

Prediction-Based Enforcement of Performance Contracts

Thomas Sandholm¹ and Kevin Lai²

¹ KTH – Royal Institute of Technology

Center for Parallel Computers

SE-100 44 Stockholm, Sweden

sandholm@pdc.kth.se

² Hewlett-Packard Laboratories

Information Dynamics Laboratory

Palo Alto, CA 94304, USA

kevin.lai@hp.com

Abstract. Grid computing platforms require automated and distributed resource allocation with controllable quality-of-service (QoS). Market-based allocation provides these features using the complementary abstractions of proportional shares and reservations. This paper analyzes a hybrid resource allocation system using both proportional shares and reservations. We also examine the use of price prediction to provide statistical QoS guarantees and to set admission control prices.

1 Introduction

Grid applications traditionally run on dedicated machines, with a fixed performance level that depends on the hardware configuration. In this model, the main source of uncertainty in predicting job deadlines is the queue waiting time. As a solution to heterogeneity, and low resource utilization various virtualized platforms are emerging, such as Xen, VMWare, and VServer. In a virtualized Grid, where the performance level is configured dynamically based on job requirements and current demand, the main source of uncertainty is the risk of not being allocated enough capacity. The allocation decisions are complicated by the scale, and distribution of the Grid resources, and the vast variability and complexity of the job requirements. Therefore, it is not feasible to make these decisions manually using static configurations or policies.

Market-based allocation is one form of allocation that is automated, distributed, and provides QoS. Market-based allocation supports two primary resource abstractions: proportional shares and reservations. A pure proportional share allocator always admits new resource requests and continuously reallocates resource shares in response to the current load. This fully utilizes the resources and always admits well-funded resource requests, but may cause an earlier request to fail a minimum resource requirement. In contrast, a pure reservation allocator fixes resource shares at purchase time. Admitted resource requests in a reservation system will always (modulo failure) meet their resource requirements, but sometimes utilization is low, and sometimes well-funded requests will be rejected admittance.

In this paper, we examine a hybrid system that mixes both proportional share and reservation abstractions to achieve the best of both worlds: satisfying quality-of-service requirements for some applications while maximizing utilization and providing resource availability for latecomers. Using simulation, we explore how such a hybrid system performs for different workloads.

In addition, we examine how prediction algorithms affect the result. Prediction of future load is critical to efficient resource allocation. Proportional share allocators require it so that purchasers can get statistical QoS guarantees. Reservation allocators require it to set the prices for reservations. However, the effect of universal prediction on a system is not obvious. For example, if low prices are predicted for a particular hour of CPU time, then many resource consumers may try to buy it, thus ruining the accuracy of the prediction.

We base this analysis on previous work on predicting demand in computational markets [1, 2], where we evaluate different prediction techniques to give accurate percentile bounds for expected demand for arbitrary probability distributions. We assume here that we have an approximation for the cumulative distribution function (CDF) of the demand. Furthermore, we assume a computational market where proportional share resource allocations are enforced (e.g., Tycoon [3])

Our contribution in this work is twofold: 1) we highlight and visualize issues with statistical guarantees in performance contracts using simulations, and 2) we propose and implement a solution to these issues using contract admission control.

The paper is organized as follows: Section 2 provides an overview of the mathematical models used to analyze and simulate our resource allocation scenario, Section 3 presents and discusses the design and results of our simulations, Section 4 reviews related work, and finally Section 5 sums up our findings with some concluding remarks.

2 Model

2.1 Statistical Guarantees

We are interested in analyzing what bids individually rational resource consumers should place on their tasks, given that they need a certain performance level to finish within a deadline. Different guarantee-levels can then be compared based on the utility they yield to consumers.

To formalize the model we use the following standard probability theory notations:

$$x \in X, P(x) = P(X = x)$$

$$D(x) = \int_{x_{\min}}^x P(\varepsilon) d\varepsilon$$

where P is the probability function (a.k.a. PDF), and D the probability distribution function (CDF). To find performance levels based on guarantees it is also useful to look at the inverse of the distribution function, or percent point function (PPF), defined as:

$$D^{-1}(D(x)) = x$$

The proportional share resource allocation model is defined as:

$$q = \frac{b}{b + c}$$

where q is the performance level or QoS in terms of resource share $(0, 1)$, given a consumer's bid, b , and a measured price, c , of a resource. The price is the sum of all existing bids on the resource.

