ClusterMiner: High Performance for Data, Text and Web Mining

Marta Mattoso[1,*], Nelson Ebecken[1,*],
Gerson Zaverucha[1], Alexandre Gonçalves Evsukoff[1],
Fernanda Araujo Baião[2], Myriam Aragão Costa[1], Guilherme Saad Terra[1]

[1]COPPE/UFRJ
P.O.Box 68511 – 21941-972 – Rio de Janeiro – RJ – Brazil

[2]Department of Applied Informatics – UNIRIO
Av Pasteur 458 – 22290-240 – Rio de Janeiro – RJ – Brazil

http://clusterminer.nacad.ufrj.br, clusterminer@yahoogroups.com
[*]Project Coordinators: marta@cos.ufrj.br, nelson@ntt.ufrj.br

Abstract. *Our work addresses a variety of inter-related issues with a focus on providing tools for efficiently processing data in mining tasks. We investigate mechanisms to efficiently access high volumes of data, as well as issues on parallel processing of data mining algorithms. We focus on the development of ClusterMiner, an environment with high performance tools for improving data, text and web mining using PC clusters. The goal is to improve performance metrics on data analysis, data sorting and summarization. ClusterMiner explores parallel data access by enhancing data distribution techniques and parallel execution of heavy-weight queries using database clusters. In addition, we investigate parallel techniques to improve typical rule-based algorithms under a variety of scenarios. In ClusterMiner, performance analysis plays an important role in evaluating design alternatives. Finally, we are working on making our research results available in form of useful tools for the community.*

1. Introduction

The *Information Age* was marked by the strong dissemination of digital repositories, where databases were spread in computational "islands" interconnected through the World Wide Web. This scenario has evolved, and nowadays we have to face the challenges of the *Knowledge Age*, with emphasis on integrating and processing this huge volume of information to reach some useful purpose. This information can be found in structured databases (e.g. relational, object-oriented), in semi-structured databases (e.g. in XML documents), in text files, as well as flat files. In sync with such (r)evolution, the techniques for database management have been radically altered since the traditional database system architectures are no longer suited for current problems.

Data mining techniques have increasingly been studied, especially their application in real-world databases. One typical problem is that databases tend to be very large, and these techniques often repeatedly scan the entire set. Sampling has been used for a long time, but subtle differences among sets of objects become less evident. There are several

advantages of using a database management system (DBMS) to manage and process data sets instead of conventional flat files. This approach has been a major concern of several researches, because it represents a natural solution for data mining since DBMS have been successfully used in business management of very large data bases and may store valuable hidden knowledge.

One requirement of data mining is efficiency and scalability of mining algorithms. Therefore, parallelism can be used to process long running tasks in a timely manner. In this context, parallel database systems come to play an important role, because they can offer, among other advantages, transparent and painless implementation of parallelism to process large data sets. Due to high costs in parallel database systems and application migration, we propose the use of a database cluster software layer combined to parallel data mining algorithms. A database cluster can be defined as a cluster of PC running off-the-shelf DBMS at each node.

ClusterMiner addresses a variety of inter-related issues with focus on high performance environment for data, text and Web mining. Several problems have been investigated from issues on how to efficiently store, fragment and replicate large databases, to issues on how to efficiently integrate parallel mining algorithms to DBMS parallel query processing. Our goal is divided in three main areas: (i) distributed database design, (ii) parallel data mining algorithms, and (iii) database clusters, outlined as follows.

Distributed database design. We are developing and evaluating fragmentation and allocation techniques to obtain high performance through data parallelism. This includes a methodology to find adequate fragmentation techniques and fragment sizes, as well as algorithms to define fragments and allocate those fragments among cluster nodes while deciding upon replication.

Parallel data mining algorithms. Implementation of data mining algorithms in high-performance parallel and distributed computing environments is crucial for ensuring system scalability and interactivity as datasets grow in size and complexity. We have implemented several DM algorithms such as: a parallel k-NN algorithm with Genetic Algorithm optimization, a parallel neural network implementation, and a fuzzy rule based classification. We have evaluated these parallel implementations with benchmark problems obtaining good results encouraging experiments with new applications and implementations of other DM techniques.

Database cluster. Parallel query processing is a key issue to efficiently access data stored in database systems. However, parallel database systems often require parallel machines and parallel DBMS software, which contribute to their high cost. In addition, database migration can be also costly. A database cluster that uses a sequential off-the-shelf DBMS running on each node of a PC-cluster as a *black-box* component is a cheaper alternative and respects the autonomy of the database. We have developed ParGRES, a database cluster middleware and evaluated it with typical queries from knowledge discovery activities, such as OLAP (on-line analytical processing) queries. Our results with a 32 dual node machine show super linear speed-up in many scenarios.

In addition to these three research tracks we also work on the combination of these techniques by integrating data and text mining into DBMS queries. Several experiments are under development considering our own developed tools, and open software as well as commercial products. Our research group has fostered contact with groups related to target applications such as environmental, business and bioinformatics. Collaborating research activities within these areas have produced initial data cleaning, data analysis and sequential data mining to prepare for the next phase where parallel data processing on such databases is expected.

The purpose of this paper is to briefly survey the main achievements of the ClusterMiner project. Mostly we will reference our own published work to emphasize our contribution. Details of the tasks executed, including the mapping of each task with the original proposal, the use of funds, papers published, difficulties encountered, students involved, approximation with companies and international cooperation can be found in the project technical reports, available in http://clusterminer.nacad.ufrj.br/.

The rest of this paper is organized as follows. In Section 2, we address the main issues in the distributed design of databases, which are the basis for data parallelism. Section 3 presents our proposal for parallelizing several data mining algorithms with the performance improvements obtained. Section 4 describes the database cluster prototypes we have developed and presents some of several experimental evaluations developed along this project. We discuss the data mining and database systems coupling issues in Section 5, including specific issues of text mining. In Section 6 we discuss the work we have been doing with potential applications for ClusterMiner environment tools. Finally, the conclusion of this paper and research tasks for the next two years are presented in Section 7.

