High Occupancy Resource Allocation for Grid and Cloud systems, a Study with DRIVE

Kyle Chard
School of Engineering and Computer Science
Victoria University of Wellington
PO Box 600
Wellington, New Zealand
kyle@ecs.vuw.ac.nz

Kris Bubendorfer
School of Engineering and Computer Science
Victoria University of Wellington
PO Box 600
Wellington, New Zealand
kris@ecs.vuw.ac.nz

Peter Komisarczuk
School of Computing
Thames Valley University
St Mary's Rd
Ealing, London, UK
peter.komisarczuk@tvu.ac.uk

ABSTRACT

Economic models have long been advocated as a means of efficient resource allocation, however they are often criticized due to a lack of performance and high overheads. The widespread adoption of utility computing models as seen in commercial Cloud providers has re-motivated the need for economic allocation mechanisms. The aim of this work is to address some of the performance limitations of existing economic allocation models, by reducing the failure/reallocation rate, increasing occupancy and thereby increasing the obtainable utilization of the system. This paper is a study of high performance resource utilization strategies that can be employed in Grid and Cloud systems. In particular we have implemented and quantified the results for strategies including overbooking, advanced reservation, just-in-time bidding and using substitute providers for service delivery. These strategies are analyzed in a meta-scheduling context using synthetic workloads derived from a production Grid trace to quantify the performance benefits obtained.

Categories and Subject Descriptors

D.4.7 [**Organization and Design**]: Distributed systems; D.4.8 [**Performance**]: Modeling and prediction—*Simulation*; C.2.4 [**Computer-Communication Networks**]: Distributed systems—*Grid Computing, Cloud Computing*

General Terms

Performance, Economics, Experimentation

Keywords

Economic Resource Allocation, Cloud Computing, Grid Computing

1. INTRODUCTION

Large scale distributed computing has changed with the adoption of utility computing models by commercial Cloud providers. Consumers now have a choice over where to submit tasks, weighing up price against the service levels delivered. This environment presents an opportunity to create high performance federated architectures that span both Grid and Cloud computing providers, effectively creating a global computing infrastructure on demand. High level meta-scheduling over Cloud providers and possibly over federated architectures necessitates the use of economic aware allocation mechanisms driven by the underlying allocation principles used by Cloud providers.

Proponents of computational economies generally cite allocation efficiency, scalability, incentives, and well understood mechanisms as reasons for using economics in distributed systems. However, adoption of economies in production systems has been limited due to criticisms relating to, amongst other things, poor performance, high latency, and high overheads.

Various strategies can be employed both through allocation protocols and by participants to increase resource occupancy and therefore optimize overall utilization. This paper looks at these strategies as a means to improve the performance of auction systems in the allocation of resources in a high performance computing context:

- **Overbooking**, allocates more resources than are available on the understanding that some resource requests will not be used due to "no-shows" or over estimated resource requirements.
- **Second Chance Substitutes**, if a provider reneges on its winning bid due to lack of resources (e.g. failure or overbooking) the next available bidder is chosen.
- **Advanced Time Flexible Reservation**, is used to give the resource providers more flexible scheduling options.
- **Just-In-Time (JIT)** bidding (or "sniping") is employed to minimize the effect of auction latency by committing/reserving resources to be allocated by the resource provider at the last possible moment.

HPDC'10, June 20–25, 2010, Chicago, Illinois, USA.

In addition to auctions, these strategies are applicable in other forms of non-economic and economic resource allocation. This paper presents analysis of these resource utilization strategies within the context of a market based Cloud or Grid. Each strategy has been implemented within the DRIVE meta-scheduler [10] and is analyzed using synthetic Grid workloads obtained by sampling production Grid traces.

The structure of this paper is as follows. Section 2 outlines potential high utilization strategies. Section 3 provides a brief overview of the DRIVE architecture. Section 4 quantifies experimental results of the strategies using synthetic workload traces and focusing on improved allocation performance. Section 5 presents an overview of related work. Finally future work is described in Section 6 and our contributions are summarized in Section 7.

2. HIGH UTILIZATION STRATEGIES

Economic allocation protocols have been widely studied in a distributed context with varying results. However, there has been little study of strategies which can be employed by participants to maximize occupancy and therefore utilization. In particular, providers may implement strategies regarding bidding time, and policies relating to oversubscribing resources. Protocol optimizations can also be used to reduce overhead and increase allocation efficiency. This section starts with a brief overview of economic resource allocation before providing a detailed examination of the allocation strategies that are the focus of this paper.

2.1 Economic Resource Allocation

Resource allocation is one of the most important (and difficult) tasks in Cloud/Grid systems. However, the task of spreading a finite group of resources across a user population is also inherent in human negotiation and forms the basis of modern economics. Economies are therefore equally well suited to distributed resource allocation. In particular they are typically scalable and adaptable to rapidly changing market conditions. They provide a well understood class of protocols used to provide effective decentralized decision making [33]. And they also provide incentives for participation in commercial environments.

Auctions are an efficient (pareto optimal) means of economic resource allocation due to their ability to establish market prices in an open market. There are four main types of auction protocol; the English, Dutch, Sealed-Bid, and Sealed-bid second price (Vickrey). The English auction is the common open outcry, ascending price, multiple bid protocol. The Dutch auction is an open outcry, descending price, single bid protocol. The Sealed-Bid auction is a sealed single bid, first price protocol in which all bids remain sealed until they are opened simultaneously. The Vickrey auction is also a sealed-bid protocol, except that the winning bidder pays the amount of the second best bid (second price). All four auction protocols yield the same return in private value auctions, hence selection of an auction protocol depends on particular properties. The Vickrey protocol is most appropriate for computational economies as the dominant strategy is for providers to bid their true value, with results in no counterspeculation, communication overhead is also minimal as bidders bid their true value in a single sealed bid.

While auction protocols provide an ideal low communication mechanism for determining the market price for a good and for producing optimal allocation they have some limitations in a high performance scenario due to their inherent latency. For this reason the worst case performance in an auction scenario is one composed of frequent short duration jobs.

2.2 Overbooking

There is potential for considerable latency in auction based resource allocation for a resource provider, from bidding to resource redemption, this latency greatly impacts utilization if resources are reserved while waiting for the result of the auction. In particular, an auction generally has a single winner, and multiple m losers. While the winner gains the eventual contract, there is no such compensation for the m losers of the auction process, and any resources r put aside during the auction will decrease the net utilization of the system by mr. From a providers perspective there is an opportunity to increase utilization and profit by participating in auctions that could exceed capacity in the knowledge that it is unlikely they will win all auctions on which they bid. Knowledge of existing bids can also be used in subsequent valuations, thus incorporating the possibility of incurring penalties for breaking agreements.

