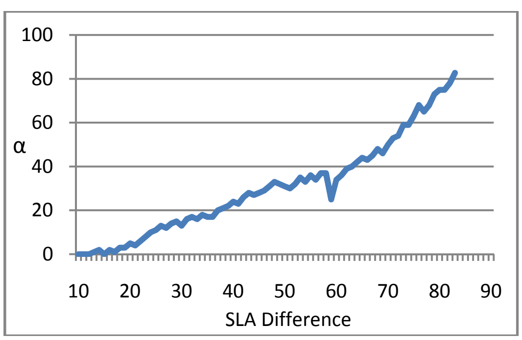


Figure 9: SLA difference vs. Angle

Our algorithm uses an “angle table” to capture the relationship between SLA difference and the angle α. An example of such a table is shown in Table 5 where the SLA difference is organized into 5 intervals. An angle α is pre-determined for each interval; the pre-determined value is the average of the α's for the SLA differences within the interval.

SLA Difference	Angle α (degree)
[0, 30)	0
[30, 62)	22
[62, 78)	52
[78, 82)	69
[82, 86.1)	77

Table 5: Angle Table

Our algorithm is included as Algorithm 2 below. We first compute $G(SLA_1)$ and $G(SLA_2)$ using Equation (4). These values are then used to compute the SLA difference D. The angle α corresponding to D is obtained from the angle table. If the intersection (λ_1, λ_2) is below the line defined by the angle α (i.e., in region 2 of Figure 8), DA is the preferred strategy; otherwise SA is the preferred strategy.

Algorithm 2

Input: λ_1, λ_2 // Arrival rates
SLA_1, SLA_2 // SLAs
Output: DA or SA // Allocation Strategy

1: Compute $G(SLA_1)$ and $G(SLA_2)$
2: Compute SLA difference $G(SLA_1) - G(SLA_2)$
3: Search angle table to obtain α
4: if $\tan^{-1} \lambda_2/\lambda_1 \le \alpha$, return DA, else return SA

3.5 Performance Evaluation

In this section, the heuristic algorithm presented in Section 3.4 is evaluated with respect to its ability to come up with a strategy (DA or SA) that results in the smallest number of servers. Our evaluation is based on the following consideration. Each time the global arbiter makes a resource allocation decision, it determines the number of servers required by the two job classes, using λ_i, x_i and y_i (i = 1, 2) as input parameters. Since these parameters may have different values at different time instants when resource allocation decisions are made, our approach is to consider a large number L, of combinations of λ_i, x_i and y_i (i = 1, 2). The performance of the heuristic algorithm for each combination is determined, and the average performance over the L combinations is used for evaluation purposes.

For each combination, the values of λ_i, x_i and y_i (i = 1, 2) are selected according to their respective probability distributions. These values are generated using random numbers. The probability distributions used in our evaluation are summarized in Table 6. These distributions represent the frequencies at which values of λ_i, x_i and y_i (i = 1, 2) are seen by the global arbiter. Note that three different distributions are used for λ_1 and λ_2, and only one distribution is used for each of the other parameters. The notation in Table 6 is explained as follows:

- U (a, b) – uniform between a and b
- N (20, σ^2) – normal with mean 20 and variance σ^2 (values ≤ 0 and > 40 are excluded)
- E (t) – exponential with mean t (values > 40 are excluded)

Parameter	Distribution
λ_1, λ_2	U (0, 40)
	N (20, σ^2)
	E (t)
x_1, x_2	U (a, b)
y_1, y_2	U (0.8, 0.95)

Table 6: Probability distributions

For our heuristic algorithm, its effectiveness is measured by: S = Prob [heuristic algorithm finds a correct strategy]. By correct strategy, we mean a strategy that requires the smallest number of servers to meet the SLA of both classes. In case DA and SA lead to the same smallest number, then both are considered as a correct strategy. The performance metric S is obtained as follows. We repeat the steps shown in Procedure 1 L times (the initial value of the variable $n_{correct}$ is zero). S is then given by: $S = n_{correct}/L$.

