International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Multiservice Product Comparison System

S. Deepa[1], S. Krishna Meena[2], Mr. S.Thirumal (AP) [3]
[1,2,3]B.E Computer Science And Engineering
GKM College Of Engineering And Technology (Affiliated to Anna University), Perungalathur, Chennai-600063

Abstract— Big-data computing is a new critical challenge for the ICT industry. Engineers and researchers are dealing with data sets of petabyte scale. Conventional way to provide comparison between products from heterogenous webservices .Unique indexes are provided for products. LSBT is implemented and MPCS is processed.

I. INTRODUCTION

Major problem complexity in broadcasting is solved by lockStep Broadcast Tree Problem By this we define a performance goal for a single LSBT, that is achieving minimum completion time by optimizing the basic bandwidth allocation, r, among LSBT nodes. Different from original problem, we allow data be divided into chunks and sent in a pipeline fashion. Formally,given a set of n nodes each node is connected to the network via an access link of upload capacities ci and a size of chunks B. The LSBT problem is to determine the upload bandwidth r of each uplink to build the LSBT t, in which node ni should allocate upload bandwidth r to each connection to its child nodes in order to minimize the maximum completion time D for propagating a data chunk. Note that it is possible to handle simultaneously several connections and to fix the bandwidth allocated to each connection . In the following definition,we define the number of edges k in each node for LSBT.LSBT with the set of upload capacity c and an upload bandwidth r, describes the function that returns the height of the LSBT Note that this general Equation removes restrictions on the location of nodes in the network, it only calculates the propagation delay of data chunks from the root to the leaves. Moreover, LSBT model addresses the data broadcasting problem by building a single broadcast tree, in which nodes can transmit data chunks in a pipeline manner.

II. PROBLEM FORMULATION

The assumption in our model is similar to the Uplink-Sharing model proposed by Mundinger et al. Each node can simultaneously connect to other nodes and the available upload capacity of a link is shared equally amongst the uploading connections. Based on the Uplink-Sharing model,we model the nodes and data transfer networks as the nodes and edges of a direct graph. We assume that there are n nodes in a network system, denoted by n1, n2, n3, . . . ; nn, where the broadcasting source is node n1 and the n 1 nodes haveupload capacities c¼ fc1, c2, c3, . . . ; cng, measured in kilobyte per second (KBps). Besides, we also assume that the source node, n1, has the data item that is divided into m chunks of equal size, to disseminate to all the other nodes, and c1 is larger than or equal to that of other nodes. Finally, we assume that the downloading capacity of each node is larger or equal to its uploading capacity. This is true for virtually all existing network access technologies, e.g., ADSL or cable modems.

III. RELATED WORKS

The data broadcasting problem established by Edmonds since the 1970s and has been studied in many articles. The broadcast problem is the core of every data distribution system,especially in peer-to-peer (P2P) overlay fields, it is of great interest to current efficient P2P data distribution systems,based on a tree or mesh design. While there is much work on system design and measurement studies of P2P data distribution systems , few papers work on theoretical analysis and fundamental limitations of P2P data distribution systems. Ezovski et al. proposed an optimal network topology and the associated scheduling

IV. ALGORITHMS

In this section, we present our LSBT algorithm that is also a heuristic for the data broadcasting problem. Given a set of node upload capacities c, we aim at finding an optimal LSBT, that is a data broadcast tree where data chunks can be sent in a pipelined manner. We provide a thorough analysis policy to achieve the min-min times, by assuming that the file is broken into infinitesimally small chunks such that there is almost no forwarding delay. The authors claimed that the proposed scheme which achieves min-min times can also achieve the minimum average finish time. However,Chang et al. disproved the claim In,the authors propose several distributed algorithms to optimize the throughput of a broadcasting operation. However, they do not consider degree constraints in each node. In , Beaumont et al. considered the maximizing throughput problem of broadcasting a large message in heterogenous networks. They introduced the bounded degree multiport model to model the capabilities of the nodes and proved that the data

Volume 4 Issue III, March 2016
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

broadcasting problem of maximizing the overall throughput is NP-Complete. Liu et al. studied the maximum streaming rate problem of peer-assisted The number of chunks versus the maximum completion time the size of file is 100 MB and the number of nodes is 100, and the unit on y-axis is second).The number of chunks versus the computation time (the size of file is 100 MB and the number of nodes is 100, and the unit on y-axis is thousand milliseconds).The maximum completion time versus the size of delivery data(the number of data chunks is 1,000 and the number of nodes is 1,000,and the unit on y-axis is million milliseconds). of LSBT in both homogenous and heterogenous network systems. We first clarify LSBT in homogenous networks cases and describe the LSBT algorithm in heterogenous.

A. Homogenous Network Systems

We present the optimal solution of LSBT when the upload capacities of nodes are identical. In general, we assume that all nodes have upload capacity of c. Mundinger et al. have presented the optimal scheduling solution for broadcasting multiple messages on the uplink-sharing model.

The following Theorem 1 (Mundinger's theorem) is proved in the article If each round costs one unit of time, then the maximum completion time of the optimal solution is m þ blog 2nc, where m is the number of chunks and n the number of nodes. Note that each node can only upload one data chunk to another node in each round. By contrast, each node can send a data chunk to k other nodes simultaneously in the LSBT model.

Theorem 1 (Mundinger's Theorem. In homogenous network systems, the minimum number of rounds required to complete the broadcasting of all data chunks is m þ blog 2nc, where m is the number of data chunks and n is the number of nodes.