Overbooking has been shown to provide substantial utilization and profit advantages [28] due to "no shows" (consumers not using the requested resources) and over estimated task duration. Positive incentives reward users who deliver services and Negative incentives (penalties) can be used to discourage "bad" behavior. The contract model used in DRIVE uses penalties to discourage dishonoring an agreement.

While overbooking may seem risky it is a common technique used in yield management and can be seen in many commercial domains, most notably air travel [26, 31] and bandwidth reservation [3]. Most airlines routinely overbook aircraft in an attempt to maximize occupancy and therefore revenue by ensuring they have the maximum number of passengers on a flight. Without overbooking full flights often depart with 15% of seats empty [26]. Overbooking policies are carefully created and are generally based on historical data. Airlines acknowledge the possibility of an unusually large proportion of customers showing up, and include clauses to "bump" passengers to later flights and specify compensation to be paid [7]. Techniques used in bandwidth reservation have strong correlation to the type of scenario seen in distributed resource allocation. Typically network traffic is inconsistent and bursty by nature, as such consumers do not use all of their allocated bandwidth all the time. Telecommunications companies utilize this knowledge when selling bandwidth by overbooking bandwidth in an attempt to increase revenue, this technique ensures maximum utilization of the finite network resources deployed [3].

Overbooking attempts to balance the revenue lost due to unused capacity and the penalties imposed by breaking contracts with consumers. Essentially creating a maximization equation from which providers can determine the optimal amount of overbooking. However, the cost of non financial penalties such as unspecified damage to a providers reputation are difficult to calculate. Due to the widespread adoption of overbooking techniques there is substantial economic

theory underpinning appropriate strategy [11, 27].

2.3 Second Chance Substitute Providers

In a highly dynamic distributed environment resource state can change rapidly, limiting available capacity. After an auction if the winning provider cannot meet their obligations, it wastes resources to go through the auction process again when there is sufficient capacity available from non winning providers. In this case, we can give the losing bidders a *second chance* to win the auction, by re-computing the auction without the defaulting bidder. This optimization will increase efficiency at the expense of some protocol security, for example more bid information will be released in a secure protocol.

Allowing second chance providers can reduce the allocation failures from overbooking and therefore increase utilization of the system. One negative aspect of this approach is the potential for increased consumer cost, as they will now pay the price of a more expensive bidder. However this cost could be offset through compensation imposed on a violating party.

2.4 Advanced Reservations

Advanced reservation support in distributed resource allocation is desirable for performance predictability, meeting resource requirements, and providing Quality of Service (QoS) guarantees [8, 15, 25, 18, 19]. As Grid and Cloud systems evolve the task planning requirements become more complex, requiring fine grained coordination of interdependent jobs in order to achieve larger goals. Often tasks require particular resources to be available at certain times in order to run efficiently. For example, a task may require temporary data storage while executing and more permanent storage after completion. This is particularly important when the task forms an intermediary stage in a workflow. Storage must be available when the task completes and for the duration of subsequent processing stages. Tasks may also require coordinated execution due to dependencies between tasks. In addition to these consumer advantages, providers also benefit by being given flexibility in terms of task execution and the opportunity for advanced scheduling techniques to optimize resource usage.

2.5 Just-In-Time Bidding

All auction protocols have inherent latency in the allocation process. During this period resource state may change therefore invalidating a providers valuation (or bid). In general, there are two ways to minimize the effect of latency:

- Reducing the duration of the auction. The problem with this approach is that there is minimal time for providers to discover the auction and to compute their bids.
- Bid as late as possible. This has the main advantage that providers can compute their bids with the most up to date resource state. It also has the advantage of requiring resources to be reserved for a shorter time. The primary problem with this approach is time sensitivity, the auction can be missed if the bid is too late or experiences unexpected network delays.

In some environments for example open outcry protocols used in online auctions, JIT bidding is common and has additional strategic advantages for combating shrill bidding and incremental bidding. For sealed bid auctions, JIT bidding traditionally has been seen to have no advantages. As pointed out in this paper JIT bidding does have significant advantages for increasing utilization in systems using sealed bid auctions.

3. DRIVE

DRIVE (Distributed Resource Infrastructure for a Virtual Economy) [10, 9] is a distributed Web service enabled economic meta-scheduler that supports arbitrary economic protocols using a generic plug-in architecture. 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 VO. This architecture minimizes the need for dedicated infrastructure and distributes management roles and functions across participants. This distributed architecture is possible due to the deployment of secure allocation protocols which provide security guarantees in untrusted environments. The DRIVE implementation uses Globus Toolkit 4 (GT4) [14] compliant stateful WSRF [13] Web services.

Figure 1 shows an overview of the major DRIVE components. Each provider is represented by a DRIVE agent, that implements standard functionality including; reservations, policies, valuation and the plug-ins for the economic protocol. The DRIVE Agents use policies and valuation functions to price goods, policies are also used to make a range of decisions including load balancing and QoS guarantees. The DRIVE marketplace includes a number of independent services and mechanisms used for different aspects of the architecture including resource discovery, security, VO management, and contract (agreement) management.

DRIVE is independent of the type of task being executed and can be used to submit jobs in a Grid environment and VMs or services in a Cloud environment. The lifecycle of user interaction with DRIVE is shown in Figure 2. In DRIVE reverse auctions are used to auction consumer tasks, allowing providers to bid on suitable auctions. DRIVE makes use of a two phase contract model. An initial agreement is created as the result of an auction (phase 1), this agreement is then *hardened* into a binding contract (phase 2) before the resources are allocated to the consumer. The main motivation behind this separation is mitigating latency.

