World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:6, No:1, 2012

Designing and Implementing a Novel Scheduler for Multiprocessor System using Genetic Algorithm

Iman Zangeneh, Mostafa Moradi, Mazyar Baranpouyan

***Abstract*—**System is using multiple processors for computing and information processing, is increasing rapidly speed operation of these systems compared with single processor systems, very significant impact on system performance is increased .important differences to yield a single multi-processor cpu, the scheduling policies, to reduce the implementation time of all processes. Notwithstanding the famous algorithms such as SPT, LPT, LSPT and RLPT for scheduling and there, but none led to the answer are not optimal.In this paper scheduling using genetic algorithms and innovative way to finish the whole process faster that we do and the result compared with three algorithms we mentioned.

***Keywords*—**Multiprocessor system; genetic algorithms; time implementation process

I. Introduction

PAY lower prices and sizes , along with their rapidly increasing use of multiple processors on a board to facilitate multiple processors and systems have made it[1].Many advantages that these systems compared to single processing speed, system reliability and high tolerance of failure, the increasing spread of these systems has been the case on large amounts of data(the usually) work.Also, items that require the above process are rapid and timely remedy to use these systems usually sees.Second and third section of this paper, basic concepts and review if the issue offer, fourth to fully describe the proposed algorithm can proceed. In the fifth section the results evaluated and recommendations are provided.

II. Topics

Pay Given the important of today, multiprocessor systems, and application have been heavy and fast, study research necessary for their optimization are well marked.Increase operational efficiency for multiple processors , in performed , can increased operational efficiency of individual processors can be done, that is not our subject and single processor systems is to followed , and one things that need to increase overall system performance to be conducted[2].

Iman Zangeneh is with Islamic Azad University of BaghmalekBranch,Iran,I_zangeneh@yahoo.com
Mostafa Moradi is with Islamic Azad University of BaghmalekBranch,Iran,mos_moradi@yahoo.com
Mazyar Baranpouyan is with Isfahan University. Isfahan, Iran, pouyan@gmail.com

These important topics that single processor is different, the issue is timing.In a multi-processor system with arbitrary number of arbitrary number of processors want inseparability process we implemented simultaneously, in a way that the TFT process for this system may be lowest as even as far as possible be close to the idea value. Issues should be resolved with genetic algorithms and resulting answer(TFT) with answers from scheduling algorithms SPT,LPT and RLPT be compared[3-6].note that the zero moment all process are ready for schedule end execution during the implementation schedule and run any other process cannot be added to the collection process.

III. Review Of The Issuse

In a multi-processor system, what is important in the implementation process, end the whole time (TFT) is done so that when a bunch of processes to run on a multi-processor, a queue can be placed, what is important and system performance criteria should be considered, is that the allocation of processes to processor, all processor, at what time are free [7].When the last running process reaches completion, the end time is the system processing process; and we then can the process for the next batch processing system, we load.Therefore, to increase system efficiency policy should be followed to reduce the TFT [8-9].We present the scheduling algorithm by itself on this issue will focus.

IV. Describe The Proposed Algorithm

Algorithm procedure before your intended action on the input we need the input of different scenarios are investigated.

Total process may run ready to pay more or even less in certain cases is. Therefore, since the TFT is to reduce the time, the first is determined whether the number of processes ready to run the number of processor is less or more?

Processes and the number of processor since N and M, each process to a processor, we display.

If N is smaller than or equal to M each process to processor, we allocate.

It is evident that in this case the TFT performance will be the longest process, and the algorithm will need apply.

If N is greater than m, their algorithm on the desired input will be applied.

Since in our issue of what is important to find the order to execute different processes on different processor, so each chromosome with each gene type as a prototype of the design

problem is the answer , should respect the point was inside be.