2 Distributed database design

The design of distributed databases is a crucial task to improve the performance of applications in a distributed environment, and involves making decisions on the fragmentation and placement of data across the sites of a computer network. The first phase of the distribution design in a top-down approach is the fragmentation phase, which clusters in fragments the information accessed simultaneously by applications. Most distribution design algorithms propose a horizontal or vertical class fragmentation. However, the user has no assistance in the choice between these techniques. In this project we present a detailed methodology for driving the choice between the horizontal and the vertical partitioning techniques, or even the combination of both, in order to assist distribution designers in the fragmentation phase of databases. Experiments using our methodology [Baião et al., 2004] [Florentino, 2003] have resulted in fragmentation schemas offering a high degree of parallelism together with an important reduction of irrelevant data.

2.1 Data fragmentation

In the design of distributed databases, the fragmentation phase is responsible for analyzing the most frequent data access pattern and for indicating the best way to group data into fragments that will be allocated to the distributed nodes. The database

fragmentation is considered a NP-Hard problem, thus requiring heuristic approaches to provide a design that decreases the volume of unnecessary data accessed by applications while increases data locality. We have developed the ODARA methodology, which implements heuristics to guide the distribution designer in the fragmentation phase of the design of distributed databases in the logical level of data representation. The ODARA methodology contemplates the object oriented, object-relational and relational data models, and may be used in a broad spectrum of existing environments in current organizations. Evaluations with ODARA generated fragmentation schemas for the Bucky benchmark with better performance when compared to results from other relevant works in the literature can be found in [Florentino, 2003].

2.2 Data allocation

Allocation algorithms are typically used to find a data distribution among the sites of the network such as to minimize the execution cost of the application. Due to the huge number of possible solutions, the allocation problem is an NP-Complete problem. In this project, we proposed two algorithms for fragment allocation in distributed database systems: Aloc – a heuristic algorithm – and GRADA – an algorithm based on the GRASP meta-heuristic [Wildemberg, 2004a],[Wildemberg et al., 2003].

Through simulations performed on top of the TPC-C benchmark, it was possible to identify scenarios where Aloc found the optimal allocation solution and other scenarios where Aloc obtained a reduced allocation cost when compared to related work [Wildemberg et al., 2003]. GRADA obtained better allocation schemas than the Aloc in scenarios with a large number of sites [Wildemberg et al., 2004a]. The XAloc software tool [Wildemberg et al., 2004a] was developed to evaluate fragment allocation scheme according to different allocation algorithms with three different cost functions embedded in the system.

2.3 Using theory revision in distributed database design

We have developed a framework to handle the fragmentation problem of the design of distributed object databases. One of the components of this framework is the Theory REvisioN on the Design of Distributed Databases (TREND3) [Baião et al., 2003] which automatically revises the heuristic algorithm of the analysis phase of the distributed database design (called analysis algorithm) through the use of the Inductive Logic Programming system FORTE. The analysis algorithm decides the fragmentation technique to be used for the distribution design of database objects. The Prolog implementation of the analysis algorithm is provided as the initial domain theory, and fragmentation schemas with previously known performance, obtained from experimental results, are provided as the set of examples.

We compared the costs of the resulting fragmentation schema obtained from the initial and the revised versions of the analysis algorithm on the OO7 benchmark application, after executing the vertical and horizontal fragmentation algorithms. Figure 1 shows the cost of executing each OO7 query. Our results show the effectiveness of the TREND3 approach in automatically improving the analysis algorithm and obtaining a

new version that produced a fragmentation schema that reduced the cost (i.e., increased the performance) of the OO7 application in 38% [Baião et al., 2003]0.

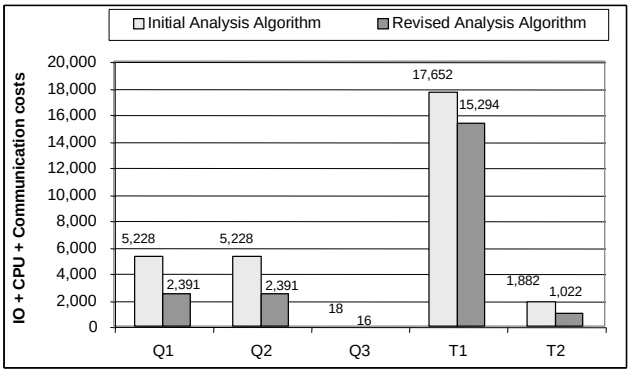


Figure 1: Comparing the costs of the fragmentation schemas

3 Parallel Data Mining Techniques

The huge amount of data generated by Data Warehousing and database transactions has led Data Mining algorithms to parallel implementations. In this section we describe some aspects of three DM algorithms, among others, 0that have been implemented in the ClusterMiner project (see reports in http://clusterminer.nacad.ufrj.br/): (i) a parallel k-NN algorithm with Genetic Algorithm optimization, (ii) a parallel neural network implementation, (iii) a fuzzy rule based classification. Parallel techniques are discussed in the following sub-sections where the results obtained show very good performance improvements.

3.1 K-NN Fuzzy with Genetic Algorithm Optimization

We developed a k-NN fuzzy classifier, optimized with a Genetic Algorithm and obtained results that corroborate the hypothesis that this approach can minimize CPU time, with a good/excellent sorting ratio 0Rosa, 2003; Rosa et al., 2003]. The machine available in this work was an academic pc-cluster of the NACAD/COPPE/UFRJ High Performance Computing Laboratory. This machine contains 8 nodes (Intel Pentium III with 256 MB memory) each connected by a Fast Ethernet network).

The parallel implementation considered the data parallelism approach that keeps a copy of the entire labeled samples, with the internal variables and functions, in each processing node partitioning training data set among nodes. This approach ensures that all values needed during the training phase like K, m and variable weights, are locally available, reducing the number of messages passing among nodes and the algorithm synchronization. The implementation of data parallelism uses a control node that performs genetic operations, realizes chromosome encoding and decoding, broadcasts the values of the parameters and summarizes the interested value (total of right classes). All nodes start with same internal parameters but with different subsets of training data. During training phase of KNN-Fuzzy method [Rosa et al., 2003]0, each node passes its training data subset producing partial errors that must be combined with partial errors produced by other nodes, generating an overall error. This error is used to update the

weights of the variables in each node, until the procedure reaches the minimal acceptable overall error.

