Available online at www.sciencedirect.com
ScienceDirect
Advances in Engineering Software 38 (2007) 295–300
ADVANCES IN ENGINEERING SOFTWARE
www.elsevier.com/locate/advengsoft

Grid implementation of the Apriori algorithm

Cristian Aflori [1], Mitica Craus *

Technical University "Gh.Asachi", Department of Computer Science and Engineering, 53 A, Dimitrie Mangeron Street, 700050 Iasi, Romania

Received 26 October 2005; accepted 8 August 2006
Available online 19 October 2006

Abstract

The paper presents the implementation of an association rules discovery data mining task using Grid technologies. For the mining task we are using the Apriori algorithm on top of the Globus toolkit. The case study presents the design and integration of the data mining algorithm with the Globus services. The paper compares the Grid version with related work in the field and we outline the conclusions and future work.

Keywords: Grid technologies; Data mining; Association rules; Apriori algorithm; Globus toolkit

1. Introduction

Data Mining (DM) or Knowledge Discovery in Databases (KDD) [1] is an interdisciplinary field with major impact in the scientific and commercial environments. Data Mining is the iterative and interactive process of discovering valid, novel, useful, and understandable patterns or models in massive databases. Data Mining means searching for valuable information in large volumes of data, using exploration and analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns and rules. The major data mining tasks [2] are prediction and description. Prediction methods use some variables to predict unknown or future values of other variables: these include classification, regression, and deviation detection. The description methods find human-interpretable patterns that describe the data: these include clustering, association rules discovery and sequential pattern discovery. KDD consists of an iterative sequence of the following steps: data selection, data cleaning, data transformation, pattern generation, validation and visualisation.

Association rule induction is a powerful method used to find regularities in data trends [3]. By induction of the association rules, sets of data instances that frequently appear together must be founded. Such information is usually expressed in the form of rules. An association rule expresses an association between items or sets of items. However, only those association rules that are expressive and reliable are useful. The standard measures used to assess association rules are the support and the confidence of a rule. Both are computed from the support of certain item sets.

2. Problem formulation

The problem of mining association rules was introduced in 1993 by Agrawal [3]. Notations and definitions:

- $I = \{i_1, i_2, \ldots, i_m\}$ – set of items;
- D = set of transactions; each transaction t is included in I;
- X = set of items from I, t contains X.
- An association rule is a pair $X \rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$, $X \cap Y = \emptyset$.
- Confidence of the rule $X \rightarrow Y$ is c, if c% of the transactions in D that contain the set X, contain also the set Y.
- Support of rule $X \rightarrow Y$ is s, if s% of the transactions in D contains the set $X \cup Y$.

* Corresponding author. Tel.: +40 742 024117; fax: +40 232 232430.
E-mail addresses: caflori@cs.tuiasi.ro (C. Aflori), craus@cs.tuiasi.ro (M. Craus).
[1] Tel.: +40 741 166401; fax: +40 232 232430.

0965-9978/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2006.08.011

Problem statement:

Given a set of transactions D, generate all the association rules (or a specified number of them) that have greater support and confidence than the user-specified minimum support and minimum confidence.

The task of data mining for association rules can be divided into two steps:

- find all large itemsets that have transaction support greater than the minimum support;
- for all large itemsets, for each itemset l, find all non-empty subsets of l for every such subset a, the rule is $a \rightarrow l - a$ if $\frac{\text{support}(l)}{\text{support}(a)} >$ minimum confidence [4].

The most popular algorithm used for the association rules discovery is the Apriori algorithm.

- Initial conditions:
  $L_k =$ set of large k-itemsets (set of items having minimum support); $C_k =$ set of candidate k-itemsets (items to be counted); $D =$ set of transactions, $t \subset D$.
- Algorithm:

```
L1 = {frequent 1-itemsets};
for(k = 2;Lk−1⟨⟩0;k++){
   Ck = set of new candidates;
   for all transactions t ⊂ D
         for all k-subsets m of t
               if (m ⊂ Ck)m.count++
   Lk = {n ⊂ Ck|n.count >= minsupp}
}
Set of all frequent itemsets = ∪ kLk;
```

The Apriori algorithm finds only the frequent itemsets and for finding the associations rules we must apply the following algorithm:

```
for (each frequent itemset l){
   generate all non-empty subsets of l
   for (each non-empty subset a of l) {
   output_rule: a → (l−a) if support(l)/support(a)> =
 min_confidence}
}
```

For the general problem of the association discovery rule, given m items, there are potentially 2^m frequent itemsets. Discovering frequent itemsets requires a lot of computation and storage resources, plus many input/output (I/O) communications. In the Apriory Algorithm case, the database is scanned for each iteration in order to obtain the support for new candidates. If the database does not fit memory then in each iteration there is a high I/O overhead for scanning.

The efficient and effective discovery of the association rules in large databases poses numerous requirements and great challenges to researchers and developers. Scalability of the data mining process implies efficient and sufficient sampling, in-memory vs. disk-based processing and high-performance computing. Recently, several KDD systems have been implemented on parallel computing platforms, in order to achieve high-performance in the analysis of the large data sets that are stored in the same location. Large data sets, the geographic distribution of data and computationally intensive analysis demand parallel and distributed infrastructure [5].

Advances in networking technology and computational infrastructure made it possible to construct large-scale high-performance distributed computing environments, or computational grids that provide dependable, consistent, and pervasive access to high-end computational resources. The term computational grid refers to an emerging infrastructure that enables the integrated use of remote high-end computers, databases, scientific instruments, networks, and other resources [6]. Grid applications often involve large amounts of computing and/or data. For these reasons, grids can offer an effective support for the implementation and use of parallel and distributed data mining systems.

3. Related work

There are several systems proposed in the field of the high-performance data mining. Most of them do not use computational grid infrastructure for the implementation of basic services such as authentication, data access, communication and security. These systems operate on clusters of computers or over the Internet. The best known systems for distributed data mining are presented below.

Kensington Enterprise data mining is a PDKD system based on a three-tier client/server architecture in which includes: client, application server and third-tier servers (RDBMS and parallel data mining service) [7]. The Kensington system has been implemented in Java and uses the Enterprise JavaBeans component architecture. Java Agents for Meta-learning (JAM) is an agent-based distributed data mining system that has been developed to mine data stored in different sites for building so called meta-models as a combination of several models learned at the different sites where data are stored. JAM uses Java applets to move data mining agents to remote sites [8]. Bio-diversity database platform (BODHI) is another agent-based distributed data mining system implemented in Java [9].

Papyrus is a distributed data mining system developed for clusters and super-clusters of workstations, composed four software layers: data management, data mining, predictive modeling, and agent [10]. Another interesting distributed data mining suite based on Java is Parallel and distributed data mining application suite (PaDDMAS), a component-based tool set that integrates pre-developed or custom packages (that can be sequential or parallel) using a dataflow approach [11].

Alongside this research work on distributed data mining, several research groups are working in the computational grid area developing algorithms, components, and services that can be exploited in the implementation of distributed data mining systems.