Our results for $L = 10{,}000$ and 6 different settings of the probability distributions are shown in Table 7; for all settings, the distribution used for y_1 and y_2 is U (0.8, 0.95). These results show that our heuristic algorithm has at least a 96% probability of finding a correct strategy for all the cases considered. They indicate that the heuristic algorithm is effective in determining a strategy that requires the smallest number of servers.

Procedure 1

1: Generate values for λ_1 and λ_2.
2: **if** $\lambda_1 + \lambda_2 > 40$, then goto step 1.
3: Generate values for x_1, x_2, y_1, and y_2.
4: Apply Algorithm 2 to obtain an allocation strategy (denoted by R_1).
5: Compute m_D and m_S using Equations (2) and (3). The correct strategy (denoted by R_2) is DA if $m_D \leq m_S$ or SA if $m_S \leq m_D$.
6: if R_1 is the same as R_2, then $n_{correct}$++.

λ_1, λ_2	x_1, x_2	S
U (0, 40)	U (2, 5)	0.973
U (0, 40)	U (2, 10)	0.979
N (20, 5)	U (2, 5)	0.961
N (20, 10)	U(2, 5)	0.966
E (10)	U (2, 5)	0.982
E (20)	U (2, 5)	0.984

Table 7: Probability of correct strategy

4 Priority Disciplines

In this section, we consider scenarios where the scheduling discipline is not restricted to FCFS. Obvious choices are disciplines that give priority to the job class that has a more demanding SLA, e.g., a smaller response time threshold x and/or a larger target probability y. Such disciplines are only applicable under shared allocation (SA). Two priority disciplines are considered: head-of-the-line priority and a new discipline called probability dependent priority.

4.1 Head-of-the-Line Priority

In head-of-the-line priority (HOL), the job class with the larger $G(SLA)$ value has higher priority. Whenever a server becomes available, jobs in the higher priority class are considered first. If the queue of the higher priority class is empty, then jobs in the lower priority class are considered. Within the same class, jobs are processed in FCFS order.

4.2 Probability Dependent Priority

Probability dependent priority (PDP) is a new scheduling discipline designed to maximize the probability of meeting a given response time goal. This should have a positive effect in terms of minimizing the number of servers required. Let τ_i be the measured frequency that the response time of class $i \leq$ the threshold x_i. The following counters are used in PDP (both are zero initially):

- $total_i$ - number of class i jobs completed so far
- met_i - number of completed class i jobs that has response time $\leq x_i$

Each time a class i job completes service, the following steps are performed:

- $total_i$ is incremented by one.
- If this job has response time $\leq x_i$, met_i is incremented by one.
- Compute a new value of τ_i using the equation: $\tau_i = met_i/total_i$.
- Update P_i, the priority of class i, which is defined as follows: $P_i = y_i - \tau_i$.

In PDP, the job class with the larger P_i has higher priority. In case both classes have the same priority value, then the next job class to receive service is selected at random. Note that with PDP, a job class has higher priority if it is meeting the SLA with a smaller margin or is falling behind by a larger margin. In addition, the priority of a job class may change over time because τ_i is updated each time a class i job completes service.

4.3 Performance Evaluation

In this section, the performance difference of FCFS, HOL, and PDP is investigated. For FCFS, results are provided by the heuristic algorithm in Section 3.4. As to HOL and PDP, analytic results for the response time distribution are difficult to obtain, so simulation is used.

Let m_F, m_H and m_P be the smallest number of servers required by FCFS, HOL, and PDP, respectively, such that the SLA of both classes are met. We say that

- FCFS is a top discipline if $m_F \leq m_H$ and $m_F \leq m_P$,
- HOL is a top discipline if $m_H \leq m_F$ and $m_H \leq m_P$, and
- PDP is a top discipline if $m_P \leq m_F$ and $m_P \leq m_H$.