We have evaluated this parallel kNN-fuzzy implementation with 1998 KDD´s cup, Sisyphus dataset, which represents a private company of life insurance data warehouse. After the data preparation, the database size becomes 64 columns and 130143 strings. The class label has only two labels. The application of Amdahl's Law for performance evaluation shows a 99,6% of parallelization coefficient for the developed algorithm. This is explained by the large time expended by KNN-Fuzzy procedure in the overall application time (in a serial evaluation the total computation time is equal 105057 s and the KNN-Fuzzy procedure evaluation time is equal 104634 seconds.

3.2 Neural Networks

Neural network model is inherently parallel with basic independent units performing local calculations. The current implementation [Costa et al., 2004]0 uses a data parallelism approach, which keeps a copy of the entire neural architecture in each processing node partitioning training data set among nodes. This approach ensures that all values needed during the training phase, like output calculation and error back propagation, are locally available, reducing the communication among nodes and the algorithm synchronization. In distributed memory machines this approach can lead to significant performance gains.

The parallel implementation uses a control node (node 0) that gets the training data set, normalizes and distributes it among the processing nodes. All nodes are started with same internal parameters but with different subsets of the training data. During training phase, each node passes its training data subset producing partial errors that must be combined with partial errors produced by other nodes, generating an overall error. This error is used to update the connection weights in each node, until the procedure reaches the minimal acceptable overall error. It can be pointed out that each node only broadcasts its partial error to other nodes at this time. All other calculations involve local data and can be made without synchronization. This drastically reduces the communications needs.

We have evaluated this parallel neural network implementation with KDD´s cup Sisyphus dataset. The preprocessing phase of this work got an improved data set with attributes and registers that properly represent data to train the neural network. Attributes with discrete and continuous values were evaluated for columns and registers removal, using graphical and statistical algorithms for columns reduction. The resulted data set presented 64 attributes and 130,143 registers. The network suitable for modeling those data has 64 neurons on input layer, 136 neurons on hidden layer and 1 neuron on output layer. The parallel neural network application reads the data set and splits it into training and testing sets, with the percentages of 70% and 30% respectively. The training set is distributed for the processing nodes before the learning phase of the algorithm starts. At the end of training phase the control node tests the obtained model with the testing set.

Applying Amdahl's Law to evaluate the parallel application performance, it can be showed that 99.3% of the serial application can be parallelized. This is due to the time spent by the training phase with respect to the whole time consumption of the application.

3.3 Fuzzy Logic

Fuzzy rule based classifier has shown good results in serial implementation. This classifier is divided in two steps: (i) derive a fuzzy rule based classifier to each variable, and (ii) aggregate the partial conclusions of each classifier into a global conclusion. This approach allows a simple parallel implementation, since each single-variable classifier (or a set of them) can be implemented in a different processor in a parallel architecture, and partial conclusions are synchronized and processed by a master processor.

In the parallel implementation [Pereira et al., 2004]0, a master processor selects inputs and distributes the inputs among the available nodes. In the learning phase, at each node, a fuzzy rule base weight matrix is computed for each sub-model for the input variables, which is assigned to that node. In the processing or testing phase, the rule base weight matrix for each sub model is used to compute a partial conclusion on class descriptions. All partial conclusions are synchronized in the master processor and final conclusions are computed by aggregation of partial conclusions of each sub-model (see Figure 2).

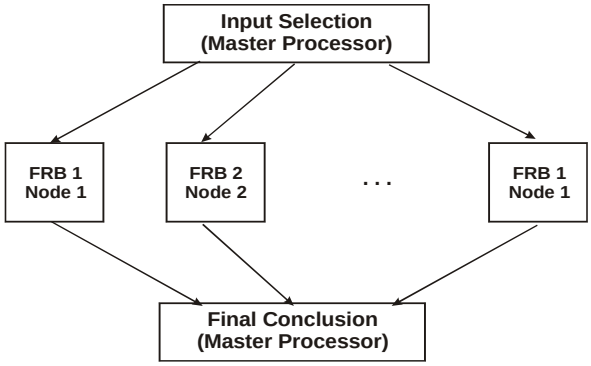


Figure 2. The parallel implementation workflow

In order to evaluate the scalability of the algorithm for larger problems, a set of synthetic problems were generated. Each data set was created using independent normally distributed random variables with different means for each class. All synthetic problems were generated using two classes, distributed with the same *a priori* probability. The number of variables was varied as $\{16, 32, 64, 128, 256\}$ and the number of registers of the data set was varied as $\{10, 20, 50, 100\}$ thousands of registers. All combinations of number of variables and number of registers were considered, resulting into 20 synthetic data sets.

The processing time for each one of the synthetic data set, running in a single processor machine, is shown in Figure 3. All experiments used five membership functions for the fuzzy partition of each input variable. The processing time increases

almost linearly with the number of variables and the number of registers. For instance, the time for the 256 variables data set with 50 thousands of registers was 85.544s, while the same number of variables and 100 thousand was 196.971s. The execution time for the 128 variables data set and 50 thousand of registers was 43.276s. Considering the size scalability described above, the speed-up study was carried out with the 100 thousands registers datasets.

Several experiments were performed in order to evaluate the speed-up analysis (Figure 4). Despite the communicating overhead due to data distribution, the performance of the algorithm is primarily influenced by the processing of base rule. Speed-up is calculated against serial execution time of the algorithm. It can be concluded that the effect of the parallel implementation is more efficient as the number of variables increases. For 16 variables the efficiency speed-up does not justify the parallel implementation. However for 256 variables the speed-up efficiency increases almost linearly with the number of processors.

