© UNIVERSAL RESEARCH REPORTS (URR) | REFEREED | PEER REVIEWED
ISSN : 2348 - 5612 | Volume : 05 , Issue : 05 | April – June 2018
http://www.uresearchr.com

Implementation on performance of parallel computing by introducing upgraded gang scheduling algorithm

Pooja, Dr.Raghuvinder

Research Scholar, Department of CSA, CDLUniversity,Sirsa, pbhambhu23@gmail.com

A P, Department of CSA, CDLUniversity,Sirsa. raghuvinder.bhardwaj@gmail.com

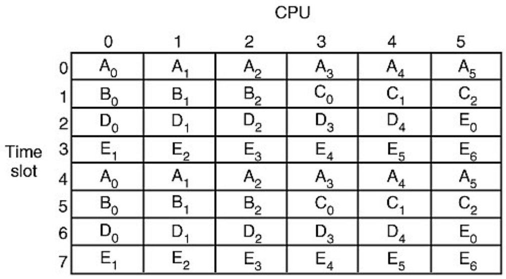
Abstract:- *In computers, parallel processing is processing of program instructions by dividing them among multiple processors with objective of running a program in less time. In this paper we have explain to speed up processing by introducing concept of cache & ram & customization of existing algorithm to provide additional support. We have study to simulation of enhanced gang scheduling algorithm with additional parameters & also make comparative analysis traditional & proposed Methods.*

ISSN : 2348-5612 © URR
9 770234 856124

Keyword:- Parallel processing, Gang scheduling,

[1] Introduction

Gang scheduling is a scheduling algorithm for parallel systems that schedules related threads or processes to run simultaneously on different processors. Usually these would be threads all belonging to same process, but they may also be from different processes. For example, when processes have a producer-consumer relationship, or when they all come from same MPI program.



Time slot	CPU 0	1	2	3	4	5
0	A_0	A_1	A_2	A_3	A_4	A_5
1	B_0	B_1	B_2	C_0	C_1	C_2
2	D_0	D_1	D_2	D_3	D_4	E_0
3	E_1	E_2	E_3	E_4	E_5	E_6
4	A_0	A_1	A_2	A_3	A_4	A_5
5	B_0	B_1	B_2	C_0	C_1	C_2
6	D_0	D_1	D_2	D_3	D_4	E_0
7	E_1	E_2	E_3	E_4	E_5	E_6

Fig 1 Gang scheduling

Gang scheduling is used so that if two or more threads or processes communicate with each other, they would all be ready to communicate at same time. If they were not gang-scheduled, then one could wait to send or receive a message to another while it is sleeping, & vice versa. When processors are over-subscribed & gang scheduling is not used within a group of processes or threads which communicate with each other, it can lead to situations where each communication event suffers overhead of a context switch.

[2] Multiprocessor Scheduling

In computer architecture, multithreading is ability of a central processing unit or a single core in a multi-core processor to execute multiple processes or In uniprocessor systems, shortest job first is a well-known algorithm for batch **scheduling**.

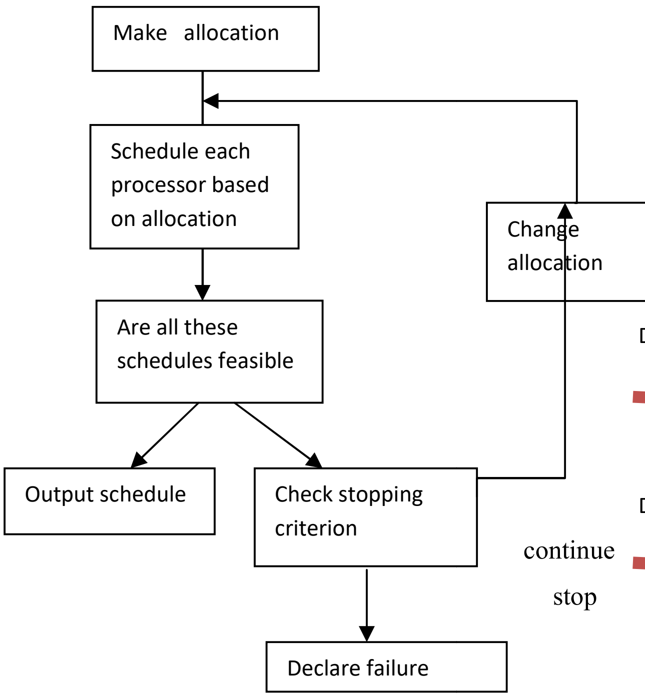


Fig 2 Computer architecture

The analogous algorithm for a **multiprocessor** is to choose process needing smallest number of CPU cycles, that is process whose CPU-count X run-time is smallest of candidates concurrently, appropriately supported by operating system. This approach differs from multiprocessing, as with multithreading processes & threads share resources of a single or multiple cores: computing units, CPU caches, & translation look side buffer.