The methodology presented in Section 3.5 is used in our evaluation. The performance metrics are q_F, q_H and q_P, the fractions of time that FCFS, HOL, and PDP are a top discipline, respectively. The steps shown in Procedure 2 are repeated L times (the initial values of n_F, n_H and n_P are zero). q_F, q_H and q_P are then given by $q_F = n_F/L$ $q_H = n_H/L$, and $q_P = n_P/L$.

Procedure 2

1: Generate values for λ_1 and λ_2.
2: if $\lambda_1 + \lambda_2 > 40$, then goto step 1.
3: Generate values for x_1, x_2, y_1, and y_2.
4: Apply Algorithm 2 to obtain a correct strategy for FCFS and use Equations (2) or (3) to determine m_F.
5: Obtain m_H and m_P by simulation.
6: if $m_F \leq m_H$ and $m_F \leq m_P$, then n_F++.
7: if $m_H \leq m_F$ and $m_H \leq m_P$, then n_H++.
8: if $m_P \leq m_F$ and $m_P \leq m_H$, then n_P++.
9: if $m_P < m_F$ and $m_P < m_H$, then n++, s_F+= $m_F - m_P$, and s_H+= $m_H - m_P$.

Our results for $L = 10{,}000$ and 6 different settings of the probability distributions are presented in Table 8. These results show that PDP is superior to HOL and FCFS in terms of the fraction of time that it is a top discipline. Specifically, PDP is a top discipline over 97% of the time, compared to less than 30% for HOL and less than 2% for FCFS.

λ_1, λ_2	x_1, x_2	q_F	q_H	q_P
U (0, 40)	U (2, 5)	1.6%	25.5%	98.3%
U (0, 40)	U (2, 10)	0.9%	29.4%	97.1%
N (20, 5)	U (2, 5)	1.3%	24.8%	98.5%
N (20, 10)	U (2, 5)	1.1%	23.1%	98.8%
E (10)	U (2, 5)	1.4%	27.5%	98.4%
E (20)	U (2, 5)	1.5%	24.6%	98.0%

Table 8: Performance Comparison

To provide further insight into the performance advantage of PDP, we compute, for those combinations of λ_i, x_i and y_i (i = 1, 2) where PDP is the top discipline (i.e., $m_P < m_F$ and $m_P < m_H$), the average difference between the number of servers required by PDP and that required by each of the other two disciplines. This is done by step 9 of Procedure 2 where s_F and s_H are used to accumulate the difference between m_F and m_P and that between m_H and m_P; n is used to keep track of the number of combinations where PDP is the top discipline (n, s_F and s_H are initially 0). The average differences are then given by $\Delta_F = s_F/n$ and $\Delta_H = s_H/n$.

Results for Δ_F and Δ_H for the 6 settings of probability distributions are shown in Table 9. These results show that the difference in number of servers required is consistent across probability distributions, with an average of about 1.4 for Δ_F and about 1.2 for Δ_H.

λ_1, λ_2	x_1, x_2	Δ_F	Δ_H
$U(0, 40)$	$U(2, 5)$	1.41	1.29
$U(0, 40)$	$U(2, 10)$	1.36	1.16
$N(20, 5)$	$U(2, 5)$	1.49	1.29
$N(20, 10)$	$U(2, 5)$	1.43	1.26
$E(10)$	$U(2, 5)$	1.33	1.12
$E(20)$	$U(2, 5)$	1.36	1.17

Table 9: Performance Difference

5 Related Work

Related work in autonomic resource management can be organized according to the approach used in the investigation, including queueing theory, control theory, machine learning, and cloud computing.

