ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Resource Allocation under Uncertainty in Cloud Storage for Computational Environment

Deepa.V[1], Gopalakrishna.L[2]

Asst. Prof, Department of CSE, Velammal Engineering College, Chennai, India[1]

M.E, Department of CSE, Velammal Engineering College, Chennai, India[2]

ABSTRACT: A global computation market could be realized by a high-performance federated architecture that has cloud computing providers. They use economic aware allocation mechanisms driven by the underlying allocation requirements of cloud providers. In cloud, resources are reserved by m participants during auction and in most cases the resources are utilized by n users. Therefore all other m-n reservations are wasted in the duration of auction. In this process last minute bidding is used which is time and cost expensive. To overcome this wastage we propose two principles in general, first avoid commitment of resources, second avoid repeating auction and allocation process. We have distilled these principles into five high-performance resource utilization strategies, namely: overbooking, advanced reservation, just-in-time (JIT) bidding, progressive contracts, and using substitute providers to compensate for encouraging oversubscription. These strategies are examined experimentally with the DRIVE Meta scheduler. Several diverse synthetic workloads have been used to measure both the performance benefits and economic implications of these strategies.

KEYWORDS: Resource allocation, Last minute bidding, JIT bidding, advanced reservation.

I. INTRODUCTION

Computational economies have long been touted as a means of allocating resources in both centralized and decentralized computing systems. The adoption of economies in production systems has been limited due to criticisms relating to, among other things, poor performance, high latency, and high overheads. The application of two general principles largely addresses avoid commitment of resources, and avoid repeating negotiation and allocation processes. The high performance resource utilization strategies are overbooking, advanced reservation, Just-in-time (JIT) bidding, progressive contracts and using substitute providers to compensate for encouraging oversubscription. Each of the strategies is examined within the context of a market-based cloud using the DRIVE meta-scheduler. Each strategy has been implemented in DRIVE and is analyzed. The high utilization strategies proposed in this paper are designed to minimize the impact of these factors to increase occupancy and improve system utilization. The high utilization strategies have each been implemented in the DRIVE Meta scheduler and evaluated using a series of batch and interactive workloads designed to model different scenarios, including multiple high throughputs, short job duration workloads in which auction mechanisms typically perform poorly. The individual strategies, and the combination of the different strategies, were shown to dramatically improve occupancy and utilization in a high performance situation.

1.1 OVERBOOKING

The overbooking strategies for cloud infrastructure providers lead to increase their profit and competitiveness. The objective of overbooking is to improve the expected resource allocation. Instead of allocating each resource once, high performance can be achieved by allocating them several times. Overbooking offers a solution for the system utilization

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

problem, by allowing the resource provider to accept more reservations than the capacity. Hence, it can be effectively used to minimize the loss of revenue. However, the challenging issues in using overbooking are determining the appropriate number of excess reservations, minimizing total compensation cost, addressing legal and regulatory issues, and dealing with market acceptance, especially the ill-will or negative effects from users who have been denied access. The EASY (Extensible Argonne Scheduling system) backfilling approach can be used to improve system utilization. Within EASY backfilling putting a job in a gap is acceptable if the first job in the queue is not delayed. This preserves starvation and leads to an increased utilization of the underling system. However, EASY backfilling has to be used with caution in systems guaranteeing QoS aspects, since jobs in the queue might be delayed.

II. LITERATURE REVIEW

The important idea of literature review is, it convey others contribution on our research area. It gave idea for new researches and related works are accomplished by others. Literature review demonstrates our understanding of the relevant proposal of others and our ability for summarizing information gained from others works. There are two key steps should follow in literature review. They are finding sources and synthesizing information. These steps are accomplished in my literature review about DRIVE meta scheduler. The literature review provides information about service reservations, policies, valuation and economic protocol. This project mainly concentrates the allocation of workload in distributed and federated computing environments.

