International Journal of Enhanced Research in Management & Computer Applications
ISSN: 2319-7471, Vol. 4 Issue 8, August-2015

The Efficient Load Balancing in Computer

Kapil Sharma[1], Mr. Lokesh[2]

[1]M. Tech Student, RIEM, Rohtak, Haryana
[2]HOD, CSE Dept. RIEM, Rohtak, Haryana

ABSTRACT

Recently CPU workload hardware Technology and multiprocessor Service are developing rapidly. Mathematically computation must be completed within a certain time period. Through the mathematical Analysis our Study Identify various useful result that should be interest in system designer. If all the parallel computer are not same type means not same configuration then proper load balance not occur so some computer finish their work earlier than other and sit ideal which degrade the performance of multicomputer system. We introduce new Algorithm for program development on multicomputer environment called ODDA (optimize data distribution algorithm) to address the proper load balancing purpose. This algorithm works in the master slave model. The main processor called the master processor which dynamically assigns the workload to the remaining cooperative slave processor in run time environment. The efficient parallel computer has 3 element - collection of device, a Network connect to these computer and s/w that enable used to share data between them. MPI and PVM is the best Software used for message passing interfacing computation between them. Multiprocessor computation is the best way of resource handling and scheduling strategy. This paper contain Dynamic approach load balancing for process migration using process level environment employed parallel programs.

Keywords: Master processor, slave processor, Data distribution, Task manager, MPI, heterogeneous platform.

I. INTRODUCTION

There is continual demand for greater computation power from computer system there is currently possible area requiring great computational speed include numerical problem of scientific and engineering problem. Multiprocessor system is very efficient at evaluate task with uniform communication and computation pattern. Complex method and Algorithm use to solve problem in parallel system in order to obtain a well balanced overall load of the system. The goal of our Algorithm to proper load balancing the multicomputer system and analysis about what amount of workload should we give to the cooperative slave processor. In our program Development modal (ODDA) the way to the Divide the problem is functional and size oriented and distributes the data to other process is capacity and functional oriented. One time one processor do one work and processor are totally unambiguous means same data must not send to more than one slave processor. Slave processor execute and compiled their work indecently and give to the main master processor for extra or Integrated result purpose. If the workload on these member processor is well balanced defiantly we will achieve high performance. If workload increase rapidly at the same time execution time as response

Time also increases. We can use limited number of System because it will increase cost which against the economical policy. The parallel computer in the user side they are group of machine but function as single virtual system .proper task distribution refer to way process are assign to run on cpu. The assignment is carried out by the scheduler and dispatcher. They are operate by 3 way architecture means primary processor, secondary processor and middle layer. The work of middle layer to connect their S/W component to their Application .MPICH2 is the best middle layer .Load balancing is mechanism which take advantage of communication facility between number of parallel system by exchange the information and task between any two node target to obtain the effective performance. An effective load balancing also need to take care of various factor of performance evaluate which include parameter such as load estimation , load cost , performance indices , system stability, communication latency, communication data volume. In our proposed algorithm we have to need cost as well as response time from every master to slave information.

II. BACKGROUND CONCEPT

In this section we discuss the load balancing concept and describe their important to multicomputer environment .with memory heterogeneity. Load balancing algorithm is two type.

1. Static : In the static load balancing algorithm the information about the data known previously . This information may be obtain either in run time or compile time. This mechanism of load balancing are restricted to predetermined workload application and can not be use as a structure or environment where dynamic data distribution occur.

2. Dynamic: It is applicable in wider and broad class of parallel application. It is two type which works in master slave message passing routine modal Centralized and Decentralised. In Centralized dynamic load balancing the master processor hold the collection of task. It maintain global queue in which task stay as well as functionality of slave processor around it .after complete the local task slave processor send result to the master processor. Centralized load balancing will further divide into two parts (1) predicate the future algorithm (2) task queue algorithm. task queue algorithm distribute the tasks they target parallel processing strategy consisting of Individual task and scheduling them in the share memory platform. Another algorithm predicate future algorithm can distinct both task and data by assumption or probability base theory future requirement based on performance of past information but this algorithm is not user friendly and same problem occur which occurs in SJF algorithm. Decentralized load balancing is same as centralized algorithm but one difference that master processor maintain more sub master processor which makes the work easier of main processor as well as reduce Burdon and enhance the scalability and reliability.

III. RELATED ALGORITHAM

There are many algorithms which have been design to proper load balancing in the parallel computer environment.

1. Domain decomposition or portion algorithm: It is very easiest algorithm target to assign to equal number of data to partition while reduce the cost of communication. The characteristics of sub domain is that they have same type of task but they have different data. also mathematically the union of the sub domain is equal to main domain and sub domain are also independent to each other in the view of data means if we take intersection then it will give null value. at the same time if load is huge per sub domain then defiantly computation cost increase to finish the work in deadline or delay occur.

