A Hybrid Modular Approach to Minimize Job's Execution Energy in Heavy Load on Cloud Network

Rajpreet Kaur[1], Diljit Kaur[2]
[1]*Reseach Scholar,* [2]*Assistant Professor*
Department of Information Technology, CEC Landran, Mohali (Punjab)

***Abstract* –** The optimizing abilities and dispersal of mobile technology increase the opportunity to evaluate potentially unstable moveable devices as sources into grids. In this proposed paper, we discussed by proposing and calculating the enhance de-centralized job scheduling method for mobile point to point communication forming a grid. Cloud computing is the internet associated mode of super-computing. It is a type of communal environment, which normally puts the large system pools together by using numerous means; (i) Distributed and (ii) Virtualization etc. Scheduling refers to a set of schemes and machinery to control the order of work to be achieved by a computer system. Job scheduling is serious if the full computational power of large scale multicomputer is to be attached effectively. The aim of job scheduling is to select the next job to be implemented while the area of processor distribution is to select the set of processors on which parallel jobs are executed. The proposed algorithm is a hybrid of two techniques known as multi-queue scheduling algorithm and P-Thread algorithm. Both of them are scheduled in such a way to reduce energy consumption for tasks execution on the cloud network. The overall working is divided into two different modules in this research. A P-thread algorithm which makes the scheduling process more efficient and less time-consuming. The P-Thread algorithm executes all the tasks in a queue with multi-threading technique. This process discussed in result analysis in the next research paper, job scheduling used with a crossbreed approach to reduce the waiting time between all the possible solutions and provides maximum throughput and reduce execution cost.

Keywords- *Cloud computing, job scheduling, P-thread, multi-queue scheduling, and throughput.*

I. INTRODUCTION

In multiprogramming systems, when there is an additional runnable process, the operating system must agree which one to activate. The decision is made by the part of the operating system called the scheduler, using a scheduling algorithm. Scheduling refers to a set of schemes and machinery to control the order of work to be achieved by a computer system. Of all the resources in a computer system that are programmed before use, the CPU is by far the most imperious. Multiprogramming is the scheduling of the CPU. The basic idea is to keep the CPU [1] busy as much as possible by executing a process until it must wait for an event, and then switch to another process. Procedures alternative between uncontrollable CPU cycles and performing I/O.

In all-purpose, job scheduling is attained in three stages [2]:

- Short
- Medium and
- Long-term.

A job scheduler is a platform that enables an initiative to schedule and, in some cases, monitor computer "batch" jobs. A job scheduler can recruit and manage jobs, mechanically by processing organized job control semantic accounts or through corresponding interaction with a human operator. Today's job schedulers classically provide a graphical user interface and a single point of control for all the work in a circulated network of computers [3]. Job scheduling is serious if the full computational power of large scale multicomputer is to be attached effectively. The aim of job scheduling is to select the next job to be implemented while the area of processor distribution is to select the set of processors on which parallel jobs are executed. The major advantages of job scheduling are the classification scheduled processes that are dependent on multiple systems. Allows IT section is to focus on carrying value added IT services rather that deal with day-to-day automation or scheduling issues. Offers the ability to achieve scheduled job load on databases and file servers from one central position. One central location is for distributing and managing alerts as and when they occur. Console view of what ensued overnight [4].

II. IMPORTANCE OF THE JOB SCHEDULING

Once you have set areas to achieve approximately, you will need to begin appropriately managing your time in order to work towards these goals. After you have shaped a To-Do list, you will next need to create a schedule which will allow you to comprehend all the significant tasks within a given period of time. No matter who you are, you will only have a certain quantity of time to get something done. Setting up a schedule will allow you to develop a systematic time frame which will allow you to complete the project on time. While most people are acquainted with the concept of schedules [5], They were showing authoritative methods can use to make sure the schedule is designed competently. In addition to this, it is

significant for you to make sure you follow whatsoever is on the schedule. The use of a schedule is significant in time management because it can allow you to know what you can do in a convinced period of time, use the time you have wisely, give you enough time to comprehend the most important things, and reserve time to deal with unexpected situations. In adding to these things, it will also keep you from taking on more than you can holder.

