Proposed An Improved Local Hierarchical Load Balancing Algorithm (ILHLBA) in Distributed Computing
7.1 Introduction

A distributed computing system (DCS) is a set of autonomous computing systems (nodes) connected through any network. DCS is heterogeneous in nature due to the several aspects like computing power, memory storage, and network bandwidth and arrival pattern of tasks to the different nodes. The prime objective of the DCS is to share the resources in a better way available in the network. So one of the best way to utilize and to share the network resources like files, printers and CPUs is through the load balancing among the processors to maximize resource utilization. In general certain processors may have more numbers of tasks and some others may have less numbers of tasks or even idle causing disparity of job distribution on different processors in DCS resulting low system performance. Most of the time it is noticed in networks that at the peak time the fastest nodes are idle or lightly loaded and the slower nodes are heavily loaded [79]. It is also observed that two third nodes remain idle in the peak hour of computing. The jobs that are fed with the overloaded nodes, will have to wait longer to get processor’s attention as there must be more numbers of preceding jobs. Thus overloaded nodes would produce high average response time. Even an application of simple load balancing algorithm produces better performance than no load balancing algorithm applied in a network [57, 149, 158]. So the load balancing concept comes into existence by transferring excess load from the heavily loaded nodes to the lightly loaded nodes so that the load on each node becomes approximately the same. The load balancing algorithms improve the overall performance of the system by exploiting the maximum power of the processors and minimizing the average response time and hence maximize the resource utilization.

The dynamic load balancing approach has three most important policies: transfer policy, information policy and location policy. Transfer policy decides depending upon some predefined value whether a job would be executed locally or remotely. In selection policy the load balancing node selects a suitable node for transferring the selected job depending upon the information collected by state information policy. Both sender initiated and receiver initiated approaches fall under the location policy.

There are two fundamental approaches to the load balancing algorithm design. In static load balancing design approach the tasks are assigned on the basis of a priori knowledge of the system and once the tasks are allocated on the nodes do not change
The performance of the static load balancing algorithms depends on the prior information about the tasks and the system. The decision to transfer the tasks does not depend on the system state change. So this approach is best suited for homogeneous distributed computing system. But the dynamic load balancing algorithms take the decision to transfer the tasks depending on the current state of the system. The tasks are transferred from heavily loaded node to the lightly loaded node. So the quality of dynamic load balancing algorithms depends on the collection of information on load on different nodes in the system. So this approach is best suited for heterogeneous distributed computing system.

In dynamic load balancing the information may be collected either by centralized or distributed approach. In centralized approach the information is collected by a specially designed central node and in distributed approach each node has the autonomy to collect the information about the load of the system. It has been reported that the collection of information by centralized approach about the system state does not cause any performance degradation for a reasonably large distributed computing systems. The drawback of this approach is that the performance of a globally distributed system would be very poor and the cost of state information collection would be too much, and maintaining a huge information by a single node will surely cause a performance degradation. In the distributed information collection policy the information is collected either by sender initiative or receiver initiative algorithm. In sender initiative approach the heavily loaded nodes search for lightly loaded nodes for transferring extra load and the receiver initiative approach is the converse of sender initiated approach. In this approach either a sender or a receiver may poll all the nodes in a network for load balancing causing huge overheads. To reduce the overheads the sender or receiver nodes poll a selected number of nodes like nearest neighbors. Another performance problem with this approach is associated with the inter-arrival times and service times.

Another type of load balancing exists which is widely known as hierarchical load balancing approach. This approach syndicates the merits of both centralized and decentralized approaches. In this approach the disadvantages of centralized approach and decentralized approach are minimized.
7.2 Hierarchical Load Balancing Strategy

In decentralized load balancing approach the sender or receiver nodes poll a number of nodes to send/receive the load while in centralized approach a single node is responsible for maintaining the state information of all the nodes in a network and every node either sender/receiver sends/gets load after consulting the central node. Decentralized load balancing strategy is successful for large distributed computing but centralized load balancing approach is the best strategy for a small network but it would very costly over a globally distributed network [150]. It has been shown that the centralized strategy is best suited for a cluster of nodes in a large distributed system [151]. Thus the concept of hierarchical load balancing strategy brought a remarkable improvement in load balancing strategies in distributed computing. In this approach it is very easy to implement both the advantages of centralized and decentralized strategies together due to its hierarchical structure.

In hierarchical approach a set of specially designated nodes is responsible for maintaining the state information of a set of nodes in below of their hierarchy. The system information is maintained by all sub tree nodes and as we move towards the root node more information would be maintained by the corresponding sub tree root nodes and the root node would maintain the global state information. Fig.7.1 illustrates the hierarchical organization of eight nodes.

The hierarchical topology is chosen for load balancing for the following advantages: hierarchical network is easier to expand; It is easy to manage and maintain the network because the whole network is divided into small clusters (segments) and error detection and correction is also easy and if one cluster is damaged, other cluster will continue to work.