2.1 Distributed Resource Infrastructure for a Virtual Economy (DRIVE)

Distributed Resource Infrastructure for a Virtual Economy (DRIVE) is a distributed economic meta-scheduler designed to allocate workload in distributed and federated computing environments. Allocation in DRIVE is abstracted through an economic market which allows any economic protocol to be used. DRIVE features a novel "co-op" architecture, in which core meta scheduling services are hosted on participating resource providers as a condition of joining the Virtual Organization (VO). This architecture minimizes the need for dedicated infrastructure and distributes management functionality across participants. The co-op architecture is possible due to the deployment of secure economic protocols which provide security guarantees in untrusted environments.

2.2DRIVEARCHITECTURAL COMPONENTS

DRIVE (Distributed Resource Infrastructure for a Virtual Economy) is a community meta-scheduler implemented within a Virtual Organization (VO) using resources contributed by the members of the VO to conduct the auction. The VO model is used to group resource providers and users to specific Grid systems and a VO membership service controls VO access to services and resources by authenticating participants.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

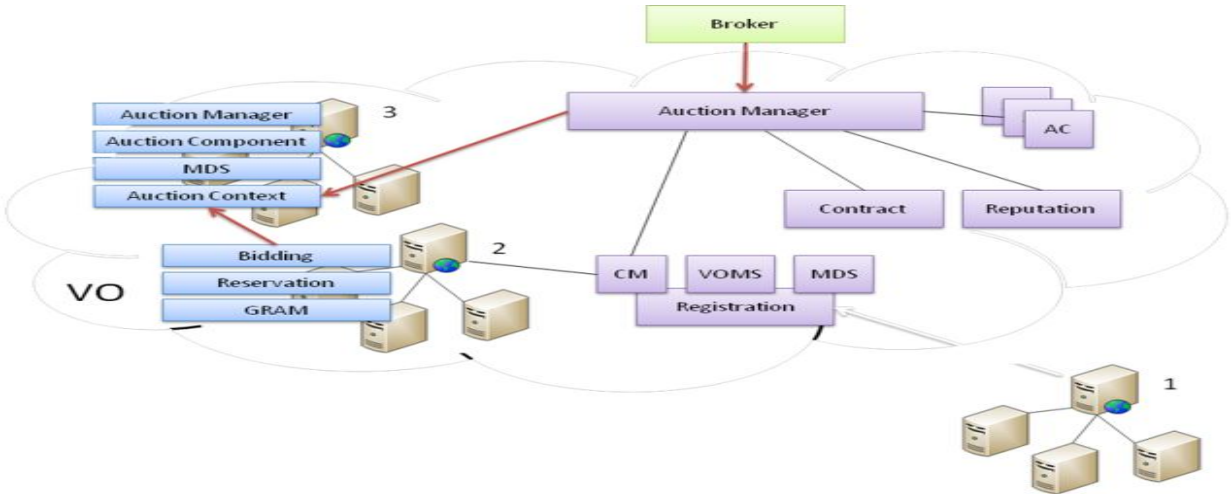


Fig:1 Overview of the DRIVE Architectural Components

The high level DRIVE architecture showed in Fig 1. The cloud outlines the scope of the VO. Resource providers (2) and (3) are members of the VO, while resource provider (1) is attempting to join the VO. On joining the VO, a resource provider exposes a set of obligation services which it contributes to the VO for the running of the meta-scheduler. Obligation services, shown by resource provider (3), can include the Auction Manager, Auction Context, Auction Component and MDS. Once a member of the VO, the resource provider also exposes participation services which are needed to bid for, accept and schedule jobs on the resource provider. Participation services, shown by resource provider (2), include the Bidding Service, Reservation Service and GRAM. The services are categorized into trusted core, obligation, and participation services.

2.3 DESIGN GOALS

DRIVE is based on a decentralized distributed architecture composed of a group of market and participant services. It is built upon a scalable distributed service-based architecture in which services can potentially be hosted on participating providers.

