Resource Scheduling in Distributed Environment

Thandar Su Nge Htwe
University of ComputerStudies,Mandalay
thandar.tdsnh@gmail.com

Abstract

Load-balancing problems arise in many applications, but, most importantly, they play a special role in the operation of parallel and distributed computing system. Scheduling in distributed operating systems has a significant role in overall system performance and throughput. In this paper, we proposed a scheduling strategy that takes into account for load balancing using Genetic Algorithm (GA).In this system, load information collector that supports to get the current status of load information and send to the scheduler. The scheduler performs the major role in our system for scheduling process. To perform the scheduling process, we used Genetic Algorithm to generate the optimize schedule. For load balancing, we demonstrate with example using genetic algorithm.

1. Introduction

Multiprocessor scheduling methods can be divided into list heuristics and Meta heuristics. In list heuristics, the tasks are maintained in a priority queue in decreasing order of priority. When a free processor is available, the task at the front of the queue is assigned to the free processor. Most list heuristics are not efficient for all situations. Genetic algorithms (GAs) are a Meta heuristic searching techniques which mimics the principles of evolution and natural genetics. These are a guided random search which scans through the entire sample space and therefore provide reasonable solutions in all situations. Many researchers have investigated the use of Gas to schedule task in homogeneous [2] and heterogeneous [3] multi-processor systems with notable success.

A distributed system provide the resource sharing as one of its major advantages, which provide the better performance and reliability than any other traditional system in the same conditions. One of the research issues in parallel and distributed systems is the development of effective techniques for distributing workload on multiple processors. The main goal is to distribute the jobs among processors to maximize throughput, maintain stability, resource utilization and should be fault tolerant in nature. Load sharing and load balancing are the further classifications of dynamic scheduling.

Load sharing struggle to avoid the unshared state in processors which remain idle while tasks compete for service at some other processor. Load balancing also do the same but it goes one step ahead of load sharing by attempting to equalize the loads at all processors. Several tasks are scheduled for separate processors, based on the current load on each CPU. Many researchers have been carried out on load balancing for many years to find the load balancing schemes with overhead as low as possible. Generally, scheduling problems are NP-hard and there are no general algorithms that can guarantee an optimal solution. In this paper, a scheduling strategy that takes into account for load balancing using genetic algorithm that can

address the scheduling problem. The objective of this paper is to get the optimized schedule for load balancing.

2. Related Works

Sandeep Sharma, et.al have studied various Static (Round Robin and Randomized Algorithms, Central Manager Algorithm, Threshold Algorithm*)* and Dynamic (Central Queue Algorithm, Local Queue Algorithm) Load Balancing Algorithms in distributed system. The performance of these algorithms are measured by following parameters: Overload Rejection, Fault Tolerant, Forecasting Accuracy, Stability, Centralized or Decentralized, Nature of Load Balancing Algorithms, Cooperative, Process Migration,, Resource Utilization. Their result shows that static Load balancing algorithms are more stable with such parameters [4].

In [5], the author presented a unified framework for resource scheduling in meta-computing systems where tasks with various requirements were submitted from participant sites. Their goal was to minimize the overall execution time of a collection of application tasks.

Derek L. Eager, et.al proposed not only a specific load sharing policy for implementation, but rather to address the more fundamental question of the appropriate level of complexity for load sharing policies. They showed that extremely simple adaptive load sharing policies, which collected very small amounts of system state information and which used this information in very simple ways, yield dramatic performance improvements[6].

Huajie Zhang proposed a five-tuple load-balancing model, on the base of four-tuple load balancing model which was proposed by other scholars. The five-tuple load balancing model involved hardware environment, scheduling environment, task allotment, load estimate, scheduling strategy and scheduling evaluation. They were all expressed through mathematical definition[**7**].

Yang Yongjian, et.al proposed EALBMA (Efficient and Adaptive Load Balancing based on Mobile Agent) and discussed its basic principles. There are some problems in the structure, reliability, performance, adaptability, and extensibility of load balancing at present. EALBMA could be applied into the load balancing on web server. It could also be used in LAN and WAN[8].

3. Background Theory

In this paper, we focus on the theory of genetic algorithm (GA) and also to balance the load.

Genetic algorithms (GAs) were first proposed by the John Holland in the 1975. The GA is a heuristic search technique that simulates the processes of natural selection and evolution. Genetic algorithm (GA) is a promising global optimization technique [9].

It works by emulating the natural process of evolution as a means of progressing towards the optimal solution. A genetic algorithm has the capability to find out the optimal job sequence which is to be allocated to the CPU. This paper proposes the genetic algorithm based technique to find out the optimal schedule. We will examine that whether genetic algorithm based scheduling will maximize the operating system performance. The algorithm starts with a population which is consists of several solution to the optimization problem. A member of population is called an individual. A fitness value

is associated with each individual. Each solution in the population or an individual is encoded as a string of symbols. These symbols are known as genes & the solution string is called a chromosome. The values taken by genes are called alleles. Several pair of individual (parents) in the population mate to produce offspring by applying the genetic operator crossover. Selection of parents is done by repeated use of a choice function. A number of individuals & off springs are passed to a new generation such that the number of individual in the new population is the same as old population. A selection function determines which string forms the population in the next generation. Each serving string undergoes inversion with a specified probability.