Queueing theory [8-11] is a well established and widely used methodology in performance evaluation of resource management strategies. In [8], the authors present utility models based on a system of N parallel M/M/1 queues and use results for the mean response time and throughput to maximize the total utility. In [9], a predictive multiclass queueing network model is used to compute the mean response time. A layered queueing network is used in [10] to study the effect of workload and system parameters on performance. A regression based approximation of the CPU demand of client transactions is introduced in [11]; the approximation is obtained using a queueing network model with each queue representing an application tier.

Control theory [12-15] has been used in the design of dynamic resource management schemes. In [12], a system is developed that can meet application-level quality of service while achieving high resource utilization. An analytic foundation of control theory for a self-managing system is described in [13]. In [14], the authors argue that control theory should be used to build and to configure self-managing systems. The 1000 Island solution architecture is presented in [15]; this architecture has multiple resource controllers that are based on control theory and optimization methods.

Machine learning has also been used in autonomic resource management [16-18]. A lightweight on-line learning of correlations between system state and response time is described in [16]. In [17], an active learning approach is used to build predictive models to forecast the completion time of batch jobs. A combination of off-line reinforcement learning and queueing theory is used to improve the accuracy of the prediction [18].

Cloud computing [19] is emerging as a new computational model in which computing is offered as a service over the Internet. A cloud can comprise a large number of hardware and software resources shared by a large number of applications. Scheduling and optimization results in clouds have been reported recently [3, 4]. Both papers consider SLAs as mean response time per class and the objective function is the cost and respectively the profit of a cloud.

6 Concluding Remarks

The results in this paper provide valuable insights into the performance of alternative resource allocation strategies and job scheduling disciplines for a cloud computing infrastructure. In our investigation, the service level agreement is based on response time distribution, which is more relevant than the mean response time with respect to the performance requirement of interactive applications. We have developed an efficient and effective algorithm to determine the allocation strategy that results in smallest number of servers required. We have also developed a novel scheduling discipline, called probability dependent priority, which is superior to FCFS and head-of-the-line priority in terms of requiring the smallest number of servers. Although our focus is on the case of two job classes, our findings can be used to develop guidelines for resource provisioning for more complex scenarios.

Acknowledgements

This work was supported by the IBM Toronto Lab Centre for Advanced Studies and the Ontario Centres of Excellence.

About the Authors

Ye Hu received his MMath degree in Computer Science from the University of Waterloo in 2009. He was an IBM CAS Fellowship student in 2007 and 2008. He is currently a System Design Specialist at Thales Group Toronto Division.

Johnny Wong received his Ph.D. degree in Computer Science from the University of California at Los Angeles in 1975. Since 1975, he has been with the University of Waterloo where he is currently a Professor in the David R. Cheriton School of Computer Science. He was Director of the School from 2003 to 2006. His research interests are in the areas of performance evaluation, distributed systems, resource management, and information delivery.

Gabriel Iszlai is a Senior Technical Staff Member with the IBM Center for Advanced Studies in Toronto, Canada. He received his B.S. degree in 1992. Prior to joining IBM he worked as Chief Scientist for ThinkDynamics, a company acquired by IBM in May 2003. He was one of the initial designers of the former ThinkControl application, known today as IBM Tivoli Intelligent Orchestrator. Prior to that, he worked for over 8 years in the IT industry for different European telecom companies.

Marin Litoiu is a professor at York University. Prior to that he was a Senior Research Staff Member with Centre for Advanced Studies, IBM Toronto Lab, where he led the research programs in Autonomic Computing, System Management and Software Engineering. He was the Chair of the Board of CSER, a Canadian Consortium for Software Engineering Research and Director of Research for Centre of Excellence for Research in Adaptive Systems. Dr. Litoiu holds doctoral degrees from University Polytechnic of Bucharest and from Carleton University. His research interests include autonomic computing; high performance software design; performance modeling, performance evaluation and capacity planning for distributed and real time systems.

References

[1] W.E. Walsh, G. Tesauro, J.O. Kephart, and R. Das. Utility Functions in Autonomic Systems. *Proc. 1st International Conference on Autonomic Computing*, New York, 2004.

