2012 International Conference on Education Technology and Computer (ICETC2012)
IPCSIT vol.43 (2012) © (2012) IACSIT Press, Singapore

The Design and Implementation of MPI Master-Slave Parallel Genetic Algorithm

Shuping LIU [1+] and Yanliu CHENG [2]

[1] North China Electric Power University，HeBei, Baoding, China
[2] North China Electric Power University,HeBei, Baoding, China

Abstract. In this paper, the MPI master-slave parallel genetic algorithm is implemented by analyzing the basic genetic algorithm and parallel MPI program, and building a Linux cluster. This algorithm is used for the test of maximum value problems (Rosen brocks function) .And we acquire the factors influencing the master-slave parallel genetic algorithm by deriving from the analysis of test data. The experimental data show that the balanced hardware configuration and software design optimization can improve the performance of system in the complexity of the computing environment using the master-slave parallel genetic algorithms.

Keywords: MPI; Cluster; Parallel Genetic Algorithm

1. Introduction

Genetic algorithm,widely used, is based on natural selection and the genetic algorithm search process.Besides the robustness, it also has other properties. The Algorithm proposed by the United States, Professor Holland in 1975 is successfully applied to various engineering problems[1] .

Parallel genetic algorithm is one of the development of genetic algorithms, it is very important to the new generation of intelligent computer.Genetic algorithm have a high degree of parallelism in operation.Many researchers are searching the strategy to implement the genetic algorithms on the parallel machine[2].With the rapid development of the massively parallel computers ,Parallel genetic algorithm has become a hot research spot . The combination of the genetic algorithms and parallel computer, which combines the high-speed of parallel machine and the parallelism of the genetic algorithm , facilitates the research and development of the parallel genetic algorithm[3].

In this article, we have finished the following jobs, namely, the building of the hardware platform, the designing of software, analyzing and testing data, and we also come to the conclusion that the performance computing of the MPI master-slave parallel genetic algorithm can be enhanced through hardware and software.

2.The Parallel Computing and MPI Programs

The parallel computing[4] is following. In the parallel computer, an application is divided into multiple sub-tasks, these tasks are assigned to different processors, and then each processor cooperates with each other and parallel completes the tasks. So as to speed up solving the task or increase the size of the application of the purpose of solving. Simplify speaking, it can be seen as supercomputing in the parallel computer or distributed computer and other high performance computer system. The parallel computer system at least includes two or more the processors .They are connected through the internet for data communication to achieve parallel processing of the processors. Parallel processing generally includes centralized and distributed structures. The design of the article is based on distributed memory parallel computing structure. It is shown in Figure 1.

[+] Corresponding author. *E-mail address*: hdlsp@163.com.

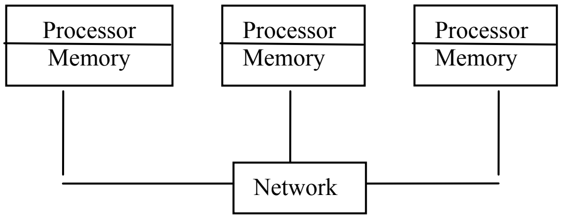

Figure1. the structure of Distributed Cluster

MPI (Massage Passing Interface) is developed by the MPI Forum. It supports the Fortran and c programming. It is a standard message passing function library specification. And MPI has become the representative of a programming model and the de facto standard. In this kind of parallel programming, each parallel process has its own independent address space, and they can not access each other, and the process must be accessed by explicit message passing. It is the main programming of massively parallel processor (MPP) and clusters (Cluster).

MPI has the following characteristics.

- Message passes parallel programming.
- Its parallel computing size is large, which is particularly suitable for large-scale scalable parallel algorithm.

3.The Analysis and Design of Parallel Genetic Algorithm

3.1. The Analysis of Genetic Algorithm

Generally speaking, the object of Genetic algorithms is a group composed of multiple individuals. Those multiple groups can be seen as divided by a large group, and then these small groups are assigned to different processors in a certain way, and they select, crossover, mutation independently, if necessary, and then the processors communicate with each other.

In order to implement parallel computing, we need to decompose the individual data and give certain data for each processor allocation. That can make it possible to handle the task in time and space in parallel, shorten the time needed for the serial operation. There are two methods of general data decomposition: the cross decomposition and block decomposition[5]. Cross decomposition make the load of the processd imbalance, so the paper adopts the block decomposition.