4. Proposed Architecture

In our proposed system, n1,n2, n3 are represented as computers .Each computer has status of load conditions. Load conditions are size of CPU, size of memory, speed of each processor, workload of each computer. And then, J1, J2 and J3 are defined job request and user requirements. In this system, load information collector and scheduler are used for performing the system operation. The two components are played a major role in our proposed system. The following figure (1) is depicted the overview of system architecture.

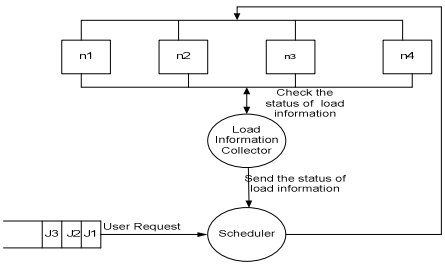

Figure1. System architecture of proposed system

According to figure (1), Firstly, a job request or user requirement enters to the scheduler to get available node. These user requirements are stored in the scheduler. And then, load information collector monitors the current status information for each node and then collected information are sent to the scheduler. This information is used by the scheduler to get available node a balance node by using Genetic Algorithm. The scheduler performs the major role in our system for scheduler process. Load balancing is to adjust the utilization of available resources among the requesting nodes to let them effectively perform the tasks of the nodes. The aim of our proposed system is to get the optimized schedule for load balancing.

A major issue is the operation of parallel computing system is that of scheduling, which is an important problem in other areas such as manufacturing, process control, economics, and operation research. To schedule is to simply allocate a set of tasks or jobs to resources such that the optimum performance is obtained. The following figure (2) describes the major work of the scheduler. The scheduler performs the vital role in our proposed system for scheduling process to get load balance.

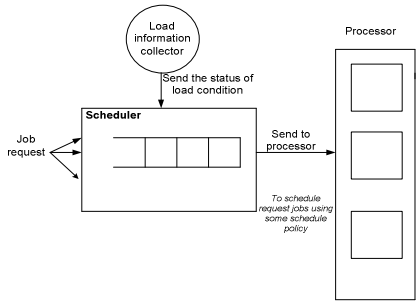

Figure2. Process of Scheduler

We have namely two kinds of components which perform major role in our system .They monitors the status of load information and the load balancing process in each node. These two components are :

Load Information collector

To achieve load balancing, load information of each node should be collected first of all. Load Information Collector is responsible for checking track of the load status of information of system resources. System resources are CPU, workload, memory, speed etc. Load Information collector collects the load status of information in each node and send to the scheduler.

Scheduler

Scheduler is responsible for the load-balanced execution of the parallel application. List of all resources of each node are stored in scheduler. In addition, it also stores job request from users. The scheduler plays a vital role in our systems for scheduling process.

5. Genetic Algorithm Approach

Now, for adapting GA to our problem, it is necessary that we develop an encoding scheme.

5.1.Encoding Scheme

If the number of tasks is **n,** each chromosome consists of tasks, T1, ….,Tn and n tasks must be scheduled on **m** processor. Each processor is shown with a number of between 1 to m and it has a task list and remaining utilization (U). At first, all Us set to 1 and task list of all processors are empty. Tasks must be scheduled on different processors. A chromosome encoding is shown in figure.3.

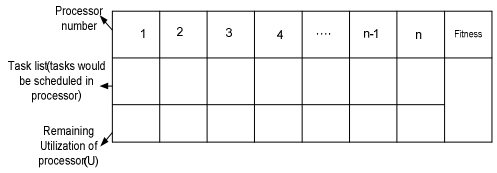

Figure3. A chromosome encoding

5.2.Generating Initial Population

Depending on the number of tasks, the number of initial population is different. With performing the following steps, chromosome would be created.

1. Set U=1 for all processors
2. Select a tasks Ti randomly from T1,….,Tn
3. Select the processor that has maximum U, if two or more processor have maximum U, select one of them by random
4. If $\leq$ U, the task is added to the task list of the selected processor
5. Repeat steps 2 to 4 until all tasks are scheduled and a new chromosome is generated.
6. Calculate fitness of the new chromosome
7. Repeat step 1 to 7 by the number of initial population.

A. Defining fitness funcition

The goal of our scheduling strategies is to use processor with the same proportion so that the balance of load on the processor is met. Therefore, the fitness value of each chromosome is computed by formula 1.

$$\text{Fitness}=\frac{\sum Ui}{\sum Pi} \quad \ldots\ldots\ldots (1)$$

Where, Ui is the number of remaining utilization and Pi is the number of processor

5.3.Example with genetic algorithm to load balance

We use an example to describe all parts of our algorithm: Seven tasks are listed in Table 1 which must be scheduled on five processors.