Classification of multiprocessor systems

Loosely coupled or distributed multiprocessor or cluster: each processor has its own main memory & I/O channels.

Functionally specialized processors: an example is an I/O processor, controlled by a master processor

Tightly coupled multiprocessing: processors share a common main memory, controlled by operating system

Parallel computing

In parallel computing multiple processors have to be scheduled, & it needs to manage resources for all processors.

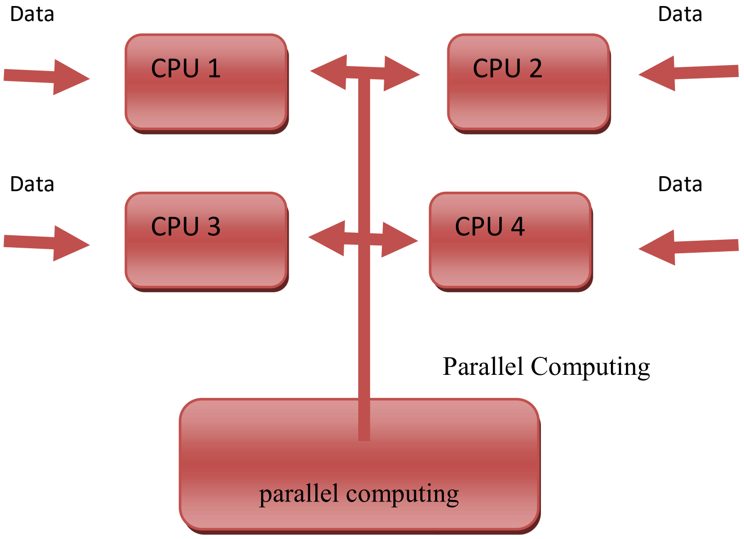


Fig 3 Parallel Computing

In managing resources for multiple processors, it should be ensured that, there should not be any overlapping of resources, & it should not give any conflicting results. So scheduling in multiprocessors is more difficult than scheduling in a single processor unit.

[3] METHODOLOGY

In computing, **scheduling** is method by which work specified by some means is assigned to resources that complete work. Work might be virtual computation elements such as threads, processes or data flows that are within turn scheduled onto hardware resources such as processors, network links or expansion cards. A scheduler is what carries out scheduling activity. Schedulers are a lot implemented so they stay all computer capital busy allow more than one users to share scheme resources effectively, or to achieve a predefined quality of service.

Dynamic multithreaded programming

As multiprocessor systems have been increasingly available, interest had been grown in parallel programming. **Multithreaded programming** involves a programming paradigm in which a single program is broken into more than one **threads** of control which interact to solve a single problem.

[4] RESULT & DISCUSSION

In this paper we have discussed use of multithreaded based processor. code of gang scheduling has been take as base so that calculation could be made fast. Here requirement & availability have been specified according to traditional model of gang scheduler. After that implementation of proposed gang scheduler that is supporting coprocessor, ram & cache have been discussed here.

Simulation of central processing unit performance

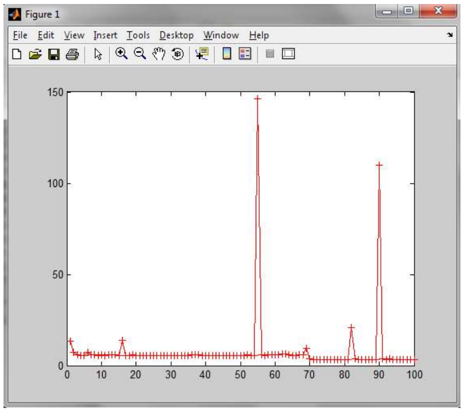


Fig 4 Plotting chart to represent central processing unit performance

Simulation in case of Multiprocessors

Following is diagram plotted using MATLAB simulator. time taken by different processors has been stated in following figure**.**

