Performance Analysis of Load Balancing Algorithms in Distributed System

Deepika¹, Divya Wadhwa² and Nitin Kumar³

Banasthali University (M.TECH(I.T.))¹, Galgotias University (M.TECH(CSE))³
E-mail: deepikasharma.seo@gmail.com, divyawadhwa1991@gmail.com, nitinsharma.meerut@gmail.com

Abstract

Load balancing is the process of improving the performance of a parallel and distributed system through a redistribution of load among the processors. Load balancing algorithm tries to balance the total system load by transferring the workload from heavily loaded node to lightly loaded node to ensure the good overall system performance. The basic goal of this algorithm is to maximize the total system throughput. In this paper we present the performance analysis of various load balancing algorithms based on different parameters, considering two typical load balancing approaches static and dynamic. The analysis indicates that static and dynamic both types of algorithm can have advantages as well as disadvantages over each other. The main purpose of this paper is to help in design of new algorithms in future by studying the behavior of various existing algorithms.

Keywords: Load balancing, distributed system, throughput.

1. Introduction

In parallel and distributed systems more than one processor process parallel programs. The amount of processing time needed to execute all processes assigned to a processor is called workload of a processor [1]. The amount of processing time needed to execute all processes assigned to processor is called load of a processor. A distributed system provides the resource sharing as one of its major advantages. One of the biggest research issues in distributed computing systems is the development of effective techniques for distributing load/processes on multiple processors. The main goal is to distribute processes among processors to maximize throughput, maintain stability,
Deepika et al.

maximize resource utilization, minimize communication delays and fault tolerant.[2]. An important problem here is to decide how to achieve a balance in the load distribution between processors so that the computation is completed in the shortest possible time.

2. Load Balancing
Load balancing is the process of improving the performance of system through a redistribution of load among processors.

2.1 Types of Load Balancing Algorithms
Load balancing algorithms can have two categories based on initiation of process as follows:

2.1.1 Sender Initiated: In this type the load balancing algorithm is initialized by the sender. In this type of algorithm the sender sends request messages till it finds a receiver that can accept the load.

2.1.2 Receiver Initiated: In this type the load balancing algorithm is initiated by the receiver. In this type of description algorithms the receiver sends request messages till it finds a sender that can get the load[1].

3. Static Load Balancing
In static load balancing, the performance of the processors is determined at the beginning of execution. Then depending upon their performance the work load is assigned by the master processor. The slave processors calculate their allocated work and submit their result to the master. A task is always executed on the processor to which it is assigned that is static load balancing methods are non preemptive. The goal of static load balancing method is to reduce the execution time, minimizing the communication delays[1].

3.1 Round Robin Algorithm
Round Robin algorithm distributes jobs evenly to all slave processors. All jobs are assigned to slave processors based on Round Robin order, meaning that processor choosing is performed in series and will be back to the first processor if the last processor has been reached. Processors choosing are performed locally on each processor, independent of allocations of other processors.

3.2 Randomized Algorithm
Randomized algorithm [5] uses random numbers to choose slave processors. The slave processors are chosen randomly following random numbers generated based on a statistic distribution.
3.3 Central Manager Algorithm
Central processor will choose a slave processor to be assigned a job. The chosen slave processor is the processor having the least load. The central processor is able to gather all slave processors load information, thereof the choosing based on this algorithm are possible to be performed. The load manager makes load balancing decisions based on the system load information, allowing the best decision when of the process created.

3.4 Threshold Algorithm
In Threshold algorithm [6], the processes are assigned immediately upon creation to hosts. Hosts for new processes are selected locally without sending remote messages. Each processor keeps a private copy of the system’s load. The load of a processor can characterize by one of the three levels:

- Under loaded: load < t_under
- Medium: t_under ≤ load ≤ t_upper
- Overloaded: load > t_upper.

4. Dynamic Load Balancing
Unlike static algorithms, dynamic algorithms allocate processes dynamically when one of the processors becomes under loaded. Instead, they are buffered in the queue on the main host and allocated dynamically upon requests from remote hosts[8].

4.1 Central Queue
It stores new activities and unfulfilled requests as a cyclic FIFO queue on the main host. Each new activity arriving at the queue manager is inserted into the queue. Then, whenever a request for an activity is received by the queue manager, it removes the first activity from the queue and sends it to the requester. If there are no ready activities in the queue, the request is buffered, until a new activity is available. If a new activity arrives at the queue manager while there are unanswered requests in the queue, the first such request is removed from the queue and the new activity is assigned to it.

4.2 Local Queue
The basic idea of the local queue algorithm is static allocation of all new processes with process migration initiated by a host when its load falls under threshold limit, is a user-defined parameter of the algorithm. The parameter defines the minimal number of ready processes the load manager attempts to provide on each processor.

