A Stochastic Model for Robust Resource Allocation in Heterogeneous Parallel and Distributed Computing Systems

Jay Smith[1,2], Howard Jay Siegel[2,3] and Anthony A. Maciejewski[2]

[1]DigitalGlobe
Longmont, CO 80503 USA
Email: jtsmith@digitalglobe.com

Colorado State University
[2]Dept. of Electrical and Computer Engineering
[3]Dept. of Computer Science
Fort Collins, CO 80503-1373 USA
Email: {hj, aam}@engr.colostate.edu

Abstract

This paper summarizes some of our research in the area of robust static resource allocation for distributed computing systems operating under imposed Quality of Service (QoS) constraints. Often, these systems are expected to function in a physical environment replete with uncertainty, which causes the amount of processing required over time to fluctuate substantially. Determining a resource allocation that accounts for this uncertainty in a way that can provide a probabilistic guarantee that a given level of QoS is achieved is an important research problem. The stochastic robustness metric described in this research is based on a mathematical model where the relationship between uncertainty in system parameters and its impact on system performance are described stochastically.

1 Introduction

In a heterogeneous computing system, task execution times may differ depending on which computer executes a given task. Often, task resource requirements lead to inconsistent performance differences between heterogeneous machines. That is, machine 1 being faster than machine 2 on some task A does not imply that machine 1 is uniformly faster on all tasks. Resource allocation in such computing environments has been shown in general to be an NP-hard problem (e.g., [7]). Thus, the design of heuristics for resource allocation is an active area of research, e.g., [1, 4, 5, 6, 9, 11].

This research was supported by the NSF under Grant CNS-0615170 and by the Colorado State University George T. Abell Endowment.

978-1-4244-1694-3/08/$25.00 ©2008 IEEE

Resource allocation decisions are often based on estimated values of task and system parameters, whose actual values are uncertain and may differ from available estimates. A resource allocation can be considered "robust" if it can mitigate the impact of uncertainties in system parameters on a given performance objective [2]. That is, a *robust* resource allocation can guarantee a certain level of *performance* under a wide range of conditions. Any claim of robustness for a given system must answer these three questions [3]: (a) What behavior makes the system robust? (b) What are the uncertainties that the system is robust against? (c) Quantitatively, exactly how robust is the system? These three questions help establish an intuitive meaning for the robustness of a system that goes beyond a simple nebulous adjective.

In some environments, there may be information available regarding the probability of variations in system parameters. For example, we may have historical information regarding past execution times for a given task that can be used to approximate the probabilities of all possible execution times. This stochastic information can be utilized to derive a robustness metric. In this paper, we define a stochastic methodology for quantifying the ability of a resource allocation to satisfy quality of service constraints in the midst of uncertainty in system parameters. System parameters are modeled as random variables and we assume that stochastic information is available that characterizes the uncertainty in system parameters beyond a simple point estimate.

To illustrate the usefulness of stochastic data in resource allocation, consider an allocation environment with two machines (A and B). In the example situation of Figure 1, some tasks have previously been assigned to machines A and B. We would like to select a machine to execute task 3 and we would like task 3 to complete by the plotted completion time constraint. If we use a

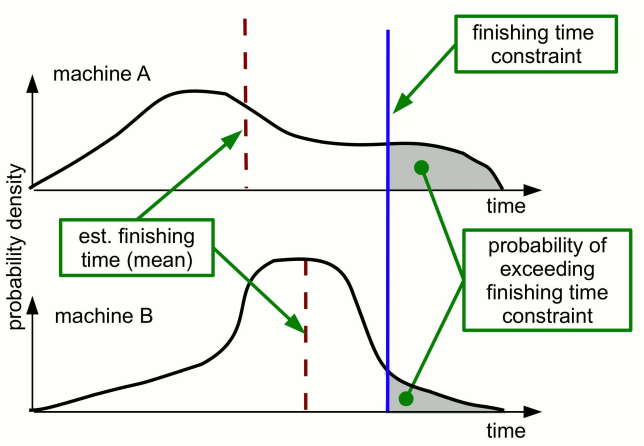

Figure 1. An example system where we are to assign task 3 to either machine A or machine B and we would like task 3 to complete prior to the plotted completion time constraint. Presented are the task completion time probability density functions for task 3 on both machine A and machine B. Using the point estimate, plotted as a dashed line in the figure, machine A appears to be the better choice. However, accounting for the complete stochastic information describing all possible completion times, machine B has a lower probability to violate the completion time constraint and is clearly the better choice.

