Intension Mining – Implementation and Integration of a User-Centric Mining Scheme.

Gupta S.K[1], Wadhera G. [2],Kaul P.[2], and Bhatnagar V[3].
[1] Department Of Computer Science and Engineering, Indian Institute of Technology, Delhi, India
skg@cse.iitd.ernet.in
http://www.cse.iitd.ernet.in/~skg
[2] Department of Computer Engineering, Netaji Subhas Institute of Technology (University of Delhi), India
{prenayan, gauhar_wadhera}@hotmail.com
[3] Department of Computer Science, University of Delhi, India

Abstract. The need for a user-centric mining approach for intelligent data analysis and domain understanding is analyzed. The paper takes a holistic view of the Knowledge Discovery process based on which a modular and generic KDD framework is built. The paper follows a central idea, which culminates in its implementation issues. The framework bases itself on the fundamentals of the three-level database architecture and incremental mining techniques. Key architectural features include scalability, concept of mining schemas and run-time flexibility w.r.t. mining parameters, data selection over the time domain. Component functionalities and workflows are discussed while integration of the architecture is the principle issue dealt with. The ideas put forth are backed by suitable implementation of components thus strengthening the proposal of a user-centric and well-monitored KDD solution.

1 Intension Mining: The intention behind the idea.

Data Mining has evolved as a discipline with great implications. Its development has been a revolution relative to the evolution of classical databases. However, a few deficiencies have been hindrances in its path, once towards the throne of the decision maker. In fact misdirected user expectations, as the above are chiefly responsible.

However, we still feel that data mining is far away from the doors of oblivion. A change in viewpoint is needed presently, as we must regard mining methods as vital data analyses tools rather than managerial replacements. The technology builds data domain semantics and does not encompass foresight and external knowledge – key ingredients to a decision.

Intension Mining attempts to establish itself as an "application-driven" technology in the industry. Certain features of the scheme include:

- Easy to use interface with components as a Mining Query Language.
- Greater control over mining process for end-user in terms of mining parameters.
- Flexibility in data selection over the time domain and types of patterns.
- Use of incremental methods to reduce overheads from repetition of mining queries.

The paradigm is attacks the two principle deficiencies in the current mining scenario – rigidity and user-control.

Intension mining addresses the rigidity issues on multiple grounds. Primarily, it offers flexibility over the types of patterns available. The concepts of Knowledge Concentrates and 'The 3 Operators' discussed in Section 2 aim to resolve the problem of merging of patterns with decent success. Overheads in case of repetitive mining queries are avoided through a layered architecture and incremental mining techniques.

It provides greater user-participation in the mining process on two chief accounts. Firstly, most algorithm schemes in Intension Mining attempt to segregate stages requiring minimal parameters as offline and permit the user to vary parameters in the online stages. Secondly, it applies data mining as an "Online Analysis" tool by maximizing background processing and providing results in affordable time.

2 Intension Mining: The Approach and The Architecture

2.1 Knowledge Discovery Process Model: An Overview.

Intension Mining is primarily taken from [VB01] and is based on the idea of planning the data mining requirements and goals of an organization in the form of Knowledge Discovery Schema (KDS). Mining requirements include data source specification (database, table, attributes), mining algorithm specification including algorithm-dependent data and periodicity of aggregation. Besides mining requirements, the schema also includes data relating to user-authorization and visualization tools. The KDS design facilitates a user-centric view, thereby enhancing productivity, ease of use and comprehension. It is fundamentally based on incremental mining concepts.

The entire approach attempts to introduce a layered approach to mining, specifically towards the 3-Level Database architecture. Consequentially the functioning of each mining method as described in [VB01] evolves into a sequence of three operators:

- **Accumulation Operator**

The incremental database is "processed" automatically at regular intervals, with the periodicity specified in the schema (KDS). The processing consists of multiple tasks, including pre-mining of data followed by preliminary analysis and/or aggregation. Since the mining requirements are available in the schema, the system is capable of carrying out, pre-mining-cum-aggregation operation in offline mode. We term this periodic operation on the incremental database as Accumulation. The result of accumulation is an intermediate knowledge form termed "Knowledge Concentrate" (KC). These are data structures dependent on their parent algorithm. Therefore, the accumulation process corresponding to each KDS is active according to the specified periodicity and creates structures to be used for actual mining at run-time. Each KC provides a non-overlapping window to the database specified in the KDS.

