International Research Journal of Engineering and Management Studies (IRJEMS)
Volume: 03 Issue: 03 | March -2019 e-ISSN: 2395-0126

Design Framework For Job Scheduling Of Efficient Mechanism In Resource Allocation Using Optimization Technique.

M.Rajarajeswari 1,
1. Research Scholar,
Dept. of Mathematics ,
Karpagam University,
Coimbatore

P.R.Kandasamy[2]
2.Professor and Head,
Dept. of M.CA, Hindusthan Institute of Technology,
Coimbatore.

T.Ravichandran [3]
3.The Principal,
Hindusthan Institute of Technology,
Coimbatore.

.

Abstract:

This work aims at building a distributed system for grid resource monitoring and prediction. We present the design and evaluation of system architecture for grid resource monitoring and prediction. Key issues for system implementation, including machine learning based methodologies for modeling and optimization of resource prediction models. We specifically focus on online updating resource information centers to use by local schedulers based on assumed hierarchical model. We used knowledge extraction methods to provide some helpful predictions to classifying grid nodes. A positive point of this research is that schedulers don't waste extra time for getting up-to-date information of grid nodes. The experimental result show the advantages of our approach compared to other conservative methods, especially due to its ability to predict the behavior of nodes. Also an experimental result proves that the efficiency and accuracy of our system meet the demand of online system for grid resource monitoring and prediction.

Keywords

Machine learning, PSO algorithm, Optimization, Prediction and Monitoring, Grid Resources

Introduction

Grid Computing removes the limitations that exist in traditional shared computing environment, and becomes a leading trend in distributed computing system. It aggregates heterogeneous resources distributed across Internet, regardless of differences between resources such as platform, hardware, software, architecture, language, and geographical location. Dynamically sharing resources gives rise to resource contention. One of the challenging problems is deciding the destination nodes where the tasks of grid application are to be executed. From the perspective of system architecture, resource allocation and job scheduling are the most crucial functions of grid computing.

hese functions are based on adequate information of available resources. Thus timely acquiring resource status information is of great importance in ensuring overall performance of grid computing [1]. There are mainly two mechanisms for acquiring information of grid resources: grid resource monitoring and grid resource prediction. Grid resource state monitoring cares about the running state, distribution, load and malfunction of resources in grid system by means of monitoring strategies. Grid resource state prediction focuses on the variation trend and running track of resources in grid system by means of modeling and analyzing historical monitoring data. Historical information generated by monitoring and future variation generated by prediction are combined together to feed grid system for analyzing performance, eliminating bottleneck, diagnosing fault, and maintaining dynamic load balancing, thus to help grid users obtain desired computing results by efficiently utilizing system resources in terms of minimized cost, maximized performance or tradeoffs between cost and performance. To reduce overhead, the goal of designing a grid resource monitoring and prediction system is to achieve seamless fusion between grid technologies and efficient resource monitoring and prediction strategies. Resource monitoring is a basic function in most of computing systems. Along with grid development, monitoring tools have been evolving to support grid computing, such as those developed in the PAPI project [2], Iperf

project [3 In addition, some projects have designed distributed monitoring module of their own, such as GMA (Grid Monitoring Architecture) project Digital Object Identifier [6] and Autopilot project [7]. The monitoring techniques employed by such projects are partly compatible to grid environment, thus fit for achieving grid resource monitoring. Resource monitoring alone, however, can only support instantaneous resource information acquisition. It cannot generalize the dynamic variation of resources. Resource state prediction is inevitable to fill this gap. Typical previous prediction systems, such as NWS [8] and RPS [9], can provide both monitoring and prediction functions. However, dynamic features of grid resources were not taken into consideration in these design frameworks. Nevertheless, these projects are usually restricted in certain applications. In summary, previous approaches have the limitation of being unable to achieve seamless fusion of various components and overall simplification of system structure using a universal scheme. This paper reports our effort aiming at building a distributed system for grid resource monitoring and prediction.