point estimate for the completion time of task 3, e.g., the mean of the completion time distributions, then it would appear that the best decision would be to assign task 3 to machine A whose point estimate of the completion time is smaller than the point estimate on machine B. However, by using the full stochastic information we can see that although the point estimate of the completion time distribution for task 3 on machine A is smaller than on machine B, the tail of the distribution is much smaller on B than on A. That is, there is a much higher probability that task 3 will violate its completion time constraint on machine A, making machine B the statistically better choice.

This paper provides a summary of our previously published results on stochastic robustness [12]. In the next section, we demonstrate the derivation of a robustness metric for an example static environment. Section 3 illustrates the use of the stochastic robustness metric through several example applications. We present open problems in robust resource allocation in Section 4.

2 Deriving a Robustness Metric

The three robustness questions provide the basis for the more formal FePIA procedure for deriving a quantitative measure of robustness [2, 9]. The procedure uses the following four steps for measuring the impact of uncertainty in estimated system parameters on a stated performance objective: (1) identify the performance features of interest within the system, (2) identify the source of uncertainty within the system (perturbation parameters), (3) clarify the impact of the system uncertainty on the performance features of interest, and (4) analyze the system to quantify robustness.

We illustrate the intuition behind the derivation of a robustness metric in a static stochastic allocation environment, where we are concerned with allocating a set of T tasks to M heterogeneous machines. In a static environment, the entire collection of tasks to be allocated is known in advance, prior to the start of allocation. For this system, the performance feature of interest is system makespan (the time required to compute all T tasks), denoted ψ. A resource allocation can be considered robust if the actual finishing time of each machine is less than or equal to a fixed constant $\beta_{\max}$, i.e., $\psi \leq \beta_{\max}$.

Uncertainty in this environment arises because the exact execution time for each task is unknown. We can model the execution time of each task i on each machine j as a random variable [13], denoted η_{ij}.

The finishing time of each machine in a stochastic environment is calculated as the sum of the execution time random variables for each task assigned to that machine [12]. Let n_j be the number of tasks assigned to machine j. The finishing time of machine j, referred to as a local performance characteristic ψ_j, can be expressed as follows:

$$\psi_j = \sum_{i=1}^{n_j} \eta_{ij}. \tag{1}$$

Thus, the system makespan can be expressed in terms of the local performance characteristics as follows:

$$\psi = \max\{\psi_1, \cdots, \psi_M\}. \tag{2}$$

Because of its functional dependence on the execution time random variables, the system makespan is itself a random variable. That is, the uncertainty in task execution times can have a direct impact on the performance metric of this system.

Finally, we conduct an analysis to determine exactly how robust the system is under a specific resource allocation. The stochastic robustness metric, denoted θ, is defined as the probability that the performance characteristic of the system is less than or equal to $\beta_{\max}$, i.e., $\theta = \mathbb{P}[\psi \leq \beta_{\max}]$. For a given resource allocation, the stochastic robustness metric measures the probability that the generated system performance will satisfy our robustness requirement. Clearly, unity is the most desirable stochastic robustness metric value, i.e., there is a zero probability that the system will violate the established robustness requirement.

Assuming no inter-task data transfers exist among the tasks to be assigned, the random variables for the local performance characteristics $(\psi_1, \psi_2, \cdots, \psi_M)$ are mutually independent. As such, the stochastic robustness metric for a resource allocation can be found as the product of the probability that each local performance feature is less than or equal to $\beta_{\max}$. Mathematically, this is given as,

$$\theta = \prod_{\forall j} \Big(\mathbb{P}\big[\psi_j \leq \beta^{max}\big] \Big). \tag{3}$$

If the execution times η_{ij} for tasks assigned to machine j are mutually independent, then the summation of Equation 1 can be computed using an $(n_j - 1)$-fold convolution of the corresponding probability mass functions (pmf) [10, 12].

