International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Selective Scheduling Based on Number of Processor Cores for Parallel Processing

Ravinder Jeet[1], Upasna Garg[2]

[1]Guru Kashi University, Punjab 151302, India

[2]Assistant Professor, Guru Kashi University, Punjab 151302, India

Abstract: ***Scheduling is essential for the proper functioning of multi-core processors for parallel processing. Scheduling of tasks onto multi-core processors is an interesting problem that is well defined and documented in the literature. Multi-core processor systems are increasingly commonplace, and have found their way into desktop machines, laptops, and even mobile devices. The rise of the multi-core processor, in which multiple CPU cores are packed onto a single chip, is the source of this proliferation. These chips have become popular as computer architects have had a difficult time making a single CPU much faster without using too much power.With the advent of multi core processors, there is a need to schedule multiple tasks on multiple cores. In this paper we are proposing a new scheduler algorithm which schedules the multiple tasks on multiple cores of a single chip processor. One task is assign to a single core and the one is on another core processor within a single chip processor for parallel processing. The main goal of this work is to allow jobs to scale well with the number of cores. The proposed algorithm will allow each processor core to execute at least one task on single instance of time. This means if the number of processor cores is four then four tasks will be execute simultaneously on all four cores.***

Keywords: Selective Scheduling, Multi-core System, Parallel Processing, Number of Processors

1. Introduction

Scheduling of tasks is essential for every system. Scheduler is one of the most important parts of an Operating system. Without scheduler, tasks may not execute in that order which a user or operating system itself want. Scheduling of tasks on single core processor is much easy by choosing existing scheduler programs like Round-Robin, Priority based, First Come First Serve bases, Shortest job First etc. Multi-core processors have two or more processing elements or cores on a single chip. These cores could be of similar architecture (Synchronous Multi-core Processors, SMPs) or of different architecture (Asynchronous Multi-core Processors, AMPs). All the cores necessarily use shared memory architecture. Multi-core processors have existed previously in the form of MPSoC (Multi-Processor System on Chip) but they were limited to a segment of applications such as networking [13]. The easy availability of multi-core has forced software programmers to change the way they think and write their applications [13]. Unfortunately, the applications written so far are sequential in nature. Multi-core processors do not automatically provide performance improvements to applications the way faster processors did. Instead applications must be redesigned to increase their parallelism. Similarly, CPU schedulers must be redesigned to maximize the performance of this new application parallelism. CPU scheduling policy (and in a large part mechanism) is unimportant to a serial application running on its own machine [2]. An important part of parallel processing in multi-core reconfigurable systems is to allocate tasks to processors to achieve the best performance. The objectives of task scheduling algorithms are to maximize system throughput by assigning a task to a proper processor, maximize resource utilization, and minimize execution time.

In a single-processor system, only one process can run at a time; any others must wait until the CPU is free and can be rescheduled. The objective of multi-tasking is to have some process running at all times, to maximize CPU utilization. Scheduling is a fundamental operating-system function. Almost all computer resources are scheduled before use. The CPU is, of course, one of the primary computer resources. Thus, its scheduling is central to operating-system design. CPU scheduling determines which processes run when there are multiple run-able processes

2. Scheduling Parameters

- **CPU Utilization**: It is the average fraction of time, during which the processor is busy.
- **Throughput**: It refers to the amount of work completed in a unit of time. The number of processes the system can execute in a period of time. The higher the number, the more work is done by the system.
- **Waiting Time:** The average period of time a process spends waiting. Waiting time may be expressed as turnaround time less the actual execution time.
- **Turnaround time**: The interval from the time of submission of a process to the time of completion is the turnaround time.
- **Response time:** Response time is the time from submission of a request until the first response is produced.
- **Priority:** give preferential treatment to processes with higher priorities.
- **Fairness:** Avoid the process from starvation. All the processes must be given equal opportunity to execute.