A rational consumer would hence bid

$$b = \frac{cq}{1 - q}$$

for any measured price, c , to maintain a service level q . However, in a competitive computational market the price adjusts dynamically to the resource demand, and can thus be viewed as a random variable C , which changes continuously over time. Since, q depends on c it can also be seen as a random variable, Q . The guarantee of delivering a certain QoS level to the consumer, g , will be expressed in terms of this random variable Q .

$$q \in Q, c \in C$$

$$g = P\left(\frac{b}{b + C} > q\right) = P\left(C < \frac{b}{q} - b\right) = D_c\left(\frac{b}{q} - b\right)$$

where D_c is the price distribution function. Now using the inverse of the price distribution function we can calculate the bids to place given a QoS level and a guarantee

$$D_c^{-1}(g) = b\left(\frac{1}{q} - 1\right)$$

which gives

$$b = \frac{D_c^{-1}(g)}{\frac{1}{q} - 1} = \frac{D_c^{-1}(g)q}{1 - q}$$

The intuition behind this is that the probability of getting a service level greater than a certain value is the same as the probability of the price being below a particular value, or

$$P(Q > q) = P(C < c)$$

2.2 Admission Control

Now, we would like to offer an admission control service with more than a statistical guarantee for an additional fee. We calculate this new price as:

$$b' = \frac{D_c^{-1}(g + r)q}{1 - q}$$

where b' is the price a user needs to pay to get share q with guarantee g , and r is the fee parameter. Note that the fee is not simply added to or multiplied with the bid, but

included in the percent point calculation of the price. This ensures that the admission control service is more expensive when there is a high price difference in offering a higher guarantee, in order to account for the expected loss the provider makes when refusing new consumers due to admission control.

In our model, a share of a resource can be requested with either an absolute guarantee paying the admission control fee, or with a statistical guarantee paying the spot (current) market price. The admission controller makes sure that no request is accepted that violates previously admitted requests with absolute guarantees. Whether a violation would occur as a result of admitting a new request is determined by enumerating and evaluating bids and required shares for all active previously admitted requests for the same resource. Consequently, all requests for the resource will need to go through the same admission control path in order to ensure reservation-like guarantees. We note that price volatility in this model is paid for directly by the user, and the admission controller operates in the interest of the provider to keep the prices at a higher level to compensate for not being able to preempt existing low-paying allocations in the event of higher-paying requests. Alternatively, the admission controller could be separated entirely from the resource being provisioned and operate like an insurance agent to put in spot market bids on the resources, and then dynamically update the bids using an insurance fund. For simplicity of evaluation and implementation we chose not to study this more advanced form of admission control here.

If strict admission control is implemented for all users only one guarantee level can be provided. To allow any number of guarantee levels, we strictly enforce only a portion of the allocation request, and make the remaining portion subject to statistical guarantees.

3 Simulations

In our simulations we study the price guarantees and dynamics, using varying levels of statistical and admission control guarantees offering multiple competing consumers service-level guarantees under different work-load situations.

The setup is as follows. A number of concurrent competing consumers submit jobs with inter-arrival-times (IAT) from an exponential distribution and performance requirements drawn from a normal distribution. The performance requirement is obtained from the number of work units that needs to be completed within a given deadline, and it translates to the share, q , of a resource that the consumer will bid for.

To simulate the fact that some users do not care about guarantees, but are only interested in best-effort service we designate a certain proportion of the work-load to be *best-effort* jobs. Those jobs are submitted by calculating the bid a consumer should spend based on the assumption that the price stays at the current mean value. This technically gives the guarantee, $g = 0.5$. All other jobs try to get a guarantee $g \geq 0.6$, and we then measure the guarantees obtained and the price paid under different levels of best-effort jobs. Each run of the simulated workload was configured with a single guaranteed service level, i.e. all jobs competing with best-effort jobs in a simulation run request the same guarantee level. We then measure and graph the average bid and

obtained guarantee for a group of eight subsequent jobs (based on completion time) requesting a certain guarantee level.

The guarantee obtained in a simulation run is calculated by measuring whether the current share of a job is greater than the required share each second that the job runs. The proportional share allocations are also recalculated each second. We configured the mean of the overall required shares to be higher than the available capacity in order to simulate resource contention and consumer competition.

The general simulation configuration is summarized in Table 1 and Table 2. #C is the number of consumers, #J is the number of jobs per consumer, t the deadline, and BE is the portion of best-effort jobs.

Table 1. General Configuration (All times in seconds)

#C	#J	q	IAT	g	t
4	32	$N(5.5/16, .25)$	$Exp(8)$	(0.6, 0.9)	16

Table 2. Individual Simulation Configuration

Simulation	BE	Strategy
I	0.75	statistical guarantee
II	0.25	statistical guarantee
III	0.25	admission control

3.1 Simulation I: 75% Best-Effort with Statistical Guarantees

In the first simulation we look at a work-load with a high portion of best-effort jobs (75%) that can make way for the smaller portions of jobs requiring guarantees. No admission control is used in this simulation, just statistical guarantees. In Figure 1, where each marked point is an average of eight subsequently completing jobs, we see that there is a clear separation left to right and from bottom to top between the different guarantee levels. Jobs with higher guarantee requirements were bidding more (x-axis) and also obtained a higher guarantee (y-axis). This tells us that statistical guarantees worked well when giving consumers their guarantees in this scenario.

We also study the price dynamics. In Figure 2 we can see that the price is stationary although it has a high variance. Note that the first 100 seconds are not shown because this time is used to bootstrap the simulations. In Figure 3 the variation is high but stable, the skew is positive and varies between 0 and 0.5. A positive skew of the price distribution means that more jobs pay a higher price for a guarantee level than would

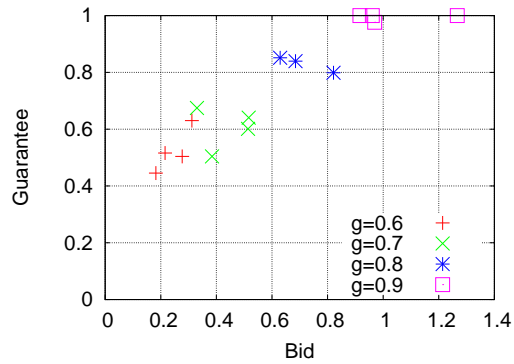


Fig. 1. Bids vs. obtained guarantees for statistical guarantees and 75% Best-Effort jobs

normally (e.g. by Gaussian distribution models) be expected from the mean and the variance. Skewness can thus be viewed as an indication of how risky the computational market is[4].

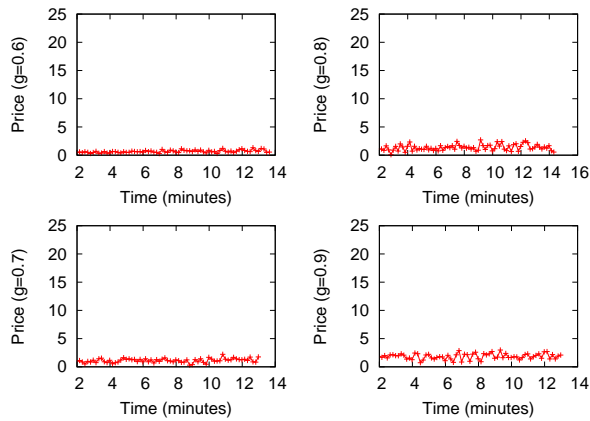


Fig. 2. Price over time for statistical guarantees and 75% Best-Effort jobs

3.2 Simulation II: 25% Best-Effort with Statistical Guarantees

We now decrease the portion of best-effort jobs to 25% and consequently the portion of jobs requiring guarantees increases to 75%. In Figure 4 we can see that the guarantees obtained for the different guarantee-levels are seemingly randomly layered. The

