© 2016 IJSRSET | Volume 2 | Issue 2 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

A Novel Approach for Scalable and Efficient Case Recommender System for E-Shoppers

Saraswathi M , Abhinav Prabhu A, Deepak Jain S, Jayaprakash J

Department of Computer Science and Engineering, Dhanalakshmi College of Engineering, Chennai, Tamilnadu, India

ABSTRACT

Big-Data Computing is a new critical challenge for the ICT industry. Engineers and researchers are dealing with data sets of petabyte scale in the cloud computing paradigm. Thus the demand for building a service stack to distribute, manage and process massive data sets has risen drastically. In this paper, we investigate the Big Data Broadcasting problem for a single source node to broadcast a big chunk of data to a set of nodes with the objective of minimizing the maximum completion time. These nodes may locate in the same datacentre or across geo-distributed datacentres. This problem is one of the fundamental problems in distributed computing and is known to be NP-hard in heterogeneous environments. We model the Big-data broadcasting problem into a Lockstep Broadcast Tree (LSBT) problem. The main idea of the LSBT model is to define a basic unit of upload bandwidth, r, such that a node with capacity c broadcasts data to a set of ⌊ c=r⌋ children at the rate r. Note that r is a parameter to be optimized as part of the LSBT problem. We further divide the broadcast data into m chunks. These data chunks can then be broadcast down the LSBT in a pipeline manner. In a homogeneous network environment in which each node has the same upload capacity c, we show that the optimal uplink rate r of LSBT is either c=2 or c=3, whichever gives the smaller maximum completion time. For heterogeneous environments, we present an O(nlog2n) algorithm to select an optimal uplink rate r and to construct an optimal LSBT. Numerical results show that our approach performs well with less maximum completion time and lower computational complexity than other efficient solutions in literature.

Keywords : Scalable and Efficient precise system , Web application building and broadcasting, Gateway application over TSV data, Web Crawling and Map reduction, recommendation of products.

I. INTRODUCTION

Hadoop is a free, Java-based programming framework that supports the processing of large data sets in a Parallel and distributed computing environment. It makes Use of the commodity hardware Hadoop is Highly Scalable and Fault Tolerant. Hadoop runs in cluster and eliminates the use of a Super computer. Hadoop is the widely used big data processing engine with a simple master slave setup.

Big Data in most companies are processed by Hadoop by submitting the jobs to Master. The Master distributes the job to its cluster and process map and reduce tasks sequencially.But nowdays the growing data need and the and competition between Service Providers leads to the increased submission of jobs to the Master. This Concurrent job submission on Hadoop forces us to do Scheduling on Hadoop Cluster so that the response time will be acceptable for each job. The main objective is to compete with the big data problems prevailing in many of the Service Recommender Systems in Market and to build a Scalable, Efficient and Precise System for Service level Comparison between products in Market.

II. METHODS AND MATERIAL

A. Related Work

Mohammad banikazemi of Ohio State University, Colombus works based on Networks of Workstations (NOW) have become an attractive alternative platform

for high performance computing. Due to the commodity nature of workstations and interconnects and due to the multiplicity of vendors and platforms, the NOW environments are being gradually redefined as Heterogeneous Networks of Workstations (HNOW) environments. This paper presents a new framework for implementing collective communication operations (as defined by the Message Passing Interface (MPI) standard) efficiently for the emerging HNOW environments. We first classify different types of heterogeneity in HNOW and then focus on one important characteristic: communication capabilities of workstations. Taking this characteristic into account, we propose two new approaches (Speed-Partitioned Ordered Chain (SPOC) and Fastest-Node First (FNF)) to implement collective communication operations with reduced latency. We also investigate methods for deriving optimal trees for broadcast and multicast operations. Generating such trees is shown to be computationally intensive. It is shown that the FNF approach, in spite of its simplicity, can deliver performance within 1% of the performance of the optimal trees. Finally, these new approaches are compared with the approach used in the MPICH implementation on experimental as well as on simulated testbeds. On a 24 - node existing HNOW environment with SGI workstations and ATM interconnection, our approaches reduce the latency of broadcast and multicast operations by a factor of up to compared to the approach used in the existing MPICH imple - mentation. On a node simulated testbed, our approaches can reduce the latency of broadcast and multicast operations by a factor of up to 3.5 . Thus, these results demonstrate that there is significant potential for our approaches to be applied towards designing scalable collective communication libraries for current and future generation HNOW environments. Chin-Jen Wu worked on it based on how to rapidly disseminate a large-sized file to many recipients is a fundamental problem in many applications, such as updating software patches and distributing large scientific data sets. In this paper, we present the Bee protocol, which is a cooperative peer-to-peer data dissemination protocol aiming at minimizing the maximum dissemination time for all peers to obtain time-critical data, such as critical patch updates. Bee is a decentralized protocol that organizes peers into a randomized mesh-based overlay and each peer only works with local knowledge. We devise a slowest peer first strategy to boost the speed of dissemination, and a topology adaptation algorithm that provides the most efficient utilization of the network capacity. Bee is designed to support network heterogeneity and deal with the flash crowd arrival pattern without sacrificing the dissemination speed.We present experimental results on the performance of Bee in terms of dissemination time and show that its performance can approach lower bound of the maximum dissemination time

