Algorithms for Data Mining and Bio-informatics

Kartick Chandra Mondal

UNIVERSITE DE NICE-SOPHIA ANTIPOLIS

ECOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

T H E S E

pour l'obtention du grade de

Docteur en Sciences

de l'Université de Nice-Sophia Antipolis

Mention : Informatique

présentée et soutenue par

Kartick Chandra MONDAL

Algorithms for Data Mining and Bio-informatics

Thèse dirigée par Nicolas PASQUIER

soutenue le 12 Juillet 2013

Jury :

Mme	Martine COLLARD	Professeur des Universités	Rapporteur
M.	Pascal PONCELET	Professeur des Universités	Rapporteur
M.	Nicolas LACHICHE	Maître de Conférences	Examinateur
M.	Andrea TETTAMANZI	Professeur des Universités	Examinateur
M.	Frederic PRECIOSO	Professeur des Universités	Examinateur
M.	Nicolas PASQUIER	Maître de Conférences	Directeur de thèse

Dedicated to

Family & Supervisor.

Université de Nice Sophia Antipolis
Département informatique

Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis
Groupe de Recherche KEIA – Pôle GLC

Letter from Supervisor

This, to certify that the thesis for the fulfillment of the Ph.D. subject entitled ***"Algorithms for Data Mining and Bio-informatics"*** is a recorded work carried out by **Mr. Kartick Chandra Mondal** under my supervision and guidance. The thesis in the present form is to obtain title of PhD in Computer Science, as specified by the Computer Science Department and as per regulation of the University of Nice Sophia Antipolis. This work report has attained the standard necessary for submission.
To the best of my knowledge, the results embodied in the report are original in nature & worthy of incorporation in the present version of the thesis for Ph. D. degree in Computer Science.

Supervisor

Dr. Nicolas PASQUIER
Senior Lecturer
Department of Computer Sciences
University of Nice Sophia Antipolis
Nice
France

External Examiner(s)

Acknowledgement

A research work is not an obligation done by one person toward a predetermined and specific goal. Rather, it is a realization conducted from many elements, some administered directly and others indirectly, to the planning and implementation of a long-term effort. Such a project requires hard work, perseverance and dedication, and above all a lot of supports, as only teamwork can make good better and better best. This work has been carried out as a project work of the KEIA Team of the I3S Laboratory of the University of Nice Sophia Antipolis. This work was conducted with the financial support of the European Union's mobility EMMA (Erasmus Mundus Mobility with Asia) program and the I3S Laboratory.

First, I would like to express my sincere gratitude to Dr. Nicolas Pasquier of Computer Science Department of University of Nice Sophia Antipolis for his careful supervision of my doctoral thesis and for all his support from the very beginning right after the thesis idea was conceived. He was always available with new ideas, advises and suggestions during the difficult phases of the thesis work. I would also thankful and express my indebtedness to all the reporters, examiners and jury members of the thesis for accepting to review this report and for their careful evaluation of the thesis presentation.

This auspicious occasion gives me a wonderful opportunity and great pleasure to express my thanks to Prof. Marc Diener and Prof. Francine Diener for giving me the opportunity to conduct this work at the University of Nice Sophia Antipolis as an EMMA scholar. Also, I render my respect to Mrs. Claudine Torres for giving her support and helps for solving all the administrative problems, troublesome for a foreign student, right from the beginning.

I am thankful and have pleasure to express my deep friendship to all the members of the KEIA Team of the I3S Laboratory, namely Prof. Frederic Precioso, Prof. Andrea G. B. Tettamanzi, Dr. Celia de Costa Pereira, Dr. Denis Pallez and Dr. Christel Dartigues, and to all other members and friends of the I3S Laboratory that supported my work and entertained me with love; that helped me a lot to progress rapidly in my work on daily basis.

I would like to express my thanks to Prof. Ujjwal Maulik from Jadavpur University, Computer Science and Engineering Department, and Prof. Sanghamitra Bandhyapadhyay, Indian Statistical Institute of Kolkata, for supporting me by giving different ideas which has leads to the betterment of the result at the end. I am very much grateful to both of them for their supports.

Last but not in any way the least, before going to the implementation detail, I would like to express all my sincere love and respect to Dr. Anirban Mukhopadhyay, Associate Professor of Computer Science and Engineering Department of Kalyani University, India, for his valuable advices, guidance and unending support right from the beginning. I have no word to express my gratitude for his constant motivation and inspiration.

Also a lot of thanks and love to my parents, my brother and sister, Anindita and all my family members who always believed and supported me in every possible way. I will always be thankful for their love and confidence, which constantly motivated me to work healthy and happily far away from my family. I sincerely appreciated all their love, support and confidence that made this work possible.

So, right at the end I would like to express my apologies to those whose names I have missed out to mention.

Kartick Chandra MONDAL

Abbreviations

AIDS: Acquired Immune Deficiency Syndrome

AR: Association Rule

API: Application Programming Interface

ARM: Association Rule Mining

CIM: Closed Itemset Mining

DB: Data Base

DM: Data Mining

FIM: Frequent Itemset Mining

FCI: Frequent Closed Itemset

FCP: Frequent Closed Pattern

FGIST: Frequent Generalized Itemset Suffix Tree

FI: Frequent Itemset

FIST: Frequent Itemset mining using Suffix-Trees

GO: Gene Ontology

HIV: Human Immunodeficiency Virus

KDD: Knowledge Discovery from Database

KEGG: Kyoto Encyclopedia of Genes and Genomes

NCBI: National Center for Biotechnology Information

NIAID: National Institute of Allergy and Infectious Diseases

NIH: National Institutes of Health

OID: Object Identifier

PPI: Protein-Protein Interaction

SFD: Sorted Frequent Database

SIFT: Scale-Invariant Feature Transform

Notations

Symbol used in the thesis	LaTeXname	Significance
Mathematical operators:		
$\neq$	neq	Inequality
$\geq$	geq	Greater than or equal to
$\leq$	leq	Less than or equal to
Set theory operators:		
$\in$	in	Membership
$\notin$	notin	Absence of membership
$\subset$	subset	Subset inclusion
$\supset$	supset	Superset inclusion
$\subseteq$	subseteq	Subset inclusion or equality
$\supseteq$	supseteq	Superset inclusion or equality
$\not\subseteq$	nsubseteq	Not a proper subset
$\not\supseteq$	nsupseteq	Not a proper superset
$\bigcup$	bigcup	Generalized union
$\bigcap$	bigcap	Generalized intersection
$\exists$	exists	Existence
$\nexists$	nexists	Non existence
$\setminus$	setminus	Set difference
$\forall$	forall	Universality
$\prod$	prod	Cartesian product
$\prec$	prec	Karp reduction
Set & logic symbols:		
$\emptyset$	emptyset	Null set
$\longrightarrow$	longrightarrow	Associative implication
$\Longrightarrow$	Longrightarrow	Logical implication
$\square$	Box	CQFD
α	alpha	Index denominator
ψ	psi	Galois connection function
γ	gamma	Galois connection function
Γ	Gamma	Galois closure operator
$\mathcal{L}$	mathcal{L}	Set denominator
$\mathfrak{R}$	mathfrak{R}	Set denominator

List of Publications

Journals

1. K. C. Mondal, A. Mukhopadhyay, U. Maulik, S. Bandhapadhyay and N. Pasquier. *MOSCFRA: A multi-objective genetic approach for simultaneous clustering and gene ranking.* Lecture Notes in Bioinformatics, LNCS 6685, p.174-187, Springer, 2011, DOI: 10.1007/978-3-642-21946-7_14.

International Conferences

2. K. C. Mondal, N. Pasquier, A. Mukhopadhyay, U. Maulik and S. Bandhapadhyay. *A new approach for association rule mining and bi-clustering using Formal Concept Analysis.* Proceedings of the 8th international conference on Machine Learning and Data Mining (MLDM'2012), Berlin, Germany, 2012.

3. K. C. Mondal, N. Pasquier, A. Mukhopadhyay, Celia de Costa Pereira, U. Maulik and Andrea G. B. Tettamanzi. *Prediction of protein interactions on HIV-1-Human PPI data using a novel closure-based integrated approach.* Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms (BIOINFORMATICS'2012), Algarve, Portugal, 2012.

4. K. C. Mondal, A. Mukhopadhyay, U. Maulik, S. Bandhapadhyay and N. Pasquier. *Simultaneous clustering and gene ranking: A multi-objective genetic approach.* Proceedings of the 7th international meeting on Computational Methods for Bio-informatics and Bio-statistics (CIBB'2010), Palermo, Italy, 2010.

Book Chapters

5. K. C. Mondal and N. Pasquier. *Galois Closure Based Association Rule Mining from Biological Data.* Mourad Elloumi and Albert Y. Zomaya Editors, Biological Knowledge Discovery Handbook: Preprocessing, Mining and Post-processing of Biological Data, Wiley Publisher, 5th July 2013, to appear.

List of Presentations

Meetings

1. *The FIST Approach for Pattern Mining in Large Database*, **Indian Statistical Institute**, Kolkata, India, December 2010.

2. *FIST: A Closure Based Approach*, **GLC Pole Day, I3S Laboratory**, Nice, France, September 2010.

International Conferences

3. *A New Approach for Association Rule Mining and Bi-clustering using Formal Concept Analysis*, **8th International Conference on Machine Learning and Data Mining (MLDM'2012)**, Berlin, Germany, July 2012.

4. *Prediction of Protein Interactions on HIV-1-Human PPI Data using a Novel Closure-based Integrated Approach*, **International Conference on Bio-informatics Models, Methods and Algorithms (BIOINFORMATICS'2012)**, Algarve, Portugal, February 2012.

5. *Simultaneous Clustering and Gene Ranking: A Multi-objective Genetic Approach*, **7th International Meeting on Computational Methods for Bio-informatics and Biostatistics (CIBB'2010)**, Palermo, Italy, September 2010.

Résumé

L'extraction de modèles de connaissances est l'un des principaux thèmes de recherche du domaine de l'Extraction de Connaissances á partir des Données (ECD) et pour l'intégration de connaissances initiales du domaine. Extraires de tels motifs à partir de grandes bases de données, entrepôts de données et autres types de répertoires massifs de données est l'une des tâches les plus ardues du processus global de fouille de données, ou data mining. *Parmi les nombreuses techniques de fouille existantes, l'extraction de règles d'association et le bi-clustering sont les deux approches descriptives majeures pour la découverte de connaissances pertinentes et utilisables, ainsi que pour l'intégration de connaissances initiales dans le processus de fouille. Même si ces approches ont fait l'objet de nombreux travaux et ont été appliquées dans de nombreux domaines, aucune approche permettant de combiner ces deux tâches dans un même processus n'a été proposée à notre connaissance. Dans cette thèse, nous proposons une approche originale pour l'extraction de modèles de connaissances de ces deux catégories en minimisant l'utilisation des ressources. Les modèles extraits, basés sur la théories des itemsets fermés fréquents et des listes d'objets support, sont utilisés pour construire des représentations conceptuelles minimales de règles d'association et de classification, et de bi-clusters. Ils étendent les modèles classiques de règles d'association et classification, ainsi que de bi-clusters, en fournissant à l'utilisateur davantage d'informations découlant des listes d'objets supportant chaque modèle. Ces modèles sont générés à partir des ensembles de générateurs, ou itemsets-clé, d'itemsets fermés fréquents et de la structure hiérarchique conceptuelle induite par les générateurs, les fermés fréquents et les listes d'objets support.*

L'approche proposée, nommée FIST *pour* Frequent Itemset mining using Suffix-Trees, *utilise une nouvelle structure de données basée sur les arbres suffixés qui permet le stockage efficace des données et l'extraction de modèles de connaissance pertinents en mémoire primaire. La stratégie utilisée par FIST est basée sur la fermeture de la connexion de Galois d'une relation binaire finie qui sert également de fondement théorique en analyse de concepts formels (FCA). Une caractéristique importante de FIST est qu'un seul balayage de l'ensemble de données est nécessaire pour en extraire tous les motifs valables. Cette caractéristique est primordiale pour un traitement efficace de grands ensembles de données stockées en mémoires secondaires, du fait des temps d'accès à ce type de mémoires qui sont environ un million de fois moins rapides que les accès en mémoire primaire. Nous avons développé deux algorithmes différents de mise en oeuvre de l'approche FIST, présentés dans ce rapport du point de vue de l'avancement continuel de mise au point de l'approche. Le premier algorithme utilise des opérations optimisées pour l'exploitation de tables et listes extraites de l'arbre suffixé. Le deuxième algorithme utilise une approche arborescente pure afin d'améliorer l'utilisation des ressources, à la fois en termes d'espace mémoire et de nombre d'opérations effectuées. Pour les deux algorithmes, l'utilisation d'une forme généralisée d'arbre suffixé permet de réduire l'utilisation de la mémoire et d'améliorer l'efficacité de l'extraction des motifs durant l'analyse de l'ensemble de données. Cette optimisation repose sur les propriétés de la structure de données qui, conjugué à un ordonnancement aproprié des données, limite l'espace de recherche des motifs fermés sans nécessiter de phase de génération de motifs candidats. En outre, les types d'opérations utilisés pour des motifs d'extraction permettent le traitement en parallèle des branches de l'arbre suffixé afin d'optimiser les temps d'exécution pour des architectures matérielles multi-processeurs et*

multi-coeurs. FIST est une approche intégrée basée sur la théorie de la fermeture de Galois qui combine l'extraction de générateurs, motifs fermés fréquents, règles d'association, de classification, et bi-clusters conceptuels, étendant ainsi les modèles classiques de connaissance pour une analyse conceptuelle.

Les résultats et analyses expérimentaux montrent les performances des différentes versions de FIST et leur comparaison avec d'autres algorithmes d'état de l'art pour l'extraction de règles d'association et itemsets fermés, et le bi-clustering. Aucun autre algorithme publié dans la littérature ne permet de générer les mêmes motifs que ceux générés par FIST à notre connaissance. Trois implémentations des deux différentes versions algorithmiques de FIST ont été implémentées en langage Java, choisi pour la portabilité. La première implémentation correspond à la première version de FIST utilisant des opérations basées sur des tables et listes. Les deux dernières implémentations correspondent à la deuxième version de FIST, l'une utilisant les collections standards de l'API Java et la seconde utilisant les collections de l'API Java Trove. Des expérimentations ont été menées avec la première implémentation sur diverses bases de données synthétiques et opérationnelles afin d'évaluer son applicabilité aux très grands ensembles de données. Cette implémentation a également été comparée aux algorithmes d'état de l'art en termes de temps d'exécution et d'utilisation de la mémoire primaire. Les résultats ont montré que les besoins en mémoire et temps d'exécution sont dans la plupart des cas équivalents à ceux des algorithmes basés sur les itemsets fermés fréquents et inférieurs à ceux des algorithmes basés sur les itemsets fréquents. Les résultats détaillés obtenus avec cette première implémentation nous ont conduit à améliorer l'approche, en transformant les opérations basées sur des tables et listes par des opérations purement arborescentes, afin de réduire les besoins en mémoire et temps d'exécution. Ces améliorations ont menés à la seconde version algorithmique de FIST implémentée en utilisant les collections standards de l'API Java et optimisée ensuite avec l'utilisation des collections de l'API Java Trove. Ces trois implémentations ont été comparées expérimentalement sur diverses configurations matérielles afin d'évaluer avec précision les gains obtenus par les améliorations successives de l'algorithme et l'utilisation des collections de l'API Java Trove. Les résultats ont confirmé les améliorations apportées par la seconde version algorithmique à la fois en termes d'utilisation mémoire et de temps d'exécution. L'utilisation mémoire et les temps d'exécution de la seconde version ont été ensuite comparés à ceux des algorithmes d'état de l'art pour l'extraction de règles d'association et itemsets fermés, et le bi-clustering.

La protéomique concerne l'étude à grande échelle des protéines dans les systèmes biologiques complexes (fluides, tissus, organes, etc.). Comme l'ensemble des domaines "omiques", la protéomique fait face à un accroissement exponentiel du volume de bioinformations, ou données biologiques, générées, tant en terme de valeurs que de variables, mais aussi de métadonnées définissant des structures complexes dans les données. Ces caractéristiques ont entraîné l'apparition de nombreuses nouvelles problématiques complexes pour l'extraction de connaissances à partir de ces données. L'approche FIST, développée pour répondre à certaines de ces problématiques, a été appliquées à l'analyse d'interactions protéomiques (PPI) entre les protéines du virus VIH-1 et de l'organisme humain. L'analyse d'interactions protéomiques est un domaine récent et complexe d'une importance majeure en bioinformatique. Les résultats obtenus ont permi de démontrer son rôle capital pour la découverte de nouveaux traitements et la prévention de diverses types de maladies. L'application aux interactions entre protéines du virus VIH-1 et de l'organisme humain vise à découvrir les interactions responsables du développement du virus du SIDA et permettre ainsi la création de traitements de la maladie. Afin d'étendre

les motifs extraits des données originelles d'interactions protéomiques entre VIH-1 et organisme humain, nous avons construits trois nouvelles bases de données intégrant les données, connaissances biologiques et annotations bibliographiques les plus récentes disponibles sur ce sujet. Les résultats expérimentaux ainsi que les nouveaux modèles de connaissances extraits par l'approche FIST pour ces bases de données sont présentées dans ce rapport. Afin de démontrer la validité de l'approche, les connaissances déjà reportées dans la littérature du domaine qui ont été extraites avec FIST sont également présentées. Les modèles extraits par FIST pour ces données sont constitutées des bi-clusters hiérarchiques conceptuels et des couvertures minimales conceptuelles de règles d'association contenant à la fois des informations d'interactions et d'annotations biologiques conernant les protéines.

Mots-clés: *Data Mining, Extraction de Connaissances à partir des Données, Bases de Règles d'Association, Règles de Classification, Règles d'Association Conceptuelles, Bi-clustering, Itemsets Fermés Fréquents, Treillis des Itemsets Fermés, Connexion de Galois, Analyse de Concepts Formels, Structures de Données, Arbres Suffixés.*

Abstract

Pattern extraction is one of the major topics in the Knowledge Discovery from Data (KDD) and Background Knowledge Integration (BKI) research domains. Extracting patterns from databases, data warehouses and other kinds of data repositories is one of the most unyielding tasks. Extensively, it is subsumed as a part of the data mining task. Out of numerous data mining techniques, association rule mining and bi-clustering are two major complementary data mining tasks for relevant knowledge extraction and integration. These tasks gained much importance in many research domains in recent years. However, to the best of our knowledge, no approach was proposed to perform these two tasks in one process. In this thesis work, we propose an original approach for extracting different categories of knowledge patterns while using minimum number of resources. These patterns, based on frequent closed sets and supporting object lists, are used to construct conceptual minimal representations of association rules, bi-clusters and classification rules. They extend the classical frameworks of association and classification rules, and of bi-clusters, by providing the user with more information using the object lists associated with these patterns. These patterns are generated from the sets of generators, or key-patterns, the sets of closed patterns and the hierarchical conceptual structure induced from generators, closed patterns and supporting object lists.

The proposed approach, named FIST *for* Frequent Itemset mining using Suffix-Trees*, is based on a new suffix-tree data structure that enables the efficient storage of data and computation of relevant patterns in primary memory. The strategy used by FIST is based on the closure of the Galois Connection of a finite binary relation theory used in the Formal Concept Analysis framework. An important feature of FIST is that it requires only a unique scan of the dataset to extract all valid patterns. This feature is required for efficient processing of large datasets, stored on hard drives, as secondary memory accesses are about one million times slower than primary memory accesses. We developed two different algorithms implementing the FIST approach; these algorithms are presented in this report as the continual advancement of the FIST application development. The first algorithm uses table and list-based optimized operations, where tables and lists are extracted from the suffix-tree. The second algorithm uses a pure tree-based approach to improve resource usage, both in terms of memory space and of number of operations performed. For both algorithms, the use of a generalized form of the suffix-tree data structure helps to reduce memory usage and improve the pattern extraction efficiency while analyzing the dataset. This optimization relies on the suffix-tree properties that, combined with item ordering, limits the search space for closed patterns without candidate generation. Moreover, the types of operations used for extracting patterns allow the parallel processing of the tree branches in order to optimize execution times in the case of multiple processors and cores computer architectures. FIST is an integrated approach based on the Galois closure framework, combining the searches for generators, frequent closed itemsets, association rules, conceptual bi-clusters and classification patterns, and extending the generated patterns for conceptual analysis.*

Experimental results and analyses show the performances of the different versions of FIST and compare them to others state-of-the-art algorithms for association rule mining, closed pattern mining and bi-clustering. To the best of our knowledge, no algorithm in the literature produces the same output patterns as are generated by FIST. Three implementations of the two different algorithmic versions of FIST were programmed in Java language that was chosen

for portability. The first implementation corresponds to the first version of FIST using tables and lists-based operations. The two latter implementations correspond to the second version of FIST, one was achieved using standard Java Collection APIs and the second using Trove Java APIs. Experiments were conducted with the first version on several synthetic and real life databases to assess its applicability to very large datasets. This version was also compared to other pattern mining state-of-the-art algorithms with respect to both execution times and memory usage. Results showed that memory requirements and execution times are in most cases equivalent to frequent closed itemsets based algorithms and lower than frequent itemsets based algorithms. After successful completion of the first version, we improved techniques from table and list-based to tree-based operations for reducing memory and time consumption. This resulted in the second algorithmic version of FIST that was implemented using standard Java API collections and Java Trove optimized API collections. All three versions were compared to show the amount of algorithmic improvements between first and second versions, and assess the impact of using Java Trove optimized collections on FIST. Results showed that the second version outperforms the first version in both execution times and memory usage. The second version was then separately compared to association rule mining, closed pattern mining and bi-clustering state-of-the-art algorithms with respect to both execution times and memory usage.

Proteomics focuses on the large scale study of proteins in complex biological systems (fluid, tissue, organ, etc.). As most of the high throughput "omics" fields, proteomics is facing a tremendous increase of the size of datasets to process (number of items as well as number of variables, but also their meta-data which structure the datasets into complex objects), leading to challenging knowledge discovery from data problems. The FIST application was applied for the analysis of a real life dataset of protein-protein interactions (PPI) between HIV-1 and Human proteins. Discovering protein-protein interactions is a recent major challenge in computational biology. Identifying interactions among proteins was shown to be useful for finding new drugs and preventing several kinds of diseases. The identification of interactions between HIV-1 proteins and Human proteins is a particular PPI problem whose study might lead to the discovery of important interactions responsible for AIDS and help designing drugs and treatments. In order to improve and extend knowledge patterns extracted from original HIV-1 and Human PPI data, we constructed three new datasets integrating the most recent biological and bibliographic annotations on proteins with PPI data. Successive experimental results for these PPI datasets, and new information discovered using the FIST approach on these datasets, are presented in this report. As proof of correctness, we have also shown that FIST successfully found the currently known information in the PPI literature. The experiments on these PPI datasets were performed by extracting with FIST the conceptual hierarchical bi-clusters and the conceptual minimal covers of association rules containing both interaction and annotation information on proteins.

Keywords: *Data Mining, Knowledge Discovery in Database, Bases of Association Rules, Classification Rules, Conceptual Association Rules, Bi-clustering, Frequent Closed Itemsets, Closed Itemset Lattice, Galois Connection, Formal Concept Analysis, Suffix-Tree Data Structure.*

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction & Scope of the Thesis

Contents

1.1 Introduction

Data mining, also known as knowledge discovery from database (KDD), is the task of extracting unknown and potentially important information from large databases. It is an important research area in Databases, Artificial Intelligence and Machine Learning [Piatetsky-Shapiro 1991b]. KDD is a combination of four different procedures:

1. Data collection: Involves collecting data from different sources like the web, data warehouses or databases. These data comes with several kind of noise in it. Filtering those noise from the data is an important step before using the data.

2. Preprocessing of collected data: Involves several different tasks, likes feature selection, feature ranking, sample selection, etc. Its purpose is to select the most important features or samples from the database needed for the data mining process. The selected features sometime comes with inappropriate format. Transforming them according to the approach involved in the data mining task is also part of the preprocessing procedure.

3. Data mining: The main phase of KDD during which data is processed and generates different kinds of outputs according to the approaches involved. Clustering, classification, regression, association rule mining, pattern selection, pattern matching are few among several data mining tasks [Han 2011].

4. Information retrieval: The mined outputs are interpreted by the application expert to get actual hidden information from the data collected in the first step. Wide acceptance in numerous application areas, from biology to spacial applications, makes it more popular for finding hidden knowledge from the dumped data.

Our main concern in this work is to the third step, i.e., data mining, that is considered as the most important and difficult in the KDD process. Among the most prominent data mining

tasks, gaining actually much importance in many application domains, are association rule mining, pattern selection and clustering that are the principle subjects of study in this thesis. Bi-clustering, that is a special case of clustering, is also gaining much popularity, specially in bioinformatics [Madeira 2004]. From this viewpoint, the work conducted during the thesis is related to the hierarchical bi-clustering task.

During this thesis work, we developed a new approach, named *FIST* for *Frequent Itemset mining using Suffix-Trees*, for extracting different types of knowledge patterns while using minimum number of resources. These knowledge patterns are conceptual minimal representations of association rules, bi-clusters and classification rules. They extend the classical frameworks of association rules, classification rules and bi-clusters by providing the user with additional information derived from the relationships between these patterns and the relationships between these patterns and the objects supporting them. They are generated from the hierarchical conceptual structure defined by generator itemsets, closed patterns and supporting object lists.

Association Rule Mining: Association rule mining (ARM) was first introduced in 1993 by Agrawal *et al.* in [Agrawal 1993]. It aims at finding significant relationships between data values, called items, in a database. ARM is a very popular and important, but computationally expensive, task in data mining. Since the ARM problem definition, several approaches have been proposed in the literature to improve ARM efficiency. See [Ceglar 2006] for a complete survey on ARM principles and algorithms. The ARM problem is generally divided into two sub-problems:

1. Find all frequent itemsets, i.e. itemsets which frequency in the database is at least equal to a user defined threshold, with their support from the database.

2. Generate all association rules with confidence greater than or equal to the user defined minimum confidence threshold.

The second sub-problem is straightforward since association rules are generated in a direct way from the selected frequent itemsets and their support. It thus requires simpler processes and less resources than the first sub-problem. On the contrary, the first sub-problem is solved using different processes and involves complex techniques to handle large and dense databases. As a result, it gives the itemset patterns that are present in a significant number of rows of the database. The problem of ARM is thus generally reduced to the problem of finding frequent itemsets. As stated before, ARM requires in most cases two user-defined threshold values: The first limits the frequent itemsets extracted in the first phase to the significant ones with respect to their frequency; The second limits the association rules generated in the second phase to the most informative ones with respect to their precision.

Frequent Itemset Mining: Frequent Itemset Mining (FIM), where frequent itemsets are defined based on the minimum support threshold, is the most important and difficult part of the ARM problem. Almost all algorithms based on the frequent itemsets approach use the subset lattice, or itemset lattice, framework. This approach relies on two following properties:

i. All subsets of a frequent itemset are frequent.

ii. All supersets of an infrequent itemset are infrequent.

For big databases, the list of all frequent itemsets is most often very large, and even larger in the case of dense data where transactions are large. So, the reputation of the frequent itemset approach falls for it's large memory requirement and execution time to extract and process the frequent itemsets. Due to these disadvantages of frequent itemsets based approaches, FIM was extended to the mining of maximal frequent itemsets (MFI). MFI are the maximal itemsets, with respect to inclusion relation, that are frequent given the minimum support threshold. It was shown in the literature that all frquent itemsets can be derived from the MFI. However, the MFI approaches also presents computational problems since from the list of MFI, one can derived all frequent itemsets but not their support that must be computed from the database.

Closed Itemset Mining: Due to the disadvantages of the FI and MFI approaches mentioned above, closed itemsets mining was introduced in 1998 in [Pasquier 1998]. The *frequent closed itemsets* (FCI) form a lossless minimal representation of the set of frequent itemsets as all frequent itemsets and their support can be generated directly from FCI in a straightforward manner, without database access. More detailed studies of properties of FCI as a condensed representation of frequent itemsets can be found in [Pasquier 1999b, Zhang 2000]. The frequent closed itemsets, defined using the *Galois closure connection* [Birkhoff 1995a, Wille 1992, Davey 1994] used in Formal Concept Analysis [Ganter 1999], form the *closed itemset lattice* [Pasquier 1999b] that is a sub-order of the subset lattice. Informally, an itemset is closed if none of its supersets are contained in exactly the same objects (database rows) as it. Since the number of FCI is in most cases much lower than the number of FI, their computation is a solution to the first ARM sub-problem and many applications showed that it clearly improves execution times and memory usage. ARM extraction using the FCI framework relies on the three following properties:

i. All subsets of a frequent closed itemset are frequent itemsets.

ii. All supersets of an infrequent closed itemset are infrequent itemsets.

iii. The support of a frequent itemset is equal to the support of the smallest closed itemset containing it.

Many algorithms have been proposed in the literature in recent years for finding FCIs [Shekofteh 2010, Yahia 2006]. Almost all of them use either the prefix tree [Agrawal 1994] or the FP-Tree [Han 2000a] as an internal data structure for compressed representation of the dataset in main memory. Their efficiency depends mainly on the properties of the database in terms of number of items, density of the data matrix, length of transactions, etc. In several cases, such as biological data, FI and MFI mining pose efficiency problems since the number of FI is very large and the set of MFI does not contain all information required to directly generate association rules. In such cases, the FCI framework is a sound alternative for the ARM problem as the set of FCI is sufficient for finding all association rules while being much smaller than the set of FI.

Bi-clustering: Clustering is the process of grouping elements that are significantly similar in *clusters* without advance knowledge of the group definitions. Clusters are constructed and evaluated to maximize the intra-cluster similarity and minimize the inter-cluster similarity. Bi-clustering is a special case of clustering that aims at finding sub-matrices associating a set of

rows and a set of columns such that all these rows have similar values for the corresponding columns. Bi-clustering has been widely used for genomic analyses in the context of bioinformatics. In gene expression data matrices, it allows to identify sets of genes, represented as rows, that have the same expression profile, i.e., are over-expressed, under-expressed or unchanged, with respect to the columns that represent gene expression for different biological conditions. Formal Concept Analysis can be related to mining bi-clusters [Ganter 1999] where *extensions* and *intentions* of concepts are comparable to row and column sets, respectively, of a bi-cluster. Bi-clusters extracted by FIST are maximal sets of related rows and columns as defined above, where maximality is defined according to the inclusion relation. In the context of gene expression data analysis, each bi-cluster extracted by FIST is a sub-matrix where each row (gene) of the *extension* has the same expression profile as all other rows of the *extension* for all columns (biological condition) of the *intension*. These bi-clusters, called *conceptual clusters*, constitute a hierarchical lattice structure defined according to the inclusion relation. These bi-clusters can also overlap with each other: Each row and each column can be member of several bi-clusters. Contrarily to recent works on gene expression time series [Madeira 2009], where columns represent the evolution of gene expressions during time for one biological experiment, our approach does not restrict bi-clusters to contiguous columns. Extracting such bi-clusters is known to be an NP-Complete problem [Peeters 2003].

Suffix Tree Data Structures: Suffix trees are important data structures for storing patterns of ordered elements and were shown to be very efficient for performing string processing tasks. A suffix tree based index structure needs $O(|s|)$ time and space resources to compute and store a string s. Suffix trees have a large number of applications in different areas including bioinformatics. For processing a set of previously unknown patterns, a suffix tree structure can be used to decompose and store the useful patterns, and all suffixes of a pattern p in the suffix tree can be found in $O(|p|)$ steps. Despite these qualities, using suffix tree data structures presents two difficulties:

- The linear time construction procedure of suffix trees are quite complex to implement.
- Although suffix trees are asymptotically linear, it's greedy for space.

In the context of frequent closed itemset extraction, the combination of a suffix tree data structure and the ordering of items according to their support in the dataset permits the FIST approach to reduce the search space by identifying almost all and only frequent closed itemsets while reading the dataset and incrementally constructing the suffix tree. For this dataset processing task, FIST uses a generalized version of the suffix tree structure in which several strings representing itemsets are composed in a single suffix tree.

HIV-1 and Human Protein-Protein Interaction Data: Acquired Immune Deficiency Syndrome (AIDS) is the last stage of HIV infection. At this stage, the human immune system fails to protect the body from infections, and this eventually leads to death. HIV is a member of the retrovirus family (lentivirus) which infects important cells in the human immune system. This kind of infection is due to the interaction between proteins of both the virus and the human host in the human cells. Predicting such interactions is an important goal of PPI research. In particular, analyzing well-known interactions and finding new interactions can provide useful information to find new drugs and discover the reasons and mechanisms of this kind of viral

disease [Arkin 2004]. PPI databases contain information about the fact that proteins can interact if they come into contact. The absence of such information does not imply that they cannot interact with each other as there is no information about non-interacting proteins. The HIV-1–Human PPI dataset is a database containing possible viral and human protein interactions. As stated above, only positive interactions are depicted in the database used for experiments.

Several approaches for predicting protein interactions have been studied in the literature. These approaches are based on Bayesian networks [Jansen 2003], random forest classifiers [Lin 2004], mixture-of-feature-expert classifiers [Qi 2007], kernel methods [Yamanishi 2004, Ben-Hur 2005a], or decision trees [Zhang 2004]. Most of them have been used to find interactions within a single organism, like yeast or human (intra-species interactions). Recently, two approaches have been proposed to predict the set of interactions between HIV-1 and human host cellular proteins [Tastan 2009, Mukhopadhyay 2010]. In particular, in [Tastan 2009] the authors proposed a supervised learning framework that integrates heterogeneous biological information to predict inter-species interactions. However, this approach solves the classification problem using the random forest classifier which, like most of the above mentioned approaches, needs both positive and negative samples of PPIs. Here negative samples are pairs of human and HIV proteins known not to interact, but such "negative interactions" (or, better, prove absence of interactions) are not known in the current state of knowledge in the PPI problem studied here. Negative samples have then to be prepared, for example by randomly selecting protein pairs that are not present in the database, thus leading to a high dependency between the classifier performance and the choice of the negative samples. The approach proposed in [Mukhopadhyay 2010] uses the well-known *Apriori* algorithm for mining association rules. The particularity of such an approach is that only information based on positive samples is used to predict viral-human interactions (inter-species interactions). This is also the case for the application presented in this thesis.

1.2 Definitions

In this section, we define the most important terms, related to the problems of association rule mining and conceptual bi-clustering in a data mining context, used in this report.

Definition 1.1 (Database) *A* database $\mathcal{D}$ *is a structured set of data representing binary relationships between elements of two finite sets called* O *and* A *corresponding respectively to objects, or instances, and attributes, or variables. A peculiar representation of the information in a database* $\mathcal{D}$*, in a matrix or transactional format, is called a* dataset.

Definition 1.2 (Dataset) *A binary matrix format dataset* M *is a triplet* (O, L, R) *where* O *is a finite set of objects (rows),* L *is a finite set of items (values of attributes or variables) represented as columns and* R *is a binary relation showing relationships between rows and columns:* $R \subseteq O \times L$*. Every couple* $(o, i) \in R$*, where* $o \in O$ *and* $i \in L$*, means that the item* i *belongs to the object* o*:* $i \in o$*. A transactional format dataset* T *is a set of instances (rows), called transactions,* $T = \{t_1, t_2, \cdots, t_n\}$ *where each transaction* t_i *is a set of items from the finite list of items* $L = \{i_1, i_2, \cdots, i_m\}$*, i.e.* $t_i \subseteq L$*.*

Definition 1.3 (Itemsets) *A non-empty finite set of items* $I \subseteq L = \{i_1, \ldots, i_m\}$ *in a database*

$\mathcal{D}$ is called an itemset. *An itemset containing k items is called a k-itemset. That is, $I_k = i_1, i_2, \cdots, i_k$ and $|I_k| = k$.*

Definition 1.4 (Support) *The support of an itemset I in a database $\mathcal{D}$, denoted* support(I), *is the frequency of occurrence of I in $\mathcal{D}$, that is the proportion of objects $o \in O$ containing I:*

$$support(I) = \frac{|\{o \in O \mid I \subseteq o\}|}{|\{o \in O\}|} \tag{1.1}$$

Definition 1.5 (Frequent Itemsets) *An itemset I with support in database $\mathcal{D}$ at least equal to the* minsupport *threshold value is called a* frequent itemset. *Let $\mathcal{F}$ be the set of frequent itemsets. $\forall I \subseteq L$, we have $I \in \mathcal{F}$ if and only if $support(I) \geq minsupport$.*

Definition 1.6 (Maximal Frequent Itemsets) *A frequent itemset $I \subseteq L$ is called a* maximal frequent itemset *if none of its proper supersets $I' \supset I$ is frequent, i.e. present in the set $\mathcal{F}$ of frequent itemsets. Let $\mathcal{MF}$ be the set of maximal frequent itemsets. $\forall I \in \mathcal{MF}$, $\nexists I' \subseteq L$ such that $I' \in \mathcal{MF}$ and $I' \supset I$.*

Definition 1.7 (Galois Connexion Closure Operator [Ganter 1999]) *Let ψ be a function associating to an itemset $I \subseteq L$ the set of objects $P \subseteq O$ such that $\forall o \in P$, $I \subseteq o$. Let γ be a function associating to a set of objects $P' \subseteq O$ the set of items $I' \subseteq L$ such that $\forall i \in I'$, $i \in o$ for all $o \in P'$.*
The ψ and γ functions are monotonically decreasing: $\forall I_1, I_2 \subseteq L$ with $I_1 \subseteq I_2$ we have $\psi(I_1) \supseteq \psi(I_2)$ and $\forall P_1, P_2 \subseteq O$ with $O_1 \subseteq O_2$ we have $\gamma(O_1) \supseteq \gamma(O_2)$.
The Galois closure operator $\Gamma = (\psi \circ \gamma)$ holds the following properties for I, I_1, $I_2 \subseteq L$ in the power set of L of size 2^L:

- *Extension:* $I \subseteq \Gamma(I)$
- *Idempotency:* $\Gamma(\Gamma(I)) = \Gamma(I)$
- *Monotonicity:* $I_1 \subseteq I_2 \Rightarrow \Gamma(I_1) \subseteq \Gamma(I_2)$

Definition 1.8 (Closed Itemsets) *An itemset $I \subseteq L$ is said to be* closed *in database $\mathcal{D}$ if the application to I of the Galois connexion closure operator Γ gives I: $\Gamma(I) = I$. If I is a closed itemset in $\mathcal{D}$, none of its proper supersets in $\mathcal{D}$ has support less than or equal to the support of I. Let $\mathcal{C}$ be the set of closed itemsets. We have: $\forall I \in \mathcal{C}$, $\nexists I' \subseteq L$ such that $I' \in \mathcal{C}$ and $support(I') \leq support(I)$.*

Definition 1.9 (Frequent Closed Itemsets) *A closed itemset $I = \Gamma(I)$ which support in database $\mathcal{D}$ is greater than or equal to the user defined* minsupport *threshold value is called a* frequent closed itemset. *Let $\mathcal{C}$ be the set of closed itemsets and $\mathcal{FC} \subseteq \mathcal{C}$ be the set of frequent closed itemsets. $\forall I \in \mathcal{C}$, we have $I \in \mathcal{FC}$ if and only if $support(I) \geq minsupport$. Maximal frequent itemsets were shown to be frequent closed itemsets: $\mathcal{MF} \subseteq \mathcal{FC}$* [Pasquier 1999b].

Definition 1.10 (Generators) *A* generator *$G \subseteq L$ of a frequent closed itemsets $C \subseteq L$ is a minimal itemset, regarding inclusion relation, which closure is C: $\Gamma(G) = C$ and $\nexists G' \subseteq L$ such that $G' \subset G$ and $\Gamma(G') = C$. The set of generators $\mathcal{G}_C = \{G_1, \ldots, G_q\}$ of a frequent closed itemset C contains all minimal itemsets, according to inclusion relation, which closure is C. Generators are the minimal itemsets with same support as the closed itemset C that is their closure: $\forall G_i \in \mathcal{G}_C$, we have $support(G_i) = support(C)$. A generator that is frequent according to the* minsupport *value is called a* frequent generator.