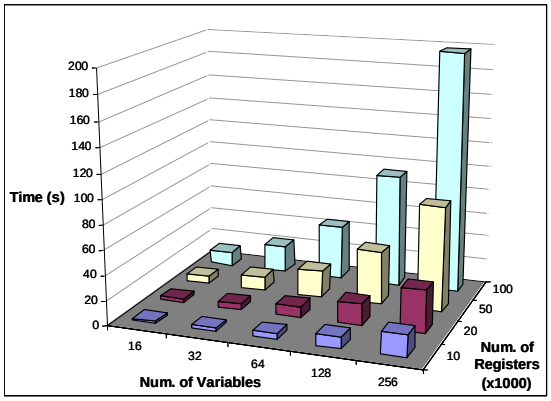


Figure 3. Size scalability analysis

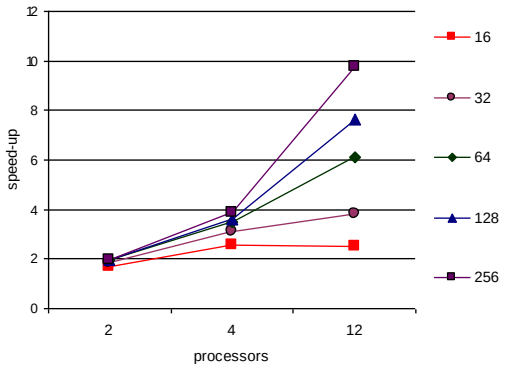


Figure 4. Speed-up analysis

4 Database Clusters

Clusters of PC servers appear as a cost-effective alternative to parallel database servers. Recently, the database cluster approach has gained much interest for various database applications. A database cluster is a set of PC servers interconnected by a dedicated high-speed network, each one having its own processor(s) and hard disk(s), and running an off-the-shelf DBMS. Similar to multiprocessors, various cluster system architectures are possible: shared-disk, shared-cache and shared-nothing. Shared-nothing (or distributed memory) is the only architecture that does not incur the additional cost of a special interconnect. Furthermore, shared-nothing can scale up to very large configurations. In the ClusterMiner project, we strive to exploit a shared-nothing architecture.

Each cluster node can simply run an inexpensive (non parallel) DBMS. In our case, we use the PostgreSQL DBMS, which is freeware. Furthermore, the DBMS is used as a "black-box" component. In other words, its source code is considered not available and cannot be changed or extended to be "cluster-aware". Therefore, extra functionality like

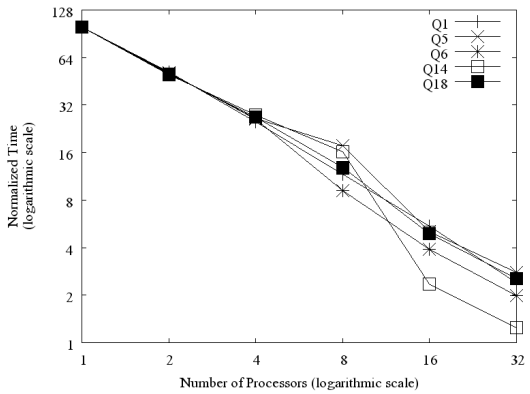
parallel query processing capabilities must be implemented via middleware [Vieira et al., 2003] . Our database cluster solution, called ParGRES [Mattoso et al., 2005a], along with its prototype implementation [Mattoso et al., 2005b], represents a generic and efficient solution to build the parallel application server devised for ClusterMiner.

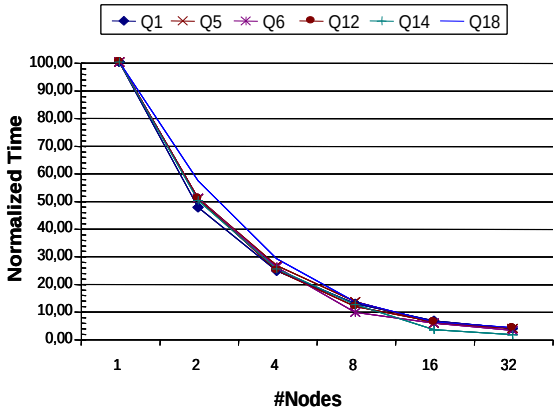
On-Line Analytical Processing (OLAP) applications typically access large databases using heavy-weight read-intensive queries. These queries belong to typical activities in knowledge discovery process. Particularly summarization and sorting queries are also part of data mining tasks. OLAP query processing in a database cluster is addressed in [Lima et al., 2004b, Mattoso et al., 2005a, Mattoso et al., 2005b]. The approach which we refer to as simple virtual partitioning (SVP) consists in fully replicating a database along a set of sites, and breaking each query in sub-queries by adding predicates. Each DBMS receives a sub-query and is forced to process a different subset of data items. Each subset is called a "virtual partition". Such strategy allows for greater flexibility on node allocation for query processing than physical (static) data partitioning. A preventive replication protocol, which scales up well in cluster systems, could be used to keep copy consistency.

We address the partition size determination problem by using an adaptive approach that dynamically tunes partition sizes. We propose adaptive virtual partitioning (AVP) which is completely DBMS-independent and uses neither database statistics nor query processing time estimates. AVP avoids full table scans on huge database tables in addition to parallel processing. It is also easy to implement. To validate our approach, we implemented AVP in a Java prototype and ran experiments on a 32-node cluster using PostgreSQL. The results show linear and sometimes super-linear speedup for many tested OLAP queries (Figure 5-a). In the worst cases, almost linear speedup is achieved, which is excellent considering the simplicity of AVP [Lima et al., 2004a].

We also evaluated AVP in the presence of non uniform load distribution. To address data skew we implemented an innovative load balance algorithm combined to the AVP strategy. Figure 6 shows normalized execution times for the OLAP queries specified by the TPC-H benchmark. These queries contain typical operations found in data mining activities such as sum, count and group by. Performance results (Figure 5-b) have shown excellent speed-up and load balance [Lima et al., 2005]. These results are part of the international collaboration with Patrick Valduriez from Inria, France, supported in part by the ClusterMiner project.

