Speeding-Up Association Rule Mining With

Inverted Index Compression

**ABSTRACT**

The growing interest in data storage has made the data size to be exponentially increased, hampering the process of knowledge discovery from these large volumes of high-dimensional and heterogeneous data. In recent years, many efficient algorithms for mining data associations have been proposed, facing up time and main memory requirements. Nevertheless, this mining process could still become hard when the number of items and records is extremely high. In this paper, the goal is not to propose new efficient algorithms but a new data structure that could be used by a variety of existing algorithms without modifying its original schema. Thus, our aim is to speed up the association rule mining process regardless the algorithm used to this end, enabling the performance of efficient implementations to be enhanced. The structure simplifies, reorganizes, and speeds up the data access by sorting data by means of a shuffling strategy based on the hamming distance, which achieve similar values to be closer, and considering both an inverted index mapping and a run length encoding compression. In the experimental study, we explore the bounds of the algorithms’ performance by using a wide number of data sets that comprise either thousands or millions of both items and records. The results demonstrate the utility of the proposed data structure in enhancing the algorithms runtime orders of magnitude, and substantially reducing both the auxiliary and the main memory requirements.

**INTRODUCTION:**

Existing efficient algorithms in the ARM field follow some frequent item set generation strategies based on reducing either the number of candidate sets, the number of transactions, or the number of comparisons. In this regard, Apriority was specifically designed to prune candidate sets by using the anti­­-monotone property, which establishes that if a length-*k* item set is not frequent, none of its length-(*k*+1) supersets can be frequent. Han *et al ,* proposed a novel approach, the frequent pattern (FP)-growth algorithm, which was designed to reduce problem of high number of transactions and comparisons. FP-growth stores the frequent item sets into a tree structure, so data need to be scanned just once. Nevertheless, it still suffers with huge number of candidate item sets since either the larger number of I/O or the high memory requirements to store all sets. In recent years, many researchers have proposed very efficient algorithms for mining frequent item sets. Borgelt proposed an improved version of Eclat, which extends an item set until it reaches the boundary between frequent and infrequent item sets. In this proposal, transactions are represented by a bit matrix, in which each row corresponds to an item and each column to a transaction. To generate itemsets, Eclat follows a depth first traversal instead of a breadth first order as Apriori does. In a similar way, and considering a frequent tree structure like FP-growth, Han *et al.* proposed a more compact data structure that extends the FP-tree structure for storing compressed and crucial information about frequent items.In this approach, known as CFP-growth, an efficient tree-based mining method is considered to mine the complete set of frequent item sets.

**LITERATURE SURVEY:**

Title :Orthogonal Multiobjective Chemical Reaction Optimization

Approach for the Brushless DC Motor Design.

Author : Haibin Duan and Lu Gan.

Year : 2015

**Description:**

The optimal design of a brushless direct-current motor (BLDCM) is a prevalent and practical issue in the field of magnetics. A major problem is to design a BLDCM so that it operates optimally in the sense of producing maximum efficiency with minimal material cost. Using the sizing model, a novel orthogonal multiobjective chemical reaction optimization (OMOCRO) algorithm is proposed to solve this problem. Chemical reaction optimization is a newly proposed heuristic algorithm, inspired by the interactions between molecules during chemical reactions. In our proposed OMOCRO, we employ a Pareto ranking scheme to deal with multiobjective optimization problems, and orthogonal experimental design is used in the initialization stage. Comparative experiments with nondominated sorting genetic algorithm and multiobjective particle swarm approach demonstrate that our proposed method is more competitivein handling complex optimization problems.

**LITERATURE SURVEY:**

Title : Semi-supervised and unsupervised extreme learning machines.

Author : Gao Huang, Shiji Song, Jatinder N. D. Gupta, and Cheng Wu.

Year : 2011

**Description:**

Extreme learning machines (ELMs) have proven to be an efficient and effective learning paradigm for pattern classification and regression. However, ELMs are primarily applied to supervised learning problems. Only a few existing research studies have used ELMs to explore unlabeled data. In this paper, we extend ELMs for both semi-supervised and unsupervised tasks based on the manifold regularization, thus greatly expanding the applicability of ELMs.