- **Merging Operator**

The KCs can be combined meaningfully to define the subset of database to be mined. Extracting Knowledge Concentrates during Accumulation typically requires multiple scans of database. However, merging is done on Knowledge Concentrates and does not essentially require a database scan. However, with new algorithms and methodologies developing to suit the Intension Mining paradigm a provision has been provided to utilize database scans via the merging operator as well. The operation of Merging uses a set of KCs as input to create a single KC possessing equivalent data and knowledge. This operator is mainly responsible for the layered approach since it segregates online mining and background pre-processing.

- **Mining Operator**

Each mining query bases itself on a particular KDS and involves a time domain specification, which results in selection of corresponding KCs for the durations specified. The KCs are merged using the Merge Operator and the mining operator for the algorithm used in the KDS is applied to the resultant KC to extract patterns.

With mining proposed as independent to accumulation, multiple mining requests can be satisfied simultaneously without affecting the database performance. An important characteristic of Intension Mining is that it perceives KDD as a continuous process. Segregation of I/O intensive and computation intensive operations of the Knowledge Discovery Process makes Intension Mining scheme naturally scalable with respect to data volume. It is also generic in the sense that it is neutral to the application domain. Algorithms made compatible to I-MIN are described in [PR02][SK00][PD02].

2.2 Intension Mining: Fundamental Architecture and Components

The entire architecture – the layers, the components as well as the essential workflows draw their inspiration from the Three-Level Database architecture to achieve numerous objectives including pattern flexibility and merging capabilities.

External modules fulfill user as well as Administrator level interactions. Fundamental porting for Update Managers as well as User Agents are provided and the actual displays left open for Application Development. The 3 layers designed are:

- **Front End Layer**

This represents interactions between the administrator/end-user and the mining system. It also interacts with various external modules as system loggers and Meta-Data Dictionaries. It serves for various purposes including

- Schema Compilation & Query Processing
- Component/Library Validation and Creation of Meta-Data.
- System Logging
- Creation of Execution Plans/Scripts

- **Data Mining Layer**

This layer controls the scheduling and management of mining activities at all operator levels. The various responsibilities it addresses include

- Accumulation/Querying Process Management and Scheduling
- Module Access as well as Data Access.
- Returning program status to upper layers for logging purposes

- **Storage Layer**

The Storage Layer principally controls the directory and file structure management. It is responsible for systematic storage and easy retrieval of Knowledge Concentrates as well as query results.

Key Components

1. *Update Manager (UM):* It functions as the client side of the Library/Component Manager Client/Server update scheme for updating the look-up table scheme in Library/Component managers.
2. *Front-End Engine (FEE):* It is the logical entity that serves as the inter-layer data flow channel. The responsibility of interacting with the Log Manager for events as accumulation processes lies with the FEE.
3. *Log Manager (LoM):* Interface for secondary storage based log database and the Intension Mining system in order to store log data in relevant tables. It is primarily the Server side of the Client/Server Logging Scheme with the FEE.
4. *Library/Component Manager (LM/CM):* They are implemented as look-up table schemes containing entries for available pre-processing/mining modules and their corresponding system paths. They act as the servers for:
   - Update scheme with Update Manager
   - Lookup/Validation scheme with Schema Compiler
5. *Schema Compiler (SC):* It functions as the initiator of the Accumulation process by creating a Knowledge Discovery Schema (KDS) and performing the various tasks as validation and others described in Section 3.2.
6. *Query Processor (QP):* It is initiator of the Mining process by parsing a query request to create an Execution Plan for further execution. Details of the entire query processing are presented in Section 3.3.
7. *Script Generator (SG):* The SG creates actual operating system level execution scripts for Accumulation and Mining processes. It also consists of a primitive parser to interpret Execution Plans sent by the SC and QP.
8. *Data Processing Engine (DPE):* Security issues regarding data access rights and direct database/warehouse access are resolved via this component. The DPE acts as an interface between the data source and any component or process seeking data. Details of its operation are described in Section 3.1.
9. *Data Mining Engine (DME):* The responsibilities of process management and scheduling as well as status and Log-related interaction with the upper layers are handled by the DME.
10. *Storage Engine (SE):* The SE is accountable for all interactions of the Intension-Mining Directory Structure Scheme (DSS) with various components as mining modules, query results access and accumulation modules.