(a)





(b)

Figure 5. TPC-H queries execution normalized times with (a) uniform values and (b) skewed values

5 Data Mining and Database System Coupling

Many data mining implementations have already used a DBMS. However, most of implementations use DBMS only to issue queries that are going to be processed by a client machine. The problem with loosely-coupled solutions is that DBMS facilities are underemployed. They treat the DBMS simply as a container from which data is extracted directly to the main memory of the computer responsible for running the data mining algorithm, just before the main execution begins.

Therefore, the idea of tightly-coupled integration between data mining operations and a DBMS is to execute user-defined computation within databases. This integration provides security (the data is controlled by the DBMS), interaction with the data miner (the user may query partial results), among others. Also, updates in the original data set may be incrementally propagated to the mine database relation. One way of implementing this integration is through DBMS query languages.

In the ClusterMiner project we are evaluating solutions that integrate typical mining data accesses with a coupled use of DBMS, addressing performance issues with database cluster parallel processing (section 4). The main goal of this approach is to combine code flexibility and simplicity provided by current commercial as well as open software DBMS with the efficiency of parallel processing.

In this section we present the several initiatives evaluated by the ClusterMiner team. Initially, we present typical SQL queries that can help data mining tasks and show how these query executions can be parallelized, for example by a our database cluster. Section 5.2 briefly discusses how we are coupling text mining algorithms based on summarizations to XML documents query processing using DBMS. We have also evaluated commercial solutions such as an industrial standard initiative to access DBMS from data mining algorithms, presented in section 5.3. We have obtained good results but still without parallel support. Finally, in section 5.4 we describe the implementation of a complete classification algorithm taking advantage of DBMS fuzzy queries.

5.1 Typical DBMS queries in data mining tasks

Most queries issued in data mining tasks, deals with two kinds of operations: grouping (GROUP BY) and sorting (ORDER BY), when working with continuous attributes. In the following SQL example, *attribute* represents one of the predictor attributes, *class* represents the target attribute.

```
SELECT attribute, class, COUNT(*)
FROM mine_relation
GROUP BY attribute, class
```

Database clusters can process this kind of query very efficiently taking advantage of parallel PC cluster resources. In distributed memory architectures such as PC clusters, each node can independently compute counts on the records stored locally. In the next phase, a coordinator process can consolidate these counts, and then return the result back to the user.

We have evaluated these types of query in several experiments through TPC-H (OLAP queries) benchmark using ParGRES, our database cluster prototype on top of PostgreSQL, an open-source DBMS [Lima et al., 2004a; Lima et al., 2004b; Lima et al., 2005]. Our results with a 32 dual node machine show super linear speed-up in many scenarios (section4), evidencing the parallelism that can be obtained. In the next phase of the project we intend to use these kinds of queries in data mining algorithms presented at section 3.

5.2 Text Mining and DBMS queries

Text mining is concerned with the management of very large document collections and the extraction of hidden knowledge from text-based data. The volume of textual data in semi-structured format, particularly in XML, is expected to grow tremendously in the next few years, resulting in huge databases of documents, such as news, patent documents, genome databases, etc. Consequently, the support for the management of XML data inside database systems is rapidly growing both in the academy and in companies. Also, these massive XML databases may contain knowledge in textual format that extrapolates their original purpose. Currently many DBMS can process, through XQuery, queries on XML documents [Curotto and Ebecken, 2003]0.

One of the main Text Mining tasks is Text Categorization (TC), which is the supervised-learning assignment of labels to documents in a collection. Today, the task of categorizing large collections of documents stored in a DBMS is done with little (if any) integration between the TC algorithm and the DBMS [Bezerra and Mattoso, 2004].

In this activity of the ClusterMiner project, we propose a new way in which TC algorithms can be constructed. We have defined a database primitive for the TC task using XQuery. A database primitive can be seen as a core operation that is data-intensive. Our database primitive aims at providing statistical summaries extracted from XML documents stored in a DBMS. These statistical summaries can be used by TC algorithms implemented on top of the DBMS or a database cluster, hence scaling these algorithms to handle large volumes of textual data. The main goal of our database

primitive is to move some of the computational processing of the TC task into the DBMS query processing capabilities.

We have evaluated our database primitive with a Naïve Bayes algorithm on top of an XML database engine to construct a predictive model built from the new Reuters collection of XML documents. Currently, we are investigating two issues in the production of statistical summaries in an XML format. First, XML data contains a lot of verbose, which increases the amount of space needed to store the results of our primitive. Second, the results must be represented in a standard format in order to be used by many text mining applications. To deal with the first problem, are investigating the use of XML compressors (like XMill) to reduce the amount of space needed to store the results of our primitive. To deal with the second problem, we are using PMML (Predictive Model Markup Language) to represent the results of the primitive [Bezerra and Mattoso, 2004; Bezerra et al., 2005].

In addition to the evaluation of text mining with DBMS we are also investigating new algorithms for text categorization. TextRISE [Pina and Zaverucha, 2004]0 is a modification of RISE (a well-known multi-strategy algorithm that combines the best characteristics of rule induction and instance-based learning in a single algorithm) for information extraction. We developed the SUNRISE algorithm [Pina and Zaverucha, 2004],0 which achieves comparable accuracy to that of the RISE algorithm but in a lower average running time.

5.3 Data mining tasks in commercial DBMS systems

Using Database Management Systems (DBMS) technology with Data Mining (DM) activities are becoming very popular. It is efficient, inexpensive, safe, adequate, comfortable and reliable to develop an application in the same environment that supports and manages all data and knowledge.

To achieve this goal, Microsoft released the Object Linking and Embedding Database for DM (OLE DB for DM) specification, a protocol based on the SQL language that provides software vendors and application developers with an open interface to more efficiently integrate data mining tools and capabilities into line-of-business and e-commerce applications. About 55 independent software vendors (ISVs) participated in the elaboration of this specification.

OLE DB for DM provides an industry standard for DM so that different DM algorithms from various DM developers can be easily plugged into user applications and specifies the Application Programming Interface (API) between DM consumers (applications that use DM features) and DM providers (software packages that provide DM algorithms). This approach seems to be a promising solution for the integration of DM and DBMS technologies. Using this technology, a Simple Naive Bayes Incremental classifier was implemented supporting numeric input attributes, multiple prediction attributes and incremental update of data. Computational experiments using real word data sets were used to evaluate the results obtained by this classifier [Evsukoff and Ebecken, 2003]0.

