Volume - 5 , Issue - 7 , February - 2018 ISSN 2347-2723

REVIEWS OF LITERATURE

An International Multidisciplinary Peer Reviewed & Refereed Journal

Impact Factor: 3.3754

UGC Approved Journal No. 48385

REVIEWS OF LITERATURE
ISSN: 2347-2723
IMPACT FACTOR : 3.3754(UIF)
VOLUME - 5 | ISSUE - 7 | FEBRUARY - 2018

FRAMEWORK TO MINIMIZE THE COMMUNICATION OVERHEAD IN DISTRIBUTED SYSTEMS

Dr. Putti SrinivasaRao
Professor & HOD in CSE & Dean of PG Studies ,
JBIET Telangana ,Hyderabad, India .

ABSTRACT: -
The communication overhead in identifying best processors, checking loads of processors by each processor, absence of a centralised monitor and current load information etc. have been the major limitations of the previous distributed systems. The framework presented here addresses these limitations in a comprehensive manner. The Model presented here is such that each of the components are self-contained and is not depend on each other. it also ensures safe, secure and reliable communication in the distributed systems .

KEYWORDS: *communication overhead , ensures safe, secure and reliable communication .*

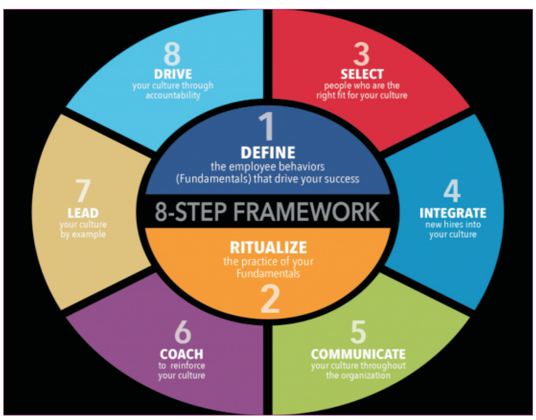
I . INTRODUCTION :

A distributed system consists of many heterogeneous processors with different processing power and all processors are interconnected with a communication channel. System performance can be improved by using proper load balancing. if some processors are less loaded or idle and others are heavily loaded, the system performance will be reduced drastically.

The most important aspect of a distributed computing environment is effective coordination of the resources available across several nodes. The distribute the tasks should be in such a manner that the task executions are completed in most optimum manner, keeping the communication overhead is minimum and ensuring the resources available are utilised effectively .

II DISTRIBUTED JOB PROCESSING SYSTEM

Distributed System consists of several systems connected by a network . The nodes may not be similar in configuration, having completely different capabilities The basic requirements of a Job Processing system are:
1. Accepting jobs from different channels
2. Scheduling the jobs



3.Dispatching to the appropriate processor
4.Ensuring the job is completed
5.Managing different job priorities
6.Handling job dependencies
7.Notification mechanisms
8.Monitoring and Control

However, there are many challenges in this type of a system Some of the important challenges are given below.
1.The input channels to be supported and the technique therein.
2.Ensuring that once a job is accepted, the system processes the job, completes the job and the job reaches a final state.

3. In case of crashes, it should be possible to re-start the job and it should continue from where it had left earlier.
4. For a long running job, it should be possible to track the progress of the job processing.

A Job can be defined as a set of computational tasks, defined in a pre-determined manner, to be performed on the identified data set and the result is generated. Thus, once submitted, the job is expected to complete its execution and submit the result in the form that is already defined.

let us consider the design to accommodate different types of jobs using different Message Queues.

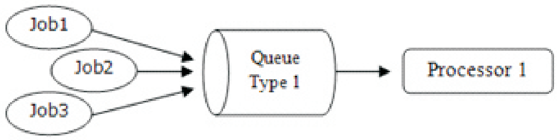


Figure1 : Less frequent jobs, one queue, one processor

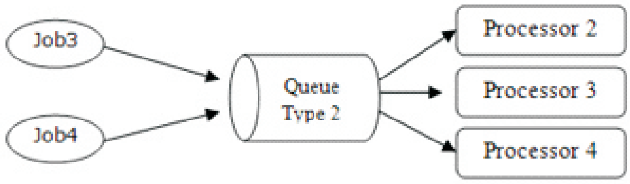


Figure2 : More frequent jobs, one queue, more processors

The Figure 1 depicts a scenario where jobs of Type 1 are provided with a queue of Type 1 and a single processor. Such jobs are not so frequent jobs and the processing time is far less than the rate at which the jobs arrive, thus a single processor is capable of processing all the jobs that come in without (or minimal) any wait time.