3 Using Robustness

Two ways of using the static stochastic robustness metric are apparent by inspecting the parameters of Equation 3. The two parameters β^{max} and θ can alternately be either optimized or specified by the user. For example, the user could specify a β^{max} value as a constraint and employ a heuristic to attempt to maximize the robustness (θ) of the resulting resource allocation. That is, in this case, the user is interested in a resource allocation that has the highest probability to complete all of the tasks by β^{max}. Maximizing the robustness of a resource allocation given a fixed β^{max} requires minimizing the ψ_j values, thus, maximizing the probability that each machine will finish before β^{max}.

For some systems, it may be unclear how to select an appropriate β^{max} value. Thus, we can instead define a minimum acceptable θ value, denoted ω, and attempt to minimize β^{max} such that the probability that all tasks complete by β^{max} is at least ω. In this section, we consider the case where the minimum acceptable robustness value ω is specified by the user and we want to minimize β^{max} such that $\theta \geq \omega$. In [12], the period minimization routine (PMR) was introduced to iterate through the possible β^{max} values for a *given resource allocation* to find the smallest β^{max} value, provided by that allocation, such that $\theta \geq \omega$.

To demonstrate the use of the stochastic robustness metric in a resource allocation, we present a static stochastic environment where a set of machines periodically receive data sets to be processed by a collection of n tasks [12]. Each data set must be processed by all of the tasks before the next data set arrives. The execution times for each of the n tasks is assumed to be inherently data dependent, thus, the exact execution time for each task is unknown prior to its execution. However, we are provided a pmf describing the probabilities of task execution times for each machine.

It is important to note that because this is a static environment, we are determining the allocation of tasks to machines *in advance* of deploying the system, i.e., before it will be required to process real data. Thus, by optimizing the allocation of machines to tasks in advance, we can reduce β^{max} and improve the frequency with which the system can process data sets.

The stochastic robustness metric for this system is the probability that the actual makespan of the system does not exceed β^{max}, i.e., $\theta = \mathbb{P}\left[\psi \leq \beta^{max}\right]$. The goal of resource allocation heuristics in this environment is to minimize β^{max} such that θ is always greater than or equal to a given fixed probability ω, i.e., $\theta \geq \omega$. Intuitively, by specifying a minimum robustness value (ω), the user is specifying an acceptable probability for the makespan to be greater than β^{max}, i.e., $1 - \omega$.

We can use this formulation of robustness along with the PMR procedure to design a two-phase greedy heuristic for resource allocation in this system [12]. The two-phase greedy heuristic first initializes the set of tasks to be executed to the entire set of tasks that are available. While there are still tasks to execute, the heuristic uses two phases to find the next task assignment. In the first phase, the heuristic determines a machine assignment for each unassigned task that minimizes β^{max} (ignoring all other unassigned tasks), where we use the PMR procedure to determine the smallest β^{max} for each possible machine assignment for each task such that the robustness constraint (ω) is still satisfied. In the second phase, the heuristic selects the task machine pair (found in the first phase) that provides the smallest overall β^{max}. The selected task is then allocated to its chosen machine and removed from the set of tasks to be assigned. The heuristic continues in this way until all tasks have been allocated.

This section presented just two example uses of the

static stochastic robustness metric, many more uses are possible. The next section considers open problems in robust resource allocation.

4 Future Work

There are a number of immediate problems to be addressed in stochastically robust resource allocation. In a stochastic environment, methods are needed for leveraging experiential data to model uncertainty in perturbation parameters [8]. This is important because prior work in stochastic resource allocation environments has assumed that these models are available, i.e., in the form of probability mass functions. Further, once a pmf has been established for a perturbation parameter, methods are needed for updating the existing pmf with new experiential data. Updating pmfs with the most current information is important in a dynamic environment because it enables the model to track changes in perturbation parameter distributions over time. In addition, pmfs based on experiential data may provide only an approximation of the true distribution of perturbation parameter values and methods are needed for determining the impact of estimation error in pmfs on robustness calculations. That is, how to make resource allocation decisions robust with respect to estimation errors in perturbation parameter pmfs is an open problem.

Many heterogeneous computing environments have inherent quality of service constraints that must be met during resource allocation. Determining such resource allocations is an active area of research. We have briefly shown an example environment with quality of service constraints and demonstrated how to apply the robustness methodology to determine resource allocations that meet those constraints. In addition to this example, other quality of service requirements might involve multiple robustness requirements, e.g., minimum bandwidth requirements, guaranteed processor time for certain users, or real-time response capabilities.

