**International Journal of Innovative Research in Engineering**
Volume 3, Issue 3 (May-June 2022), PP: 28-30.
www.theijire.com
***ISSN No: 2582-8746***

# Resource management using parallel computing: A Review

**Shivani Satish Bhange[1], Shruti Rajesh Shastrakar[2], Smital Ravindra telang[3] , Divya Waman Raut[4] , Diskha Rupesh mendhe[5], Surbhi Khare[6]**
*[1,2,3,4,5]Information Technology/ Priyadarshini college of Engineering, RTM Nagpur University, India.*
*[6]Assistant professor, Information Technology/ Priyadarshini college of Engineering, RTM Nagpur University, India.*
.

**How to cite this paper:**
Shivani Satish Bhange[1], Shruti Rajesh Shastrakar[2], Smital Ravindra telang[3] , Divya Waman Raut[4] , Diskha Rupesh mendhe[5], Assistant professor Surbhi Khare[6], "Resource management using parallel computing: A Review", IJIRE-V3I03-28-30.



***Abstract:*** *In this project we have discussed the challenges and opportunities for efficient parallel data processing environment and the first data processing framework to exploit the dynamic resources provision offered today's peer-to-peer system. We have described basic architecture and presented a performance comparison to the well-establish data processing framework Hadoop. The performance evaluation gives a first impression on how the ability to assign specific virtual machine types to specific task of the processing job, as well as the Possibility to automatically allocate/deallocate virtual machines in the course of the job execution, can help to improve overall resource and the time utilization and consequentially reduce the processing cost. Over current profiling approach builds a valuable basis for this however, at the moment the system still requires a reasonable amount of user annotation. In general, we think over work represent an important contribution to the growing field of cloud computing services and point out exiting new opportunities in the field of parallel data processing.*

***Key Word:*** *data processing, time utilization, parallel data processing*

## I. INTRODUCTION

Today a growing number of companies have to process huge amount of data in a co-efficient manner. Classics representative for these companies are operators of internet search engines like google, yahoo or Microsoft. The vast amount of data they have to deal with every day has made traditional data base solution prohibitively expensive. instead, these companies have popularized an architecture Paradigm based on a large number of commodity servers. Problems like processing crawled document of regenerating a web index are split into several independent subtask, distributed among the variable nodes, and compute in parallel in order to simplify the development of distributed application on a top architecture, many of these companies have also build customize data framework. Examples are; goggle map reduce, Microsoft's dryad, or Yahoo!'s map-reduce-merge they can be classified by terms like high throughput computing (HTC) or many task computing (MTC), depending on the amount of data and the number of tasks evolved in the computation, although this system different design, there programing models share similar objectives, namely heading the hassle of parallel programming, fault tolerance and execution optimization from the developer. Developer can typically continue to write sequential program the processing framework then takes care of distributing the program among the available nodes and execute each instance of the program on the appropriate fragment of data static.

## II. PROBLEM DEFINITION

The goal of a cloud-based architecture is to provide some form of elasticity, the ability to expand and connect capacity on-demand. The implication is that at some point additional instances of an application is that at some point additional instances of an application will be needed in order for the architecture to scale and make demand. That means there needs to be some mechanism in place to balance request between two or more instances of that application. A Load balancer provides the means by which instance of application can be provisioned and deprovisioned automatically without requiring changes to the network or its configuration.it automatically handles the increase and decrease in capacity and adapt its distribution, decision based on the capacity available at the time a request is made. Load balancing ensures that all the processor in the system or every node in the network does approximately the equal amount of work at any instant of time. This technique can be sender initiated, receiver initiated or symmetric type (combination of sender initiated and receiver-initiated types). Our objective is to develop and effective load balancing algorithm maximize or minimize different performance parameters (Through put, latency for example) for the clouds of different sizes (virtual topology depending on the application requirement) The system will take the service as input which is requested by the cloud customer and then the role of load of Balancer begins. It will communicate with the client present in its network and assign the load to the least loaded one.