Definition 1.11 (Equivalence Classes) *An* equivalence class E_C *is a dual set* $E_C = \{\mathcal{G}_C, C\}$ *where* $\mathcal{G}_C = \{G_i \subseteq L \mid \Gamma(G_i) = C\}$ *is the set of generators of the frequent closed itemset* $C \in \mathcal{FC}$ *for database* $\mathcal{D}$*. All itemsets contained, regarding inclusion relation, in an equivalence class* E_C *have the same support as the closed itemset* C*:* $\forall I \subseteq L$ *where* $I \supseteq G_i \mid G_i \in \mathcal{G}$ *and* $I \subseteq C$ *we have* $support(I) = support(C)$*. By extension, we say that itemset* I *is* contained *in the equivalence class* E_C*. The closed itemset* C *is the maximal itemset of the equivalence class:* $\nexists I \in E_C$ *such that* $C \subset I$*.*

Definition 1.12 (Association Rules) *The implication relationship between two itemsets* I_1 *and* I_2 *with the form* $r : I_1 \longrightarrow I_2$ *where* $I_1, I_2 \subset L$ *and* $I_1 \cap I_2 = \emptyset$ *is called an* association rule*.* I_1 *and* I_2 *are respectively called the antecedent and the consequent of the rule. The support of an association rule* r *is the support of the union of its antecedent and consequent:* $support(r) = support(I_1 \cup I_2)$*.*

Definition 1.13 (Confidence) *The* confidence *of an association rule* $r : I_1 \longrightarrow I_2$ *is the ratio of the support of the itemset* $I_1 \cup I_2$ *to the support of the antecedent of the rule:*

$$confidence(r) = \frac{support(I_1 \cup I_2)}{support(I_1)} = \frac{support(r)}{support(I_1)} \tag{1.2}$$

The confidence measures the proportion of database $\mathcal{D}$ *objects that verify the association rule. By extension, the proportion of counter-examples of the association rule* r *in the database* $\mathcal{D}$ *is:* $1 - confidence(r)$*.*

Definition 1.14 (Valid Association Rules) *An association rule* $r : I_1 \longrightarrow I_2$ *where* $I_1 \cup I_2$ *is a frequent itemset and which confidence* $confidence(r)$ *is greater than or equal to the* minconfidence *threshold value is called a* valid association rule*. Let* $\mathcal{R}$ *be the set of valid association rules.* $\forall r \in \mathcal{R}$ *we have* $confidence(r) \geq minconfidence$ *and* $support(r) \geq minsupport$*.*

Definition 1.15 (Exact Association Rules) *Valid association rules with confidence equals to 1 are called* exact association rules*. Let* $\mathcal{R}_E$ *bet the set of exact association rules.* $\forall r \in \mathcal{R}$ *we have* $r \in \mathcal{R}_E$ *if and only if* $confidence(r) = 1$*. Exact association rules are rules verified by all objects in database* $\mathcal{D}$*, thus having no counter-examples.*

Definition 1.16 (Approximate Association Rules) *Association rules with confidence less than 1 are called approximate association rules. Let* $\mathcal{R}_A$ *bet the set of approximate association rules.* $\forall r \in \mathcal{R}$ *we have* $r \in \mathcal{R}_A$ *if and only if* $confidence(r) < 1$*. Approximate association rules are rules not verified by all objects, and thus having counter-examples, in database* $\mathcal{D}$*.*

Definition 1.17 (Clusters) *A* cluster *is a subset of objects that are similar according to a distance metric* $d(o_i, o_j)$ *which value is computed by comparing variable values of objects* o_i *and* o_j*. For a database* $\mathcal{D}$*, clusters are defined as* $C_k = \{o \subset O \mid \forall o_i, o_j \in o, d(o_i, o_j) \leq \sigma\}$ *where* σ *is a user-defined threshold.*

Definition 1.18 (Bi-clusters) *A* bi-cluster B_k *extracted from a matrix representation* M *of a database* $\mathcal{D}$ *is a sub-matrix associating a subset of rows and a subset of columns such that all these rows have a similar value for each of these columns and all other rows have non-similar values for these columns.* $B_k = \{P, I\}$ *with* $P \subseteq O, I \subseteq L$ *such that:*

i) $\forall o_p, o_q \in P$ *we have* $d_{\cup i \in I}(o_p, o_q) < \sigma$.

ii) $\nexists o_r \in O, o_r \notin P$ *such that* $\forall o_s \in P$ *we have* $d_{\cup i \in I}(o_r, o_s) > \sigma$.

where $d_{\cup i \in I}(o_p, o_q)$ *is the distance between objects* o_p *and* o_q *computed on all items* $i \in I$ *and* σ *is a user-defined similarity criterium.*

1.3 Background and Motivation

Early approaches to association rule mining showed that the problem can be divided into two parts: First, find frequent itemsets with their supports, which is the most time-consuming part, and then generate association rules from these itemsets [Agrawal 1996]. Then, the frequent closed itemsets (FCI) framework was defined to improve the efficiency of the mining in case of non-sparse data [Pasquier 1999b, Zhang 2000]. The frequent closed itemsets, defined using the Galois closure [Ganter 1999], are a sub-order of the subset lattice, or itemset lattice. This framework was later used to define minimal covers, or *bases*, of association rules [Bastide 2000, Pasquier 2005, Zaki 2004]. The FCI approach relies on the property that the frequent closed itemsets with supports constitute a non-redundant minimal representation of the frequent itemsets and their supports. It was experimentally shown that the set of frequent closed itemsets is on average much smaller for real-life datasets, thus making this process faster than directly mining frequent itemsets. Association rules, or association rule bases, are then directly generated from the frequent closed itemsets. See [Ceglar 2006] for a comprehensive survey on association rule mining.

The FIST approach aims at providing the user with a minimal lossless set of knowledge patterns representing relationships between data values in the dataset. These compact sets of patterns can then be searched for specific information such as intra and inter-species protein interactions, or relationships between protein interactions and features (biological annotations and characteristics, publications, etc.). Extracted patterns depict relationships between viral proteins, host proteins and between both of them.

Let $V = \{v_1, \ldots, v_N\}$ be the set of viral proteins and $H = \{h_1, \ldots, h_M\}$ the set of human host proteins. We consider three possible kinds of patterns:

- $r_1 : v_1, v_2, \ldots, v_n \Longleftrightarrow h_1, h_2, \ldots, h_m$ where $v_i \in V$, $h_j \in H$;
- $r_2 : v_1, v_2, \ldots, v_n \Longrightarrow v_{n+1}, v_{n+2}, \ldots, v_{n+p}$ where $\{v_1, v_2, \ldots, v_n\} \cap \{v_{n+1}, v_{n+2}, \ldots, v_{n+p}\} = \emptyset$ and $v_i \in V$;
- $r_3 : h_1, h_2, \ldots, h_m \Longrightarrow h_{m+1}, h_{m+2}, \ldots, h_{m+q}$ where $\{h_1, h_2, \ldots, h_m\} \cap \{h_{m+1}, h_{m+2}, \ldots, h_{m+q}\} = \emptyset$ and $h_j \in H$.

Type r_1 relationships capture interactions between some viral proteins and some host proteins (inter-species PPI). Identifying such rules is similar to the problem of bi-clustering, that is, in the context of FIST, finding frequent closed itemsets with related object identifiers. Type r_2 and r_3 relationships are association rule patterns showing implications among viral proteins and host proteins respectively (intra-species PPI). Classification methods usually need both positive and negative examples of the predicted class, e.g., interacting and non-interacting protein pairs, in order to achieve an optimal supervised classification. However, in the case of HIV-1–Human PPI, information on non-interacting pairs of proteins is not available [Fu 2009, Ptak 2008].

Hence, descriptive methods, such as unsupervised classification (clustering) and association rule extraction, seems better suited to this PPI problem.

FIST was designed both to extract in one process different kinds of knowledge patterns, bi-clusters and association rules, and to extract additional information for each of these patterns compared to classical approaches. It can process discrete numerical, boolean, textual and nominal data. As for the majority of similar methods, in the case of continuous numerical data, a discretization method has to be applied before processing the data with FIST. This is for example the case for numerical gene expression data where numerical values must be discretized to identify "up-regulated", "unchanged" and "down-regulated" genes (rows) for each experimental biological conditions (columns). See [Yang 2010] for a recent discussion on discretization methods used in data mining.

Consider the example dataset D_1 in Table 1.1 where H_1 to H_6 are human proteins, V_1 to V_5 are viral proteins and Annot columns represent annotations of human proteins extracted from biological knowledge bases (Gene Ontology, KEGG, etc.) and publication bases (Pubmed, Reactome pathways, etc.). These annotations, represented as nominal data, describe biological knowledge on human proteins such as biological functions or characteristics (F_n) or bibliographic citation references (B_m). A "1" in column V_i for row H_j means that there is a positive (i.e., experimentally verified) interaction between H_j and V_i, while "-" means that no interaction has been reported. For example, we can state that there is a positive interaction between human protein H_1 and viral proteins V_1, V_3 and V_4, while no interaction between H_1, V_2 and V_5 has been reported. Besides, we can also state that H_1 is annotated by biological annotations F_1 and F_2 and referenced by bibliographical annotation B_1.

Table 1.1: Example Dataset D_1

OID	**V_1**	**V_2**	**V_3**	**V_4**	**V_5**	**Annot**	**Annot**	**Annot**
H_1	1	-	1	1	-	F_1	F_2	B_1
H_2	1	-	1	-	-	F_2	B_1	B_2
H_3	-	-	1	-	1	B_3	-	-
H_4	1	-	1	1	-	F_2	F_3	B_1
H_5	-	1	-	-	-	F_4	-	-
H_6	1	-	1	1	1	F_2	B_1	B_3

Conceptual bi-clusters extracted by FIST form a hierarchical structure and both HIV and Human proteins can participate to several bi-clusters according to their co-occurrences in the data. In the context of HIV-1-Human PPI, each conceptual cluster associates a list of HIV proteins and a list of Human proteins that interact. FIST bi-clusters also associate to each bi-cluster the minimal set of common properties, called *generators*, required to construct it [Hamrouni 2008a, Pasquier 1999b]. Moreover, unlike most clustering methods, conceptual clustering does not need to define the number of clusters before the process as data are grouped according to their co-occurrences in the dataset. The Hasse diagram of the lattice structure of the four bi-clusters extracted from D_1 for $minsupport = 2/6$ is shown in Figure 1.1. The top bi-cluster in this figure is irrelevant from the viewpoint of informativeness and is not generated by FIST; it is represented here for completeness of the lattice. Examining the rightmost bi-cluster, we can see in this lattice that human proteins H_3 and H_6 both interact with viral proteins V_3 and V_5 and are cited in bibliographical reference B_3. The leftmost bi-clusters show that human proteins H_1, H_2, H_4, and H_6 all interact with viral proteins V_1 and V_3, are all

annotated with F_2, and are cited in bibliographical reference B_1 and that human proteins H_1, H_4, and H_6 all interact with viral proteins V_1, V_3, and V_4, are all annotated with F_2, and are cited in bibliographical reference B_1. We can also see that the viral protein that interacts with the greatest number of human proteins is V_3, which interacts with H_1, H_2, H_3, H_4, and H_6, and that this interaction is the only property common to these five human proteins. It should be noted that for this *minsupport* value, there are 4 frequent closed itemsets, whereas there are 37 frequent itemsets for dataset D_1. These frequent closed itemsets are represented in the left element of the bi-clusters.

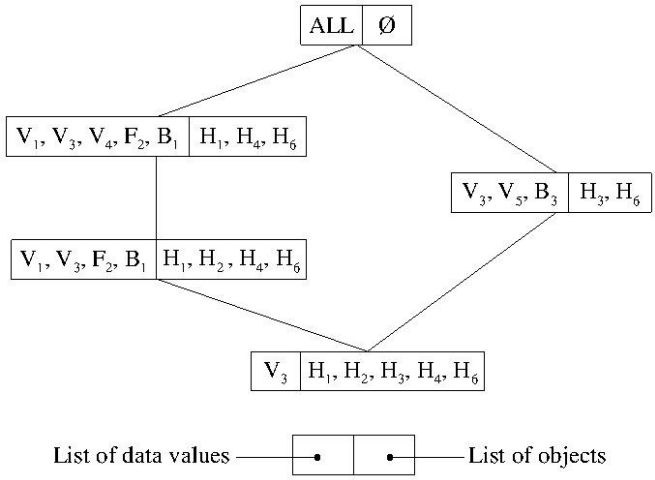


Figure 1.1: Hierarchical Conceptual Clusters of the database D_1.

Association rules are implication rules of the form: $\{r$: *antecedent* $\Longrightarrow$ *consequent*, support(r), confidence$(r)\}$ where *antecedent* and *consequent* are sets of data values, support(r) is the number of objects (rows of the dataset) supporting the rule and confidence(r) is the proportion of rows verifying the rule in the dataset. FIST aims at improving the process compared to frequent itemsets based approaches. First, the number of extracted rules can be reduced by a significant proportion as redundant rules can represent the majority of extracted rules [Bastide 2000, Zhang 2000]. Association rules extracted by FIST are constructed using generators, as antecedents, and frequent closed itemsets, as consequents. These rules, also called *min-max association rules*, constitute the *informative base of association rules* [Pasquier 2005]. FIST extracts rules in two distinct sets: *exact association rules* that have confidence = 1, i.e., with no counter example in the dataset, and *approximate association rules*, having confidence < 1. It also extends the association rules by adding information to each rule: The list of objects (rows) supporting each one is also generated, allowing the user to see which objects verify this rule in the dataset as shown in Table 1.2. We can see that for *minsupport* $= 2/6$ and *minconfidence* $= 2/6$, 6 exact and 6 approximate min-max association rules are generated by FIST from dataset D_1 whereas 192 association rules (117 exact and 75 approximate) are generated by classical Apriori-like approaches.

The motivation comes from two different works proposed in two different data mining tasks:

- The main motivation came from the work proposed in [Madeira 2009] by Madeira *et al.*

Table 1.2: Minimal Non-Redundant Association Rules

Association rule	**Support**	**Confidence**	**Objects**
$B_1 \Longrightarrow V_1, V_3, F_2$	4	1	H_1, H_2, H_4, H_6
$F_2 \Longrightarrow V_1, V_3, B_1$	4	1	H_1, H_2, H_4, H_6
$V_1 \Longrightarrow V_3, F_2, B_1$	4	1	H_1, H_2, H_4, H_6
$V_4 \Longrightarrow V_1, V_3, F_2, B_1$	3	1	H_1, H_4, H_6
$B_3 \Longrightarrow V_3, V_5$	2	1	H_3, H_6
$V_5 \Longrightarrow V_3, B_3$	2	1	H_3, H_6
$V_3 \Longrightarrow V_1, F_2, B_1$	4	0.80	H_1, H_2, H_4, H_6
$B_1 \Longrightarrow V_1, V_3, V_4, F_2$	3	0.75	H_1, H_4, H_6
$F_2 \Longrightarrow V_1, V_3, V_4, B_1$	3	0.75	H_1, H_4, H_6
$V_1 \Longrightarrow V_3, V_4, F_2, B_1$	3	0.75	H_1, H_4, H_6
$V_3 \Longrightarrow V_1, V_4, F_2, B_1$	3	0.60	H_1, H_4, H_6
$V_3 \Longrightarrow V_5, B_3$	2	0.40	H_3, H_6

where a CCC approach (Contiguous Column Coherent clusters) is proposed for extracting bi-clusters. In this work, a classical suffix tree structure is used for finding related contiguous columns from time series gene expression data, where columns represent the evolution of the gene expressions during time for one biological experiment.

- Also the use of object lists for pattern mining was introduced with the CHARM algorithm in [Zaki 2002]. In this paper, the authors use an Itemset-Tidset Tree (IT Tree) and complex mechanisms to handle the tree. However, the CHARM algorithm discards object lists as soon as they are no more needed whereas FIST keeps them in main memory for generating the conceptual extensions of association rules and bi-clusters and generate patterns with explicit names/labels of objects and data values (items). Conceptual extensions correspond to the lists of supporting objects for each association rule and the minimal sets of objects for one pattern, i.e., generators. Contrarily to classical ARM approaches where only support value of association rules are generated, conceptual association rules provide the user with more information as it allows him/her to examine the list of objects concerned by each rule.

The work presented here is an extension and combination of several ideas pointed out in the above two works. We extended these works and do not restrict the extraction to contiguous columns and also keeps track of all the object list till the end of the results. The approach proposed in [Madeira 2009] can be seen as a donor behind the idea of constructing the data structure used in FIST and the approach proposed in [Zaki 2002] contributes to the idea of keeping object id lists and showing it's importance for association rule mining.

1.4 Contribution

The objective of this thesis work was to integrate two different problems of data mining into a common paradigm to generate a user-friendly set of knowledge patterns and to apply the resulting solution to different problems in interaction proteomics. Both association rule mining

and bi-clustering have shown an important potential to support proteomic analysis. The FIST approach integrates association rule extraction and bi-clustering using the frequent pattern mining framework. Most often, these three aspects are applied separately even when used for the same application and databases. This induce complex tasks to merge and compare results of these different approaches. Moreover, resulting patterns are usually represented as lists of row and item sequential numbers, and a resource expensive task must be conducted to map these identifiers to values that are understandable and exploitable by the user. This last stage can be particularly important when biological knowledge, such as annotations or relationships of biological entities, is required to interpret extracted patterns and push the analysis further. These problems can be a critical limiting factor for using these data mining tools in an operational context [Hirji 2001].

The FIST approach was proposed to solve these problems simultaneously, in one platform, to minimize the resources required to produce the final user-friendly knowledge. Knowledge patterns generated by FIST are conceptual minimal representations of association rules, bi-clusters and classification rules. They extend the classical frameworks of association and classification rules, and of bi-clusters by providing the user with more information using the object lists associated with these patterns. This additional information can be particularly useful in a certain number of applications such as genomics or proteomics where identifying specific genes or proteins concerned by a profile or a rule is important, particularly if the profile or rule contains semantic information such as biological annotations. These patterns are generated from the sets of generators, or key-patterns, and closed itemsets and the hierarchical conceptual structure induced from generators, closed patterns and supporting object lists. From the viewpoint of association rules, the size of these object list identifiers, corresponding to the number of occurrences of the patterns in the database, gives the support of the FCI. From the viewpoint of bi-clustering [Madeira 2004], the generated bi-clusters form a hierarchical lattice structure and can overlap, allowing an object to belong to several bi-clusters, if relevant. This hierarchical structure of bi-clusters can show important additional information depicted by inclusion and overlapping relationships between sets of common features and sets of objects supporting these features.

The FIST approach is based on the FCI framework and an adapted suffix-tree data structure used for computations in main memory. Informally, a suffix-tree is tree structure that stores all suffixes of an ordered set of elements. The suffix-tree based data structure used is called *Generalized Suffix-Tree*, or *Suffix-Forest*, and is composed of multiple suffix-trees under a unique root. Unlike many tree based data structures, this data structure does not require complex operations, such as maintaining transverse chained lists of items or linking procedure for suffix relations, and can be readily implemented using standard data structures like standard Java Collection API. This feature allows the data structure to be easily extended or adapted, for example to introduce additional background domain knowledge directly in the data and/or in the process. It also allows to easily reproduce the results of FIST in the cases of implementations in other applications and/or languages. Using the generalized suffix-tree data structure, FIST limits to one the number of dataset scans required and the number of data accesses in main memory. The main steps of the FIST approach are the following:

1. The *Frequent Generalized Itemset Suffix Tree* (fGIST) is created from the database in secondary memory and stored in main memory for further processing.

2. Frequent closed patterns are extracted from the fGIST by performing inclusion and in-

tersection operations.

3. Bases of conceptual association rules, conceptual bi-clusters and generators are generated in a straightforward manner.

Bases of conceptual association rules are minimal non-redundant covers of association rules with supporting object sets. Depending on the parameters used, FIST can generate different bases corresponding to different criteria, e.g., minimal or maximal antecedents, or classification rules with a class label in the consequent of the rule are some examples.

Two algorithmic versions of the FIST approach were developed and implemented. The first version makes use of the generalized suffix-tree to efficiently store and process the dataset in main memory and remaining operations, i.e., inclusion tests and intersection operations, are performed on ordered tables generated from the tree. These operations are required as some closed itemsets are not directly inserted in the generalized suffix-tree while creating it and some non-closed itemsets need to be suppressed from the suffix-tree. The number of such itemsets is however limited, as their proportion is very low compared to the total number of closed itemsets, and few operations have to be performed during this phase. In the second version, inclusion tests and intersections are performed in the generalized suffix-tree directly and the leaf node data structure of the tree was modified to optimize these treatments. This second version was implemented using two different standard Java collections, the Java Collections API and the Trove for Java Collections API, to compare performances in both memory usage and computational efficiency of these popular data structure collections. The Trove for Java collections[1] was chosen after comparison with the Google Guava collections[2], the Javolution library[3], the Apache Commons Collections[4] and the SourceForge PCJ Project collections[5]. This second algorithmic version of FIST allows parallel processing of the generalized suffix-tree branches in multi-threated environments, although the current version does not support parallel processing/multi-threading.

The FIST approach was applied for biological knowledge discovery from different HIV-1 and human protein-protein interaction databases. The Human Immunodeficiency Virus type 1 (HIV-1) is the predominant retrovirus (lentivirus) that causes the last stage of Acquired Immune Deficiency Syndrome (AIDS). At this stage, the human immune system is unable to protect the body from infections, and this eventually leads to death. The HIV-1 infection of human immune system cells is caused by interactions of the virus proteins with the human host cell proteins. Predicting and analyzing protein-protein interactions (PPI) between HIV-1 and Human cells is thus a major goal for the development of safe and efficient strategies to combat AIDS. In particular, analyzing known interactions and identifying new interactions can provide important information to discover the reasons and mechanisms of this kind of viral disease and find new medical treatments [Arkin 2004, Brass 2008]. Most existing approaches applied in the interaction proteomics domain extract relationships within a single organism [Mukhopadhyay 2010] whereas FIST extracts bi-clusters and association rules showing relationships involving viral proteins, host proteins, or both at the same time. The aim of applying the FIST approach on these PPI data was to find out interactions between proteins

[1] http://trove4j.sourceforge.net/
[2] https://code.google.com/p/guava-libraries/
[3] http://javolution.org/
[4] http://commons.apache.org/collections/
[5] http://pcj.sourceforge.net/

and features and to extract relationships between annotations (biological and bibliographic) and interactions. Data used for this application were collected from different knowledge bases and merged to constitute several databases combining different types of information and data. The first database constructed integrated human and HIV-1 protein interaction information. The second database constructed contains human and HIV-1 protein interaction information and biological annotations from the Gene Ontology bio-ontology [GO 2000] and the Reactome knowledge base on pathways [Matthews 2008, Matthews 2007, Vastrik 2007, Joshi-Tope 2003] and bibliographic annotations from the PubMed biomedical bibliographic repository [PubMed]. The third database constructed contains human and HIV-1 protein interaction information and interaction types depicting the directed relationships between both organism proteins. Conceptual patterns extracted by FIST on this database showed show that the approach efficiently identifies interactions previously predicted in the literature and can be used to predict new interactions based on previous biological knowledge.

1.5 Layout of the Report

This report is organized in 4 parts as follows.

Part I presents a state of the art on topics related to this work. The first chapter in this part focuses on biological applications of pattern mining in the "omics" fields, i.e., pre-genomics, genomics and proteomics. The second chapter presents a study on itemset pattern extraction frameworks, that are divided into three categories: "classical" itemset frameworks, regular frameworks and evolutionary computation frameworks. A section in this chapter also presents interestingness measures used in pattern mining to guide the search and select the most relevant patterns.

Part II focuses on the algorithmic and data structure development works achieved during the thesis. The first chapter in this part presents the evolutions of the FIST approach, with the two algorithmic versions, data structures and their major implementation features, chronologically. Examples comparing the two versions and the main properties of the FIST approach are presented in an extensive way in the second chapter of this part.

Part III presents experimental data and results, the conclusion, and perspectives of future works. Experiments comparing the two algorithmic versions of FIST and other state of the art pattern mining algorithms are presented in the first chapter of this part. These experiments were conducted on genomics and proteomics datasets, that are the main domains of interest in this work. The three new datasets constructed from different HIV-1 and human protein interaction data and knowledge bases are presented in the first section of this chapter. These datasets integrate different representations of protein-protein interactions and, biological and literature annotations of proteins. Performance and applicability evaluations of the algorithms are presented in the second section. These experiments were conducted on two different systems, with different processing power and memory capacities, to assess efficiency and applicability of the algorithms in different computational contexts. Studies of extracted conceptual knowledge pattern sets are presented in the third section. A discussion of biological results and the potential of background knowledge integration to improve the relevance of the results and generate user-friendly knowledge patterns is also presented in this section. The second chapter in this part presents the conclusion and some perspectives of future works.

Part IV contains annexes and bibliographical references cited in the thesis report.

Part I

State of the Art

Chapter 2

Frequent Pattern Mining from Biological Data

Contents

2.1 Introduction

Association rule mining has multifarious applications ranging from market basket data analysis, customer relationship management, web mining, hardware and network intrusion detection, finance, telecommunication to biology. Hereabout, we are mainly interested in applications to biology, or more explicitly in the fields of computational biology, computational genomics, molecular biology, medical and biological information processing, pharmaceutical and disease related research, biotechnology and bioinformatics.

Association rule applications in these fields aim at characterizing relationships between sets of genes, proteins, or other cell members and the participation of different biological processes to health and diseases of cells. Such information are important for the analysis of diseases and

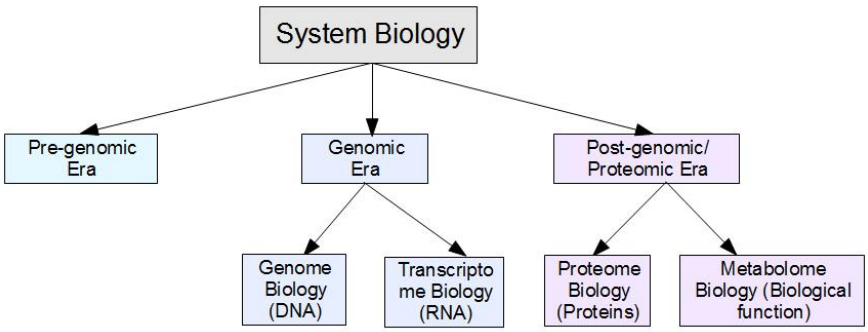


Figure 2.1: Categories of Applications in Biology

to help the finding of cures, by the development of new drugs and adapted medical treatments for instance. This area of biological applications can be divided into three epochs [Kell 2004, Mondal 2013]:

Pre-genomic era: This involves the highest levels of application in biology, mainly in medical and biological information processing and pharmaceutical and disease related researches. The main focus of this study is in characteristic changes in the healthy and diseases patients in different perspectives.

Genomic era: Genome analysis involves the operations on gene or DNA, i.e., mainly computational genomics. This also involves the manipulation or functioning related research on functional RNA, micro-RNA, etc.

Post-genomic and proteomic era: This last era includes the protein components, intermediate biological factor between gene/DNA and protein like different types of Acid, residues, and other bio-cells related system components in cellular biology. It also includes the metabolomic reactions in cell.

The most prominent association rule mining applications in biology are presented according to this classification in the following sections. Figure 2.1, shows a hierarchical classification of the different areas of data mining applications in biology.

2.2 Applications in Pre-genomics

Pre-genomic researches mainly involve the analysis of medical and environmental behaviour in the living body. Analysis of behavioural characteristics and socio-demographic data on healthy and unhealthy patients are important for the analysis of the different diseases, reactions of miscellaneous drugs, prediction of risk factors, etc [Brisson 2004, Nahar 2008]. Some applications in this context are performed under specific drugs or environmental conditions. Instead of using behavioural changes in the cellular components, like genes or proteins, these background knowledge information, that show relationships between diseases and the circumstances, are highly important for deciding particular disease, finding disease cures, reducing treatment cost for high-cost diseases or producing good and improved varieties of categories etc.

Each of these applications have their own particular database for scrutinization. These heterogeneous databases are of different type and contains a verity of information according to the type of research work will done on it. These data usually comprises of vast range of information about a large number of characteristics, such as historical background of the patient, addiction of the patient in any individuals, behavioural changes of different parts of the body for instance. Few example of such kind of databases are:

- Administrative Database
- Pharmaceutical Database
- Clinical Laboratory Database
- Health Care Database
- Historical Medical Database
- Biomedical Concept Database
- Cultivation Survey Database

General approaches for medical data mining are normally mined with classification, regression or clustering data mining task. A very small number of scattered research analysis was done with the association rule mining. Here our main aim to focus on this issues for mining medical data with association rules mining and also rule based classifier (Associative Classifier). Association rule mining in such research analyses includes different medical, pharmaceutical and behavioural datasets mentioned above. A classical and evolutionary approach for medical data investigation was studied in [Kwasnicka 2005]. A small discussion on formal concept analysis in medical data mining domain was presented in [Gupta 2005]. The study in this paper mainly shows the way of rule mining in classification rule analysis which is very much different from the classical classification task. The paper indicates some redundant medical tests which can be avoidable by doing some cheaper medical diagnosis and can reduce the cost of medical treatments. Another work in [Berardi 2005] shows the association relationship between different biomedical terms by analysing multi-level association rules or generalised association rules. The dataset used here are generated by using PUBMED queries in the biomedical sector. A recent work on similar kind of problem was addressed in [Hu 2010] to find the unknown information among biomedical motifs using semantic-based association rules.

These application fields of association rule mining can be distinguished in many ways the pre-genomics domain of research. In this report, these pre-genomic research fields are classified into four main groups: Adverse Drug Reaction Analysis, Disease Prediction Problem, Health Care Analysis, Cultivation Analysis. Figure 2.2, shows a hierarchical classification of the different areas of data mining applications in pre-genomics according to these four groups.

2.2.1 Health Care Analysis.

Association rule mining in health care industries come in the form of regulatory management of different rules and regulations, disease management for finding the best treatment, health insurance for finding the better scheme for different peoples. In [Semenova 2001], they showed a new way for generating rules and finding the relationships by using the classical association

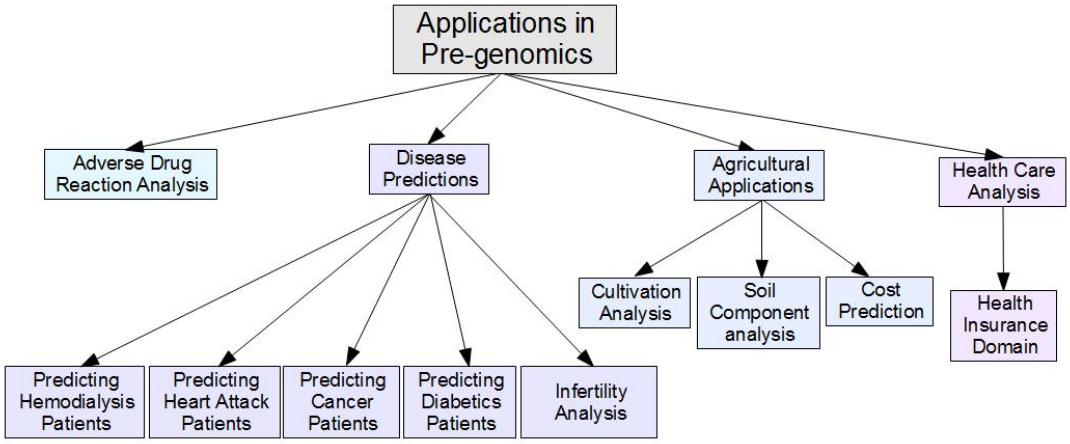


Figure 2.2: Categories of Pattern Mining Applications in Pre-genomics

rule mining by using a newly proposed "polydict" method. Their new approach "polydict" avoid the candidate generation of the rule mining problem and used a hash-table database "Pathol". Another work for analysing the insurance policies using association rules was proposed in [Chae 2001]. The study showed the way to predict the outcomes and policy information in health and used a database from the Korea Medical Insurance Corporation. They compare their method's outcomes with logistic regression and decision tree algorithm, two other approaches in health care analysis. Their analysis showed that the association rules are also an effective process as others for the health care analyzation. Positive and negative association rule generation and creating associative rule based classifier for the prediction analysis in health care was presented in two very recent article [Ramaraj 2008] and [Soni 2010], respectively. The paper [Ramaraj 2008] was proposed a new positive and negative rule generation algorithm and run on a medical survey database for diabetes and hypertension prediction analysis. In [Soni 2010], authors proposed a association rule based prediction analysis using classifier called associative classifier (AC). They have used the medical data consisting of personal information, medical test results etc to predict the probability of occurring a certain disease which may help for both the insurance company and also to the patients of a probable risk of certain disease.

2.2.2 Adverse Drug Reactions Analysis.

Adverse drug reactions (ADRs) are the reactions happened on body after consuming some medicines. It shows the correlation between unfortunate affects caused by any drug and the drugs used for the cure of diseases. Mining drug reactions dataset containing this kind of information helps to find the probable risk factor for the patients and to help the doctors for understanding the drugs diagnosis. In a study done in [Jin 2006, Jin 2008], the authors proposed an association rule mining method to solve this problem by generating unexpected temporal association rules from the administrative health database. They predict that atorvastatin may be suggested to tackle the side effects comes during the use of nizatidine or dicloxacillin for the cure of stomach ulcer or urinary tract infection. The study used the Queensland Linked Data Set from CSIRO and also predict some of the known adverse drug reactions which were already

discovered to show it's accountability. A recent work on ADRs was presented in [Ji 2011] where the authors used information coming from different datasets like administrative health, clinical or pharmaceutical data. The experiment results some "potential casual association rules" which shows the potential and casual relationship between drugs and the ADR symptoms.

2.2.3 Disease Prediction.

A work on association rule mining for medical diagnosis is presented in [Gamberger 1999] Several studies on medical data interpretation for disease diagnosis and prediction of disease by association rule mining are presented in [Doddi 2001, Ordonez 2000, Tremeaux 2006]. An overview of association rule mining particularly on the disease diagnosis in various areas of biomedical research was presented in [Rajak 2008].

2.2.3.1 Heart Attack Patients.

A major area in medical data analysis is prediction of heart diseases in patients having different heart problems. A work done in [Ordonez 2001] for mining constrained association rules from a database contained numeric, categorical, time and image information and finding important knowledge about coronary heart disease was done. At first they showed how to map medical data into binary form and find the low minimum support and high minimum confidence constrained rules. These rules predicts the presence or absence of a heart disease and find the correlation between them with the probable patients. Other automatic diagnosis of the heart disease using association rule mining for different characteristics of the healthy and unhealthy patient was studied in [Ordonez 2000, Ordonez 2006a, Ordonez 2006b, Patil 2009]. A specific solution recently proposed by using the association rule mining for the problem of predicting heart attack patients was presented in [Deepika 2011]. The study uses the Pima Indian Heart Attack Data Warehouse and done some pre-processing to discretize the continuous valued attributes. Then using their association rule generation system, generates the classification association rules and identify the class label and predict different heart attack systems. Few recent works on this particular problem of heart attack prediction in data mining was done in [Naidu 2012].

2.2.3.2 Cancer Patients.

Cancer is one of the major challenges in the health industry due to a major number of death caused by it. It comes in a several forms in the human body like bladder cancer, breast cancer, cervical cancer, lung cancer, prostate cancer, skin cancer etc. Insufficient research finding can't control the death of human due to cancer. Finding the causes and probable prevention factor for cancer disease is the beginning to find the cure of the disease. A study on extracting prevention factors those are most expressive for a specific cancer was recommended in [Nahar 2008, Nahar 2011]. The first study uses the association rule for this kind of problem which uses a prevention factor dataset containing related information about different types of cancer. The experiments shows the importance of this kind of approach for prevention factor extraction using rule generation. An extensive literature survey on this topic did not find any further work in this area. This is an important open problem and need much research focus by the researchers in the domain of predicting different factors for cancer disease.

2.2.3.3 Azoospermia Patients.

Infertility is a major problem in the third world country due to different causes like hypertension, long work schedule etc. A special group in male infertility called azoospermia where sperm is completely absent in the ejaculation. A successful data mining approach can classify the azoospermic patients and build some association between different characteristics similar to the clinical experimental analysis. A data mining work on the classification of azoospermia patients was done in [Mikos 2005]. Study shows this kind of data mining task may produce some clinically significant knowledge from a large set of clinical data for azoospermia patients which may reduce the cost and time of the experiments to find the disease and it's cure.

2.2.3.4 Diabetic Patients.

Another major risk in the modern world in health issues is the increasing number of diabetic patients. Analysis of diabetes using association rule mining of the clinical data of different diabetic patients was studied in [Stilou 2001]. The study uses the classical association rule mining algorithm and shows the usefulness of this approach in managing this disease and relationships among different parameters of diabetic patients.

2.2.3.5 Hemodialysis Patients.

Prediction of hemodialysis patients for the probability of the risk of hospitalization is an important problem in end stage renal disease (ESRD). The cost for the hemodialysis treatment is much higher and it includes a lot of measures to take a decision about the patient. Data mining approaches made it easy to find the relationship between these measures and the survival of the patients. A study in [Kusiak 2005] shows different mining strategy for this particular problem. Another recent work using multiple minimum support association rule mining was proposed to predict the hospitalization of hemodialysis patients in [Yeh 2011].

2.2.4 Agricultural Applications.

In addition to the above problems, an interesting place of research in pre-genomic era is the study of different applications in agricultural field. A work on developing state-of-art approaches for the utilisation in agricultural field using different data mining algorithms was presented in [Cunningham 1999]. This kind of research work helps the farmer to use the more developed, new and modern seeds and reduce the cost of cultivation. Particularly in the developing country, the use of this kind of mining on different survey data on soil or the use of other ingredient help to gain more dipper knowledge on the improvement of their way of cultivation and reducing the cost and use of insecticides. A work of association rule mining on finding the relation between different soil properties was done in [Calero 2003] A particular study on olive grove cultivation on the record collect from the province of Granada in the Mediterranean coastal border was done in [Delgado 2009]. The study shows how to improve the production of olive in the Mediterranean region along with the other geographical areas.

These are few studies done using different association rule mining framework. Although a lot more studies are needed to have an enhance understanding on each topic. Many more areas are still untouched in this domain for the research using rule mining to have a better insight and improving the prediction, production, analysis in a better way on this domain.

2.3 Applications in Genomics

Investigation of genome includes a part of DNA (Deoxyribonucleic Acid) string being able to coded known as gene and also the non-coding part of DNA along with some viral RNA. Application of genomic covers perhaps all genes in an organism. These genes contains the formula for the protein whose application in association rule mining has been discussed in the next subsection of this section. Genome analysis is important to understanding the behavior of gene under different biological reactions and aspects. There are many high-throughput techniques are available for generating information of the gene profile which used to analyze the function of gene for different kind of applications.

Micro-array is gene profiling technique that generates huge amounts of information called gene expression data. These data have been intensively used and studied to investigate and address issues related to genes. Gene expression data contain expression profiles of several genes under different biological conditions. These expression profiles are generated as continuous numerical values but, depending on applications and end user requirements, they are usually discretized into three classes: over expressed, highly repressed and unchanged. Sometimes, this gene expression data also come-up with certain information based on the applications to correctly analyze the gene behavior. Genomic databases can have several forms according to the information they contain:

- Temporal Gene Expression Data (T-GED).
- Spatial Gene Expression Data (S-GED).
- Gene Expression Data with Cellular Environmental Descriptions and Ontological and Bibliographic Annotations.
- Gene Expression Data with Transcription Factors.
- Gene Mutation Data.
- Gene Sequence Data.

Gene mutation data is a peculiar type of gene related data used to identify the cause of gene tumors and diseases [Krawczak 2000, Lewis 2000]. There are several application areas under this genomic topic research. They can be classified into four principal categories: Gene expression analysis, gene regulatory analysis, functional genomic analysis and row-wise gene group analysis. Figure 2.3 shows a hierarchical classification of the different areas of association rule mining applications in genome analyses.

2.3.1 Gene Expression Analysis.

Researches in gene expression analysis mainly include different kinds of micro-array gene expression data to understand the mechanism of cellular process under different conditions. Advances in technologies accelerate the growth of huge gene expression datasets that are difficult to manage and costly to process for finding interesting information. Association rule mining is an efficient approach that is well-fitted to the processing of such datasets.

Gene expression analysis can be divided into two parts according to the nature of the gene expression profiles analyzed. First part includes the analysis of gene expression to understand

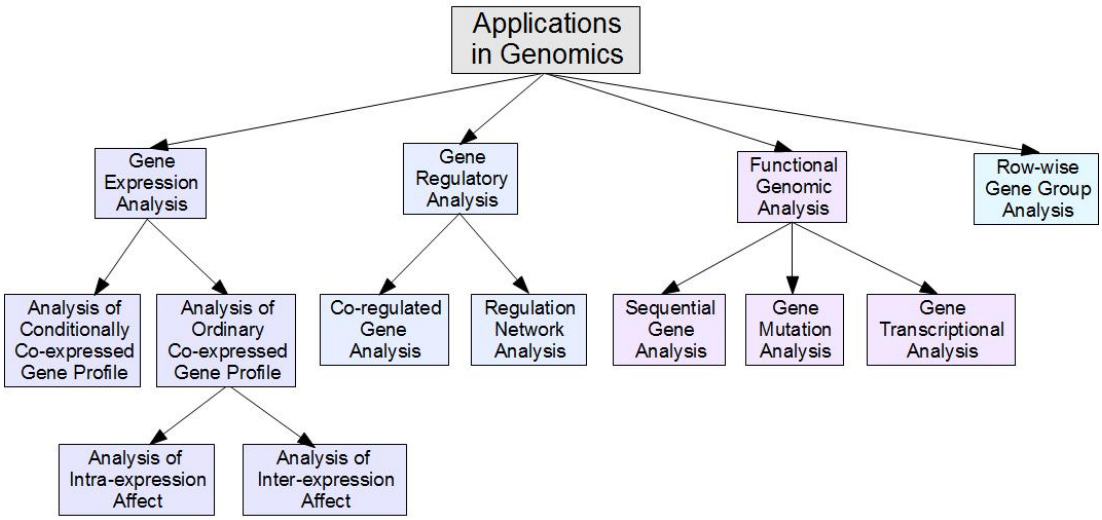


Figure 2.3: Categories of Pattern Mining Applications in Genomics

the times and conditions of expression of genes that are basically not expressed in healthy samples. This kind of study is known as conditionally expressed gene analysis. Second part includes analyzes the affect of one gene expression profile over another gene expression profile. In this case, two situations arises: either both genes are of the same group, for example belonging to the same gene regulation network, and it is then called intra-expression affect analysis, or genes belong to two different groups and it is then called inter-expression affect analysis. A conglomerating research in this field, using different kind of association rule analysis methods, is presented in [Gauthaman 2008].