B. Existing System

Existing Systems only provide users, with the products in their stocks and will render the Comparison within their products only. Thereby limiting the users to analyze before buying a product. Existing Service Recommender Systems suffers from big data Problems like scalability and Time Consumption and thus lack of preciseness.

III. RESULTS AND DISCUSSION

A. Proposed System

We propose a Scalable, efficient and Precise Service Comparison and Recommender System which enables the shoppers to deeply analyze on what product to choose and in which Application, ease and fair with our Gateway .The Shoppers will be provided with Clean Indexes of various products with its spec ,cost and also Service Ratings which is done in a statistical way .Our System crabs the data's from various web application and loads in its datasets collaboratively and process with Batch jobs so as to Categories classify and to Index the data's in a distributed and Parallel processing Manner.

Shoppers can Analyze, Get Recommendations and Can Pick Products and Add to Cart irrespective Of the Service Provider. Hence Our Applications Stands unique as it does not rely on the Single Service Provider. The Cart can be reviewed at any time and can be Processed Whenever the Shopper Wants the Product. All the Information Will be Securely and Precisely Stored in the Users Session. The Purchase phase look up for the Web services of the Products Service Provider and can make the Online Payment with the Banks from Service Provider. Once it got over Process Gets Back to our Gateway bringing out the Track Id's from Product Service Provider.

B. User Classes and Characteristics:

1. Various Web applications Building and Broadcasting:

Sample web applications were built so that the users can compare their products with different service providers. The applications use sample datasets that has been crawled in amazon previously. Similar datasets were prepared for other Applications too using the Meta model that has been crawled earlier. Each data set was loaded independently in various web applications. Features and other specifications have been loaded differently for each application based on the service providers' requirement. These applications have been deployed in web servers so that the application is up and running. Web Services have been written on each web application so that any third party can communicate with secure authentication.

- ✓ Preprocessing(Loads the data from HDFS and makes it ready for process)
- ✓ Clustering(Groups the data depending upon the Category –Tree Structure)
- ✓ Classification(Splits up the Relevant Resources-Info, Features)
- ✓ Distribution(Distributes the data to various Web servers-Broadcasting)

2. The Gateway Application and Batch Processing over the TSV Data:

The gateway application is built which gives users with recommendations and comparisons between the products in the market .Generally the resources provided by various web servers are in TSV (Tab Separated Values) format and should be batch processed before proceeding. For that we use our own API for TSV manipulation. The TSV files were parsed for data. These data's are used for further processing (i.e. For Recommendation and comparison). Admin login, conversion of text to object for information and conversion of object to text for information are the following steps takes place.

3. Web Crawling for Resources and MapReduce:

The users can register and can login to view various products available in market. This is done by writing a web service client process for each service provider. It can connect to the various web applications web service and can pull all the needed data's to our backend. A huge amount of data got accumulated now .Web crawling looks for web services provided by various web applications. The crawled resources are then reduced by mapreduce framework and converted into a single object .This reduced object contains all the necessary information for providing comparison and recommendations. User registration, user login, mapreduce and product list.

Map Reduce – It is a combination of a map task and multiple reduce tasks which is used for clubbing all the slave process outcomes.