4.3 Policies or Strategies in dynamic load balancing
There are different policies in dynamic load balancing [10]:
1. Transfer Policy: The part of the dynamic load balancing algorithm which selects a job for transferring from a local node to a remote node is referred to as Transfer policy or Transfer strategy.
2. Selection Policy: It specifies the processors involved in the load exchange (processor matching).
3. **Location Policy**: The part of the load balancing algorithm which selects a destination node for a transferred task is referred to as location policy or Location strategy.

4. **Information Policy**: The part of the dynamic load balancing algorithm responsible for collecting information about the nodes in the system is referred to as Information policy or Information strategy.

5. **Load estimation policy**: which determines how to estimate the workload of a particular node of the system.

6. **Process transfer policy**, which determines whether to execute a process locally or remotely.

7. **Priority assignment policy**: which determines the priority of execution of local and remote processes at a particular node. **8. Migration limiting policy**: which determines the total number of times a process, can migrate from one node to another.

5. **Performance Parameters**
   The performance of various load balancing algorithms is measured by the following parameters.
   1. **Nature**: This factor is related with determining the nature or behavior of load balancing algorithms that is whether the load balancing algorithm is of static or dynamic nature, pre-planned or no planning.
   2. **Overload Rejection**: If Load Balancing is not possible additional overload, rejection measures are needed. Static load balancing algorithms incurs lesser overhead as once tasks are assigned to processors, no redistribution of tasks takes place, so no relocation overhead. Dynamic Load Balancing algorithms incur more overhead relatively as relocation of tasks takes place.
   3. **Reliability**: This factor is related with the reliability of algorithms in case of some machine failure occurs. Static load balancing algorithms are less reliable because no task/process will be transferred to another host in case a machine fails at run-time. Dynamic load balancing algorithms are more reliable as processes can be transferred to other machine in case of failure occurs.
   4. **Adaptability**: This factor is used to check whether the algorithm is adaptive to varying or changing situations. Static load balancing algorithms are not adaptive. Dynamic load balancing algorithms are adaptive towards every situation.
   5. **Stability**: Static load balancing algorithm considered as stable as no information regarding present workload state is passed among processors. However in case of dynamic load balancing such kind of information is exchanged among processors.
   6. **Predictability**: This factor is related with the deterministic or nondeterministic factor that is to predict the outcome of the algorithm. Static load balancing algorithm’s behavior is compile-time. Dynamic load balancing algorithm’s behavior is unpredictable.
7. **Forecasting Accuracy:** Forecasting is the degree of conformity of calculated results to its actual value that will be generated after execution.

8. **Cooperative:** This parameter gives that whether processors share information between them in making the process allocation decision other are not during execution. Static algorithms are cooperative and Dynamic algorithms are non cooperative.

9. **Resource Utilization:** Static load balancing algorithms have lesser resource utilization as static load balancing methods just tries to assign tasks to processors in order to achieve minimize response time ignoring the fact that may be using this task assignment can result into a situation in which some processors finish their work early and sit idle due to lack of work. Dynamic load balancing algorithms have relatively better resource utilization as dynamic load balancing take care of the fact that load should be equally distributed to processors so that no processors should sit idle.

10. **Process Migration:** Process migration parameter provides when does a system decide to export a process. The algorithm is capable to decide that it should make changes of load distribution during execution of process or not.

11. **Preemptiveness:** This factor is related with checking the fact that whether load balancing algorithms are inherently non-preemptive as no tasks are relocated. Dynamic load balancing algorithms are both preemptive and non preemptive.

12. **Response Time:** How much time a distributed system using a particular load balancing algorithm is taking to respond? Static load balancing algorithms have shorter response time. Dynamic load balancing algorithms may have relatively higher response time.

13. **Waiting Time:** Waiting Time is the sum of the periods spent waiting in the ready queue.

14. **Turnaround Time:** The interval from the time of submission of a process to the time of completion is the turnaround time.

15. **Throughput:** Throughput is the amount of data moved successfully from one place to another in a given time period.

16. **Processor Thrashing:** Processor thrashing occurs when most of the processors of the system are spending most of their time migrating processes without accomplishing any useful work in an attempt to properly schedule the processes for better performance. Static load balancing algorithms are free from Processor thrashing as no relocation of tasks place. Dynamic load balancing algorithms incurs substantial processor thrashing.

6. **Result**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Round Robin</th>
<th>Random</th>
<th>Local Queue</th>
<th>Central Queue</th>
<th>Central Manager</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature</td>
<td>Static</td>
<td>Static</td>
<td>Dynamic</td>
<td>Dynamic</td>
<td>Static</td>
<td>Static</td>
</tr>
<tr>
<td>Overload Rejection</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
7. Conclusion
Load balancing algorithms is totally dependent upon in which situations workload is assigned, during compile time or execution time. The above comparison shows that static load balancing algorithms are more stable than dynamic. But dynamic load balancing algorithms are always better than static as per as overload rejection, reliability, adaptability, cooperativeness, fault tolerant, resource utilization, response & waiting time and throughput is concert.
References