The rest of this paper is organized as follows: Section 2 gives the problem statement of grid resource monitoring and prediction. Section 3 provides the overall system architecture based on design principles defined. Section 4 discusses the key issues for building prediction components. Section 5 gives a description of proposed optimization algorithm. Section 6 explains the prototype system and evaluates its performance and overhead. Section 7 closes the paper with conclusions as well as indication for future works.

2. Problem statement of Grid resource monitoring and prediction.

We first outline the main design principles of our system. Then, we present our overall system architecture design that seamlessly integrates various cooperating components to achieve high performance. We discuss the key issues in machine learning based prediction, and justify our decisions by comparative studies through extensive simulations.

We present a new optimization algorithm called Parallel Hybrid Particle Swarm Optimization (PH-PSO), and show its effectiveness. We then discuss the implementation and performance evaluation of a prototype system.

3. Proposed System

3.1Construction of Grid Computing Architecture:

In our system, resource sensors and prediction models are periodically has been placed and executed to generate up-to-date information for users. Thus, monitoring and prediction components can work well if some of nodes are down. Executing jobs is the fundamental function of a grid system, so embedded monitoring or prediction components should minimize overhead to guarantee grid's normal service. We deploy resource sensors on computing nodes since it's inevitable, they also run and sleep dynamically to reduce overhead, while we deploy other components out of computing nodes to avoid extra overhead.

Grid users do not need traversal of all the nodes or grid expertise to get information. We design a uniform and friendly interface component for accessing the information monitored or predicted. Computing grid system architecture maintains a service container for taking grid jobs, such container should be reused for seamless fusion between a grid environment and our system. Therefore, we design a series of supporting services: monitoring service, prediction service, evaluation service, and information service. These services are deployed on service containers of distributed nodes, and all the functions are realized through dynamic collaboration among them.

Besides, the resource information is managed using a hierarchical structure. Monitoring service is deployed on each computing resource node. Two types of mechanism are defined for information acquisition: local register and group register. Local register timely collects information from resource sensors to monitoring service and from prediction models to prediction service, while group register timely collects information from both services and aggregates them for storage or publication. In order to provide friendly interface to grid users, a web server is set on information node for customizing request and publishing information. Therefore, grid user needs nothing but a browser.

3.2 Prediction of Task and Resources Using Machine learning based strategies

Fix a machine learning algorithm for the prediction model, and set its default hyper parameters. Separate the sample set into three parts: training set, validation set, and test set. Feed the learning algorithm with a sample of training set, and repeat it one by one until all samples are used. For some algorithms, the training procedure runs only once; for others, iterations are needed. Feed

the trained model with all samples of validation set, and record the errors between true data and predicted ones. Grid user logs on information node and customizes three terms before sending a monitoring/prediction request: which node, which resource type, and how long the prediction will last. A prediction request is then created accordingly and sent to information service. Information service launches the monitoring work flow by sending a monitoring request to the monitoring service.

A resource sensor is activated as requested, and timely updated monitoring data are then sent back to information service through local and group registers; historical records are stored in the database for achieving prediction. Information service chooses a prediction service and sends a customized prediction request. Prediction service sends subtasks to evaluation services for feedback, and then fixes the model with best performance out of comparison on evaluation results. If an evaluation service is time out, prediction service will redirect the subtask to another one. Prediction service feeds the fixed prediction model with timely updated monitoring data for prediction, and then timely updated prediction data are sent to information service through local and group register; for checking prediction error.

Grid user gets the resource information monitored or predicted from a browser. If upon request or prediction error exceeds a certain threshold, the prediction service will reload the latest historical data for model optimization. Information service terminates the monitoring or prediction procedure when the customized time is used out.