2. Traditional load balancing algorithm of Iterative routine: It is dynamic approach algorithm it is iterative nature means loop is continue till the proper load balancing not occur . Main processor collect the information from slave processor continually. The structure of Iterative routine is denoted by loop and it is form of function. The value of present n dimensional matrix evaluated by previous n dimensional data of matrix and it will execute continue till our condition not fail. Traditional load balancing algorithm measure the computation time of one iteration calculate the further iteration and re distribution this workload if previous result is not satisfactory. Generally other algorithm can fail in that environment where the load is overestimated but this algorithm work properly to use proper cost estimation modal.

3. Round Robin algorithm : In this algorithm the data packet is divided evenly to all processor . The order of process allocation is locally independent. This algorithm works well when the data is equally and number of processor smaller than no. Of processor . one of its benefit is that it does not require IPC means inter process communication . This algorithm is used in spatial purpose Application where HTTP request are similar nature and there by distributed equally.

4. Randomize Algorithm: The randomize
Algorithm is an algorithm applies as a probabilistic approach oppose to deterministic nature, which is followed by the Round Robin algorithm. In this algorithm the task is handling by particular process i with probability pi. This work s very well where each processor have equal load but fail in that environment where various load come and probability to handle the data varying. It is use in alteration of Round Robin when there are large numbers of node as maintaining the queue of node for Round Robin become an overhead.

5. Central Queue Algorithm: This algorithm works on principal of Dynamic distribution .It maintain the central queue and store new activities and un fulfillment request as FIFO queue in the main master processor . new task come and store in the central queue and of master first. When the slave processor Ideal then they request for data then master processor allot the data to the appropriate slave processor and remove the task from central queue otherwise task remain in the queue. This algorithm is slave initiated program environment.

IV. METHODOLOGY OF PROPOSED ALGORITHM

We want to distribute the data to the number of slave processor around the master processor attempt to minimize load imbalance as well as communication delay between them. Our modal of algorithm is ODDA and its occurs in message passing routine master slave architecture. Here master processor is very power full processor in the view of high cost equipment use in , high speed , high level interfacing to slave processor , huge structure etc. it maintain and handle all the cooperative slave processor their functionality , data and result. The master processor have 3 type of queue exist for different different purpose the size of queue are different different size .first queue is Q[i] where i is the size of queue i will enough size it is generally same size of page size which resistant in the main memory .when new job or task allotted by the Task manager then this new job stay in this queue. A other queue is the TD (task distribution queue) which is Q[j] smaller than job queue some data copied randomly from job queue and store in the Q[j]. j is the number of slot in this queue which must be greater than the number of processor + 1 .there is third queue called Response queue which is smallest.

It store the time as well as cost of total communication and computation in particular slot of Q[k] between each slave processor to master processor. Now First of all the task manager request to the main master processor whether master is ready to receive the data or not. Master can receive the data when it have been distribute all of data to slave processor and Q[i] is empty otherwise no task can be enter master acknowledgment to task manager ready to receive data is true or false. if false then task manger continuously check the master . if ack. is true means master processor ready to receive data and put in Q[i].if the memory is the paging modal then the master can not reject the data of the task manager but if the main memory is the segmentation modal then it can be happened that segment can greater than the page size also Q[i]. So main processor reject the data and response to task manger to send data in page size so paging in segmentation occur and task manager send the task to main processor. now in the next step some random data copy from Q[i] to Q[j].now Q[j] broadcast these data to the all processor including master processor. Every processor evaluate the data and send the data as well total time spent in this processing. the result store in response queue here check the result and remove if true, it store the cost and time ratio in every slot i of processor i. Cost of processing previous known by master processor .

If the architecture is of symmetric multicomputer means they belong to same distance then propagation delay will be same and cost of processing will be inversely proportion to is(total time – propagation time).means if cost is high then defiantly response time will be early. but if the architecture not symmetric means slave are remotely exist in different different distance then we logically put the system in average distance and apply the same algorithm .now the main processor give the data according to cost time ratio. if C/T is low then master give small data to particular slave processor according to formula which we have to use in algorithm part. when processor complete it's task then they give result to master processor .the result store in Q[j].master set the counter count which is initially set to the number of processor around the master processor how many processor around it master know by the size of communicator in the mpi routine. So when data is receive then counter decrement by 1. If the counter ==0 then all processor give response to the master and at last master evaluate final result and give to task manager. In this way this algorithm work. Now we write the algorithm.

V. PROPOSED ALGORITHM

1. Task manager request to master processor whether Q[i] is empty or not.
2. If (Q[i] ==true and ready =true and (busy= 0 or
1))
{
Master response to task manager that it is ready to receive data.
}
Else
{
Acknowledgment to task manager to wait some times. Repeat step 1 and 2.
}
3. If step 2 is true then task manager send the data packet D to master processor.
4. If (D>Q[i])
{
Master reject the packet response to task manager to further fragment to data equal to Q[i] or less than Q[i].Q[i]== size[page in main memory].
Else
{
D store in Q[i].

}
5. Copy some random element from Q[i] to Q[j].
6. Q[j] broadcast data to all processor by broadcast routine in message passing interface. J> number of processor (p)
7. If (Finish(Di == True))
{
Processor send result R(i) to master and store in Q[j]
}
8. Master set counter cont and initially= p(total number of processor.
9. While(ack_recv (i))
{
Cont= cont-1;
Calculate total time spent for processing Ti by mpi function time(T1 , T2).
Time of processing t(i) = (total time – propagation delay)i.
10. Master statically know the cost of processing between each master slave couple which is store in Q[j].
11. If(arch is symmetric)
{
Take ratio of cost and time between each master slave pair. And cost of processing = K/(absolute time between master slave i) . Where K is constant.
}
Else
{
Take the average distance of each slave processor from master and repeat 1 to 8 step
}
12. Obtain the radio result C1/T1 ,C2/T2 ,C3/T3 etc.
13. Data distribution by master processor.
14. Data given to i number processor=
((Ci/K)/∑Ci/K))* D or ((1/ti)/∑1/ti))* D
The range of ∑Ci/ti is from i=1 to total number of processor P.
15. The slave solve the problem and send to the master processor.
16. Master set the counter cont which is set to P
17. while(result_recv (i))
{
Cont=cont-1;
Print : the result of i number of processor obtain. }until cont=0;
18. if (cont==0)
{
All processor give result and store in Q[j].
}
19. master evaluate total result and send to task
manger.
20. Repeat step 1 to 19 for new task.

VI. RESULT AND MATHEMAICAL ANALYSIS

Master processor 3 queue which store the task as well as record the response time which is spent between master and slave processor for data processing purpose .suppose there is a job count of 100 number come in the queue of master processor. If there are 4 slave processor around master processor S1,S2,S3,S4 if the cost of the processing of task between master and other processor are (constant factor in cost time ratio ÷ absolute time)if suppose that constant factor is 6 then cost are respectively 0.75,1,0.5,1.2,3 unit and it has been clear that master processor which is processor 5 take highest fraction of job lowest fraction of job given to processor 3. if the structure is symmetric so all slave processor sit just equally distance so cost of processing is inversely proportional to the processing time between each slave and master processor. suppose that the time of each slave processor S1 ,S2 ,S3, S4 to take to evaluate the task of sample data is 8 ,6 ,12 , 5 time unit and 2 time unit spend in the master processor. Then we take the ratio of cost and evaluate processing time.

1. Cost and approximate time ratio between master processor and slave S1 = (6/8^2) =0.09375
2. Cost and approximate time ratio between master processor and slave S2 = (6/6^2)=0.1666.
3. Cost and approximate time ratio between master processor and slave S3 = (6/12^2) =0.041.

4. Cost and approximate time ratio between master processor and slave S4 = (6/5^2) = 0.24
5. Cost and approximate time ratio of master processor are = (6/2^2) = 1.5
Sum of all to all cost time ratio are = 0.09375 +0.1666 + 0.0410+ 0.24 +1.5 = 2.04

If we want to reduce the response time then we have to multiply have to multiply (expected absolute time ÷constant factor K) in every cost and approximate time ratio.here constant factor we have take 6.then we sum all time ratio of all processor is = 0.125+ 1.66 +0.0833+0.2+0.5=1.074.

Data given to slave processor 1 = 11.63=11. Data given to slave processor 2 = 15.51=16. Data given to slave processor 3 = 7.75= 08. Data given to slave processor 4 = 18.62 =19. Data given to slave processor 5 = 46.55= 46

Response time = (absolute time taken by slave processor÷.number of dummy data broadcasted)×number of data allotted to that processor. suppose number of dummy data is 8.

If we equally divide the data to all processor then response time given by slave processor

1 = Response time = (absolute time taken by slave processor ÷ .number of dummy data broadcasted) ×
(number of total data ÷number of processor) = (8÷8) ×(100/5) =20 unit time
2. Same response time by processor 2 =15 time unit.
3. Same response time by processor 3 =30 time unit.
4. Same response time by processor 4 =12.5 time unit.
5. Same response time by processor 5 = 5 time unit

REFERENCES

[1]. Legrand, A., Renard, H., Robert, Y., Vivien, F.: Mapping and load-balancing iterative computations. IEEE T. Parall. Distr. 15, 546--558 (2004).
[2]. Hor93] G. Horton, A Multi-level Diffusion Methods for Dynamic Load Balancing, Parallel Computing 19(1993), pp. 209-218.
[3]. Legrand, A., Renard, H., Robert, Y., Vivien, F.: Mapping and load-balancing iterative computations. IEEE T. Parall. Distr. 15, 546—558 (2004)
[4]. S. Malik Dynamic Load Blancing in Network of Work Stations.
[5]. search Report, 19 November, 2000
[6]. G. Cybenko, Dynamic load balancing for distributed-memory multiprocessors, J. Parallel Distrib. Comput. 7 (1989), 279– 301..
[7]. J. Y. YU and P. H. J. CHONG,‖A Survey of Clustering Schemes for multiprocessor.
[8]. R. k . das,‖ IEEE Communications Surveys and Tutorials, First Quarter 2005,Vol. 7, No. 1, pp. 32-48.