Prediction Based Task Scheduling in Distributed Computing*

Mehrdad Samadani and Erich Kaltofen

Department of Computer Science
Rensselaer Polytechnic Insitute
Troy, New York 12180-3590 USA

1 Introduction

The effectiveness of task scheduling in a distributed environment is critically dependent on the timely identification of the least loaded nodes. Whether the issue of interest is load-sharing or distributed parallel computation, overall system performance is determined in large part by the characteristics of the nodes participating in a particular computation. The diversity of node characteristics across the network frequently results in a spectrum of available compute powers on different nodes. Due to the high cost of task migration, effective evaluation of the relative available compute powers of the nodes in the network and the use of that information in task distribution are essential components of successful task scheduling in a distributed environment.

A few implementations of systems for distributing large scale computations over a network of computers rely on schemes based on the current as well as the prior state information on each node to make better task distribution decisions [2]. The goal of such algorithms is to uncover the hidden information that may be present in the past load data, and use that information to better evaluate the available compute power of the nodes in the network. These approaches tend to rely on extensive fine-tuning often due to the ad-hoc design of the data analysis procedures employed.

We present a systematic and statistically sound method for uncovering the information present in the past load data and using that information to predict the future processor load. Using time series statistical methods we analyze the compute intensive load on actual and simulated processors to identify stochastic models that suitably represent the load behavior on those processors. We then

*This material is based on work supported in part by the National Science Foundation under Grant No. CCR-93-19776. Appears in Languages, Compilers and Run-Time Systems for Scalable Computers, B. K. Szymanski and B. Sinharoy editors, pp. 317–320, Kluwer Academic Publ., Boston, 1996.

use these models to obtain the minimum mean square error load forecasts for various times in the future and discuss how they can be used in task scheduling in distributed systems. In particular, we use these forecasts to reduce the performance degradation caused by the outdated state information resulting from delays in collecting it.

The performance of the minimum mean square error load forecasts are then compared against those obtained by using the current load levels as predictors of the future state of the nodes. We show that the relative performance of these approaches depends on factors including the statistical characteristics of the load on each processor and the number of nodes in the system.

2 Time Series Analysis

Conceptually, the load on a processor can be modeled by a theoretical stochastic process. In the following, we express the computation load on a processor by the length of its CPU ready queue. Let x_t denote the load on the processor X at time t. We refer to the sequence $\{x_{t-j}\}$, where $j = 1, 2, \ldots$ as the load time series on node X. The special feature of this type of load time series is the fact that successive observations are usually not independent and that any analysis of such time series must take into account the time order of the observations. Since successive observations are dependent, future values may be predicted from past observations. The load time series are stochastic in that the future is only partly determined by past values, so that exact predictions are impossible and must be replaced by the idea that future values have a probability distribution which is conditioned by a knowledge of past values. The information present in the load time series can be extracted by appropriate statistical analysis methods.

2.1 Preliminaries

Suppose $\{a_t\}$ is a sequence of independent and identically distributed random variables from a Normal distribution with mean zero and variance σ_a^2. A powerful model for describing time series is the general autoregressive integrated moving average (ARIMA) process of order (p, d, q), defined by

$$\Phi(B) w_t = \Theta(B) a_t$$

where

$$w_t = \nabla^d x_t = (1-B)^d x_t,$$

$$\Phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p,$$

$$\Theta B = 1 - \theta_1 B - \theta_2 B^2 - \ldots - \theta_q B^q,$$

and where B is defined by $B z_t = z_{t-1}$ for any time series $\{z_t\}$. The operators B and ∇ are called the backward shift and the backward difference operator respectively. The coefficients ϕ_i and θ_i are constants.

Given a model ARIMA(p,d,q) and a time series $\{x_{t-j}\}$, where $j = 1,2,\ldots$ generated by it we would like to forecast the value of x_{t+l}, where $l \geq 1$, when we are currently standing at time t. Equivalently, we would like to obtain $\hat{x}_t(l)$, the minimum mean square error forecast at origin t for lead time l. It can be shown ([1]) that

$$\hat{x}_t(l) = E[x_{t+l} \mid x_t, x_{t-1}, x_{t-2}, \ldots]$$

where the right hand side denotes the conditional expectation of x_{t+l} given $\{x_{t-j}\}$, $j = 1,2,\ldots$. When $\hat{x}_t(l)$ is regarded as a function of l for fixed t, it is referred to as the forecast function for origin t.