**Advantages of parallel computing:**

1. It saves times and money as many resources working together will reduce the time and cut potential cost
2. It can be impractical to solve larger problems on serial computing.
3. It can take advantages of non-local resources when the local resources are finite.
4. Serial computing 'waste' the potential computing power thus parallel computing makes better work of the hardware.

**Applications of parallel computing:**
1. Database and data mining
2. Real time simulation of system
3. Science and engineering.
4. Advanced graphics vs augmented reality, and virtual reality.

## III. PARALLEL IMPLEMENTATION OF RESOURCE MANAGEMENT AND OPTIMIZATION

An effective resource utilization of the modern high performance computing platform is a subject for many scientific research investigations. The resource management optimization for those platforms is an essential part for optimal resource allocation while solving hard problems. An effective resource management algorithm strongly determines the overall parallel performance of the high-performance computing system.
Resource optimization is the set of processes and methods to match the available resources with the needs of the organization in order to achieve established goals optimization consists in achieving desired results within a set timeframe and budget with minimum usage of the resources themselves. The need to optimize the resource is particularly evident when the organization demands tends to saturate and /for exceed the resources currently available. When a company is managed using a systematic approach, resource optimization is strictly linked to the concept of concept of constant and a systematic vision of the company indeed, without a systematic vision of the company we are unable to identify the global effectiveness of resource allocation and we run the risk of using resource available mainly to respond to emergencies that daily occur in the various parts of the organization.
A project is itself a system a network of elements that are interconnected and interdependent that work together to achieve a precise goal.

## IV. OBJECTIVES

**Concurrency:** The system distributed each task and sub tasks over several machines foe simultaneous execution. Conversely individual participant nodes are capable for executing multiple subtask (possibly form different tasks) at the same time multi-threading.
**Decentralization:** The structure of centralized topologies yield "inefficiencies, bottlenecks, and wasted resources". A fully-decentralized system dived the cost of ownership of fairly amongst all the participant, and avoids any central points of failure.
**Scalability:** The system may accommodate any number of participant nodes without experiencing n any significant performance degradation. The throughput of the system (in terms of number of tasks it can complete in given time interval) scales almost linearly with the number (and processing power) of participating nodes available.
**Efficiency:** The system embodies a degree of intelligence of distributing tasks in an optimal manner each node maintains a partial or complete awareness of the composition of the global mesh and the delegates subtasks in a way that reduce their execution times and promotes overall throughput.
**Fall tolerance:** Since. A P2P system involves nods leaving the network in the unannounced and non-deterministic manner, recovery mechanism is enacted for resuming subtask which had been delegated to the departed nodes. This is achieved through a checkpointing strategy.
**Ease-of-Use:** The API is aspired to the appealing, user friendly and unobtrusive to the extent that it could be easily utilize without requiring technical knowledge of the inner working of the system.

## V. FEATURES

**1]** The initialization and management of virtual computers which enroll the requested workers number of workers to participant in particular task.
**2**] The management and deployment of virtual threads through virtual computers, with each virtual thread Encapsulating the execution of specific subtask.
**3**] The returning of results to the user applications once a virtual threads complete execution (note that his features along with the previous to, would suffice of a task-farming applications).
**4]** The reporting of partial results to the user applications during the virtual thread's execution using an event -driven approach.
**5]** Direct communication between any pair of virtual threads using buffered message passing.
**6]** Mandatorily – synchronized access the global variable, whose value would be common to all virtual threads within a virtual computer.
**7]** Checkpointing of the execution state of each individual virtual threads (losing process migration), allowing it to be resumed on another peer if its hots should disconnect or die.