The key advantages of the proposed algorithms are:

1. both the semi-supervised ELM(SS-ELM) and the unsupervised ELM(US-ELM) exhibit the learning capability and computational efficiency of ELMs.
2. both algorithms naturally handle multi-class classification ormulti-cluster

clustering. and

3) both algorithms are *inductive* and can handle unseen data at test time directly.

Moreover, it is shown in this paper that all the supervised, semi-supervised and unsupervised ELMs can actually be put into a unified framework. This provides new perspectives for understanding the mechanism of *random feature mapping*, which is the key concept in ELM theory. Empirical study on a wide range of data sets demonstrates that the proposed algorithms are competitive with state-of-the-art semi-supervised or unsupervised learning algorithms in terms of accuracy and efficiency.

**LITERATURE SURVEY:**

Title : An efficient accelerator for attribute reduction from incomplete data in

rough set framework.

Author : Yuhua Qian, JiyeLiang ,WitoldPedrycz, ChuangyinDang.

Year : 2014

**Description:**

Feature selection (attribute reduction) from large-scale incomplete data is a challenging problem in areas such as pattern recognition, machine learning and data mining. In rough set theory, feature selection from incomplete data aims to retain the discriminatory power of original features. To address this issue, many feature selection algorithms have been proposed, however, these algorithms are often computationally time-consuming. To overcome this shortcoming, we introduce in this paper a theoretic framework based on rough set theory, which is called positive approximation and can be used to accelerate a heuristic process for feature selection from incomplete data. As an application of the proposed accelerator, a general feature selection algorithm is designed. By integrating the accelerator into a heuristic algorithm, we obtain several modified representative heuristic feature selection algorithms in rough set theory. Experiments show that these modified algorithms outperform their original counterparts. It is worth noting that the performance of the modified algorithms becomes more visible when dealing with larger data sets.

**LITERATURE SURVEY:**

Title **:** Generating Databases for Query Workloads.

Author : Eric Lo, Nick Cheng, WingKai Hon.

Year : 2012

**Description:**

To evaluate the performance of database applications and DBMSs,we usually execute workloads of queries on generated databases of different sizes and measure the response time. This paper introduces MyBenchmark, an offline data generation tool that takes a set of queries as input and generates database instances for which the users can control the characteristics of the resulting workload. Applications of MyBenchmark include database testing, database application testing, and application-driven benchmarking. We present the architecture and the implementation algorithms of MyBenchmark.We also present the evaluation results of MyBenchmark using TPC workloads.

**LITERATURE SURVEY:**

Title **:** Large-Scale Video Hashing via Structure Learning.

Author : Guangnan Yey, Dong Liuy, Jun Wangz, Shih-Fu Chang.

Year : 2009

**Description:**

Recently, learning based hashing methods have become popular for indexing large-scale media data. Hashing methods map high-dimensional features to compact binary codes that are efficient to match and robust in preserving original similarity. However, most of the existing hashing methods treat videos as a simple aggregation of independent frames and index each video through combining the indexes of frames. The structure information of videos, e.g., discriminative local visual commonality and temporal consistency, is often neglected in the design of hash functions. In this paper, we propose a supervised method that explores the structure learning techniques to design efficient hash functions. The proposed video hashing method formulates a minimization problem over a structure-regularized empirical loss. In particular, the structure regularization exploits the common local visual patterns occurring in video frames that are associated with the same semantic class, and simultaneously preserves the temporal consistency over successive frames from the same video. We show that the minimization objective can be efficiently solved by an Accelerated Proximal Gradient (APG) method. Extensive experiments on two large video benchmark datasets (up to around 150K video clips with over 12 million frames) show that the proposed method significantly outperforms the state-of the-art hashing methods.

**LITERATURE SURVEY:**

Title **:** Big Data Mining using Map Reduce: A Survey Paper.

Author : Shital Suryawanshi1, Prof. V.S.Wadne.