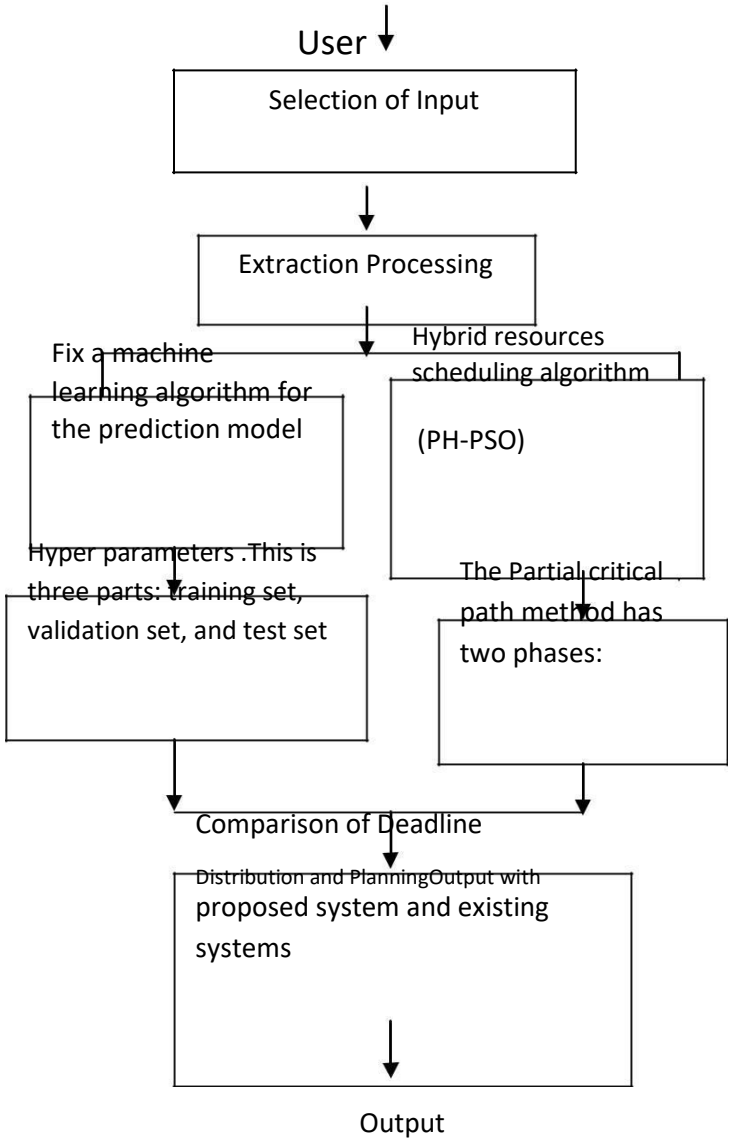
3.3. Construction of Hybrid resources Scheduling Algorithm

PSO is selected as the auto optimization strategy for Hybrid resources scheduling algorithm
(PH-PSO). There are mainly two types of PSO distinguished by different updating rules for calculating the positions and velocities of particles which is task and resource. Hyper parameter selection is a kind of continuous optimization problem, and feature selection is a kind of binary optimization problem. Concerning our optimization problem definition, this is a parallel optimization algorithm which hybridizes continuous PSO and binary PSO together, namely PH-PSO.

The algorithm is initialized with a population of random particles and searches a multi dimensional solution space for optima by updating particle historical prediction records are stored in the database

generations. Each particle moves based on the direction of local best solution discovered by itself and global best solution discovered by the swarm. Each particle calculates its own velocity and updates its position in each iteration until the termination condition is met.

. Preprocessing the sample set with corresponding features as well as candidate model with corresponding hyper parameters according to particle representation. Fitness evaluation in parallel by using validation set to evaluate candidate model, and then calculate fitness of particle.



"Figure.1 Construction of Hybrid resource"

Need to update velocity and position of each particle, if the condition is satisfied. Output the global best position, and prepare the sample set with selected features and prediction model with selected hyper parameters according to the representation of the global best position.

