MEASURING PERFORMANCE OF DYNAMIC LOAD BALANCING ALGORITHMS IN DISTRIBUTED COMPUTING APPLICATIONS

Priyesh Kanungo
Professor and Senior Systems Engineer (Computer Centre), School of Computer Science and Information Technology
Devi Ahilya University, Indore-452001, India

Abstract: In distributed applications, performance issues have become more critical due to proliferation of heterogeneous devices, large variety of communication medium and increased security concerns. This paper highlights the issues in performance measurement in Dynamic Load Balancing Algorithms (DLB) used for distributed scheduling. Various parameters used to measure the performance of scheduling algorithms have been described. The simulation model has been used illustrate performance issues associated with load balancing.

Keywords: Performance Measurement, Distributed Scheduling, Slowdown, Response Time, Processor Utilization, Dynamic Load Balancing.

I. INTRODUCTION

With the growing demand of resource intensive distributed computing applications, the need of using sophisticated performance improvement techniques has also increased. DLB is one of the techniques used extensively to improve scalability and overall throughput in distributed systems in the rapidly growing resource intensive distributed applications. It is responsible for task scheduling as well as monitoring load variation in the system. In such distributed applications, uneven process arrival may cause load imbalance, where some nodes are overloaded while some other nodes are idle [2, 7]. The fundamental aspect of load balancing in large clusters is that it needs to take into account many different parameters for driving process transfer decisions. DLB approach can create additional overhead in collecting system state, analysing the data, making load balancing decisions and transferring the processes from one node to another [8,10]. Thus, even though it is established that load balancing facility is necessary for improving the performance of a distributed system, the important issues of performance measurement parameters and quality of algorithm needs to be investigated further and are being considered in this paper.

II. PERFORMANCE MEASUREMENT

Load balancing facility improves the performance of the distributed system. Overall system performance can be measured by the following parameters [1, 3, 5]:

A. Mean Response Time
Performance of a load balancing algorithm can be measured by the response time. Response time is the time elapsed between start of execution of a process and its completion. To achieve the good response time, processes must be distributed evenly among the nodes using appropriate load balancing technique. Good response time also means that the processes don’t have to wait too much in the system. Response time can be computed as follows:

\[ T = F - A \]

Where,

- \( T \) is Response Time
- \( F \) is finish time of current CPU burst
- \( A \) is start time of current CPU burst

Note that a process arrives and executed several times on the central processing unit during its lifetime.

B. Processor Utilization
Utilization of processor means the percentage of time for which the node is busy in running processes. This index is useful at lower load conditions. At the higher load
conditions, even after maximum utilization of the processor, some of the processes are waiting for execution. These processes can’t be taken into account for measuring load.

C. Mean Slow Down
Sometimes response time or waiting time cannot give correct idea about the suffering of processes, particularly when there is huge variation in their processing times. Slowdown or penalty ratio can be used to measure proportionate suffering of processes in the system irrespective of whether the process is long or short. Slowdown of a process is defined as the ratio of total time spent by the process in the system to the execution time of the process and is defined as:

\[ P = \frac{(t + w + m)}{t} \]

where,
\( P \) is mean slowdown or penalty ratio
\( t \) is execution time
\( w \) is missed or waiting time of a process in queue
\( m \) is migration time

III. DESCRIPTION OF THE ALGORITHM

DLB algorithms can be defined by their implementation of policies for load estimation, process transfer, status information exchange, node allocation and process migration. Load estimation policy specifies what workload information is to be collected, when it is to be collected and from where the information is to be gathered and what load indices will be used e.g. queue length, execution time or process age. Process transfer policy detects if the load imbalance conditions are prevailing and decides appropriate period of triggering the load balancing operation. State information exchange policy is necessary for exchange of load information among the nodes in the system to identify the nodes, which are either overloaded or under-loaded. Polling, broadcasting or on-demand techniques may be used to exchange state information. Broadcasting technique may lead to increased network traffic whereas other techniques allow selected information exchange [4, 6]. Node Allocation policy is needed to define the processes on an overloaded source node and to select an under-loaded destination node, where these processes will be migrated. Bidding, threshold, shortest or pairing technique may be used to decide a destination node. Process migration technique is required to actually transfer processes from source node to the destination node [9]. The formal algorithm for comparing various load indices and performance parameters is formally described below:

**Algorithm** performance-measurement;
/*Algorithm for measuring performance in DLB. Following load indices have been used: 1=No load balancing, 2 = Queue Length, 3=Process Age, 4= Execution Time.*/
{
    store threshold, avg-queue-length, avg-age, avg-exec-time of the system

    for each node \( P \) in the system
    store its ready-queue, fptr, rptr, load-level, mean-response-time, mean-utilization, mean-slowdown, mean-queue-length, mean-age, mean-exec-time;