2.3.2 Gene Regulatory Analysis.

Gene regulatory analysis involves co-regulated gene analysis where genes are regulated under at-least one common transcription factor [Allocco 2004, Yeung 2004]. This area of investigation also concerns gene regulatory network analysis which aim at understanding the underlying complex genetic regulatory processes [Shang 2009]. These researches are of importance as they concern several domains, such as pharmaceutical industry or analyzes of complex diseases for instance, that are essential for public health. Several studies pointed out that association rule mining is a particularly relevant approach in the field of gene regulatory analysis [Becquet 2002, Carmona-Saez 2006, Creighton 2003, Huang 2007, Tuzhilin 2002].

2.3.3 Functional Genomic Analysis.

Functional genomic analysis include the analysis of different function related matters inside a cell, such as sequential analysis of gene, gene mutation analysis or analyzing transcription factor for example. These analyses use SAGE (Serial Analysis of Gene Expression) data or gene expression data with different biological sample information. Sometimes, redundant information can be added to the gene sequence dataset originating from different sources generating DNA

sequencing information. These redundancies may be due to the absence of cross-referencing of identical sequential information in different databases, to the presence of multiple instances of the same sequence in a database, or to the fact that a sequence can be divided into several parts distributed over the database. Analysis of gene sequence using association rule mining makes it possible to remove automatically such redundancies in data [Y.Koh 2005].

Gene mutation analysis is another functional genomic problem that aims at identifying the causes of tumors in genes and the diseases resulting from this kind of abnormal mutations. Association rule mining was successfully applied to the gene mutation databases containing information about mutations, mutagens, diseases, etc. over each gene [Lewis 2000, Krawczak 2000]. Detailed analyses about the process, mechanism and controls for transcription factor can be found in [Cho 1998, White 2001]. The analysis of transcriptomic data in the context of association rule mining is discussed in [Karel 2007].

2.3.4 Row-wise Gene Group Analysis.

A quantitative analysis of relationships between co-regulation, co-expression and functional genome was conducted in [Allocco 2004]. Another study shows the relationships between co-expression and co-regulation of gene micro-array data [Yeung 2004]. Usually, gene expression data are processed in a column-wise manner for extracting association rules from the dataset and generate the required information for the end user. However, the huge number of attributes in this kind of data can cause important efficiency problems because of the large search space, that is proportional to the number of attribute values, and the large number of association rules generated. To solve this problem, finding association rule groups by analyzing the dataset in a row-wise manner is a quite recent trend in genomics [Cong 2004, Pan 2003, Xu 2004].

Association rule mining was successfully applied to analyze the relationships between genes in different expression levels and to find conjunctions between gene expressions and biological functions. Almost all types of association rule mining methods have been applied or adapted to discover knowledge from genome related data. Most of these methods are basically application-type dependent as different techniques are sometimes required for different kinds of applications or data. Some examples of application dependent works and the different kinds of rule generation methods adapted for the different data types are:

- Distance-based association rule mining [Icev 2003].
- Heterogeneous association rule mining [Anandhavalli 2010].
- Quantitative association rule mining [Georgii 2005, Karel 2007].
- Dynamic association rule mining [Gauthaman 2008].
- Ant-based association rule mining [He 2009].
- Multi-objective association rule mining [Han 2009].
- Fuzzy association rule mining [Lopez 2007].
- Partition-based association rule mining.

Recent detailed studies on these approaches can be found in [Anandhavilli 2010, Gauthaman 2008].

2.4 Applications in Proteomics

Proteins which serve as signals generated from a cell and act as a signaling components for transmitting information outside of the cell. It performs as a receptor to the signals coming to the cell and produce from the surrounding or environment of that cell. It also plays a major role in the DNA and RNA manipulation and determine the portions of gene are expressed and that mRNAs are translated into proteins. Proteomic is the field of studying protein components in a complete genome of an organism or in a specific tissue. Proteomic research constitutes of analysing the structures and functions of biological system by analysing the available and known proteins. These analysis include comparing protein expression profiles, correlate protein interaction data, interrogate the protein structures between healthy and disease cells. Moreover, identifying the unknown proteins, storing the protein information in different kind of databases, designing intelligent drugs are also covered in the study of the proteomics. Other applications in proteomic include the analysis of protein sequences, acid sequences and sometimes genomic related sequences [Marcotte 1999]. Studies in these fields help to understand the protein interactions in metabolic, synthesis of ATP (Adenosine Triphosphate), gene replication, quantify proteins or peptides etc.

Study of proteomic raised the question of it's need while several successful studies in genomic already presented to analyse different cell operations. After the genomic analysis, understanding the function of protein is a major challenge for the biologists [Tyers 2003]. Proteomics study taking over the research where genomics and transcriptomics stop explaining what, where and why different biological functions are happening inside the cells. It was first introduced in 1994 [Wilkins 1994] to understand and explain the unknown facts about the cells which was unsolved before the post-genomic era. Studies on correlations between gene expression information and protein interaction information are described in [Grigoriev 2001, Thorton 2001, Zhang 2004]. An investigation of the relationship between mRNA and protein are exemplified in [Anderson 1997, Gygi 1999].

Genetic and biochemical experimentation, such as two-hybrid screenings, generate huge amount of protein interaction data. Information about the different protein databases can be found in [Barker 2001, Boeckmann 2003] and a detailed analysis of this field of proteomics can be found in [Liebler 2002]. A few different types proteomic databases contain proteomic information and used for proteomic study are as:

- Interaction Data
- Structural Data
- Sequence Data
- Isoforms Data
- Modifications Data
- Localisation Data
- Protein Abundance Data
- Dynamic Protein Data
- Clinical profile data

- Variant Data
- Disease Data

One can find these above mentioned types of the proteomic databases in the sites: SWISS-PROT, TrEMBL, PIR-International, NCBInr, ESTdb, MIPS, DIP, BIND, GRID, dbEngine etc. Some of these sites are cross-referenced and maintain a common standard of nomenclature. A list of the sites and resources related to proteomic study available in the websites mentioned at the end of this section. These databases are generated from different proteomic technologies available in the field. Different technologies used in the proteomic study listed below are used to generates a verities of information to build these databases:

- Interaction proteomics
  - Mass Spectrometry
  - Two-hybrid
  - Array based proteomics
    - Antibody Array
    - Functional Protein Array
    - Peptide Array
    - Carbohydrate Array
    - Flurosence Array
    - Chemical Array
    - Small molecule Array
- Structural proteomics
- Amino Acid composition
- 2D Gel Electrophoresis
- Differential Proteomic
  - Differential Gel Electrophoresis
  - Multiplexed proteomics
  - Isotope-coded affinity tagging
  - Differential Gel Exposure

Study of proteomics has several difficulties which ranges from the sample being used for the experiments, technologies being used on the samples or due to the dynamic nature of the proteins in different environmental conditions. These limitations may include (i) the limited sample materials for the type of cells or tissues to be examined, (ii) samples are degraded and perturbed rapidly with the environmental conditions, (iii) post-translational modification and vast dynamic range of proteins make it more complicated to understand the actual state of the protein in a specific condition, (iv) limitations in new proteomic technologies like mass spectrometer, 2D gel electrophoresis etc which generates different protein data. Limitations in

the very recent analytical methods to analysis the biological information also made the proteomic analysis more difficult. Although some limitations for proteomic analysis have been reported [Garbis 2005, Lippolis 2010], proteomics and system biology applications help to understand system level functionality and to analyse the mechanisms behind the working principles [Giannopoulou 2008, Zhang 2009a].

Several application fields of association rule mining can be distinguished in the proteomic domain of research. These proteomic research fields can be classified into four main groups: Structural Proteomic Analysis, Functional Proteomic Analysis, Protein Expression or Differential Proteomic Analysis and Clinical Proteomic Analysis. Figure 2.4, shows a hierarchical classification of the different areas of data mining applications in proteomics according to these four groups.

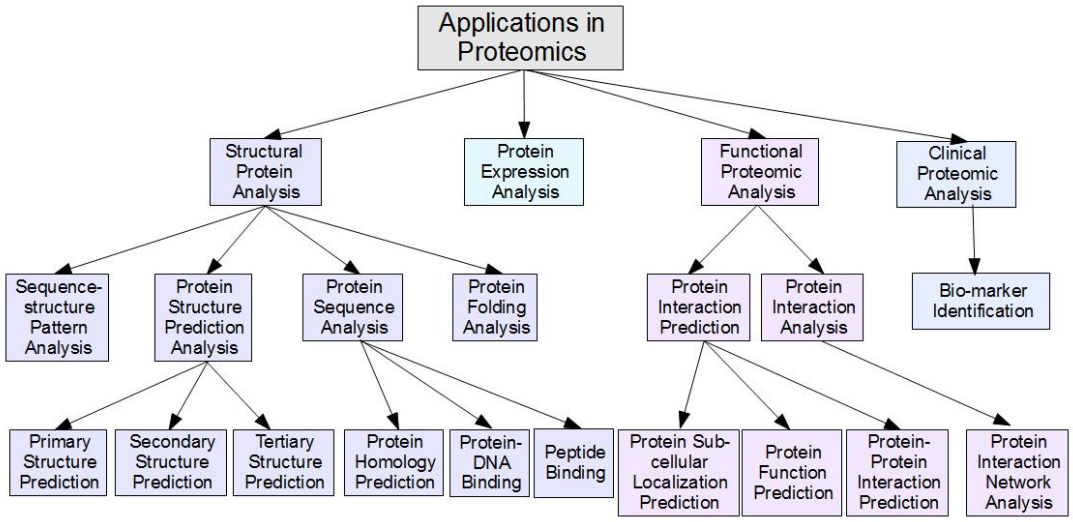


Figure 2.4: Categories of Pattern Mining Applications in Proteomics

The following paragraphs presents the most prominent applications in each of these areas using association rule mining approaches. Several interesting results have been produced using association rule analysis of proteomic data, but more works are required to understand relationships between the different components and activities of proteins. A wide area of proteomic research are still has many open problems in the data mining domain.

2.4.1 Structural Protein Analysis.

It has studied and considered that protein sequences are non-random patterns of amino acid molecules in nature. There are 20 different types of amino acids are available in three dimensional protein structure. These amino acid sequences are mainly responsible for different functions performed by proteins. To understand the multifarious biological operations it is indispensable to understand the protein functions. So as, it is essential to understand the sequence motif of a protein or the sequence of amino acids in protein. Due to the non-randomness of the sequences, finding associations and co-occurrences of amino acids using association rule mining

is efficient, as others data mining methods. Association rules show the relationships between different residues in the sequence for a particular protein function. Whereas, the co-occurrences predict the occurrences of a motif if some motifs are known to be present in the amino acid sequence. Because of these reasons, association rule mining shows it's importance in protein sequence analysis. Several protein sequencing methods generate a huge number of protein sequences which increases the relationship gap between sequences and structures. Predicting the protein structure and sequence analysis is one of the most important research activity in system biology [Thorton 2001, Yang 2011a].

An association rule based sequence searching and interpretation of sequence motif was presented in [Kam 2003]. They found some association on which basis Ryanodine and IP3 receptor proteins are could be separated. They have shown the usefulness of the association rule mining in this particular proteomic problem where they have used IBM intelligent Miner for Data as a data mining tool and InterPro database of protein motif to find new useful information. A quantitative association rule mining application to find the global correlations between amino acids in protein is presented in [Gupta 2006]. Their results discover the associations on the presence and absence of amino acids in the sequence which is different in case of motif based approach where only the presence of amino acids are considered. The study shows that the importance of absences of any residue to the protein structure and function. They have used the normalized frequencies of amino acid instead of present or absent of it in the protein. The normalisation is done by diving the frequency of an amino acid with the length of the protein. The quantitative association rules derived from this experiment uses these frequencies to find the association between the amino acids. The protein sequences are taken from the SCOP Astival File with proteins only less than 40% homologous to each other. Finding amino acid sequences in proteins has an important role in bioinformatics research in association rule mining [Zhou 2010]. Few recent works were published for protein sequence motif analysis using positional association rule mining in [Chen 2009, Chen 2010, Chen 2011]. All these experiments were done on the protein sequence database collected from the Protein Sequence Culling Server (PISCES) [Wang 2003]. They have introduced a new measure for association rule mining called "distance assurance" to analyses the distance between two motifs. The positional association rules generated from these experiments predict the associations between presence of a motif with respect to other motif which must satisfy one frequent distance measure in the protein sequence.

2.4.2 Protein Structure Prediction Analysis

Another problem in proteomic analysis is to understand the structure of the proteins based on the characteristics and functions performed by the protein components. Structural information is stored in the amino acids sequences that bind them and form the polypeptide chains in proteins. Similarly to the amino acid sequence interpretation, understanding the protein structure is important for pharmaceutical and disease researches. Predicting the protein structure using different data mining methods like clustering, classification, association rules gain much importance than the traditional or Ab initio methods in recent years.

Association rules and rule based classifiers proved their efficiency for understanding the similarities and differences in protein structures. This classification of the protein structure is very much important to manage the information and finding the relationship between the proteins. But manual classification of the structure is time consuming and difficult task. Few

works was proposed to classify the protein structure automatically using association rule mining in [Rattanakronkul 2002, El-Houby 2010]. In [Rattanakronkul 2002], they have collected different protein information like signature, functions and organisms from heterogeneous online protein data bank like PDB, PROSITE, CATH etc. The whole data have divided into training and test set and apply their hybrid method combined with data classification and association rule mining for structural class prediction problem. Their result shows the better accuracy to predict different protein classes compared to the classical methods for protein class prediction. The recent work published in [El-Houby 2010] is a memory and time efficient algorithm called protein mining algorithm (PMA) is based on the associative classifier. The classifier predicts the class of a protein based of the protein features presented on the database. It needs only one scan on the protein data to generate the class association rules. In [Zadzic 2006], they try to find the sub-structured patterns from the Human Prions Proteins information stored in the prions database, a rooted ordered labelled sub-trees database. The experiment finds the frequently occurring subtrees from the prions database and interpreted the results using the previously found patterns. Frequent sub-trees generated in the form of rules from this experiments predict some association relationship between the sub-structures of the proteins. Study done in [Zaki 2000] tried to predict the structure of protein molecules by using small local structure profiles or motif of different proteins stored in a library of sequence structure motif database. It is basically a combination of hidden markov model and association rule mining method. At the beginning they have used the hidden markov model to reduce the parameters to the adjacencies of different motif structures and apply the association rule mining to predict the local structure of the proteins. Out of different structures of the protein, problem of predicting secondary structures of protein are most important for it importance in biological activity. The first association analysis method was proposed for this particular predicting problem in [Yang 2009]. They have build the compound pyramid model where their proposed KAAPRO method is acts as kernel of the model. The association rules generated from this method can be interpreted and shows the involvement of the physical-chemical properties to the secondary structures. In [Zhou 2010], a same compound pyramid model was build using a structural association algorithm for predicting the secondary structure of a protein accurately and precisely. In this experiments, an association rule based classifier was used to analyze secondary structure sequences for predicting protein secondary structure that is the three-dimensional form of local segments. The same compound pyramid model was used in [Yang 2011b] with support vector machine. In [Stelle 2011], an association rule based approach was presented to this protein structure prediction problem. This method works on two different steps, first to collect the data from different sources and arrange them. In the second step, the secondary structure formation rules are generated using the classical apriori association rule algorithm. The study shows some important relationship between α-helixes and β-sheets with class hydrophobicity profiles. These profiles are the building block of any proteins and can have the possibility to act on motif. This also played an important role in tertiary structure prediction that corresponds to specific atomic positions in three-dimensional space. Protein structure prediction, such as primary and secondary structure prediction, are sequence studies for which association rule mining and association rules based classification has proved their best. However, very few studies have conducted in this field of research.

2.4.3 Protein Sequence Analysis

An important branch of proteomics is the prediction of homologies between protein sequences in biological information processing. Protein homology prediction is useful for grouping proteins on the basis of functional and structural similarities. In this context, the main objective of protein homology prediction is to predict the value of protein sequences. Few works were done in this field using association rule mining for homology prediction problem [Yin 2003]. In [Tang 2005], a study was conducted by combining an association rule based classifier with support vector machines. For the experiments, they have used the widely used KDDCUP04 homology prediction dataset. The study showed that the predictions are much improved than the previously predicted protein homologies found by other methods and this mechanism gives us a good utility based classification method for complex problem.

Based on structural and functional properties, the protein-DNA bindings are classified into eight different groups, namely, Helix-Turn-Helix Proteins, Zinc-coordinating Proteins, Zipper-Type Proteins, Other α-helix Proteins, β-Sheet Proteins, β-hairpin/ribbon Proteins, Other DNA-binding Proteins and Enzyme. Binding between protein and DNA plays an important role in transcription, replication, packaging, mutation, etc. [Luscombe 2000, Luscombe 2002]. To know the fundamentals of the binding processes between proteins and DNAs, it is highly important to understand the nature of the complexes between these two units. Experimenting on the structure of the complexes and classifying them is the most significant operation to comprehend the DNA-binding proteins. Association rule mining and association rule based classifier is one of the most important methods were shown to be very useful for this kind of operations. The discovery of protein DNA-binding sequence models as the binding uses the association information between amino acids and nucleotide sequence pairs was studied recently in [Leung 2010]. In this article, they have focused on the binding between transcription factors (TFs) and transcription factor binding sites (TFBSs) using the classical Apriori algorithm and evaluated on the basis of quantitative, empirical, annotation and experimental analysis. The experimentally found results were validated with the information provided in Protein Data Bank (PDB), by homology modelling and by a random analysis. A comparative study conducted in the field of predicting protein-DNA interaction is presented in [Zhou 2008]. No further work has been found after an extensive search in this binding prediction analysis with the help of association rule mining or based on associative classifier. So, we can say that it's a very new and interesting open problem and still needs much work to solve this protein-DNA binding prediction problem.

Major Histocompatibility Complex (MHC) molecules are basically a wide range of genes responsible for the immune system. In human, it calls as Human Leukocyte Antigen (HLA) and most widely studied HLA are (i) MHC Class I of HLA-A, HLA-B, HLA-C and MHC Class II of HLA-DPA1, HLA-DPB1, HLA-DQA1, HLA-DQB1, HLA-DRA, HLA-DRB1. Class I MHC molecules are responsible for the cellular immune response when antigen enters into the cell and Class II MHC molecules are responsible for the humoral immune response [Kloetzel 2004]. Analysing MHC-Peptide binding has a great impact on the reaction of immune system and the amino acid sequence motif in a peptide give aids to the MHC-Peptide binding. Prediction of the antigenic peptides or T-cell epitopes of MHC-peptide binding is one of the major study field in peptide binding domain. T-cell epitopes are the binding between MHC and peptides those are acknowledged by the T-cells. These peptide binding information has a vital role playing against designing vaccine. An encyclopedic database for MHC-Peptide binding information is

available in MHCBN [Bhasin 2003].

A novel association rule mining method for predicting T-cell epitopes of MHC Class I HLA-A*0201 was first introduced in [Milledge 2004]. The authors uses a combination of association rule mining and sequence-structure pattern method for the epitopes prediction. The apriori algorithm first generates the primary association rules and then apply the sequence-structure pattern method to filter and generates sequence structure patterns (SSPs). A combinatorial method for predicting the structure and sequence of the peptide binding was presented in [Zeng 2001] They have tried to predict the characteristics of the type of interaction between T-cell epitopes bound by HLA-A*0201 in MHC-peptide binding. Here each rule is an association between the residue and position of the peptide. Then by applying sequence pattern method, they try to find the group of residues that occur concurrently and thereafter by applying the structural analysis on all the known HLA-A*0201. Then using a regression model, they validate their generated outputs against some profile based methods for epitope prediction. Another work was done in [Ozbek 2005] by using a position specific association rule mining on apriori method by integrating the position information in the original algorithm. By adding the positional information, they try to find the sequential patterns of the peptide sequence and predict the peptide binding with HLA-A*0201. The position information is more important in this prediction task because position of the amino acids within the peptide have a great impact on MHC-Peptide binding. Another recent and probably last work on peptide binding prediction problem using association rule mining was proposed in the article [Yardimci 2006]. In the article, authors are mainly focused on the "antigenic fragments" of the Class I MHCs. They combine the structural methods and association rule mining techniques to predict the T-cell epitopes. Association rules find the correspondence between peptide binding positions for deciding the MHC binding motifs of each type. They mainly did their experiments on the database of positively binding peptides whereas they have ignored the non-binding peptide information. To perform the accuracy test and validating the generated rules, they have used the four fold cross validation methods.

2.4.4 Protein Expression Analysis.

The area of protein expression analysis or mapping between different proteins expression levels are known as differential proteomics. This field of study differentiates the distinct and related proteomes for identifying bio-markers of different biological states and diagnose the regulation functions for proteins. Extensive surveys of analyses in the field of differential proteomics are presented in [Monteoliva 2004, Zhang 2009a]. Data mining techniques present interesting perspectives to get relevant information from the massive databases related to differential proteomics. In this field of research, data mining approaches are used for creating efficient databases for query processing, preliminary data analysis, experimental design etc. [Zhang 2009a].

2.4.5 Functional Proteomic Analysis.

Functional proteomic comprises the analysis of protein interactions and the prediction of interactions between different proteins [Oyama 2002]. The prediction part involves protein function prediction, protein-protein interaction prediction, prediction of protein sub-cellular localisation etc. This field also includes the determination of protein interactions in a proteome for a specific

protein to ascertain the unknown functions. Protein sub-cellar localisation prediction refers to the problem of predicting the location of a functional protein within a living cell. A machine learning classifier based approach was proposed in [Lu 2004] and an association rule mining based classifier system was proposed in [Liu 2011] for predicting sub-cellular localisation of proteins.

Understanding the functions of proteins are important to characterise their role in diseases and to discover efficient treatments for these diseases. The unknown functional activities of proteins are predicted either from protein-protein interaction data [Deng 2003], from protein interaction networks [Pandey 2007], via the analysis of interactomes [Nabieva 2008], or by analyzing genome sequences [Marcotte 1999]. The use of association rule mining in the context of protein function prediction is studied in [Pandey 2007].

Protein-protein interaction network discovery play a major role in the construction of biological pathways, protein networks and drug discovery in particular [Segal 2003]. In the molecular biology field of research, this interaction prediction application is currently the major focus for data mining applications. Association rule mining methods were successfully applied for discovering and predicting interactions by using association memberships of known and unknown proteins [Chiu 2008, Mondal 2012a]. Other recent research results on protein-protein interaction prediction based on association rule mining and related associative classification can be found in [Kotlyar 2006, Mukhopadhyay 2010, Nafar 2006, Oyama 2002]

The analysis of protein interaction networks contributes to the understanding of protein annotation studies [Schwikowski 2000]. Association rule mining methods play a significant role to discover and identify protein annotation relationships within or between protein-protein interaction networks [Oyama 2002]. In [Besemann 2004], the authors show that differential association rule mining is an efficient solution for the analysis of protein annotations.

2.4.6 Clinical Proteomic Analysis.

Another new research application area in proteomic is the clinical proteomic analysis. Identifying the type and stage of cancer patient at an early age is an important and necessary condition for the guaranteed recovery of the patient from that disease. Studies show that identifying the cancer at an early stage have the greater chances to be cure. The absence of the suitable biomarkers for any cancer disease is an important open problem in the Clinical Proteomic Analysis. Many biomarkers have already been found but the verification of those markers and finding new biomarkers are a real challenge in clinical proteomic [Kohn 2007]. Study of these biomarkers are useful for the management of the disease and finding the right therapy and drug for the patients. Clinical proteomic is the discipline of the proteomic analysis to the molecular medicines and clinical observations [Fung 2005]. Study of biomarkers is one of the important and major task in clinical proteomic. The biomarkers together form a signature called biosignature which is validated by the clinical proteomic study [Rifai 2006]. Different technologies and strategies in the proteomic are used for the validation of the biosignature of any disease [Petricoin 2003, Colantonio 2005]. Some of these technologies along with their application in clinical proteomic was studied in [Latterich 2008, Street 2010].

These technologies reproduce a huge amount of data and it requires an efficient computational method to analyse them and to find useful information. Mass Spectrometer (MS), Two dimensional sodium dodecyl sulphate polyacrylamide gel electrophoresis (2D-PAGE), High Performance Liquid Chromatography (HPLC), Two Dimensional Liquid Chromatography (2D-

LC) are few of those technologies which are used to generate different profiling data of the proteins. Serum profile is such kind of data generated from the mass spectrometer and used for the biomarker discovery. Identifying the serum biomarkers for disease identification is a good choice for identifying cancers in early stage. A study on identification of serum biomarker using a combination of feature selection method and association rule mining technique on the mass spectral serum profile is given in [Ressom 2006].

Another research on the clinical sample for identifying the disease specific biomarker identification from the data generated by the mass spectrometer of clinical proteomic was presented in [Gopalakrishnan 2006]. The study successfully found the Amyotrophic Lateral Sclerosis (ALS) panel biomarkers by using the wrapper based association rule learning method for the motor neuron disease. The data samples are generated from the cerebrospinal fluid using the surface-enhanced laser desorption/ionisation time of light (SELDI-TOF-MS) analysis.

A rule based biomarker identification method for classifying metabolic profiles in prostate cancer by mass spectrometer was discussed in [Osl 2008]. Their proposed method is basically a associative classification rule based feature selection technique called associative voting which was used for the prostate cancer management. A 10 fold-cross validation technique for the better evaluation of the result biomarkers used for the metabolic candidates in cancer patients. A recent article have published for the identification of the breast cancer biomarker using fuzzy association rule mining technique in [Lopez 2012]. It's a new and promising field of research in clinical proteomic and in fact in proteomic and very few works have found in the literature with the association rule mining for the biomarker identification.

Chapter 3

Study of Itemset Pattern Frameworks

Contents

3.1 Introduction

Research works in the association rule mining domain can be divided into two main segments: First, selecting the most relevant sets of items and second, generating the most pertinent association rules, either directly or indirectly, from these selected sets of items. Works in the first segment focus on the generation of different categories of sets of items, such as frequent itemsets, closed itemsets, free itemsets, regular itemsets, etc. with different structural properties in regard to the dataset. In the second segment, works focus on association rule generation either indirectly, by using the itemsets previously extracted from the dataset, or directly from the dataset, for example, when they are based on some kind of evolutionary approaches. In both segments, some measuring units are used to calculate the importance of these itemsets or rules generated from the dataset. These measuring units are known as interestingness measures

and are calculated on the statistical appearances of the items on the database present in the itemsets or rules.

This chapter describes the above mentioned issues of different frameworks, interestingness measures for association rule mining. At first, we will present a brief description of different statistical measures used for association rule mining. Later sections demonstrate heterogeneous types of rule mining frameworks for finding multifarious number of itemsets and association rules. According to the notions mentioned above, the frameworks for association rule mining can be classified into different categories:

- Based on the theoretical framework of the algorithms.
- Based on the search space traversal of the algorithms.
- Based on the data structure used by the algorithms.
- Based on the databases dealing by the algorithms.
- Based on the applications used for the algorithms.

In this chapter, we will describe few frameworks based of different itemsets from the above mentioned categories available nowadays. The most prominent of these frameworks: Frequent itemset (or classical) framework, closed itemset framework, free set framework, regular itemset framework and evolutionary computation framework.

3.2 Dataset Representations

Initially, association rule mining algorithms used an horizontal representation of the dataset, also called row-mode. In this representation, each data line represents an object and contains a list of items materialised as an enumeration set, as in table 3.2, or a bit vector, as in table 3.1. This representation was used by the first level-wise algorithms proposed for mining frequent itemsets, maximal frequent itemsets and frequent closed itemsets. The vertical representation of the dataset, as given in table 3.3, also called column-mode, was proposed in [Zaki 1997]. In this representation, each item is associated to the list of objects containing it. These lists can be materialised as enumeration sets, as in the Clique and Eclat algorithms [Zaki 1997] or as bit vectors, as in the HBM algorithm [Gardarin 1998]. This data representation allows to efficiently compute the support of an itemset by intersecting enumeration sets or vectors representing its items instead of counting the number of objects containing the itemset with the horizontal layout. However, it increases the number of dataset accesses required and complex data structures are needed to moderate this problem. The use of the vertical representation for frequent closed itemset discovery was introduced with the Charm algorithm [Zaki 2002]. A vertical representation by diffsets was proposed for frequent itemset discovery by [Gouda 2001]. A diffset represents the difference between the list of objects containing a k-itemset and the list of objects containing one of its (k+1)-supersets. Diffsets can significantly reduce the memory space required to store the lists of objects for vertical representations of data in the case of dense datasets. An approach combining vertical and horizontal representations was proposed by [Shenoy 2000] for frequent itemset discovery.

Data representation has an important impact on the efficiency of any algorithm for extracting association rules. When data are coming from several sources, all approaches require to

combine them into a unique matrix for processing. Horizontal transactional and binary matrices are the most widely used data representations used by different association rule mining frameworks. Table 3.1 and table 3.2 give the representations in binary matrix and horizontal transactional format, respectively, of an example dataset D. This dataset will be used as a supporting example throughout the chapter for describing different concepts and frameworks. It contains five objects or rows, identified by their *O-id* or transaction number ranging from 1 to 5, and each of these object contains a list of at most six items namely A, B, C, D, E and F. Throughout this chapter, we also use 0.4 (40%) for minimum support and confidence threshold values unless explicitly stated otherwise.

O-id	**A**	**B**	**C**	**D**	**E**	**F**
1	1	1	1	1	0	0
2	0	1	0	1	0	0
3	1	0	1	0	1	0
4	0	1	1	1	0	0
5	1	1	1	1	0	1

Table 3.1: Binary Matrix Representation of Database D

Transaction	**Items**
1	A B C D
2	B D
3	A C E
4	B C D
5	A B C D F

Table 3.2: Horizontal Transactional Representation of Database D

Transaction	**Items**
A	1 3 5
B	1 2 4 5
C	1 3 4 5
D	1 2 4 5
E	3
F	5

Table 3.3: Vertical Transactional Representation of Database D

3.3 Interestingness Measures

Several interestingness measures were proposed to assess the usefulness of an association rule or a group of association rules. These measures aim at filtering extracted association rules to minimize the size of the rule set and help the end user in the deduction of information for decision making. In Section 3.3.1, different types of interestingness measures for single association rules and their properties, along with the criteria used to define them, are studied. In Section 3.3.2, an overview of interestingness measures for assessing interestingness of groups of association rules are given.

A hierarchical representation of the different categories of interestingness measures is shown in Figure 3.1. In this figure, the diagram shows the classification of the different interestingness measures of association rules according to their domain of application (individual or group of rules), the type of measures (objective, subjective or semantic) and their structural properties (symmetric or asymmetric).

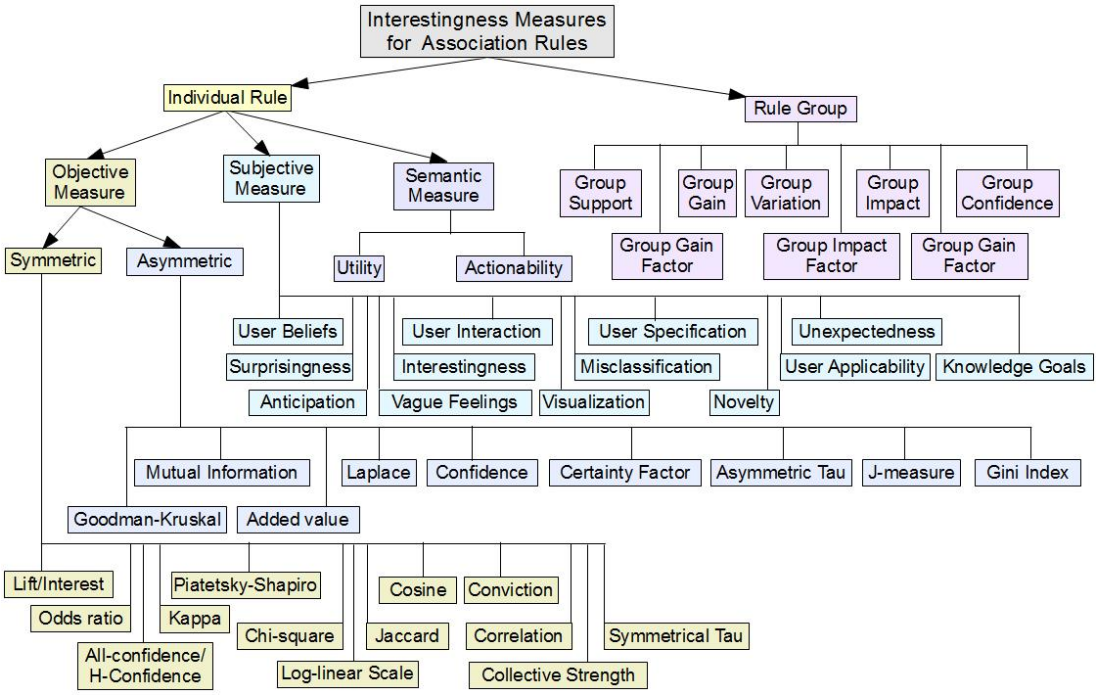


Figure 3.1: Categories of Interestingness Measures for Association Rule Mining

These measures follows several criteria for deciding whether a pattern, i.e. a rule or a group of rules, is really interesting. In [Geng 2006], the following nine such criteria, that are used in most works on this topic, are studied in details:

Conciseness: Conciseness describes how compact one pattern is.

Generality: Generality is defined in terms of maximal coverage of the items in the dataset.

Reliability: Characterizes whether a pattern is reliable in terms of occurrences in the application areas.

Peculiarity: Depicts how a pattern is different from the others in the generated output.

Diversity: Defined in terms of differences of the item characteristics in an itemset and between the itemsets in the extracted patterns.

Surprisingness: A pattern is said to be surprising if it is out of the end user's expectedness.

Novelty: Defines to what extent the patterns mined are novel to the end user's knowledge.

Utility: Defines the usefulness of a pattern.

Applicability: Characterizes whether extracted patterns are appropriated to the application or not.

3.3.1 Interestingness Measures for Individual Association Rules

These interestingness measures are used for evaluating the importance of association rules, on their own, in the dataset. These measures help to reduce the size of the list of extracted rules by selecting the most interesting ones. A recent overview of such interestingness measures can be found in [McGarry 2005]. These measures can be divided into three categories: Objectives, subjectives and semantical.

3.3.1.1 Objective Measures

Measures in this category evaluate the statistical significance of the rule in the dataset. Several properties were defined to assess the relevance of these interestingness measures. Detailed reviews of these properties can be found in [Geng 2006, Tan 2002]. Values of objective measures are calculated under the contingency table, and they can be evaluated according to different properties:

- Symmetry property: This property refers to the symmetry of measure values under variable permutation. A measure of association rule interestingness which value is not modified when the antecedent and the consequent are interchanged is called a symmetric measure. Other measures, whose value changes when interchanging antecedent and consequent, are called asymmetric. A recent review on symmetric interestingness measures is presented in [Merceron 2007].

- Null-addition property: This property refers to whether the value of the measure remains constant when the number of objects not covering the rule is increased in the dataset. A measure with this property is called null-invariant. Cosine and Jaccard measures are null-invariant but odds ratio, correlation coefficient and interest factor are not for instance.

- Row/column scaling property: This property refers to the constant of the measure value after rescaling the contingency table with some positive constant. Odd ratio is one example of this property.

- Inversion property: This property, that is a special case of row/column scaling property, refers to whether the measure value remains constant after exchanging the frequency counts in the contingency table. Inversion property include the correlation coefficient, odds ratio, collective strength and kappa symmetric measures and, interest, cosine, Piatetsky-Shapiro and Jaccard asymmetric measures.

- Piatetsky-Shapiro's properties: Piatetsky and Shapiro proposed three properties for qualifying relevant measures [Piatetsky-Shapiro 1991a]. These properties are:

  i. The value of the measure is 0 when antecedent and consequent parts are statistically independent.

  ii. The value of the measure monotonically increases with the support of the rule when the individual support of both antecedent and consequent are unchanged.

  iii. The value of the measure monotonically decreases with the support of the antecedent (resp. the consequent) when the supports of the rule and of the consequent (resp. the antecedent) remain constant.

  Added value, two-way support, and Piatetsky-Shapiro's measures are examples of measures following these three properties. An extended version of these properties is described in [Kamber 1996].

- Lenca *et al.*'s properties: Eight properties for classifying interestingness measures were proposed in [Lenca 2004]. Among these eight properties, five are normative properties and three are subjective properties. These five normative properties are:

  i. Easiness to define the threshold values.

  ii. Ability of the measures to identify asymmetricity of the rule.

  iii. Constance of the measure values in the absence of counter examples.

  iv. Uniformity of measure values in the case of independent attributes.

  v. Decrease of the measure values with the increase in the number of records covering only the consequent of the rule.

  The three subjective properties are based on the multi-criteria decision analysis performed by the analyst. They assess measure interestingness according to the end user's viewpoint and depend on the following criteria:

  vi. Increase of the measure values when the size of the database is increased.

  vii. Semantics of the measures are easily expressed and are thus relevant.

  viii. Without significant loss of measure values to tolerate some counter-examples that do not notably lessen rule interestingness.

  These properties were applied to classify a list of 20 measures in [Lenca 2004].

- Gang and Hamilton's properties: Gang and Hamilton proposed in [Geng 2006] some properties of interestingness measures to determine if support and confidence are increasing functions when contingency tables are definitive or fixed to their margins.

A critical review of criteria used for the evaluation and classification of objective interestingness measures is given in [Suzuki 2008].

A list of objective measures following these properties is given, with corresponding references, in Table 3.4. Recent overviews on objective measures of association rule interestingness can be found in [Steinbach 2007, Tan 2004]. Methods for choosing the adequate measures were also developed to further help the end user [Lenca 2008].

3.3.1.2 Subjective Measures

End user's view plays a crucial role during the evaluation and the interpretation of the extracted association rules. The end users are most often application domain experts having their own views and beliefs that influence their analysis and choice of information. For example, trivial rules or rules depicting well-known relationships are generally uninteresting whereas rules confirming some hypothetical relationships are very relevant from the end user's viewpoint.

Incorporating subjective criteria, such as end user's beliefs and hypotheses or surprisingness and novelty of the rule for example, to assess the interestingness of extracted association rules has been the subject of many research works. Figure 3.1 shows 13 subjective interestingness measures used in many association rule mining applications. Extensive recent reviews of these subjective interestingness measures can be found in [Geng 2006, Natarajan 2005].

Comparative evaluation of objective and subjective measures of interestingness for association rules are presented in [Lenca 2007, McGarry 2005]. A comparison of the strengths and weaknesses of subjective and objective measures, throughout the analysis of twelve association rule interestingness measures, is presented in [Zhang 2009b]. A review and practical evaluation of association rule interestingness measures application in the medical domain is given in [Ohsaki 2004]. These reports conclude that the combination of both types of measures can optimize interestingness evaluation.

3.3.1.3 Semantical Measures

These interestingness measures aim at describing the importance of semantic behavior and informativeness of patterns. They are for most application-context dependent. Among such measures, the utility and the actionability criteria play an important role.

Utility refers to practical measures of rule usefulness in that they can reflect the actual amount of output achieved by applying each rule. In initial works on this topic, a particular rule can have a different utility value depending on how well the rule fits the purpose of the application it was extracted for. Nevertheless, several recent works were conducted to define a uniform standard for assessing rule interestingness disregarding application context [Yao 2006b, Yu 2008]. Several algorithms incorporating utility measures were proposed, some being dedicated to computational biology applications. For instance, in [Laxmi 2011, Sandhu 2011], weightage (W-Gain), utility (U-Gain) and diminution (D-sum) measures are introduced in the association rule mining process to generate a score assessing utility of each rule. Reviews of association rule utility measures can be found in [Anandhavalli 2010, Yao 2006a].

Actionability of discovered knowledge patterns aims at evaluating how meaningful are these patterns to support decision-making actions. A rule is said to be *actionable* if the end user can utilize it to make a decision or gain some advantage [Rezende 2009]. Actionability is a major

Sl.	Measure	Reference	Sl.	Measure	Reference
1	Accuracy	[Spruit 2007]	33	J-Measure	[Smyth 1991]
2	Added Value	[Sahar 1999]	34	Jaccard	[Tan 2002]
3	All-Confidence/H-Confidence	[Omiecinski 2003, Xion 2006]	35	Kappa	[Cohen 1960]
4	Any-confidence	[Omiecinski 2003]	36	Klosgen	[Klosgen 1996]
5	Asymmetric Tau	[Zhou 1991]	37	Laplace	[Clark 1991]
6	Bayes Factor	[Jefreys 1935]	38	Leverage	[Piatetsky-Shapiro 1991a]
7	Bond	[Omiecinski 2003]	39	Least Contradiction	[Aze 2002]
8	Brin's Conviction	[Brin 1997a]	40	Lift/interest	[Brin 1997b]
9	Centered Confidence	[Bras 2010]	41	Log-Linear Analysis	[Agresti 1996]
10	Certainty Factor	[Shortliffe 1975, Klosgen 1996]	42	Loevinger	[Loevinger 1947]
11	Chi-square	[Brin 1998]	43	Mutual Information	[Lallich 2007]
12	Confidence/Precision	[Agrawal 1993, Pagallo 1990]	44	Normalize Mutual Information	[Lallich 2007]
13	Collective Strength	[Aggarwal 1998]	45	Odds Ratio	[Mosteller 1968]
14	Conviction	[Brin 1997b]	46	Odd Multiplier	[Lallich 2007]
15	Correlation Coefficient	[Agresti 1990]	47	Piatetsky-Shapiro	[Piatetsky-Shapiro 1991a]
16	Cosine/IS Measure	[Tan 2000]	48	Prevalence	[Lallich 2007]
17	Coverage/True positive rate	[Spruit 2007]	49	Probabilistic discriminant index	[Lerman 2003]
18	Conditional Influence	[Chen 2001]	50	Recall	[Yao 1995]
19	Confidence Gain	[Tamir 2006]	51	Relative Risk	[Lallich 2007]
20	Confidence Factor/Predictive Accuracy	[Ghosh 2004]	52	Sebag-Schoenauer	[Sebag 1988]
21	Comprehensibility	[Ghosh 2004]	53	Specificity	[Lallich 2007]
22	Completeness	[Spruit 2007]	54	Strength	[Fhar 1993, Klosgen 1996]
23	Entropy Intensity of Implication	[Gras 2001]	55	Support	[Agrawal 1993]
24	Example & Counter example rate	[Lallich 2007]	56	Symmetrical Tau	[Zhou 1991]
25	Ganascia	[Ganascia 1991]	57	Weighting Dependency	[Gray 1998]
26	Generality	[Yao 1999]	58	Yao's one way Support	[Yao 1999]
27	Gini Index	[Breiman 1984]	59	Yao's two way Support	[Yao 1999]
28	Goodman-Kruskal	[Goodman 1968]	60	Yao's two way Support variance	[Yao 1999]
29	Implication Index	[Lerman 1981]	61	Yule's Q	[Yule 1900]
30	Implication Intensity	[Gras 1979]	62	Yule's Y	[Yule 1912]
31	Influence	[Chen 2001]	63	Zhang	[Zhang 2000]
32	Interestingness	[Spruit 2007]			

Table 3.4: Interestingness Measures for Individual Association Rules

feature for association rules in most applications since actionable rules are the most useful to achieve the application objectives [Liu 2000]. A two-way significance framework for evaluating actionability, based on both technical interestingness and domain-specific expectations, was proposed in [Cao 2007]. A description and formalization of actionability based knowledge discovery according to four types of generic frameworks, namely post-analysis based, unified interestingness based, combined mining based and multi-source combined mining based, is proposed in [Cao 2010]. Actionability of association rules can also be characterized using the closed itemset and the δ-free sets [Boulicaut 2008]. Detailed reviews of actionability measures and techniques, and descriptions of trends in actionable knowledge discovery, are available in [Cao 2012, He 2005]. An application perspective on actionable knowledge discovery and its trends from this viewpoint can be found in [Cao 2006, Li 2010].

A detailed description of utility and actionability based interestingness measures is available in [Geng 2006].

3.3.2 Interestingness Measures for Association Rule Groups

An association rule group is a set of rules defined on a certain group of itemsets for grouping rules based on common attributes in the antecedent part [Jimenez 2010]. Let I be a frequent itemset containing n items $I_1, I_2, \cdots, I_n$ and G be a group of itemsets on I such that $\forall G_i \in G$, $G_i \subseteq I$. The group of association rules defined on group G is denoted as $GAR = \{\bigcup R : I_1 \rightarrow I_2 \mid I_1 \in G\}$. Grouping association rules in such a way allows both to identify all rules concerning a particular group of items and to reduce the number of rules concerning a group of itemsets that have a similar behavior in the dataset.

Few measures have already been proposed to assess the interestingness of such kind of group rules and the adaptation of standard individual interestingness measures to association rule groups is studied in [Jimenez 2010]. Table 3.5 summarizes some available measures for evaluating the interestingness of group association rules.

Sl.	Measure	Notation	Definition
1	Group Support (I)	$supp_G(I)$	$= \frac{P(G\|I)}{P(G)} = conf(G \rightarrow I)$
2	Group Support$(I_1 \rightarrow I_2)$	$supp_G(I_1 \rightarrow I_2)$	$= \frac{P(G\|I_1 \cup I_2)}{P(G)} = conf(G \rightarrow I_1 I_2)$
3	Group Confidence	$conf_G(I_1 \rightarrow I_2)$	$= \frac{supp_G(I_1 \rightarrow I_2)}{supp_G(I_1)} = conf(G\ I_1 \rightarrow I_2)$
4	Group Gain	$gain_G(I_1 \rightarrow I_2)$	$= conf_G(I_1 \rightarrow I_2)$ - $supp_G(I_2)$ $= conf(GI_1 \rightarrow I_2)$ - $conf(G \rightarrow I_2)$
5	Group Gain Factor	$GGF_G(I_1 \rightarrow I_2)$	$= \frac{gain_G(I_1 \rightarrow I_2)}{1 - supp_G(I_2)}$ if $gain_G(I_1 \rightarrow I_2) \geq 0$ $= \frac{gain_G(I_1 \rightarrow I_2)}{supp_G(I_2)}$ if $gain_G(I_1 \rightarrow I_2) < 0$
6	Group Variation	$variation_G(I_1 \rightarrow I_2)$	$= \frac{gain_G(I_1 \rightarrow I_2)}{supp_G(I_2)}$ $= \frac{conf_G(I_1 \rightarrow I_2) - supp_G(I_2)}{supp_G(I_2)}$
7	Group Impact	$impact_G(I_1 \rightarrow I_2)$	$= supp(GI_1) * gain_G(I_1 \rightarrow I_2)$
8	Group Impact Ratio	$GIR_G(I_1 \rightarrow I_2)$	$= \frac{impact_G(I_1 \rightarrow I_2)}{supp(G)}$

Table 3.5: Interestingness Measures for Association Rule Groups

The support of frequent itemsets and valid association rules is measured with respect to

the total number of objects present in the dataset. For these measures, the support of an itemset in a group or an association rule group is measured with respect to the dataset objects involved in that particular group. Similarly, all other measures in Table 3.5 are calculated by considering only the dataset objects containing the group itemsets. Computing group measures can be less time consuming than classical measures as its need to scan only those transactions containing the group itemsets instead of the whole dataset. However, because of differences between the definitions of support and confidence of itemsets and rules, and the group support and confidence more calculations can be required. For example, some standard measures require to calculate only one probability where two probabilities must be calculated in the case of group measures, such as for the gain measure:

$$\begin{aligned} gain(I_1 \rightarrow I_2) &= conf(I_1 \rightarrow I_2) - supp(I_2) \\ &= \frac{P(I_1 \cup I_2)}{P(I_1)} - P(I_2) \end{aligned}$$

$$\begin{aligned} gain_G(I_1 \rightarrow I_2) &= conf_G(I_1 \rightarrow I_2) - supp_G(I_2) \\ &= \frac{P(G \cup I_1 | I_2)}{P(G \cup I_1)} - \frac{P(G \cup I_1)}{P(G)} \end{aligned}$$

This can make a big difference for measures that require the calculation of a large number of support and confidence values. However, the concept of association rule groups presents several advantages:

- The number of items in rules can be reduced, to make easier their interpretation, by regrouping the common items in their association rule groups.
- The end user's can more easily apprehend information from these rule groups, as they define a subspace of the solution space made up of rules with identical properties, and handle huge sets of rules.
- Group measures can rank the groups according to the end user's interest and determine the interestingness of a group of rules instead of each rule independently of others.

Although, quite few measures for association rule groups have been proposed actually, almost all types of standard association rule measures can be adapted to the concept of groups of association rules.

3.4 Frequent Itemset Framework

The first theoretical framework for pattern mining, defined in [Agrawal 1993] together with the association rule mining problem, is the *frequent itemset framework*. In this work, the frequent, or large, itemsets were defined as itemsets with support greater than or equal to the user defined *minsup* threshold value in the dataset. It was shown that all supersets of an *infrequent itemset*, i.e., an itemset with support lower than *minsup*, are infrequent. Association rule mining problem is based on the discovery of association relationship between a set of items. The valid, or strong, association rules were also defined along with their support, that is the support of the itemset resulting from the union of the antecedent and the consequent of the rule, and their confidence, that is the ratio between the support of the consequent and the support of the union of the antecedent and the consequent of the rule. It was also shown that all valid association rules, i.e., association rules with support and confidence greater than or

equal to the user defined *minsup* and *minconf* threshold values respectively, can be generated from the frequent itemsets and their support. Considering the properties mentioned above, the association rule mining problem can be divided into two successive phases:

- Mining frequent itemsets, with their support, in the dataset.
- Generating valid association rules, with their support and confidence, from frequent itemsets and their support.

Once frequent itemsets and their support are extracted, all valid association rules can be generated from them in a straightforward manner. Generating these rules is a much less computationally expensive task compared to the frequent itemset mining process that requires accessing the dataset. Optimizing the frequent itemset mining process has thus been the subject of intensive research works for nearly two decades and many algorithms and theories have been proposed for this in the literature.

The difficulties of the frequent itemset extraction lies in two points: First, it requires several costly scans of the dataset that is usually stored in secondary memory; Second, the size of the search space is exponential in the number of items in the dataset. If the dataset contains n items, then the size of the search space is 2^n and the frequent itemset mining problem was shown to be NP-Complete [Angiulli 2001]. This search space constitutes the subset lattice, or *itemset lattice*. The itemset lattice for database D that contains six items is given in Figure 3.2. We can see that, for a minimum support threshold of 40%, there are 15 frequent itemsets and one maximal frequent itemset, all other itemsets being infrequent. Since all supersets of infrequent itemsets are infrequent, the maximal frequent itemsets, defined according to the inclusion relation, form a border above which all itemsets are infrequent and below which all itemsets are frequent.

Several algorithms for traversing the itemset lattice have been proposed in the literature to optimize this computationally expensive task. The state-of-the-art Apriori algorithm [Agrawal 1994] introduced the principle of *level-wise traversal* to improve the efficiency of frequent itemset extraction. Many algorithms derived from Apriori have then been proposed since. A detail discussion on different approaches for itemset lattice traversal has been given in the next subsection. However, frequent itemset based approaches suffer from the problem of the huge number of frequent itemsets and association rules generated in case of dense datasets [Brin 1997b]. Overviews of algorithms and descriptions of trends in frequent itemset and association rule mining can be found in [Ceglar 2006, Pramod 2010, Zhang 2010].

3.4.1 Search Space Traversals for Itemset Mining

The level-wise breadth-first itemset lattice traversal was introduced with the Apriori algorithm by [Agrawal 1994] and independently by [Mannila 1996]. In this approach, all items of a "level" in the search space lattice are considered at once: A set of candidate k-itemsets, that are potentially frequent sets of k items, is constructed and supports are computed from the dataset. Frequent k-itemsets are then combined to construct candidate (k+1)-itemsets. This approach was proposed to limit the number of accesses to the disk resident dataset and reduce processing time. Apriori is a level-wise bottom-up algorithm: levels of the itemset lattice are considered in increasing size order, starting from 1-itemsets. Additionally, numerous works have been conducted to optimize algorithm's efficiency using adapted data representations, data structures and implementation techniques [Ceglar 2006]. The level-wise bottom-up approach was

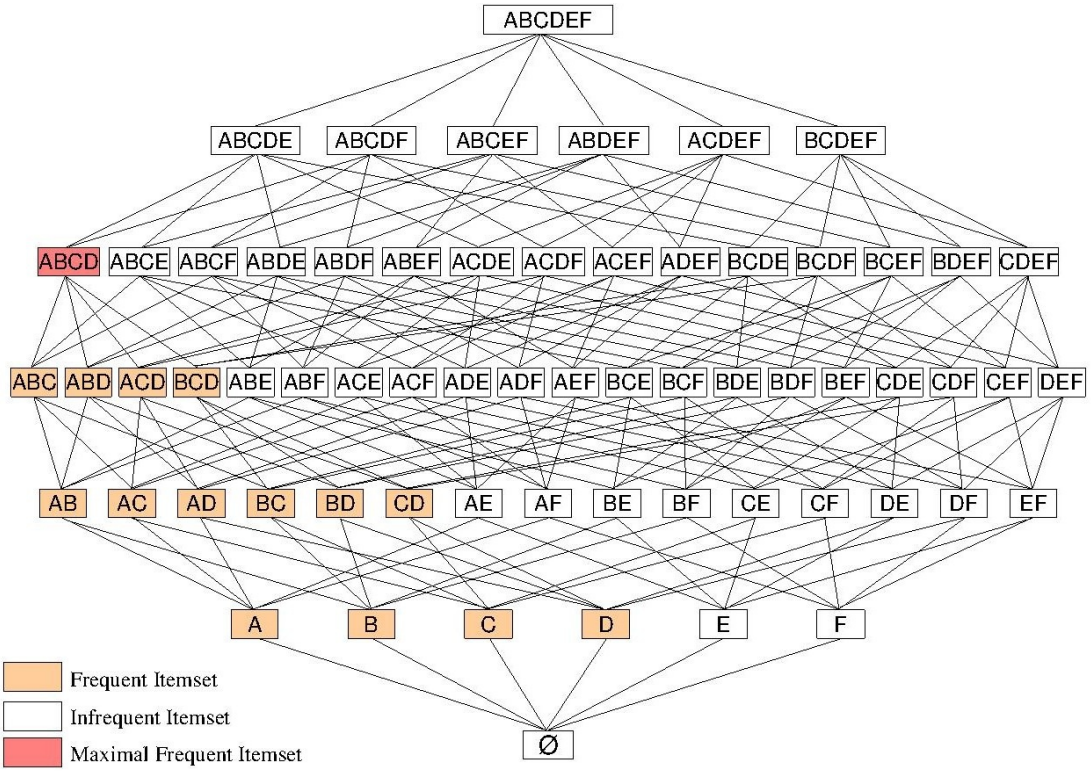


Figure 3.2: Itemset Lattice with Infrequent, Frequent and Maximal Frequent Itemsets (*min-support* = 40%)

also applied for discovering frequent closed itemsets in the Close [Pasquier 1999b] and A-Close [Pasquier 1999a] algorithms, and in several derived algorithms. See [Yahia 2006] for a review of these algorithms. The simultaneous bottom-up and top-down level-wise approach was introduced in MaxMiner [Bayardo 1998] and simultaneously in PincerSearch [Lin 1998]. In this approach, levels of the itemset lattice are considered both in increasing size order starting from 1-itemsets and in decreasing size order starting from M-itemsets, where M is the maximal size of an itemset.

The depth-first traversal of the itemset lattice [Zaki 1997, Agarwal 1999] identifies frequent itemsets in lexicographic order. Following this order, itemsets in the example dataset D are considered in the following order: [a, ab, abc, abcd, abcde, abd, abde, abe,..., d, de, e]. This approach is well suited to frequent itemset mining when data are represented in a vertical layout, that is a data line represents an item and is constituted of a list of objects containing this item (TID-list). This is equivalent to accessing the example dataset D in table 1 by columns instead of rows. Supports of itemsets are then computed by intersecting TID-lists of their subsets. See [Hipp 2000] for a comparison of breadth-first and depth-first approaches for frequent itemset discovery. A depth-first approach is used in the well-known FP-Growth algorithm [Han 2000b] and its extensions. This approach is known as pattern growth and does not require to use candidate itemsets contrarily to level-wise approaches. In this approach, frequent 1-itemsets

are successively extended with other items until their maximal (longest) frequent supersets are obtained. A top-down pattern growth approach was proposed for frequent itemset discovery in the TD-FP-Growth algorithm [Wang 2002]. The pattern growth approach was adapted for frequent closed itemset discovery in the Closet algorithm [Pei 2002] and several derived algorithms were proposed since.

3.5 Closed Itemset Framework

The *closed itemset framework* was introduced to address the problem of association rule mining in dense datasets [Pasquier 1998]. Closed itemsets are defined according to the closure operator γ of the Galois connection used in Formal Concept Analysis and concept lattice theories [Birkhoff 1995b, Ganter 2005]. The Galois closure of an itemset is equal to the intersection of all objects containing it and an itemset I is closed if it is equal to its closure, i.e., if $I = \gamma(I)$. Each closed itemset defines a membership class, or *equivalence class*, that includes all itemsets contained in the same dataset objects, and consequently have the same support value, as the closed itemset [Bastide 2000]. By definition, the unique maximal itemset of an equivalence class is a closed itemset. An itemset that is both frequent, according to the *minsup* threshold, and closed is called a *frequent closed itemset*. The frequent closed itemsets and their equivalence class in the itemset lattice for dataset D and for a minimum support threshold of 40% are shown in Figure 3.3.

Since all frequent itemsets and their support can be deduced from the set of frequent closed itemsets and their support, this set constitutes a lossless condensed representation of all frequent itemsets. Using this property, the search space of frequent itemset mining can be reduced to the frequent closed itemsets and the association rule mining problem can be divided into the two following successive phases:

- Mining frequent closed itemsets, with their support, in the dataset.
- Generating valid association rules, with their support and confidence, from frequent closed itemsets and their support.

The search space of frequent closed itemsets based algorithms is a sub-order of the itemset lattice. The reduction of this search space depends on the proportion of frequent itemsets that are closed since once a frequent closed itemset is identified, all frequent itemsets in its equivalence class can be deduced. In the case of dense datasets, equivalence classes are large as few frequent itemsets are closed and the first phase execution times are notably improved compared to frequent itemset mining. However, this approach does not improve association rule mining for sparse datasets, such as market basket data sets. For such datasets, equivalence classes are small, as most frequent itemsets are closed, and closure computations increase execution times compared to frequent itemset mining.

To improve the efficiency of this approach, the minimal itemsets of an equivalence class, called *generators*, can be identified in a level-wise manner and used to generate the frequent closed itemsets [Pasquier 1999b]. This subsequently reduces the computational cost as they are the smallest itemsets in an equivalence class. The lattice structure of the sub-order defined by the frequent closed itemsets for example database D and $minsup = 40\%$ is depicted in Figure 3.4.

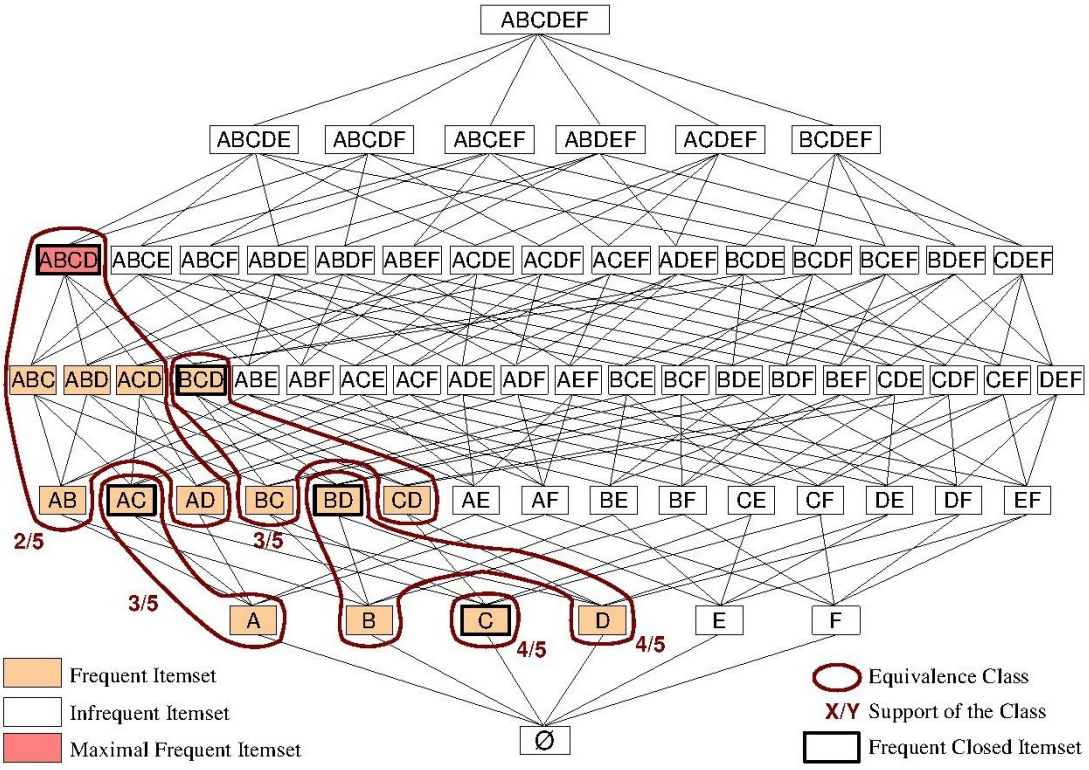


Figure 3.3: Itemset Lattice with Frequent Closed Itemsets and Equivalence Classes (*minsupport* = 40%)

This lattice contains five equivalence classes corresponding to the five frequent closed itemsets and their seven generators. For the second phase, condensed representations for association rules can be generated from frequent closed itemsets and generators which further reduces the processing time of association rule generation. The use of generators in the definition of condensed representations is important for association rules generation. Recent surveys on frequent closed itemset based mining of associating rules can be found in [Shekofteh 2010, Yahia 2006].

3.6 Free Set Framework

Frequent itemset and closed itemset frameworks are sensitive to noise in the data that is common in some applications and also difficult to mine for the dense dataset. The *free set framework* was proposed to overcome this problem in [Boulicaut 2000, Boulicaut 2003]. Free sets are the concise representations of the entire frequent sets. This framework is based on the ε-adequate condensed representation presented in [Mannila 1996] and the proposed algorithm is an illustration of the work on levelwise search and border theory presented in [Mannila 1997]. In this representation, data are used to answer the queries with ε lost of accuracy. Free sets constitute an approximate representation of the frequent itemsets as an approximation of the

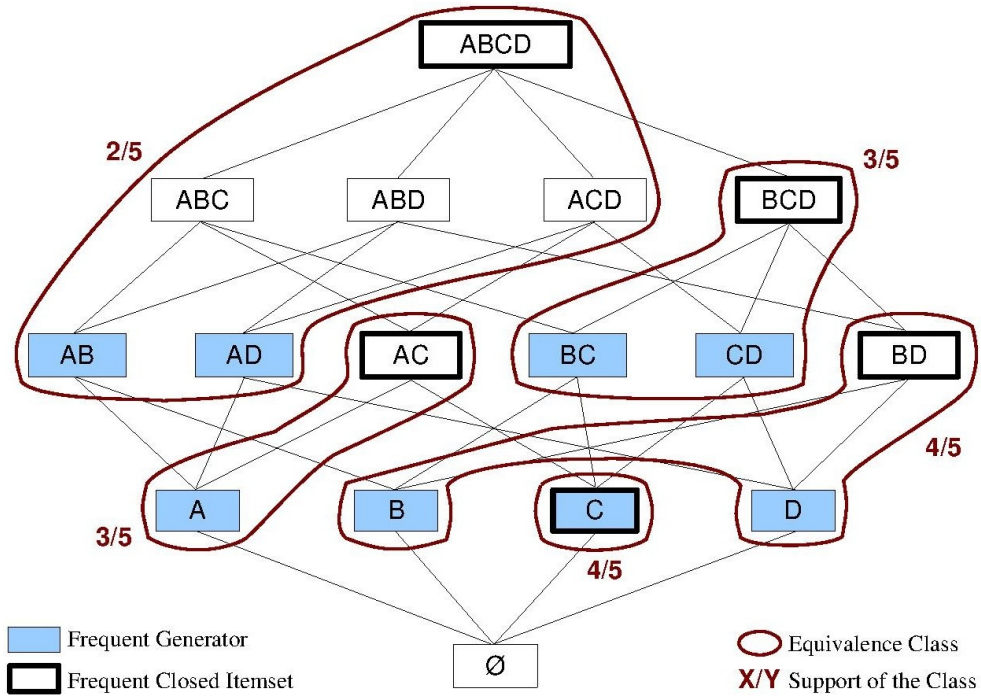


Figure 3.4: Sub-order of the Itemset Lattice with Frequent Closed Itemsets and their Generators ($minsupport = 40\%$)

frequent itemset supports can be deduced from their supports [Boulicaut 2003]. This kind of probabilistic proposition for frequent itemset mining was studied in [Pavlov 2000].

An ε-adequate representation of a class of structure (S) with respect to a class of queries (Q) is an another representation (C) of the class of structure represented by a mapping function $f_s : S \rightarrow C$ and a query evaluation function $f_q : Q \times C \rightarrow [0, 1]$, such that $\forall$ structure $\in$ S and $\forall$ queries $\in$ Q, $\|Q(structure) - f_q(Q, f_s(structure))\| \leq \varepsilon$. ε-adequate condensed representation of a database is much smaller in size than the original size of the data. The support of any free sets can be used to approximate the support of an itemset found in the condensed representation of the database. The support of a set is approximated to 0, if any subset of a set is free but not frequent, otherwise the support of the set is considered as the smallest support among the frequent free subsets of the set. Error in approximating the support and confidence value for frequent itemsets and association rules using frequent free sets abide low in practice [Boulicaut 2003]. A level-wise frequent free set mining algorithm was proposed in [Boulicaut 2003] with an additional search space reduction for the case where $\delta \neq 0$.

Free sets are also referred to as δ-free sets defined according to δ-strong rules. An itemset I is a δ-free set if and only if, there is no δ-strong rule $I_1 \rightarrow I_2$ with I_1 and $I_2 \subset I$ and $I_1 \notin I_2$. An association rule of the form $I_1 \rightarrow I_2$ with $I_1, I_2 \in I$ is a δ-strong if $support(I_1)$ - $support(I)$ $\leq \delta$. The value of δ is inversely proportional to the confidence of the rule, i.e., minimizing δ is equivalent to maximizing the confidence of rules. δ-strong rules for $\delta = 0$ are exact association

rules, i.e., rules with confidence equals to 1, and mining δ-free sets for $\delta = 0$ is equivalent to mining closed itemsets but mining δ-free sets where $\delta \neq 0$ proposed much reduction on search space with the lost of uncertainty in support threshold. Frequent δ-free sets with respect to an user defined threshold support value are the free sets which has no δ-strong rules and the support of the itemset is greater than the support threshold value. Finding the complete set of δ-free sets of a structure is usually not feasible. This problem can be handled by the negative border of frequent δ-free sets where the negative border composed by the smallest itemsets which are non-frequent δ-free sets with respect to set inclusion.

An important property of free sets with respect to inclusion of itemsets is anti-monotonicity which says, all subsets of a free set are also free. The minimal elements of an equivalence class, i.e., the generators of the class, are free sets [Calders 2005, Ruggieri 2010]. The frequent δ-free sets in dataset D for $\delta > 0$ and a minimum support threshold of 40% are shown in Figure 3.5. A constrained based frequent free set mining algorithm was proposed in [Boulicaut 2001]. A recent review on free sets based condensed representations can be found in [Calders 2005].

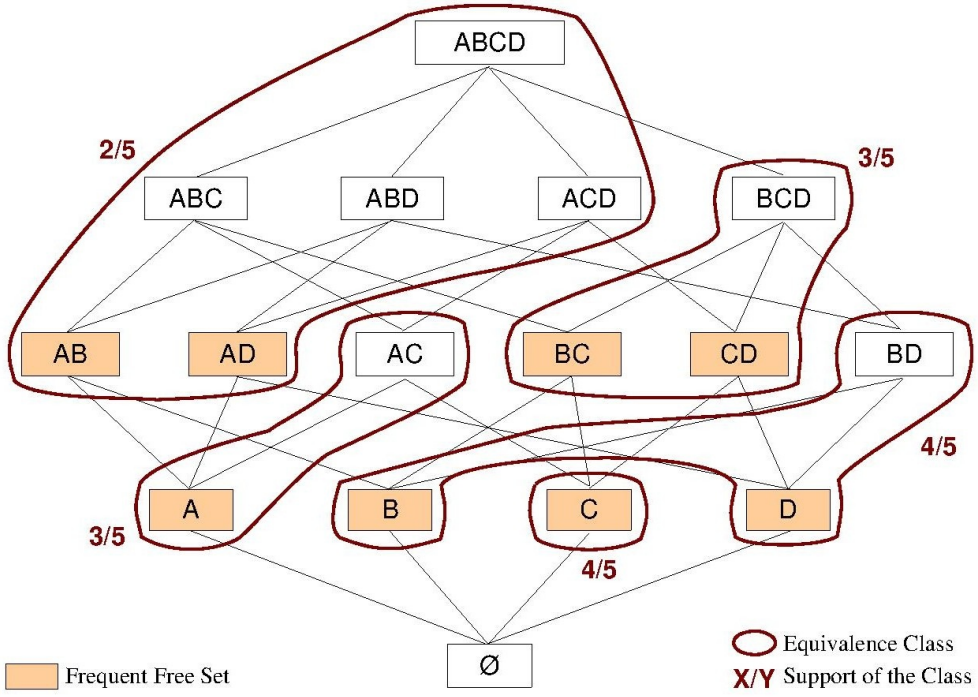


Figure 3.5: Frequent δ-free Sets for $\delta > 0$ and Equivalence Classes ($minsupport = 40\%$)

3.7 Regular Framework

Condensed representations aim at optimizing memory usage and execution time of the extraction phase of association rule mining while allowing to get the correct complete information during a post-processing operation. In the *regular framework*, regular itemsets allow to specify

which items or itemsets may or may not be present in the itemsets of the condensed representation. A regular itemset E can be represented by the *regular grammar* and is called an *extended representation of the itemset*, or *extended itemset* for short:
$E ::= X|X?|\{X_1, \cdots, X_i\}^*|\{X_1, \cdots, X_j\}^+$
where X is an itemset and X_n are items with $i \geq 0$ and $j > 0$.
In this regular itemset:

- First argument X represents an itemset.
- Second argument $X?$ indicates that the itemset X may or may not be present in the regular itemsets.
- Third argument $\{X_1, \cdots, X_i\}^*$ means that any subset of the itemset $\{X_1, \cdots, X_i\}^*$, including null or empty set, participates in the formation of the regular itemsets.
- Last argument $\{X_1, \cdots, X_j\}^+$ represents the participation of the non-empty subset of the itemset $\{X_1, \cdots, X_j\}$ to the formation of regular itemsets.

For example, the extended itemset AB{CE}+ represents itemsets containing both A and B, and possibly C and E. This itemset thus designates the set of itemsets {AB, ABC, ABE, ABCE}. The semantic of an extended itemset E is represented as $S(E)$ and E is called regular when the $cover(I_1) = cover(I_2)$, where $cover(I)$ is the list of dataset objects containing I, which implies $support(I_1) = support(I_2)$, for all itemsets $I_1, I_2 \in S(E)$. This representation is a lossless representation unlike free sets. The set of regular itemsets for dataset D is presented in Table 3.6.

Closed Itemsets	**0-Free Sets**	**Regular Itemsets**	**Support**	**Cover**
ABCD	AB, AD	$AC?\{BD\}^+$	2	1, 5
BCD	BC, BD	$B\{CD\}^*$	3	1, 4, 5
AC	A	AC?	3	1, 3, 5
BD	B, D	$\{BD\}^+$	4	1, 2, 4, 5
C	C	C	4	1, 3, 4, 5

Table 3.6: Frequent Regular Itemsets for Database D ($minsupport = 40\%$)

The Cover column of the table represents the list of dataset objects containing the corresponding itemset. The RegularMine algorithm for generating a set of regular itemsets as a concise representation of frequent itemsets and a detailed discussion on semantics and procedural descriptions can be found in [Ruggieri 2010].

3.8 Evolutionary Framework

Evolutionary approaches have been used for association rule mining to overcome some problems encountered with preceding frameworks in some applications:

- The number of attribute and value pairs, i.e., items, can be huge in some databases rendering it difficult to handle these data.

- In the case of continuous valued attributes, making a binary discretization of those attributes, to transform them in discrete values, can be difficult. Hence, choosing the proper discretization method for making the process lossless can be non trivial.
- Supplying appropriate values for the minimum support and confidence threshold parameters is most often application specific and can be difficult.

The *evolutionary framework* was introduced to address applications where the above mentioned problems appear. This framework regroups several strategies such as genetic algorithms, genetic programming, genetic network programming and differential evolution among others. Some approaches perform the rule mining task as a combination of several of these strategies, such as for example combining genetic algorithms and genetic network programming as in [Gonzales 2009]. In this framework, the association rule mining process is divided into the two following phases:

Rule generation: During the rule generation process, the rules are encoded in the chromosome according to one of the following approaches:

- In the *Pittsburg Approach*, a set of rules are encoded in a single chromosome and the length of the chromosome restricts the number of generated association rules [Dehuri 2004].
- In the *Michigan Approach*, each rule is encoded in a single chromosome divided into the antecedent and the consequent parts of the rule [Ghosh 2004].

Rule selection: After the rule set generation, several evolutionary criteria, such as support, confidence or one of the measures presented in Section 3.3, are used to select the most pertinent rules. The rule sets generated by this selection process are also known as *Pareto-optimal rules sets*.

An interesting feature of evolutionary approaches is that, instead of using only support and confidence measures, they can extend the problem to the optimization of multiple objective measures such as those presented in table 3.4. This principle, called *multi-objective optimization*, for generating Pareto-optimal rule sets can notably improve the results. Multi-objective optimization has been successfully applied to classification association rules and associative rule mining using support and confidence. For example, the Pareto-optimal associative rules based on combined support and confidence measure optimization remarkably improved results in some cases as was demonstrated in [Bayardo 1999]. The most prominent evolutionary strategies used for association rule mining are described below.

3.8.1 Genetic algorithms.

These algorithms represent rules as individual chromosomes with a fixed population size, i.e., number of chromosomes, that is a user-defined parameter. This can be a problem in some applications as interesting rules can be missed if an inadequate population size was defined. Increasing the population size is not a viable solution to this problem as too high values will lead to important execution times. An interesting review of single objective and multi-objective optimizations for association rule mining based on genetic algorithms can be found in [Davis 1991]. Genetic algorithms were used for Pareto-optimal association rule mining [Ghosh 2004], Pareto-optimal fuzzy association rule mining [Kaya 2006], Pareto-optimal classification rule mining [Ishibuchi 2005] and multi-objective fuzzy rule based classifier

[Ishibuchi 1997, Ishibuchi 2004]. Recent surveys on genetic algorithms based association rule mining are available in [Fung 2012, Ghosh 2004, Ishibuchi 2006, Zhang 2010].

3.8.2 Genetic programming.

This theory is a biologically inspired evolutionary theory for solving multi-objective optimization problems. However, this approach suffers from the drawback of a possible generation of invalid itemsets or association rules due to crossover and mutation operations used [Mendes 2001]. Genetic programming was used for mining association rules in temporal data [Hetland 2002] and an extension called G3P (Grammar Guided Genetic Programming) for solving the problem of generating invalid individuals was proposed [Couchet 2007, Whigham 1995]. The generation of valid association rules from continuous valued numerical attributes using the G3P method was also the subject of research works [Freitas 2001, Luna 2010].

3.8.3 Genetic network programming.

This extension of genetic programming is a graph-based evolutionary approach for optimizing multi-objective searches. It uses directed graph data structures for solving the problem of continuous valued attributes in association rule mining. This approach was introduced in the association rule mining domain in [Shimada 2006] and implemented using genetic algorithms in [Gonzales 2009]. Recent developments in association rule mining using genetic network programming are described in [Shimada 2008, Taboada 2007, Yang 2011c]. Few research works have done in this field and some interesting perspectives on this topics require more attention by researchers.

3.8.4 Differential evolution.

This evolutionary approach is an adaptation of natural evolution to solve optimization problems. Algorithms based on this theory were the firsts adapted to association rule mining as they are well fitted to solve the problem of association rule mining from continuous valued numerical attributes [Alatas 2008]. Although differential evolution was introduced as a single objective optimization approach, it was shown to be an effective solution to the multi-objective association rule mining problem [Alatas 2008]. The use of differential evolution to solve multi-objective optimization problems is studied in [Abbass 2001, Sarker 2004, Storn 1996].

3.8.5 Hybrid multi-objective

The use of evolutionary approaches for association rule mining presents interesting perspectives for several practical problems encountered in some applications. Their combination with deterministic symbolic approaches, to benefit from their respective capabilities and strengths, is a promising field of research.

Part II

The FIST Approach for Data Mining

Chapter 4

Algorithmic Description

Contents

In this chapter, we present in brief the FIST approach for extracting bases of conceptual association rules and conceptual clusters in one run, without extra processing time or database scan. The methodology is based on the frequent generalized suffix-tree data structure and the concept of closed itemset lattice. Working flow of the FIST approach is based on the three principal ideas:

1. Preprocessing of the database.
2. Finding frequent closed patterns from the preprocessed data.
3. Generating bases of conceptual association rules and hierarchical conceptual clusters from the patterns.

The general algorithmic flow of the FIST approach is shown in Algorithm 1. Its input is a dataset represented as a dataset in which rows represent *objects* and columns represent *variables* (or *attributes*). Each distinct value of a variable constitutes an *item*. FIST performs one scan of the input dataset to generate a compressed database from which are extracted frequent closed itemsets, generators, bases of conceptual association rules, and bi-clusters.

Two different algorithmic versions of the FIST approach were developed based on the decomposition given in algorithm 1. The first algorithm is presented in section 4.1. In this algorithmic version, operations to generate the different outputs use a table representation of frequent patterns extracted from the frequent generalized suffix-tree. However, theoretical and experimental studies showed that this pattern generation phase requires important resources. Consequently, a new algorithmic version of the approach, in which operations to generate the

Algorithm 1 FIST algorithm.

Input: Dataset, *minsupport* value, *minconfidence* value
Output: Frequent closed itemsets, generators, conceptual clusters, association rules

Phase 1: Preprocessing the dataset
1: Phase 1.1: Generate Item Table
2: Phase 1.2: Generate Sorted Frequent Database

Phase 2: Mining frequent closed itemsets
3: Phase 2.1: Create frequent Generalized Itemset Suffix-Tree
4: Phase 2.2: Find frequent closed itemsets

Phase 3: Generating knowledge patterns
5: Phase 3.1: Find generators of each frequent closed itemsets
6: Phase 3.2: Find conceptual bi-clusters
7: Phase 3.3: Generate basis of conceptual exact association rules
8: Phase 3.4: Generate basis of conceptual approximate association rules

different outputs are directly performed on the frequent generalized suffix-tree, was developed. This second version, named FIST 2.0, is discussed in section 4.2. Experimental evaluations showed that resource requirements, in memory usage, and efficiency, in execution times, of this second algorithmic version are considerably improved compared with the first algorithmic version.

4.1 FIST Version 1: First Algorithmic Version

This section describes in detail the first algorithmic version of the FIST approach for frequent closed pattern mining and bases of conceptual association rule generation. Section 4.1.1 describes the first phase of the algorithm for preprocessing the dataset. Section 4.1.2 describes the second phase for building the frequent generalized suffix-tree from the frequent sorted database created in the first phase. This section also describes how frequent closed itemsets are collected from the tree. Section 4.1.3, describes the process for generating the different patterns: frequent generators, conceptual bi-clusters and bases of conceptual association rules.

4.1.1 Phase 1: Creating Sorted Frequent Database

Before creating the tree, preprocessing operations on the initial data are performed. This preprocessing phase aims at minimizing execution times and memory usage of the subsequent phases. The main operations performed consist in deleting infrequent items, given the *minsupport* threshold parameter, and sorting resulting frequent items in increasing order of their support in the database. These operations allow to reduce the size of the data and to optimize the size and the construction process of the generalized suffix-tree. The output of this preprocessing phase is called the *Sorted Frequent Database* (SFD).

The preprocessing operations performed on the dataset, and the corresponding advantages gained from these operations, are described below:

- First operation consists to scan the dataset to identify items, count their support in the dataset and delete infrequent items, i.e., items with support less than the *minsupport*

threshold parameter. This deletion operation of infrequent items allows to ignore items that are considered as irrelevant for the actual result of pattern mining process. This phase is based on the property that all supersets of an infrequent itemset are also infrequent. It is a major phase to minimize both execution time and memory usage of the subsequent phases of the algorithm.

- Second operation consists to sort the frequent items obtained according to their support in the database. Theoretical and experimental results showed that ordering items in increasing order of their support minimizes the number of nodes in the generalized suffix-tree compared to other orderings, such as appearance order or decreasing order of their support in the dataset. Having Less nodes in the tree implies that both less memory is needed to store the tree in main memory and that the time required to traverse the tree during further operations is minimized. The property has been examples in the next chapter of this part under this thesis.

This preprocessing phase is fast, compared to the other later phases of the algorithm, and plays an important role for the next two phases in terms of performances and required resources. Moreover, data resulting from this phase can be stored in secondary memory in order to avoid repeating it if different *minconfidence* threshold values are used to generate association rules for the same *minsupport* value. However, this phase must be performed for each different *minsupport* threshold value used. This preprocessing operation have been done into two steps, one for creating the numerical table from the original data by considering each attribute value pair.

The different tasks performed during the preprocessing are described in detail in the following subsections. Section 4.1.1.1 describes the one time preprocessing procedure for generating the *Number Table* corresponding to the initial dataset and minimizing the physical memory required to represent the data. The second task is based on the user's defined *minsupport* threshold provided by the user. It consists in the creation of the SFD by deleting infrequent items and sorting frequent items. This task, described in detail in section 4.1.1.2, outputs the SFD that is used for creating the frequent generalized suffix-tree from which the frequent closed itemsets are extracted. For each different support value, the second step of this phase should perform before going to the next phase of the algorithm. Figure 4.1 shows an organigram type representation of the preprocessing operations and corresponding outputs of FIST.

4.1.1.1 Preprocessing the Dataset

During this step of the preprocessing phase, dataset values are mapped to discrete numbers and their number of occurrences in the dataset is counted. As for all itemsets based approaches, if the initial dataset contains numerical continuous values for the variables, a suitable discretization of variables is required before this step is performed. This mapping creates a table called *Item Table* (IT) in which all pairs {variable, value} in the dataset, where values can be boolean, numerical, nominal or textual, are mapped to *items* represented as discrete numbers. To create this table, the mapping function associates a unique number to each pair {variable, value}. Simultaneously with the mapping, the *support* of each item in the dataset (number of occurrences) is counted. The IT table generated during this phase will be used after the frequent patterns extraction to generate a user understandable result in which interpretable identifier are used instead of discrete numbers. This process allows FIST to treat datasets where vari-

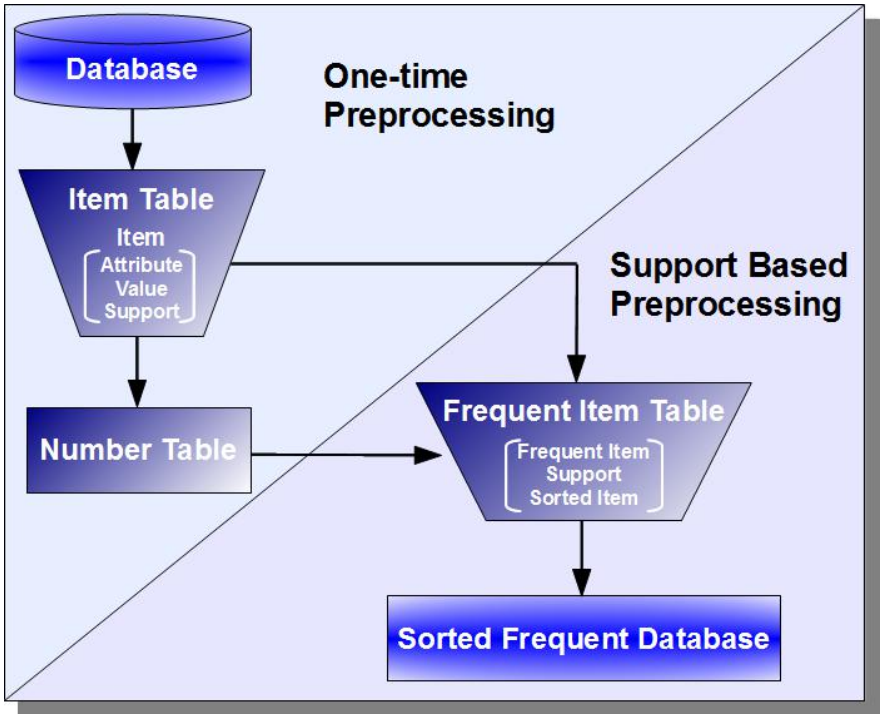


Figure 4.1: Database Preprocessing Phase

ables can be represented by more than one column. This feature is important to be able to process datasets integrating annotations, such as GO or PubMed annotations for instance, that are represented by several columns since more than one annotation can be associated to each object (row).

After the IT table was generated using the mapping function, the corresponding representation, with item discrete numbers, of the original dataset is created. This new representation, called the *Number Table* (NT) of the dataset, optimizes the memory space required for storing itemsets and improves the efficiency of comparison operations used in later phases of the algorithm. This IT table is a one-time process and its result can be reused for preparing the different SFD corresponding to different *minsupport* values, which is the purpose of the next step.

The detail process of this preprocessing phase is presented for two different example datasets below. The first is a biological dataset represented in multi-valued data matrix format in table 4.1. The second is a biological dataset represented in binary data matrix format in table 4.2.

First, a mapping of each distinct *variable* = *value* pair, where variables are matrix columns and values are the different values found in rows for the corresponding column, is created. The result is the IT table in which each pair *variable* = *value* is associated to an unique item that is a discrete number. During this process, the number of occurrences of each distinct item in the dataset is counted as its *support*. The IT tables of the two example biological datasets $D1$ and $D2$ are shown in table 4.3 and 4.4 respectively.

From the IT table, a new table in which the original data are replaced by item numbers is

Gene	S1	S2	S3	A1	A2	O1	O2
g1	o	u	n	abc	xyz	ooo	oo1
g2	o	n	o	abd	wxy	oo1	oo2
g3	n	u	o	abc	xyz	ooo	oo2
g4	o	o	o	bca	yxz	oo1	oo1
g5	n	u	u	bca	zxy	oo1	oo1

Table 4.1: Multi-valued Data Matrix Representation of the Biological Database $D1$

OID	P_1	P_2	P_3	P_4	P_5	A_1	A_2	A_3	A_4
O_1	1	?	1	?	?	?	1	?	1
O_2	?	1	?	?	1	?	?	1	?
O_3	1	?	1	?	?	1	1	?	?
O_4	?	?	?	1	?	1	?	?	?
O_5	?	1	?	1	1	1	?	1	?
O_6	1	?	1	?	?	?	1	?	1
O_7	?	1	?	?	1	?	?	1	?
O_8	1	?	1	1	?	?	1	?	1
O_9	?	1	?	?	1	?	?	1	?
O_{10}	1	1	1	?	1	?	1	?	1

Table 4.2: Binary Data Matrix Representation of the Biological Database $D2$

created as a new compressed representation of the dataset. This table is called *Number Table* (NT). The NT tables for the two example datasets $D1$ and $D2$ are shown in table 4.5 and 4.6 respectively. These IT and NT tables will be used during the next phases of the approach to generate the SFD.

4.1.1.2 Preprocessing with *minsupport*

The last step of the preprocessing phase is the creation of SFD corresponding to the NT for a given *minsupport* value. This step aims at creating a dataset containing only frequent items,

Item	Attribute : Value	Support	Item	Attribute : Value	Support
1	S1 : o	3	11	A2 : wxy	1
2	S2 : u	3	12	O1 : oo1	3
3	S3 : n	1	13	O2 : oo2	2
4	A1 : abc	2	14	S1 : n	2
5	A2 : xyz	2	15	S2 : o	1
6	O1 : ooo	2	16	A1 : bca	2
7	O2 : oo1	3	17	A2 : yxz	1
8	S2 : n	1	18	S3 : u	1
9	S3 : o	3	19	A2 : zxy	1
10	A1 : abd	1			

Table 4.3: Item Table for Dataset $D1$

Data	Support	Item
P_1	5	1
P_3	5	2
A_2	5	3
A_4	4	4
P_2	5	5
P_5	5	6
A_3	4	7
A_1	3	8
P_4	3	9

Table 4.4: Item Table for Dataset $D2$

Object	Numbers
g1	1 2 3 4 5 6 7
g2	1 8 9 10 11 12 13
g3	14 2 9 4 5 6 13
g4	1 15 9 16 17 12 7
g5	14 2 18 16 19 12 7

Table 4.5: Number Table for Dataset $D1$

by eliminating the infrequent items as these don't play any role in the finding of frequent closed itemsets, to minimize the size of data processed. This to optimize memory space required for storing itemsets and improve the efficiency of comparison operations. Unlike NT table generation, this phase is repeated for each different *minsupport* value provided by the user.

This preprocessing task is performed into two steps. First, the infrequent items, defined according to the *minsupport* threshold, are deleted from the IT table. Second, the remaining items are sorted in increasing order of their support in the NT and, if more than one item have the same support value they are sorted in lexicographic order to optimize further comparisons. Discrete numbers associated to items in the IT table are then updated, taking into account the suppressed items, to optimize their numerical representation. The resulting table

Object	Numbers
O1	1 2 3 4
O2	5 6 7
O3	1 2 8 3
O4	9 8
O5	5 9 6 8 7
O6	1 2 3 4
O7	5 6 7
O8	1 2 9 3 4
O9	5 6 7
O10	1 5 2 6 3 4

Table 4.6: Number Table for Dataset $D2$

is called the *Frequent Item Table* (FIT). The FIT tables for the two example datasets $D1$ (for *minsupport*=2/5) and $D2$ (for *minsupport*=4/10) are given in table 4.7 and 4.8 respectively.

Frequent Item	**Support**	**New Item**	**Frequent Item**	**Support**	**New Item**
4	2	1	1	3	7
5	2	2	2	3	8
6	2	3	7	3	9
13	2	4	9	3	10
14	2	5	12	3	11
16	2	6			

Table 4.7: Frequent Item Table for Dataset $D1$ and *minsupport*=2/5

Frequent Item	**Support**	**New Item**
4	4	1
7	4	2
1	5	5
2	5	3
3	5	6
5	5	4
6	5	7

Table 4.8: Frequent Item Table for Dataset $D2$ and *minsupport*=4/10

Then, the NT table is updated with the updated item numbers to generate the SFD. It should be noted that if a row in the NT table contains only infrequent items, then it will not represented in the SFD. Consequently, rows of the initial dataset containing only infrequent items are not represented in the SFD that can thus contain fewer objects (rows) than the initial dataset. The sorted frequent databases for the two example datasets $D1$ (for *minsupport*=2/5) and $D2$ (for *minsupport*=4/10) are shown in table 4.9 and 4.10, respectively.

Object	**Items**
g1	1 2 3 7 8 9
g2	4 7 10 11
g3	1 2 3 4 5 8 10
g4	1 6 9 10 11
g5	5 6 8 9 11

Table 4.9: Sorted Frequent Database for Dataset $D1$ and *minsupport*=2/5

If the itemsets to be considered can appear in only one row of the dataset, then the SFD creation consists only in the mapping from item numbers in the NT to item numbers in the FIT. This will be the case if all patterns supported by at least one row are searched for.

Object	Items
O1	1 3 5 6
O2	2 4 7
O3	3 5 6
O5	2 4 7
O6	1 3 5 6
O7	2 4 7
O8	1 3 5 6
O9	2 4 7
O10	1 3 4 5 6 7

Table 4.10: Sorted Frequent Database for Dataset $D2$ and $minsupport$=4/10

4.1.2 Phase 2: Mining Frequent Closed Itemsets

This second phase is the core of the FIST approach, where the frequent closed itemsets are mined from the SFD database, and the most computationally intensive phase of the process. This phase is carried out in two steps. First step is the generation of the *Frequent Generalized Itemset Suffix-Tree* (FGIST) that is a main memory data structure specific to the FIST algorithm. It can be viewed as the compressed representation of the SFD in primary memory. Each internal node in the FGIST stores:

- An item number.
- A reference to a list of object identifiers if the path from the root to this node corresponds to a complete suffix itemset.
- A link to the next node of the branch, except if the node is a leaf.

Each branch from the root to a leaf represents an itemset and the object list contains identifiers of objects supporting the itemset. The details description of this step is given in section 4.1.2.1.

The second step of this phase is the extraction of the frequent closed itemsets from the FGIST primary memory structure. This extraction is based on intersection and inclusion operations performed between the branches, i.e., itemsets collected from the branches, and between the object lists present in the subsequent leaf nodes, respectively. This step, that is the most complex task of the process, is detailed in section 4.1.2.2.

4.1.2.1 Building the Frequent Generalized Itemset Suffix-Tree

During this process, the whole SFD database is accessed only once, from the first to the last row, and stored in primary memory as a FGIST that is a condensed representation of the dataset. Rows are processed one by one, and the list of item numbers read from the row, constituting an itemset, is pushed into the tree as a set of suffixes. This data structure is optimized as almost all frequent itemsets resulting from intersections of the dataset rows are represented as branches in the FGIST. This relies on the fact that branches of the tree are stored in ascending order and this property is ensured by the fact that items are ordered in ascending order of their support in the dataset.

For each itemset read from the dataset, suffixes of the itemset are created by deleting successively one item of the itemset from the first to the last. Then suffixes, associated to the

object identifier represented as a sequential order number (OID), are processed to build the FGIST. The insertion of a suffix in the FGIST is a recursive procedure starting from the root node and successively descending in the branches corresponding to the consecutive items of the suffix. If a suffix was already inserted, and is thus already represented as a branch in the FGIST, then only the object list is updated by adding the object identifier of the suffix. The pseudo-code of the algorithm for building the FGIST is presented in algorithm 2.

Algorithm 2 Building the FGIST

Input: Sorted Frequent Database (SFD)
Output: Frequent Generalized Itemset Suffix-Tree

```
1: begin
2: initialize k with |SFD|
3: for i = 1 - k do
4:    S_i ← φ
5:    S_i ← D(i,:)
6:    s ← φ
7:    s = genSuffix(S_i, i)
8:    for all s_j ∈ s do
9:       if s_(j,1) ∈ ROOT.children then
10:         match(node(i), s_(j, i))
11:      else
12:         ROOT.children ← s_j,1
13:         edge ← (ROOT → ROOT.children)
14:         delete(s_j,1);
15:         buildEdge(ROOT.children, s_j)
16:      end if
17:   end for
18: end for
19: end
```

The first step in this process is use to count the total number of objects in the database (step 2). The number of rows will help to create the character terminator for each vector and determine how many time the whole process for creating and updating the tree will continue. We can count this number only by using the object id column, no database pass is necessary for this step. Step 3 will verify which row will process next and each row in the database passes through the two phases. In the first phase (step 7), generates all the suffixes of the vector corresponding to object being processed described in the *genSuffix* function described in the section 4.1.2.1, which takes the whole object and the terminating symbol as argument and generates a set of suffixes. For this phase, we need to access one row of the database at a time and mapped it as a vector of numbers (step 5). The second phase of the process (step 8 to 17) is applicable for each suffix of the suffix set s generated by the *genSuffix* function. The very first step in this phase is to check the ROOT node of the tree to determine whether any edge has been created previously from the ROOT with the first item of the suffix being process (step 9). If any edge has already been created from ROOT of the tree with the first item of the suffix then go for the next unmatched position and do the necessary steps in that specific branch of the tree using the *match* function (step 10). It passes the current node and the suffix to the

match function. The description of the *match* function is given in section 4.1.2.1. If there is no node present under the root node (step 11), then create a node with the first item of the suffix and make an edge between root and the new node (step 12-13). After creating the first node of the branch, delete the first item of the suffix (step 14) and redirect to the *buildEdge* function (step 15). It passes two argument to the *buildEdge* function, the new node created under the root node and the suffix after deleting the first item from it. The *buildEdge* function, described in section 4.1.2.1, is used to build an edge from the node specified in the argument and using the portion of the suffix passed to the function. In this algorithm, each row read is represented as a vector S_i of numbers and the OID of the row in the SFD is used as a *suffix terminator* and passed to the suffix generation function. The suffixes are the subsets of the itemset S_i obtained by deleting successively one item to S_i from the first to the penultimate. The k^{th} item of S_i is denoted S_i^k in the algorithms. During this process, the SFD is accessed only once. This minimizes disk accesses that are time expensive operations as they are about of the order of million times longer that primary memory accesses.

genSuffix() Function This function takes one vector (string) at a time, with a *suffix terminator*, and generates the set of all the suffixes of the vector. The pseudo-code of this function is given in algorithm 3. It works as follows. First, the function initializes variable k with the length of the vector (step 2). This value indicates how many suffixes will be generated, or the number of times the loop will be performed. For each loop, the elements from the vector are collected and a suffix of the vector is created (step 3-4). To create the m^{th} suffix, the elements from m^{th} position to the end of the vector are collected (step 5-7). Finally, the suffix terminator, that is the OID of the object considered, is added at the end of the suffix (step 8). After processing the vector, the function outputs the set of all the suffixes generated (step 10) that is returned back to the main algorithm as a suffix set (step 11).

Algorithm 3 Function: genSuffix(S_i, C_i)

1: **begin**
2: **initialize** k **with** the length of S_i
3: **for** $m = 1$ - k **do**
4: $s_m \leftarrow \phi$
5: **for** $n = m$ - k **do**
6: $s_m \leftarrow s_m \bigcup S_{i,[n]}$
7: **end for**
8: $s_m \leftarrow s_m \bigcup C_i$
9: **end for**
10: s = $\bigcup \{s_m\}$
11: **return(s)**
12: **end**

Structure of FGIST Nodes Nodes in the FGIST data structure will store three different types of values depending on their position in the tree. The structure of FGIST nodes is given in table 4.11. When a node is created for the first time, it only stores the item number identifier in the *Item* field and other two fields are initialized to null. When one child node is created under the node then the *HashTable* field is initialized to store the links to the child nodes.

When a suffix which last item correspond to this node is inserted in the tree then the *ObjectIds* field is initialized with the suffix terminator (OID) of the suffix.

Field	Contains	Generic Type
Item	Identifier of the item	Number
ObjectIds	Supporting objects (OIDs)	List of numbers
HashTable	Links to child nodes	Hashtable

Table 4.11: Structure of FGIST Nodes

match() Function This function is used when two or more suffixes contain a similar sequence of items at their beginning. Then, when the second is inserted in the FGIST, it finds the next unmatched node in the branches of the tree. In such situations, we have to traverse a particular branch of the tree from its root to the last matching item (node) of the suffix. This recursive function goes inside the tree recursively to find the first unmatched node present, if any. When this unmatched node is found, the *buildEdge* function, described below, is called to insert the remaining portion of the suffix from the last matched item to the last.

Algorithm 4 Function: match(node(I), s_j)

```
1:  begin
2:  delete s_(j,1)
3:  if node(I).children is not empty then
4:    for all element J in node(I).children do
5:      if s_(j,1) = child(J) then
6:        match(node(I), s_j)
7:      else
8:        buildEdge(node(i), s_j)
9:      end if
10:   end for
11: end if
12: return
13: end
```

The first step of this function is to delete the first item of the suffix being passed to it (step 2). If any children is found under the current node, then for each children it checks the similarity between the first element of the remaining part of the suffix with the items present in the child nodes (step 3-4). If any match is present in the children of the node, then it goes inside the branch following the matched child and checks recursively the remaining items of the suffix (step 5-6). If no similarity has been found under the children of the current node, then the *buildEdge* function is called from the current node to create a sub-branch from the last matched node with the remaining suffix (step 7-9). After creating the sub-branch it returns back to the main algorithm (step 10-12).

buildEdge() Function This function receives as its arguments a node of the FGIST and a suffix (vector of items and suffix terminator). It creates in the FGIST a sub-branch correspond-

ing to the suffix starting from the node passed as argument. The pseudo-code of the function is given in algorithm 5. As mentioned previously, the function is called from two different places. First, from the main algorithm when a particular node is created for the first time from the root. Second, inside the match function, when an unmatched is found between the children of the current node and the suffix to be inserted. In the first case, the first item of the suffix is assigned to the ROOT.item in the main algorithm and the remaining items of the suffix are added to the successively created child nodes in a recursive manner for instance.

Algorithm 5 Function: buildEdge($node(I)$, s_j)

```
1: begin
2: l← s_i.length()
3: if l ≠ 1 then
4:    node(I).children = s_{j,1}
5:    edge ← (node(I) → node(I).children)
6:    delete(s_{j,1})
7:    buildEdge(node(I).children, s_j)
8: else
9:    node(I).object = s_j
10: end if
11: return
12: end
```

This function first initializes the l variable with the length of the suffix received as argument (step 2). If the length is greater than one, then a node with the first item of the suffix passed as an argument is created (step 3-4). Then, a link between the current node and the new node is created and the first item from the suffix is deleted (step 5-6). In a recursive way, it goes inside the new node to process the remaining part of the suffix (step 7). When the length is equal to one, this means that the suffix received as argument is the suffix terminator, i.e., OID (step 8). Then, we add this OID to the *ObjectIds* field of the current node (step 9).

4.1.2.2 Extracting Frequent Closed Patterns

The second step of the second phase consists in extracting the frequent closed itemsets (FCI), with the list of dataset objects containing each, from the FGIST tree. Each entry in the *FCI* table contains two elements: A list of items and the list of identifiers of objects containing this itemset (OID). First, each branch of the FGIST tree from the root to a leaf is traversed and a new entry in the *FCI* table is created for the itemset corresponding to that branch. The associated list of OID is initialized with the list in the leaf node of that branch. The size of this OID list corresponds to the support of the itemset in the dataset. Then, the non-closed itemsets in the *FCI* table are identified using associated OID lists as follows. If an itemset is included in another itemset and both have identical OID lists, then the included itemset is not closed and then deleted. Finally, the frequent closed itemsets not already found are identified by performing intersections between two closed itemsets in the *FCI* table and verifying if the resulting itemset is not infrequent, using OID lists, and inserted in the table if not already present. The associated OID list is the result of the union of the OID lists of the two intersected itemsets. If a new frequent closed itemset is generated, then the process is repeated for the

new generated itemsets and this iterative process ends when no new frequent closed itemset is generated. At the end, the *FCI* table contains all frequent closed itemsets with associated list of objects containing each of them. The pseudo-code of this step is presented in algorithm 6.

Algorithm 6 Extracting Frequent Closed Patterns

Input: Frequent Generalised Itemset Suffix Tree
Output: FCI_Table containing [Itemset, Object Ids]

1: **begin**
2: **for each** itemset L_i **in** $fGIST$
3: **insert** pattern $\{L_i, L_i.OIDs\}$ **into** FCP
4: **for each** itemset L_j successor of L_i **in** $fGIST$ **with** length(L_j) > length(L_i)
5: **if** $L_i \subset L_j$ **and** $L_i.OIDs = L_j.OIDs$ **then**
6: **delete** pattern $\{L_i, L_i.OIDs\}$ **from** FCP
7: **end if**
8: **end for**
9: **end for**
10: $NFCP \leftarrow FCP$
11: **while** $NFCP \neq \emptyset$ **do**
12: $NFCP \leftarrow \emptyset$
13: **for each** itemset L_k **in** $NFCP$
14: **for each** itemset L_i **in** FCP
15: $L_m \leftarrow L_i \cap L_k$
16: **if** $L_m \neq \emptyset$ **and** $L_m \notin FCP$ **then**
17: $L_m.OIDs \leftarrow L_i.OIDs \cup L_k.OIDs$
18: **insert** pattern $\{L_m, L_m.OIDs\}$ **into** FCP
19: $NFCP \leftarrow NFCP \cup \{L_m, L_m.OIDs\}$
20: **end if**
21: **end for**
22: **end for**
23: **end while**
24: **end**

The output is the *FCP* set containing the list of FCIs with their associated OID list. During this step, each itemset corresponding to a branch of the FGIST from root to leaf is traversed and the algorithm tests if this itemset is closed or not as follows (steps 2-9). Properties of closed itemsets says that an itemset is not closed if it is included in another itemset and both have identical OID lists. Using this property, by traversing from root to leaf a new entry is created in the *FCP* set for the collected items L_i and the corresponding OID list $L_i.OIDs$ in the leaf node (step 3). The non-closed itemsets are then identified and deleted from the *FCP* set: If an itemset is included in another itemset and both have identical object lists, then the included itemset is deleted from, or not added to, *FCP* set (steps 5-6).

The second operation consists to identify the remaining few frequent closed itemsets not in the FGIST (steps 10-23). These FCIs are those that can be obtained only by intersecting two FCIs and that do not corresponds to suffixes of itemsets in the SFD. For this, intersection operations are performed between each pair of FCIs in the *FCP* set and if the resulting set is frequent and not present in the *FCP* set, a new FCP is generated (steps 16-18). The object

list of this new FCP is the union of the object lists of the two intersected FCIs (step 14). This procedure for identifying new FCPs continues till no new FCP is found in this way (step 11-23). However, all pairs of FCIs don't have to be tested and only newly created FCIs L_k are intersected with other FCIs in the *FCP* set. For this, new FCPs are stored in the *NFCP* reference set (step 12 and 19). For the first iteration, the *NFCP* reference set is initialized with *FCP* member references (line 10). At the end, the final *FCP* set contains all frequent closed itemsets with associated OID list.

4.1.3 Phase 3: Generating Knowledge Patterns

The final phase of the approach is to generate the conceptual patterns from FCI and associated OID lists in the *FCI* table. These patterns are the frequent conceptual bi-clusters, generators and bases of conceptual association rules. Association rule set extracted contains both the minimal cover for exact association rules and the structural minimal cover for approximate association rules. These minimal covers, or bases, contain, respectively, the non-redundant exact and approximate association rules with minimal antecedent (predictor itemset) and maximal consequent (predicted items). Minimality and maximality are defined here according to the inclusion relation. The pseudo-code of the knowledge pattern generation is given in algorithm 7.

First, rows in the *FCI* table are sorted in increasing order of the size of FCI (step 2) and the output sets BIC (for bi-clusters), GEN (for generators), and AR (for all association rules) are initialized to empty set (step 3). For each entry of the *FCI*, the table is processed successively (step 4-26) for creating hierarchical bi-clusters (step 5) and identifying generators and association rules (step 6-25). All subsets stored in SUB of itemset FCI[i].Itemset are generated, sorted in increasing order of their sizes (step 8-9) and processed one by one (step 11-23). The algorithm first determines if S is a generator of FCI[i].Itemset (step 12-14). If it is a generator then stored in the set GEN. Then, all association rules with S as antecedent are generated if their confidence is greater than or equal to the *minconfidence* threshold (step 15-22). Finally, knowledge patterns in the BIC, GEN, and AR sets are mapped to the original data values and object identifiers in order to simplify their interpretation by the end-user using data structures generated during the preprocessing phase (step 27).

4.2 FIST Version 2: Second Algorithmic Version

In this section, we describe the second algorithmic version of the FIST approach. This version results from theoretical and experimental studies of the first version to improve both memory usage and processing time efficiency. In this version, the three main phases of the first version are divided into five steps. The pseudo-code of the FIST 2.0 algorithm, showing these five steps, is given in algorithm 8.

The following subsections describe each step of the five steps separately. Section 4.2.1 describes the transformation of the initial algorithm for building the SFD to the second version. Section 4.2.2 presents the second algorithmic process for building the FGIST from the SFD that, unlike first algorithmic version where table operations are used, performs all required operations to represent the frequent closed itemsets in the FGIST. The function testing the lexicographic order between two itemsets, that is a major feature for optimizing the next two steps, is presented in the section 4.2.3. The table based operations of the first algorithmic

Algorithm 7 Generating Knowledge Patterns

Input: FCI table, *minconfidence* value, IT table
Output: Bi-clusters (BIC), Generators (GEN), Association Rules (AR)

```
1: begin
2: sort FCI in increasing size of itemsets
3: GEN, BIC, AR ← ∅
4: for all row FCI[i] in FCI do
5:    BIC ← {FCI[i].Itemset, FCI[i].Object_list}
6:    M ← FCI[i].Itemset.size()
7:    if (M ≥ 2) then
8:       SUB ← list of subsets of FCI[i].Itemset
9:       sort SUB in increasing size of subsets
10:      K ← SUB.size()
11:      for all subset S in SUB do
12:         if (S ∉ GEN) and (S ∉ FCI.Itemset) then
13:            GEN[i] ← S
14:         end if
15:         for j = 1 - K do
16:            if (S.size() + SUB[j].size() = M) and (S ≠ SUB[j]) then
17:               create rule R : {S ⟹ SUB[j]}
18:               if (confidence(R) ≥ minconfidence) and (R ∉ AR) then
19:                  AR ← {R, support(R), confidence(R), FCI[i].Object_list}
20:               end if
21:            end if
22:         end for
23:      end for
24:      SUB ← ∅
25:   end if
26: end for
27: map patterns in BIC, GEN, AR to dataset values in IT
28: end
```

version correspond to two new steps: Updating the FGIST, presented in section 4.2.4, and then pruning useless nodes from the FGIST, presented in section 4.2.5. The optimized process for getting different pattern outputs from the FGIST are presented in section 4.2.6.

4.2.1 Step 1: Generating the Sorted Frequent Dataset

This step, similar to the first phase of the first algorithmic version, generates the SFD from the initial dataset. Compared to the first version, this creation is straightforward and requires fewer operations to complete. Unlike the first version, in which the IT, NT and FIT are used, this version directly computes the SFD via the IT. First, the list of all *variable* = *value* pairs is generated, and the support of each is counted, from the initial dataset. Then, infrequent pairs are deleted using the *minsupport* value, the remaining frequent pairs are sorted in increasing order of their support and a unique numerical value is assigned to each pair. These numerical

Algorithm 8 FIST 2.0: General Algorithm

Input: Source dataset; minsupport; minconfidence
Output: Frequent conceptual patterns
1: **begin**
2: **build** the Sorted Frequent Database from source dataset
3: **construct** the Frequent Generalized Suffix-Tree from Sorted Frequent Database
4: **update** object lists in Frequent Generalized Suffix-Tree
5: **prune** non-closed and infrequent itemsets from Frequent Generalized Suffix-Tree
6: **generate** conceptual knowledge pattern outputs
7: **end**

values, representing items, are then used to generate the SFD by replacing frequent attribute-value pairs in the initial dataset by the corresponding item numbers in appropriate position in the SFD. The pseudo-code of this function is given in algorithm 9.

Algorithm 9 FIST 2.0: Generate Sorted Frequent Database

Input: Source dataset; minsupport
Output: Sorted Frequent Dataset
1: **begin**
2: **compute** support **of** each source dataset value
3: **insert** frequent source dataset values in IT
4: **sort** frequent source dataset values **in** increasing support order
5: **associate** successive item numbers **to** source dataset values
6: **for all** row R **in** source dataset **do**
7: **if** at least one value **in** R is frequent **then**
8: **write** ordered items corresponding to R values **in** Sorted Frequent Dataset
9: **end if**
10: **end for**
11: **end**

The support of data values is first computed (step 2) and frequent values are inserted in the IT table (step 3). These values are then sorted in increasing order of their support (step 4) and a unique sequential number is assigned to each (step 5). Then, the data values in the initial dataset are replaced by the corresponding item number row by row and sorted in the row (steps 6-10). Thus, source dataset rows that do not contain at least one frequent item will not appear in the SFD.

This function improves global efficiency, in time and space, compared to the first version. However, this process must be repeated for each different *minsupport* value, whereas in the first version the one-time preprocessing is done only once for a dataset and the SFD creation is performed for each different *minsupport* value.

4.2.2 Step 2: Constructing the Frequent Generalized Itemset Suffix-Tree

The second version of algorithm for FGIST creation is given in algorithm 10. Few changes have been performed for constructing the FGIST compared to the first version of FIST. The main change is in the processing of suffixes. This version generates only one suffix of a row, and no suffix terminator is used, at a time, whereas in first version, all suffixes of a row, with suffix terminator, are generated and passed to the function inserting them in the FGIST at once. This change saves memory and avoid time bottleneck for very large rows, for which it is possible to have hundreds of suffixes. Insertion of the suffixes in the FGIST is not discussed in detail here since it is identical to the first version: It uses the *match()* and *buildEdge()* functions presented in section 4.1.2.1.

Algorithm 10 FIST 2.0: Construct Frequent Generalized Suffix-Tree

Input: Sorted Frequent Database (SFD)
Output: Frequent Generalized Itemset Suffix-Tree (FGIST)

```
1:  begin
2:  row_num ← 1
3:  for all row R of SFD do
4:    for i = 1 to |R| do
5:      suffix ← φ
6:      for j = i to |R| do
7:        suffix ← R_j
8:      end for
9:      insert suffix, row_num in FGIST
10:   end for
11:   row_num ← row_num + 1
12: end for
13: end
```

First, the *row_num* variable is initialized to 1 (step 2). This variable is used for counting the number in the dataset of the row processed and is used as suffix terminator (OID) for the suffix, and thus inserted in the *object_list* field of the appropriate FGIST node. Then, for each row of the SFD (step 3), generate one suffix of the row and send it, along with *row_num*, for insertion in the FGIST (step 4-10). Insertion in the FGIST is performed in the same way as in first algorithmic version using the *match()* and *buildEdge()* functions. After inserting all the suffixes of a given row, the *row_num* is incremented (step 11).

4.2.2.1 Frequent Generalized Itemset Suffix-Tree Data Structure

The FGIST data structure used for the second algorithmic version shows few differences with the one used in the first version. First, the root node is represented using a data structure called *HTree* that is basically a list of pointers toward other data structures called *HNodes* representing nodes. The HNodes contain three fields corresponding to different type of information: Item, Children, and Object_list. Item stores the item identification number used to represent itemsets. Children is a list of pointers toward children HNodes of the node. Object_list contain

the list of OIDs of objects of the dataset containing the item. HNodes are divided into three categories according to their position in the tree and of the item they represent in the itemset:

Internal HNode: Stores an item identifier and pointers to children nodes. The Object_list of this node is assign to null.

Internal HNode with Object List: This non-leaf node represents the final item of a suffix. It stores the item identifier, pointers to children nodes and the OID list of objects containing the suffix.

Leaf HNode: This node represents the final item of a suffix that has no superset in the FGIST. It stores only the item identifier and the OID list of objects containing the suffix. Its Children field is assign to null.

The structure of the four different types of elements (root and branch nodes) constituting the FGIST are represented in figure 4.2.

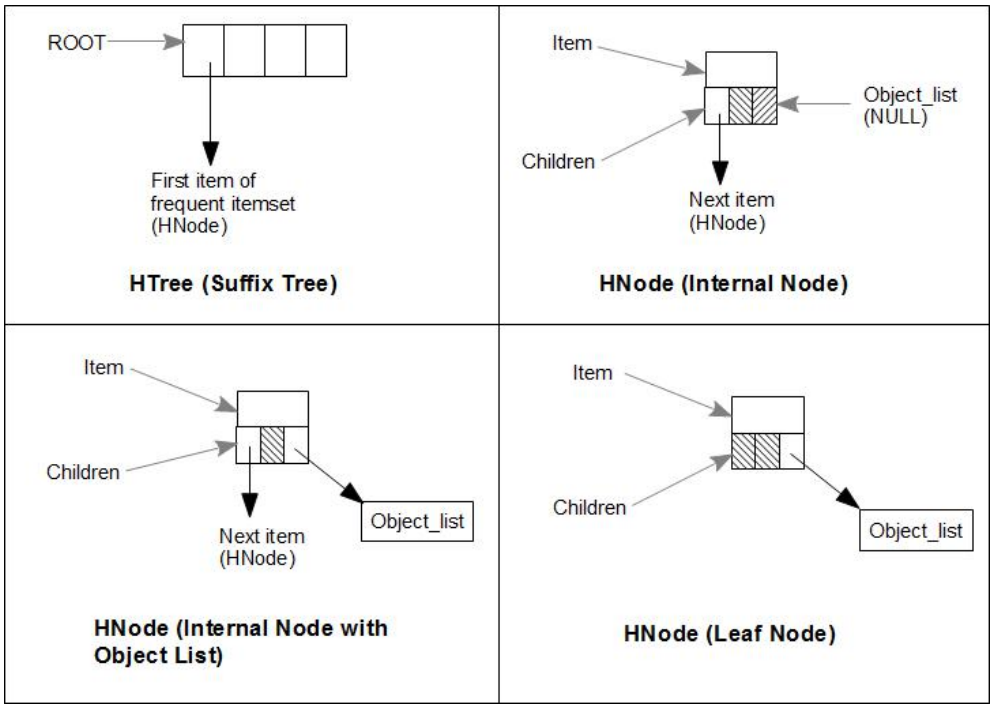


Figure 4.2: Structure of Nodes in the FGIST

4.2.3 Lexicographic Order *greaterThan*() Function

The next two steps for FCP generation are updating and pruning the FGIST. These two steps use a function to determine order between two itemsets and this function is crucial for the efficiency of these steps. The pseudo-code of this function, called *greaterThan*(), is given in algorithm 11. It receives two itemsets $I2$ and $I1$ as arguments and determines if $I2$ is

lexicographically greater than $I1$. It corresponds to the *Itemset Lexicographic Order Relation*, that is a total order relation between itemsets, given in definition 5.6 in section 5.3.2. Table 4.12 shows nine examples of comparison cases. This function avoids the execution of irrelevant update and prune operations and thus optimizes the performance.

Itemset $I1$	{1, 2}	{1, 2, 3}	{1, 2}	{1, 2}	{1, 2, 3}	{1, 2}	{1, 3}	{1, 3}	{1, 3, 4}
Itemset $I2$	{1, 2}	{1, 2}	{1, 2, 3}	{1, 3}	{1, 3}	{1, 3, 4}	{1, 2}	{1, 2, 3}	{1, 2}
$I2 >_{lex} I1$	False	False	True	True	True	True	False	False	False

Table 4.12: Output of *greaterThan* Function for Different Conditions

Algorithm 11 Function: greaterThan(Itemset I2, Itemset I1)

```
1: begin
2: if I2.equalsTo(I1) then
3:    return false
4: end if
5: greater ← true
6: stop ← false
7: i ← 0;
8: while stop = false and i < I2.size and i < I1.size do
9: if I2[i] < I1[i] then
10:    stop ← true
11:    greater ← false
12: else
13:    if I2[i] > I1[i] then
14:       stop ← true
15:    end if
16: end if
17: i ← i + 1
18: end while
19: if stop = false and I2.size < I1.size then
20:    greater ← false
21: end if
22: return greater
23: end
```

First, the function checks if I2 is equal to I1 and exits if true (step 2-4). If I2 and I1 are different, the three variables – *greater*, used to store the boolean result of the call; *stop*, used to stop the execution when I1 is found greater than I2; i, indicating the position in the itemsets – are initialized (step 5-7). Then, each pair of items at the same position in I1 and I2 are compared successively (steps 8-18). The loop stops and returns false as result when the i^{th} item of I2 is less than the i^{th} item of I1 (steps 9-11). Else, if i^{th} item of I2 is greater than i^{th} item of I1, the loop stops and returns true (steps 12-18). If both of the above conditions are not satisfied, then the loop continues until end of one itemset is reached (step 17). If the loop terminates and above conditions are not satisfied, then if I2 is smaller than I1 the function returns false (steps 19-21).

4.2.4 Step 3: Updating the Frequent Generalized Itemset Suffix-Tree

Since table based operations used in the first algorithmic version are expensive in both time and memory usage, the second version uses tree-based operations for the FGIST completion after its construction from the SFD. This process is decomposed into two different phases: First, update the tree to get the complete set of OID lists for each and all itemset in the FGIST, and second, prune from the FGIST to non-closed and infrequent itemsets. In this section, we present the FGIST update step. The general schema of this update is given in algorithm 12.

Algorithm 12 FIST 2.0: Updating the Frequent Generalized Itemset Suffix-Tree

Input: Frequent Generalized Itemset Suffix Tree
Output: Updated Frequent Generalized Itemset Suffix-Tree

```
1: begin
2: repeat
3:   for all itemset I1 with object list O1 ≠ ∅ in FGIST do
4:     for all itemset I2 with object list O2 ≠ ∅ such as greaterThan(I2, I1) in FGIST do
5:       if I1 ⋂ I2 ≠ ∅ then
6:         update FGIST with I1 ⋂ I2, O1 ⋃ O2;
7:       end if
8:     end for
9:   end for
10: until no update
11: end
```

This function is based on the *greaterThan* function testing lexicographic order between itemsets (step 4). If an itemset is greater than another itemset, then their intersection, and the union of the corresponding OID lists, are used to update the tree (step 5 to 7). The optimized process of the FGIST update is performed through the three functions *HTree.Intersect*(), *HNode.Intersect*() and *HNode.Intersect*() described in the following paragraphs.

HTree.Intersect() This function, which pseudo-code is given in algorithm 13, is called on the root of the tree for the first time to update the OID list and stops after completing the updating operation. For each children under the root, it calls the *HNode.Intersect*() function and repeats the process until no changes was performed (step 4-7). These iterations generate the few frequent closed itemsets missing after FGIST creation form the SFD, and also updates the OID list in the internal node if required.

HNode.Intersect(Itemset prefix, HNode first) This function is called for the first time by the *HTree.Intersect*() function with two arguments: An empty (for initialization) itemset named *prefix* and the reference toward the first node of the branch corresponding to the first item of the processed itemset. Its pseudo-code is given in algorithm 14. First, the *Item* of the node is added to the *prefix* itemset (step 2). Then, for checking the OID list of the node, if the *Object_list* field is not empty, the *HNode.Intersect2*() function is called (step 3-5). This call will recursively collect subsequent itemsets, according to lexicographic order, and update them if required as presented in the next paragraph. After this call, it checks each children of the current node and goes inside each till a leaf node is reached (step 6-8). When a leaf node

Algorithm 13 Function: HTree.Intersect()

```
1: begin
2: boolean change ← true
3: for all child C in ROOT.HashTable do
4:    while change = true do
5:       change ← false
6:       C.Intersect({}, C)
7:    end while
8:    change ← true
9: end for
10: end
```

is reached, it returns from the leaf to the root and deletes the *item* of the node from the *prefix* itemset while unstacking items (step 9).

Algorithm 14 Function: HNode.Intersect(Itemset prefix, HNode first)

```
1: begin
2: add Item to prefix
3: if Object_list ≠ ∅ then
4:    first.Intersect2({}, prefix, Object_list, first)
5: end if
6: for all child C of prefix do
7:    C.Intersect(prefix, first)
8: end for
9: remove last item from prefix
10: return
11: end
```

HNode.Intersect2(Itemset current, Itemset I, ObjectList O, HNode first) This recursive function is called for the first time by the *HNode.Intersect*() function when updating the FGIST is required. Its pseudo-code is given in algorithm 15. When called, it receives four values corresponding to the following arguments:

- An itemset named *current* corresponding to the items gathered by the recursive calls performed till this node. This itemset is initialized to empty set at first call by the *HNode.Intersect*() function.
- An itemset named I corresponding to the itemset for which updates are performed. This itemset is initialized with *prefix* itemset at first call.
- An object list named O corresponding to the OID list of the I itemset. This list is initialized with the object list of the *prefix* itemset at first call.
- A reference to an HNode named *first* corresponding to the first node of the branch currently processed.

It updates the itemsets in nodes that are subsequent to the *first* node received as argument from the *HNode.Intersect*() function using the I itemsets and O object list also received as arguments from the *HNode.Intersect*() function.

First, the item in the *Item* field of the node is appended to the *current* itemset (step 2). If the object list of the current node is not empty and the *current* itemset is greater than the I itemsets according to the *greaterThan*() function, then the intersection between I and *current* is stored in J (step 3-4). If the result of the intersection is not empty, the union of the object list of *current* and I itemsets is stored in P (step 6). If the intersection J is equal to *current*, then update the object list of the current node with the union of P and the object list of the node (step 5-8). Otherwise, if J is equal to *prefix*, then update the object list of *prefix* with P traversing the tree from the *first* node (step 9-13). If the J is different from both *current* and *prefix*, then J is inserted in the FGIST, traversing the tree from the *first* node, with P as object list (step 14-19). Then, it recursively traverse the child nodes to perform the update with the I itemset and its O object list (step 20-22). When all itemsets of the branch have been processed, the function returns to the root of the tree, deleting the *Item* in the node from the *current* itemset (step 23).

Algorithm 15 Function: HNode.Intersect2(Itemset current, Itemset I, ObjectList O, HNode first)

```
1:  begin
2:  add Item to current
3:  if Object_list ≠ ∅ and greaterThan(current,I) then
4:    Itemset J ← current ⋂ I
5:    if J ≠ ∅ then
6:      ObjectList P ← Object_list ⋃ O
7:      if J = current then
8:        Object_list ← Object_list ⋃ P
9:      else
10:       if J = I then
11:         first.Update(J, P)
12:         change ← true
13:       end if
14:     else
15:       first.Insert(J, P)
16:       change ← true
17:     end if
18:   end if
19: end if
20: for all child C of current do
21:   C.Intersect(current, I, O, first)
22: end for
23: remove last item from current
24: return
25: end
```

4.2.5 Step 4: Pruning the Frequent Generalized Itemset Suffix-Tree

After update phase, the FGIST contains all frequent closed itemsets with supporting object list. The pruning phase described in this section prunes potential infrequent itemsets, resulting from intersections of the update phase, and non-closed itemsets in the FGIST using the *minsupport* threshold and an optimized closure checking method. The general process of this phase is described in algorithm 16.

Algorithm 16 FIST 2.0: Delete Non-Closed and Infrequent Itemsets

Input: Frequent Generalized Itemset Suffix Tree, minsupport
Output: Updated Frequent Generalized Suffix-Tree

```
1:  begin
2:  for all itemset I1 with object list O1 ≠ ∅ in FGIST do
3:    if size(O1) < minsupport then
4:      delete {I1, O1} from FGIST
5:    else
6:      for all itemset I2 with object list O2 ≠ ∅ and greaterThan(I2, I1) in FGIST do
7:        if I1 ⊂ I2 and O1 = O2 then
8:          delete {I1, O1} from FGIST
9:        end if
10:     end for
11:   end if
12: end for
13: end
```

First, the frequency of itemsets in the FGIST, identified by their non empty object list, is checked using the *minsupport* threshold and the object list size, and infrequent itemsets are pruned from the tree (step 3-4). Itemsets corresponding to internal nodes are pruned by nullifying their object list of the node and itemsets corresponding to leaf nodes are pruned by suppressing the node (and its ancestors with empty object list). In the itemset is frequent, its closeness is checked by comparing subsequent itemsets in the FGIST (step 6). If a subsequent itemset is a superset of the itemset considered and the object lists of both itemsets are equal, then the itemset considered is non-closed and thus pruned (step 7 and 8). The detailed operations of the FGIST pruning is performed through the two functions *HTree.Prune*() and *HTree.TestInclusion*() on the *HTree* root node, and the three functions *HNode.Prune*(), *HNode.TestInclusion*() and *HNode.TestInclusion2*() on the *HNode* nodes described in the following paragraphs.

HTree.Prune(int minsupport) This function is used to initialize the pruning process through the FGIST using the *minsupport* threshold value to check the frequency of itemsets. It works on the root of the FGIST and calls the *HNode.Prune*() function on the first node of each branch (step 2-4). For each node, it gives as arguments of the call an empty set, the reference to the root node and the *minsupport* threshold value (step 3).

HNode.Prune(Itemset prefix, HTree ROOT, int minsupport) This prune function is applied to internal *HNode* nodes of the FGIST and recursively traverse a branch, checking the

Algorithm 17 Function: HTree.Prune(int minsupport)

```
1: begin
2: for all child C in ROOT.HashTable do
3:    C.Prune({}, ROOT, minsupport)
4: end for
5: end
```

frequency and closeness of the itemsets corresponding to traversed nodes with non empty object lists. Its pseudo-code is given in algorithm 18. It is called by the *HNode.Prune*() function and receives three arguments: :

- An itemset named *prefix* corresponding to the items gathered by the recursive calls performed till this node. This itemset is initialized to empty set at first call by the *HNode.Prune*() function.
- A reference to the root *HTree* node of the FGIST.
- The *minimum support* threshold value.

It first appends the item in *Item* field of the node to the *prefix* itemset (step 2). Then, it tests if the current node represents the last item of an itemset, checking for the presence of an object list (step 3). If the size of the object list is less than the minimum support value, the itemset is infrequent and the object list of the node is emptied to prune this itemset (step 4-6). Otherwise, the *HTree.TestInclusion*() function is called to check the closeness of the *prefix* itemset (step 7). Then, the pruning of subsequent itemsets in the branch is performed by a recursive call of this function on children nodes (step 10-12). After pruning of subsequent nodes in the branch and if the current node is unnecessary, that is if all its subnodes were pruned and this node corresponds to an itemset that is infrequent, the current node is deleted (step 13-15). This situation is identified by the fact that the *Object_list* and *Children* fields are empty (step 13). Finally, when all subnodes have been processed, the function returns, unstacking the *Item* item of the current node from the *prefix* itemset (step 9-10).

HTree.TestInclusion(Itemset I, ObjectList O) This function is called by the *HNode.Prune*() function to check the closeness of an itemset, passed as argument with its object list, in the FGIST. It makes use of the *HNode.TestInclusion*() and *HNode.TestInclusion2*() functions described in the following paragraphs. The *HNode.TestInclusion*() and *HNode.TestInclusion2*() functions both collect subsequent itemsets of I to check closeness properties against the I itemset. These two functions are identical except that the first uses the *greaterThan*() function to compare itemsets to optimize the search space in the FGIST when possible. Its pseudo-code is given in algorithm 19. It uses two arguments:

- An itemset named I corresponding to the itemset to check. This itemset is initialized with the *prefix* itemset when called by the *HNode.Prune*() function.
- An object list named O corresponding to the OID list of the I itemset. This list is initialized with the object list of the *prefix* itemset when called.

Algorithm 18 Function: HNode.Prune(Itemset prefix, HTree ROOT, int minsupport)

```
1: begin
2: add Item to prefix
3: if Object_list ≠ ∅ then
4:   if size(Object_list) < minsupport then
5:     Object_list ← ∅
6:   else
7:     ROOT.TestInclusion(prefix, Object_list)
8:   end if
9: end if
10: for all child C of prefix do
11:   C.Prune(prefix, ROOT, minsupport);
12: end for
13: if Object_list = ∅ and Children = ∅ then
14:   delete this node
15: end if
16: remove last item from prefix
17: return
18: end
```

Fist, for each branch under the root node of the FGIST, it compares the first item of the I itemset and the item in the *Item* field of the first node of the branch (step 2). If they are identical, then it calls the *HNode.TestInclusion*() function (step 3-4). In this case, itemsets represented in this branch of the FGIST begin with the same item as the I itemset. This means that we must check whether the itemsets represented in this branch of the FGIST are lexicographically greater than I or not before testing closeness property. Otherwise, if the *Item* item is greater than the first item of the I itemset, it calls the *HNode.TestInclusion2*() function (step 5-9). In this case, there is no need to check if the itemsets represented in this branch of the FGIST are lexicographically greater than I. Branches under the root node of the FGIST that contain itemsets beginning with an item lower than the first item of the I itemset do not need to be tested, thanks to their ordering in the FGIST. This is an important optimization for the efficiency of the pruning phase.

HNode.TestInclusion(Itemset current, Itemset I, ObjectList O) This function, called by the *HTree.TestInclusion*() function, is used to check closure property of itemsets in the branch of the FGIST where represented itemsets begin by the same item as the tested itemset received as argument. The recursive function, which pseudo-code is given in algorithm 20, uses three arguments:

- An itemset *current* corresponding to the items gathered by the recursive calls performed till this node. This itemset is initialized to empty set at first call by the *HTree.TestInclusion*() function.
- An itemset I corresponding to the itemset to check. This itemset is initialized with the *prefix* itemset at first call.

Algorithm 19 Function: HTree.TestInclusion(Itemset I, ObjectList O)

```
1: begin
2: for all child C in ROOT.Hashtable do
3:   if C.item = I[1] then
4:     C.TestInclusion({}, I, O)
5:   else
6:     if C.item > I[1] then
7:       C.TestInclusion2({}, I, O)
8:     end if
9:   end if
10: end for
11: return
12: end
```

- An object list named O corresponding to the OID list of the I itemset. This list is initialized with the object list of the *prefix* itemset at first call.

First, it appends the item in the *Item* field of the node to the *current* itemset (step 2). Then, if verifies if the current node corresponds to an itemset for which the closure property checking is needed, that is, if the object list of the current node is not empty and *current* is greater than I (step 3). If true, it checks whether the *current* itemset is a subset of the I itemset, and if the object list of the current node is equal to the object list O of I (step 4). In such a case, the *current* itemset is not closed and its object list is set to null, which indicates the deletion of the *current* itemset from the tree (step 5). After these operations, the function recursively traverse subnodes in the branch for checking subsequent itemsets (step 8-10). After checking of subsequent nodes in the branch, if the current node is unnecessary, i.e., if all its subnodes were pruned and this node corresponds to an itemset that is non-closed, the current node is deleted (step 11-13). This situation is identified by the fact that the *Object_list* and *Children* fields are empty (step 11). Finally, when all subnodes have been processed, the function returns, unstacking the *Item* item of the current node from the *current* itemset (step 9-10).

HNode.TestInclusion2(Itemset current, Itemset I, ObjectList O) This function, called by the *HTree.TestInclusion*() function, is used to check closure property of itemsets in the branch of the FGIST where represented itemsets begin with a greater item, given lexicographic order on their identifiers, than the first item of the tested itemset received as argument. This function is identical to the *HNode.TestInclusion*() function described above except for step 3 that do not use the *greaterThan*() function. Hence, given the fact that the first item of itemsets in this branch is greater than the first item of the tested I itemset, this test is not required. Theoretical and experimental studies have shown that this optimization is important from a computational efficiency viewpoint. The pseudo-code of this function is given in algorithm 21.

4.2.6 Step 5: Generating Conceptual Knowledge Pattern Information

This is the last phase of the second algorithmic version of the FIST approach. During this phase, conceptual bi-clusters, generators and bases of conceptual association rules are generated using the frequent closed patterns (FCP), constituted of frequent closed itemsets and

Algorithm 20 Function: HNode.TestInclusion(Itemset current, Itemset I, ObjectList O)

```
1: begin
2: add Item to current
3: if Object_list ≠ ∅ and greaterThan(current,I) then
4:    if current ⊂ I and Object_list = O then
5:       Object_list ← ∅
6:    end if
7: end if
8: for all child C of this do
9:    C.TestInclusion(current, I, O)
10: end for
11: if Object_list = ∅ and Children = ∅ then
12:    delete node current
13: end if
14: remove last item from current
15: return
16: end
```

associated object lists, in the FGIST. Unlike in first algorithmic version, the different pattern outputs are extracted from the FGIST in two different steps. The first step, discussed in section 4.2.6.1, directly extract bi-clusters and generators from the FCP. The second step, presented in section 4.2.6.2, extracts different rule bases from the FCP using both generators and FCP. This second algorithmic version can extract two bases of approximate conceptual association rules: The proper base, containing approximate rules between two frequent closed itemsets, and the structural base, containing the min-max rules between generators and frequent closed itemsets. Depending on the application context, one or the other base can be generated, in association with the base of exact conceptual association rules. Compared with traditional ARM approaches, these association rules provide more information to the end-user as the list of objects supporting each rule is generated instead of only the support of the rule. Moreover, if these information are present in the initial dataset, user-understandable identifiers (names, labels, codes, etc.) of objects and variable values can be used to construct user-friendly knowledge pattern sets.

4.2.6.1 Extracting Conceptual Bi-clusters and Generators

Algorithm 22 gives the pseudo-code of the conceptual bi-cluster creation and generator identification. These patterns are extracted from the FGIST containing all frequent closed itemsets with their list of supporting objects in the dataset.

During the process, the frequent closed patterns in the FGIST are considered in increasing order of their itemset size (step 3-29). For each frequent closed pattern F (step 4-30), a bi-cluster is created in the BIC set, with the itemset corresponding to the set of items sequentially collected while traversing the branch from the root to this node as extent and the object list of the node as intent (step 4). These extent and intent represent a maximal set of related rows and columns, respectively. Then, generators of each FCI are created in a level-wise manner (step 7-28). The subsets of the FCIs are created in increasing order of their size (step 8). For

Algorithm 21 Function: HNode.TestInclusion2(Itemset current, Itemset I, ObjectList O)

```
1: begin
2: add item to current;
3: if Object_list ≠ ∅ then
4:   if current ⊂ I and Object_list = O then
5:     Object_list ← ∅
6:   end if
7: end if
8: for all child C of this do
9:   C.TestInclusion(current, I, O)
10: end for
11: if Object_list = ∅ and children = ∅ then
12:   delete node current;
13: end if
14: remove last item from current
15: return
16: end
```

each subset, its presence among the list of generators already found (step 11-13) or among the FCIs (step 14-18) is tested. If both tests fail, a new entry is created in the GEN set for generators of the FCI (step 20-24) and the boolean variable found_gen is set to true to avoid testing subsets of greater size (step 21). This process relies on the property that generators constitute an ideal of order, and ,consequently, that all generators of a frequent closed itemset have the same size. If all subsets were proceeded and no generator was found, then the frequent closed itemset is itself its own only generator (step 26-28). As a final operation (step 31), items in the generators and bi-clusters are mapped to their original value in the source dataset to simplify their interpretation by the end-user. To limit the number of extracted patterns, objective or subjective measures for selecting patterns according to the application objectives and requirements can easily be integrated in the process.

4.2.6.2 Generating Bases of Association Rules

This section presents the generation of bases of conceptual association rules using the GEN set of frequent generators and the frequent closed patterns (FCP) in the FGIST. The pseudo-code description of this step is presented in algorithm 23.

Three sets are generated during this step: The base of min-max exact conceptual association rules (AR_E) and, either, or both, the structural (AR_SB) and the proper (AR_PB) bases for approximate conceptual association rules. These bases are minimal, regarding their size, non-redundant covers defined according to different criteria on the structure of rules. Informally, these bases are minimal means that if one rule is delete from a base, all classical association rules cannot be deduced from the base, with support and confidence. They are non-redundant means that no rule in a base can be deduced, with support and confidence, from the other rules of the base. The base of min-max exact rules contains rules between a generator (minimal set) in antecedent and the frequent closed itemsets that is its closure (maximal set) in consequent. The base of min-max approximate rules contains rules between a generator in antecedent and

Algorithm 22 FIST 2.0: Finding Conceptual Bi-clusters and Generators

```
1: begin
2: GEN, BIC ← ∅
3: for all FCP F in increasing order of the itemset sizes in FGIST do
4:   insert{F.Itemset, F.Object_List} in BIC
5:   found_gen ← false
6:   gen_size ← 1
7:   while found_gen = false and gen_size < |F.Itemset| do
8:     SUB ← subsets of F.Itemset of size gen_size
9:     for all subset S in SUB do
10:      not_gen ← false
11:      for all G.Generator in GEN do
12:        if S = G.Generator then not_gen ← true
13:      end for
14:      if not_gen = false then
15:        for all C∈FCP preceding F∈FCP do
16:          if S ⊆ C.Itemset then not_gen ← true
17:        end for
18:      end if
19:      if not_gen = false then
20:        insert{S, F} in GEN
21:        found_gen ← true
22:      end if
23:    end for
24:    gen_size = gen_size + 1
25:  end while
26:  if found_gen = false then
27:    insert{F.Itemset, F.Itemset} in GEN
28:  end if
29: end for
30: map items in BIC, GEN to dataset values
31: return(BIC, GEN)
32: end
```

frequent closed itemsets that are supersets of its closure in consequent. The proper base of approximate rules contains rules between two frequent closed itemsets related by inclusion. A detailed discussion on the rule bases is given in the state of the art part of the thesis.

The bases are created by considering each frequent generator in GEN successively (step 3-19). First, the (frequent closed) itemsets of the FCP in the FGIST are compared to the closure of the generator (step 4-5). If they are identical (step 5), an exact rule is created in the AR_E set with the generator as antecedent and the difference between the itemset and the generator as consequent. This rule has a support equals to the size of the *Object_ list* of the itemset and a confidence equal to 1, and its associated supporting object list is *Object_ list* (step 8-9). Otherwise, if the closure of the generator is a subset of the itemset (step 13), a min-max approximate rule is created in the AR_SB set with the generator as antecedent and the difference between the itemset and the generator as consequent. This rule has a support equals to the size of the *Object_ list* of the itemset and a confidence equal to the ratio between the size of the *Object_ list* of the of the generator's closure and the size of the itemset *Object_ list*,

Algorithm 23 FIST 2.0: Generating Bases of Conceptual Association Rules

```
1: begin
2: AR_E, AR_SB, AR_PB ← ∅
3: for all G.Generator in GEN do
4:   for all F.Itemset of FCP in FGIST do
5:     if G.Closure = F.Itemset then
6:       [[ Min-Max Exact Conceptual Rules ]]
7:       if G.Generator ≠ F.Itemset then
8:         create rule r: {G.Generator ⇒ F.Itemset \ G.Generator, sup(r) =
           |F.Object_list|, conf(r)=1, F.Object_list}
9:         insert r into AR_E
10:      end if
11:    else
12:      [[ Min-Max Approximate Conceptual Rules ]]
13:      if G.Closure ⊂ F.Itemset then
14:        create rule r: {G.Generator → F.Itemset \ G.Generator, sup(r) =
           |F.Object_list|, conf(r)=|F.Object_list|/|G.Object_list|, F.Object_list}
15:        insert r into AR_SB
16:      end if
17:    end if
18:  end for
19: end for
20: [[ Proper Approximate Conceptual Rules ]]
21: for all F_i.Itemset in FCP do
22:   for all F_j.Itemset in FCP where F_j.Itemset ⊃ F_i.Itemset do
23:     create rule r: {F_i.Itemset → F_j.Itemset \ F_i.Itemset, sup(r)=|F_j.Object_list|,
        conf(r)=|F_j.Object_list|/|F_i.Object_list|, F_j.Object_list}
24:     insert r into AR_PB
25:   end for
26: end for
27: map items in AR_E, AR_SB, AR_PB to dataset values
28: return(AR_E, AR_SB, AR_PB)
29: end
```

and its supporting OID list is the *Object_ list* of the itemset (step 14-15). The generation of the proper base is performed by comparing pairs of frequent closed itemsets in the FCP of the FGIST (step 21-26). If one is a subset of the other, a proper approximate rule is created in the AR_PB set with the subset as antecedent and the difference between the superset and the subset as consequent. This rule has a support equals to the size of the *Object_ list* of the superset and a confidence equal to the ratio between the size of the *Object_ list* of the superset and the size of the *Object_ list* of the subset. Its supporting OID list is equal to the *Object_ list* of the superset among them (step 23). The last step of the algorithm consists to map the item identifiers in rules to their original name in the source dataset (step 27).

Chapter 5

Examples and Properties

Contents

In this chapter, we present examples of executions of the two algorithmic versions of FIST and the properties on which the FIST approach relies. In section 5.1, the trace of the execution on the example dataset of the first algorithmic version of FIST is detailed. The execution of the second algorithmic version of FIST for the same dataset and parameters is presented in section 5.2. The properties of the approach and algorithmic versions of FIST are discussed in section 5.3.

The example database D1 used for the example executions is given in binary matrix and transactional formats in table 5.1. It contains six objects identified by their OID: {*1, 2, 3, 4, 5, 6*} and five binary variables: {A, B, C, D, E}. In binary matrix format, if an object possesses a variable, the cell corresponding to the object row and the variable column contains a 1; otherwise it contains a 0. In the transactional formal, if an object possesses a variable, the identifier of the variable is present in the row corresponding to the object.

5.1 FIST 1.0 Execution

This section depicts the execution of the first algorithmic version of FIST. Each of the following subsections presents separately one of the three phases of the algorithm.

Object	A	B	C	D	E
1	1	0	1	1	0
2	1	1	1	0	1
3	0	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1
6	0	1	1	0	1

A. Binary Matrix Representation

Object	Items
1	A C D
2	A B C E
3	B C E
4	B E
5	A B C E
6	B C E

B. Transactional Representation

Table 5.1: Example Database D1

5.1.1 Phase 1: Creating Sorted Frequent Database

The first phase is the preprocessing of the dataset to generate a minimal representation of data, the Sorted Frequent Database. First, the Item Table is created by scanning the dataset to identify the *variable* = *value* pairs while counting their number of appearances (support). A unique number (item) is then assigned to each pair. The Item Table for database D1 is presented in table 5.2. The item numbers are then used to generate the Number Table in which each relation between an object and a *variable* = *value* pair is represented by the presence of the corresponding item on the row representing the object. The Number Table of database D1 is given in table 5.3.

Items	Attribute=Value	Support
1	A=1	3
2	C=1	5
3	D=1	1
4	B=1	5
5	E=1	5

Table 5.2: Item Table of Database D1

Object	Items
1	1 2 3
2	1 4 2 5
3	4 2 5
4	4 5
5	1 4 2 5
6	4 2 5

Table 5.3: Number Table of Database D1

Infrequent items, according to the *minsupport* value, are then deleted from the Item Table and the remaining items are sorted in increasing order of their support count and assigned a new item number according to this ordering. This resulting Frequent Item Table for *minsupport* = 2/6 is presented in table 5.4. A mapping between the initial item numbers and these new item numbers is performed on the Number Table to generate the Sorted Frequent Database. The Sorted Frequent Database for database D1 and *minsupport* = 2/6 is presented in table 5.5.

For further comparison operations between itemsets, the lexicographic order on item numbers will be used, which is equivalent to comparing items on their support. Hence, amid the series of item numbers, the smallest values correspond to items with the smallest support counts in the dataset. By construction, if two data values have the same support, then their order of appearance in the dataset will determine their item number order.

Frequent Items	Support	Sorted Item
1	3	1
2	5	2
4	5	3
5	5	4

Table 5.4: Frequent Item Table of Database D1 for *minsupport* = 2/6

Object	Items
1	1 2
2	1 2 3 4
3	2 3 4
4	3 4
5	1 2 3 4
6	2 3 4

Table 5.5: Sorted Frequent Database of Database D1 for *minsupport* = 2/6

5.1.2 Phase 2: Mining Frequent Closed Itemsets

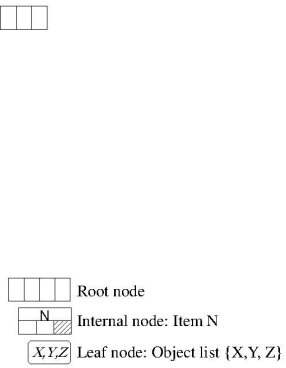
This section presents examples of the frequent generalized itemset suffix-tree construction from database D1 and the finding closed itemsets for *minsupport* = 2/6.

5.1.2.1 Step 1: Building Frequent Generalized Itemset Suffix-Tree

Objects of the Sorted Frequent Database are accessed sequentially to build the FGIST. The trace of the row-wise development of the tree is depicted in table 5.6. The *Step* column shows read operations of rows in the dataset. The *Vector* column shows the item vector of the rows read in the dataset. The *OID* column shows the object identifier (sequential number) of the row. The *Suffixes* column shows the list of suffixes generated for the object. The successive calls of the *match*() and *buildEdge*() functions are shown with the suffix passed as argument. The graphic given for each call shows the edge and node creations and updates in the FGIST.

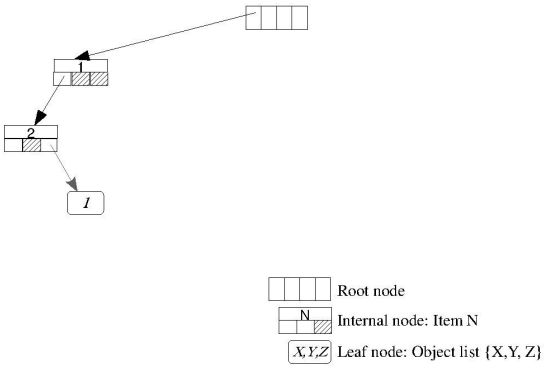
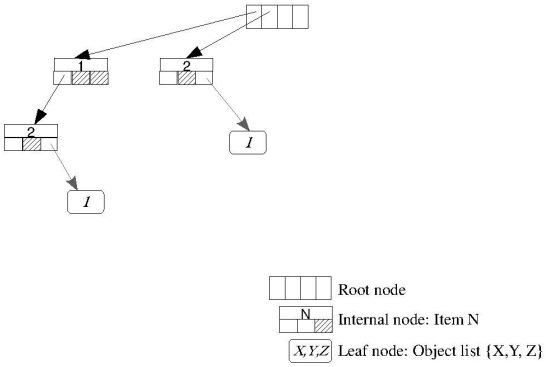
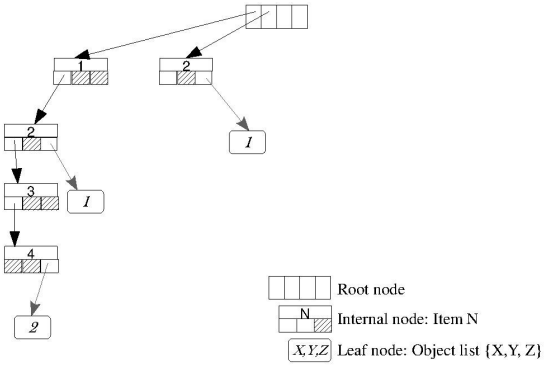
Table 5.6: Frequent Generalized Itemset Suffix-Tree Creation for Example Database D1

Step	Vector	OID	Suffixes
Create root			



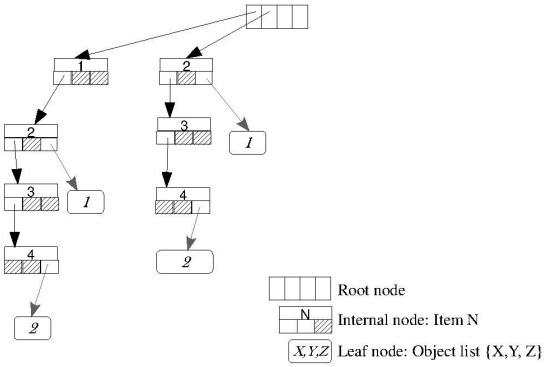
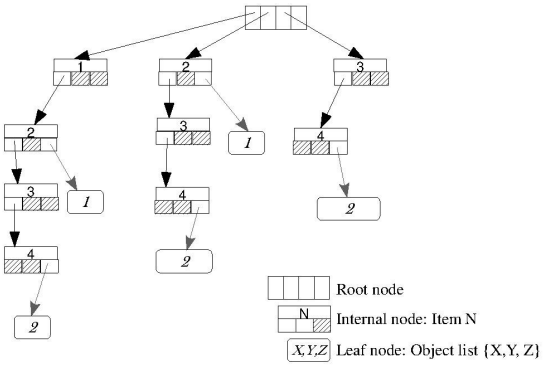
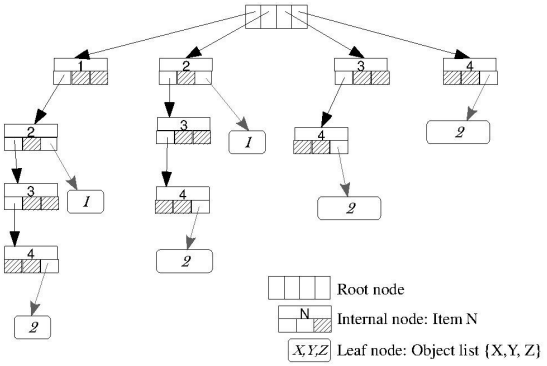
Continued on next page...

Table 5.6 – *Continued*

Step	**Vector**	**OID**	**Suffixes**
Read row 1:	1 2	*1*	12*1*, 2*1*
Call functions *match*() and *buildEdge*() with suffix 12*1*			
			
Call functions *match*() and *buildEdge*() with suffix 2*1*			
			
Read row 2:	1 2 3 4	*2*	1234*2*, 234*2*, 34*2*, 4*2*
Call functions *match*() and *buildEdge*() with suffix 1234*2*			
			

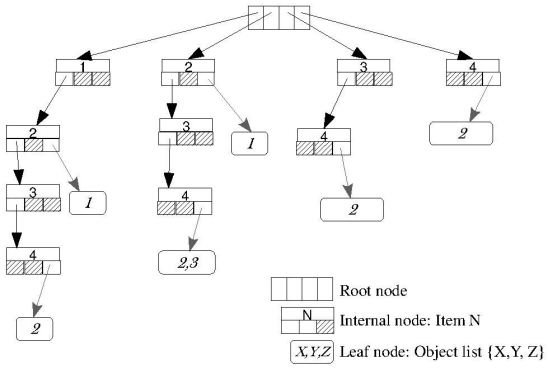
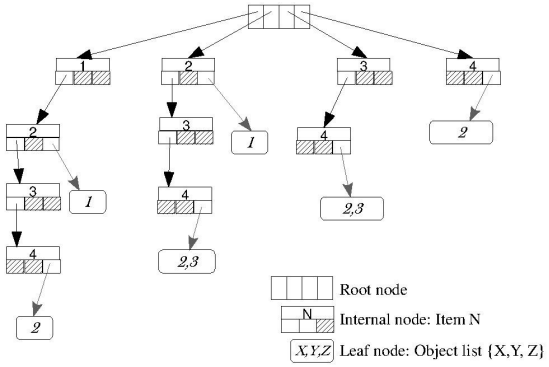
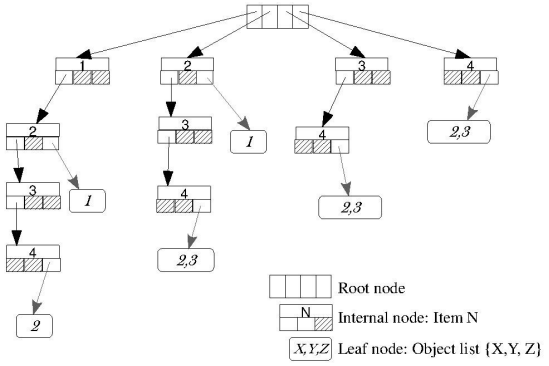
Continued on next page...

Table 5.6 – *Continued*

Step	Vector	OID	Suffixes
Call functions *match*() and *buildEdge*() with suffix 234*2*			
			
Call functions *match*() and *buildEdge*() with suffix 34*2*			
			
Call functions *match*() and *buildEdge*() with suffix 4*2*			
			

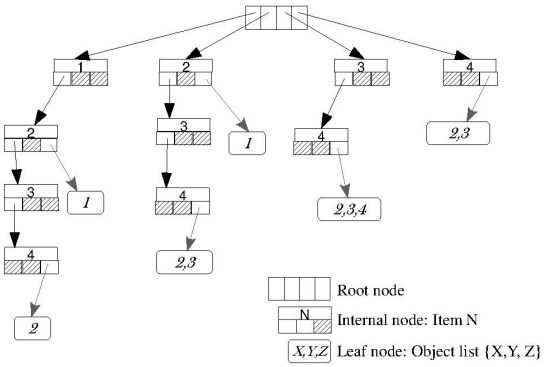
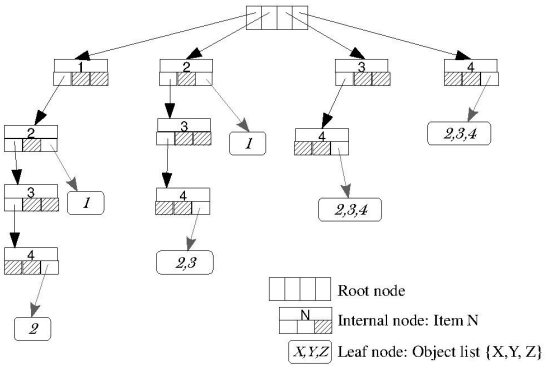
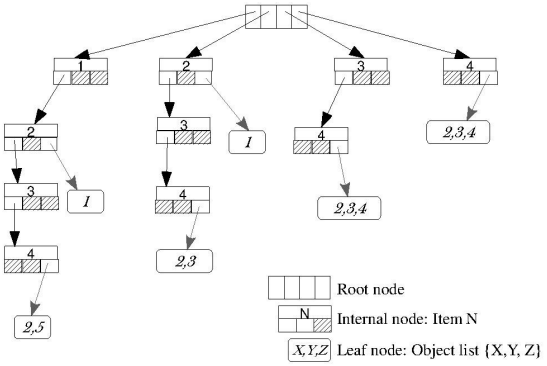
Continued on next page...

Table 5.6 – *Continued*

Step	**Vector**	**OID**	**Suffixes**
Read row 3:	2 3 4	*3*	234*3*, 34*3*, 4*3*
Call functions *match*() and *buildEdge*() with suffix 234*3*			
			
Call functions *match*() and *buildEdge*() with suffix 34*3*			
			
Call functions *match*() and *buildEdge*() with suffix 4*3*			
			

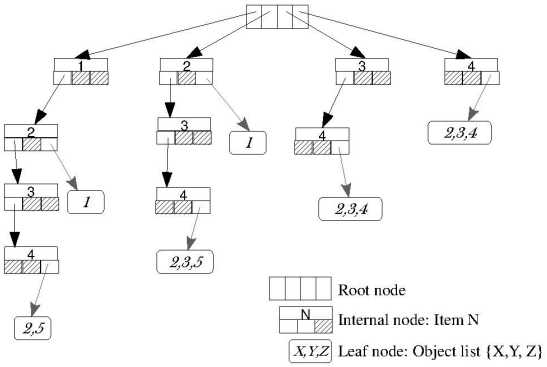
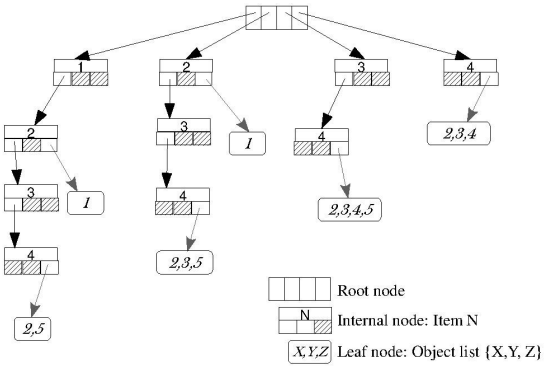
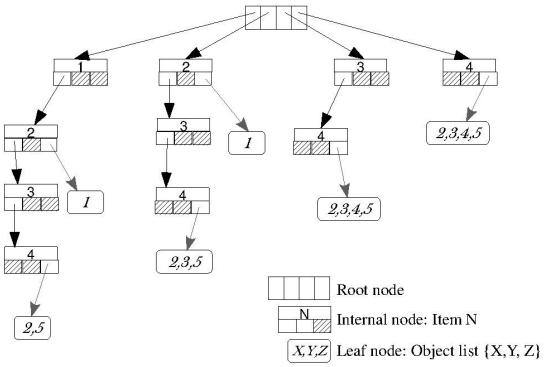
Continued on next page...

Table 5.6 – *Continued*

Step	**Vector**	**OID**	**Suffixes**
Read row 4:	3 4	*4*	34*4*, 4*4*
Call functions *match*() and *buildEdge*() with suffix 34*4*			
			
Call functions *match*() and *buildEdge*() with suffix 4*4*			
			
Read row 5:	1 2 3 4	*5*	1234*5*, 234*5*, 34*5*, 4*5*
Call functions *match*() and *buildEdge*() with suffix 1234*5*			
			

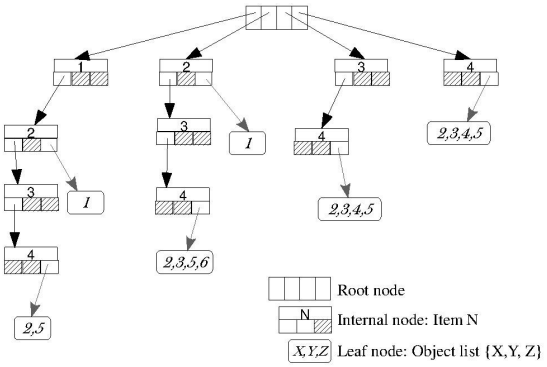
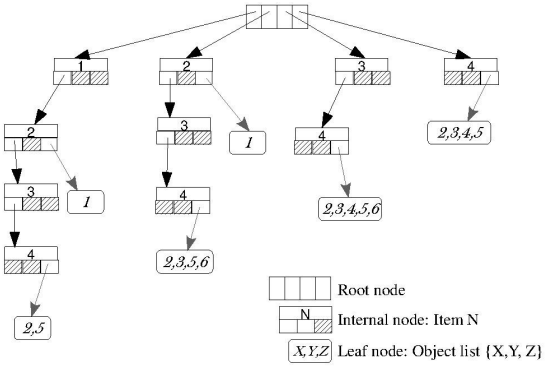
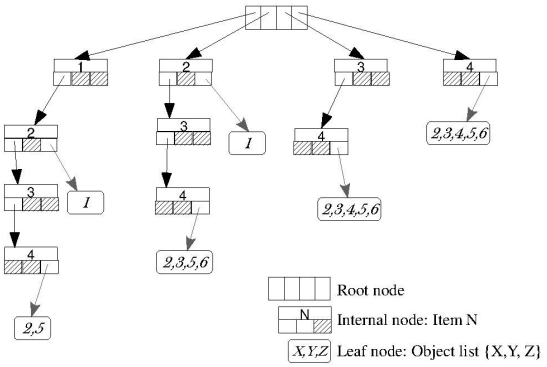
Continued on next page...

Table 5.6 – *Continued*

Step	Vector	OID	Suffixes
Call functions *match*() and *buildEdge*() with suffix 2345			
			
Call functions *match*() and *buildEdge*() with suffix 345			
			
Call functions *match*() and *buildEdge*() with suffix 45			
			

Continued on next page...

Table 5.6 – *Continued*

Step	**Vector**	**OID**	**Suffixes**
Read row 6:	2 3 4	*6*	234*6*, 34*6*, 4*6*
Call functions *match*() and *buildEdge*() with suffix 234*6*			
			
Call functions *match*() and *buildEdge*() with suffix 34*6*			
			
Call functions *match*() and *buildEdge*() with suffix 4*6*			
			

5.1.2.2 Step 2: Extracting Frequent Closed Patterns

After creating the FGIST, the itemsets are collected from the tree to generate the set of all frequent closed patterns (FCP). First task of this step consists to collect frequent patterns from the FGIST. The process starts from the root of the tree and then traverse each branch to a leaf. The itemsets collected during this traversal, identified by their object list, are stored in the output FCP set.

The trace of the branch-wise execution of this recursive process is depicted in table 5.7. The *Current Node(I)* column shows the value (root or item number) in the node of the tree presently processed. When a leaf has been reached, and the process starts returning, this column contains *Return*. The *Child Nodes* column shows the list of items in child nodes of the actual node. When a child node has already been processed, it is marked with a check-mark in the list. The *Object List* column shows the list of OIDs of the actual node. The *Item List* column shows the itemset corresponding to the path between the root and the actual node. The *Output* double column shows the identified frequent patterns. The *Itemset* sub-column shows the frequent itemset of the frequent pattern. The *Objects* sub-column shows the list of objects supporting the frequent itemset of the frequent pattern.

Table 5.7: Mining Frequent Patterns from FGIST for Example Database D1

Current Node(I)	**Child Nodes**	**Object List**	**Item list**	**Output**	
				Itemset	**Objects**
Begin					
Root	1, 2, 3, 4				
1	2	ϕ	1		
2	3	*1*	12	12	*1*
3	4	ϕ	123		
4	ϕ	*2, 5*	1234	1234	*2, 5*
Return					
3	✓4				
2	✓3				
1	✓2				
Root	✓1, 2, 3, 4				
2	3	*1*	2	2	*1*
3	4	ϕ	23		
4	ϕ	*2, 3, 5, 6*	234	234	*2, 3, 5, 6*
Return					
3	✓4				
2	✓3				
Root	✓1, ✓2, 3, 4				
3	4	ϕ	3		
4	ϕ	*2, 3, 4, 5, 6*	34	34	*2, 3, 4, 5, 6*
Return					
3	✓4				
Root	✓1, ✓2, ✓3, 4				
4	ϕ	*2, 3, 4, 5, 6*	4	4	*2, 3, 4, 5, 6*

Continued on next page...

Table 5.7 – *Continued*

Current Node(I)	Child Nodes	Object List	Item list	Output	
				Itemset	Objects
Return Root	✓1, ✓2, ✓3, ✓4				
End					

Final Table	12	*1*
	1234	*2, 5*
	2	*1*
	234	*2, 3, 5, 6*
	34	*2, 3, 4, 5, 6*
	4	*2, 3, 4, 5, 6*

Once frequent patterns have been collected from the FGIST, infrequent and non-closed ones are first identified. For this checking, the object lists associated to frequent itemsets in frequent patterns are compared according to inclusion and, if an itemset is a subset of another itemset and their object lists are identical, then the subset itemset is not closed and suppressed from the FCP table.

The trace of the execution of this iterative process is shown in table 5.8. The *Set* column shows the number of the step in the present iteration that is also the order number of the frequent pattern in the FCP table processed during this step. The *Itemset* column shows the itemset in the frequent pattern processed during the present step. The *Object list* column shows the object list supporting the itemset of the frequent pattern processed during the present step. The *Comparison with* column shows the order number in FCP (*Set* value) of the itemset that is compared with the frequent itemset processed during the present step. The *Is subset?* column shows the result of the test checking if the frequent itemset processed during the present step is a subset of the itemset it is compared with (*Comparison with* value). The *Equal objects?* column shows the result of the test checking if the object list of the frequent itemset processed during the present step is identical to the one of the itemset it is compared with (*Comparison with* value). The *Delete pattern?* column indicates if the frequent pattern presently processed is deleted, i.e., if infrequent or non-closed.

Table 5.8: Reduction of Frequent Patterns using Inclusions for Example Database D1

Set	Itemset	Object list	Is subset?	Comparison with	Equal objects?	Delete pattern?
Begin						
First iteration						
1	12	*1*				
2	1234	*2, 5*	No	1	No	
3	2	*1*	Yes	1	Yes	Yes
4	234	*2, 3, 5, 6*	No	1	No	
5	34	*2, 3, 4, 5, 6*	No	1	No	
6	4	*2, 3, 4, 5, 6*	No	1	No	

Continued on next page...

Table 5.8 – *Continued*

Set	**Itemset**	**Object list**	**Is subset?**	**Comparison with**	**Equal objects?**	**Delete pattern?**
Second iteration						
1	12	*1*	Yes	2	No	
2	1234	*2, 5*				
3	~~2~~	~~*1*~~	-	-	-	-
4	234	*2, 3, 5, 6*	Yes	2	No	
5	34	*2, 3, 4, 5, 6*	Yes	2	No	
6	4	*2, 3, 4, 5, 6*	Yes	2	No	
Third iteration						
1	12	*1*	No	4	No	
2	1234	*2, 5*	No	4	No	
3	~~2~~	~~*1*~~	-	-	-	-
4	234	*2, 3, 5, 6*				
5	34	*2, 3, 4, 5, 6*	Yes	4	No	
6	4	*2, 3, 4, 5, 6*	Yes	4	No	
Fourth iteration						
1	12	*1*	No	5	No	
2	1234	*2, 5*	No	5	No	
3	~~2~~	~~*1*~~	-	-	-	-
4	234	*2, 3, 5, 6*	No	5	No	
5	34	*2, 3, 4, 5, 6*				
6	4	*2, 3, 4, 5, 6*	Yes	5	Yes	
Fifth iteration						
1	12	*1*				
2	1234	*2, 5*				
3	~~2~~	~~*1*~~	-	-	-	-
4	234	*2, 3, 5, 6*				
5	34	*2, 3, 4, 5, 6*				
6	~~4~~	~~*2, 3, 4, 5, 6*~~				
End						

1	12	*1*	Final Table
2	1234	*2, 5*	
3	234	*2, 3, 5, 6*	
4	34	*2, 3, 4, 5, 6*	

The last step of this phase consists to identify the few missing frequent closed patterns, and when necessary delete infrequent ones, in the FCP table. These patterns are identified by performing intersection operations between frequent closed itemsets in the FCP table that have items in common.

The trace of the execution of this process for example database D1 is depicted in table 5.9. The iterative process performed is stopped when no new closed pattern is generated during an iteration. A new closed pattern is identified by performing intersection between a pair of

frequent closed itemsets in the FGIST. The supporting object list of a new closed itemset is the union of the object lists of the two itemsets involved in the intersection operation. the final FCP table obtained contains the complete set of frequent patterns that are used in the next phase to generate conceptual clusters, generators and rules. The *Set* column shows the order number of the frequent pattern in the FCP table processed during the actual step. The *Itemset* column shows the itemset in the frequent closed pattern processed during the actual step. The *Object list* column shows the object list supporting the itemset of the frequent closed pattern processed during the actual step. The *Patterns involved* column shows the order number in FCP (*Set* column) of the frequent closed patterns compared during the present step. The *Comparison results* multi-column shows the results of the comparison of the frequent closed patterns involved (*Patterns involved* column) during the present step. The *Intersection* column shows the items that itemsets of the frequent closed patterns involved have in common. The *Union* column shows the list of objects supporting the intersection between the itemsets involved. The *Add/Replace?* column shows the operation performed when a new information is found. *Add* means that a new entry in FCP is created and *Replace* means that the object list of the itemset resulting from intersection (*Intersection* column) is replaced by the object list computed by union (*Union* column).

Table 5.9: Mining Frequent Closed Patterns from FGIST for Example Database D1

Set	**Itemset**	**Object list**	**Patterns involved**	**Comparison results**		
				Intersection	**Union**	**Add/Replace?**
Begin						
First iteration						
First loop						
1	12	*1*				
2	1234	*2, 5*	1, 2	12	*1, 2, 5*	Replace
3	234	*2, 3, 5, 6*	1, 3	2	*1, 2, 3, 5, 6*	Add
4	34	*2, 3, 4, 5, 6*	1, 4	ϕ	-	-
Second loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*	2, 3	234	*2, 3, 5, 6*	-
4	34	*2, 3, 4, 5, 6*	2, 4	34	*2, 3, 4, 5, 6*	-
5	2	*1, 2, 3, 5, 6*	2, 5	2	*1, 2, 3, 5, 6*	-
Third loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*				
4	34	*2, 3, 4, 5, 6*	3, 4	34	*2, 3, 4, 5, 6*	-
5	2	*1, 2, 3, 5, 6*	3, 5	2	*1, 2, 3, 5, 6*	-
Fourth loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*				

Continued on next page...

Table 5.9 – *Continued*

Set	**Itemset**	**Object list**	**Patterns involved**	**Comparison results**		
				Intersection	**Union**	**Add/Replace?**
4	34	*2, 3, 4, 5, 6*				
5	2	*1, 2, 3, 5, 6*	4, 5	ϕ		
Second iteration						
First loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*	1, 2	12	*1, 2, 5*	-
3	234	*2, 3, 5, 6*	1, 3	2	*1, 2, 3, 5, 6*	-
4	34	*2, 3, 4, 5, 6*	1, 4	ϕ	-	-
5	2	*1, 2, 3, 5, 6*	1, 5	2	*1, 2, 3, 5, 6*	-
Second loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*	2, 3	234	*2, 3, 5, 6*	-
4	34	*2, 3, 4, 5, 6*	2, 4	34	*2, 3, 4, 5, 6*	-
5	2	*1, 2, 3, 5, 6*	2, 5	2	*1, 2, 3, 5, 6*	-
Third loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*				
4	34	*2, 3, 4, 5, 6*	3, 4	3, 4	*2, 3, 4, 5, 6*	-
5	2	*1, 2, 3, 5, 6*	3, 5	2	*1, 2, 3, 5, 6*	-
Fourth loop						
1	12	*1, 2, 5*				
2	1234	*2, 5*				
3	234	*2, 3, 5, 6*				
4	34	*2, 3, 4, 5, 6*				
5	2	*1, 2, 3, 5, 6*	4, 5	ϕ		-
No new closed pattern found						
Stop the iterations						

5.1.3 Phase 3: Finding Conceptual Bi-clusters, Generators and Rules

The final phase of the approach consists to generate bases of conceptual association rules and conceptual bi-clusters, along with frequent generators of the closed itemsets, from the frequent closed patterns in FCP. The fist operation is to sort frequent patterns in increasing order of the itemset sizes. Then, frequent closed patterns are considered successively to identify bi-clusters, the generators of its itemset and the rules, from these generators and the frequent closed itemset, in this order. Conceptual association rules are generated into two distinct sets: The base for exact rules and the base for approximate rules.

The trace of the execution of this process is depicted in table 5.10. For each frequent closed pattern in the FCP set, are depicted the bi-cluster creations in the BIC set, generator identifications in the GEN set and rule generations in the AR_Exact and AR_Approx sets.

New inserted elements in these sets are depicted in blue color characters. The M variable shows the size of the frequent closed itemset processed. The SUB set shows the list of subsets processed for the frequent closed itemset. The K variable shows the total number of subsets.

Table 5.10: Output Patterns for Example Database D1

Begin			
Sort on size			
Order #	Closed set	Object Ids	
1	2	*1, 2, 3, 5, 6*	
2	12	*1, 2, 5*	
3	34	*2, 3, 4, 5, 6*	
4	234	*2, 3, 5, 6*	
5	1234	*2, 5*	
Initialize output sets BIC, GEN, AR_Exact, AR_Approx to ϕ			

First closed pattern		
BIC		
Rows	Columns	
2	*1, 2, 3, 5, 6*	
M = 1		

Second closed pattern		
BIC		
Rows	Columns	
2	*1, 2, 3, 5, 6*	
12	*1, 2, 5*	
M = 2 SUB = [{1}, {2}] K = 2		

Update generators			
GEN			
Itemset	Object List	Closure	
1	*1, 2, 5*	12	

Generate rules					
Rules set	Antecedent	Consequent	Support	Confidence	Object list
AR_Exact	{1}	{2}	3	1	*1, 2, 5*
AR_Approx	{2}	{1}	3	0.6	*1, 2, 5*

Third closed pattern		
BIC		
Rows	Columns	
2	*1, 2, 3, 5, 6*	

Continued on next page...

Table 5.10 – *Continued*

12	*1, 2, 5*	
34	*2, 3, 4, 5, 6*	
M = 2 SUB = [{3}, {4}] K = 2		

Update generators			
GEN			
Itemset	Object List	Closure	
1	*1, 2, 5*	12	
3	*2, 3, 4, 5, 6*	34	
4	*2, 3, 4, 5, 6*	34	

Generate rules					
Rules set	Antecedent	Consequent	Support	Confidence	Object list
AR_Exact	{1}	{2}	3	1	*1, 2, 5*
	{3}	{4}	5	1	*2, 3, 4, 5, 6*
	{4}	{3}	5	1	*2, 3, 4, 5, 6*
AR_Approx	{2}	{1}	3	0.6	*1, 2, 5*

Fourth closed pattern		
BIC		
Rows	Columns	
2	*1, 2, 3, 5, 6*	
12	*1, 2, 5*	
34	*2, 3, 4, 5, 6*	
234	*2, 3, 5, 6*	
M = 3 SUB = [{2}, {3}, {4}, {2,3}, {2,4}, {3,4}] K = 6		

Update generators			
GEN			
Itemset	Object List	Closure	
1	*1, 2, 5*	12	
3	*2, 3, 4, 5, 6*	34	
4	*2, 3, 4, 5, 6*	34	
23	*2, 3, 5, 6*	234	
24	*2, 3, 5, 6*	234	

Generate rules					
Rules set	Antecedent	Consequent	Support	Confidence	Object list
AR_Exact	{1}	{2}	3	1	*1, 2, 5*

Continued on next page...

Table 5.10 – *Continued*

	{3}	{4}	5	1	*2, 3, 4, 5, 6*
	{4}	{3}	5	1	*2, 3, 4, 5, 6*
	{2,4}	{3}	4	1	*2, 3, 5, 6*
	{2,3}	{4}	4	1	*2, 3, 5, 6*
AR_Approx	{2}	{1}	3	0.6	*1, 2, 5*
	{2}	{3,4}	4	0.8	*2, 3, 5, 6*
	{3,4}	{2}	4	0.8	*2, 3, 5, 6*
	{3}	{2,4}	4	0.8	*2, 3, 5, 6*
	{4}	{2,3}	4	0.8	*2, 3, 5, 6*

Fifth closed pattern

BIC

Rows	Columns
2	*1, 2, 3, 5, 6*
12	*1, 2, 5*
34	*2, 3, 4, 5, 6*
234	*2, 3, 5, 6*
1234	*2, 5*

M = 4
SUB = [{1}, {2}, {3}, {4}, {1,2}, {1,3}, {1,4}, {2,3}, {2,4}, {3,4}, {1,2,3}, {1,2,4}, {1,3,4}, {2,3,4}]
K = 14

Update generators

GEN

Itemset	Object List	Closure
1	*1, 2, 5*	12
3	*2, 3, 4, 5, 6*	34
4	*2, 3, 4, 5, 6*	34
23	*2, 3, 5, 6*	234
24	*2, 3, 5, 6*	234
13	*2, 5*	1234
14	*2, 5*	1234

Generate rules

Rules set	Antecedent	Consequent	Support	Confidence	Object list
AR_Exact	{1}	{2}	3	1	*1, 2, 5*
	{3}	{4}	5	1	*2, 3, 4, 5, 6*
	{4}	{3}	5	1	*2, 3, 4, 5, 6*
	{2,4}	{3}	4	1	*2, 3, 5, 6*
	{2,3}	{4}	4	1	*2, 3, 5, 6*
	{1,3}	{2,4}	2	1	*2, 5*
	{1,4}	{2,3}	2	1	*2, 5*
AR_Approx	{2}	{1}	3	0.6	*1, 2, 5*

Continued on next page...

Table 5.10 – *Continued*

	{2}	{3,4}	4	0.8	*2, 3, 5, 6*
	{3,4}	{2}	4	0.8	*2, 3, 5, 6*
	{3}	{2,4}	4	0.8	*2, 3, 5, 6*
	{4}	{2,3}	4	0.8	*2, 3, 5, 6*
	{1}	{2,3,4}	2	0.67	*2, 5*
	{2}	{1,3,4}	2	0.4	*2, 5*
	{3}	{1,2,4}	2	0.4	*2, 5*
	{4}	{1,2,3}	2	0.4	*2, 5*
	{1,2}	{3,4}	2	0.67	*2, 5*
	{2,3}	{1,4}	2	0.5	*2, 5*
	{2,4}	{1,3}	2	0.5	*2, 5*
	{3,4}	{1,2}	2	0.4	*2, 5*

Last step of this phase consists, if required by the user, to map number values of items and objects in the output sets to explicit values in the original database. This phase is required to generate user-friendly understandable knowledge patterns. The final outputs obtained after mapping for the example database D1 are given in table 5.11. It should be note that for this example database, no object label is present. Also, the pairs *variable* = *value* are represented by the *variable* term alone for simplicity since only one value is considered (1) for each in this example. In the case of genomics or proteomics databases for instance, object identifiers in final patterns could be gene or protein names if these appear in the initial dataset.

Table 5.11: Resulting Outputs After Mapping of Identifiers to Example Database D1 Labels

Frequent Closed Patterns		
Order #	Closed set	Object Ids
1	C	*1, 2, 3, 5, 6*
2	AC	*1, 2, 5*
3	BE	*2, 3, 4, 5, 6*
4	BCE	*2, 3, 5, 6*
5	ABCE	*2, 5*

Generators		
Itemset	Object List	Closure
A	*1, 2, 5*	AC
B	*2, 3, 4, 5, 6*	BE
E	*2, 3, 4, 5, 6*	BE
BC	*2, 3, 5, 6*	BCE
CE	*2, 3, 5, 6*	BCE
AB	*2, 5*	ABCE
AE	*2, 5*	ABCE

Conceptual association rules					
Rule set	Antecedent	Consequents	Support	Confidence	Object List

Continued on next page...

Table 5.11 – *Continued*

AR_Exact	{A}	{C}	3	1	*1, 2, 5*
	{B}	{E}	5	1	*2, 3, 4, 5, 6*
	{E}	{B}	5	1	*2, 3, 4, 5, 6*
	{CE}	{B}	4	1	*2, 3, 5, 6*
	{BC}	{E}	4	1	*2, 3, 5, 6*
	{AB}	{CE}	2	1	*2, 5*
	{AE}	{BC}	2	1	*2, 5*
AR_Approx	{C}	{A}	3	0.6	*1, 2, 5*
	{C}	{BE}	4	0.8	*2, 3, 5, 6*
	{BE}	{C}	4	0.8	*2, 3, 5, 6*
	{B}	{CE}	4	0.8	*2, 3, 5, 6*
	{E}	{BC}	4	0.8	*2, 3, 5, 6*
	{A}	{BCE}	2	0.67	*2, 5*
	{C}	{ABE}	2	0.4	*2, 5*
	{B}	{ACE}	2	0.4	*2, 5*
	{E}	{ABC}	2	0.4	*2, 5*
	{AC}	{BE}	2	0.67	*2, 5*
	{BC}	{AE}	2	0.5	*2, 5*
	{CE}	{AB}	2	0.5	*2, 5*
	{BE}	{AC}	2	0.4	*2, 5*

5.2 FIST 2.0 Execution

This section depicts the execution of the second algorithmic version of the FIST approach for the same example database D1, presented in table 5.1, and for the same parameter values. Each of the following subsections presents separately one of the five main steps of the algorithm.

5.2.1 Step 1: Generating the Sorted Frequent Database

The first step is the creation of the Sorted Frequent Database from the source dataset. This step is decomposed in four different tasks. First task is to create the list of distinct *attribute = value* pairs, while counting their number of occurrences (support), from the dataset. The resulting Item Table is given in table 5.12. The second task is to delete infrequent pairs using the user defined *minsupport* threshold. The resulting Frequent Item Table is given in table 5.13.

Attribute values	Support
A=1	3
C=1	5
D=1	1
B=1	5
E=1	5

Table 5.12: Item Table for Database D1

Attribute values	Support
A=1	3
C=1	5
B=1	5
E=1	5

Table 5.13: Frequent Item Table for Database D1 and *minsupport* = 2/6

Then, frequent pairs are sorted in increasing order of their support and, to simplify future textual mapping operations, if two pairs have the same support they are ordered in lexicographic order. Each pair is then assigned a unique item numeric identifier as shown in the table 5.14. These item numbers are then used to construct the Sorted Frequent Database by mapping attribute and value pairs in the dataset to their corresponding item. This mapping gives the transactional format SFD database in which each row contains a set of items. Items are inserted in objects of the SFD database in lexicographic order as shown in table 5.15.

Attribute=Value	Support	Item Number
A=1	3	1
B=1	5	2
C=1	5	3
E=1	5	4

Table 5.14: Item Identifiers for Example Database D1 and *minsupport* = 2/6

Object	Items
1	1 3
2	1 2 3 4
3	2 3 4
4	2 4
5	1 2 3 4
6	2 3 4

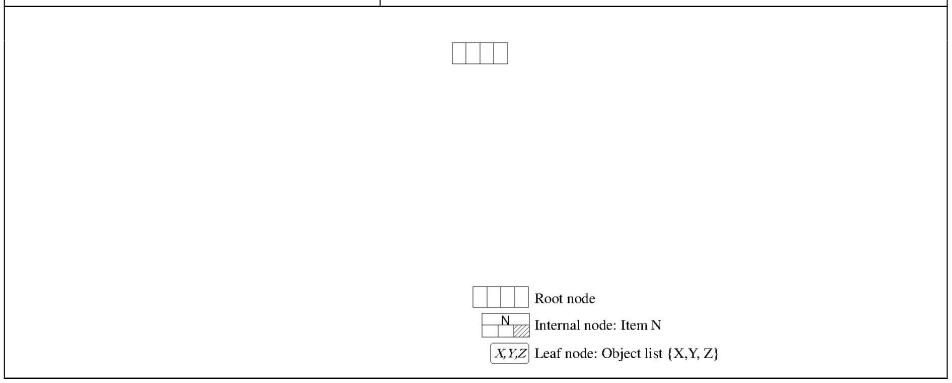
Table 5.15: Sorted Frequent Database for Database D1 and *minsupport* = 2/6

5.2.2 Step 2: Constructing Frequent Generalized Itemset Suffix-Tree

The second step consists to create the FGIST from the SFD. This step uses a process similar to the one of the first algorithmic version with some exceptions. Objects in the SFD are also read sequentially to insert suffixes in the FGIST. The main difference lies in the fact that suffixes are passed one at a time to the function updating the tree. At the end of the dataset scan, the FGIST is created and will be updated and pruned during next steps.

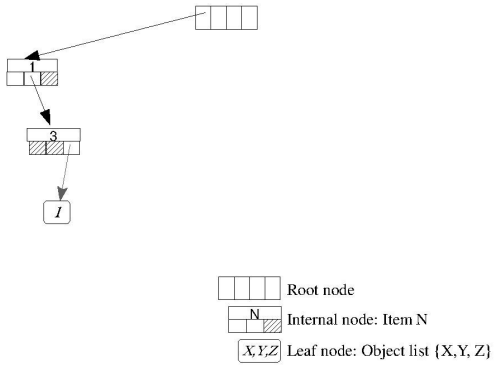
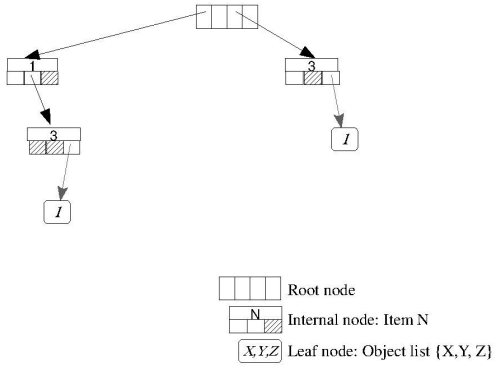
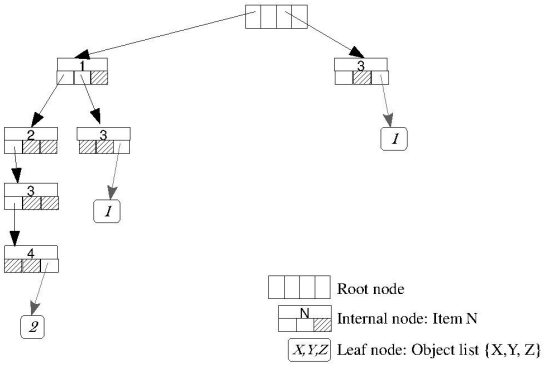
The trace of the row-wise process for building the FGIST for example database D1 is shown in table 5.16. The *Step* column represents row reads in the dataset and function call operations. The *Object* column represents the vector of items corresponding to the row. The *OID* column represents the object identifier (sequential number) corresponding to the row. The *Suffix* column represents the suffix created and inserted for the object read.

Table 5.16: FGIST Generation for Example Database D1 with FIST 2.0

Step	**Object**	**OID**	**Suffix**
Create root			
			

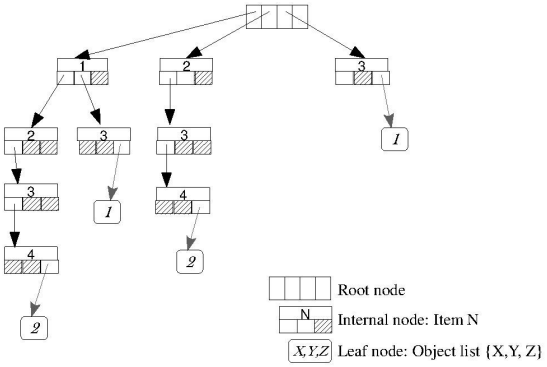
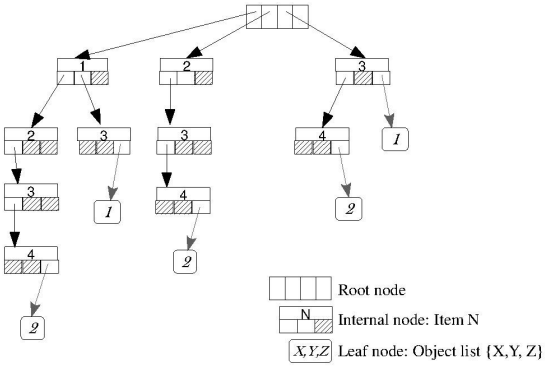
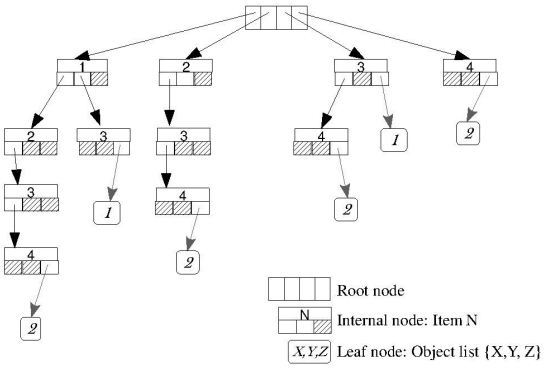
Continued on next page...

Table 5.16 – *Continued*

Step	**Object**	**OID**	**Suffix**
Read row 1	1 3	*1*	
Call *match*() and *buildEdge*()			13*1*
			
Call *match*() and *buildEdge*()			3*1*
			
Read row 2	1 2 3 4	*2*	
Call *match*() and *buildEdge*()			1234*2*
			

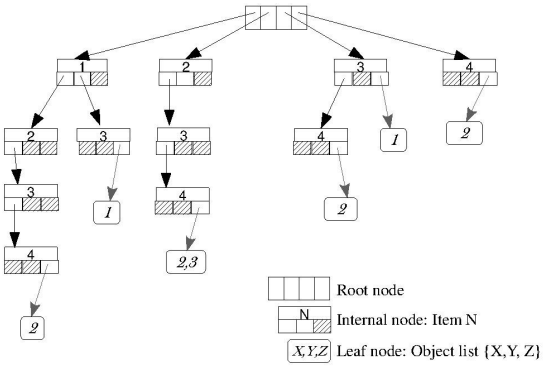
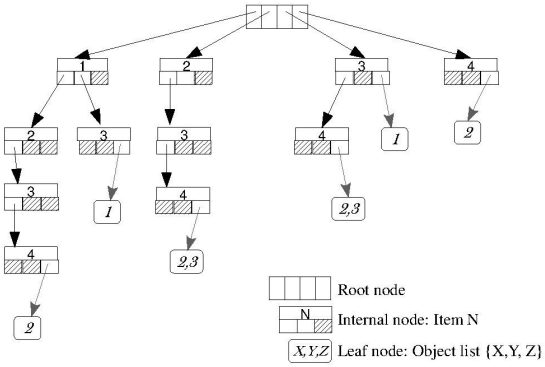
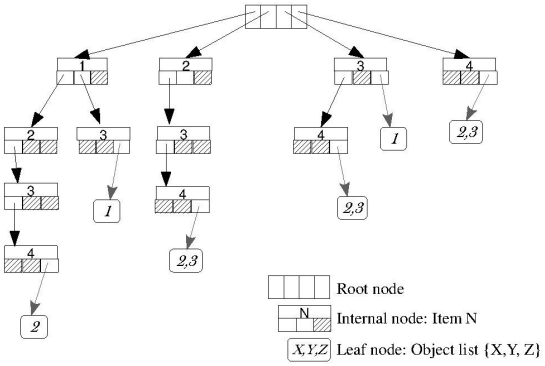
Continued on next page...

Table 5.16 – *Continued*

Step	Object	OID	Suffix
Call *match*() and *buildEdge*()			234*2*
			
Call *match*() and *buildEdge*()			34*2*
			
Call *match*() and *buildEdge*()			4*2*
			

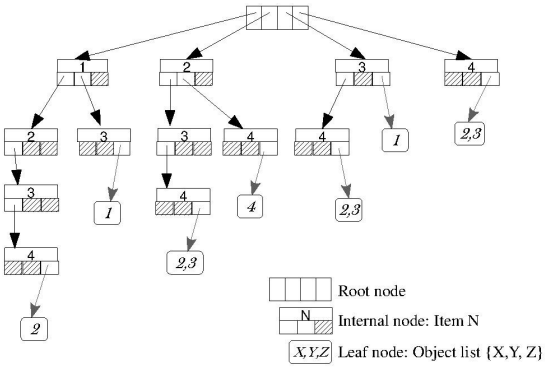
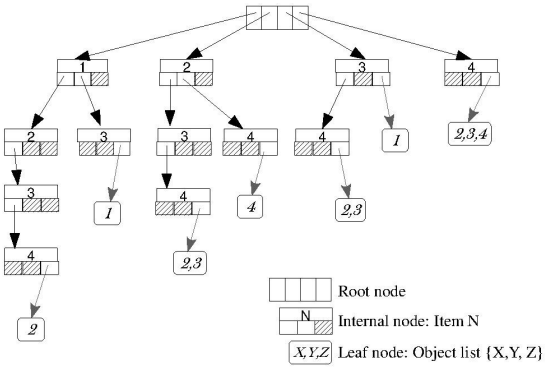
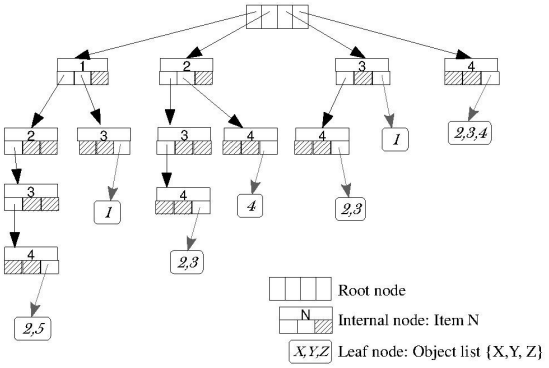
Continued on next page...

Table 5.16 – *Continued*

Step	**Object**	**OID**	**Suffix**
Read row 3	2 3 4	*3*	
Call *match*() and *buildEdge*()			234*3*
			
Call *match*() and *buildEdge*()			34*3*
			
Call *match*() and *buildEdge*()			4*3*
			

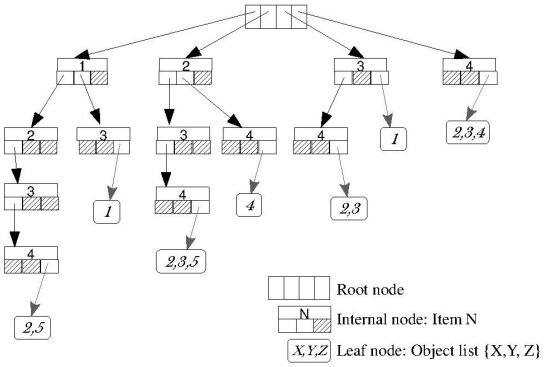
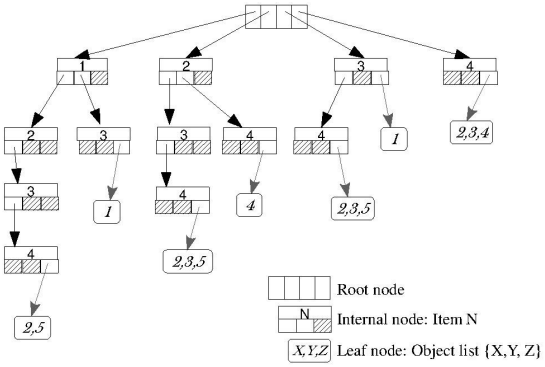
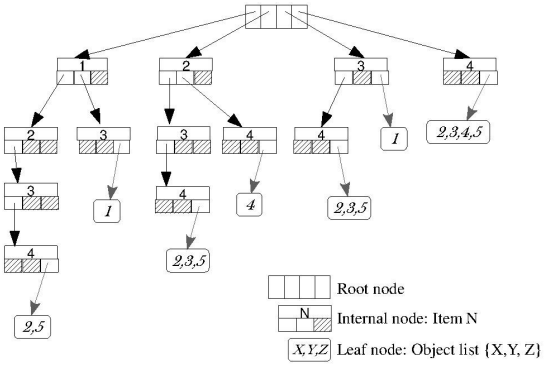
Continued on next page...

Table 5.16 – *Continued*

Step	**Object**	**OID**	**Suffix**
Read row 4	2 4	*4*	
Call *match*() and *buildEdge*()			24*4*
			
Call *match*() and *buildEdge*()			4*4*
			
Read row 5	1 2 3 4	*5*	
Call *match*() and *buildEdge*()			1234*5*
			

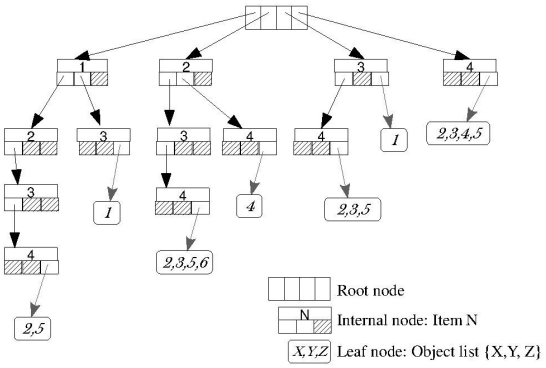
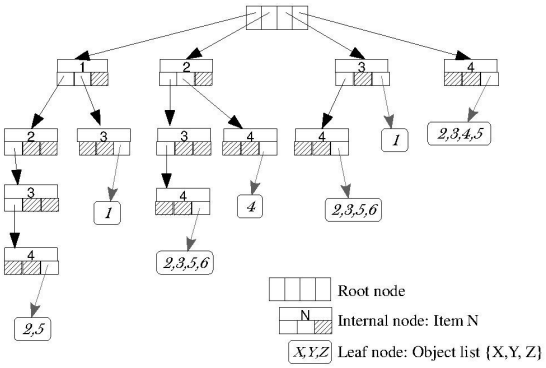
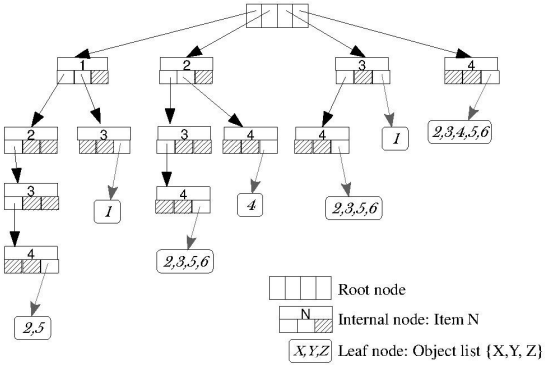
Continued on next page...

Table 5.16 – *Continued*

Step	Object	OID	Suffix
Call *match*() and *buildEdge*()			2345
			
Call *match*() and *buildEdge*()			345
			
Call *match*() and *buildEdge*()			45
			

Continued on next page...

Table 5.16 – *Continued*

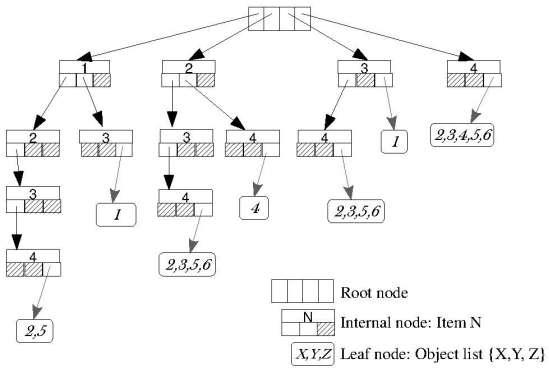
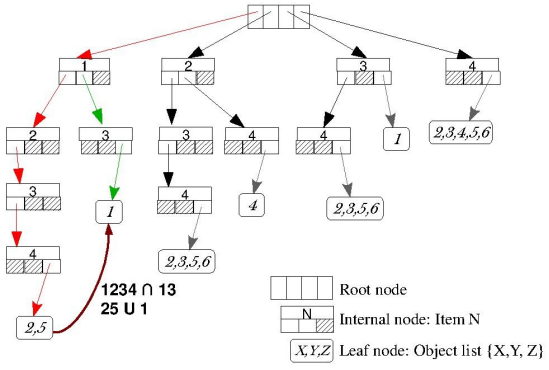
Step	**Object**	**OID**	**Suffix**
Read row 6	2 3 4	*6*	
Call *match*() and *buildEdge*()			234*6*
			
Call *match*() and *buildEdge*()			34*6*
			
Call *match*() and *buildEdge*()			4*6*
			

5.2.3 Step 3: Updating Frequent Generalized Itemset Suffix-Tree

After construction of the FGIST from the SFD, update operations of the object lists are performed.

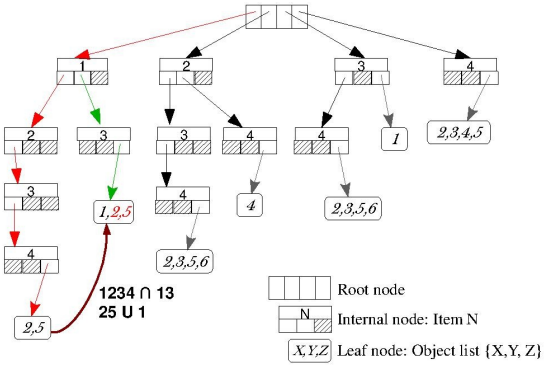
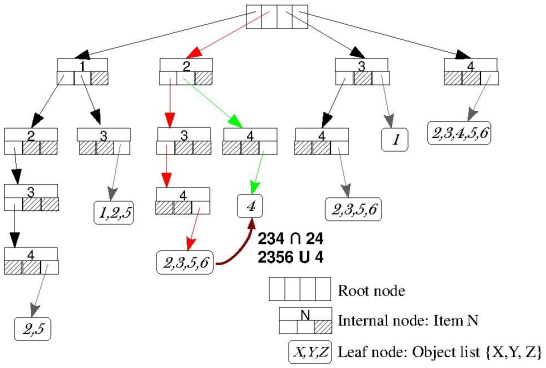
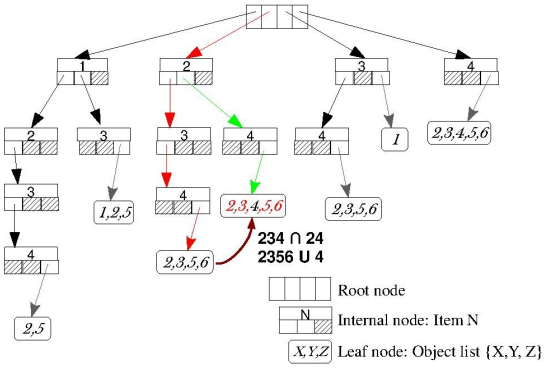
The trace of the execution of this recursive process is presented in table 5.17. It shows how a *Prefix* itemset, represented by edges in red color in the FGIST graphics, is compared with other *Current* itemsets, represented by edges in green color in the FGIST graphics. Both itemsets, represented by a path from the root to a node with an object list, are collected by traversing the FGIST in a depth-first manner while collecting successive items. For pairs of itemsets considered, the *Intersection* between the itemsets and the *Union* between the object lists are computed. This depth-first process is performed branch by branch and when an update of the *Current* itemset node is required it is depicted in red in the FGIST. This update consists to replace the object list of the *Current* itemset node by the computed *Union* object list. The output is the complete FGIST shown at the end of the table.