The entire architecture and its key components have been designed in order to lend a high level of modularity to the entire scheme. The objective is to further this idea as a technology, amenable to being, updated both at system and component level.

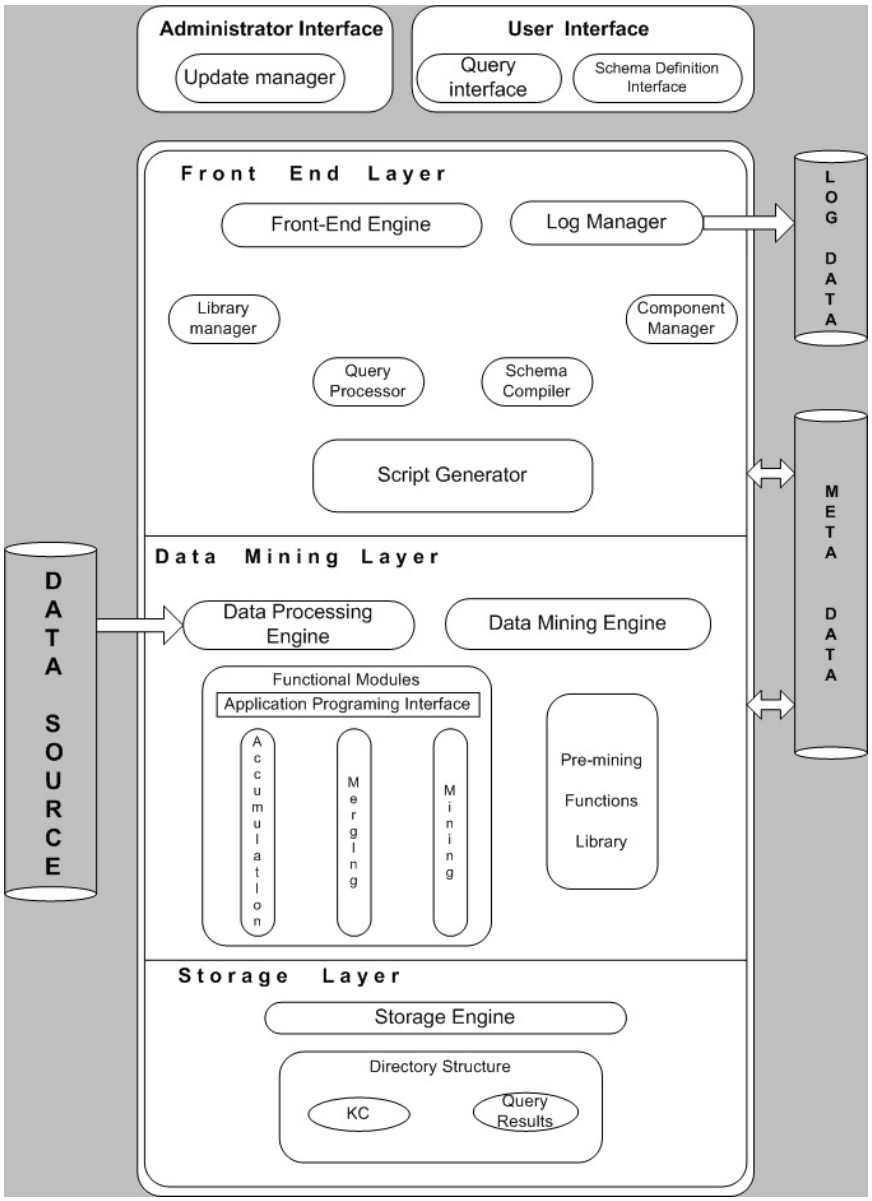

Fig. 1. Intension Mining Architecture (as evolved from [VB01]).

3 Intension Mining: Workflow and Implementation Issues

The architecture can be best understood through an explanation of the underlying workflows. The system functioning can be characterized in two chief process models:

- Schema Compilation Model & Query Processing Model

3.1 Central Implementation Issues

Client-Server Architecture

Streamlined component interaction is emphasized upon through a Client-Server approach for an apt solution via secure and effective communication. The functionality of each inter-module interaction is well encapsulated within each client-server interaction. The introduction of any new interaction involves configuration of client-server relationships. Examples in <Client/Server> syntax:[1]

- UM/CM
- SC/CM
- SC/LoM
- SC/SG

Data Processing Engine

Data security has been given due consideration in the architecture. All data access operations are performed by the DPE, which is responsible for secure data access by preventing direct database/warehouse access to any system module. A generic approach to data provision has been implemented in the form of flat files. DBMS platform independence is provided by creation of various ODBC interface modules.