5.4 Automatic Generation of Fuzzy Queries

Typical DBMS queries, expressed in query languages such as SQL have several limitations to deal with uncertain and vague information. Moreover, standard queries in DBMS are processed using Boolean logic resulting in a non-ordered sequence of registers. Many applications, e.g. database marketing, need a ranking of the registers to perform target selection. Fuzzy queries and Fuzzy Queries Languages can deal with uncertain and vague query information and also return an ordered sequence of registers.

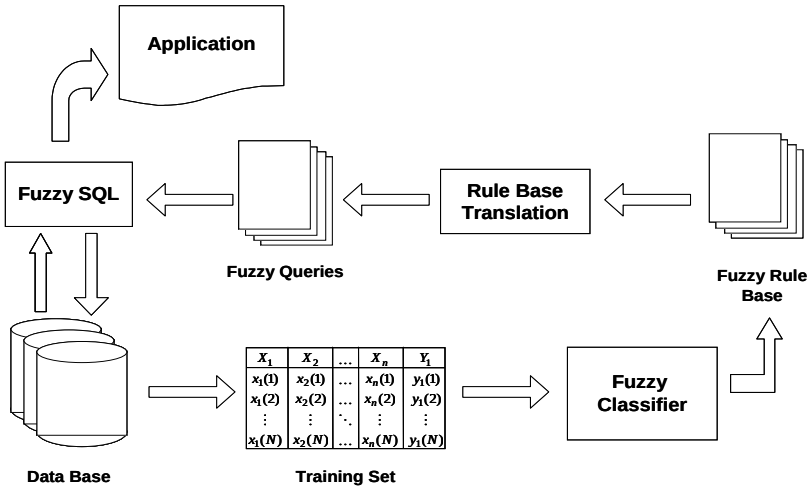


Figure 6. Automatic generation of fuzzy queries

The generation of fuzzy queries by supervised learning methods has been subject of several studies. An original methodology (Figure 6) has been developed [Branco, 2004] to generate a set of fuzzy queries for classification tasks. Those queries are obtained from a set of fuzzy rules that are learned from a Fuzzy Classification System and can select the members of each class from the database.

A simple approach to evaluate the weighted fuzzy queries is presented and some results are examined using a commercial tool. A fuzzy query simplification approach is also derived, showing the same classification accuracy with much better interpretability characteristics. The results show that the proposed methodology may improve target selection in a database market application.

This methodology is a time consuming task, however, it can be directly extended through parallel implementations since the parallelization of this fuzzy classification technique has already been developed (see section 3.2). In addition, the database cluster middleware can be adapted to parallelize the execution of fuzzy SQL queries providing high performance.

6 Applications

This section describes potential applications areas where the techniques described at the previous sections can be applied. We have experimented sequential data mining algorithms on applications of environmental and commercial areas. We also describe collaboration initiatives with bioinformatics groups, however with these applications we

are still structuring and gathering data. These experiences are briefly presented as follows.

Insolvency Detection in a Telecom Operator. Insolvency impacts all companies, generating injuring processes and flow of investments, yielding revenue losses and, specially, customer losses. The involuntary actions of insolvency are semi-controlled events that can be analyzed and classified. After implementing and executing cluster and classification models, there are many opportunities to develop and deploy a prediction model to prevent the involuntary events of insolvency and detection of fraud actions. However, frauds happen in different ways, and due to its intentional characteristic, they can be more volatile in a time series, changing the form and the mode of its occurrence, making the events, more difficult to cluster, classify and predict. We have developed [Evsukoff et al., 2004]0 a system for insolvent customer detection, using unsupervised learning neural network-models. In [Evsukoff et al., 2004]0 we present a case study on the development of a model to identify and prevent insolvency events in a telephone operator database. The model identifies the defaulters' profile, indicating the variables related to the insolvency.

Coast Management. Environmental data often need to be analyzed in order to obtain information necessary for environmental management decision. The main task today is to shift what is natural and atrophic variability and the assessment of trophic status to forecast the future ecosystem behavior. Sometimes it is necessary to use the data to build a model of the environmental processes that we want to manage. In other cases, we must identify and understand the interrelationships of different physical, chemical and biological parameters. The case studied is the algal community growth in coastal upwelling area of Cabo Frio Island at Rio de Janeiro state, southeastern of Brazil as an indicator since this place is becoming an operational support base of oil drilling companies [Evsukoff and Ebecken, 2004]0. A machine learning approach is used to elicit regularities and dependences that include both numerical and logical conditions.

Traffic Information Systems. This application [Pinheiro et al., 2003]0 presents a fuzzy system for pattern recognition in a real application: the selection of traffic information messages to be displayed in Variable Message Signs located at the main routes of the city of Rio de Janeiro. In this application, flow and occupancy rate data is used to fit human operators' evaluation of traffic condition, which is currently done from images of strategically located cameras. The fuzzy rule-base mining is presented considering the symbolic relationships between linguistic terms describing variables and classes. The application presents three classifiers built from data.

Geochemistry Exploration. In this study [Gama et al., 2004]0, fuzzy reasoning numerical techniques were applied to integrate surface geochemical (headspace C1 to C6+ concentrations from soil samples) and geologic data in a Sub-Andean sedimentary basin. A methodology is proposed to compute anomalous regions combining Fuzzy c-Means clustering and fuzzy classifiers. The results of the proposed approach have allowed a good definition of anomalous areas, taking into account all the geochemical parameters in an integrated way.

Key Account Management. Key Account Management is the strategic marketing approach that provides an effective, practical and rather simple method for companies interested in increasing their profits by adequate customer and relationship management.

In companies which data is decentralized in business units, the implementation of a Key Account program passes through the accounts selection problem, and through the implementation process, that can be seen as a Knowledge Discovery in Database process where the goal is to search for accounts under given restrictions in the marketing utility function. This work [Nahm and Mooney, 2002]0 presents a case study based on the implementation of a KAM program in a Brazilian Insurance Company, where a naive KDD approach supported an information system construction.