Year : 2014

**Description:**

Big data is large volume, heterogeneous, distributed data. Big data applications where data collection has grown continuously, it is expensive to manage, capture or extract and process data using existing software tools. For example Weather Forecasting, Electricity Demand Supply, social media and so on. With increasing size of data in data warehouse it is expensive to perform data analysis. Data cube commonly abstracting and summarizing databases. It is way of structuring data in different n dimensions for analysis over some measure of interest. For data processing Big data processing framework relay on cluster computers and parallel execution framework provided by Map-Reduce. Extending cube computation techniques to this paradigm. MR-Cube is framework (based on map reduce) used for cube materialization and mining over massive datasets using holistic measure. MR-Cube efficiently computes cube with holistic measures over billion-tuple datasets.

**LITERATURE SURVEY:**

Title : Fast Algorithms for Mining Association Rules.

Author : Rakesh Agrawal, Ramakrishnan Srikant.

Year : 2015

**Description:**

We consider the problem of discovering association rules between items in a large database of sales transactions.We present two new algorithms for solving this problem That are fundamentally different from the known algorithms. Empirical evaluation shows that these algorithms outperform the known algorithms by factors ranging from three for small problems to more than an order of magnitude for large problems. We also show how the best features of the two proposed algorithms can be combined into a hybrid algorithm, called AprioriHybrid.Scale-up experiments show that AprioriHybrid scales linearly with the number of transactions. AprioriHybrid also has excellent scale-up properties with respect to the transaction size and the number of items in the database.

**MODULES:**

1**.**USER INTERFACE DESIGN:

2.FILE UPLOAD:

3.COMPRISE THE FILE:

4.STORE FILE TO THE DATABASE:

5.DECOMPRISE THE FILE:

**MODULE DESCRIPTION:**

**1.User Interface Design:**

To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server.Server will create the account for the entire user to maintain upload and download rate.Name will be set as user id.Logging in is usually used to enter a specific page.

**2.File Upload:**

In this module the file will being uploaded where the file contains large amount of items and records.

**3**.**Compress The File:**

In this module comprising attributes of any domain, either categorical or

continuous, and considering any number of both items and transactions.

4. **Store File to The Database**:

In this module the methodology of considered to store data records into the

Database.

5.**Decompress The File:**

In this module the user decompress the file as their requirements, from proposed the file will be as simplifies andreorganizes data items in order to reduce data size and providea faster access to the stored information.

**MODULES DIAGRAM:**

1.User Interface Diagram:

Welcome to Login

Server

Login

Register

New user

2.File Upload:

Database

File 1

Upload

User

File 3

File 2

3.Compress The File:

Database

*com*press

File1

Compress the file

File3

File2

User

4.Store File To Database:

Database

Stores files

File1

Compressed file

File2

User

File3

5.Decompress The File:

Database

Compressed file

File 1

Decompress

File2

User

File3

**SYSTEM ALGORITHM:**

**Algorithm 1** Inverted Index Mapping Procedure

**Input:** *sorted*\_*data*

**Output:** *dataindex*\_*based* **procedure** Inverted index mapping

1: *dataindex*\_*based* ← ∅

2: *indexvalue* ← ∅

3: *indexattribute* ← ∅

4: *list*\_*attributes* ← *getAttributes(sorted*\_*data)*

5: **for all** *attribute* in *list*\_*attributes* **do**

6: **for all** *value* in *attribute* **do**

7: **for all** *transaction* in *sorted*\_*data* **do**

8: **if** *value* satisfies *transaction* **then**

9: *indexvalue* ← *indexvalue* ∪ *transaction*

10: **end if**

11: **end for**

12: *indexattribute* ← *indexattribute* ∪ *indexvalue*

13: *indexvalue* ← ∅

14: **end for**

15: *dataindex*\_*based* ← *dataindex*\_*based* ∪ *indexattribute*

16: *indexattribute* ← ∅

17: **end for**

18: **return** *dataindex*\_*based*

**end procedure**

**SYSTEM REQUIREMENTS:**

**HARDWARE:**

PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.

RAM : 512 MB DD RAM

MONITOR : 15” COLOR

HARD DISK : 40 GB

**SOFTWARE:**

Front End : JAVA (j2ee, Servlets, jsp)

Back End : My SQL

Operating System : Windows 07

IDE : Eclipse

Use Case Diagram:

Login request

user

file3

file upload

file2

file1

compress

register decompress

**EXPLANATION:**

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. In our use case diagram first Data mining user login into Data mining. If a data mining want to access another data through Agent he need to get access from data mining admin after getting the access the data can be able to view product Support.