[2] L. Kleinrock. *Queuing Systems Volume 2: Computer Applications*. Wiley-Interscience, New York, 1976.

[3] A. Lenk, M. Klems, J. Nimis, et al. What's Inside the Cloud? An Architectural Map of Cloud Landscape. *Proc. ACM/IEEE Symposium on Cloud Computing Challenges*, 23-31, Vancouver, 2009.

[4] J. Li, J. Chinneck, M. Woodside, and M. Litoiu. Fast Scalable Optimization to Configure Service Systems having Cost and Quality of Service Constraints. *Proc. IEEE International Conference on Autonomic Systems*, Barcelona, 2009.

[5] Y. Hu. *Resource Allocation for Multiple Job Classes*. Master's Thesis, University of Waterloo, 2009.

[6] M. Kwok. *Performance Analysis of Distributed Virtual Environments*. PhD Thesis, University of Waterloo, 2006.

[7] J.W. Wong and S.S. Lam. Queueing Network Models of Packet-Switching Networks, Part I: Open Networks. *Performance Evaluation*, 9-21, 1982.

[8] G. Tesauro, R. Das, W.E. Walsh, and J.O. Kephart. Utility-function-driven resource allocation in autonomic systems. *Proc. 2nd International Conference on Autonomic Computting*, Seattle, 2005.

[9] M.N. Bennani and D.A. Menasce. Resource Allocation for Autonomic Data Centers using Analytic Performance Models. *Proc. 2nd International Conference on Autonomic Computing*, Seattle, 2005.

[10] M. Woodside, T. Zheng, and M. Litoiu. Service System Resource Management Based on a Tracked Layered Performance Model. *Proc. 3rd International Conference on Autonomic Computing*, Dublin, 2006.

[11] Q. Zhang, L. Cherkasova, and E. Smirni. A Regression-Based Analytic Model for Dy-

namic Resource Provisioning of Multi-tier Applications. *Proc. 4th International Conference on Autonomic Computing*, Jacksonville, Florida, 2007.

[12] P. Padala, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, K. Salem, and K. Shin. Adaptive Control of Virtualized Resources in Utility Computing Environments. *Proc. European Conference on Computer Systems*, Lisbon, 2007.

[13] Y. Diao, J.L. Hellerstein, S. Parekh, R. Griffith, G.E. Kaiser, and D. Phung. A Control Theory Foundation for Self-Managing Computing Systems. *IEEE Journal on Selected Areas in Communications*, 2213-2222, 2005.

[14] C. Karamanolis, M. Karlsson, and X. Zhu. Designing Controllable Computer Systems. *Proc. USENIX Workshop on Hot Topics in Operating Systems*, Santa Fe, New Mexico, 2005.

[15] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian and L. Cherkasova. 1000 Islands: Integrated Capacity and Workload Management for the Next Generation Data Center. *Proc. 5th International Conference on Autonomic Computing*, Chicago, 2008.

[16] S. Ghanbari, G. Soundararajan, J. Chen, and C. Amza. Adaptive Learning of Metric Correlations for Temperature-Aware Database Provisioning. *Proc. 4th International Conference on Autonomic Computing*, Jacksonville, Florida, 2007.

[17] P. Shivam, S. Babu, and J. Chase. Learning Application Models for Utility Resource Planning. *Proc. 3rd International Conference on Autonomic Computing*, Dublin, 2006.

[18] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani. A Hybrid Reinforcement Learning Approach to Autonomic Resource Allocation. *IEEE Internet Computing*, 22-30, 2007.

[19] M. Litoiu and G. Iszlai. Performance Model Driven QoS Guarantees and Optimization in Clouds. *Proc. ACM/IEEE Symposium on Cloud Computing Challenges,* 15-22, *Vancouver*, 2009.

Full Papers

CASCON x EVOKE 2019