III. RELATED WORK

Dongfeng Jiang et al., 2007 [6] survey was presented on the difficult and the different features of job scheduling in grids such as

(a) Fault tolerance;
(b) Security; and
(c) Reproduction of grid job scheduling strategies is deliberated.

This paper also offers a conversation on the future research subjects and the challenges of job scheduling in grids.

Bo Yang et al., 2011 [7] termed as, cloud computing' service concerned with features improvement a new way of provision provisioning called utility-based computing. However, toward the applied application of commercialized Cloud, they meet two challenges:

i) There is no well-defined job scheduling algorithm for the Cloud that reflects the system state in the coming, mostly under congestion circumstances;
ii) The remaining job scheduling algorithms under usefulness computing pattern do not take hardware or software failure and recovery in the Cloud into account.

In an effort to address these experiments, they introduce the disappointment and recovery situation in the Cloud computing entities and suggest a Reinforcement Learning based procedure make job scheduling faults bearable while maximizing utilities achieved in the long term.

Ivan Rodero et al., 2009 [8] described and evaluated them synchronized grid scheduling strategy. They take as a reference the FCFS job preparation policy and the matchmaking method for the resource selection. They also current a new job scheduling strategy based on backfilling that aims to advance the workloads implementation a performance, avoiding famine and the SLOW-coordinated resource selection policy that deliberates the average bounded stoppage of the resources as the main limitation to perform the resource selection.

S.K.Aparnaa et al.,2014 [9] considered the memory requirement of each collection, which is one of the main reserves for scheduling data exhaustive jobs. Due to this the job disappointment rate is also very high. To deliver a solution to that problem Greater Adaptive Scoring Job Scheduling algorithm is familiarized. The jobs were recognized whether it is data concentrated or computational intensive and based on that the jobs were scheduled. The jobs were allocated by computing Job Score along with the memory condition of each cluster. Due to the vibrant nature of grid environment, each time the position of the resource changes and each time the Job Score was calculated and the jobs were allocated to the most suitable resources.

Keqin Li et al,2004[10] compared the performance of various job scheduling and processor allocation procedures for grid computing on meta -processors. They appraise the performance of 128 arrangements of two job scheduling algorithms, four original job ordering strategies, four processor distribution algorithms, and four Meta computers by widespread simulation.

Saad Bani-Mohammad et al.,2012 [11] performance of the well-known Greedy Available, Busy List non-contiguous distribution strategy for 2D mesh-connected multiprocessors is re-visited seeing several important job scheduling approaches. These are the First- Come-First-Served, Out-of-Order, and Window- Based job scheduling strategies. They were compared using thorough flit-level simulations. Widespread simulation results based on artificial and real workload models specify that the Window-Based job scheduling strategy exhibitions good performance when the scheduling window size is large and heavy system loads.

IV. PROPOSED METHODS IN JOB SCHEDULING

In this section, we discussed the working of the proposed methods in job scheduling (i) Multilevel queue and (ii) P-thread job scheduling algorithms. Proposed approach is a hybrid of these two approaches. Multilevel queue scheduling approach is used to handle various input queues on the cloud server by connected users. It handles all the queues and processes them efficiently to reduce energy and execution time. Once all the queues are managed with Multilevel queue scheduling it moves to another part of the hybrid approach. Here in this, the threading concept is used to manage the processing of various internal tasks of the queues. Tasks management is also an important issue in the scheduling process. The proposed approach creates multiple threads for inner tasks and reduce the waiting time of execution in the overall execution process.