Therefore, we designed chromosome, a one dimensional array with N house, which also houses the content of each issue is a CPU that process marked with array index must be run on it. Array called A hypothetical example, if we have A[6]=2 means that the process should NO 6 on 2 processors run.Chromosome now that we have solved the problem should be designed to produce the first generation of chromosomes pay.In this implementation be number of chromosomes per generation number k we consider the number p as they both operate the two together will do the crossover and the possibility of residual k-p chromosome mutation will be.

Optimal k and p values to aid analysis and sensitivity tests on different test data sets we obtained (Table I).

TABLE I
PARAMETERS USED IN THE PROPOSED METHOD OF COMPUTER SIMULATION

Number of chromosomes in early generations (K)	30
Chromosome number for the Crossover of practice (P)	28
Total generation production (G)	30

It is evident that the system quickly with programing running on it and the amount of memory dedicated to the program to the program can be run while the number of applications processes are running very high, such as Web server, the number the number of chromosomes are most probable solution in every generation have.Now that the number of chromosomes per generation, we specify, turns filling them. The first generation of chromosomes to fill in the following cases is considered.Chromosomes, one-dimensional array of N elements (index) is the values of each cell (each gene) of M to M number of host processors, will change.Implementation process for all states, when the number of processes the CPU number is higher, best executive is that all processors are busy.Since the scheduling multi-processor systems service provider, such as web servers find important. In practice, each of these servers also provide a special service pay. For example, web server application to multi-processor system text answers to a server or other graphic requests answered. Therefore, the implementation time applications (processes) provided on each system at a time close to the time limits will be.We perform the above process to each set of N/M subsets, we partition (process busy condition). And the number of different elements of each partition (except possibly one of them) against the [N/M] lies in (meet requirement 3 above).Expression means is that such a system with three processors and 10 processes, each process 10 sets the following three sets in the partition, we count them all except one [10/3]=3 there is a process. 3 shows that on the third processor and process on the one remaining here four process will run.We each chromosome with the above considerations we produce. Random order for each chromosome be filled with another chromosome in the same generation is different, every house of the chromosome array, we randomly filled.How to allocate processors so that if, after undergoing the process of being determined by the function Random, to determine the processor number assigned to the selective process, called the Random function to numeric 1 to M to produce.These processors are allocated to process. For the next processor in the array in unknown future processors will be set up processes to make partition characteristics are preserved.According to the above operations for k-1 in other chromosomes, we repeated the first generation. Obviously considering the use of the function to fill array elements each, the content of each chromosome with other chromosomes will be different.Now that we have produced the first generation, can the operations necessary to produce the next generation of chromosomes, we attempted. Here are our two Crossover and the Mutation operator we have used. Also good for us that the possible solution in each generation, select the ultimate answer is preserved, we have also used the elite setting.So, after production and before the first generation change in the chromosomes using Crossover and Mutation, to select the elite chromosome.And stop the algorithm for selecting chromosomes to the selecting chromosomes according to the following three things are essential:Fitness of each chromosome: the maximum run time to consider the various processors; As such, when filling the house with the amount of chromosome No. processors, each processor processing time variability, we defined.Each processor to each process (element) is spent, the amount of time to implement the process variable corresponding to the CPU processing time is added. Finally, after filling chromosome genes in processing time between different processor as the highest fitness criteria we consider. In fact, we see that TFT is the fitness criterion.Select elite, since the time of implementation processes can be different values of relative criterion we use. For example, if the time difference as the fitness criteria desired ideal time for TFT, is less than 10 percent, the chromosome can be as elite in a secondary storage array to be displayed at the end after the implementation of their algorithm to compare be used to select the best answer. In addition, if the generation of chromosomal condition can be found as the best chromosome of each generation would save the elite.Stop us if your genetic algorithm produced a fixed number of generations of generations to generations such g we have mentioned. Moreover, this selective condition that if we had also added before the last generation, with different fitness criteria ideal time of less than one percent, for example, the algorithm accepts the end. We plan to increase the number of restrictions, as fully as fast implementation allows our system does not exist. And is also on the percentage difference between the ideal time to stop condition.Now to explain Crossover and Mutation operators in producing the intended chromosomes. Since the number of chromosomes per generation, k is a number, we k-1 on their mutually act on Crossover and one of them we do practice jumps.Exchange of one type of point and point to a point desired exchange of chromosomes as the middle point [N/M] we consider.Action here means that the processors Crossover a series of specific processes with other processes to specific processors are

crossover; and act on a gene mutation, i.e. a process that specific custom processor to be replaced.Since random chromosomes are the first generation of full and productive breeding generations chromosomes in other locations will be randomly selected stop exchange a few points to a certain point, such as median chromosome exchange, no restrictions on production. Another useful compound chromosome for next generation does not establish.All operations, said, up to g times using a loop g-1 each (except for filling the initial first generation) is performed.In each selected elite are stored, and Crossover and Mutation operation is performed and are transferred to the next generation.

Finally, the final step, TFT elite stored, compared and then choose which chromosome has the lowest TFT respectively, the implementation process on pay, the answer will be obtained from the chromosome, and genetic algorithm as a final answer on from program output will be displayed. Obviously, if other condition, stop, i.e. the middle stage of selective differences with the elite of less than one percent of ideal TFT is selected, end the algorithm step were accepted and implementation process of the elite chromosome is calculated.

V. Conclusion

Since other algorithms such as STP, LPT and RLPT algorithms, classic and famous are the results of implementing the proposed method itself on the set of data varied taken from our test results with algorithms SPT, LPT and RLPT were compared to in Table II and III and IV we see.

TABLE II
RESULT ON CATEGORY A PROCESS

Finally the whole time	Method
37	LPT
46	RLPT
60	SPT
35	Propossed (GA bassed)

TABLE III
RESULT ON CATEGORY B PROCESS

Finally the whole time	Method
88	LPT
75	RLPT
79	SPT
71	Propossed (GA bassed)

TABLE IV
RESULT ON CATEGORY C PROCESS

Finally the whole time	Method
95	LPT
90	RLPT
98	SPT
88	Propossed (GA bassed)

Evolutionary algorithms and methods compound other initiative like PSO, TS; SA to create new solutions could improve the method useful. Ideas useful means of these methods to create new solutions in the proposed method should be used.

References

[1] Andrew S. Tanenbaum, Maatrten va Steen, Distributed systems, Fifth Edition, Prentice Hall, 2002.
[2] Hou E, Ansari N, Ren H, A generic algorithm for multiprocessor scheduling,IEEE Trans Parallel Distrib Syst 5(2), pp. 113-120, 1994.
[3] Correa R, Ferreia A, Rebreyend P, Scheduling multiprocessor task with genetic algorithm, IEEE Trans Comput 30(3), pp. 207-214, 1999.
[4] Zomaya A, ward C, Macey B, Geneticscheduling for parallel processor systems comparative studies and performance issues. IEEE Trans Parallel distrib Syst 10(8), pp. 795-812, 1999.
[5] Wu A, Yu H, Jin S, Lin K, Schiavone G, An incremental genetic algorithm approach to multiprocessor scheduling, IEEE Trans Parallel Distrib Syst 15(9), pp. 824-834, 2004
[6] Yoa W, You J, Li B, Main seguences genetic algorithm for multiprocessor systems using task duplication, Microprocessor Microsyst 28, pp. 85-94, 2004.
[7] Ceyda O, Ercan M, A genetic algorithm for multilayer multiprocessor task scheduling, In: TENCON 2004, IEEE region 10 conference, vol 2, pp. 68-170, 2004.
[8] Cheng S, Huang Y, Scheduling multi-processor task with resource and timing constaints using genetic algorithm. In: 2003 IEEE international symposium on computational intelligence in robotics and automation, vol 2, pp. 624-629, 2003.
[9] Rahmani A.M, Vahedi M.A, A novel task Scheduling in Multiprocessor System with Genetic Algorithm by using Elitism stepping method, INFOCOMP, Journal of Compuetr Science, vol 7, number 2, 2008.