Our current research in robust resource allocation, is also investigating combining multiple types of perturbation parameters into a single robustness metric, e.g., combining machine failure probabilities and task execution time uncertainty into a single metric. Combining multiple perturbation parameters into a single measure of robustness will extend the applicability of robust resource allocation into problem domains where these uncertainties occur simultaneously.

In a stochastic environment, resource allocation decisions may depend on combining perturbation parameter pmfs. For example, pmfs for perturbation parameters can be used to produce an overall metric for the robustness of a resource allocation. In a dynamic environment, where resource allocation decisions must be made quickly, it is important to identify new fast methods for combining perturbation parameter distributions to produce a robustness value during resource allocation. If heuristics can quickly combine perturbation parameter distributions, then we can determine methods for using the stochastic robustness metric to guide resource allocation decisions during execution.

5 Summary

Robust resource allocation is an important research area within heterogeneous parallel and distributed computing systems. The three robustness questions are fundamental to the understanding of robustness in any system: (1) What behavior makes the system robust? (2) What uncertainties must the system be robust against? (3) Quantitatively, exactly how robust is the system? These three core questions led to the development of the FePIA procedure for deriving a robustness metric. In this paper, we presented our application of the FePIA procedure to a static stochastic environment to a derive robustness metrics. In addition to demonstrating the derivation of a robustness metric, we have also demonstrated two ways to incorporate a robustness metric into resource allocation heuristics.

References

[1] S. Ali, T. D. Braun, H. J. Siegel, A. A. Maciejewski, N. Beck, L. Bölöni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, and B. Yao. *Characterizing Resource Allocation Heuristics for Heterogeneous Computing Systems*, volume 63 of *Advances in Computers*, pages 91–128. Elsevier, Amsterdam, The Netherlands, 2005.

[2] S. Ali, A. A. Maciejewski, H. J. Siegel, and J.-K. Kim. Measuring the robustness of a resource allocation. *IEEE Transactions on Parallel and Distributed Systems*, 15(7):630–641, July 2004.

[3] S. Ali, H. J. Siegel, and A. A. Maciejewski. The robustness of a resource allocation in parallel and distributed computing systems. In *Proceedings of the joint meeting of ISPDC 2004: Third International Symposium on Parallel and Distributed Computing, and HeteroPar '04: Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks*, pages 2–10, July 2004.

[4] L. Bölöni and D. Marinescu. Robust scheduling of metaprograms. *Journal of Scheduling*, 5(5):395–412, Sept. 2002.

[5] T. D. Braun, H. J. Siegel, N. Beck, L. Bölöni, R. F. Freund, D. Hensgen, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, and B. Yao. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing

systems. *Journal of Parallel and Distributed Computing*, 61(6):810–837, June 2001.

[6] A. Dogan and F. Özgüner. Genetic algorithm based scheduling of meta-tasks with stochastic execution times in heterogeneous computing systems. *Cluster Computing*, 7(2):177–190, Apr. 2004.

[7] O. H. Ibarra and C. E. Kim. Heuristic algorithms for scheduling independent tasks on non-identical processors. *Journal of the ACM*, 24(2):280–289, Apr. 1977.

[8] M. A. Iverson, F. Özgüner, and L. Potter. Statistical prediction of task execution times through analytical benchmarking for scheduling in a heterogeneous environment. *IEEE Transactions on Computers*, 48(12):1374–1379, Dec. 1999.

[9] D. L. Janovy, J. Smith, H. J. Siegel, and A. A. Maciejewski. Models and heuristics for robust resource allocation in parallel and distributed computing systems. In *Proceedings of the* 21^{st} *International Parallel and Distributed Processing Symposium (IPDPS 2007)*, Mar. 2007.

[10] A. Leon-Garcia. *Probability & Random Processes for Electrical Engineering*. Addison Wesley, Reading, MA, 1989.

[11] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R. F. Freund. Dynamic mapping of a class of independent tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, 59(2):107–121, Nov. 1999.

[12] V. Shestak, J. Smith, A. A. Maciejewski, and H. J. Siegel. Stochastic robustness metric and its use for static resource allocations. *Journal of Parallel and Distributed Computing*, accepted to appear, 2008.

[13] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Science+Business Media, New York, NY, 2005.