Multi-core Processor Architecture

In Multi-core processor architecture two or more chips are embedded in to single processor chip that are called cores. A single core is a single complete microprocessor in itself which can perform all operations as a full microprocessor. Memory is shared by all available cores in single chip. Most

of Multi-core processor have single shared cache storage which leads to mutual accesses in processing.

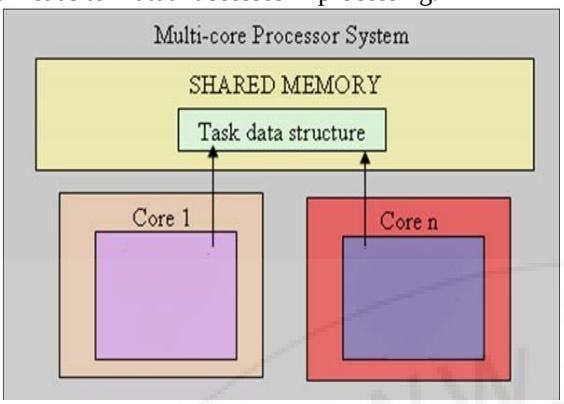

Figure 1: Block diagram of Multi-core Processor

The efficiency of a parallel computing system is commonly measured by completion time, speedup, or throughput, which in turn reflect the quality of the scheduler. Many algorithms have already been developed which provide effective solutions. Most of these methods, however, can solve only limited classes of the scheduling problem [1]. The scheduling problem is known to be NP-complete for the general case and even for many restricted cases [15].

3. Related Work

Many researchers have defined various scheduling algorithms for multi-core processor systems. Vinay G. Vaidya's *"Dynamic Scheduler for Multi-core Systems"* scheduler program runs on each processor core to schedule tasks on each core individually. But the limitation is that a main program is needed to control all cores [13]. Some other multi-core schedulers are uses one scheduling algorithm like "*SUBCONTRARY MEAN DYNAMIC ROUND ROBIN (SMDRR) SCHEDULING ALGORITHM"* on each core [21]. SMDRR scheduler runs on each core but there is no main scheduler to control all tasks which decides which tasks execute on which core.

4. Problem Formulation

Various schedulers are available for scheduling jobs on multi-core processors. Many scheduling algorithms have been proposed in the past. With the advent of multi core processors, there is a need to schedule multiple tasks on multiple cores. The scheduling algorithm needs to utilize all the available cores efficiently. The multi-core processors may be SMPs (Symmetric multi-processor) or AMPs (Asymmetric multiple-processors) with shared memory architecture [13]. This work propose a dynamic scheduling algorithm called 'Selective Scheduling' in which the scheduler resides on all cores of a multi-core processor and accesses a shared Task Data Structure (TDS) to pick up ready-to-execute tasks. This method is unique in the sense that the processor has the capability of picking up tasks whenever it is idle.

In MS windows by default only one processor core is active if the hardware has multiple cores in its architecture.

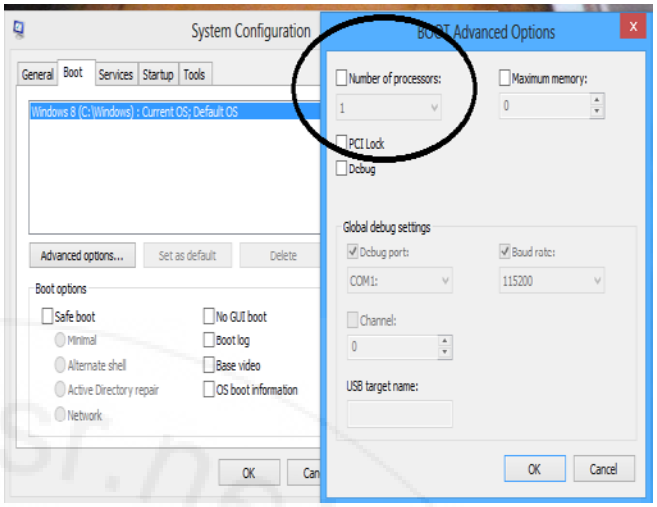

Figure 2: windows 8 default single core setting

When open this setting by default active number of processor core is one. This can be changed by checking the option of number of processor. Windows 8 uses Round-Robin scheduling [22], which is not enough for multi-core processor system architecture. In this work we proposes a new strategy which uses existing traditional algorithms like Priority based, First Come First Serve bases, Shortest job First etc.