## VI. CONCLUSION AND FUTURE SCOPE

The first objective of this paper was to demonstrate that the performance of parallel applications could be measured at the run-time. We started from the idea that the system cannot really on user's requirements. Based on the premise we propose that the system considered user requirements as hints, but that a dynamically evaluate the real performance that applications are reaching. We assume that the behavior of several iterations cane be extrapolated to the behavior of the complete application. The speedup measure as the ratio between the average execution times of several iterations with the base line number of

processors and the average executions times with the available number of processors. We construct the dynamic job arrival rate pattern and carry out rigorous simulation experiments to compare the performance of the three algorithms under different system loads, with different status exchange intervals. Our system model and experimental study can be directly extended to the large size network, such as multidimensional hyper cube networks to test their performance finally, in this paper we have to rigorously demonstrated the performance of the algorithm for a single class of job.in our near future work we intent to divide the job in the system into serval classes and assign each class of the job its own priority.to test their performance a dynamic job arrival rate pattern and carry out rigorous simulation experiment to compare the performance of the three algorithm under different system loads ,with the different status exchange intervals. The computational graph has undergone a great transition from serial computing to the parallel computing tech giant such as intel has already taken a step towards parallel computing buy employing multicore processors. Parallel computation will revolution the way computers work in the future, for the better goods with all the world connecting to each other when more than before, parallel computing does a better role in helping us stay that way. With faster network, distributed system, and multiprocessor computers, it become even more necessary.

## Reference

*1] D.Battr'e, S.Ewen, F.Hueske, O.Kao, V. Markl and D.Warnek. Naphele/PACTs: Programming models and execution framework for web-scale analytical processing in SoCC'10: Proceeding to the ACM Symposium on cloud computing 2010.*

*2] Hanan M.Shupur, Aubhi R.M.zeebare, Addulraheer Jamil Ahmed, Rizgar, Omar M .Ahmed, Bareen Shamsadleen Tahir, Mohammed A.M.Sadeeq. A state of art survey for concurrent computation and clustering the parallel computing for distributed system 2019.*

*3] D.Warnek and O.Kow.Nephene: efficient parallel data processing in the cloud. In MTAGS'09: Proceeding of the second workshop in many tasks computing on a grid and super comuter2009.ACM.*

*4] I.Raicu, I.Foste, Y.Zaho.many task computing for grid and supercomputers in many task computing on a grid and super computer MTAGS2008.*

*5] D.S.Milojicic, et al., "peer-to-peer" technical report HPL-2000-57R1,HPlaboritries, palo alto,California,2002.*

*6] A.W.Loo, "The future of peer-to-peer computing" communication of the ACM, Sep 2003.*

*7] V. Donaldson , F.Baramn , and R.Paturi, "program speedup in a heterogenous computing network ," general of parallel and distributed to computing june1994.*

*8] Gropp,W.Lusk, E.,Doss, N. And Skjellum, A., 1996.A high -performance, portable implementation of the MPI message we passing interface standard Parallel computing.*

*9] Satto,M.,2002,October. OpenMP: Parallel programming API for shared memory multiprocessor and on -cheap multiprocessor.in the proceeding of the fifteenth international symposium on system synthesis ACM.*

*10] Qin, X. and Jiang H.2005. A dynamic and reliability driven scheduling algorithms for the parallel real dash time jobs. Executing on heterogenous clusters. Journal of parallel and distributed computing.*

*11] D. Eans and W.Butt, "dynamic load balancing using task – transfer probability," parallel computing 1993.*

*12] J.Watt and S.Tellered, "a practicular approach to dynamic load balancing ," IEEE turns. Parallel and distributed system 1998.*

*13] Z.Zeng and V.Bharadwaj , " a static load balancing algorithm via virtual routing" proc. Conf. Parallel and distributed system2003.*

*14] J.Li and H. Kameda, "load balancing problems for multi class jobs in distributed /parallel computers system," IEEE trans. Computers,1998.*

*15] C.Walshaw and M.Derzins,"dynamic load balancing for PDE solvers and adaptive on structure Meshes," concurrency ; practice and experience 1995.*