4. Discusses the key issues for building prediction components

4.1 Estimation of Partial critical paths of Work flows:

The Partial critical path method has two main phases to estimate the work loads namely Deadline Distribution and Planning.

In the first phase, the overall deadline of the workflow is distributed over individual tasks, such that if each task finishes before its sub deadline then the whole workflow finishes before the user defined deadline.

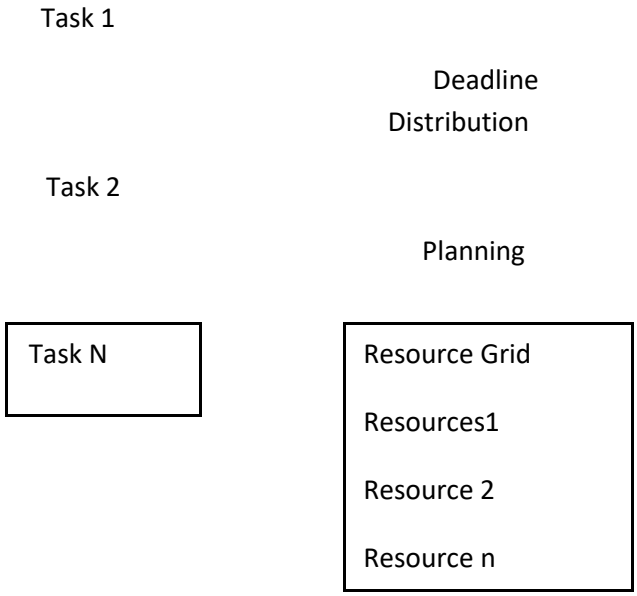
In the second phase, the scheduler has to select the cheapest service for each task while meeting its sub deadline. Our contribution is the deadline distribution method which is based on a Critical Path heuristic. Critical path heuristic are widely used in workflow scheduling. The critical path of a workflow is the longest execution path between the entry and the exit tasks of the workflow.

Most of these heuristics try to schedule critical tasks (nodes), i.e., the tasks belonging to the critical path, first by assigning them to the services that process them earliest, in order to minimize the execution time of the entire workflow.

Our deadline distribution method is based on a similar heuristic, but it uses the critical path to distribute the overall deadline of the workflow across the critical nodes. After this distribution, each critical node has a sub deadline which can be used to compute a sub deadline for all of its parent nodes, i.e., its (direct) predecessors in the workflow.

Then, we can carry out the same procedure by considering each critical node in turn as an exit node with its sub deadline as a deadline, and creating a partial critical path that ends in the critical node and that leads back to an already assigned node, i.e., a node that has already been given a sub deadline.

In this method, this procedure continues recursively until all tasks are successfully assigned a sub deadline. Finally, the planning method schedules the tasks according to their sub deadlines.



"Figure 2: critical Path Scheduling Workflow mechanism"

5. Construction of Resources Management systems based on Workloads

Resource management system is composed of multiple components like strategies model for estimating the resources and task. After Estimating grid resource information, we need to estimate the critical path for the Task to complete the execution in short time using deadline distribution method and planning method.

Having all these functionality we need to design the algorithm to assign a job to predicted resource through a scheduling mechanism. Dynamic scheduling that selects services during the execution when a task is ready to start through Decrease Cost Path Assigning method, Optimized Path Assigning method, Assigning Deadline to the Parents method, at last Scheduling method for computing to achieve the QOS for the resource management and by placing a cost function as important strategies.

6. Experimental Results

6.1Performance is computed against Memory usage And CPU usage with Existing System

The parallel CPU time of combinational/individual optimization is compared in Figs.2. From each layer, we can see that the optimization time does not show a remarkable tendency as step q increases. 0H costs more time than FH and F0. This indicates that the model's training time can be obviously reduced by feature selection rather than hyper parameter selection. It is clear that the optimizing time of combinational optimization FH is rather short by means of parallelization, namely within 3 seconds on both data sets.

