System Sensitive Runtime Management of Adaptive Applications

Shweta Sinha **Manish Parashar**
TASSL, Rutgers University
Piscataway, NJ 08855-8060
{*parashar, shwetas*}*@caip.rutgers.edu*

Abstract

In this project we will design and evaluate an adaptive, system sensitive distribution/load balancing framework for distributed adaptive grid hierarchies that underlie parallel adaptive mesh-refinement (AMR) techniques for the solution of partial-differential equations. System sensitive adaptation investigates the use of current system state to drive runtime adaptation.

The application will be run in a dynamic and heterogeneous networked computing environment. These environments require the selection and configuration of application components based on available resources. However, the complexity and heterogeneity of the environment make selection of a "best" match between system resources, mappings and load distributions, communication mechanisms, etc., non-trivial. System dynamics coupled with application adaptivity makes application and run-time management a significant challenge. What is unique to our approach is the fact that we will take into account the dynamism of the heterogeneous network as well as the application in an attempt to enhance system performance. However, this adds an extra dimension of complexity to our problem and is non-trivial.

Our System model is shown in Figure 1. We first monitor the necessary resources associated with the different processors and then calculate their relative capacities. The relative capacities are then used by the Heterogeneous Partitioner in distributing the work load proportionately among the different processors. We are currently evaluating the performance improvement that can be achieved by using this "system sensitive" approach to load distribution. We discuss below each of the system components in Figure 1 in a little more detail.

To determine the system characteristics, we use the NWS (Network Weather Service) resource monitoring tool developed at UCSD [2]. The NWS is a distributed system that periodically monitors and dynamically forecasts the performance delivered by the various network and computational resources over a given time interval. The service operates a distributed set of performance sensors (network monitors, CPU monitors, etc.) from which it gathers readings of the prevailing system conditions. It then uses numerical models to generate forecasts of what the conditions will be for a given time frame. This functionality is analogous to weather forecasting, and as such, the system inherits its name. The current implementation of NWS supports measuring the fraction of

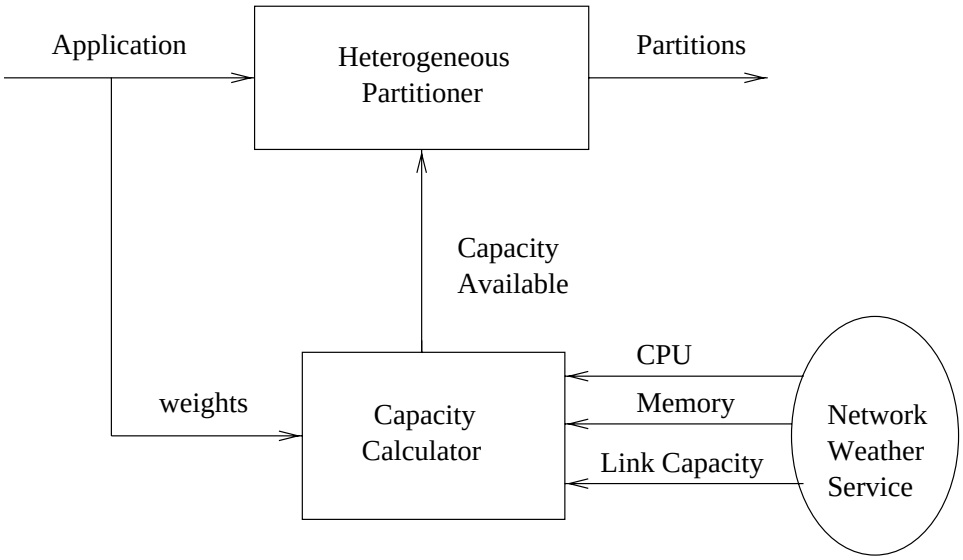

Figure 1: *Block Diagram of the System Model*

CPU time available for new processes, the fraction of CPU available to a process that is already running, end-to-end TCP network latency, end-to-end TCP network bandwidth, free memory, and the amount of space unused on a disk. The NWS forecaster applies a set of forecasting models to the entire series of measurements and dynamically chooses the forecasting technique that has been most accurate over the recent set of measurements. When a forecast of a future value is required, the forecaster makes predictions for each of the existing measurements in the series. Every forecasting model generates a prediction for each measurement, and a cumulative error measure is tabulated for each model. The model generating the lowest prediction error for the known measurements is then used to make a forecast of future measurement values.

As shown in Figure 1, after gathering resource information from NWS, we compute a capacity metric for each processor based on its system characteristics such as free memory, CPU availability and link bandwidth. We propose a linear model for the calculation of the relative capacity of each processor. Let us assume that there are K processors in the system among which the partitioner distributes the work load. For the kth processor, let $\mathcal{P}_k$ be its CPU availability, $\mathcal{M}_k$ its available memory, and $\mathcal{L}_k$ its link bandwidth. We can also express these resources of the kth user as a fraction of the total available resources in the system as $P_k = \mathcal{P}_k / \sum_{i=1}^{K} \mathcal{P}_i$, $M_k = \mathcal{M}_k / \sum_{i=1}^{K} \mathcal{M}_i$, $L_k = \mathcal{L}_k / \sum_{i=1}^{K} \mathcal{L}_i$. The relative capacity of a processor C_k is then defined as the weighted sum of these normalized quantities

$$C_k = w_p P_k + w_m M_k + w_l L_k \tag{1}$$

where w_p, w_m, and w_l are the weights associated with the relative CPU, memory, and link bandwidth availabilities, respectively. Also, $w_p + w_m + w_l = 1$. We need to weigh each of the system characteristics because different applications may have different resource requirements. These weights are application specific and may be varied based on the requirements of the application. For example, if an application is memory intensive, then w_m will be greater than w_p and w_l so that the relative capacity reflects the importance of the available memory appropriately. Currently, we assume that all three system characteristics are equally important to the application and hence we

choose $w_p = w_m = w_l = 1/3$. Since C_k is the relative capacity of processor k, $\sum_{k=1}^{K} C_k = 1$. This model is very general and can be easily extended to accommodate other system resources as well.

Once the relative capacities of the processors are computed, the work load is distributed proportionately among them. The Heterogeneous Partitioner uses the GrACE infrastructure [1], to distribute AMR grid hierarchies. The system state is monitored at run time, and if system characteristics change, the work load is redistributed amongst the processors. To evaluate the performance of this system sensitive approach, the overall execution time of the application is measured and compared against other partitioning schemes.

References

[1] Manish Parashar and James C. Browne, "*On Partitioning Dynamic Adaptive Grid Hierarchies*," Proceedings of the 29th Annual Hawaii International Conference on System Sciences, January, 1996.

[2] Rich Wolski, Neil T. Spring and Jim Hayes "*The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing*," Future Generation Computing Systems, 1999, http://www.www.cs.ucsd.edu/groups/hpcl/apples/hetpubs.html

[3] Silvia M. Figueira and Francine Berman "*Mapping Parallel Applications to Distributed Heterogeneous Systems*, " UCSD CS Tech Report # CS96-484, June 1996

[4] Jerrel Watts, Marc Rieffel and Stephen Taylor "*Dynamic Management of Heterogeneous Resources*," High Performance Computing '98.

[5] Muthucumaru Maheswaran and Howard Jay Seigel " *A Dynamic Matching and Scheduling Algorithm for Heterogeneous Computing Systems*," 7th IEEE Heterogeneous Computing Workshop (HCW '98), Mar. 1998, pp. 57-69.