Proceedings of the 5th WSEAS International Conference on Telecommunications and Informatics, Istanbul, Turkey, May 27-29, 2006 (pp273-275)

A Dynamic Task Scheduling Algorithm in Grid Environment

Wang Da-zhen, Zhan Jun-shan,Wan Fang, Zhu lei
Computer Science Department
Hubei University of Technology, Wuhan 430068
P.R.China
http://www.newtelesoft.com

Abstract: Task scheduling is one of the effective methods in grid computing environment. In this paper, we introduce swarm intelligence mechanism into task scheduling, and propose a new dynamic task-scheduling algorithm. This algorithm is used in the simple resource pool model and can effectively organize independent tasks based on the interaction model between a wasp colony and its environment. Through the experiment, the algorithm has been proved more efficient and more adaptive to the dynamic grid environment than other ones in the simple resource pool model of grid environment.

Key-Words: - Grid, Task scheduling, Dynamic, simple resource pool model

1 Introduction

Two main resource models in current grid compute environment are: simple resource pool model and two-layer resource model, which is represented by Globus. In the resource pool model, all the effective resource information are recorded in the central node, usually a super computer or mainframe. In this paper, our algorithm and experiments taken are on the first environment. As Tasks allocation to different resources reasonably and optimally, is one of the key problem unsolved. Task scheduling is also one of the bottlenecks in grid computing. Most algorithms based on a meta-task model lack good adaptability, and cannot fit the need of rapid respond. In the recent years, swarm intelligence has received further attention in the cooperation field. In our paper, we proposed a new dynamic task-scheduling algorithm based on the wasp algorithm. We proved through experiment that the proposed algorithm is more adaptive to the simple pool grid environment than other task-scheduling algorithms.

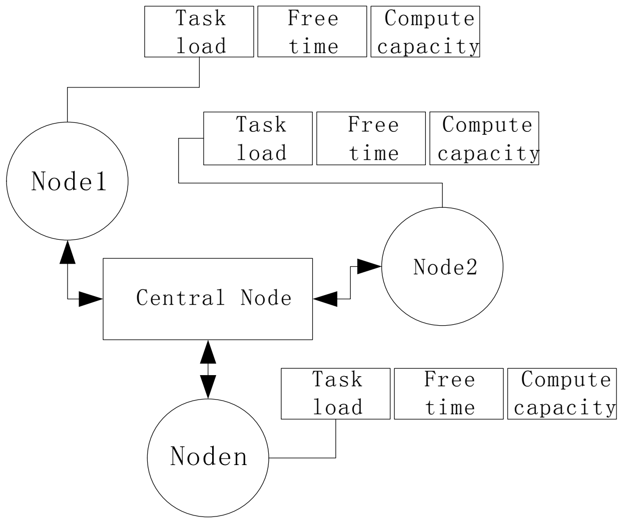

Figure 1: simple resource pool model

Central node information table
Node ID FreeTime Compute Capacity Task Load Probability

Figure 2: information structure on central node

2 Proposed Algorithms

2.1 simple resource pool model

Figure 1 is a topology of the gird environment built on simple resource pool model.

Every grid node is denoted by a table, which is maintained by node itself and will rapidly refreshed the central node resource table (described in the figure 2). Among all the fields in this table, Node ID, Free Time, Compute Capacity and Task Load are to be modified automatically by wasp agent, and the Probability will be computed out by the newest data some seconds after the table modified.

Grid node is like a wasp agent, able to collaborate with other same layer nodes and the central node. Let $N=\{ n_1, n_2, \cdots n_n \}$ denote computing nodes, $T=\{ t_1, t_2, \cdots t_m \}$ denote a set of tasks. Tasks are submitted to the grid system randomly and each task needs one computing node. Each computing resource n_j has an associated agent wasp w_j, which is in charge of finding and leaving its computing node and returning with tasks ID. Tasks submitted by users first enter into the task list in the central node of the grid. The central node is responsible for receiving tasks and is not involved in task scheduling. Let Time1 (t_i, n_j) denote the execution time of a task t_i on n_j. So, the Probability is computed as followed:

$$p(n_{j}) = \kappa + u \cdot Time1(t_i, n_j) - \eta \cdot taskload + \alpha \cdot freeTime + \beta \cdot computecapacity. \quad (1)$$

Where k., μ, η, α, β are constants.

2.2 Design of the Proposed Algorithm

A grid task often needs to match with many kinds of resources in the real environment; the situation is very complicated. At first, some introduce and relevant concepts of algorithm are as bellowed:

Internal queue: one node scheduled a task, no longer be managed other task until this task finished, this kind of node will be deployed in an internal queue.