**Class Diagram:**

Database



Store all data

User



UserName



'Passsword



Login()



upload()

File upload



All files are upload

Compress



Compress all files

Decompress

**EXPLANATION:**

The class diagram is the main building block of [object oriented](http://en.wikipedia.org/wiki/Object_oriented) modeling. It is used both for general [conceptual modeling](http://en.wikipedia.org/wiki/Conceptual_model) of the systematic of the application, and for detailed modeling translating the models into [programming code](http://en.wikipedia.org/wiki/Programming_code). In data user we took the user login and user constraints for uploading file and acess the stored information from database.

**Object Diagram:** files

Database

**user**

Decompress the file

Compress the file

**File uploaded**

**EXPLANATION:**

Object diagram we are telling about the flow of objects how the process is running. In the above digram tells about the flow of objects between the classes.The main object of this diagram is data mining user login his window and send the his constraints to compress file and store information to the database.

**State Chart Diagram:**

Decompress

Login

File uploaded

Compress the file

Database

**EXPLANATION:**

State diagrams require that the system described is composed of a finite number of states. sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics. In our state diagram first data mining user login into data and how items and records are accessing from Database.

**Activity Diagram:**

Login

File Upload

Compress the file

Database

Decompress

**EXPLANATION:**

In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. In our activity diagram first data mining user login into the database. the large number of data size are reduce in memory storage.

**Sequence Diagram:**

Login

File & Compress

Database store

database

Decompress

User Name, Password

File Upload

File original size

File decompress

Compress the file

**EXPLANATION:**

In our sequence diagram specifying processes operate with one another and in order. In our sequence diagram user login into data. This new representation simplifies and reorganizes data items in order to reduce data size and provide a faster access to the store information.**Collaboration Diagram:**

Database store

Login

File upload

reduce &

decompress

1: User Name, Password

2:Data Service

3: Getting Key Permission

4:reduce data size

5:file compressed

**EXPLANATION:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system. now comprising attributes of any domain, either categorical or continuous, and considering any number of both items and transactions. This new representation simplifies and reorganizes data items in order to reduce data size and provide a faster access to the stored information

Data Flow Diagram:

Level 1:

File uploaded

User login

Compress and reduce data size

Level 2:

File upload

Login

Decompress

Files

Compress the file

Size reduced and faster access

Database

**EXPLANATION:**

The data size reduction with regard to the data structure used with a series of ARM algorithms, the computational time required to load the data, and the main memory requirements. the reduction ratio in time when the new data structure is used by ARM algorithms that find all association rules that satisfy certain constraints.

E-R Diagram:

User Login

File to upload

File 1

File 2

File3

**EXPLANATION:**

Entity-Relationship Model (ERM) is an abstract and conceptual representation of data. Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database.