"Table 1. Comparison Memory and CPU cost for PSO and PH-PSO

	PSO	PH-PSO
Memory	4.3	2.4
CPU	2.5	1.4

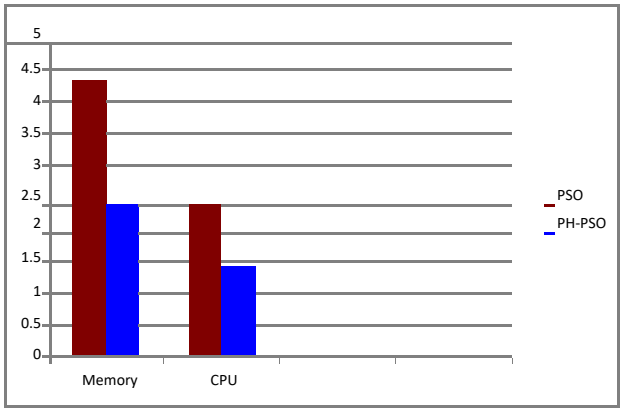


Figure 3. Memory and CPU cost for PSO and PH-PSO

6.2 Mean Absolute Error is calculated against load of resource and Bandwidth of Network.

High accuracy and efficiency is the primary design goal of prediction subsystem. We present the prediction and optimization results of bandwidth and host load data sets.

The global best fitness of combinational optimization FH during each iteration was recorded to evaluate the convergence performance. Landscape comparisons among different q values are shown in Fig.4

"Table 2: Load of resources and Bandwidth against Existing with proposed system"

Load of Resources	PSO	PH-PSO
Band Width	4.3	2.4
	2.5	2.3

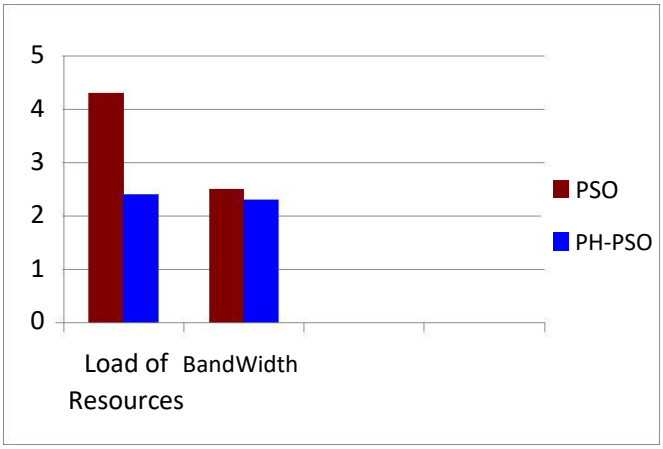


Figure 4: Load of resources and Bandwidth against Existing with proposed system

A trend is obvious on host load data set that the global best fitness decreases clearly as the prediction step q increases. While such trend is not found on bandwidth data set, which implies that the bandwidth variation has got more noise than host load. It is also implied in these sub-figures that the combinational optimization converges during proper iterations for most of the q values considered.

7. Conclusion

We proposed a distributed resource monitoring and prediction architecture that seamlessly combines grid technologies, resource monitoring and machine learning based resource state prediction. An important side of Grid middleware is resource management. These functions are based on adequate information of available resources. Timely acquiring resource status information is of great importance in ensuring overall performance of grid computing.

We specifically focuses on online updating resource information centers to use by local schedulers based on assumed hierarchical model. Moreover, we used knowledge extraction methods to provide some helpful predictions to classifying grid nodes based on job's features. A positive point of this research is that schedulers don't waste extra time for getting up-to-date information of grid nodes.

The experimental result show the advantages of our approach compared to other conservative methods, especially due to its ability to predict the behavior of nodes based on comprehensive data tables on each node. Evaluations are performed on a prototype system. Also an experimental result proves that the efficiency and accuracy of our system meet the demand of online system for grid resource monitoring and prediction.