Table1. Attributes tasks

Task	T_1	T_2	T_3	T_4	T_5	T_6	T_7
C	3	2	5	3	4	1	1
D	6	10	8	5	7	3	10

P	1	**2**	3	4	5	Fitness
T	T_5,T 7	**T_3**	T_2	T_1, T_6	T_4	0.448
U	0.33	**0.3 7**	0.9	0.17	0.4	

Figure 4. A chromosome of initial population

Mutation

T6 is selected for a chromosome as shown in figure 5. T6 is in the task list of p4. After applying mutation, T6 is transferred to the task list of P2 and then fitness is calculated.

P	1	2	3	4	5	Fitness
T	T_5 , T_7	T_3	$T_{2,}$	T_1,	T_4	
U	0.4	0.3	0.57	0.5	0.4	

Figure5.chromosome before mutation

P	1	2	3	4	5	Fitn ess
T	T_5 , T_7	$T_{3,}T_6$	T_2	T_1	T_4	0.5 2
U	0.33	0.4	0.9	0.5	0.4	

Figure6. Chromosome after mutation

Crossover

Firstly, two parent chromosomes are randomly selected for crossover operation as shown in figure 7 and 8. In this paper, we used one point crossover. Point number 2 is chosen for crossover point. Figure 9 and 10 show two generated children by applying crossover. Figure 11 and 12 show two children after correction.

P	1	**2**	3	4	5	Fitness
T	T_5 ,T_7	**T_3**	T_2	T_1, T_6	T_4	0.448
U	0.33	**0.37**	0.9	0.17	0.4	

Figure7.Parent1 before crossover

P	1	2	3	4	5	Fitness
T	T_5 , T_7	$T_{3,}T_6$	T_2	T_1	T_4	0.52
U	0.33	0.4	0.9	0.5	0.4	

Figru8. Parent2 before crossover

P	1	2	3	4	5	Fitness
T	T_5 , T_7	T_3	$T_{2,}$	T_1,	T_4	
U	0.4	0.3	0.57	0.5	0.4	

Figrue9. Child1 after crossover

	1	2	3	4	5	Fitness
	T_5 , T_7	T_3	$T_{2,}$T6	T_1	T_4	0.45
	0.4	0.38	0.57	0.5	0.4	

Figure10. Child2 after crossover

P	1	2	3	4	5	Fitness
T	T_5 ,T_7	T_3, T_6	T_2	T_1,T6	T_4	
U	0.4	0.4	0.9	0.5	0.4	

Figure11. Child1 after correction

P	1	2	3	4	5	Fitness
T	T_5 ,T_7	T_3, T_6	T_2	T_1, T6	T_4	0.52
U	0.4	0.4	0.9	0.5	0.4	

Figrue12.Child2 after correction

Finally, we can choose maximum fitness for load balancing.

6. Conclusion

In this paper a new scheduling strategy is suggested that takes into account for load balancing. And then, we used genetic algorithm to generate the optimized schedule. Thereafter, we demonstrate with example using genetic algorithm approach. We will expect the method of proposed system is outperformed their results. Therefore, as our future work, we expect to perform a full implement of this proposed method.

References

[1] M.Nikravan,M.H. Kashani,"A Genetic algorithm for process scheduling in distributed operating systems considering load balancing", *Proceedings 21st European Conference on Modelling and Simulation Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007*

[2]

[3] E.Hou, n.Ansari, and H.Ren. A genetic algorithm for multiprocessor scheduling. *IEEE Transactions on parallel and Distributed Systems*,5(2):113-120,February 1994.

[4] Dynamic task scheduling using genetic algorithms for heterogeneous distributed computing. *Proceedings of the 19th IEEE international Parallel and Distributed processing symposium(IPDPS 05).*

[5] Sharma Sandeep, Singh Sarabjit and Sharma Meenakshi (2008), Performance Analysis of Load Balancing Algorithms, *World Academy of Science, Engineering and Technology*.

[6] Ammar H. Alhusaini et.al," A Unified Resource Scheduling Framework for HeterogeneousComputing Environments", 6 Aug 2002 - ieeexplore.ieee.org.

[7] Derek L.Eager, et.al," Adaptive Load Sharing in Homogeneous DistributedSystems", *IEEE Transactions on Software Engineering*, VoL. SE-12, No.5,May 1986.

.

[8] Huajie Zhang," On Load Balancing Model for Cluster Computers**",** IJCSNS *International Journal of Computer Science and Network Security*, VOL.8 No.10, October 2008.

[9] YANG Yongjian, et.al "EALBMA: A Load Balancing Model Based on Mobile Agent", *Journal of Communication and Computer*, ISSN1548-7709, USA

[10] David E.Goldberg, Genetic Algorithms in Search Optimization & Machine learning, Second Reprint, Pearson Education Asia pte.Ltd., 2000.