External queue: one node can be multi-used by different task, this kind of nodes will be deployed in an external queue.

Internal and External queue's design aim to make the choice of the node more accurate and deduce probability of repletion of filtration of nodes.

Task pressure: Show the load situation of nodal resources.

Task scheduling of grid issue to look for the node suitable in the grid to implement the task, make the whole grid user have shortest task execution time. Through which, the whole grid has high handling up rate. The algorithms ran at each node of the grid, is a totally distributed algorithm based on nodal autonomy. The concrete algorithm is described as follows:

First step: Initialize. After Node was putted into grid, calculate the probability of task scheduling according to the formula (1).

Second step: Schedule task in the internal queue. Firstly, judge whether the internal formation has been already full; if so, turn to the external queue.

Third step: Schedule task in the external queue.

Fourth step: Nodal crash handling. Judge whether some node has not carried out any task of nodal deployment within certain time interval. If so, use formula (1) to calculate to take the other optimized node.

The simulation code are as followed:

```
Do while task not initiated
If(central node found)
  If(interval time not too long)
    //Go into internal queue to find fitted nodes
    If(internal queue is not full)
          {search in the internal queue to find
the node with biggest probability
          Nodes implement the task;
        Nodes modified the information on
 the central node
}
    Else
      { search in the external queue to find the node
with biggest probability}
  Else
{
    Abandon current node;
    Return to find new node;
}
Else
{
    Warning Message
   Find backup central node
   return
    }
End do
```

3 Simulations and Experiment

In this section, we will evaluate the proposed algorithm and use a simulation environment based on discrete event simulation. We simulate six computing resources and ten data sources and implement our scheduling algorithm, the min algorithm, and the heuristic algorithm. [2]. Figure 3. Comparison of the makespan(s). Three groups of tasks are chosen in our experiment. The amounts of three groups are 100, 200 and 1000, respectively.

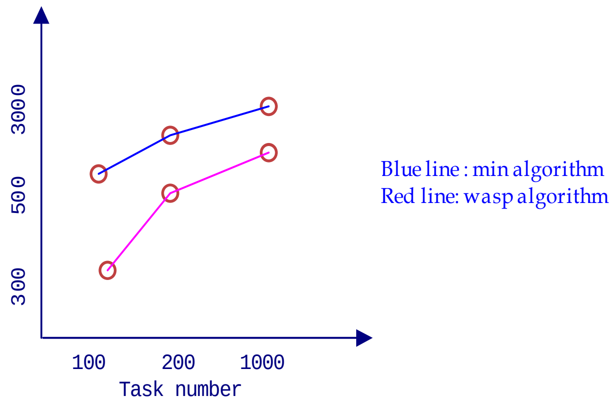

Figure 3: makespan comparison between min/wasp algorithms

From Figure 3, it is obvious that the makespan of the wasp-based algorithm is less than the min algorithm for each number of tasks. Our experiment concludes that the wasp-based algorithm performs better than the other one algorithm and shows a fine adaptability to the complex grid environment.

4 Conclusion

In this paper, we apply swarm intelligence to solve task scheduling in the simple poor model grid environment, and design a dynamic task-scheduling algorithm based on a wasp algorithm. In our algorithm, task scheduling is implemented by adding tasks to the waiting list of each computing resource instead of allocating the required resources to a group of tasks once. Results show that the proposed algorithm is higher in efficiency than the Min algorithms, and more fitted to the simple poor model grid environment.

References:

[1]. Foster I., Kesselman C.: The GRID Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, San Francisco (1998)

[2].Ding Q., Chen G.-L.: A Benefit Function Mapping Heuristic for a Class of Meta-tasks in Grid Environments. In: Proc. of First IEEE/ACM International Symposium on Cluster Computing and the Grid, IEEE Computer Society, Brisbane, Australia (2001) 654–659

[3]. Hamscher, V., Schwiegelshohn, U., Streit, A., Yahyapour, R.: Evaluation of Job-Scheduling Strategies for Grid Computing. In Buyya, R., Baker, M. (eds.): Proc of 7th Int'l Conf on High Performance Computing. Lecture Notes in Computer Science, Vol. 1971. Springer-Verlag, Berlin Heidelberg New York (2000) 191–202

[4]. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial System. Oxford University Press, Oxford (1999)

[5]. Theraulaz, G., Goss, S., Gervet, J., Deneubourg, J.-L.: Task Differentiation in Polistes Wasp Colonies: A Model for Self-organizing Groups of Robots. In Meyer, J.A., Wilson, S.W. (eds.): Proc. of the First International Conference on Simulation of Adaptive Behavior on From Animals to Animats, MIT Press, Paris, France, (1991) 346–355