To support advanced reservation DRIVE makes use of a Reservation Service per resource provider, this service stores commitments made by providers in a secure manner. Reservation information is specified in the initial task description, allowing provider consideration during the allocation phase. The Reservation Service allows future commitments to be considered prior to negotiating for subsequent allocation, it also supports arbitrary advanced scheduling techniques to optimize task execution. When participating in resource negotiation the DRIVE Agent consults the local Reservation Service to check whether the task requirements can be met, any existing reservations are considered by the agent when valuing the request and scheduling algorithms can be used to determine the best fit for the task. Local policies control

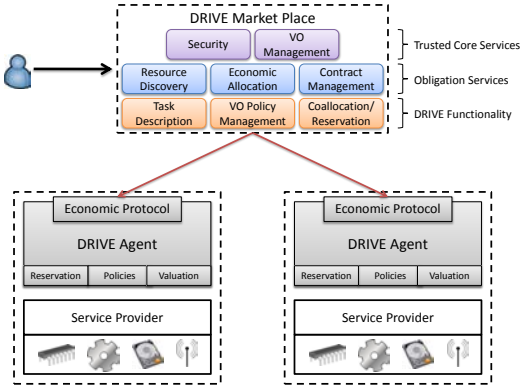

Figure 1: High level DRIVE Architecture. The DRIVE market place is composed of several independent services used to provide meta-scheduling functionality. *Trusted Core* services require dedicated infrastructure to ensure secure operation for example VO management and security. *Obligation* services are contributed by service providers for resource discovery, allocation and contract management. Service providers are represented by a DRIVE Agent which values resources and participates in auctions using arbitrary economic protocols.

Figure 2: DRIVE Lifecycle.

how this information is used in the valuation process.

4. EVALUATION

The following section analyzes the strategies outlined in Sections 2.2 – 2.5, using workload data based on a Grid trace from AuverGrid, a production Grid located in the Auvergne region of France. The AuverGrid project is part of the EGEE (Enabling Grids for E-science in Europe) project and uses LCG (Large hadron collider Computing Grid) middleware on its 475 computational nodes organized into 5 clusters (each has 112, 84, 186, 38, 55 nodes). This trace was chosen as it was the most complete trace available in the Grid Workloads Archive [16]. While this is a relatively small scale Grid the model obtained from the workload can be scaled up to be used in the analysis of DRIVE. Using the entire workload as a basis for the following experiments is infeasible due to the duration (1 year) and cumulative utilization (475 processors). To reduce the size, smaller traces can be generated by random sampling of the full trace to produce a synthetic trace.

4.1 Synthetic Workload

To create a high performance synthetic trace, the time based attributes of the trace have been reduced by a factor of 1000 (i.e a second of real time is 1 millisecond of simulated time). By reducing each parameter equally we maintain relativity between parameters and therefore do not effect the distribution. This also has the effect of producing high frequency short duration jobs, which is the worst case situation for auction performance. The performance analysis looks specifically at the allocation performance without considering the duration of the jobs. However, this reduction in time effects the ratio of interarrival time to auction latency which exaggerates the effect of auction latency in the following experiments. In the reduced format a 30 second auction period is a much greater percentage of the entire task than in the original workload, this will result in providers bidding on more auctions within the period of a single auction than would be the case in the original workload.

Figure 3 and Table 1 show a summary of the three different synthetic workloads used in this analysis. The resulting workloads, low utilization, medium utilization, and high utilization contain 2111, 4677, and 11014 jobs respectively, with each spread over almost 9 hours. The average job run time for each model is approximately 59 seconds with average job CPU utilization of 93-94%. The workloads aim to represent increasing overal utilisation and are based on the utilisation limit of the testbed. The low utilisation model has a peak of 86.55% overall utilisation which can be completely hosted in the test bed. The medium model slightly exceeds the capacity of the testbed with various points above the available capacity. The high utilisation model greatly exceeds the available capacity of the testbed with most values well above 100%. The arrival rate of tasks also increases with each workload as the number of jobs increases over the same period of time, the maximum arrival rate of the medium workload matches the maximum arrival rate seen in the original trace. The arrival rate for the high utilisation workload greatly exceeds the arrival rate of the original trace, with a maximum arrival rate more than double the peak arrival rate seen in the original trace. Each trace typifies the important characteristics of the original trace by providing similar or greater maximum throughput whilst mainting the duration of jobs relative to arrival time and one another.

4.2 Experimental Testbed

In these experiments the testbed is configured with 20 virtualized providers distributed over a 10 machine Grid (5 Windows Vista, 5 Fedora Core) connected by a gigabit Ethernet network. The machines each have Core 2 Duo 3.0 GHz processors with 4 GB of RAM. A single Auction Manager and Contract Manager are run on 1 host, with each allocated 1

	Jobs	Average Arrival Rate (jobs/hour)	Maximum Arrival Rate (jobs/hour)	Average Run Time (ms)	Average Job CPU (%)	Maximum Total Utilization (%)
Low	2111	234	357	58181	93.02	86.55
Medium	4677	519	814	59128	93.67	178.20
High	11014	1223	1892	59579	93.62	424.40

Table 1: Experiment Workload Characteristics.

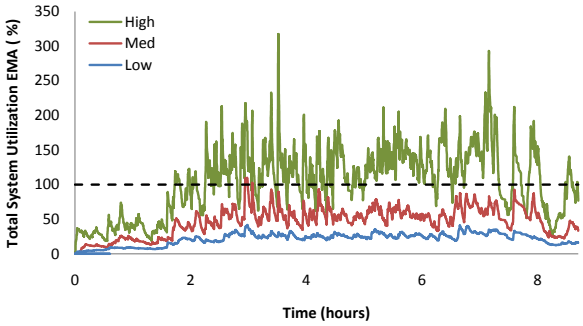

Figure 3: Total system utilization of the three synthetic workload shown using an Exponential Moving Average (EMA) with 0.99 smoothing factor. The dashed line indicates the total system capacity of our testbed.

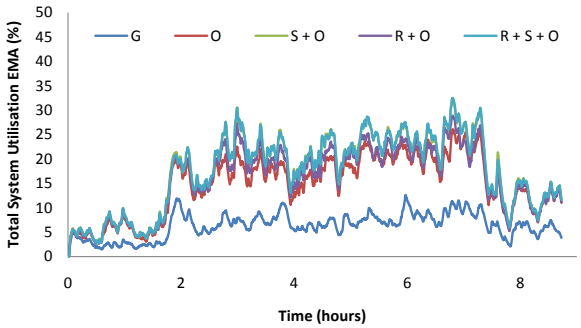

Figure 4: Exponential Moving Average of total system utilization over time for the Low workload.

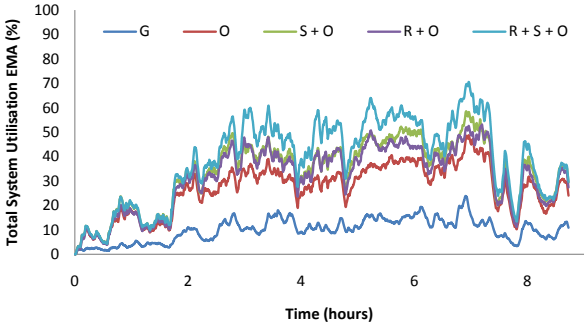

Figure 5: Exponential Moving Average of total system utilization over time for the Med workload.

GB of memory. The 20 providers each have 512 MB of memory allocated to the hosting container. A sealed bid second price protocol is used to allocate tasks and each provider implements a random bidding policy where values range between 0-100, prices are randomly generated irrespective of current or predicted load. The distribution is then analyzed using different bidding behavior such as overbooking, computing substitute providers, and using advanced reservations.

4.3 Strategy Evaluation

Table 2 outlines the results for each strategy when applied to each of the workloads. In particular the table outlines the number of auctions completed, contracts created, and overall system utilization. The various strategies are denoted: Overbooking (O), Second chance substitutes (S), Reservation (R). We also implement a Guaranteed (G) strategy, a baseline strategy against which to compare our high utilization strategies. Figures 4, 5 and 6 show an Exponential Moving Average (EMA) of total system utilization for each of the strategies on the low, medium and high workloads respectively. Individual system utilization graphs for each of the strategies on the high workload are presented in Appendix A.

4.3.1 Guaranteed Bidding Strategy

In the baseline configuration providers implement a guaranteed bidding strategy where every bid by a provider has a guarantee that there will be sufficient capacity. In this bidding strategy providers include contracts and any potential future contracts (outstanding bids) when calculating their projected utilization. No bid will be made if the calculated utilization exceeds the maximum capacity of the provider.

It is evident in each of the graphs that the overall utilization is extremely low, average utilization is 6.35%, 10.5% and 15.72% and the maximum utilization is 29.65, 43.65, and 53.8 for each of the three sample workloads, these results are well below the available capacity of the system. In all three workloads the rejection rate is substantial with only 34.41%, 26.75%, and 16.96% of tasks allocated respectively. As expected these rejections occur during auctioning and no contracts are rejected, as no provider should ever bid outside its means.

These results highlight the issue with bidding only on auctions a provider can guarantee to satisfy. A large number of tasks are rejected even though there is sufficient available

	Tasks	Auctions		Contracts			Average System Utilization	Max System Utilization
		Failed	Completed	Rejected	Substitutes	Allocated		
Low								
G	2111	1384.67	726.33	0	N/A	726.33 (34.41%)	6.35	29.65
O	2111	0	2111	326.33	N/A	1784.67 (84.54%)	15.03	61.90
S + O	2111	0	2111	1.33	375.67	2109.67 (99.94%)	17.71	86.55
R + O	2111	0	2111	204.33	N/A	1906.67 (90.32%)	16.18	66.80
R + S + O	2111	0	2111	0	225.33	2111 (100%)	17.72	86.55
Medium								
G	4677	3426	1251	0	N/A	1251 (26.75%)	10.50	43.65
O	4677	0	4677	1537.33	N/A	3139.67 (67.13%)	27.13	82.25
S + O	4677	14	4663	675	1489	3988 (85.27%)	34.28	97.90
R + O	4677	0	4677	1013	N/A	3664 (78.34%)	32.26	92.90
R + S + O	4677	0	4677	75.67	1146.67	4601.33 (98.38%)	39.43	99.05
High								
G	11014	9146	1868	0	N/A	1868 (16.96%)	15.72	53.80
O	11014	42	10972	6303	N/A	4669 (42.39%)	39.99	98.35
S + O	11014	325.33	10688.67	5052.67	2463.33	5636 (51.17%)	48.09	99.00
R + O	11014	30.33	10983.67	4843	N/A	6140.67 (55.75%)	56.67	99.00
R + S + O	11014	419.67	10594.33	3375.67	3091	7218.67 (65.54%)	68.91	99.05

Table 2: Summary of allocation rate and system utilization for each high utilization strategy.

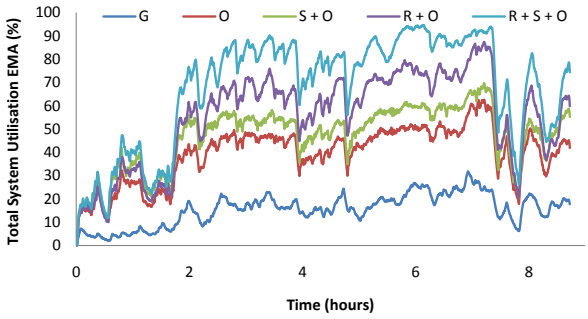

Figure 6: Exponential Moving Average of total system utilization over time for the High workload.

overall capacity. The reason for this is the number of concurrent auctions taking place and the latency between submitting a bid and the auction concluding. In this testbed there is a $\frac{1}{20}^{th}$ chance of winning an auction (when all providers bid) so for the duration of the auction all providers have reserved their resources with only a probability of 0.05 of actually winning. The advantage of a guaranteed approach is no auctions are won without sufficient available capacity, ensuring no contracts are ever rejected. Additionally the burden on the allocation infrastructure is reduced as no attempts are made to establish contracts which cannot be honored.

In this strategy the key factor to consider is the ratio of auction latency to interarrival time. If the auction latency was reduced or the frequency of tasks being submitted was reduced, the number of tasks allocated and total utilization would improve as bidders have a clearer picture of utilization when bidding on subsequent auctions.

These results motivate overbooking, as system utilization for each of the providers can be increased when factoring in the possibility that only 1 of the bidders will effectively win an auction.

4.3.2 *Overbooking Strategy*

As discussed in Section 2.2 the potential benefits of bidding beyond provider capacity can result in increased utilization and profit, assuming the penalties for breaking contracts are sufficiently small. Using an overbooking strategy providers compute their utilization considering only established contracts and therefore compute bids irrespective of any outstanding bids. In the high workload sample max system utilization approaches the maximum available capacity. The average utilization and percentage of tasks allocated for all three workloads is more than double that of the guaranteed strategy showing the value of overbooking. However, there is still substantial underutilization of the system when viewed across all providers.

In each workload very few auctions fail (e.g 42/11014 in the high workload) as providers only reach close to maximum capacity for a short period of time. The issue with overbooking however is the number of auctions completed that are then unable to be converted into contracts, this approaches 60% of all tasks in the high workload, 33% in the medium workload, and 15% in the low workload. The number of contracts unable to be established effects system performance as the auction and contract negotiaton processes are wasted and can be considered pure overhead. One way to optimise this process is to reuse the existing auction information rather than re-executing the whole auction process.

4.3.3 *Second Chance Substitutes and Overbooking*

In the event of a rejected contract, a losing bidder can be offered a second chance to seamlessly satisfy the auction. Allowing second chance substitutes reduces the contract failures due to the overbooking strategy and therefore increases utilization of the system.

	S + O Average Depth	R + S + O Average Depth
Low	1.46	1.23
Medium	2.29	2.04
High	2.55	2.76

Table 3: Average number of substitutes considered.

Average utilization is shown to be improved from the overbooking strategy by up to 26% (low - 17%, med - 26%, high - 20%) and task allocation is increased to 99.94%, 85.27%, and 51.17% for the low, medium, and high workloads respectively. These results show a large improvement from the previous strategies. Interestingly the depth of substitute providers needed to be consulted is less than 3 on average for each workload (Table 3) indicating that there is sufficient reserve capacity in the system. This information could be used to define policies regarding the maximum number of subsitute providers able to be consulted.

4.3.4 Reservations and Overbooking

Reservations are widely used in large scale distributed and parallel systems as a means of coordinating resource usage, reservations have been shown to increase performance and provide flexibility. In the AuverGrid workload trace there is no explicit execution windows so in order to test this strategy and analyze the effect on allocation and utilization we define an execution window for each task as 50% of the task duration. This provides a reasonable approximation of a reservation window allowing providers room to schedule tasks (sensitivity analysis is shown in Section 4.4). In this experiment providers implement a simple First Come First Served scheduling policy. There are better techniques for scheduling reservations, such as backfilling [20], however, FCFS is sufficient to provide comparability for these experiments.

Each provider again uses an overbooking strategy in this experiment due to the considerable utilization improvements seen over guaranteed bidding. Each task is submitted with a flexible reservation window, defining start time, end time and duration. The end time has been extended by 50% of the duration.

As the density of the workload increases the improvement gained by using reservations is greater than that of computing substitutes. For the low and medium workloads (90.32% and 78.34%) the auction performance is less than that of using substitutes, however in the dense high workload the improvement exceeds the gains made using substitutes peaking at 55.75% of all tasks allocated. The same pattern is seen in average utilisation as it is proportional to the number of tasks allocated.

4.3.5 Reservations, Substitutes, and Overbooking

The final configuration combines the use of reservations with the ability to compute substitute providers and overbook resources. As expected this combination gives the best results for each of the workloads, with 100% of low workload tasks allocated, 98.38% of the medium workload allocated and 65.54% of high workload allocated. In the low and medium workload no auctions fail as providers are not fully utilised for long periods of time, 75.67 contracts are rejected in the medium workload due to short periods of fully utilised providers. Due to the nature of the high workload which consistently exceeds the capacity of the test bed, 65% of tasks allocated represents very high degree of workload allocation, close to the maximum obtainable allocation.

4.3.6 Just-In-Time Bidding

Auction latency effects the bidding process due to agents bidding without absolute knowledge. In each of the previous experiments (except guaranteed bidding) the number of contracts rejected by providers is large. This occurs as providers bid to host a task, but by the time the auction completes the providers state has changed and required resources are unavailable.

Auction Latency

To analyze the effect of auction latency on allocation performance we use a series of symmetric workloads allocated over a 10 host virtualized testbed. Table 4 details each of the workloads used, each has increasing individual job resource requirements (and therefore increasing maximum system utilization). The total system utilization for each workload is shown in Figure 7. In these workloads tasks are submitted every 5 seconds for a duration of 2 minutes each. Individual task utilization ranges from 20% of a single provider allowing 5 tasks to be hosted simultainously on a single host through to 50% which allows only 2 tasks to be hosted on a single provider simultainously. When using 20% tasks all tasks can be hosted using only 50% of total cumulative provider capacity, increasing task utilisation increases the total system capcity used by the workload up to 50% which exceeds the total capacity of the testbed.

Workload	Job Size (relative to one host)	Maximum System Utilization
W1	20%	48%
W2	25%	60%
W3	33%	79.2%
W4	50%	120%

Table 4: Synthetic workload characteristics for auction latency.

Figure 8 shows the number of contracts rejected for each workload when increasing the auction duration. The graph also includes the auction failure percentage for workload W4 as this workload exceeds provider capacity and therefore some auction must fail. As expected when the auction duration (latency) is less than the period in which tasks are submitted no contracts are rejected for any of the workloads. For workload W4 nearly a quarter of the auctions fail because the providers do not have sufficient capacity. For each of the workloads the rejection percentage increases with the increase in auction duration highlighting the effect of auction latency. The number of auction failures exhibited in workload W4 decrease due to increased underutilization of the system.

JIT Bidding

Having demonstrated the negative effect of auction latency

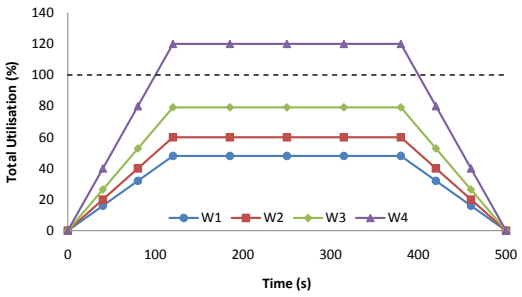

Figure 7: System Utilization for each workload used to test latency.

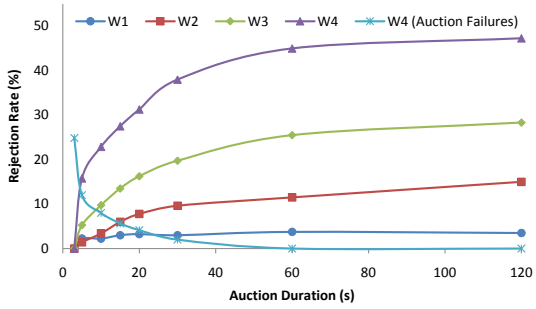

Figure 8: Contract (and Auction) rejection rate for each of the synthetic workloads.

Time Before Auction Close	S + O Average Depth	R + S + O Average Depth
Medium		
30 seconds	2.29	2.04
10 seconds	1.81	1.69
3 seconds	1.68	1.49
0.5 seconds	1.13	1.23
High		
30 seconds	2.55	2.76
10 seconds	1.98	2.16
3 seconds	1.77	1.74
0.5 seconds	1.19	1.29

Table 5: Average number of second chance substitute providers considered as JIT bidding gets closer to auction close.

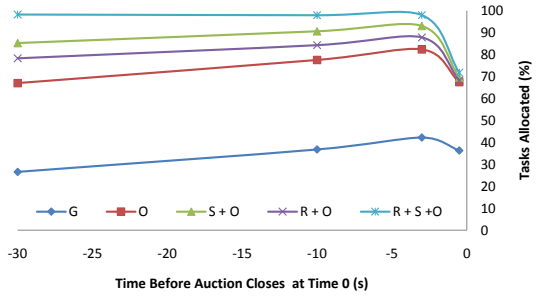

Figure 9: JIT Bidding for each strategy on the medium workload.

on the workloads in table 4, we conducted a series of experiments to investigate the effect of bidding closer to the auction closing time on the various strategies outlined in Sections 2.2 - 2.5. Figure 9 and Figure 10 show the increased allocation when bidding closer to the close of the auction for the medium and high utilization workloads respectively. For both workloads the number of tasks allocated increases by approximately 10% for each strategy up until a point of saturation - at which point bids are not being received before the auction closes. The two strategies employing second chance subsitutes in both workloads do not exhibit as much of an improvement, as auctions will not fail as long as alternative substitutes are available. Table 5 shows the average number of substitutes considered for each strategy as JIT bidding gets closer to the auction close. Although the utilization improvements are smaller for the second chance strategies, the number of substitutes consdidered decreases as bidding occurs closer to the auction close. This is beneficial as it reduces the overhead required to compute second chance providers.

4.4 Reservation Window

The results shown in the previous section use an artificial flexible reservation window defined as 50% of the job duration. Figure 11 presents a sensitivity analysis that shows the effect of altering this articifial window as a percentage of job duration for each of the workloads/strategies using reservations. The low workload with reservations and second chance substitutes is not included as there are only 1.3 rejections without any reservation window. As expected the rejection rate decreases for each strategy with an increased reservation window. For all combinations the greatest decrease occurs between not having reservations and a 10% reservation window, this is indicitive of the short duration jobs and the finely packed schedules the providers are operating with. Even a minimal increase in job reservation window gives the flexibility to greatly increase allocation. The strategies with both reservations and substitutes decrease more gradually than those with only reservations, this is probably due to the fact that the reservation and substitute series are closer to maximum utilisation and therefore have fewer oportunities to improve auction performance. As the degree of time flexibility increases, we expect all algorithms tend to 100%, the rate of increase depends on the workload.

5. RELATED WORK

Computational economies have long been "posited" to allocate resources in both centralized and decentralized systems. The earliest published market was the futures market [30] in 1968, which allowed users to bid for compute time on a shared departmental machine. Over time these architectures were extended from single processor economies through to distributed computational economies with the introduction of systems such as Spawn [32] and Enterprise [17].

There are many current examples of mature economically enabled brokers, meta-schedulers and distributed architec-

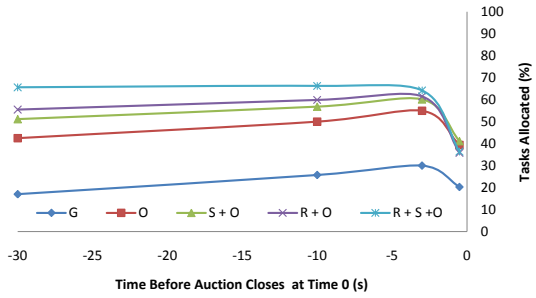

Figure 10: JIT Bidding for each strategy on the high workload.

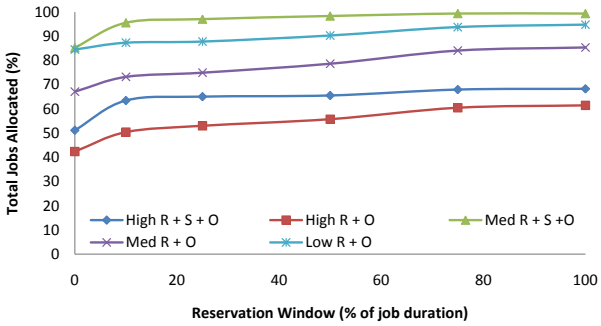

Figure 11: Allocation percentage when altering the reservation window as a percentage of job duration.

tures such as Nimrod/G [6], DRIVE and SORMA [21]. A range of economic protocols have been deployed in each of these systems, all offer different approaches to contracts, security and architecture.

Overbooking has been developed in computational domains as a way to increase utilization and profit [28, 4]. In [28] overbooking is used to compensate for "no shows" and poorly estimated task duration. In [4] backfilling is combined with overbooking to increase provider profit, overbooking decisions are based on SLA risk assessment generated from job execution time distributions.

GARA (Globus Architecture for Reservation and Allocation) [15] was one of the first projects to define a basic advanced reservation architecture supporting QoS reservations over heterogeneous resources. Since this time other schedulers have evolved to support advanced reservations, such as Catalina [1], Moab/Maui [12], Sun Grid Engine [29], Platform Load Sharing Facility (LSF) [23] and Portable Batch System (PBS) Pro [22]. Reservation aware schedulers have been shown to improve system utilization due to the additional flexibility specified by some consumers, additionally these architectures have realized various reservation aware scheduling algorithms [20].

Various papers have looked at Last minute bidding and "sniping" in the context of open outcry online auctions [24, 2]. Typical motivation for last minute bidding is to combat "shill bidders" (fake bidders raising the price) and incremental bidding (bidding in increments rather than bidding ones true value or proxy bidding). Just-In-Time (JIT) bidding for sealed bid auctions was proposed in our earlier work [5] as a means of reducing the effect of auction latency in distributed auctions. However, no quantifiable analysis was performed at the time.

The first use of second chance substitutions in a two phase contract structure was in our previous paper [5] and was presented as a means of reducing coallocative failures. We have adapted and generalized this mechanism to directly support overbooking to increase utilization. As pointed out for JIT there was no quantifiable analysis of the second chance mechanism performed at the time.

The general focus of prior work has been on economic efficiency. In particular existing systems using auctions suffer from significant latency and consequently reduced utilization. In addition techniques such as advanced reservation, have in general been approached from a consumers perspective rather than concentrating on the flexibility given to providers, an exception to this is our previous work with Netto and Buyya in [20].

The four utilization techniques presented have not been collectively examined from the view of improving auction performance in any prior work.

6. FUTURE WORK

In this paper we have analyzed the performance side of the strategies from Sections 2.2 – 2.5, however, the economic aspects are just as important for Cloud and Grid based systems. We aim to explore the economic performance of these strategies analyzing profit margins using different bidding and penalty models. We have evaluated these strategies with respect to a single class of economic allocation, however strategies such as overbooking, second-chance substitution and Just-In-Time bidding are just as applicable in other domains. We aim to explore more general use of these techniques as a means of increasing allocation performance.

7. CONCLUSIONS

The emergence of commercial cloud providers has re-motivated the need for efficient, responsive and economically enabled resource allocation within high performance computing. Outsourcing workload amongst viable resource providers is seen as increasingly complex due to the different allocation models supported. Using high level meta-schedulers to provide transparency over Grid/Cloud providers, abstracting different allocation protocols and execution methods is effective, however meta-schedulers must support the allocation principles used by underlying providers.

Economic allocation has been stereotyped as a low performance allocation solution due to the overheads and latency in the allocation process. This paper has presented several economic strategies that can be employed in a distributed computational economy to increase occupancy and optimize utilization. Specifically our contributions are:

- The implementation and testing of various occupancy and utilization enhancing economic strategies in a meta-scheduling context,
- A number of synthetic workloads developed from real workload traces, used to produce severe allocation conditions for auctions, and
- Quantified the performance benefits of using the high occupancy scheduling strategies in a variety of workload scenarios.

8. ACKNOWLEDGMENTS

The AuverGrid traces were kindly provided by the AuverGrid team (special thanks to Dr. Emmanuel Medernach), the owners of the AuverGrid system.

9. REFERENCES

[1] Catalina scheduler. www.sdsc.edu/catalina/ [Accessed April 2010].

[2] P. Bajari and A. Hortacsu. Economic insights from internet auctions. *Journal of Economic Literature*, 42:457–486, 2004.

[3] R. Ball, M. Clement, F. Huang, Q. Snell, and C. Deccio. Aggressive telecommunications overbooking ratios. In *Proceedings of the IEEE International Conference on Performance, Computing, and Communications*, pages 31–38, 2004.

[4] G. Birkenheuer, A. Brinkmann, and H. Karl. The gain of overbooking. In *Lecture Notes in Computer Science (LNCS), Job Scheduling Strategies for Parallel Processing*, volume 5798/2009, pages 80–100, 2009.

[5] K. Bubendorfer. Fine grained resource reservation in open grid economies. In *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing*, page 81, Washington, DC, USA, 2006. IEEE Computer Society.

[6] R. Buyya, D. Abramson, and J. Giddy. Nimrod/G: an architecture for a resource management and scheduling system in a global computational grid. In *Proceedings of the Fourth International Conference/Exhibition on High Performance Computing in the Asia-Pacific Region, 2000.*, volume 1, pages 283–289, 2000.

[7] M. Campbell. Resource overbooking in a market-oriented grid resource allocation architecture. Honours thesis, School of Mathematicas, Statistics and Computer Science, Victoria University of Wellington, 2008.

[8] C. Castillo, G. Rouskas, and K. Harfoush. Efficient resource management using advance reservations for heterogeneous grids. In *IPDPS'08: Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, pages 1–12, April 2008.

[9] K. Chard and K. Bubendorfer. A distributed economic meta-scheduler for the grid. In *CCGRID '08: Proceedings of the Eighth IEEE International Symposium on Cluster Computing and the Grid*, pages 542–547, Washington, DC, USA, 2008. IEEE Computer Society.

[10] K. Chard and K. Bubendorfer. Using secure auctions to build a distributed meta-scheduler for the gird. In R. Buyya and K. Bubendorfer, editors, *Market Oriented Grid and Utility Computing*. Wiley Press, New York, USA, 2009.

[11] C. Chiu and C. Tsao. The optimal airline overbooking strategy under uncertainties. In M. G. Negoita, R. J. Howlett, and L. C. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems*, pages 937—945. Springer-Verlag, 2004.

[12] Cluster Resources. Maui Cluster Scheduler. http://www.clusterresources.com/products/maui-cluster-scheduler.php/ [Accessed April 2010].

[13] K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe. The WS-Resource Framework. Technical report, Globus, 2004. http://www.globus.org/wsrf/specs/ws-wsrf.pdf.

[14] I. Foster. Globus toolkit version 4: Software for service-oriented systems. *Journal of Computational Science and Technology*, 21(4):523–530, 2006.

[15] I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt, and A. Roy. A distributed resource management architecture that supports advance reservations and co-allocation. In *IWQoS'99: Proceeings of the Seventh International Workshop on Quality of Service*, 1999.

[16] A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L. Wolters, and D. H. J. Epema. The grid workloads archive. *Future Generation Computer Systems*, 24(7):672–686, 2008.

[17] T. W. Malone, R. E. Fikes, K. R. Grant, and M. T. Howard. Enterprise: A market-like task scheduler for distributed computing environments. In *The Ecology of Computation*, pages 177–205. Elsevier Science Publishers (North-Holland), 1988.

[18] A. S. Mcgough, A. Afzal, J. Darlington, N. Furmento, A. Mayer, and L. Young. Making the grid predictable through reservations and performance modelling. *The Computer Journal*, 48(3):358–368, 2005.

[19] T. Meinl. Advance reservation of grid resources via real options. In *Proceedings of the 10th IEEE Conference on E-Commerce Technology and the 5th IEEE Conference on Enterprise Computing, E-Commerce and E-Services*, pages 3–10, July 2008.

[20] M. S. Netto, K. Bubendorfer, and R. Buyya. SLA-based Advance Reservations with Flexible and Adaptive Time QoS Parameters. In *Lecture Notes in Computer Science (LNCS), proceedings of the International Conference on Service Oriented Computing*, volume 4749, pages 119–131, Vienna, Austria, September 2007. Springer Verlag.

[21] D. Neumann, J. Stößer, A. Anandasivam, and N. Borissov. SORMA - Building an Open Grid Market for Grid Resource Allocation. In *Lecture Notes in Computer Science: The 4th International Workshop on Grid Economics and Business Models (GECON 2007)*, pages 194–200, Rennes, France, 2007.

[22] B. Nitzberg, J. M. Schopf, and J. P. Jones. Pbs pro: Grid computing and scheduling attributes. In *Grid resource management: state of the art and future trends*, pages 183–190. Kluwer Academic Publishers, 2004. 976127.

[23] Platform Computing. Platform LSF.

http://www.platform.com/workload-management/platform-lsf [Accessed April 2010].

[24] 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. *American Economic Review*, 92(4):1093–1103, 2002.