Algorithm 1. GLSBT: An algorithm for generatin; LSBT t

Input: a set of upload capacities c and a rate r for LSBT t
Output: a LSBT t

```
1: BEGIN
2: /* Computing the degree of the node n_i */
3: for i ← 1 to n do
4:    LSBTree[i].Degree ← ⌊c_i/r⌋
5: end for
6: NodeIndex ← 1
7: NodeCount ← 2
8: while NodeCount ≤ n do
9:    k ← LSBTree[NodeIndex].Degree
10:   for j ← 1 to k do
11:      LSBTree[NodeCount].parent ← NodeIndex
12:      NodeCount = NodeCount + 1
13:   end for
14:   NodeIndex = NodeIndex + 1
15: end while
16: return LSBTree[]
17: END
```

Algorithm 2. A discretization algorithm for the candidateset

Input: a set of upload capacities c and the *upper* and *lower* bounds of r^*
Output: *CandidateSet*

```
1: BEGIN
2: UnionSet ← empty
3: CandidateSet ← empty
4: for i ← 1 to n do
5:    UnionSet ← UnionSet ∪ c_i
6: end for
7: UnionSet ← Sort(UnionSet)
8: for k ← 1 to u − 1 do
9:    for all u in UnionSet do
10:      r ← u/k
11:      if r ≥ upper then
12:         continue
13:      else if r < lower then
14:         break
15:      end if
16:      CandidateSet ← CandidateSet ∪ r
17:   end for
18: end for
19: return CandidateSet
20: END
```

Let Candidate Set denote the set of the possible value of r, and the binary search will be performed on it. In Algorithm 2, it first reduces the redundance of ci by preforming an union operation (named Union Set) of each ci, for 1 i n and sorting the set (in line 4-7). Next, theloop from line 8 to 18 is used to discretize the value of r and to filter out the extreme r values restricted by the upper and lower bounds. In the loop, it gets candidates of r by computing u=k; 8 u 2 Union Set and 1 k ðn 1Þ, and puts those candidates into the Candidate Set. Note that the number of candidates is O(n2) if each LSBT node has an unique upload capacity. However, the filter scheme can significantly reduce the number of candidates. We will show the experimental results in the next section. Before we present the binary search algorithm for selecting the value of r, we first show the following lemma and theorem which provide properties to derive the efficient binary search algorithm on r.

B. Clustering Algorithm

The goal of clustering is to minimize the number of subscription checks. In the static approach clustering decisions are taken given the global knowledge of all subscriptions in the system and the knowledge of statistics about incoming event streams. But subscription and event patterns may change over time, degrading an initial optimal clustering. To cope with this problem a first solution consist of periodically recomputing from scratch a clustering instance that is adapted to the new situation. Due to the complexity of this reorganization, this solution is well suited for applications where subscription and event patterns are relatively stable during large time intervals. But this static approaches clearly impracticable when patterns are evolving continually.
In this section we describe a clustering algorithm that incrementally adapts clustering to alterin subscription and event patterns. Our algorithm dynamically decides

When to redistribute subscription form a given cluster to another more profitable cluster.
When to delete a hash table and redistribute its subscriptions and
When to create a new hash table and what table to create

These decisions rely on tow metrics called cluster benefit margin and hash table benefit. A cluster is redistributed if its benefit margin is high. A hash table is crated when its benefit Is sufficiently high and removed when its benefit is too small.

given :
c the current cluster and H_c its hash table.
$\mathcal{H}$ the set of hash tables that have been already created.
PH a set of potential hash table.
$candidate\_clusters$ a set of pairs (H, cls) where H belongs to PH and cls is a set of clusters.
BODY :
if $BM(c) \geq BMmax$ // Cluster c has an excessive benefit margin
Cluster_distribute(c)
While(Exists H in PH such that $B(H) \geq Bmin$
$\mathcal{H} = \mathcal{H} \cup \{H\}$; $PH = PH - \{H\}$
Foreach cluster c' in $H.candidate$ do
Cluster_distribute(c')
END BODY :
Cluster_distribute(c)
Foreach subscription s in c do
let $Hbest(s)$ be the hash table in $\mathcal{H}$ such that $\nu(H, s)$ is minimal
if$Hbest(s) \neq H$ do
move s to H
if s is marked do
Foreach table H in $PH \cap GA(s)$ do
B(H)-=1; delete mark from s od
// The redistribution did not improve enough the Benefit margin
if $BM(c) \geq BMmax$ do
Foreach subscription s in c such s is not marked do
Foreach table H in $PH \cap GA(c)$ do
B(H)+=1
add c to $H.candidate$
mark s od

V. METHODOLOGY

A. 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.

B. 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.

VI. ACKNOWLEDGEMENT

We would like to express our gratitude and appreciation to our prof Mr. Thirumal for giving us the opportunity to work in this project.

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, vol. 440, pp. 413–414, Mar. 2006.
[3] G. Brumfiel, "High-energy physics: Down the petabyte highway," Nature, vol. 469, pp. 282–283, Jan. 2011.
[4] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Proc. Oper. Syst. Design Implementation, 2004, pp. 137–150.
[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. Oper. Syst. Design Implementation, 2006, pp. 205–218.
[6] W. D. Hillis and G. L. Steele, Jr., "Data parallel algorithms," Commun. ACM, vol. 29, pp. 1170–1183, Dec. 1986.
[7] U. Rencuzogullari and S. Dwarkadas, "Dynamic adaptation to available resources for parallel computing in an autonomous network of workstations," Proc. 8th ACM SIGPLAN Symp. Principles Practices Parallel Program., 2001, pp. 72–81.
[8] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, and I. Stoica, "Managing data transfers in computer clusters with orchestra," Proc. ACM Special Interest Group Data Commun., 2011, pp. 98–109.
[9] D. Nukarapu, B. Tang, L. Wang, and S. Lu, "Data replication in data intensive scientific applications with performance guarantee," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 8, pp. 1299–1306, Aug. 2011.