A Minor Project Report

on

Prevention of Shilling attacks in Recommender systems using clustering algorithms

Submitted by

Abraham G Sebastian
13IT233
Raghav
13IT
Sagar R Alavandar
13IT233
V Sem B.Tech (IT)

in partial fulfillment for the award of the degree
of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

At

Department of Information Technology
National Institute of Technology Karnataka, Surathkal.
July 2015

CERTIFICATE

This is to certify that the project entitled "Prevention of Shilling attacks in Recommender systems using clustering algorithms" is a bonafide work carried out as part of the course Advanced Web Technologies (IT702), under my guidance by Abraham G Sebastian, Raghav, Sagar R Alavandar, students of III Sem B.Tech(IT) at the Department of Information Technology, National Institute of Technology Karnataka, Surathkal, during the academic year 2015-16, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, at NITK Surathkal.

Place: Mangalore

_____

(Signature of the Guide)

Date: 17-08-2015

DECLARATION

I hereby declare that the project entitled "Nearest Neighbor Task Allocation for Large-scale Distributed Systems", submitted by me for the course IT891 as part of the partial course requirements for the award of the degree of Master of Technology in Information Technology at NITK Surathkal is my original work. I declare that the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles elsewhere.

_____

(Signature of the Student)

Place: Mangalore
Date: 17-08-2015

Abstract

Distributed Systems are becoming very much popular these days in order to meet increasing computing demand. To make efficient use of distributed systems, allocation of tasks to each computing node should be appropriate. This can be done using current load information of nodes. But in reality, load information is not up to date while allocating tasks. Load information can be updated in two ways, either periodically or when load of a node is changed. In latter case, it may be outdated due to communication delay. The performance of task allocation suffers because of old load information. To deal with this problem, Nearest Neighbor Task Allocation Scheme is introduced here. The effect of communication delay is reduced because communication is restricted only between the neighboring nodes. The proposed approach shows better performance when compared to the existing one.

Contents

List of Figures

List of Tables

1 Introduction

With increased computing demand, the distributed systems are also growing faster. For using these distributed systems efficiently, tasks allocation to each node in the network should be appropriate. If a proper task allocation scheme is not used, few nodes may be highly loaded while others may be idle. Waiting time of the tasks should also be less. To tackle such problems, tasks arriving at the heavily loaded nodes need to be transferred to the nodes which are lightly loaded.

The classification of task allocation schemes can be done in two perspectives: whether it is centralized or distributed and on the basis of load information available at each node. In centralized approach, one specific node will decide everything about allocating tasks. But this approach is only suited for small systems. In distributed approach, all the nodes present in the system contribute to perform load balancing operations. It provides the most important characteristics required for the distributed systems i.e., scalability and high fault tolerance. On the basis of load information of remote nodes also, there are two possibilities. If the load information of other nodes is not available, the decision about transfer of tasks is made on the basis of its own load. A random destination node is selected for transfer of task. With availability of remote information, the lightest remote node will be selected as a destination for transferring the task.

However, using information about load for decision making is not an easy task. Because in practical, load information is outdated. In cases of periodic updation, it remains old till next update. If updated on status change of a node, then it remains old due to communication delay. The idea of this work is to keep load information fresh so that the bad effect of outdated load information can be reduced. Nearest neighbor is the approach used for the same. Communication happens only between neighboring nodes and whenever load of some node is being changed, all its neighbors are immediately updated. The performance of this approach is tested comparing to the previous approaches which are using randomness.

2 Literature Survey

2.1 Background

The task allocation schemes show bad performance because of old load information. So these schemes should have some mechanism which reduces the effect of old load information. The previous approaches are based upon randomness. They focus on choosing a random node for allocating a task when old information is growing. One such strategy is Random Subset which is explained as follows:

- Location strategy: Initially, a subset of nodes are selected randomly by the local node. Then the node which is having lightest load is selected. The subset of nodes is taken so that only limited number of nodes need to be searched for selecting the destination node. This is a tuning parameter in the algorithm and tries to reduce the effect of outdated load information.
- Information strategy: In Random Subset, each and every node doesn't store the load related information of the all the nodes. The load information is collected on demand whenever it is required. For selecting the final destination to transfer the task, all the nodes present in the subset of local node are probed by it.
- Transfer strategy: Each and every time a task is arrived, the local node tries to determine on which node it should be allocated. If there is some node found on probing which has lesser load than others and that of local node, the task is transferred to that node. If it's not found, task is executed by local node itself.