The core design goals and their implications on the architecture are:

a). Provider and task independent: DRIVE is designed to be independent from a single provider or class of providers such that different types of provider can participate in a federated market. For example, DRIVE can be used to submit jobs in cloud environment.

b). Support efficient flexible allocation mechanisms: The core responsibility of a meta-scheduler is providing efficient allocation. DRIVE focuses on efficient allocation using computational (or service) economies for two reasons: first, economic allocation principles are necessitated as commercial providers may be included in the VO.

c). Secure: The core responsibility of a meta-scheduler is providing efficient allocation. DRIVE focuses on efficient allocation using computational (or service) economies for two reasons: first, economic allocation principles are necessitated as commercial providers may be included in the VO.

d). Provide a strong contract management framework: With the development of utility computing models there is a requirement to define, monitor and enforce agreed upon levels of service. Consumers will not be prepared to pay if negotiated service levels are not delivered. DRIVE creates standardised binding contracts as a result of allocation such that consumers and providers clearly define their respective requirements and obligations. Contract monitoring and enforcement is outside the scope of this thesis, however both have been considered to future-proof the architecture.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

2.4 DRIVE SERVICE

- Trusted Core Services

Category Manager: A registration service for resource providers. Resource providers register their bidding profile when joining the VO. This profile is used to categorize the type of job a resource is interested in hosting. When an auction is conducted the Category Manager is consulted and appropriate resource providers are selected based on their resource profiles. This has the effect of reducing auction size and therefore increasing the efficiency of the allocation process.

Contract Service: Stores contracts that have been issued within the VO. It is used in the contract hardening process before redemption, ensuring that the contract can be honored. It is also used to monitor SLAs contained in the contracts.

- Obligation Services

Auction Manager: The central auction service, it manages the auction process and advertises the auction to suitable bidders. The broker starts an auction by passing the job description (JSDL) to the Auction Manager which then creates the appropriate auction resources and selects the services required for the particular auction protocol.

Auction Context: Stores state for individual auctions, including status and protocol specific state. Clients can register for notifications or poll the service for status, results and contracts.

Auction Component: A universal component which can be used for different auction protocols, it is a shell that wraps protocol specific services. The Auction Component takes different forms depending on the protocol, for example in a verifiable Secure Generalized Vickery Auction (vSGVA) the Auction Component is an Auction Evaluator used to compute the winner of the auction. In the Garbled Circuit protocol the Auction Component is an Auction Issuer used to garble the circuit and is required for the VPOT protocol.

- Participation Services

Bidding Service: Responds to auction events from the Auction Manager and computes bids for jobs it wishes to host using pricing algorithms, local policy, and consideration of future commitments.

Reservation Service: Stores any commitments made by the provider, normally in the form of contracts. This service is the key component for supporting advanced reservation in DRIVE, the Bidding Service is able to check resource commitments using its Reservation Service prior to bidding on advanced reservation auctions.

III. OVERBOOKING MODELS AND ALGORITHMS

3.1 SERVICE LEVEL AGREEMENT (SLA)

Cloud contracts will be based on the negotiation of SLAs between the service providers and their customers. During SLA negotiation, the customers reserve the amount of required computing and storage resources for a given time period .If the computation takes longer than expected, jobs are commonly killed at the end of the SLA-defined lifetime. Therefore, users are cautious not to lose their jobs and tend to overestimate their job's execution time and reserve resources accordingly. In practice, this leads to underutilized resources, as jobs finish often much earlier than expected. Overbooking is used in many commercial fields, where more people buy resources than actually use them. To improve profit, airlines have e.g. become used to sell seat reservations more than once. The number of reservations that will not be used is estimated based on prior experiences and used in the planning processes. This estimated number is only correct with a