    /* load level may be heavily-loaded, moderately-loaded or lightly-loaded */

    for each process in the system
    store its creation-node, PID, arr-time, ser-time, response-ratio, dep-time;
    /* new processes arrive at \( P \) */

    CreateProcessorQueue (struct processes ());

    while (true) do
        compute-threshold (\( T \));

        for every processor \( P \) in the DCS /*at every node concurrently*/
        {
            calculate-load (\( W, P \));

            if \( W > T \) then
                load-level=heavily-loaded;
            else
                load-level= lightly-loaded;

            while (\( P = \) heavily-loaded) do
            {
                choose-shortest-dest-node(dest-node);
                if (destination-node-found)
                {
                    select (newprocess);
                    migrate-process (newprocess, P.dest-node);
                    execute (newprocess, dest-node);
                }
            }

        }

        calculate-load (\( W, P \))
        {
            if load-index = qlength
\[ W = \text{sum of queue lengths on processor}; \]
\[ \text{if load-index is process-age} \]
\[ W = \text{sum of age of processes on processor queue}; \]
\[ \text{if load-index is execution-time} \]
\[ W = \text{sum of execution times of processes on processor queue}; \]
\}
\[ \text{compute-threshold (load-level);} \]
\[ \{ \]
\[ \text{if load-index = qlength} \]
\[ T = \text{avg-queue-length of processors}; \]
\[ \} \]
\[ \text{if load-index = process-age} \]
\[ T = \text{avg-age of the processes}; \]
\[ \} \]
\[ \text{if load-index = exec-time} \]
\[ T = \text{avg-exec-time of the processes}; \]
\} \]
\} \]
\}
\] End of Algorithm

A. Simulation and Result Discussion

The simulator was designed and implemented to evaluate load indices for different performance measure parameter viz. Processor utilization and mean slowdown. The simulator used artificial workload to carry out comparisons as this provides greater flexibility as compared to real workload. We assumed random process arrival and random service time distribution. The load balancer consists of server module that collects load information and makes job placements and migration module for remote execution of processes. Table I, Table II, Fig. 1 and Fig. 2 compare the results of simulation.

**TABLE I**
COMPUTATION OF PROCESSOR UTILIZATION USING DIFFERENT LOAD INDICES

<table>
<thead>
<tr>
<th>Node Id.</th>
<th>Without DLB</th>
<th>Queue Length</th>
<th>Process Age</th>
<th>Exec. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>35</td>
<td>56</td>
<td>68</td>
<td>79</td>
</tr>
<tr>
<td>5</td>
<td>42</td>
<td>61</td>
<td>70</td>
<td>82</td>
</tr>
<tr>
<td>4</td>
<td>51</td>
<td>65</td>
<td>71</td>
<td>83</td>
</tr>
<tr>
<td>7</td>
<td>59</td>
<td>69</td>
<td>77</td>
<td>87</td>
</tr>
<tr>
<td>8</td>
<td>65</td>
<td>74</td>
<td>81</td>
<td>89</td>
</tr>
<tr>
<td>6</td>
<td>71</td>
<td>77</td>
<td>83</td>
<td>90</td>
</tr>
<tr>
<td>2</td>
<td>79</td>
<td>83</td>
<td>87</td>
<td>92</td>
</tr>
<tr>
<td>1</td>
<td>95</td>
<td>85</td>
<td>89</td>
<td>93</td>
</tr>
</tbody>
</table>

**TABLE II**
COMPUTATION OF MEAN SLOWDOWN USING DIFFERENT LOAD INDICES

<table>
<thead>
<tr>
<th>Node Id.</th>
<th>Mean Slowdown Time of Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Without DLB</td>
</tr>
<tr>
<td>3</td>
<td>1.4</td>
</tr>
<tr>
<td>5</td>
<td>1.6</td>
</tr>
<tr>
<td>4</td>
<td>2.0</td>
</tr>
<tr>
<td>7</td>
<td>2.3</td>
</tr>
<tr>
<td>8</td>
<td>2.7</td>
</tr>
<tr>
<td>6</td>
<td>3.8</td>
</tr>
<tr>
<td>2</td>
<td>4.2</td>
</tr>
<tr>
<td>1</td>
<td>5.6</td>
</tr>
</tbody>
</table>

The graph in Figure 1 compares the three load indices on the basis of mean response time of the processes. Figure 1 compares the different load indices on the basis of processor utilization on different nodes. Figure 2 compares the load indices on the basis of mean slowdown of the
load indices is better than no load balancing at all. Among the three load indices, execution time as a load index gives better results. But we know that it is difficult to estimate execution time of the processes before actually executing them. However, it works as standard to compare other implementable algorithms. Process age as load indices gives better results than queue length.

IV. CONCLUSION
In this paper, we have investigated various issues in DLB and studied various parameters for effective performance measurement in a computing cluster with respect to different load indices. We have also compared various indices used to measure the load on the nodes using different performance parameters.

REFERENCES