2.2 Outcome of Literature Survey

Our objective is to provide a task allocation scheme which is good enough to reduce effect of old load related information and performs well in context of waiting time and response time of a task. We'll compare this approach with previously existing one and analyze the results with respect to response time, throughput etc.

2.3 Problem Statement

To propose a task allocation scheme that provides load balancing among the nodes and also reduces waiting time of task before being assigned to a node for processing.

3 Proposed approach

3.1 Nearest Neighbor

Nearest neighbor is distributed task allocation scheme which uses load information of nodes before allocating tasks. In Nearest neighbor approach, every node can communicate only to its neighbors. This is done in order to reduce the communication delay. This approach can be explained as follows with respect to three strategies:

1. Location strategy: The destination node for transferring the task is selected among the subset of nodes called neighbors. The node with the lightest node among the neighbors is selected as destination node.

2. Information strategy: Every node is aware of load information of its neighbors. This information is updated whenever the load of a particular node changes. It can change when some new task arrives and it is put in the queue for execution or when a task has finished executing. Receiver node updates its load related information after getting this message.

3. Transfer strategy: Transfer strategy of nearest neighbor is same as that of Random subset. Whenever some new task is arrived, local node will try to determine appropriate destination for this. If it is not found, task will be executed by local node itself.

3.2 System Model and Assumptions

Distributed system consists of many number of nodes which communicate with each other. Some communication delay is introduced due to their communication. Each node is able to process one task at a time. The task which is being executed can't be interrupted in between. Each node is having one run queue which contains task already allocated to the node but can't be executed immediately. The definition of load on a node is the length of its run queue. Length means number of tasks present in the queue. The size of each task is determined as the amount of calculation required for its processing.

Following assumptions are made:

- All tasks are independent of each other. On task's execution makes no effect on other.

- Node doesn't have information about task such as burst time of task etc.

- The communication between nodes make no effect on the execution of tasks. That means communication delay doesn't reduce execution speed of task.

4 Methodology

The figure given below shows the topology used to analyze the behavior of nearest neighbor and random subset schemes. All the nodes used are of same configuration. The burst time of each task is proportional to its size and is same for each task. Neighbors of a node are the nodes which are directly connected to it.

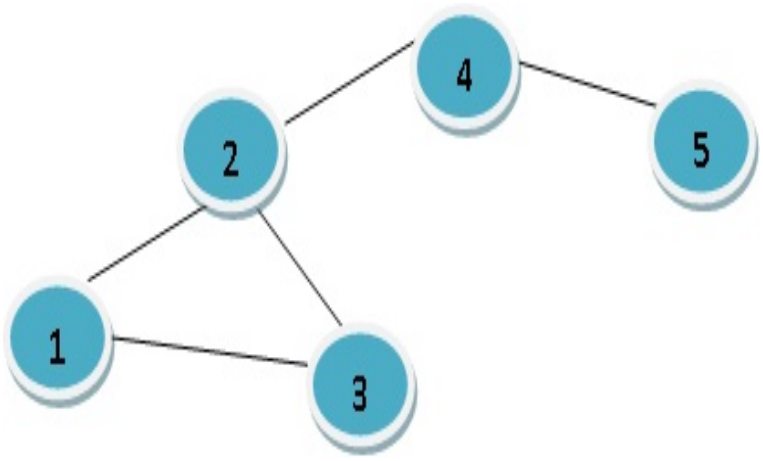

Figure 1: Nearest neighbor topology

- Arrival of tasks at the nodes is based on poisson process.
- Each task arrived is sent to randomly selected nodes initially. Then in accordance with task allocation scheme used, they are executed locally or transferred to some other node.
- In case of random subset scheme, The number of nodes present in subset is taken as 2.
- The communication delay between a node and its neighboring nodes is fixed to a certain value(10ms).
- Mean response time and waiting time are taken as performance metrics.

  Response time = Task completion time – Arrival time

  Waiting time = Response time – Service time
- Simulations are performed for 1000ms.

Flowcharts

The flowchart of random subset approach is shown below:

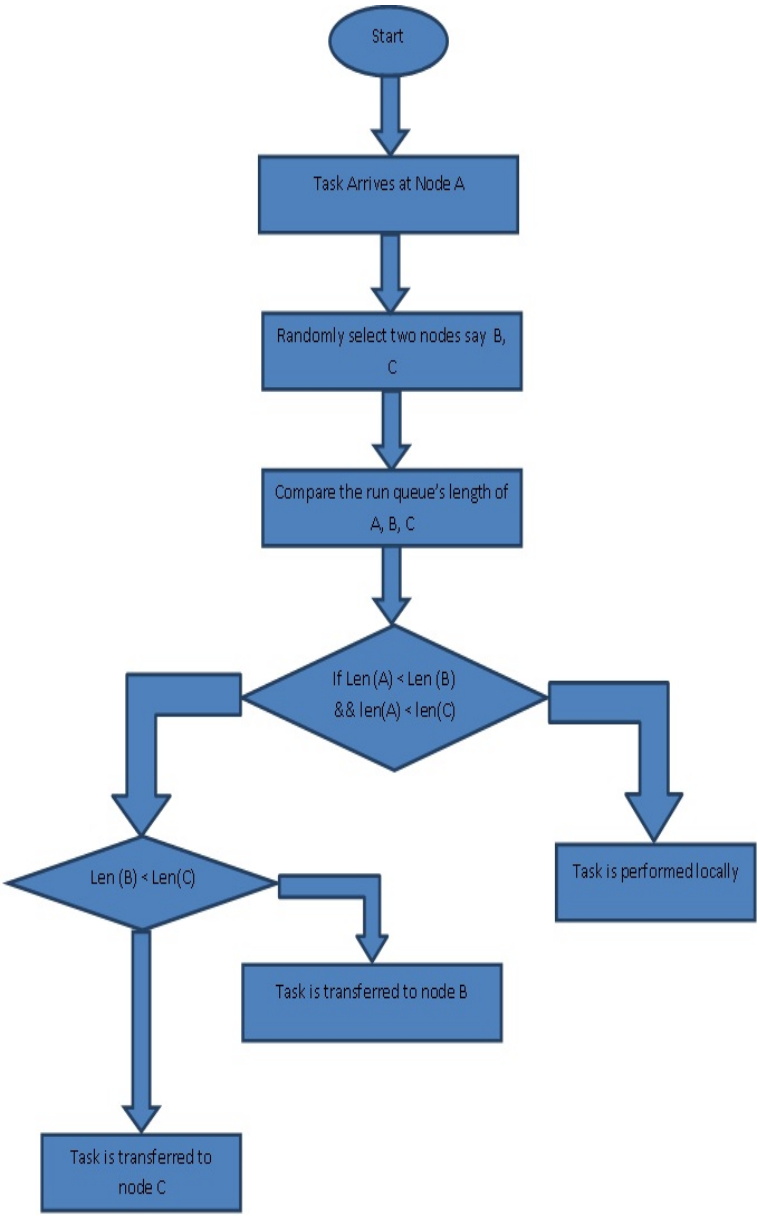

Figure 2: Random Subset Approach

The flowchart of Nearest Neighbor's approach is shown below:

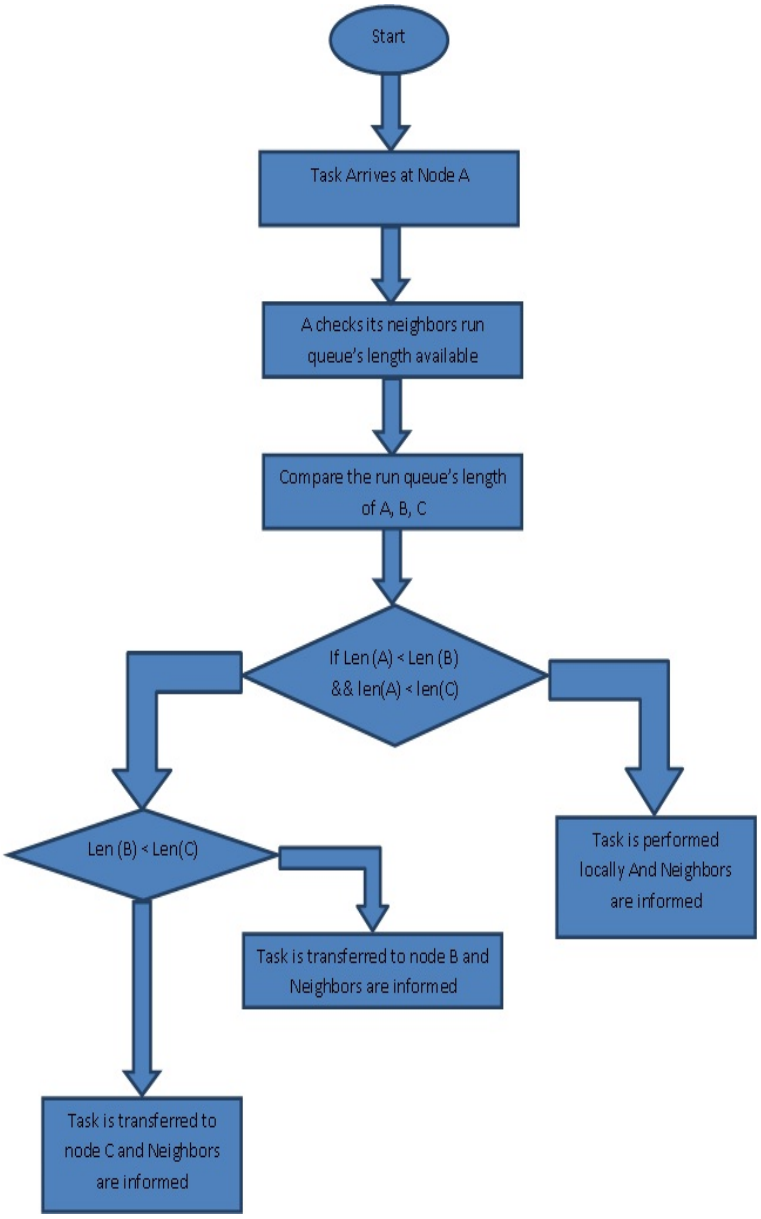

Figure 3: Nearest Neighbor Approach

5 Implementation

5.1 Work Done

Five nodes of same configuration are taken for performing simulation. At first, random subset algorithm is executed on each system parallel. It chooses two systems randomly sense them, compares the size of run queue of these systems also with the run queue of its own. Whichever system has least queue size, is selected for allocation of task. For next 1000ms, execution is continued and response time and waiting time are analyzed.

Nearest neighbor algorithm is then executed on each system in the same way. Each node has information about its neighbors. It compares run queue length of its neighbors and along with its own run queue length. Whichever node is having least queue size, task is transferred to that node. For same period of time, this algorithm runs and response time and waiting time are compared.

5.2 Results and Analysis:

Response Time The following table shows the values of response time noted for both the algorithms.

Response time(in sec)		
System	Random Subset Approach	Nearest Neighbor Approach
Machine 1	0.17	0.02
Machine 2	0.50	0.06
Machine 3	0.45	0.05
Machine 4	0.38	0.08
Machine 5	0.28	0.03

Table 1: Response time of Different Machines

Above tabular values are shown in the following figure:

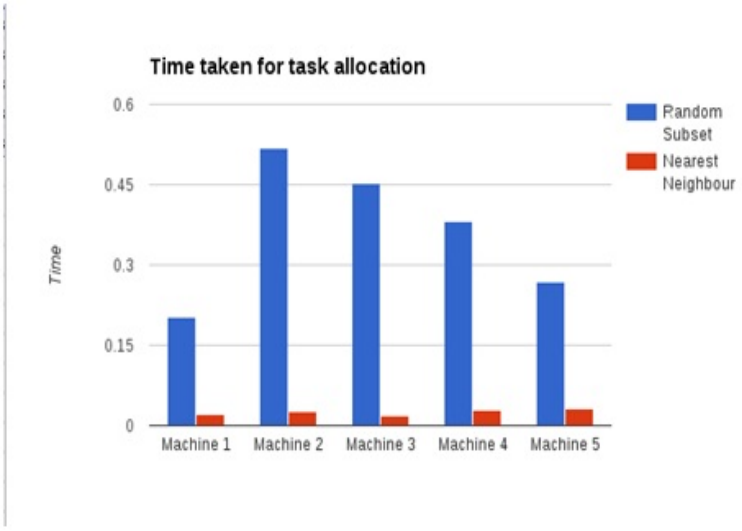

Figure 4: Response time graph

Waiting Time The following table shows the values of Waiting time noted for both the algorithms.

Waiting time(in sec)		
System	Random Subset Approach	Nearest Neighbor Approach
Machine 1	0.20	0.15
Machine 2	0.18	0.16
Machine 3	0.21	0.21
Machine 4	0.11	0.05
Machine 5	0.18	0.09

Table 2: Waiting time of Different Machines

Above tabular values are shown in the following graph:

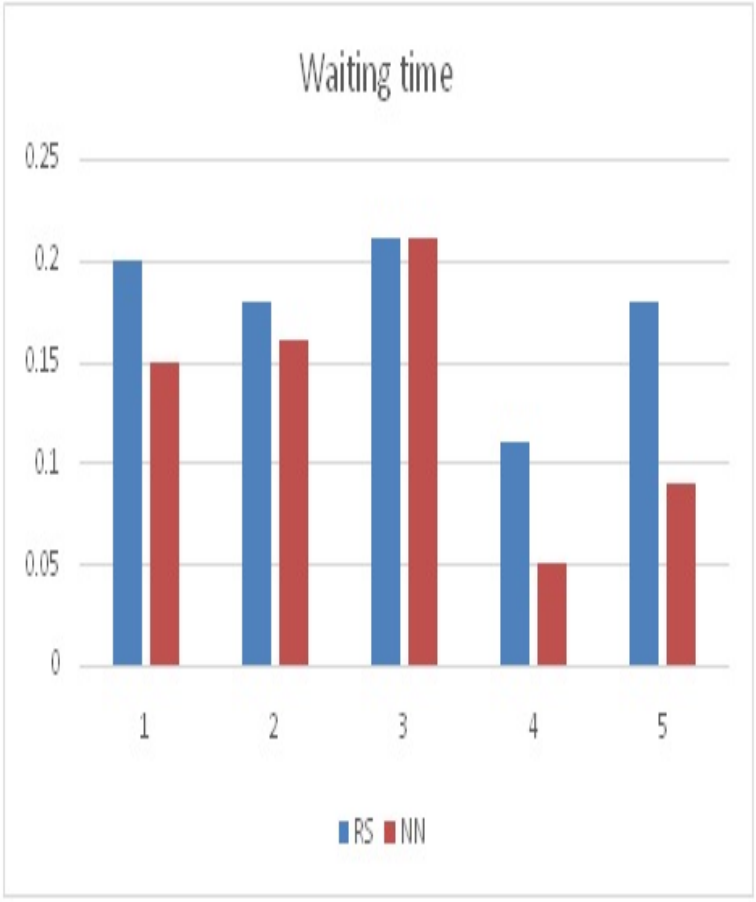

Figure 5: Waiting time graph

6 Conclusion and Future Work

This paper deals with task allocation schemes in large scale distributed system. In order to select the node for transferring the task, load information of task is very much useful. But the problem here is, the performance of whole system degrades if the load information is old. nearest neighbor approach is proposed in order to tackle this problem. the impact of communication delay is also reduced. It is shown that the proposed scheme gives better performance compared to the previous approach based on randomness.

But the evaluation of performance is only limited to the steady-state. The task allocation scheme in general, should be able to deal with various abnormal conditions such as failure of nodes, burst arrival of tasks etc. The future work is to evaluate the proposed approach in such abnormal conditions.

References

[1] R. Mirchandaney, D. Towsley, and J. Stankovic, "Analysis of the effects of delays on load sharing," IEEE Trans. on Computers, vol. 38, no. 11, pp. 1513-1525, 1989. (Pubitemid 20642376)

[2] M. Mitzenmacher, "How Useful Is Old Information?" IEEE Trans. On Parallel and Distributed Systems, vol. 11, no. 1, pp. 6-20, 2000. (Pubitemid 32132996)

[3] M. Dahlin, "Interpreting stale load information," in Proc. 19th IEEE International Conference on Distributed Computing. Systems, 1999, pp. 285-296.

[4] A. M. Alakeel, "A Guide to Dynamic Load Balancing in Distributed Computer Systems," International Journal of Computer Science and Information Security, vol. 10, no. 6, pp. 153-160, 2010.