

Scheduling Co-Reservations with Priorities in Grid Computing Systems

Rui Min and Muthucumaru Maheswaran

Advanced Networking Research Laboratory
Department of Computer Science
University of Manitoba
Winnipeg, MB R3T 2N2, Canada
Email: {ruimin@ee, maheswar@cs}.umanitoba.ca

This paper presents three algorithms for co-reserving resources. A novel mechanism for specifying QoS constraints with the requests is introduced. The performance of the algorithms are studied through extensive simulations. The results indicate that depending on the objective of optimization different algorithms should be used. One of the distinguishing features of our algorithms is that they operate in a pseudo-online mode (called the “batch” mode).

1. Introduction

Advance resource reservations and co-reservations in the context of Grid computing have been the focus of few studies [1, 2]. Algorithms for supporting advance reservations in high-performance computing environments are presented in [2]. These algorithms treat reservation requests and requests for resources from “batch” jobs in a unified manner and they allow users to request multiple resources simultaneously (i.e., perform co-reservations). However, [2] does not allow different users to “space share” the resources during the same time intervals. Further, the batch applications are assumed to have lower priority than the reservation requests. Another system for performing co-reservation in Grid systems is the *Globus Architecture for Reservation and Allocation* (GARA) [1].

This paper proposes algorithms for scheduling co-reservations on heterogeneous resources. Although, only CPU resources are considered here, the approach may be generalized to other resources such as network and storage. Further, in this paper, immediate reservations are modeled as advance reservations with current time as the start time and a predefined length of time for the duration. This allows us to unify advance and immediate reservations.

The *Co-reservation Scheduler with Priorities and Benefit functions* (Co-RSPB) presented here considers the relative priorities of the different reservation requests. In Co-RSPB, each request has an associated benefit function that quantifies the “profit” accrued by the client by securing the resource at the requested level. When a client is willing to negotiate for lower service levels, it could indicate this by

providing a benefit function that shows a reduced but positive benefit.

2. Co-Reservation Algorithms and Evaluation

The algorithms are based on the following assumptions. Once a request is granted reservation, a contract for the reservation is signed between the application and the system. The co-reservation scheduler won’t examine the same request more than once except when a QoS violation occurs. In this study, each co-reservation request involves multiple resources. If any one required resource is not available to the application, the overall request will be rejected by the reservation scheduler.

All three algorithms assemble a *meta-request* by accumulating several reservation requests. The meta-request is then scheduled by the chosen heuristic. A request is referred to as *floating* if any resource set can satisfy the request and called *fixed*, otherwise. The Co-RSPB, schedules the floating requests as follows. The requests in the meta-request are sorted by the priority and the requests are considered in descending order by the priority. Because the requests are co-reservation requests, they can have sub-requests. The sub-requests are sorted by the minimum CPU requirement and are considered for allocation in descending order by minimum CPU requirement. In the *Co-reservation scheduler with Best Fit Scheme* (Co-RSBF), the requests are sorted by the sum of the sub-requests’ CPU requirements. The rest of the Co-RSBF algorithm is same as the Co-RSPB algorithm. The *Co-reservation scheduler with Best Fit and Refining* (Co-RSBFR) algorithm uses the Co-RSBF algorithm and admits the reservation requests and increases the benefit delivered to the requests via the refinement process.

Simulations were performed to evaluate the performance of the different co-reservation algorithms. Figure 1 shows the variation of system benefit and the number of rejections with the number of requests. Figure 2 shows the variation of system benefit with the number of machines. Figure 3 shows the variation of the number of rejections with the average of duration.

The lower system benefit provided by Co-RSBF can be explained by noting that Co-RSBF selects the reservation that provides the applications the lowest possible benefit even when there is no resource scarcity. Although Co-RSBF

This research was partially supported by a Natural Sciences and Engineering Research Council of Canada Grant RGP 220278 and equipment used was provided by a Canada Foundation for Innovation grant.

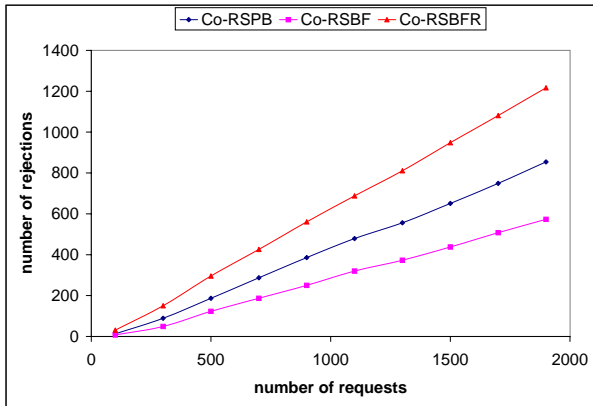


Figure 1. Variation of the number of rejections with number of requests.

consistently admits more requests than Co-RSPB, it performs worse based on the system benefit measure. This is because some of the requests may be more important than the others. By not considering the priority, the Co-RSBF is unable to increase the system benefit measure.

Further, it can be noted that Co-RSBFR performs poorly than the other two approaches on both metrics. Because the requests arrive in a dynamic fashion, the refinement performed by the Co-RSBFR “locally” optimizes the reservation scheduling and consumes resources that are sought by the requests that arrive in the future. Therefore, the refinement process increases the number of rejections as well as decreases the system benefit.

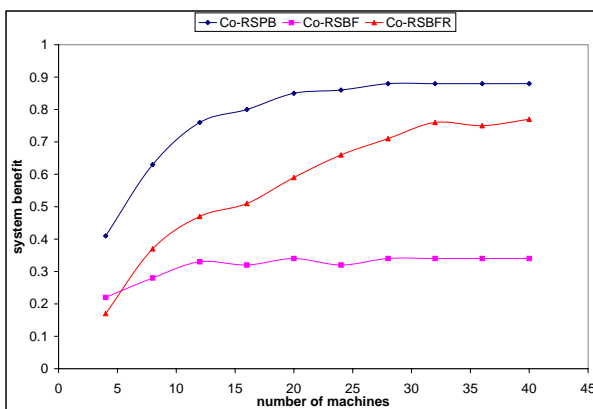


Figure 2. Variation of the system benefit with number of machines.

From Figures 1 and 3, we can note that the number of rejections for Co-RSBFR is much higher than that for Co-RSPB and Co-RSBF. It is interesting to note that as the number of requests increase in Figure 1, the performance of Co-RSBFR with respect to number of rejections becomes significantly worse. This is because the Co-RSBFR attempts to refine the reservation by providing more benefit for requests that are already reserved. By this process, depending on the resource demand, a significant portion of the resources may be taken by the requests that arrive first. Thus, leaving little resources to requests that arrive latter.

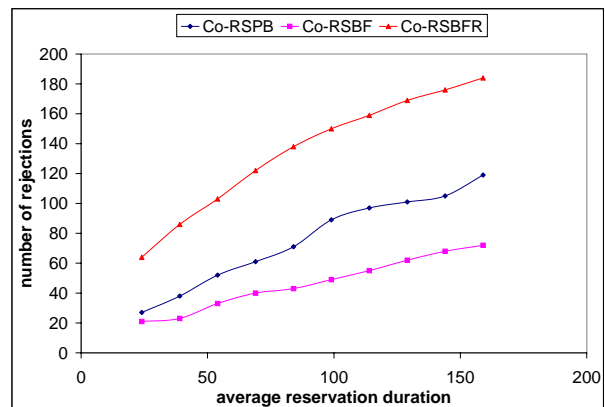


Figure 3. Variation of the number of rejections with average request duration.

3. Conclusions and Future Work

From the simulation studies, it can be observed that depending on the policy of “optimization” different algorithms should be used. For example, if the objective is to minimize the number of reservation requests rejected, then we must use Co-RSBF. On the other hand, if we intend to maximize a global measure such as the system benefit, then we must use Co-RSPB.

References

- [1] I. Foster, C. Kesselman, C. Lee, B. Lindell, K. Nahrstedt, and A. Roy, “A distributed resource management architecture that supports advance reservation and co-allocation,” *Seventh IEEE International Workshop on Quality of Service (IWQoS 99)*, May 1999.
- [2] W. Smith, I. Foster, and V. Taylor, “Scheduling with advanced reservations,” *International Parallel and Distributed Processing Symposium (IPDPS '00)*, May 2000.