The decomposition is as follows:

Supposing there are n elements and p processor nodes. The distribution of each node's resources is followi ng: firstly, the remaining number of elements after the average allocation is r(r=n mod p).
(1) Supposing r=0, so each processor should get n/p elements;
(2) Supposing r>0,so each r's processors get n/ p distributor-s,and the(p-r)
processor each processor’s distributor is n/p,in this way of calculating the(ith) element of process co-ntrol is i* n / p+min(i,r), the last element of process contr-ol is (i+1)*n/p+min(i+1,r-1).In this way ,it not only can improve the balance of system load capacity but also enhance portability of the system.

3.2. Master-Slave Parallel Computing

1) Master-Slave resource allocation

Master processor M: select one node 141 or node143, and Initialize all the information of the group. Copy all the initialized group to the other slave processor (include itself).And also accept the information sent by the slave processor. Deal with the data and distribute the results to the slave processors. Master processors control the transfer and optimization of the information of the population. The slave processors are responsible for the specific evolution of the subgroup.

Slave processor S: node141and node143 are responsible for the evaluation, selection, crossover and mutation process of each individual among the node group. Select and record the information of the optimal individual among the group, deal with the results of the evolution.

2) Subgroup evolves synchronous parallel for N1 times

When the computer is parallel computing, the time spent in the communication between the computers is a great expenditure. Therefore, in the evolutionary process of the subgroup, Coarse-grained parallel algorithm is used to minimize the number of their communication, and the parallel processing of computing systems will be more efficient

In the article, after each subgroup has evolved for N1>10 times, it must communicate with the primary process. According to the characteristics of the genetic algorithm, Master-slave process uses synchronous parallel algorithm, that is, after the subgroup accomplishes the evolution, it must keep concurrent evolution to this location to wait for other process. The Slave process and the master process communicate in the way of collection, after the information is processed by the master process, and then the corresponding information is assigned to the corresponding process to achieve the evolution of the group.

3.3. The Design of Parallel Genetic Algorithm

The design process of the algorithm is shown in Figure 2.

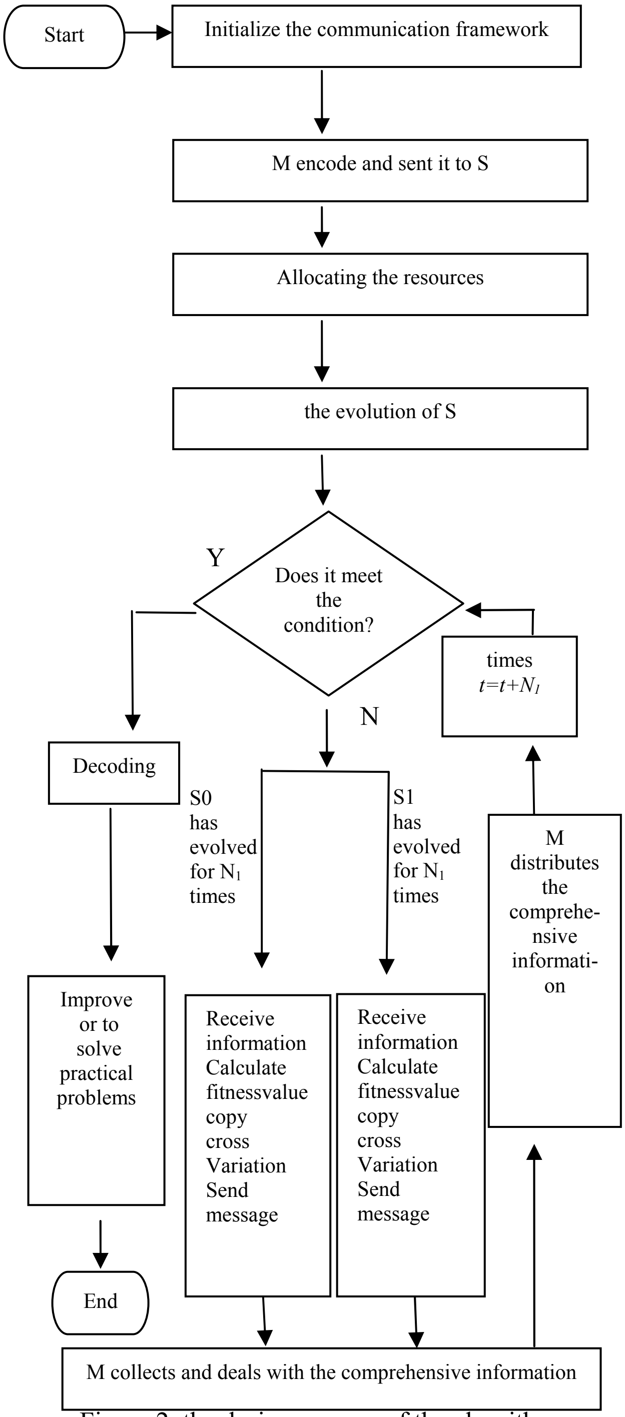
Start
Initialize the communication framework
M encode and sent it to S
Allocating the resources
the evolution of S
Y
Does it meet the condition?
N
times $t=t+N_1$
Decoding
S0 has evolved for N_1 times
S1 has evolved for N_1 times
M distributes the comprehe-nsive informati-on
Improve or to solve practical problems
Receive information Calculate fitnessvalue copy cross Variation Send message
Receive information Calculate fitnessvalue copy cross Variation Send message
End
M collects and deals with the comprehensive information