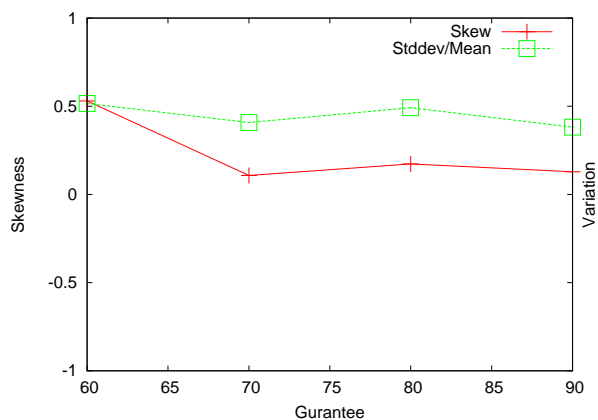


Fig. 3. Price skewness and variation for statistical guarantees and 75% Best-Effort jobs

higher bids and requested guarantees do not necessarily yield a higher obtained guarantee as before. This can be explained by the load being too high for the provider to offer everyone the required guarantees.

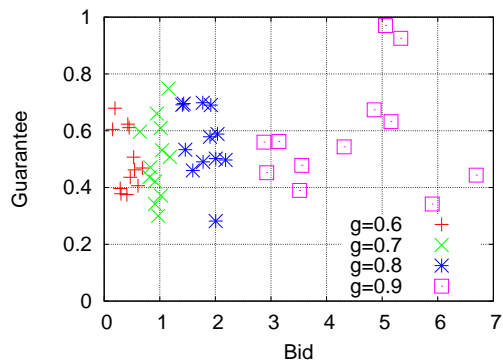


Fig. 4. Bids vs. obtained guarantees for statistical guarantees and 25% Best-Effort jobs

Looking at the price fluctuations in Figure 5, there is a clear trend of inflation in particular for $g = 0.9$ (bottom right). Also note that simply compensating for the bid based on expected inflation would just accelerate this trend. In Figure 6 we see that both the variance and the skewness of the price distribution exhibit similar behavior as in Simulation I.

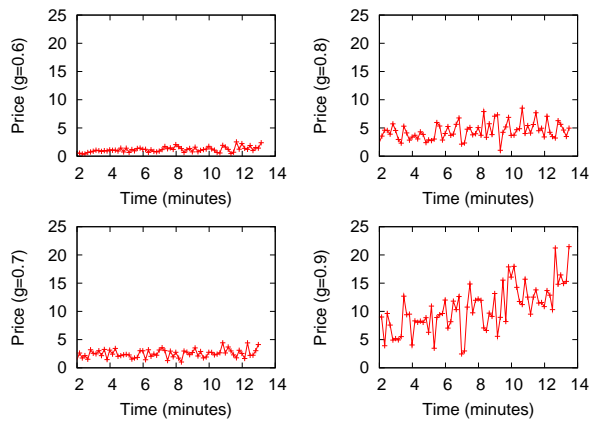


Fig. 5. Price over time for statistical guarantees and 25% Best-Effort jobs

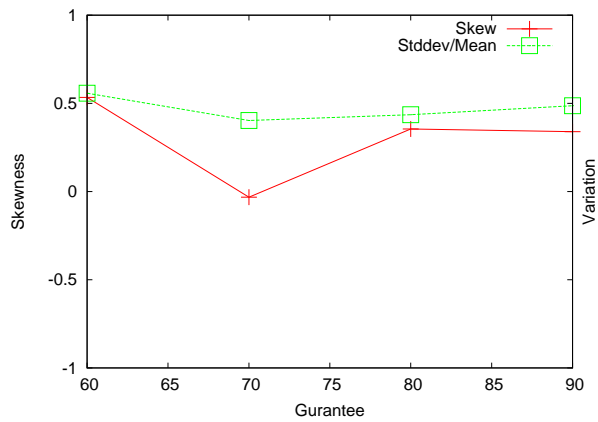


Fig. 6. Price skewness and variation for statistical guarantees and 25% Best-Effort jobs