A. Multi-level queue schedule permits a process to move between queues. This movement is facilitated by the type of the CPU burst of the process [12]. If a developer uses too much CPU time, it will be refreshed to a lower-priority queue. This structure leaves I/O-bound and interactive processes in the higher priority queues. In addition, a process that waits too long in a lower-priority queue may be moved to a higher priority queue. This form of aging also helps to avert starvation of convincing lower priority processes. Multiple FIFO queues are used and the action is as surveys:

1. A new process is inserted at the end (tail) of the top-level FIFO queue.
2. At some stage, the process extends the head of the queue and assigns the CPU.
3. If the process is complete within the time quantum of the given queue, it leaves the system.
4. If the process happily ends control of the CPU, it leaves the queuing network, and when the process develops readily again, it is inserted at the tail of the same queue which it abandoned earlier [13].
5. If the process uses all the quantum time, it is pre-empted and inserted at the end of the next lower level queue. This subsequent lower level queue will have a time significant which is more than that of the previous higher level queue.
6. This scheme will endure until the process finalizes or it reaches the base level queue.

- At the base level queue, the processes circulate in round robin fashion until they complete and leave the arrangement. Processes in the base level queue can also be scheduled on a first come first served basis.
- Optionally, if a progression block for I/O, it is 'promoted' one- level, and placed at the end of the next-higher queue. This allows I/O bound processes to be favored by the scheduler and allows processes to 'escape' the base level queue.

B. P-Thread Job Scheduling: The operating system incessantly selects a single thread to run from a system-wide collection of all threads that are not waiting for the completion of an I/O request or are not blocked by some other activity.

- Many threaded programs have no reason to interfere with the default behavior of the system's scheduler. However, the Pthreads typical defines a thread-scheduling border that allows programs with real-time tasks to get involved in the process.
- Using the Pthreads preparation feature, you can select how threads share the available processing power. You may decide that all outfits should have equal access to all available CPUs, or you can give some threads preferential treatment.
- In some requests, it's beneficial to give those outfits that perform important tasks an advantage over those that complete related work. For illustration, in a process-control submission, a thread that answers to input for special strategies could be given priority over a line that simply maintains the log [14].
- Used in conjunction with POSIX real-time extensions, such as memory, securing and real-time clocks, the P-threads scheduling feature lets you create real-time applications in which the threads with imperative tasks can be defined to comprehensive their jobs in an expectable, finite amount of time[15].

C. Hybrid Approach: Hybrid approach in this research is having two different modules to process the data on the cloud network. The first module in this approach used to manage the queues and other input tasks on the cloud server. It reduces the load on a cloud server and starts execution process with well-managed structure. The process of queue handling is an outer part of the scheduler that pre-processes the input tasks and provides an optimized solution to the inner module. Execution pattern is decided in the inner module so here need to introduce an another processing architecture which helps the system to balance the load on the cloud network. The second module processes all the threads and manages the waiting time of the overall process. This approach saves energy consumption and responds to user queries faster than the other existing approaches.

Here in the figure, 1 proposed architecture is a block diagram of overall execution scheme. It defined the execution pattern of input queues and their processing on cloud network with multiple threading techniques.

Proposed approach can achieve a maximum saving of energy consumption in future development for cloud processing environment.

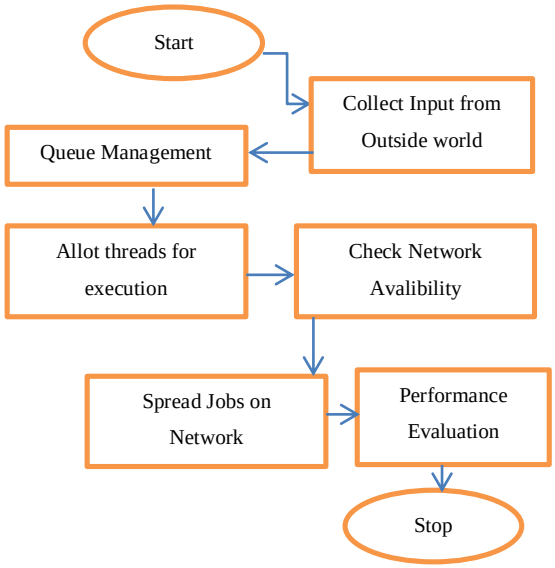

Fig.1: Proposed Architecture

V. CONCLUSION

In this research, the paper focus is on maintaining the job scheduling system which is degraded during the sorting process. A novel technique is proposed based upon reliability

to overcome the problem of maintaining structural of jobs. We design a system framework & draw the flow chart of research work and methodology, in the proposed Multi-queue scheduling & p-thread algorithm. These algorithms have been taken into consideration by replacing the already existing techniques. The system was planned with the prime focus on optimization so as to achieve accurate results.Job Scheduling is intended to accomplish not solitary competent processing as well as the use of computing infrastructure, but also minimize energy consumption. It is important for confirming in which the future development of Cloud-computing is quite sustainable. We will deploy hybridization of multi-queue scheduling and Pthreads scheduling algorithm to achieve an overhead problem. The mainstream of the prior investigation work done in the area of analyzing power/ energy utilization mainly distillates on Job Scheduling in the middle with respect to Job allocation among the application servers, directed power saving or the criteria seeing thermal factors and hybridization of multilevel queue scheduling and Pthreads scheduling algorithm performance shows in a further paper.

VI. REFERENCES

[1]. Hummel, Karin Anna, and Gerda Jelleschitz. "A robust decentralized job scheduling approach for mobile peers in ad-hoc grids." Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on. IEEE, 2007.

[2]. Edwin, A. Chinnu, and A. Neela Madheswari. "Job Scheduling and VM Provisioning in Clouds." Advances in Computing and Communications (ICACC), 2013 Third International Conference on. IEEE, 2013.

[3]. Al-Najjar, Hazem, and Moath Jarrah. "Smart job scheduling with backup system in grid environment." Networks (ICON), 2012 18th IEEE International Conference on. IEEE, 2012.

[4]. Dwivedi, Sanjay K., and Rajesh Gupta. "A simulator based performance analysis of multilevel feedback queue scheduling." Computer and Communication Technology (ICCCT), 2014 International Conference on. IEEE, 2014.

[5]. Zeng, Chengkuan, Jiafu Tang, and Huabo Zhu. "Two heuristic algorithms of job scheduling problem with inter-cell production mode in hybrid operations of machining." Control and Decision Conference (CCDC), 2013 25th Chinese. IEEE, 2013.

[6]. Jiang, Congfeng, et al. "A survey of job scheduling in grids." Advances in Data and Web Management. Springer Berlin Heidelberg, 2007. 419-427.

[7]. Yang, Bo, et al. "An utility-based job scheduling algorithm for cloud computing considering reliability factor." Cloud and Service Computing (CSC), 2011 International Conference on. IEEE, 2011.

[8]. Rodero, Ivan, Francesc Guim, and Julita Corbalan. "Evaluation of coordinated Grid scheduling strategies." High Performance Computing and Communications, 2009. HPCC'09. 11th IEEE International Conference on. IEEE, 2009.

[9]. Aparnaa, S. K., and K. Kousalya. "An Enhanced Adaptive Scoring Job Scheduling algorithm for minimizing job failure in heterogeneous grid network." Recent Trends in Information Technology (ICRTIT), 2014 International Conference on. IEEE, 2014.

[10]. Li, Keqin. "Experimental performance evaluation of job scheduling and processor allocation algorithms for grid computing on metacomputers."Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International. IEEE, 2004.

[11]. Bani-Mohammad, Saad. "On the performance of job scheduling for noncontiguous allocation in 2D mesh-connected multicomputers."Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean. IEEE, 2012.

[12]. Schwiegeishohn, Uwe, &RaminYahyapour. "Improving first-come-first-serve job scheduling by gang scheduling." Job Scheduling Strategies for Parallel Processing. Springer Berlin Heidelberg, 2010.

[13]. R. Maggiani, "Cloud computing is changing how we communicate," 2009 IEEE International Professional Communication Conference, IPCC 2009,Waikiki, HI, United states ,pp 1, July 2009.

[14]. Schrage, Linus E., & Louis W. Miller. "The queue M/G/1 with the shortest remaining processing time discipline." Operations Research 14.4 (1966): 670-684.

[15]. VasiliosAndrikopoulos, Zhe Song, Frank Leymann, "Supporting the Migration of functions to the Cloud through a Decision Support System", Institute of Architecture of function Systems, IEEE, pp. 565-672, 2013.