The Figure 2 depicts a little more complex scenario where the job incoming rate is high. So, having a single processor may not be sufficient to handle the load within the acceptable wait time. So, such types of jobs are provided with a separate Message Queue (of Type 2) and multiple processors.So, depending on the type of jobs, we can decide (and configure) on the number of Message Queues that can handle the load and the number of processors that would listen on the respective queues.

III Drawbacks of the Conventional communication in DS

A node is responsible for processing as well as job forwarding if it is loaded.

When a node receives a job and is loaded beyond its capacity, it needs to query the status of other nodes to find out if any other node can share the load.

Quite a good amount of time is wasted at each node to query other nodes. This time could have been utilized for processing the job.

In a complex network, having multiple sub-networks, configuring each node for locating other nodes is a complex task.

Assuming, the nodes use broadcast to announce their status, this also causes enormous load on the network.

When every node requests the status of other nodes periodically, the network overhead increases many fold

III PROPOSED APPROACH TO REDUCE THE COMMUNICATION OVERHEAD

The communication overhead for the processors in enquiring loads of peer processors is reduced by making the load information available at a central place (with clustered and failover mechanism to avoid single point failures). The proposed model also introduced the cost factor for Job processing by introducing network cost, storage cost and computational cost for each processor. Apart from the number of Jobs pending with a processor, the load computation also considered the cost factor by assigning weights to a varying degree for each of the above mentioned costs. The cost is computed as overhead percentage and the best processor is calculated based on the processor having the least cost.

The various components of the system are

VOLUME - 5 | ISSUE - 7 | FEBRUARY - 2018

1.Job Dispatcher accepts new job requests validates them and places the jobs in the Job Queue for scheduling.
2.Job Scheduler accepts the job requests from the Dispatch queue and schedules jobs for processing based on the load .
3.Job processor picks up a job request from the Processing queue, processes
4.Job Monitor monitors the status messages and updates the database
5.Dispatch Queue stores the job requests dispatched until it is picked up by the scheduler Processing queue that stores the scheduled jobs until a processor picks them up for processing
6.Progress / Status Queue store the job status sent by either dispatcher or processor
7.Database / Persistence All the information about the Job, the Processors, the state of processing and the availability of processors are maintained