3.3 Simulation III: 25% Best-Effort with Admission Control Guarantees

Finally we run a simulation with the same load configuration as in the previous simulation, i.e., 25%, best-effort jobs, but now we offer admission control for all non best-effort jobs. An admission control fee of $r = 0.05$ percent points and an enforcement portion of 30% was used. To simulate the important task of an admission control mechanism to allow users to defer their job submissions based on admission results, we defer and resubmit all guarantee jobs that cannot get at least 70% of their work load guaranteed. Time to wait before resubmission is determined randomly with a uniform distribution ranging 1 – 10 seconds. In Figure 7 it is now again apparent that higher bids also give higher guarantees. Although the separation is not as clear as in Simulation I, it is clearly better than in Simulation II. The separation received is related to the proportion of the job that is strictly enforced. In the case of the entire job being strictly enforced all requests levels obtain a 100% guarantee. If the enforcement proportion is made too low, the results will converge to those of Simulation II, that is, requested guarantees cannot be met reliably.

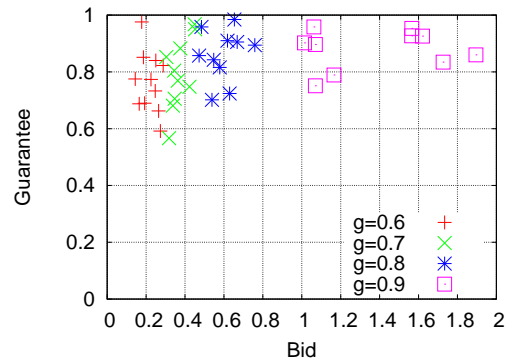


Fig. 7. Bids vs. obtained guarantees for admission control guarantees and 25% Best-Effort jobs

Figure 8 indicates that the inflation is now gone, and Figure 9 shows that the price distribution variation and skewness are similar to the previous two simulations. The penalty for the higher guarantees for some users rests partly on the best-effort jobs and partly on the fact that only a portion 70% of the entire job run is strictly reserved. We should note that the overall load in this simulation is lower and thus the average bid for the jobs that are let through are obviously lower due to some jobs being refused to run by the admission control. The main point here, though, is that we can add admission control as a compromise between reservations and best-effort allocations in scenarios when statistical guarantees fail.

We summarize the results of the simulations in Table 3. We note that although the price distribution variation is the same in all the simulations, Simulation II exhibits a higher variance in guarantee levels delivered, in addition to not being able to deliver on the requested guarantee level.

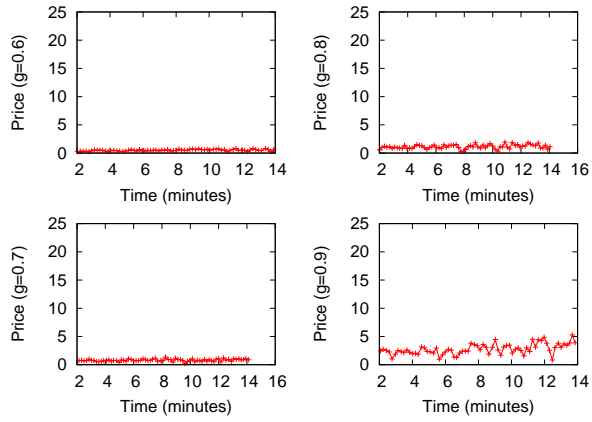


Fig. 8. Price over time for admission control guarantees and 25% Best-Effort jobs