Figure 2 .the design process of the algorithm

(Note: In the figure,M refers to the master processor, S refers to the slave processor , t refers to the times of the evolutionary of the processor, N1 refers to the times of the concurrent evolution of the subgroup.)

4.Using the Greatest Value to Realize the Algorithm

4.1. Questions

The calculation of the global maximum of Rosenbrock function max f(x1,x2)=100*(x1-x2)+ (1-x2) (1) -2.048 . xi . 2.048 (xi=1,2)

4.2. Construction Process

1) Encoding, Decoding

Designing the evaluation function of genetic algorithm[1] .Each individual uses chromosome encoding method. The chromosome is composed of x1andx2, x1 and x2 are both 10-bit binary. Decoding formula is

$$xi=4.096* (yi / 1023)-2.048 \quad (i = 1,2) \quad . \qquad (2)$$

2) Determine the operating parameters:

The size of the group: M=100
Termination of Algebra: T=300
Crossover probability: pc=0.6
Mutation probability: pm=0.001

3) The information of Data structure

Individual Information:

```
struck individual
{char chrom[CHROMLENGTH+1];
double value;
double filess;}
```

The information between the master process and the slave process:

```
struct populationsinfo
{int best_index;
int worst_index;
struct individual bestindividual;
struct individual worstindividual;
struct individual currentbest;}
```

4) Test the execution time

Firstly, master processes create the nodes, and then they parallelly take the genetic algorithm operations. The end of the master process marks that the communication is completed. The formula is: t_{all} - $t_{tranfaction} + t_{evolution}$, this test time includes the total running time and evolutionary time. Test method is that marking the current time at the right location, and then using (endevaluertime-startevaluertime) to cumulatively record the total execution time, using (endgathertime-startgathertime) cumulatively record total running time.

4.3. MPI code

The characteristic of MPI parallel program execution is that perform the same procedure at the same time on each machine, therefore, in order to realize the parallel processing and communications between the nodes, that is, the dynamic logic of resources must be clear. The storage location of resource allocation function-Assign Resources (myid, numprocs) plays a key role here. When each process receives a group message, we call this function， and use resources block decomposition method to calculate startpopuindex, endpopuindex according to the process of passing the current number and the number of processes, and then get the individual range which the current process will deal with.

Due to the space constraints, only list this framework of the code, as follows:

```
Start popuindex =0;
```

```
    endpopuindex =PopSize-1;
    MPI_Init(&argc,&argv);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    InitMasterAndSlaves();
    ReadFile();
    TransportingmessageBetweenMSInfo();//master process replicate the information to the  slave process
    AssignResources(myid,numprocs);
    EvaluatePopulations();//Evaluation group
    while(mastergeneration <MaxGeneration)
    {mastergeneration+=N;
    MPI_Barrier(MPI_COMM_WORLD); TheSlaveEvalue(myid); SOperateMessageFromEvaluateInfo();
    SGatherMessageFromSlavesInfo();
    MPI_Barrier(MPI_COMM_WORLD);
    if(myid == 0)//the master process do some related processing
    {MOperateMessageFromSlavesInfo(numprocs);// The integrated treatment of all the slave process
information MCatterMessageToSlavesInfo();}
    ShowResult();//Print all information
    WriteFile();
    return 0; }
```

5.Experimental Results and Analysis

5.1. Test group and Drawing

After testing a large amount of data, the results of the final value are obtained. In (-2.048, 2.048), the maximum of the Rosenbrock function is near the point 3905.9262268416 and the point 3898.7342268416.

In table I, the measurement data is divided into A, B, C, and D four groups. Group A is a single-node 141; Group B is a single-node 143; Group C is a two-node, and the node 143 is the master node ； Group D is a two-node, and the node 141 is the master node 。The total time spent in testing the master node is t_{all} (the total program execution time), the sum of the time spent in the evolution of the program is $t_{evolution}$ (the sum of evolutionary time). Each group must be tested for 40 times. We get two kinds of time-consuming, as follows in figure 3 and figure 4.

（Note: The number of the decimal digits of the test data calculated by computer is 6, and the number of the decimal digits of the data shown in the vertical axis is 3 in the process of the drawing and programming. These do not affect our analysis of performance.）

TableI.THE GROUPS OF THE TEST

The group	The number of the processors	Master process	Slave process
A	1	141	nothing
B	1	143	nothing
C	1	143	143 ， 141
D	2	141	141 ， 143

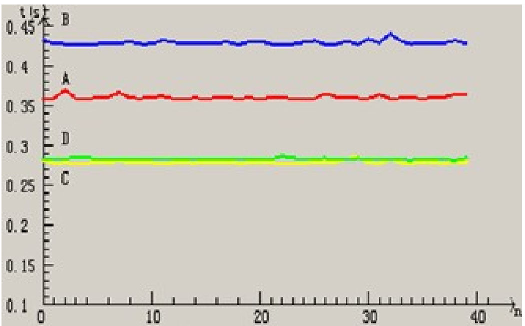

Figure 3. The sum of the time spent in testing the master node

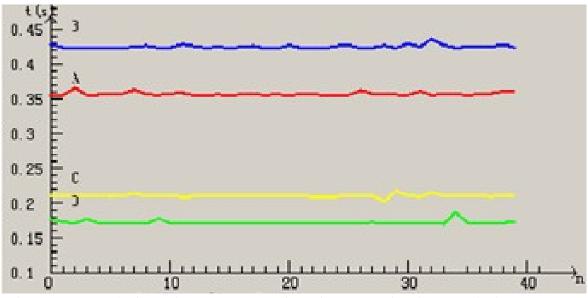

Figure 4 .The sum of the time spent in the evolution of the master node

5.2. Experimental Analysis

1) We get the data from the figures, Units(s):

Figure 3:

tA_{all} =0.360 tB_{all} =0.425 tC_{all} =0.280 tD_{all} =0.280

Figure 4:

$tA_{evolution}$ =0.355 $tB_{evolution}$ =0.425 $tC_{evolution}$ =0.210 $tD_{evolution}$ =0.175

2) Analyze these data and Get the following conclusion

a)Comparing group A、B with group C、D，we find that the computing capability of two-node is more powerful than the single-node.

b)We know that the performance of 141 is better than 143.In the group C, the communication time account for 25% of the total execution time(（tC_{all} - $tC_{evolution}$ ） /tC_{all} *100%=25%).In the group D, the communication time account for 37.5% of the total execution time(（tC_{all} - $tC_{evolution}$ ） /tC_{all} *100%=37.5%).

So the hardware structure can affect the ability of the algorithm computing. We can improve the performance through the software and hardware. First, when parallel computing is implementing, it requires multi-node parallel and orderly transfer the data, and then make the appropriate treatment. And it also requires a computer to keep the same hardware configuration. The task of each processor can be done trough load balance, which can decrease the Latency time of the communication. Secondly, at the software level, increasing the granularity of parallel computing can reduce the number of the communication .At the same time the specific application, consider the relationship between the granularity and dealing with the evolution of the node, test the critical point of the algorithm, solve the bottleneck of the granularity, and make the combination of the granularity and the capability of dealing with the information reach the best state.

6.Conclusion

In this paper, we have designed MPI master-slave parallel genetic algorithm on the basis of genetic algorithm and MPI programming, and use the maximum problem to test the algorithm. The test results show that the algorithm has better operational ability. And the analysis results of the test data show that balanced hardware configuration and software design optimization can improve the performance of the system.

7. Reference

[1] D E Goldberg Genetic and evolutionary algorithms come of age 1994(03).

[2] ZHAO yipeng,MENG lei,PENG chengjing."The Overview of Genetic Algorithms and the development "[J], Hei long jiang Science and Technology Information2010(13):79-80.

[3] W Rivers Scalable parallel genetic algorithms 2001(02).

[4] Du Zhihui ."High Performance Computing parallel programming techniques "[M],Tsinghua University Press, Beijing, 2001

[5] Timothy G.Mattson. Parallel programming model.[M] ,Tsinghua University Press, Beijing, 2005

[6] ZHOU Ming, SUN Shudong.”Principle and Application of genetic algorithms "[M], NationalDefense University Press , Beijing , 1999:27-28

[7] Michael J. Quinn , Parallel Programming in C with MPI and OpenMP [M].Tsinghua University Press, 2004.

[8] Qu gang, Deng jianqing, and Han yulu . "Study on Linux cluster technology"[J],Microcomputer Applications, 2005:100-107.