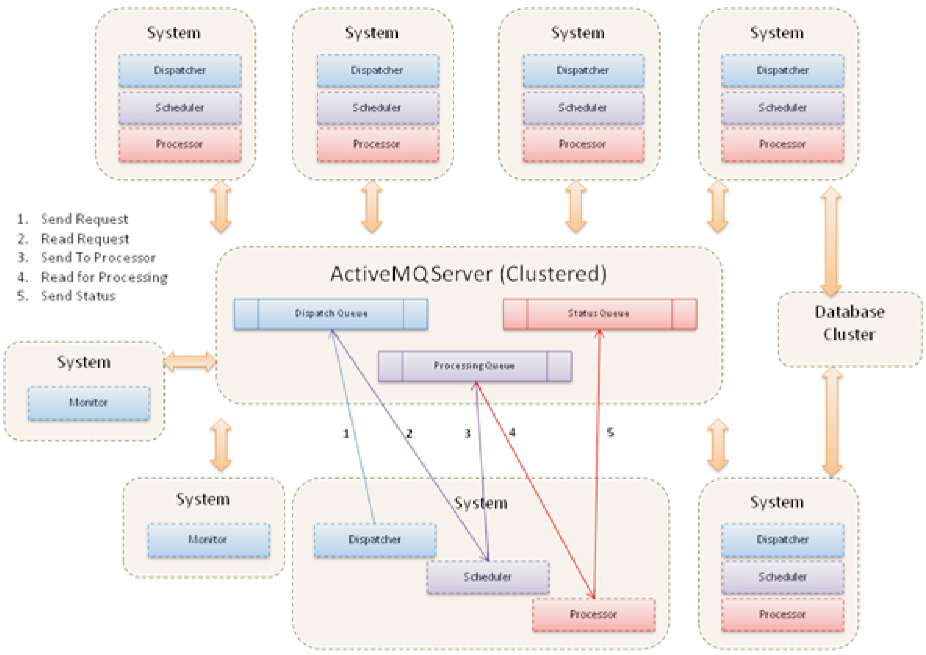


Fig: Architecture of the proposed System

V COMPARISON EXITING AND PROPOSED MODELS

Parameter	Sender Initiated approach	Receiver initiated approach	Proposed Approach
Nature	Dynamic	Dynamic	Dynamic
Overall Rejection	Yes	Yes	No
Reliability	Low	Low	High
Adaptability	Low	Low	High
Stability	Low	Low	High
Predictability	Low	Low	High
Fault Tolerant	No	No	Yes
Resource Utilization	Less	Less	Medium/high
Process Migration	yes	Yes	No
Response Time	Low	medium	Low
Waiting Time	High	Low	Low/little more
Turn-around Time	Medium	Medium	Low
Throughput	Low	Low	High
Processor Thrashing	high	High	Low
Cost	high	High	Low

VOLUME - 5 | ISSUE - 7 | FEBRUARY - 2018

VI CONCLUSION:

This Paper provided a Framework to Minimize the communication overhead in Distributed Systems. it also showed the comparison of different performance improvement parameters with existing models and proposed models

REFERENCES :

[1] Zeng Zeng, and Bharadwaj Veeravalli "Design and Performance Evaluation of Queue-and-Rate-Adjustment Dynamic Load Balancing Policies for Distributed Networks", IEEE Transactions on computers, VOL. 55, NO. 11, NOVEMBER 2006.

[2] Abhijit , Rajguruand and S.S. Apte "A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters", International Journal of Recent Technology and Engineering (IJRTE) , ISSN: 2277-3878 Volume-1, Issue-3, August 2012.

[3] Iman Sadoop, Sandeep Palur and Ioan Raicu "Achieving Effiecient Distributed Scheduling with Mesage Queue in Cloud Computing for Many-Task Computing and High –Performance Computing".

[4] Andrei Radulescu, Arjan J.C. and van Gemund "Low-Cost Task Scheduling for Distributed-Memory Machines" , IEEE Transactions on Parallel and Distributed Systems VOL. 13, NO. 6, JUNE 2002.

[5] K.Q. Yan, S.C. Wang, C.P. Chang and L.Y. Tseng "The Anatomy Study of Load Balancing in Distributed System", Proceedings of the Seventh International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'06). 2006.

[6] "Kevin Barker, Andrey Chernikov, Nikos Chrisochoides, and Keshav Pingali "A Load Balancing Framework for Adaptive and Asynchronous Applications", IEEE Transactions on Parallel and Distributed Systems, VOL. 15, NO. 2, FEBRUARY 2004.

[7] Jorge E. Pezoa "Decentralized Load Balancing for Improving Reliability in Heterogeneous Distributed Systems ", 2009 International Conference on Parallel Processing Workshops.

[8] Jorge E. Pezoa and Majeed M. Hayat. "Reliability of Heterogeneous Distributed Computing Systems in the Presence of Correlated Failures" . IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 4, APRIL 2014.

[9] Young Joon Lee, Geon Yong Park, Ho Kuen Song, and Hee Yong Youn "A Load Balancing Scheme for Distributed Simulation Based on Multi-Agent System".2012 IEEE 36th International Conference on Computer Software and Applications Workshops.

[10] Deng Huafeng, Zhong Linhui, and Ye Maosheng "An Efficient Load Balancing Algorithm in Distributed Systems". 2010 International Forum on Information Technology and Applications.

[11] Gábor Vincze, Zoltán Novák, Zoltán Pap, and Rolland "RESERV: A Distributed, Load Balanced Information System for Grid Applications", Eighth IEEE International Symposium on Cluster Computing and the Grid.

[12] Yichuan Jiang "A Survey of Task Allocation and Load Balancing in Distributed Systems". IEEE Transactions on Parallel and Distributed Systems 2015.

[13] Robson Eduardo De Grande, Mohammed Almulla and Azzedine Boukerche "Autonomous Configuration Scheme in a Distributed Load Balancing System for HLA-based Simulations".17th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications.