The Journal of Systems and Software 107 (2015) 187–203

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

A small world based overlay network for improving dynamic load-balancing

Eman Yasser Daraghmi, Shyan-Ming Yuan*

DCS Lab, Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan, ROC

ARTICLE INFO

Article history:
Received 5 January 2015
Revised 25 May 2015
Accepted 1 June 2015
Available online 9 June 2015

Keywords:
Diffusion
Distributed systems
Dynamic load-balancing

ABSTRACT

Load-balancing algorithms play a key role in improving the performance of distributed-computing-systems that consist of heterogeneous nodes with different capacities. The performance of load-balancing algorithms and its convergence-rate deteriorate as the number-of-nodes in the system, the network-diameter, and the communication-overhead increase. Moreover, the load-balancing technical-factors significantly affect the performance of rebalancing the load among nodes. Therefore, we propose an approach that improves the performance of load-balancing algorithms by considering the load-balancing technical-factors and the structure of the network that executes the algorithm. We present the design of an overlay network, namely, functional small world (FSW) that facilitates efficient load-balancing in heterogeneous systems. The FSW achieves the efficiency by reducing the number-of-nodes that exchange their information, decreasing the network diameter, minimizing the communication-overhead, and decreasing the time-delay results from the tasks re-migration process. We propose an improved load-balancing algorithm that will be effectively executed within the constructed FSW, where nodes consider the capacity and calculate the average effective-load. We compared our approach with two significant diffusion methods presented in the literature. The simulation results indicate that our approach considerably outperformed the original neighborhood approach and the nearest neighbor approach in terms of response time, throughput, communication overhead, and movements cost.

1. Introduction

Load-balancing algorithms have become increasingly popular and powerful techniques in modern distributed computing systems in recent years (Chang et al., 2014). They provide opportunities for increasing the performance of large-scale computing systems and applications since they are designed to redistribute the workloads over the components of the distributed system in a way that ensures expanding resource utilization, maximizing throughput, minimizing response time, and avoiding the overload situation (Abdelmaboud et al., 2014). To achieve the goal of maximum performance, it is prerequisite to smoothly spread the load among the nodes to avoid, if possible, the situation where one node is heavily loaded with excess of workloads while another node is lightly loaded or idle (Chwa et al., 2015; Luo et al., 2014).

Load-balancing algorithms can be categorized into either static or dynamic (Cybenko, 1989; Fang and Wang, 2009). Static load-balancing necessitates complete information of the entire distributed system and workloads information, whereas dynamic load balancing requires light assumption about the system or the workloads. As in practical applications (i.e. real world networks) the workloads are generally not completely known, and each node has different capacity and runs at different speed, it is more efficient to employ the dynamic load balancing algorithms for practical applications. The diffusion approach (Hu and Blake, 1999; Luque et al., 1995) is one of the dynamic load balancing techniques that have received much attention by researchers in the past decades to solve the load-balancing problem. In standard diffusion approach, a system which has different nodes exchanges workloads via the communication links between these nodes. The workloads are distributed among the nodes, and the load balancing process works in sequential rounds. In every round, each node is allowed to balance its load with all its neighbors by exchanging the workloads to balance the total system load globally, meaning to minimize the load difference between the nodes with minimum and maximum load. The nearest-neighbor approach (Tada, 2011) is another dynamic technique that allows the nodes to communicate and migrate the excess workloads with their immediate neighbors only. Each node balances the workload among its neighbors in the hope that after a number of iterations the entire system will approach the balanced state.

* Corresponding author. Tel.: +886 3 5715900; fax: +886 3 5721490.
E-mail addresses: eman.yasser85@gmail.com (E.Y. Daraghmi), smyuan@cs.nctu.edu.tw, smyuan@gmail.com (S.-M. Yuan).

http://dx.doi.org/10.1016/j.jss.2015.06.001
0164-1212/

Since load-balancing algorithms play an important role of improving the performance of practical distributed computing systems, researchers have been motivated to propose several dynamic algorithms for balancing the workloads among nodes. However, dynamic load-balancing algorithms still present fundamental challenges when being executed at large-scale heterogeneous distributed systems. Previous research (Hui and Chanson, 1999, 1996, 1997) concluded that three **structural factors**, which refer to the structure of the network that executes the load-balancing algorithm, decrease the performance of any load-balancing algorithm as well as affect the algorithm convergence rate. The factors are: (1) increasing the number of nodes in the system (i.e. the number of the nodes that exchange their workload information); (2) increasing the network diameter which is defined as the longest shortest path between any two nodes of the network; (3) increasing the communication overheads or the communication delays among the nodes. These factors, from one hand, make it not feasible for a node to collect the load-information of all other nodes in the system. Moreover, even if a node collects the load-information of all other nodes in the system, this information will be not up to date when it is used (i.e. old information may not reflect the current load of a node) as more communication delays make this information old and thus the task of balancing the load is significantly damaged. From the other hand, it is intuitive that a network with longer diameter will take longer time to converge as the number of iterations to propagate the workloads to all nodes is proportional to the network diameter. *Therefore, the first objective of this research aims at improving the performance of load-balancing algorithms by considering the structural factors of the network that executes the algorithm.*