Directory Structure

A clean and well-segregated directory structure is provided for systematic data organization for Schema definition, KCs, Query results as:

IMIN (Dir.)
……..SCHEMA_NAME (Dir.)
_____……..KC (Dir.)
_____……..QUERY_DATA (Dir.)
_____……..OUTPUT_DATA (Dir.)
_____……..SCHEMA_FILE (File)
_____……..ALGORITHM_DEPENDENT_DATA (File)
_____……..HISTORY_FILE (File)

I-MIN Protocol (IMP)

IMP has been developed to resolve system integration conflicts. The protocol encompasses a messaging paradigm. Salient protocol features include:

- Standard nomenclature for KCs and query results, incorporating time-stamps
- Standard syntax for command line arguments
- Use of a standard message structure involving headers, text delimiters etc.

[1] Refer to Section 2.2.2 for list of acronyms.

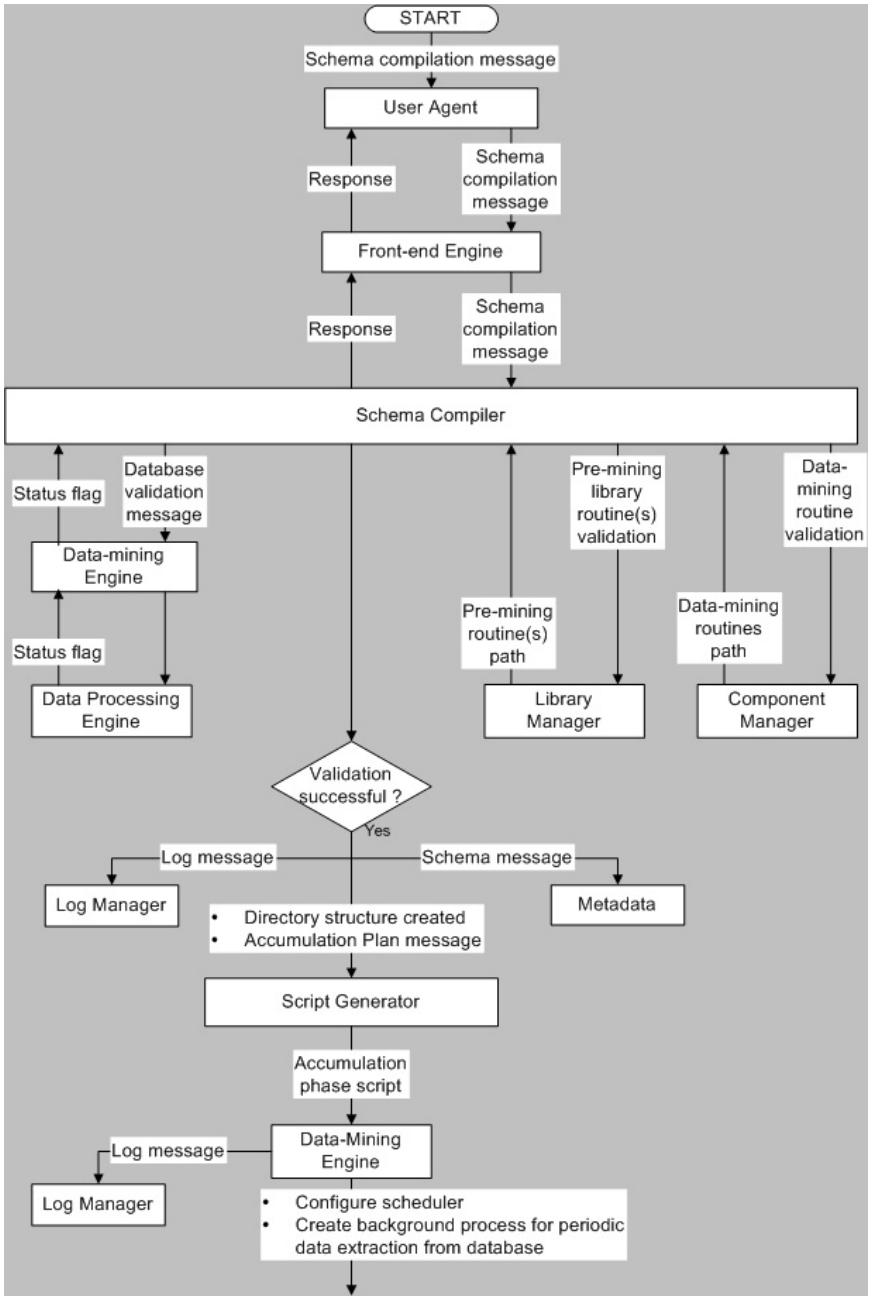

Fig. 3. Schema Compilation Model.