In general the nodes are considered as only router except the leaf nodes in a hierarchical distributed network system [53, 141]. The leaf nodes are considered to be the computing processor and rest are as manager (coordinator) nodes responsible for load balancing only. But intermediate nodes with finite buffer capacity also have been utilized as front-end node which is responsible for both load balancing and computing purpose [152].
7.3 Proposed Hierarchical Model

In local hierarchical load balancing (LHLBA) algorithm [53] each sub tree root node maintains only arithmetic sum of the state information which means that only summary state of nodes under any sub tree root node maintains the state information and hence the root maintains the summary state of all nodes in a whole network. In this policy load is distributed locally first if any sender is available there but if sender is not available the request for sender goes up to root node and still sender is not found then a message “no job” is sent back to the sender node.

In local hierarchical algorithm the status of an overloaded node is taken as +1 and an under load node is taken as -1 and moderate loaded node is taken as 0 by sub tree root nodes. And the sum of +1 and -1 is taken as zero which implies that the sub tree is in balanced state. But this situation only arises when the amount of the load deficiency at one node is equal to extra load at another node under a sub tree which is one situation out of three possibilities. Fig. 7.2 illustrate the three possible states of a sub tree node at any moment of time. So we make an improvement here that we cannot say a node is in moderate state if one of its children is in +1 state and another is in -1 state.

![Figure 7.1 A Hierarchical Topology with Eight Nodes](image-url)
Figure 7.2 Situations after local load balancing a) both nodes either over loaded or one is over loaded and another is moderate loaded, b) both nodes are moderate loaded c) both are under loaded or one is under loaded and other is moderate loaded.

In local hierarchical algorithm request message for job by a receiver for a sender reaches up to root node when there would not be any sender present in that sub tree. And if no sender is found a message “no job” would be transferred back to the corresponding receiver and if the sender is no longer remains as sender at the time of delivery of the request message for a job then, a “false sender” message would be sent to the corresponding receiver.

We have proposed an improved version of local hierarchical load balancing algorithm (ILHLBA) which minimizes the problems encountered in LHLBA. In our ILHLBA approach the sub tree node only maintains the overall status of a sub tree 0, +1 or -1. For a lightly loaded system if the left sub tree does not have any sender, the request for a job by a receiver will reach up to the root and at the same time if the right sub tree also does not have any sender, a lot of requests will reach to the root at the same time and thus will cause a huge gathering of messages to the root resulting low performance due to the congestion on the links R0R1 and R0R2 and also due to the “no job” message to each receiver. These two problems have been minimized by exchanging the latest status of R1 and R2 by R0. Suppose that R1 and R2 are in -1 state and this status is being exchanged between them by R0 whenever there is a change of status. Thus R1 will not send any job request message to R0 and hence R0 will not have to manage all the requests coming from both sub trees. For a large
number of nodes suppose $2^{100}$ nodes, the root in LHLBA will have to deal with $2^{99}$ messages when 50% of the nodes are in receiving mode and the whole tree does not have any sender in worst case. But in ILHLBA, root only will have to deal just few messages.

7.4 Simulation and Results

The proposed load-balancing algorithm ILHLBA is compared with LHLBA. The simulation framework has been designed with the following setup. We considered a hierarchical topology with eight number of computing nodes. The connectivity of all the nodes is ensured. The comparison has been carried out in terms of response time and throughput for the varying loads. Load Balancing Simulator in Java is used for this purpose for testing the behavior of different load balancing algorithms under the same conditions and evaluates the behavior of the algorithms for different arrival patterns.

Fig. 7.3 and Fig. 7.4 represent the response time for different system utilization for 25% and 50% of receiving nodes present in the system respectively. Both Fig. 7.5 and Fig. 7.6 also compare the throughput between LHLBA and ILHLBA against system utilization. Our proposed algorithm ILHLBA algorithm is better than LHLBA in a lightly loaded system having 25% receiving nodes by 29.8% and for 50% receiving nodes by 25.7% in respect of response time respectively. And ILHLBA algorithm is better than LHLBA in a lightly loaded system having 25% receiving nodes by 17.8% and for 50% receiving nodes by 14.1% in respect of throughput respectively.
Chapter 7: Proposed An Improved Local Hierarchical Load Balancing Algorithm (ILHLBA) in Distributed Computing

Figure 7.3 Response Time vs System Utilization at 25% of Receiving Nodes

Figure 7.4 Response Time vs System Utilization at 50% of Receiving Nodes
Chapter 7: Proposed An Improved Local Hierarchical Load Balancing Algorithm (ILHLBA) in Distributed Computing

![Graph](image)

**Figure 7.5** Throughput vs System Utilization at 25% of Receiving Nodes

![Graph](image)

**Figure 7.6** Throughput vs System Utilization at 50% of Receiving Nodes

### 7.5 Summary

In distributed approach either a sender or a receiver may poll all the nodes in a network for load balancing causing huge overheads. Hierarchical load balancing approach syndicates the merits of both centralized and decentralized approach by removing disadvantages of centralized and decentralized approaches. The hierarchical topology is chosen for load balancing for the following advantages: hierarchical