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

specific probability. Consequently, if more passengers appear than expected and not enough seats are available in the aircraft, the airline has to pay a penalty to its customers. Obviously, the objective of overbooking is to improve the expected profit. Instead of selling each seat once, profit can be increased by selling them several times. This opportunity has to be compared with the risk implied by overbooking, i.e. the compensation for the buyer if no seat is available combined with the probability of that event. The best estimation of risk and opportunity will provide the best profit. In this paper, we propose different overbooking strategies for, cloud to increase their profit and competitiveness. These strategies differ in many aspects from traditional overbooking strategies for aircraft seats or hotel beds. On the one side, the number of concurrent users of a compute resource is smaller than the number of passengers of an airplane, making it harder to predict expected behavior. On the other side, computing jobs can be started nearly anytime, while a plane only takes off once for each flight. Conservative scheduling strategies, which do not use overbooking, do not accept a job, if the maximum estimated job duration is even slightly longer than any gap in the current schedule. Applying overbooking, the scheduler can assess the risk to place the job in a gap that is smaller than the estimated execution time. For such an overbooked job, the probability of failure is no longer only dependent on machine failure rates (as in conservative scheduling), but it also depends on the likelihood that the real execution time of the job is longer than the gap length. The proposed strategies are based on an analytical model for overbooking that uses the convolution of the probability density functions of the runtime estimates of the jobs to calculate the probability of failure (PoF) for a SLA. When the calculated risk is acceptably small in comparison to the opportunity, the service provider can accept the SLA.

3.2 SCHEDULING

Most scheduling strategies for cluster systems are based on a first-come first-serve (FCFS) approach that schedules jobs based on their arrival times. FCFS guarantees fairness, but leads to a poor system utilization as it might create gaps in the schedule. The gaps can occur, because each job description does not only contain execution time information, but also information about its earliest starting time / release time. As standard FCFS schedules jobs strictly according to their arrival times, resulting gaps will remain idle and waste resources.

3.2.1 BACKFILLING APPROACH

To increase system utilization and throughput in this scenario Backfilling has been introduced . Backfilling schedules a new job not necessarily at the end of the plan, but is able to fill gaps if a new job fits in. The additional requirement for the ability to use backfilling instead of simply FCFS is an estimation about the runtime of each job. This allows to determine if the job fits in a gap in the schedule.

EASY (Extensible Argonne Scheduling system) backfilling approach can be used to improve system utilization. Within EASY backfilling putting a job in a gap is acceptable if the first job in the queue is not delayed. This preserves starvation and leads to an increased utilization of the underling system. However, EASY backfilling has to be used with caution in systems guaranteeing QoS aspects, since jobs in the queue might be delayed.

Conservative backfilling approach which only uses free gaps if no previously accepted job will be delayed. Thus, conservative backfilling still preserves fairness. Additionally, it is possible to plan a job, this means to determine the latest start time for every job. The latest start time is the time a job starts when all its predecessors use their complete estimated runtime.

3.2.2 JOB SCHEDULING INFORMATION

Overbooking means to put a job in a gap in a schedule that is smaller than the job's maximum execution time. In fact, this job may actually try to use more time than the available size of the gap, leading either to a loss of this job or to a postponement of the following job. Both cases might lead to a penalty for the service provider.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

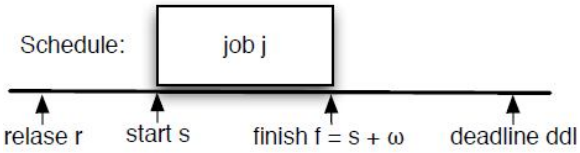
Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Variable	Content	Comment
r	release time	earliest start time
ω	estimated execution time	given by the user
ddl	deadline	given by the user
s	start time	planned start time
f	finish time of the job	planned job end

We assume a system with a single resource that has a failure rate ! and a repair rate μ, which are distributed according to a Poisson distribution. A job j has an earliest release time r, an estimated execution time ", and a deadline ddl . When the job is placed, the start time s is either its release time or the finish time of the previous job. The finish time f is important if the scheduling strategy follows conservative backfilling, where the job should not delay following jobs. Therefore, the job will be killed at $f = s_{next}$



3.2.3 OVERBOOKING ALGORITHM

This paragraph briefly defines possible scheduling strategies for backfilling with overbooking. Generally, the scheduler holds a list of all jobs in the schedule. For each new job j_{new} arriving in the system, the scheduler computes the PoS (Probability of Success) for the execution of this job in every space free in the schedule where the job might be executed. For the concrete implementation of the scheduling algorithm, several decision strategies could be applied.

– A conservative approach could be chosen, where the job is placed in the gap with the highest PoS.

– A best fit approach uses the gap providing the highest profit, while still ensuring an acceptable PoF.

– A first fit approach, where the job is placed in the first gap with an acceptable PoS.

If the job is not placeable within the schedule, it can be planned as last job, if it is still executable before the user given deadline of the job.

IV. OVERBOOKING POLICIES

The cancellation and no shows affect the resource utilization in overbooking concept. The overbooking can protect a resource provider against unanticipated cancellations and no-shows. We define a cancellation as a reservation that is terminated by a user before the service start. Moreover, a no-show as a reservation that fails to arrive and run on the resource without a cancellation notice. We adopt several static overbooking policies in our work. These static policies only calculate the ideal overbooking limit periodically prior to ts, when the state and probabilities change over time. Thus, we assume the following things:

a)The cancellations and no-shows are independent of the number of total bookings.

b)The probability of a cancellation is markovian, i.e. it only depends on the current time.

C)The no-shows are treated as cancellations on ts. Hence, we can define q(t) as a show rate or a probability that reservations show up from the time remaining until ts.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

4.1 ADVANCED RESERVATION

A QoS scheduler must allow consumers to reserve resources in advance. When a provider accepts an advance reservation, the consumer expects to be able to access the agreed resources at the specified time. However, changes may occur in the scheduling queue between the time the consumer submits the reservation to the time the consumer receives the resources. There are a number of reasons for such changes including: consumers cancelling requests, consumers modifying requests, resource failures, and errors in estimating usage time in the consumer requests. Therefore, from the resource provider's perspective, a good time-slot for the consumer at the time the SLA was agreed may be a bad time-slot in the future due to increased fragmentation. This fragmentation reduces the potential scheduling opportunities and results in lower utilization. Before the scheduling of a new job, the state of the system is consistent, this means that the current scheduling of all jobs meets the users QoS requirements. The QoS scheduler uses SLAs to efficiently schedule advance reservations for computation services based on their flexibility. When a provider accepts an advance reservation, the consumer expects to be able to access the agreed resources at the specified time.

4.1.1 JOBSCHEDULING

The scheduling of a job consists on finding a free time-slot that meets the job requirements. Rather than providing the user with the resource provider's scheduling queue, we assume that the user asks for a time-slot and the resource provider verifies its availability. Scheduling takes place in two stages a) all jobs that are currently awaiting execution on the machine are sorted based on some criteria and b) then this list is scheduled in order, and if the new job can be scheduled, the SLA is accepted. If the job cannot be scheduled, then the scheduler can return a set of schedulable alternative times.

- Sorting

Firstly we separate the jobs currently allocated into two queues: running queue N and waiting queue. The first queue contains jobs already in execution and cannot be rescheduled. The second queue contains jobs that can be rescheduled. The approach we adopt here is to try to reschedule the jobs in the waiting queue by sorting them first and then attempting to create a new schedule.

- Scheduling

We scheduling a new job j_k at the current time CT, returning true if it is possible to scheduled it, or false and a list of optional possible scheduling. Before the scheduling of a new job, the state of the system is consistent, this means that the current scheduling of all jobs meets the users QoS requirements. Therefore, during the scheduling process, if a job j_i is rejected there are two options (a) $j_i = j_k$, the new job could not be scheduled, (b) $j_i = j_k$, the new job was scheduled but generated a scheduling problem for another job.