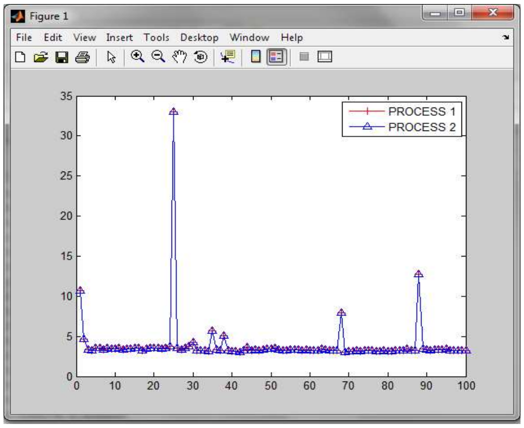


Fig 5 Plotting chart to represent central processing unit performance

Simulation in case of Unicode & Multicore processor

Saving time consumption by multiprocessor in case of Unicode & multicore. Here we have discussed implementation of Unicode & multicore processor.

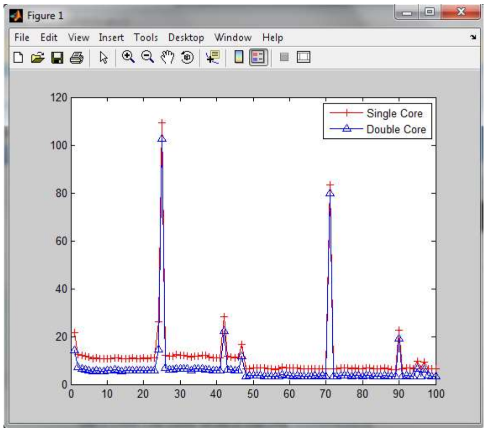
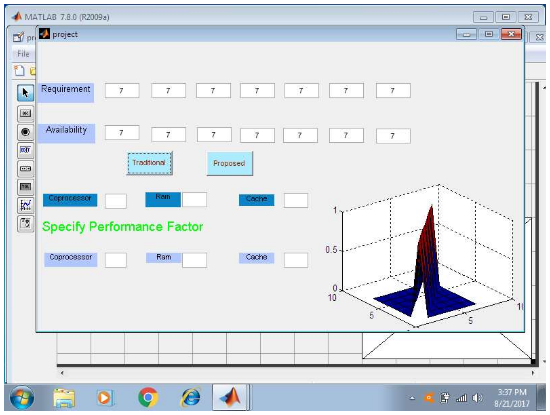


Fig 6 Comparative analysis of central processing unit performance in case of unicore & multicore

The above chart is representing single core & double core by checking time of calculation of arithmetic operation using MATLAB simulator.

The following is Matlab simulation a GUI has been developed. Here requirement & availability has been mentioned for traditional. If user is going to use proposed implementation then he need to coprocessor, ram & cache details.

Following is implementation in case of traditional pattern. histogram is plotted after inserting requirement & availability in traditional gang scheduling algorithm & a matrix is generated



Requirement 7 7 7 7 7 7 7

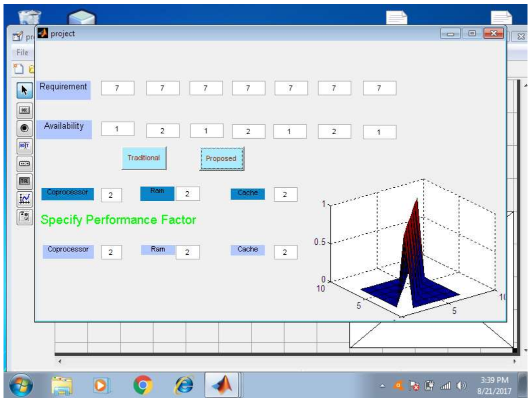
Availability 7 7 7 7 7 7 7

Coprocessor Ram Cache

Coprocessor Ram Cache

Fig 7 Histogram is plotted after inserting requirement & availability in traditional gang scheduling algorithm

Following is implementation in case of proposed pattern. histogram is plotted after inserting requirement & availability in proposed gang scheduling algorithm & a matrix is generated. Here number of coprocessor, ram & cache is specified. Then performance factor of coprocessor, ram & cache is specified.