4. Picking Products from Recommendations and Purchase:

The Recommendations were given based on the QOS, Availability, Delivery, Offers, Price and Specifications of the particular product. The users can pick any product so that our application provides with a most genuine recommendation and a set of comparisons. The Users are provided with neat and clean indexes so that he can pick a best provider for a particular product. The picked products were added in cart and can be purchased later. The user cart is equipped with case based recommender systems. It uses case-based reasoning (CBR) to identify and recommend the items that seem more suitable for completing a user's buying experience provided that he or she has already selected some items. The system models complete transactions as cases and recommended items come from the evaluation of those transactions. Because the cases aren't restricted to the user who purchased them, the developed system can generate accurate item recommendations for joint item selections, both for new and existing users. Having analyzed the previous transactions and identified the concepts within which concrete items appear, the given part of a new transaction is matched over the existing Ones to find the more adequate solution. i.e. the best way to fill this basket.

When the user initiates transaction our gateway will connect to the banking web services directly on behalf of the service provider and completes the transaction securely with help of OTP sent to their mail id given on user registration .A bank account is needed for complete the transaction which can be created earlier through our

banking application .The Process will be back to our application as soon as the transaction is over and the purchased products will be reflected on the bag List. i.e. purchased items list. Product comparisons, add to cart, case based recommendation, purchase and user bag.

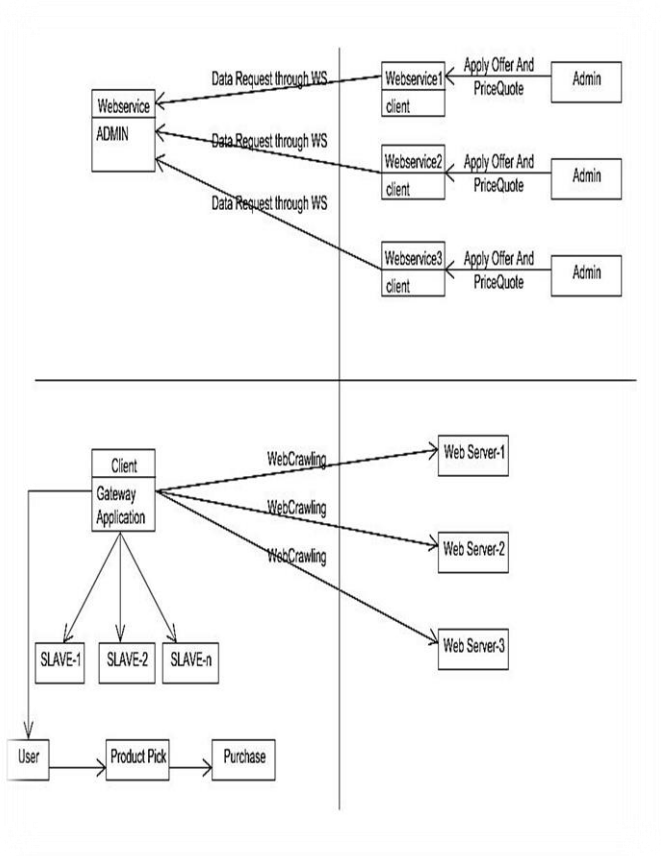

Figure 1. Architecture Diagram

IV. CONCLUSION

Thus in proposed system the shoppers can analyze, get recommendations and can pick products and add to cart irrespective of the service provider. Hence our applications stands unique as it does not rely on the single Service Provider. The Cart can be reviewed at any time and can be processed whenever the shopper wants the product. All the information will be securely and precisely stored in the users session. The purchase phase look up for the web services of the products service provider and can make the online payment with the banks from service provider. Once it got over process gets back to our gateway bringing out the Track Id's from product service provider.

V. REFERENCES

[1] R. E. Bryant, R. H. Katz, and E. D. Lazowska, "Big-data computing: Creating revolutionary break throughs in commerce, science, and society," In Computing Research Initiatives for the 21st Century., 2008.

[2] A. Szalay and J. Gray, "2020 computing: Science in an exponential world," Nature 440, 413-414, March, 2006.

[3] G. Brumfiel, "High-energy physics: Down the petabyte highway," Nature 469, 282-283 January, 2011.

[4] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," Proc. of Operating Systems Design and Implementation (OSDI), 2004.

[5] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, , and R. E. Gruber, "Bigtable: A distributed storage system for structured data," Proc. of Operating Systems Design and Implementation (OSDI), 2006.

[6] W. D. Hillis and G. L. Steele, Jr., "Data parallel algorithms," Communications of the ACM, vol. 29, pp. 1170–1183, December 1986.