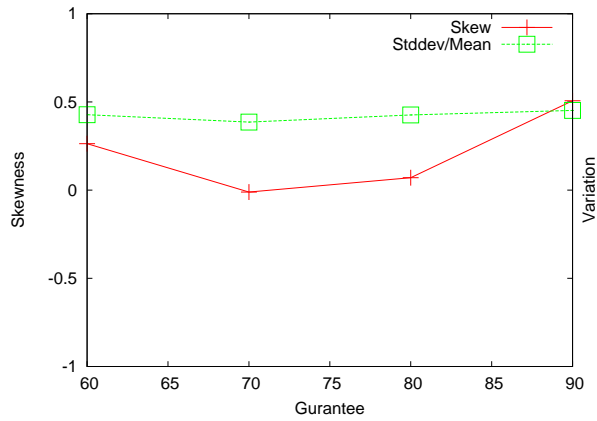


Fig. 9. Price skewness for admission control guarantees and 25% Best-Effort jobs

Table 3. Summary of mean and variation of obtained guarantee levels when requesting 60, 70, 80, and 90% guarantees. All values are in percent.

Simulation	60		70		80		90	
	μ	σ/μ	μ	σ/μ	μ	σ/μ	μ	σ/μ
I	52	15	61	12	83	3	99	1
II	50	22	50	27	56	22	58	34
III	76	14	79	16	86	11	88	8

4 Related Work

There is a substantial body of work on Internet Protocol quality-of-service enforcement, represented by the two IETF specifications IntServ [5], and DiffServ [6]. The IntServ specification takes the approach of reserving paths for individual users, and thus does not scale as well as the DiffServ approach, which is based on marking individual packets with different *per-hop behaviors* in a stateless and decentralized architecture. Wang [7] gives an overview of lessons learned and the pros and cons of the reservation approach which can be implemented with IntServ versus the proportional share approach which can be built on top of DiffServ. The conclusion was that fixed allocations over a point-to-point path incur too much overhead for most of the web traffic, it is difficult to determine the resource requirements a priori, inter-ISP relationships make end-to-end reservations complicated, and traffic policing breaks down in the event of partial allocation failures. All of these factors result in many IP reservation providers over-provisioning their network capacity, leading to poor utilization. Wang therefore makes a case for a proportional share model [8] where each user receives a proportional share of the currently available bandwidth according to her contribution or spending. We are facing the same issues and trade-offs when allocating computational resources across large distributed systems. However, new virtualization technology and the fact that many of the resources are localized (e.g. CPU, memory, disk) makes it worth revisiting the reservation concepts.

One of the most critical parts of the IntServ architecture is the admission control component, and consequently there has been an extensive effort on designing efficient algorithms for deciding which packets are to be dropped versus served, and how routers and switches should be configured to shape the traffic according to the QoS levels promised to users. Knightly and Shroff provide an evaluation of the different admission control algorithms available for IP traffic shaping in [9]. The dilemma of denying access to flows that might have been served leading to underutilization compared to serving requests that will break existing QoS contracts makes it hard to use coarse statistical bounds and too simplified assumptions about traffic flow distributions. Put differently, both accuracy maximization and risk minimization are desired. The algorithms that accounted for economies of scale and not simply looked at the statistical properties of individual flows were shown to perform much better on average. Again, our admission control decision differs from the IP flow one, in that we can, through virtualization, more directly enforce that an admitted request stays within its bounds. Our decision is thus more about making sure that the provider does not lose out on utilization or profit by admitting low priority tasks prematurely.

MacKie-Mason et. al. [10] investigate how price predictors can improve users' bidding strategies in a market-based resource scheduling scenario. Their conclusion is that even very simple predictors, such as taking the average of the previous round of auctions, help improving expected bidder performance. Another interesting result is that the main reason the predictor strategies outperform memory-less strategies is the fact that the binary decision of whether to participate in an auction can save the bidder more money than accurately estimating exactly how much to bid to obtain a certain performance level. Although, the high-level goal of this work is strikingly similar to ours they investigate a very different allocation and auction scenario, where combinatorial prefer-