Requirement 7 7 7 7 7 7 7

Availability 1 2 1 2 1 2 1

Coprocessor 2 Ram 2 Cache 2

Coprocessor 2 Ram 2 Cache 2

Fig 8 Histogram is plotted after inserting requirement & availability in proposed gang scheduling algorithm

This implementation confirms that use of ram, coprocessor & cache with their performance factor could raise performance of traditional work.

[5] Conclusion

In this paper on basis of implementation of research work, conclusion ion has been made & discussed. On other hand, future scope of this research work is also presented.

The multithreading idea had go off to more all range as hard work to further develop teaching level parallelism have puzzled. This allowed concept of throughput computing to re-emerge from more specialized field involves transaction processing even though it is very difficult to further speed up a single thread or single program most computer systems are really multitasking among multiple threads or programs. Thus techniques that improve throughput of all tasks result within overall performance gains.

Conceptually it is similar to cooperative multi-tasking used within real-time operating systems, within which tasks voluntarily give up execution time when they need to wait upon some type of event.

This kind of multithreading is renewed as block supportive or coarse-grained multithreading.

REFERENCE

1. Yeh-Ching Chung wrote on 2006 “Performance Analysis & Applications of A Compile-Time Optimization Approach for List Scheduling Algorithms on Distributed Memory Multiprocessors”
2. Ishfaq Ahmad1 wrote on 2008 “On Parallelizing Multiprocessor Scheduling Problem”
3. Maruf Ahmed wrote on 2009 List Heuristic Scheduling Algorithms for Distributed Memory Systems within Improved Time Complexity
4. Wayne F. Boyer wrote on 2010 “Non-evolutionary algorithm for scheduling dependent tasks within distributed heterogeneous computing environments”
5. A research titled “Chip Multithreading: Opportunities & Challenges” by Lawrence Spracklen on 2012
6. Rajeev Garg, Ali El-Moursy authored research on 2012 “Partitioning Multi-Threaded Processors using Large Number of Threads”
7. Another research paper published by Bart Jacobs on 2014 titled “Verification of Multithreaded Object-Oriented Programs with Invariants”
8. Krishna (2014) their research paper on topic “MODELING MULTITHREADED APPLICATIONS USING PETRI NETS”
9. Sridhar (2015) Comparative Study of Ant Colony Optimization & Gang Scheduling
10. Another research by authors Ivan Voras (2015) titted “Characteristics of multithreading models for high-performance IO driven network applications”
11. Prathmesh Deshmukh 2015 in their research titled “Effective use of Multi-Core Architecture through Multi Threading towards Computation Intensive Signal Processing Applications”
12. Sawati (2016) Enhancing Capability of Gang Scheduling by Integration of Multi Core Processors & Cache IJSRE Volume 4 Issue 8 August 2016
13. Arun Seth(2016) Types of Scheduling Algorithms in Parallel Computing
14. Yongsheng Hao, Guanfeng Liu, Rongtao Hou,Yongsheng Zhu, Junwen Lu, Performance Analysis of Gang Scheduling in a Grid‖, Springer Science, April 2014.
15. Ioannis A, Performance & Cost evaluation of Gang Scheduling in a Cloud Computing System with Job Migrations & Starvation Handling‖, IEEE 2011.
16. Y. Zhangy H. Frankez J. E. Moreirazy, “An Integrated Approach to Parallel Scheduling Using Gang-Scheduling, Backfilling & Migration”, IEEE Transactions on industrial informatics, Vol. 10, pp. 11, November 2013.
17. M. Dorigo & T. St¨utzle, Ant Colony Optimization. MIT Press, 2004.
18. H.Li, & C.Peng Lam, —Software Test Data Generation Using Ant Colony Optimization,‖ In Transactions on

Engineering, Computing & Technology, 2005.

19. P.Zhao, P.Zhao, & X.Zhang, ―New Ant Colony Optimization for Knapsack Problem,‖ in Proceedings of 7th International Conference on Computer-Aided Industrial Design & Conceptual Design, Nov 2006, pp 1-3.
20. H. LI , C. Peng LAM, ―An Ant Colony Optimization Approach to Test Sequence Generation for Statebased Software Testing,‖ In Proceedings of Fifth International Conference on Quality Software, September 2005, pp.255-264.
21. X. Chen, Q. Gu, A. Li, D. Chen, ‖Building Prioritized Pairwise Interaction Test Suites with Ant Colony Optimization,‖ In Proceedings of Ninth International Conference on Quality Software [QSIC], 2009, pp. 347-352.