FiDoop: Parallel Mining of Frequent Itemsets Using MapReduce

ABSTRACT:

Existing parallel mining algorithms for frequent itemsets lack a mechanism that enables automatic parallelization, load balancing, data distribution, and fault tolerance on large clusters. As a solution to this problem, we design a parallel frequent itemsets mining algorithm called FiDoop using the MapReduce programming model. To achieve compressed storage and avoid building conditional pattern bases, FiDoop incorporates the frequent items ultrametric tree, rather than conventional FP trees. In FiDoop, three MapReduce jobs are implemented to complete the mining task. In the crucial third MapReduce job, the mappers independently decompose itemsets, the reducers perform combination operations by constructing small ultrametric trees, and the actual mining of these trees separately. We implement FiDoop on our in-house Hadoop cluster. We show that FiDoop on the cluster is sensitive to data distribution and dimensions, because itemsets with different lengths have different decomposition and construction costs. To improve FiDoop’s performance, we develop a workload balance metric to measure load balance across the cluster’s computing nodes. We develop FiDoop-HD, an extension of FiDoop, to speed up the mining performance for high-dimensional data analysis. Extensive experiments using real-world celestial spectral data demonstrate that our proposed solution is efficient and scalable.

INTRODUCTION

Frequent itemsets mining (FIM) is a core problem in association rule mining (ARM), sequence mining, and the like. Speeding up the process of FIM is critical and indispensable, because FIM consumption accounts for a significant portion of mining time due to its high computation and input/output (I/O) intensity. When datasets in modern data mining applications become excessively large, sequential FIM algorithms running on a single machine suffer from performance deterioration. To address this issue, we investigate how to perform FIM using MapReduce—a widely adopted programming model for processing big datasets by exploiting the parallelism among computing nodes of a cluster. We show how to distribute a large dataset over the cluster to balance load across all cluster nodes, thereby optimizing the performance of parallel FIM.
EXISTING SYSTEM

In Existing System Rather than considering Apriori and FP-growth, we incorporate the frequent items ultrametric tree (FIU-tree) in the design of our parallel FIM technique. We focus on FIU-tree because of its four salient advantages, which include reducing I/O overhead, offering a natural way of partitioning a dataset, compressed storage, and averting recursively traverse.

DisADVANTAGE OF Existing SYSTEM

✓ Parallel algorithms lack a mechanism that enables automatic parallelization, load balancing, data distribution, and fault tolerance on large computing clusters.

PROPOSED SYSTEM

In Proposed System a new data partitioning method to well balance computing load among the cluster nodes; we develop FiDoop-HD, an extension of FiDoop, to meet the needs of high-dimensional data processing.

ADVANTAGE OF PROPOSED SYSTEM

✓ FiDoop is efficient and scalable on Hadoop clusters.
HARDWARE REQUIREMENTS:

- System: Pentium IV 2.4 GHz.
- Hard Disk: 40 GB.
- Floppy Drive: 44 Mb.
- Monitor: 15 VGA Colour.

SOFTWARE REQUIREMENTS:

- Coding Language: Java 1.7, Hadoop 0.8.1
- Database: MySql 5
- IDE: Eclipse