3.2 Schema Compilation Model: Workflow and Implementation Issues

The Schema Compilation Model defines the application domain per user needs and is also responsible for the Accumulation Operator. Only administrator-level users have permission to participate since domain knowledge is a prerequisite in this stage.

The User-agent performs user verification at the interface level and sends the schema definition as an input message via the Front-End Engine (FEE) in the format discussed in [VB01]. This serves as the primitive form of mining intension. The Schema Compiler (SC) commences compilation on receipt of the message from the FEE. The SC parses the message to extract the various components. Content validation of the message forms the first action of the SC and is split into three phases - *Database validation, Pre-mining validation and Mining validation.*

During 'database validation', SC validates the data-source information provided by using the Data Processing Engine (DPE), which scans the 'data source' for validation of specified items and returns status flags to indicate success or type of error encountered. During the 'pre-mining validation', SC ascertains availability of required pre-mining routines for data cleaning, transformation and other steps specified in the KDS. It queries the Library Manager server and if the specified routine is present, its complete path is returned to the SC otherwise appropriate error code is generated. 'Mining validation' is similar to the previous case, except that here the Component Manager server is queried. The SC collects the validation responses and sends a message to the user. In the event of successful validation a log message is sent to the Log-manager. The meta-data for the system gets updated and compiled schema file is created. The Model also creates an appropriate directory structure. An Accumulation-Plan is sent to the Script Generator for generating an executable script.

Inside SG, the Accumulation-Plan message is received, parsed and appropriate command-line arguments appended to the various pre-mining routines to generate executable statements, which are compiled into a shell executable file representing the accumulation process, and passed onto the Data Mining Engine, the process manager.

The DME parses the message to configure the 'Anacron' scheduler for creation of a background Accumulation process to periodically create KCs for the KDS. It also communicates status details to the Log Manager.

3.3 Query Processing Model: Workflow and Implementation Issues

The Query Processing Model is primarily concerned with run-time queries involving pattern extraction from the knowledge created via accumulation performed on the data source. Every query fired by a user maps onto a pre-defined KDS.

The user submits the query in the format specified in [VB01] via the User-Agent. After systematic user authentication, the query message is passed down to the Query Processor (QP) through the Front End Engine. QP parses the query and validates its parameters against those specified in the schema definition file. The validation of a query does not involve component validation from look-up servers. On successful validation the system log is updated through the Log Manager. The validation phase results are also displayed to the user. Further, a Mining-Plan is created, which involves determining the KCs, the merging and mining functions. The KCs required