Our approach to forecasting is first to derive a suitable stochastic model based on the available data for the particular load time series under study. Once an appropriate model for the series has been determined, it will be used in obtaining the forecast function.

2.2 Application to Task Scheduling

A daemon task is placed on each node in the system. The daemon is responsible for periodic collection of load data on its node. Once fifty or more load observations have been recorded, the daemon uses the data to test the validity of the current stochastic model by applying diagnostic checks to the model. One possible check could be carried out by computing the residuals $a_t = x_t - \hat{x}_{t-1}(1)$ from the data and testing whether or not they appear to be random. Alternatively, the daemon could simply identify and estimate a new stochastic model periodically as discussed in the above. In either case, the new model is communicated to the central scheduler in the next status update message. The daemon sends a status update message to the central scheduler at regular intervals. These messages contain the current stochastic model as well as the most recent data required by the model for forecasting purposes.

The central scheduler maintains the stochastic model and forecasts for each node in the system. The scheduler periodically receives status update messages from the local daemons. It then uses the new information to update the stochastic model it uses for prediction purposes and computes new forecasts for the affected node. When a task arrives at the central scheduler, the forecasts are used to identify the node with the highest expected available compute power and the task is assigned to the identified node.

3 Numerical Study

As part of our numerical study, load data were obtained and analyzed for actual as well as simulated systems. The results were used to compare the performance of the minimum mean square error load forecasts against those obtained by using the current load levels as predictors of the future state of the nodes. We refer to the latter approach as the single point prediction method.

Virtually every load time series obtained from the experiments turned out to be non-stationary. The model fitted to one of the actual load time series is the ARIMA$(2, 1, 0)$

$$(1 + 0.30B + 0.32B^2)\nabla x_t = a_t$$

or

$$x_t = 0.70x_{t-1} - 0.02x_{t-2} + 0.32x_{t-3} + a_t.$$

In the case of the simulated nodes, the models fitted to a few of the load time series were very close to Markov process, or ARIMA$(0, 1, 0)$

$$\nabla x_t = a_t.$$

However, in other simulated cases, we obtained models such as the ARIMA$(1, 1, 1)$

$$(1 - 0.22B)\nabla x_t = (1 - 0.79B)$$

or

$$x_t = 1.22x_{t-1} - 0.22x_{t-2} + a_t - 0.79a_{t-1}.$$

The performance comparison study that was carried out using this ARIMA(1, 1, 1) model resulted in increasing number of instances where the two forecasting methods led to different scheduling decisions as the number of nodes was increased. When they disagreed, the minimum mean square error forecast was about twice as likely to make a better decision than the single point predictor.

References

[1] G. E. P. Box, G. M. Jenkins, "Time Series Analysis: Forecasting and Control", Holden-Day, 1976.

[2] A. Diaz, M. Hitz, E. Kaltofen, A. Lobo, T. Valente, "Process Scheduling in DSC and the Large Sparse Linear Systems Challenge", *Proc. DISCO '93, Springer Lect. Notes Comput. Sci.*, A. Miola (ed.), Vol. 722, pp. 66–80, 1993. *J. Symbolic Computation*, to appear.

[3] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, V. Sunderam, "PVM Parallel Virtual Machine A Users' Guide and Tutorial for Network Parallel Computing", MIT Press, Cambridge, MA, 1994.

[4] K. K. Goswami, M. Devarakonda, R. K. Iyer, "Prediction-Based Dynamic Load-Sharing Heuristics", *IEEE Trans. Parallel and Distributed Systems*, Vol. 4, No. 6, June 1993.

[5] K. G. Shin, C.-J. Hou, "Design and Evaluation of Effective Load Sharing in Distributed Real-Time Systems", *IEEE Trans. Parallel and Distributed Systems*, Vol. 5, No. 7, July 1994.