**Component Diagram:**

Database storage

user

Decompress the file

Compress the file

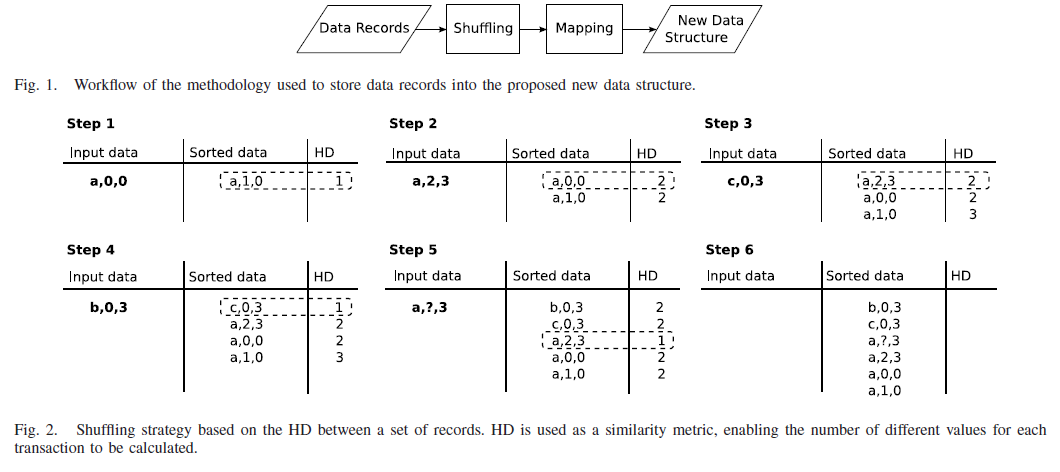
and reduce size

File upload

**EXPLANATION:**

In the Unified Modeling Language, a component diagram despites how components are wired together to form larger components and they are used to illustrate the structure of arbitrarily complex systems. comprising attributes of any domain, either categorical or continuous, and considering any number of both items and transactions. This new representation simplifies and reorganizes data items in order to reduce data size and provide a faster access to the stored information.

**System Architecture:**



**FUTURE ENHANCEMENT:**

In this a low number of instances implies a low probability of having repeated attribute values, so the RLE compression could seem to be meaningless. Nevertheless, our proposal also performs well when small data sets (in terms of the number of instances) are considered.

**ADVANTAGE:**

* Reducing the data size and the access time in high levels.

**CONCLUSION:**

In this paper, we proposed a novel data structure specifically designed to be used by ARM algorithms. This data structure enables ARM algorithms to deal with much larger volumes of data reducing both the main memory requirements and the data access time. The motivation and idea behind our approach is to improve the performance of existing algorithms without any modification in their original schemata. Thus, the algorithms remain unchanged, so it is a great advantage since the performance capabilities of many existing algorithms could be improved. The proposed structure is expected to provide a high utility when that the domain sizes of attributes in a data set are not so large. Hence, The experimental results reveal that the proposed data structure is extremely promising, reducing the data size, the main memory requirements, and the access time in high levels. The new structure has been also applied to a varied set of both ARM algorithms and data sets. The results obtained have shown that the use of the proposed data structure is highly recommended where high efficient data accesses are required, especially when dealing with large data sets.

**REFERENCES:**

[1] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules in

large databases,” in *Proc. 20th Int. Conf. Very Large Data Bases (VLDB)*,

Santiago, Chile, 1994, pp. 487–499.

[2] J. Han, J. Pei, Y. Yin, and R. Mao, “Mining frequent patterns without

candidate generation: A frequent-pattern tree approach,” *Data Min.Knowl.*

*Disc.*, vol. 8,no.1, pp. 53–87, 2004.

[3] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,”

*J.Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[4] S. García and F. Herrera, “An extension on statistical comparisons of

Classifiers over multiple data sets for all pairwise comparisons,”

*J.Mach.Learn. Res,*vol. 9, pp. 2677–2694, Dec. 2008.

[5] M. Hall *et al.*, “The WEKA data mining software: An update,” *SIGKDD*

*Explor.*, vol. 11, no. 1, pp. 10–18, 2009.

[6] J. Vreeken, M. Leeuwen, and A. Siebes, “KRIMP: Mining item sets that

compress,” *Data Min. Knowl. Disc.*, vol. 23, no. 1, pp. 169–214,2011.

[7] V. Marx, “Biology: The big challenges of big data,” *Nature*, vol. 498 no.

7453, pp. 255–260, 2013.

[8] H. Gao, S. Shiji, J. N. D. Gupta, and W. Cheng, “Semi-supervised and

unsupervised extreme learning machines,” *IEEE Trans. Cybern.*,vol. 44,

no.12, pp. 2405–2417, Dec. 2014.

[9] S. Zhang, Z. Du, and J. T. L. Wang, “New techniques for mining frequent

patterns in unordered trees,” *IEEE Trans. Cybern.*, vol. 45, no. 6, pp. 1113–

1125, Jun. 2015.

[10] X. Liu, Y. Mu, D. Zhang, B. Lang, and X. Li, “Large-scale unsupervised

hashing with shared structure learning,” *IEEE Trans. Cybern.*,vol. 45, no. 9,

pp. 1811–1822, Sep. 2015. [Online].

Availablehttp://dx*.*doi*.*org/10*.*1109/TCYB*.*2014*.*2360856