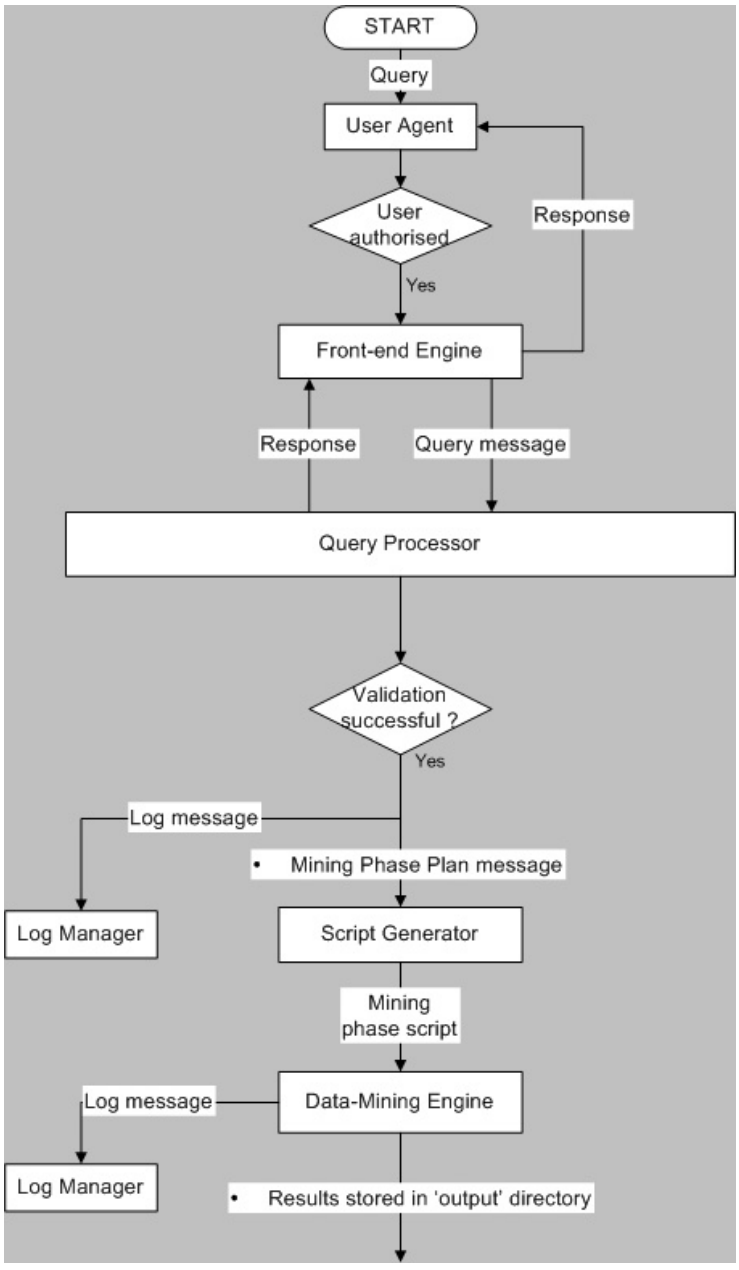

Fig. 4. Query Processin g Model.

during actual mining are handpicked by QP and listed in a file. Finally a Mining-Plan message is passed to the SG to create an executable mining script. The handpicked KCs are merged using the Merging Operator and mined using the Mining Operator. The Mining Phase script creation is similar to script creation from an Accumulation-Plan by the SG. Mining results are stored in the designated 'output' directory.

4 Intension Mining: The Effort

Challenges Faced

Various issues have been addressed during the course of implementation to ensure adherence to the core ideas. The mining algorithms [PR02][SK00][PD02] incorporated were suitably modified on two accounts. Firstly, certain algorithms had to be re-engineered to make them incremental, which is central to the idea. Secondly, the inherent workflow was suitably altered for compatibility with the evolved protocol. Scalability has been worked upon as a client-server model developed along with an underlying protocol has ensured future-readiness of the package.

The Road Ahead

This work seeks to reform the mining process and considerable work is being done presently to substantiate it further. The next step towards consolidating the idea as relevant and feasible is under progress by way of rigorous component analysis along with system testing in order to extract efficiency statistics.

Further expansion has been left open-ended in various directions. A potential area of expansion is fraud detection via system logs and Meta-data. While Meta-data provides a broad domain overview, system logs provide active usage information and can be effectively mined to reveal possible frauds by detecting pattern deviations.

A costing model can easily be incorporated to keep an account of system usage in a commercial environment. The system logs again are of use in this case as time and resource utilization information can be easily extracted from them.

The ideas put forth are well in tune with the current emphasis on realizing data mining as a real world intelligent data analysis tool. The work being done in the field of structured data mining and upcoming database ideas like Hippocratic databases [RA02] share various levels of commonality with the I-MIN model in their core ideas.

5 References

[VB01] Gupta S.K., Bhatnagar V., Wasan S.K. and Somayajulu DVLN: Intension Mining: A New approach to Knowledge Discovery in Databases (2001)

[RA02] Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Hippocratic Databases (2002)

[DH02] Dhamija A.K.: Schema Compilation and Query Evaluation in Intension Mining Framework (2002).

[PR02] Pratibha V.: The CACTUS Algorithm in an Intension Mining Framework (2002).

[SK00] Gupta S.K, Bhatnagar V., Wasan S.K: User Centric Mining of Association Rules. (2000)

[PD02] Padmaja U.K.: Iceberg Queries in Intension Mining Approach (2002).