This system consists of a set of distributed services to accomplish all required resource monitoring, data gathering, and resource state prediction functions. We defined a universal procedure for modeling and optimization of resource state prediction. In building prediction model of multi-step-ahead, ANNs and SVRs were compared concerning both efficiency and accuracy criteria. Comparative For the expectation of achieving higher performance, we compared Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) for prediction models' hyper-parameter selection. Comparative simulations indicate that the PSO achieves lower error and costs less optimizing time than GA.

In the prototype system, we implemented a series of sensors which cover most of resource measures. Overhead evaluation shows that the monitoring subsystem does not bring obvious influence on computing nodes. A Parallel Hybrid Particle Swarm Optimization (PH-PSO) algorithm was proposed which combines discrete PSO and continuous PSO, for the purpose of combinational optimization of prediction model. Evaluation results indicate that the model of PH-PSO meets the accuracy and efficiency demand of a demand system.

8. Reference

[1]L.F. Bittencourt and E.R.M. Madeira, "A Performance-Oriented Adaptive Scheduler for Dependent Tasks on Grids," Concurrency and Computation: Practice and Experience, vol. 20, no. 9, pp. 1029- 1049, June 2008.

[2] F. Wolf and B. Mohr, "Hardware-Counter Based Automatic Performance Analysis of Parallel Programs," Proc. Conf. Parallel Computing (ParCo '03), pp. 753-760, Sept. 2003.

[3] J. Dugan et al., "Iperf Project," http://iperf.sourceforge.net/, Mar. 2008.

[4] M. Livny et al., "Condor Hawkeye Project," Univ. of Wisconsin- Madison, http://www.cs.wisc.edu/condor/hawkeye/, Sept. 2009.

[5] M.L. Massie, B.N. Chun, and D.E. Culler, "The Ganglia Distributed Monitoring System: Design, Implementation, and Experience," Parallel Computing, vol. 30, no. 7, pp. 817-840, July 2004.

[6] A. Waheed et al., "An Infrastructure for Monitoring and Management in Computational Grids," Proc. Fifth Int'l Workshop Languages, Compilers and Run-Time Systems for Scalable Computers, vol. 1915, pp. 235-245, Mar. 2000.

[7] J.S. Vetter and D.A. Reed, "Real-Time Performance Monitoring, Adaptive Control, and Interactive Steering of Computational Grids," Int'l J. High Performance Computing Applications, vol. 14, no. 4, pp. 357-366, 2000.

[8] D.M. Swany and R. Wolski, "Multivariate Resource Performance Forecasting in the Network Weather Service," Proc. ACM/IEEE Conf. Supercomputing, pp. 1-10, Nov. 2002.

[9] P.A. Dinda and D.R. O'Hallaron, "Host Load Prediction Using Linear Models," Cluster Computing, vol. 3, no. 4, pp. 265-280, 2000.

[10] E. Caron, A. Chis, F. Desprez, and A. Su, "Design of Plug-in Schedulers for a GRIDRPC Environment," Future Generation Computer Systems, vol. 24, no. 1, pp. 46-57, 2008.

[11] P.A. Dinda, "Design, Implementation, and Performance of an Extensible Toolkit for Resource Prediction in Distributed Systems," IEEE Trans. Parallel and Distributed Systems, vol. 17, no. 2, pp. 160-173, Feb. 2006.

[12] A.C. Sodan, G. Gupta, L. Han, L. Liu, and B. Lafreniere, "Time and Space Adaptation for Computational Grids with the ATOPGrid Middleware," Future Generation Computer Systems, vol. 24, no. 6, pp. 561-581, 2008.

[13]M. Wu and X.H. Sun, "Grid Harvest Service: A Performance System of Grid Computing," J. Parallel and Distributed Computing, vol. 66, no. 10, pp. 1322-1337, 2006.