In addition, previous studies concluded that (Zomaya et al., 2001) **technical load-balancing factors**, which refer to the algorithm policies that should be considered when designing a load-balancing algorithm, such as the load migration rule, affect the performance of load-balancing algorithm. Therefore, these studies propose improved algorithms that consider these factors to enhance the performance of load-balancing (i.e. improvements include: the derivation of a faster algorithm that transfers less workloads to achieve a balanced state than other algorithms, or a mechanism for selecting and transferring the workloads to other nodes). However, when applying a dynamic load-balancing to practical distributed system, the functionality of the node and the migrated task must be checked to ensure that the node can process that received task. Thus, if the nodes distributed randomly, some situations that affect the performance of the load-balancing algorithm negatively may occur. For instance, n_i is a node in a practical distributed system. Since n_i is overloaded, it migrates a task to another lightly loaded node n_j. When n_j receives the migrated task, the load-balancing algorithm runs at n_j checks the scope of services of node n_j to ensure that the task can be processed by n_j. Thus, if the migrated task is out of n_j services scope, then the task will be migrated again to another node. Moreover, the task may be retuned again to n_i. Practically, re-migrating the task to another node decreases the performance of load-balancing algorithms because of the task re-migration time delay. Increasing the number of re-migrating task increases the time delay and thus decreases the performance of load-balancing. *Therefore, the second objective of this research aims at improving the performance of load-balancing algorithms by decreasing the time delay results from re-migrating tasks (i.e. re-migrating tasks results from the node out of services scope). To achieve our goal, we construct the FSW to allow nodes migrate tasks to other nodes that have similar services scope.*

In this research, we aim at improving the performance of load balancing algorithm by considering both the structural and the technical load-balancing factors. We also consider the node services scope to decrease the negative effect of tasks re-migration process. To achieve our goal, we propose a two-stage approach that, first, designs an overlay network which employs both the concept of small world network and the node services scope, and then, proposes an improving load-balancing to be applied within the overlay network.

First, practically, the nodes of practical distributed systems execute various computational-functions (each node has services scope). These computational-functions can be easily derived from the role of a node within the system and identified by k-element set (i.e. the role of the node within the system refers to the node services scope), namely the functional set (FS). Each element in the set represents a particular function that can be executed within the system. The FS of a node can be mapped to a point in a cluster and thus can be seen as a point in that cluster. In real-world distributed systems, each node plays a key role within the system. For instance, the m-cafeteria recommendation system is a practical distributed system that consists of several cafeteria nodes. Each cafeteria serves a menu, set of meal (i.e. the menu is considered as the FS of a cafeteria node, FS = {serving orange juice, serving butter waffle, etc.}). A user can via his/her mobile phone request a meal from a cafeteria node, if a cafeteria node is overloaded, then the request will be migrated to another node that has similar functionality. Similar functionality is defined as the difference between the amount of functions in-common among nodes and the amount of functions unique to nodes. It is clear that functions in common increase similarity, whereas functions that are unique to one node decrease similarity.

In fact, a small world (SW) network has a small average path length and large cluster coefficient properties. Thus, constructing an overlay network that satisfies the small world network properties and considers functional similarity minimizes the negative effects of the structural and technical factors (i.e. 1. decrease the number of nodes that exchange the workloads information, 2. minimize the network diameter, 3. deteriorate the communication overhead, and 4. decrease the impact of out of services scope and thus decrease the time delay results from re-migrating tasks). In this research, we construct an overlay network based on the small world principle, namely, the functional small world (FSW) that supports efficient load-balancing and thus increasing the performance of distributed computing systems.

Second, this research also presents an efficient load-balancing algorithm that considers the capacity of each node and the load-balancing technical factors, such as the initialization rule, the information exchange rule, the load-measurement rule and the load-migration rule.

Precisely, the advantages of creating the FSW instead of randomly distribute the nodes into clusters are: (1) simulating real world heterogeneous distributing systems which facilitate applying the load-balancing algorithm to real world distributed system; (2) decreasing the effect of time delay results from task re-migration that occurs because of the node out of services scope.

In summary, this paper presents the design of the FSW overlay network to support efficient dynamic load-balancing in heterogeneous systems. The primary contribution of this work is fourfold:

1. We adopt an effective clustering strategy that places nodes in clusters based on the nodes functional similarity and satisfies the properties of the small world principle.
2. We show a way of building a functional small world (FSW) overlay network that supports dynamic load-balancing, which is scalable to large network sizes yet adapts to dynamic membership and content changes. For simplicity, we refer to the functional small world overlay network as FSW in the rest of this paper.
3. We propose an efficient and improved dynamic diffusion load-balancing algorithm to be executed in the constructed FSW.
4. We conduct extensive experiments to evaluate the performance of proposed solution on various aspects, including throughput, response time, communication overhead and movements cost.