ences exist and there is a risk of only receiving subsets of the preferred resources. Furthermore, first price winner-takes-it-all auctions are employed, as opposed to proportional share auctions in our work. Nevertheless, their results are encouraging. Another successful use of economic predictions to optimize bidding strategies is described by Wellman et. al. in [11], where bidding agents determine their bids and auctions to enter based on the expected market clearing price in a competitive or Walrasian equilibrium. To find this price they employ the process of *tatonnement* which involves determining users' inclination to bid a certain value given a price-level. Wellman et. al. compare their competitive analysis predictor to simple historical averaging and machine learning models as employed in the Trading Agent Competition (TAC) and conclude that strategies not only considering background history data but also instance-specific data in the predictions provided a competitive advantage. Finally, their competitive predictor performed on-par with the best machine learning predictor. The conditional probability of price dynamics given a certain price-level would be very useful to collect in our case too to get a full picture of the usage pattern. However, in large-scale systems with users entering and leaving the market at will, and large real-valued price ranges it quickly becomes impractical for our purposes, so we assume this behavior is incorporated in the price history itself.

5 Conclusions

We have studied the effects of bidding for virtualized resource shares using price predictions and admission control. For the predictions to be effective there must either be a sufficiently large portion of best-effort bidders, who can decrease their shares when there is contention, or an admission control mechanism refusing access to requests that would break the existing QoS contracts.

Whether a consumer should spend extra money on getting a higher level of guarantee through an admission control contract, thus depends on the contention among consumers requiring high guarantees. Price history and price distribution analysis serve as good indicators for determining whether this is the case. Conversely, providers would be interested in knowing how to partition their resources between the admission control market versus the best effort market depending on the price fluctuation characteristics and usage pattern.

Future work includes reproducing the simulation results in experiments in a live Grid market deployment (presented in [2]), more in-depth analysis of how providers can dynamically partition their resources for contract markets, and adding more sophisticated option and risk-hedging reservations to the admission control mechanism presented here.

6 Acknowledgments

We thank our colleagues Bernardo Huberman, Li Zhang, Fang Wu, Ali Ghodsi and Scott Clearwater for fruitful discussions. This work would not have been possible without the funding from the HP/Intel Joint Innovation Program (JIP), our JIP liason, Rick McGeer, and our collaborators at Intel, Rob Knauerhase and Jeff Sedayao.

References

1. Sandholm, T., Lai, K.: Evaluating Demand Prediction Techniques for Computational Markets. In: GECON '06: Proceedings of the 3rd International Workshop on Grid Economics and Business Models, Singapore (2006)
2. Sandholm, T., Lai, K., Andrade, J., Odeberg, J.: Market-based resource allocation using price prediction in a high performance computing grid for scientific applications. In: HPDC '06: Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing. (2006) 132–143 <http://hpdc.lri.fr/index.php>.
3. Lai, K., Rasmusson, L., Adar, E., Sorkin, S., Zhang, L., Huberman, B.A.: Tycoon: an Implementation of a Distributed Market-Based Resource Allocation System. Technical Report arXiv:cs.DC/0412038, HP Labs, Palo Alto, CA, USA (2004)
4. Mandelbrot, B., Hudson, R.L.: The (Mis)behavior of Markets: A Fractal View of Risk, Ruin, and Reward. Basic Books, New York, NY, USA (2004)
5. Braden, R., Clark, S., Shenker, S.: Integrated services in the internet architecture. RFC 1633, IETF (1994)
6. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An architecture for differentiated services. RFC 2475, IETF (1998)
7. Wang, Z.: A case for proportional fair sharing. In: IWQoS '98: Proceedings of the Sixth International Workshop on Quality of Service, IEEE (1998) 33–35
8. Wang, Z.: Usd: Scalable bandwidth allocation for the internet. In: HPN. (1998) 351–361
9. Knightly, E.W., Shroff, N.: Admission control for statistical qos: Theory and practice. **13**(2) (1999) 20–29
10. MacKie-Mason, J.K., Osepayshvili, A., Reeves, D.M., Wellman, M.P.: Price prediction strategies for market-based scheduling. In: ICAPS. (2004) 244–252
11. Wellman, M.P., Reeves, D.M., Lochner, K.M., Vorobeychik, Y.: Price prediction in a trading agent competition. *J. Artif. Intell. Res. (JAIR)* **21** (2004) 19–36