[25] M. Siddiqui, A. Villazón, and T. Fahringer. Grid capacity planning with negotiation-based advance reservation for optimized qos. In *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, page 103, New York, NY, USA, 2006. ACM.

[26] B. C. Smith, J. F. Leimkuhler, and R. M. Darrow. Yield Management at American Airlines. *INTERFACES*, 22(1):8–31, 1992.

[27] J. Subramanian, S. Stidham, Jr., and C. J. Lautenbacher. Airline yield management with overbooking, cancellations, and no-shows. *Transportation Science*, 33(2):147–167, 1999.

[28] A. Sulistio, K. H. Kim, and R. Buyya. Managing cancellations and no-shows of reservations with overbooking to increase resource revenue. In *CCGRID '08: Proceedings of the Eighth IEEE International Symposium on Cluster Computing and the Grid*, pages 267–276, Washington, DC, USA, 2008. IEEE Computer Society.

[29] Sun Microsystems. Sun Grid Engine. http://gridengine.sunsource.net/ [Accessed April 2010].

[30] L. E. Sutherland. A futures market in computer time. *Communications of the ACM*, 11(6):449–451, 1968.

[31] Y. Suzuki. An empirical analysis of the optimal overbooking policies for us major airlines. *Transportation Research Part E: Logistics and Transportation Review*, 38(2):135–149, 2002.

[32] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta. Spawn: A distributed computational economy. *IEEE Transactions on Software Engineering*, 18(2):103–117, 1992.

[33] M. P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1(1):1–23, 1993.

APPENDIX

A. WORKLOADS

This appendix outlines complete system utilization for each of the strategies presented in Section 2.2 - 2.5 using the high utilization workload. Figure 12 shows the total system utilization represented by the synthetic high workload. Figures 13, 14, 15, 16, and 17 show the total utilization across all 20 providers over the period of the high utilization workload. The graphs show the utilization increasing as each strategy is added approaching the maximum capacity of the testbed.

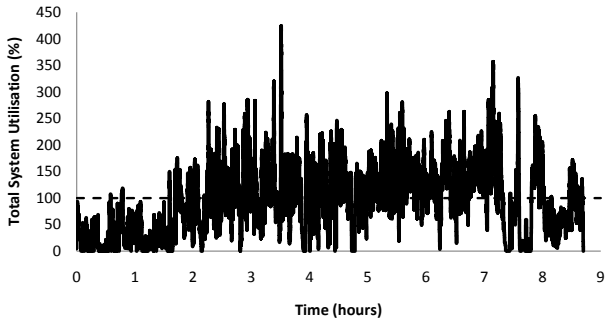

Figure 12: Total system utilization of the high workload, the dashed line indicates the total capacity of the testbed.

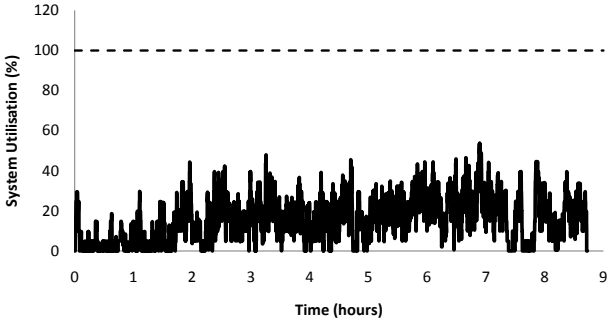

Figure 13: Total system utilization using the Guaranteed strategy on the high workload.

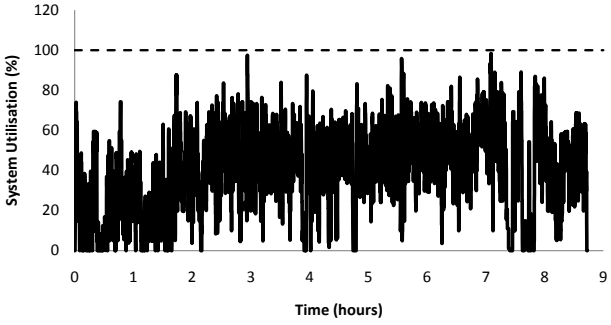

Figure 14: Total system utilization using the Overbooking strategy on the high workload.

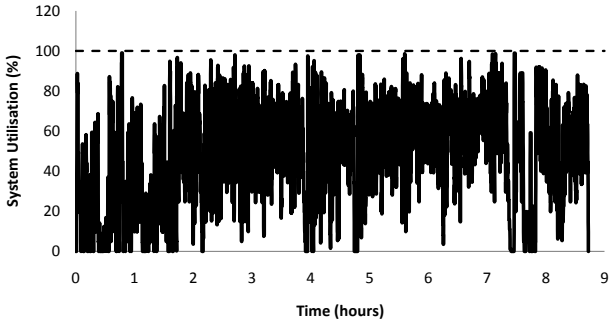

Figure 15: Total system utilization using the Second Chance Substitutes and Overbooking strategies on the high workload.

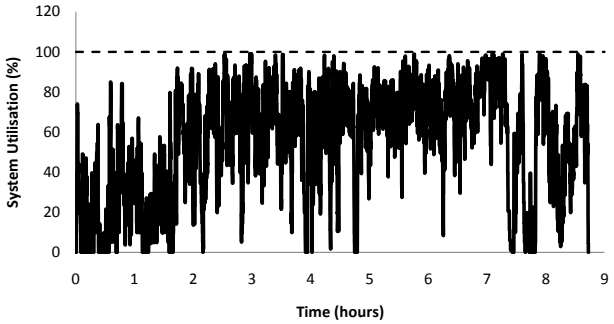

Figure 16: Total system utilization using Reservations and the Overbooking strategy on the high workload.

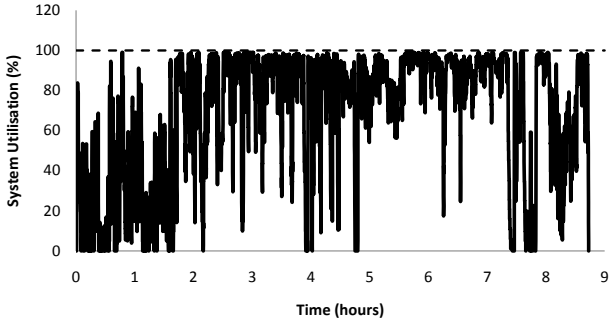

Figure 17: Total system utilization using Reservations, Second Chance Substitutes and Overbooking strategies on the high workload.