Dynamic Systems Identification. This application [Evsukoff et al., 2004]0 presents an algorithm for identification of fuzzy recurrent models of non-linear dynamic systems. The identification algorithm is based on a general purpose genetic algorithm. The resulting recurrent fuzzy system is encoding into a fuzzy finite state automaton in which the linguistic terms of the fuzzy model are the states and rule base weights are transition possibilities. The identification algorithm is tested against benchmark identification problems found in literature.

Bioinformatics. *In-silico* scientific experiments encompass multiple combinations of program and data resources. Each resource combination in an execution flow is called a scientific workflow. In bioinformatics environments, program composition is a frequent operation, requiring complex management. We have combined metadata support with Web services within a framework that supports scientific workflows. We have used this framework with a real structural genomic workflow, showing its viability and evidencing its advantages [Cavalcanti et al., 2004]. In addition, we used data parallelism to improve the performance of typical bioinformatics workflows [Meyer et al., 2004]. Our main efforts were in managing bioinformatics workflows which generate specific datasets to be mined.

7 Conclusion and Future Work

In this report paper we have outlined some research results from the ClusterMiner project. We were able to make advances in the areas of parallel database clusters, parallel data mining algorithms and coupling of mining with database techniques, following our original goals. We have made progress both in obtaining theoretical results as well as implementing and evaluating new parallel techniques, as planned for our high performance environment. We successfully accomplished the development of key components for the ClusterMiner environment, such as: the parallel execution of typical data mining tasks, the parallel implementation of important data mining algorithms and prototypes that combine data, text mining with parallel database systems, parallel data mining with database systems, among other DM-DB couplings. Results show very good performance improvements through parallel processing. Therefore, we have successfully achieved the goals devised for the project.

We have also been in contact with potential applications of the ClusterMiner environment. This contact aims at gathering data, analyzing data, structuring data and performing some sequential data mining analysis, which has produced important results in publications. Future work include (i) fine tuning and evaluation of other applications with our database cluster, (ii) coupling and evaluations of parallel data mining algorithms with our database cluster intra-query parallelism, (iii) implementations of Web mining tools and text mining algorithms, (iv) continue the developments on data

preparation and analysis of potential applications of the ClusterMiner environment, (v) integration of software components into the ClusterMiner environment, and (vi) evaluation of target applications with ClusterMiner tools.

Acknowledgement. The authors are grateful to CNPq for the financial support for this project.

References

BAIÃO, F., MATTOSO, M. L. Q., SHAVLIK, J., ZAVERUCHA, G. Applying Theory Revision to the Design of Distributed Databases In: International Conference on Inductive Logic Programming (ILP 2003), 2003, Szeged. Lecture Notes in Artificial Intelligence. Springer-Verlag, 2003. v.2835. p.57 – 74

BAIÃO, F., MATTOSO, M. L. Q., ZAVERUCHA, G. A Distribution Design Methodology for Object DBMS. Distributed and Parallel Databases. Kluwer Academic Publishers, v.16, n.1, p.45 - 90, 2004.

BEZERRA, E. MATTOSO, M. L. Q., On the integration of Text Mining and Database Systems, In: ClusterMiner Technical Report RT-007-04 (http://clusterminer.nacad.ufrj.br/), 2004.

BEZERRA, E. XEXÉO, G. MATTOSO, M. L. Q., A Database Primitive for Classification in Text Mining, submitted to ACM SAC Data Mining Track 2005.

BRANCO, A. C. S. Fuzzy Queries Generation for Data Mining. D.Sc. Thesis, COPPE/UFRJ, Brazil, 2004.

CAVALCANTI, M. C., TARGINO, R., BAIÃO, F., ROSSLE, S., BISCH, P., PIRES, P. F., CAMPOS, M. L. M., MATTOSO, M. L. Q. Managing Structural Genomic Workflows using Web Services. Data & Knowledge Engineering., 2004. Available at http://authors.elsevier.com/sd/article/S0169023X04001120

COSTA, M. C A., EVSUKOFF, A., EBECKEN, N. Data Mining High Performance Computing Using Neural Networks. In: ClusterMiner Technical Report RT-008-04 (http://clusterminer.nacad.ufrj.br/), 2004.

CUROTTO, C. L. Integrating Data Mining Resources and Database Managers. D.Sc. Thesis, COPPE/UFRJ, Brazil, 2003.

CUROTTO, C. L., EBECKEN, N. F. F. Evaluating the Scalability of Data Mining Provider Classifiers, Data Mining IV – Fourth International Conference on Data Mining, WIT Press, ISBN 1-85312-806-6, Rio de Janeiro, 2003, p. 651-660.

EVSUKOFF, A. EBECKEN, N. F. F. Mining fuzzy rules for a traffic information system. V. Palade, R. J. Howlett e L. Jain (editors). *Proceedings of the Knowledge – based Intelligent Information and Engineering Systems – KES2003*, LNAI 2773, 2003, p. 237-243.

EVSUKOFF, A., COSTA, M. C A., EBECKEN, N. Parallel Implementation of Fuzzy Rule Based Classifier. In: VECPAR2004 - High Performance Computing for Computational Science, Valencia. Lecture Notes in Computer Science. Springer Verlag. 2004

EVSUKOFF, A., EBECKEN, N. F. F.. Identification of Recurrent Fuzzy Systems with Genetic Algorithms. *Proceedings of the International Conference on Fuzzy Systems – FUZZ'IEEE 2004*, 2004.

EVSUKOFF, A., GONÇALVES, F. T. T., BEDREGAL, R., EBECKEN, N. F. F, Fuzzy Classification of Surface Geochemistry Data Applied to the Determination of HC Anomalies. Joint 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems SCIS & ISIS 2004

FLORENTINO, P. V. ODARA: Methodology for Database Fragmentation Design. M.Sc. Dissertation, COPPE/UFRJ, Brazil, 2003.

GAMA, C. A., EVSUKOFF, A., MOTTA, J. P. A Naive KDD Approach in Key Account Management Framework: A case study. *Data Mining 2004*. Wessex Institute of Technology, Malaga, 2004.

HUSCHKA Jr, E. R., HRUSCHKA, E. R., EBECKEN, N. F. F. A Feature Selection Bayesian Approach for Extracting Classification Rules with a Clustering Genetic Algorithm. Applied Artificial Intelligence. , v.17, n.5, p.489 - 506, 2003.

LIMA, A. A., ESPERANÇA, C. MATTOSO, M. L. Q., Efficient Processing of Heavy-Weight Queries in Database Clusters In: ClusterMiner Technical Report RT-001-04 (http://clusterminer.nacad.ufrj.br/)

LIMA, A. A., MATTOSO, M. L. Q., VALDURIEZ, P. Adaptative Virtual Partitioning for OLAP Query Processing in a Database Cluster In: Simpósio Brasileiro de Banco de Dados, 2004, Brasília. XIX Simpósio Brasileiro de Banco de Dados. SBC, 2004a.

LIMA, A. A., MATTOSO, M. L. Q., VALDURIEZ, P. Load Balancing of OLAP Queries in a Database Cluster, ICDE 2005, 2005, Int. Conf. Data Engineering, IEEE.

LIMA, A. A., MATTOSO, M. L. Q., VALDURIEZ, P. OLAP Query Processing in a Database Cluster In: Euro-Par 2004, Pisa. Lecture Notes in Computer Science, Springer Verlag, 2004b.

MATTOSO, M., ZIMBRÃO, G. ,LIMA, A., BAIÃO, F., BRAGANHOLO, V., AVELEDA, A., MIRANDA, B., ALMENTERO, B., NUNES, M. ParGRES: uma camada de processamento paralelo de consultas sobre o PostgreSQL. In: WSL2005 - 6 Workshop Software Livre 2005, 2005a, Porto Alegre. 6 Fórum Internacional Software Livre, 2005. p. 259-264.

MATTOSO, M., ZIMBRÃO, G. ,LIMA, A., BAIÃO, F., BRAGANHOLO, V., AVELEDA, A., MIRANDA, B., ALMENTERO, B., NUNES, M. ParGRES: Middleware para Processamento Paralelo de Consultas OLAP em Clusters de Banco de Dados. In: Simpósio Brasileiro de Banco de Dados - Sessão de Demos, 2005, Uberlândia. Sessão de Demos em Bancos de Dados, 2005b.

MEYER, L. A. V. C., ROSSLE, S., BISCH, P., MATTOSO, M. L. Q. Parallelism in Bioinformatics Workflows In: VECPAR2004 - High Performance Computing for Computational Science, Valencia. Lecture Notes in Computer Science. Springer Verlag. 2004.

NAHM, U. Y., MOONEY, R. J. Text Mining with Information Extraction. In: Proceedings of the Spring Symposium on Mining Answers from Texts and Knowledge Bases. AAAI Press, Stanford, CA (2002) 60□ 67

PEREIRA, G. C, EVSUKOFF, A., COUTINHO, R., EBECKEN, N. F. F. Coastal Environmental Management by Fuzzy System Modeling. *Proceedings of the International Conference on Fuzzy Systems – FUZZ'IEEE 2004*, 2004.

PINA, A., ZAVERUCHA, G. The SUNRISE Algorithm: Improving the Performance of the RISE Algorithm. 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Pisa, Lecture Notes in Artificial Intelligence, Springer-Verlag. 2004.

PINHEIRO, C. A. R., EVSUKOFF, A., EBECKEN, N. F. F. Identifying the insolvency profile in a telephony operator database. Data Mining 2003, Wessex Institute of Technology, Rio de Janeiro, 2003.

ROSA, J.L.A. Classificação de Dados através da Otimização do Método KNN-Fuzzy em Ambiente de Computação Paralela. 2003. Tese (Engenharia Civil) - Universidade Federal do Rio de Janeiro

ROSA, J.L.A., EBECKEN, N.F.F., COSTA, M.C.A. Towards on an Optimized Parallel KNN-FUZZY Classification Approach, Data Mining IV – Fourth International Conference on Data Mining, WIT Press, ISBN 1-85312-806-6, Rio de Janeiro, 2003, p. 81-88.

RUBERG, N., RUBERG, G., MATTOSO, M. L. Q. Digging Database Statistics and Costs Parameters for Distributed Query Processing In: International Conference on Cooperative Information Systems - CoopIS 2003, 2003, Catania. Lecture Notes in Computer Science. Springer-Verlag, 2003. v.2888. p.301 – 318

VICTOR, A.O., MATTOSO, M. L. Q. Distributed Query Routing in a Cluster of Autonomous Databases, In: ClusterMiner Technical Report RT-002-04 (http://clusterminer.nacad.ufrj.br/), 2004.

VIEIRA, H., RUBERG, G., MATTOSO, M. L. Q. Xverter: Querying XML Data with OR-DBMS In: ACM Fifth International Workshop on Web Information and Data Management (WIDM'03). 2003. v.1. p.37 – 44

WILDEMBERG, M. Fragment Allocation Techniques in Distributed Database Design, M.Sc. Dissertation, COPPE/UFRJ, Brazil, 2004.

WILDEMBERG, M., PAULA, M., BAIÃO, F., MATTOSO, M. L. Q. Alocação de dados em Bancos de Dados Distribuídos In: Simpósio Brasileiro de Banco de Dados, 2003, Manaus. XVIII Simpósio Brasileiro de Banco de Dados. SBC, 2003. p.215 – 228

WILDEMBERG, M., PAULA, M., BAIÃO, F., MATTOSO, M. L. Q. Data Allocation in Distributed Databases, submitted to SBBD´03 Post Proceedings, Lecture Notes in Computer Science, 2004a.

WILDEMBERG, M., PAULA, M., BAIAO, F., MATTOSO, M., XAloc – Uma ferramenta para avaliar algoritmos de alocação de dados. In: Simpósio Brasileiro de Banco de Dados, SBBD-Demos, 2004b.