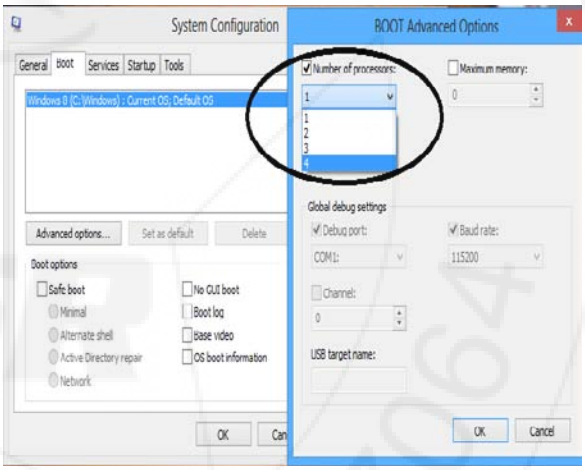

Figure 3: Total number of cores in processor

This setting can be changed by type 'msconfig' in run command window and setting can be seen to set 1 core by default. But the limitation is by activating all the cores there is no difference in the performance of the system, because the scheduling is Round-Robin in windows 8 [22]. Thus to solve this problem 'Selective Scheduling' is proposed in this work. This scheduling algorithm take care of all active cores available in the hardware of the system and assigns jobs to all cores for better performance.

5. Research Methodology

The scheduling technique explores in this work is based on number of cores and can be implemented in the operating system to schedule jobs.

6. Selective Scheduling

Selective scheduling is based number processor cores. Various scheduling algorithms are used in this work such as FCFS, SJF, Round-Robin, Priority and Pre-emptive scheduling. Priority is assigned to each job according to nature of job. In this work scheduling is implemented on four cores processor.In spite multiple core processors are available in modern systems [13]. But the limitation is that at a particular time period one job is executed at one core, at the same time all other cores are in idle state. Therefore CPU utilization is decreased continuously.

So there is a solution of this problem in this work. Scheduling named selective scheduling based on number processor core is introduced in this work.

Assumptions are, at start point CPU is idle and performance is measured from the time when the jobs are available on the system for execution. Then priority is assigned to each job by the nature of the jobs like Run time, Burst time, job is either system or Application process. After assigning priorities each job is assigned to a particular processor core out of four processor core. There are four cores in processor therefore 4 jobs are executed at single piece of time. Each core has its cache memory. Main memory is shared by all four cores and memory is also divided in to four sections based on four cores so that memory access is made fast by each core and collision does not occur between all cores at the time of execution.

Response time is also reduced by this scheduling. When a new job is occurred it is assigned to one processor core for first response and running job on that core is primped at that time. Context switching is used in this condition. At first response priority of new job is compared with priority of existing job. If priority of new job is higher than existing job then new job is assigned to that core and existing job is put in waiting state.

References

[1] Albert Y. Zomaya, Chris Ward, and Ben Macey "Genetic Scheduling for Parallel Processor Systems: Comparative Studies and Performance Issues", IEEE Transactions On Parallel And Distributed Systems, Vol. 10, No. 8, August 1999.
[2] Joseph T.Meehean Doctoral Dissertation at University of Wisconsin–Madison, "Towards Transparent CPU Scheduling" 2011.
[3] Andrew J. Page, Thomas M. Keane, Thomas J. Naughton, "Multi-heuristic dynamic task allocation using genetic algorithms in a heterogeneous distributed system" Journal of Parallel & Distributed Computing 70 (2010) 758_766, 2010.
[4] Raj Mohan Singh and Harsh K Verma," An Analysis of Scheduling Strategies Based on Criticality of Jobs" International Journal of Computing, Communications and Networking, ISSN 2319-2720, Volume 2, No.1, January – March 2013.
[5] Arpaci-dusseau," Multiprocessor Scheduling (Advanced)", 2014.
[6] Jonathan Weinberg," Job Scheduling on Parallel Systems", in Proc. International Conference on Job Scheduling Strategies for Parallel Processing *(IPPS)* CA 92093-0505, 2007.
[7] Y. Chow and W.H. Kohler,"Models for Dynamic Load Balancing in a Heterogeneous Multiple Processor System", IEEE Trans. Computers, vol. 28, pp. 354-361, 1979.
[8] Sangsuree Vasupongayya, Su-Hui Chiang "On Job Fairness in Non-Preemptive Job Scheduling", in Proc.International Conference on Parallel and Distributed Computing, Phoenix, AZ, USA, Nov 14-16, 2005.
[9] Jonathan Weinberg, "Job Scheduling on Parallel Systems**"**, in Proc. International Conference on Job Scheduling Strategies for Parallel Processing (IPPS), 2002.
[10] Edi Shmueli, Dror G. Feitelson. On "Simulation and Design of Parallel-Systems Schedulers: Are We Doing the Right Thing?" IEEE transaction on Parallel and Distributed System, Vol. 20, No. 7, pp 983-996, July 2009.
[11] Mahima Shrivastava, "Analysis and Review of CPU Scheduling Algorithms", International Journal of Scientific Engineering and Research (IJSER) ISSN (Online): 2347-3878 Volume 2 Issue 3, March 2014.
[12] Debashee Tarai, "Enhancing Cpu Performance Using Subcontrary Mean Dynamic Round Robin (Smdrr) Scheduling Algorithm", International Journal of Scientific Engineering and Research (IJSER) Volume 2 Issue 3, March 2011.
[13] Vinay G. Vaidya," Dynamic Scheduler for Multi-core Systems",2010 2nd International Conference on Software Technology and Engineering(ICSTE) 2010.
[14] Imran Qureshi, "CPU Scheduling Algorithms: A Survey", Int. J. Advanced Networking and Applications Volume: 05, Issue: 04, Pages: 1968-1973 (2014) ISSN : 0975-0290.
[15] Rakesh Mohanty, H. S. Behera, Khusbu Patwari, Monisha Dash, Design and Performance Evaluation of a New Proposed Shortest Remaining Burst Round Robin (SRBRR) Scheduling Algorithm, In Proceedings of International Symposium on Computer Engineering & Technology (ISCET), Vol 17, 2010.
[16] Yu-Kwong Kwok," Amalysis, Evaluation, and Comparison of Algorithms for Scheduling Task Graphs on Parallel Processors", 1087-4089/96 $5.00 0 1996 IEEE.
[17] Thu D. Nguyen, "Using Runtime Measured Workload Characteristics in Parallel Processor Scheduling", National Science Foundation (Grants CCR-9123308 and CCR-9200832) 1999.
[18] Thu D. Nguyen, "Parallel Application Characterization for Multiprocessor Scheduling Policy Design", National Science Foundation (Grants CCR-9123308 and CCR-9200832) 2001.
[19] Sujatha R. Upadhyaya." Parallel approaches to machine learning—A comprehensive survey", J. Parallel Distrib. Comput. 73 (2013) 284–292. 2012.
[20] Krste Asanovic. "A View of the Parallel Computing Landscape" communications of the acm vol. 52, no. 10, october 2009.

[21] Sourav Kumar Bhoi "Enhancing Cpu Performance Using Subcontrary Mean Dynamic Round Robin (Smdrr) Scheduling Algorithm" 2014.
[22] Kalpana Gupta "A Comparative Study Of Two Operating Systems:
[23] Windows 7 And Windows 8" *International Refereed Multidisciplinary Journal Of Contemporary Research* Eissn 2320-3145, Issn 2319-5789, 2013.