4.1.2 IMPLEMENTING METASCHEDULER WITH OVERBOOKING

Software as a Service (SaaS) is a scalable application deployment model in which applications are provided to consumers on demand through a service oriented interface. The SaaS model forms the top layer in Cloud computing models operating above infrastructure and platform services. There are many reasons why users may wish to expose an application as a service, for example sharing data or tools, multi-user efficiency, flexibility, extensibility and scalability. As we want to improve the profit by overbooking certain time slots, we need to estimate the Probability of Failure PoF and Probability of Success PoS, which is (1- PoF),for each overbooked schedule. Overbooking means to put a job in a gap in a schedule that is smaller than the job's maximum execution time. In fact, this job may actually try to use more time than the available size of the gap, leading either to a loss of this job or to a postponement of the following job. Both cases might lead to a penalty for the service provider. Generally, the scheduler holds a list of all jobs in the schedule. For each new job arriving in the system, the scheduler computes the PoS for the execution of this job in every space free in the schedule where the job might be executed. For the concrete implementation of the scheduling algorithm, several decision strategies could be applied. Before the scheduling of a new job, the state of the system is consistent, this means that the current scheduling of all jobs meets the users QoS requirements.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

V. CONCLUSION

In this project we explored the resource allocation strategies. We also analyze the Distributed Resource Infrastructure for a Virtual Economy (DRIVE) in a distributed system. Main focus of this project is implementing overbooking concept. For this we use DRIVE meta-scheduler and evaluated using a series of batch and interactive workloads designed to model different scenarios. This project also explains how overbooking concept is useful to achieve high utilization and negotiation latency.

REFERENCES

[1] K. Chard and K. Bubendorfer, "High Performance Resource Allocation Strategies for Computational Economies" IEEE Transactions on parallel and distributed systems, Vol. 24, No. 1, January 2013.

[2] I.E. Sutherland, "A Futures Market in Computer Time," Comm. ACM, vol. 11, no. 6, pp. 449-451, 1968.

[3] G. Birkenheuer, A. Brinkmann, and H. Karl, "The Gain of Overbooking," Proc. 14th Int'l Workshop Job Scheduling Strategies for Parallel Proc. (JSSPP).

[4] K. Chard and K. Bubendorfer, "A Distributed Economic Meta-Scheduler for the Grid," Proc. IEEE Eighth Int'l Symp. Cluster Computing and the Grid (CCGRID '08).

[5] A. Sulistio, K.H. Kim, and R. Buyya, "Managing Cancellations and No-Shows of Reservations with Overbooking to Increase Resource Revenue," 2008.

[6] K. Chard, K. Bubendorfer, and P. Komisarczuk, "High Occupancy Resource Allocation for Grid and Cloud Systems, a Study With Drive," Proc. 19th ACM Int'l Symp. High Performance Distributed Computing (HPDC '10), 2010.

[7] J. Subramanian, S. Stidham Jr., and C.J. Lautenbacher, "Airline Yield Management with Overbooking, Cancellations, and No- Shows," Transportation Science, vol. 33, no. 2, 1999.

[8] B.C. Smith, J.F. Leimkuhler, and R.M. Darrow, "Yield Management at American Airlines," INTERFACES, vol. 22, no. 1, 1992.

[9] A.E. Roth and A. Ockenfels, "Last-Minute Bidding And the Rules for Ending Second-Price Auctions: Evidence from Ebay and Amazon Auctions on the Internet," Am. Economic Rev., vol. 92, no. 4, 2002.

[10]. R. Buyya, R. Ranjan, and R.N. Calheiros, "Intercloud: Utility- Oriented Federation of Cloud Computing Environments for Scaling of Application Services," Proc. 10th Int'l Conf. Algorithms and Architectures for Parallel Processing, p. 20, 2010.

[11] K. Chard, "Drive: A Distributed Economic Meta-Scheduler for the Federation of Grid and Cloud Systems," PhD. dissertation, School of Eng. and Computer Science, Victoria Univ. of Wellington, 2011.

[12] M.A. Netto, K. Bubendorfer, and R. Buyya, "Sla-Based Advance Reservations with Flexible and Adaptive Time Qos Parameters," Proc. Fifth Int'l Conf. Service-Oriented Computing (ICSOC '07), 2007.

[13] K. Chard and K. Bubendorfer, "Using Secure Auctions to Build A Distributed Meta-Scheduler for the Grid," Market Oriented Grid and Utility Computing, series Wiley Series on Parallel and Distributed Computing, R. Buyya and K. Bubendorfer, eds.,pp. 569-588, Wiley, 2009.