Table 5.17: Updating the FGIST for Example Database D1 with FIST 2.0

Prefix	**Current**	**Union**	**Intersection**
Begin			
			
For branch beginning by 1			
1234	13		
			

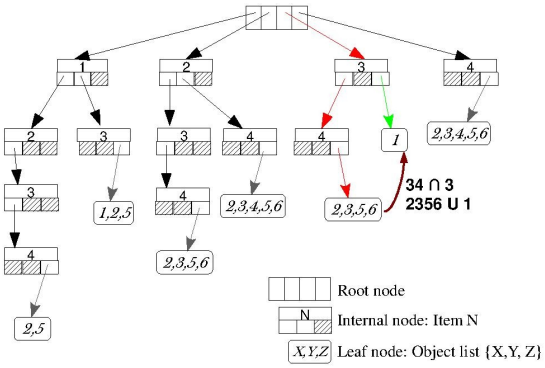
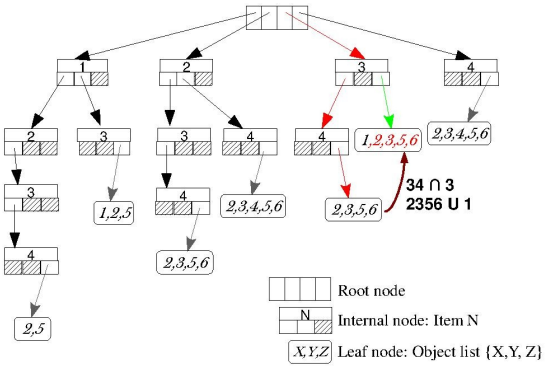
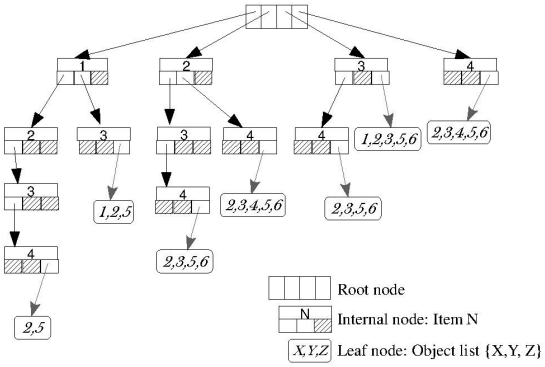
Continued on next page...

Table 5.17 – *Continued*

Prefix	Current	Intersection	Union
1234	13	13	*1,2,5*
			
For branch beginning by 2			
234	24		
			
234	24	24	*2,3,4,5,6*
			

Continued on next page. . .

Table 5.17 – *Continued*

Prefix	**Current**	**Intersection**	**Union**
For branch beginning by 3			
34	3		
			
34	3	3	*1,2,3,5,6*
			
For branch beginning by 4			
4			
			
End			

5.2.4 Step 4: Pruning Frequent Generalized Itemset Suffix-Tree

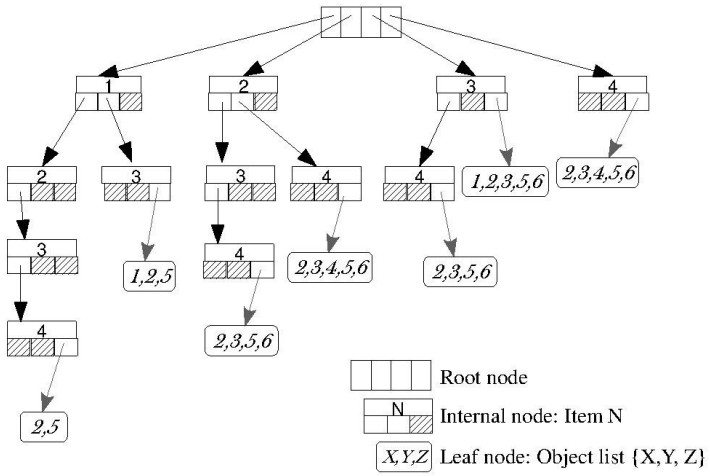
The penultimate step consists to prune the FGIST by deleting nodes that are unnecessary for knowledge pattern generation. These nodes represent infrequent and non-closed itemsets. The frequent patterns in the FGIST which itemset is included in the itemset of another frequent pattern and which object list is a subset of the object list of the other frequent pattern are pruned from the tree.

The trace of the execution of this recursive process is presented in table 5.18. Traversing the FGIST in a depth-first manner, itemsets, identified by the presence of an object list, are collected in the *Prefix* and *Current* itemsets. These *Prefix* itemsets considered successively are represented by edges in red color in the FGIST graphics. They are compared to the *Current* itemsets represented by edges in green color in the FGIST graphics. The result of the inclusion test between the *Prefix* and the *Current* itemsets is shown in the *Prefix* $\supset$ *Current?* column. The result of the equality test between the object lists of the *Prefix* and the *Current* itemsets is shown in the *Prefix.OIDs* = *Current.OIDs?* column. In case these two tests are true, the node representing the *Current* itemset is pruned from the FGIST. In its final state, depicted at the end of the table, the FGIST contains all and only frequent closed patterns.

Table 5.18: Pruning FGIST for Example Database D1 with FIST 2.0

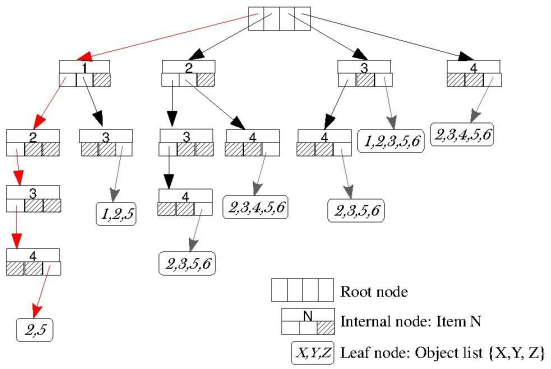
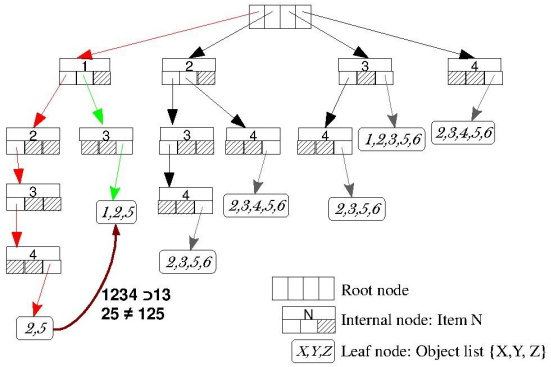
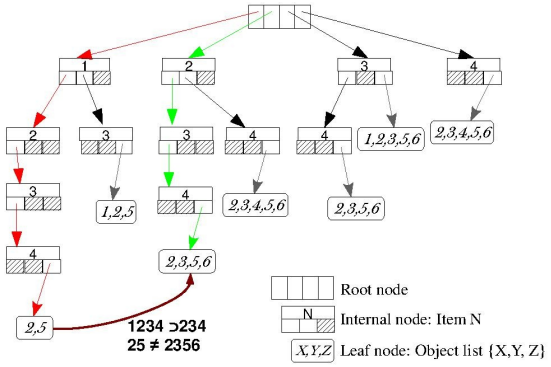
Prefix	**Current**	**Prefix $\supset$ Current?**	**Prefix.OIDs = Current.OIDs?**
Begin			

Initial FGIST:



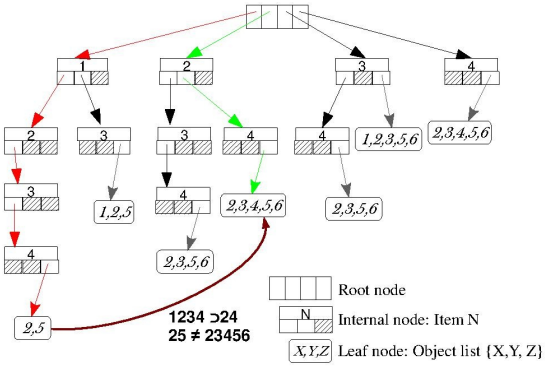
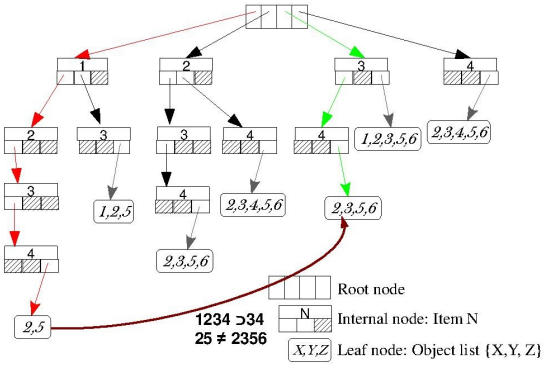
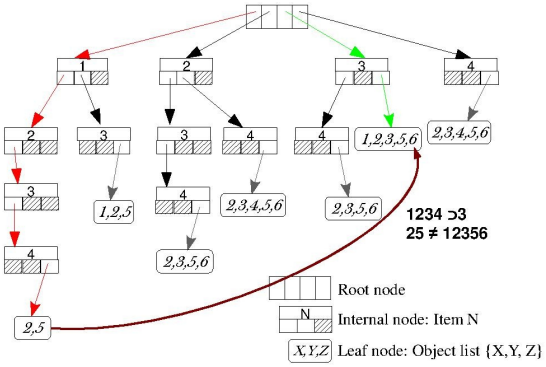
Continued on next page...

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
1234			
			
1234	13	Yes	No
			
1234	234	Yes	No
			

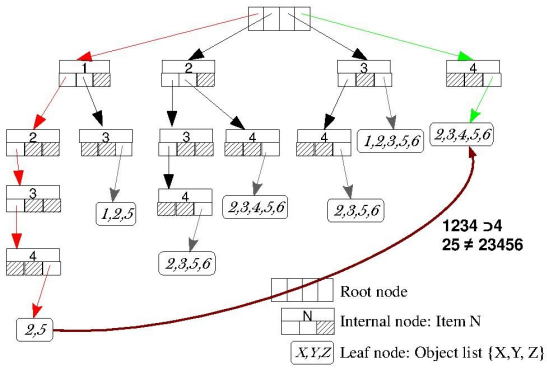
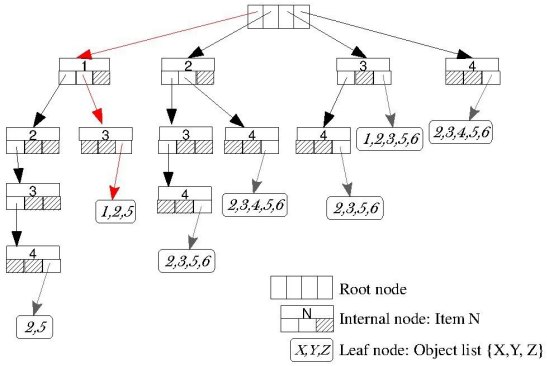
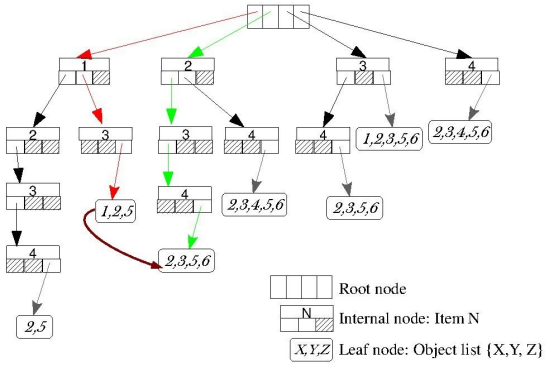
Continued on next page...

Table 5.18 – *Continued*

Prefix	Current	Prefix ⊃ Current?	Prefix.OIDs = Current.OIDs?
1234	24	Yes	No
			
1234	34	Yes	No
			
1234	3	Yes	No
			

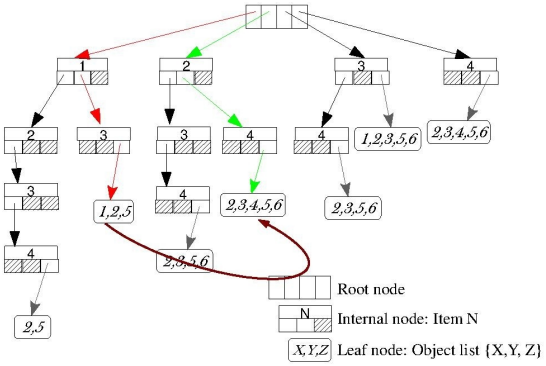
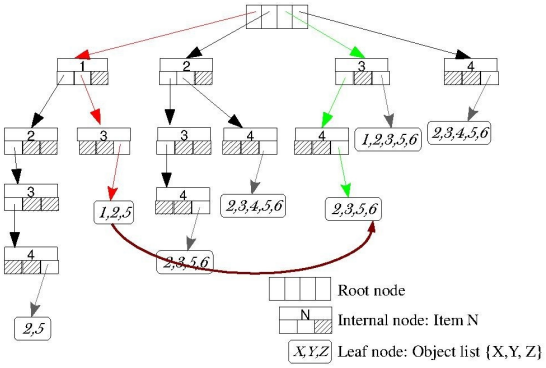
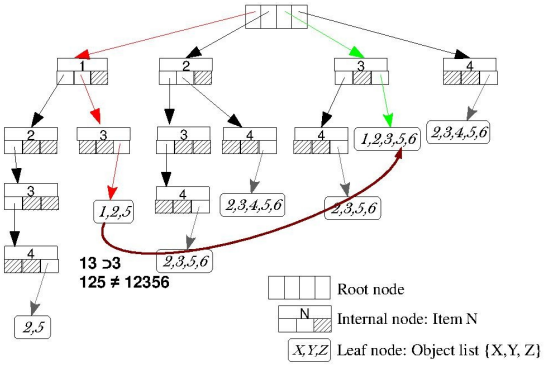
Continued on next page. . .

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
1234	4	Yes	No
			
13			
			
13	234	No	No
			

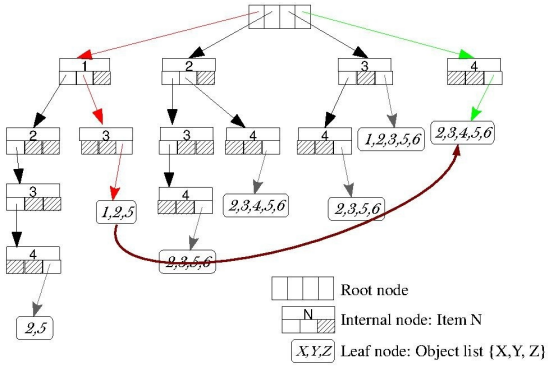
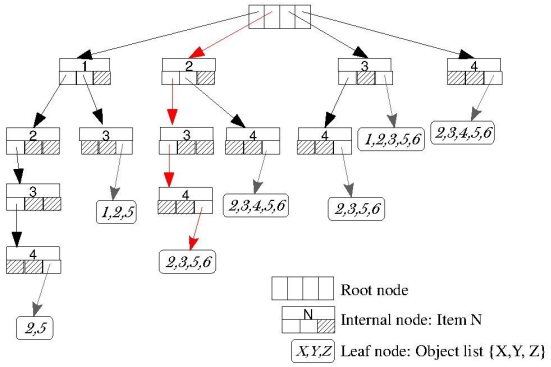
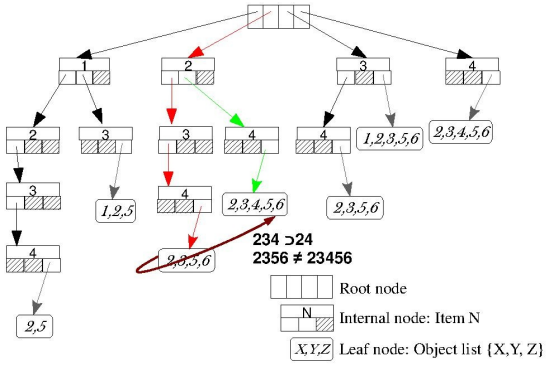
Continued on next page...

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
13	24	No	No
			
13	34	No	No
			
13	3	Yes	No
			

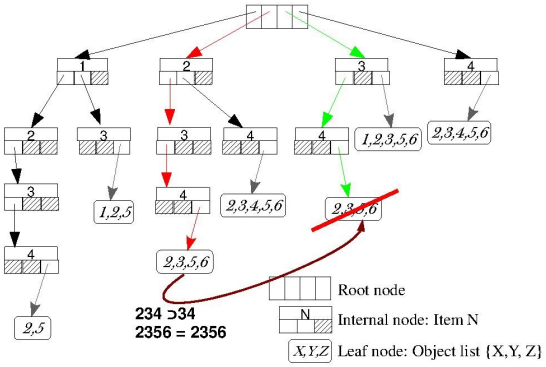
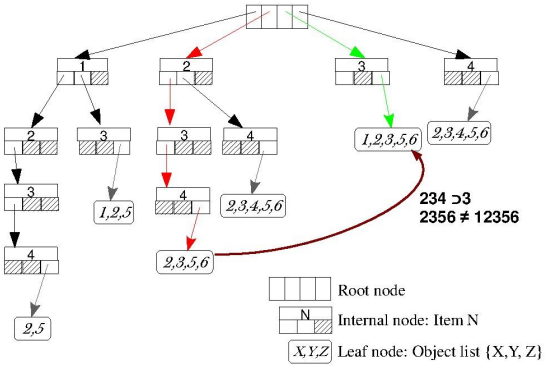
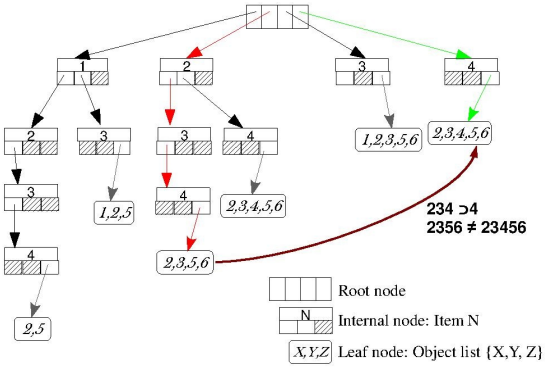
Continued on next page. . .

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
13	4	No	No
			
234			
			
234	24	Yes	No
			

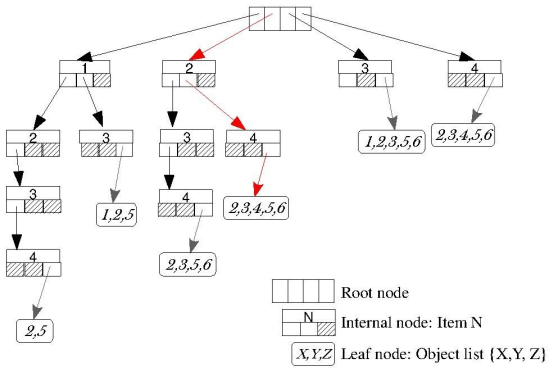
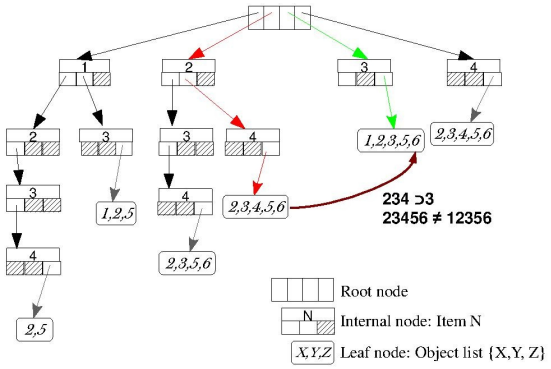
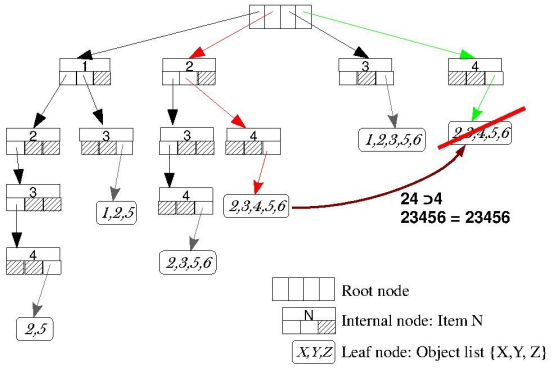
Continued on next page...

Table 5.18 – *Continued*

Prefix	Current	Prefix ⊃ Current?	Prefix.OIDs = Current.OIDs?
234	34	Yes	Yes
			
234	3	Yes	No
			
234	4	Yes	No
			

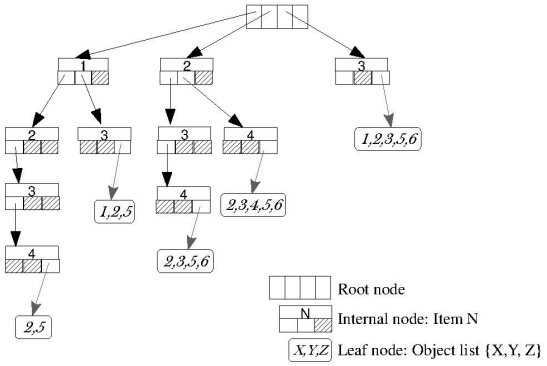
Continued on next page. . .

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
24			
			
24	3	No	No
			
24	4	Yes	Yes
			

Continued on next page...

Table 5.18 – *Continued*

Prefix	**Current**	**Prefix ⊃ Current?**	**Prefix.OIDs = Current.OIDs?**
3			
			
End			

5.2.5 Step 5: Generating Conceptual Knowledge Pattern Information

The last step consists to generate the conceptual knowledge patterns from the frequent patterns in the FGIST. In order to authorize and simplify the extension of this process, by considering other pattern generation and selection criteria for instance, the output sets are generated into two distinct tasks.

Table 5.19 shows incremental updates of the BIC and GEN sets of conceptual bi-clusters and frequent generators respectively. New inserted elements in these sets are depicted in blue color characters. For this process, frequent patterns are considered in increasing order of the size of their itemset to minimize the number of pattern comparisons. For each pattern considered, a bi-clusters associating the itemset and the object list of the pattern is created in BIC. Then, the SUB list of subsets of its (frequent closed) itemset is created and subsets are tested in increasing order of their size. The test consists to determine if the *subset* is a generator of the frequent closed itemset; the result of this test is shown in variable *not_gen*. Once the first generator has been determined, the size of generators of this frequent closed itemset is indicated in the *gen_size* variable. Hence, as generators of the frequent closed itemsets constitute an order ideal, all its generators have the same size. Larger subsets than the *gen_size* variable will then not be considered.

Table 5.19: Extracting Bi-clusters and Generators for Example Database D1 with FIST 2.0

Begin
Initialize the output sets BIC and GEN to ϕ

First closed pattern: {{3}, {*1, 2, 3, 5, 6*}}	
BIC	

Continued on next page...

Table 5.19 – *Continued*

Intent	Extent
3	*1, 2, 3, 5, 6*
SUB = ϕ	

Second closed pattern: {{13}, {*1, 2, 5*}}		
BIC		
Intent	Extent	
3	*1, 2, 3, 5, 6*	
13	*1, 2, 5*	
Initialize found_gen = false and gen_size = 2		
SUB = {{1}, {3}, {13}}		
subset = {1}		
not_gen = false		
gen_size = 1		
GEN		
Itemset	Object List	Closure
1	*1, 2, 5*	12
subset = {3}		
not_gen = true		
Stop on criterium gen_size = 1		

Third closed pattern: {{24}, {*2, 3, 4, 5, 6*}}		
BIC		
Intent	Extent	
3	*1, 2, 3, 5, 6*	
13	*1, 2, 5*	
24	*2, 3, 4, 5, 6*	
Initialize found_gen = false and gen_size = 2		
SUB = {{2}, {4}, {24}}		
subset = {2}		
not_gen = false		
gen_size = 1		
GEN		
Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
subset = {4}		
not_gen = false		
GEN		
Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
4	*2, 3, 4, 5, 6*	24

Continued on next page...

Table 5.19 – *Continued*

Stop on criterium gen_size = 1

Fourth closed pattern: {{234}, {*2, 3, 5, 6*}}

BIC

Intent	Extent
3	*1, 2, 3, 5, 6*
13	*1, 2, 5*
24	*2, 3, 4, 5, 6*
234	*2, 3, 5, 6*

Initialize found_gen = false and gen_size = 3

SUB = {{2}, {3}, {4}, {23}, {24}, {34}, {234}}

subset = {2}

not_gen = true

subset = {3}

not_gen = true

subset = {4}

not_gen = true

subset = {23}

not_gen = false

gen_size = 2

GEN

Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
4	*2, 3, 4, 5, 6*	24
23	*2, 3, 5, 6*	234

subset {24}

not_gen = true

subset {34}

not_gen = false

GEN

Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
4	*2, 3, 4, 5, 6*	24
23	*2, 3, 5, 6*	234
34	*2, 3, 5, 6*	234

Stop on criterium gen_size = 2

Fifth closed pattern: {{1234}, {*2, 5*}}

BIC

Intent	Extent
3	*1, 2, 3, 5, 6*
13	*1, 2, 5*

Continued on next page...

Table 5.19 – *Continued*

24	*2, 3, 4, 5, 6*	
234	*2, 3, 5, 6*	
1234	*2, 5*	

Initialize found_gen = false and gen_size = 4

SUB = {{1}, {2}, {3}, {4}, {12}, {13}, {14}, {23}, {24}, {34}, {123}, {124}, {134}, {234}, {234}}

subset = {1}

not_gen = true

subset = {2}

not_gen = true

subset = {3}

not_gen = true

subset = {4}

not_gen = true

subset = {12}

not_gen = false

gen_size = 2

GEN		
Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
4	*2, 3, 4, 5, 6*	24
23	*2, 3, 5, 6*	234
34	*2, 3, 5, 6*	234
12	*2, 5*	1234

subset = {13}

not_gen = true

subset = {14}

not_gen = false

GEN		
Itemset	Object List	Closure
1	*1, 2, 5*	13
2	*2, 3, 4, 5, 6*	24
4	*2, 3, 4, 5, 6*	24
23	*2, 3, 5, 6*	234
34	*2, 3, 5, 6*	234
12	*2, 5*	1234
14	*2, 5*	1234

subset = {23}

not_gen = true

subset = {24}

not_gen = true

subset = {34}

Continued on next page...

Table 5.19 – *Continued*

not_gen = true
Stop on criterium gen_size = 2

Table 5.20 shows the result of the generation of the bases of conceptual association rules from the GEN set. Depending on the user's requirements, the AR_E base of min-max exact conceptual association rules and, either, or both, the AR_PB base of proper approximate conceptual association rules and the AR_SB base of min-max approximate conceptual association rules can be generated. This generation is straightforward as for each generator, a min-max exact rule is created between the generator and its closure if they are different. A min-max approximate rule is created between the generator and the frequent closed itemsets that are supersets of its closure. The antecedent of such a rule is the generator and its consequent is the set difference between the frequent closed itemset and the generator. Proper approximate rules are created between two frequent closed itemsets if one is a subset of the other. The subset frequent closed itemset is the antecedent and the set difference between the superset and the subset is the consequent of such a rule.

Table 5.20: Generating Association Rule Bases for Example Database D1 with FIST 2.0

Begin

Sort generators of same size in lexicographic order

GEN

Generator	Closure	Object list
1	13	*1, 2, 5*
2	24	*2, 3, 4, 5, 6*
4	24	*2, 3, 4, 5, 6*
12	1234	*2, 5*
14	1234	*2, 5*
23	234	*2, 3, 5, 6*
34	234	*2, 3, 5, 6*

Initialize output sets AR_E, AR_SB and AR_PB to ϕ

Generating Min-Max Exact Rules in AR_E

Rule #	Antecedent	Consequent	Support	Confidence	Object List
1	{1}	{3}	3	1	*1, 2, 5*
2	{2}	{4}	5	1	*2, 3, 4, 5, 6*
3	{4}	{2}	5	1	*2, 3, 4, 5, 6*
4	{23}	{4}	4	1	*2, 3, 5, 6*
5	{34}	{2}	4	1	*2, 3, 5, 6*
6	{12}	{34}	2	1	*2, 5*
7	{14}	{23}	2	1	*2, 5*

Generating Proper Approximate Rules in AR_PB

Rule #	Antecedent	Consequents	Support	Confidence	Object List
1	{3}	{1}	3	0.6	*1, 2, 5*

Continued on next page...

Table 5.20 – *Continued*

2	{3}	{24}	4	0.8	*2, 3, 5, 6*
3	{3}	{124}	2	0.4	*2, 5*
4	{13}	{24}	2	0.67	*2, 5*
5	{24}	{3}	4	0.8	*2, 3, 5, 6*
6	{24}	{13}	2	0.4	*2, 5*
7	{234}	{1}	2	0.5	*2, 5*

Generating Min-Max Approximate Rules in AR_SB					
Rule #	Antecedent	Consequents	Support	Confidence	Object List
1	{1}	{234}	2	0.67	*2, 5*
2	{2}	{34}	4	0.8	*2, 3, 5, 6*
3	{2}	{134}	2	0.4	*2, 5*
4	{4}	{23}	4	0.8	*2, 3, 5, 6*
5	{4}	{123}	2	0.4	*2, 5*
6	{23}	{14}	2	0.5	*2, 5*
7	{34}	{12}	2	0.5	*2, 5*

The final task consists to map item and object numeric identifiers in patterns to database values if required. This mapping is inevitable to provide knowledge patterns that are understandable by the end-user if he/she must evaluate results. It can be omitted if there is not such a requirement, for example if resulting patterns will be used only for an automatic processing. The patterns obtained for the example database D1 after mapping of the item numbers with attributes values in the initial dataset are given in table 5.21. For this example database, no mapping is performed for object identifiers in object lists as no label is given for them in the initial dataset. Since variables in this example are unary variables (one value considered for each), the pairs *variable* = *value* are represented by the *variable* term alone for simplicity.

Table 5.21: Resulting Patterns After Mapping Values for Example Database D1 with FIST 2.0

Frequent Item Table		
Attribute=Value	Support	Item Number
A=1	3	1
B=1	5	2
C=1	5	3
E=1	5	4

Mapping Conceptual Bi-clusters		
Order #	Closed set	Object Ids
1	C	*1, 2, 3, 5, 6*
2	AC	*1, 2, 5*
3	BE	*2, 3, 4, 5, 6*
4	BCE	*2, 3, 5, 6*
5	ABCE	*2, 5*

Continued on next page...

Table 5.21 – *Continued*

Mapping Generators		
Itemset	Object List	Closure
A	*1, 2, 5*	AC
B	*2, 3, 4, 5, 6*	BE
E	*2, 3, 4, 5, 6*	BE
BC	*2, 3, 5, 6*	BCE
CE	*2, 3, 5, 6*	BCE
AB	*2, 5*	ABCE
AE	*2, 5*	ABCE

Mapping Min-Max Exact Rules in AR_E					
Rule #	Antecedent	Consequents	Support	Confidence	Object List
1	{A}	{C}	3	1	*1, 2, 5*
2	{B}	{E}	5	1	*2, 3, 4, 5, 6*
3	{E}	{B}	5	1	*2, 3, 4, 5, 6*
4	{BC}	{E}	4	1	*2, 3, 5, 6*
5	{CE}	{B}	4	1	*2, 3, 5, 6*
6	{AB}	{CE}	2	1	*2, 5*
7	{AE}	{BC}	2	1	*2, 5*

Mapping Proper Approximate Rules in AR_PB					
Rule #	Antecedent	Consequents	Support	Confidence	Object List
1	{C}	{A}	3	0.6	*1, 2, 5*
2	{C}	{BE}	4	0.8	*2, 3, 5, 6*
3	{C}	{ABE}	2	0.4	*2, 5*
4	{AC}	{BE}	2	0.67	*2, 5*
5	{BE}	{C}	4	0.8	*2, 3, 5, 6*
6	{BE}	{AC}	2	0.4	*2, 5*
7	{BCE}	{A}	2	0.5	*2, 5*

Mapping Min-Max Approximate Rules in AR_SB					
Rule #	Antecedent	Consequents	Support	Confidence	Object List
1	{A}	{BCE}	2	0.67	*2, 5*
2	{B}	{CE}	4	0.8	*2, 3, 5, 6*
3	{B}	{ACE}	2	0.4	*2, 5*
4	{E}	{BC}	4	0.8	*2, 3, 5, 6*
5	{E}	{ABC}	2	0.4	*2, 5*
6	{BC}	{AE}	2	0.5	*2, 5*
7	{CE}	{AB}	2	0.5	*2, 5*

5.3 Properties of the FIST Approach

The properties on which relies the FIST approach and the two algorithmic versions presented are studied in this section. These properties, that allow to minimize resources required for the extraction of conceptual knowledge patterns, are discusses with their proof.

5.3.1 Search-space and Theoretical Framework

Search-space of association rule mining approaches was initially defined as the itemset lattice $\mathcal{L}_I$, or subset lattice. The size of this search-space is exponential in the number of items in the database: $|\mathcal{L}_I| = 2^{|I|}$. Frequent itemset and maximal frequent itemset based approaches use different traversals of this lattice to identify frequent itemsets in this lattice, and compute their support, from the dataset. The Hasse diagram of the itemset lattice $\mathcal{L}_I$ for the example database D1, given in table 5.1, is shown in figure 5.1. Frequent itemsets for $minsupport = 2/6$ are shown in colored rectangles.

The connexion of the Galois closure $\Gamma(\psi, \gamma)$ theoretical framework allows to reduce the search-space for association rule mining to frequent closed itemsets. Using this framework, equivalence classes, that are sub-orders of the itemset lattice, are defined by frequent generators and frequent closed itemsets. Each equivalence class regroups a set of itemsets included in the same objects of the dataset (see Proposition 5.1) and that consequently have the same support. Frequent equivalence classes for the example database D1 and $minsupport = 2/6$ are shown in figure 5.2. The list of dataset objects supporting each frequent equivalence class are depicted in red characters in the figure.

The frequent closed itemsets theoretical framework was shown to be an optimized framework for association rule mining as the set of frequent closed itemsets with their support contains all information necessary to generate all association rules with support and confidence. It is minimal, as no smaller set contains the same information, and non-redundant since no frequent closed itemset can be derived from other frequent closed itemsets. The generators, that are the minimal itemsets, regarding inclusion relation, of an equivalence class, were defined to optimize the search for frequent closed itemsets and to generate minimal non-redundant covers of association rules.

The FIST approach first extracts from the dataset all frequent closed itemsets that are represented together with their supporting object list in the FGIST as frequent closed patterns. The size of supporting object lists corresponds to the support of the corresponding itemset in the dataset. From this information, the lattice of frequent conceptual bi-clusters, each bi-cluster associating a frequent closed itemset as *intent* and the supporting object list of this itemset as *extent*, is extracted.

Definition 5.1 (Frequent Conceptual Bi-clusters) *A* frequent conceptual bi-cluster *is a sub-matrix of the dataset matrix representation associating a subset of objects and a subset of items such that all these objects contain all these items and all other objects of the matrix do not contain all these items. That is, a frequent conceptual bi-cluster B_k in a binary dataset $M = (O, L, R)$ is a dual set $B_k = \{I, P\}$ where $I \subseteq L$ is a maximal set of items and $P \subseteq O$ is a maximal set of objects defined as follows:*

i) $I \in \mathcal{FC}$.

ii) $\forall i \in I$ and $\forall o \in P$, we have $i \in o$ and $\nexists i' \neq i$ such that $\forall o \in P$ we have $i' \in o$.

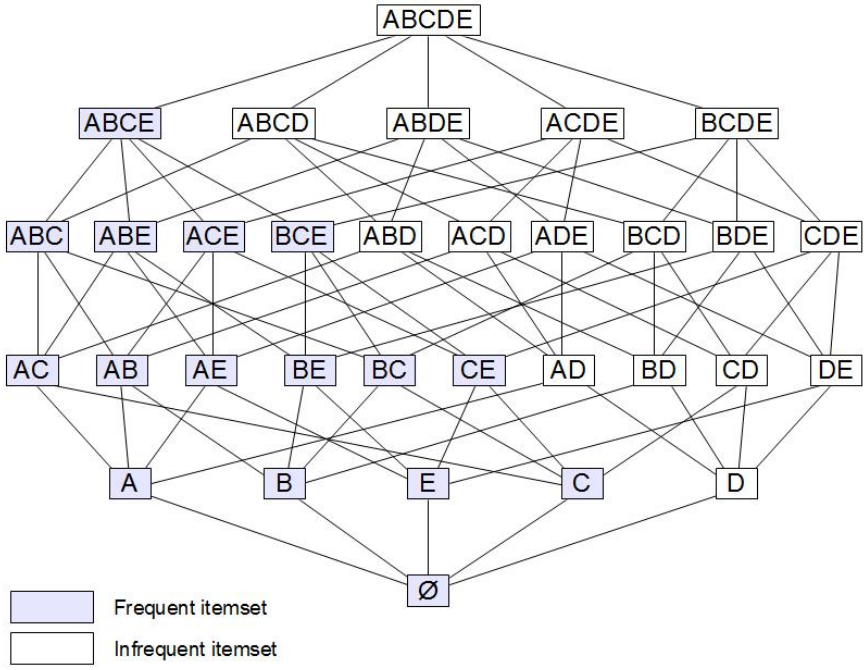


Figure 5.1: Itemset Lattice for Example Database D1 and *minsupport* = 2/6

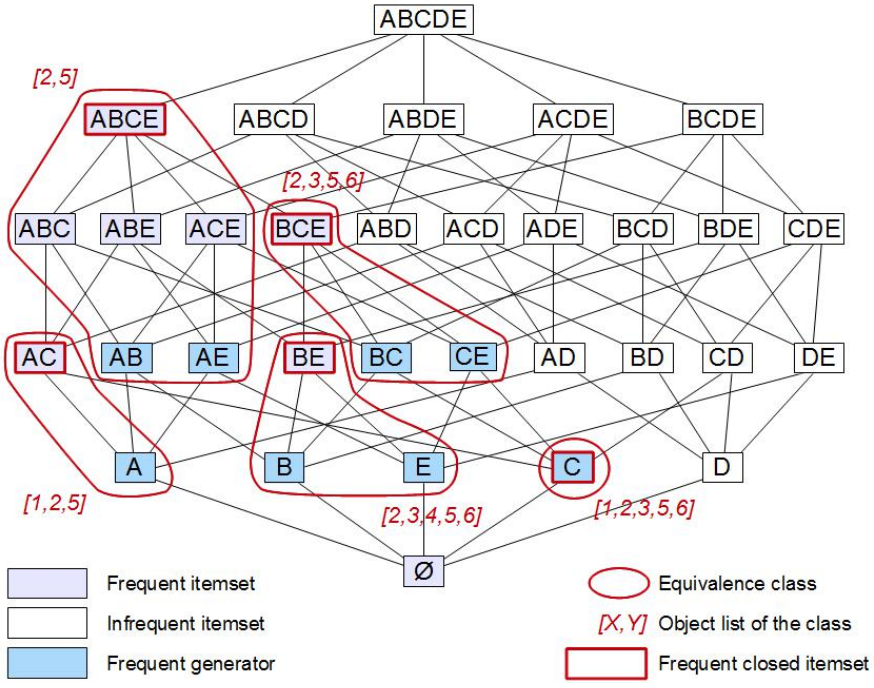


Figure 5.2: Equivalence Classes for Example Database D1 and *minsupport* = 2/6

iii) $\nexists o' \in O$ *with* $o' \notin P$ *such that* $\forall i \in I$ *we have* $i \in o'$.

The Hasse diagram of this dual lattice for example database D1 and *minsupport* $= 2/6$ is given in figure 5.3. Conceptual bi-clusters are related by the dual order relation $\Re(\subset, \supset)$ depicted by edges in the Hasse diagram. The bottom bi-cluster $\{\emptyset, \{1, 2, 3, 4, 5, 6\}\}$ is represented for completeness of the lattice and will not be extracted for this example as its intension is empty.

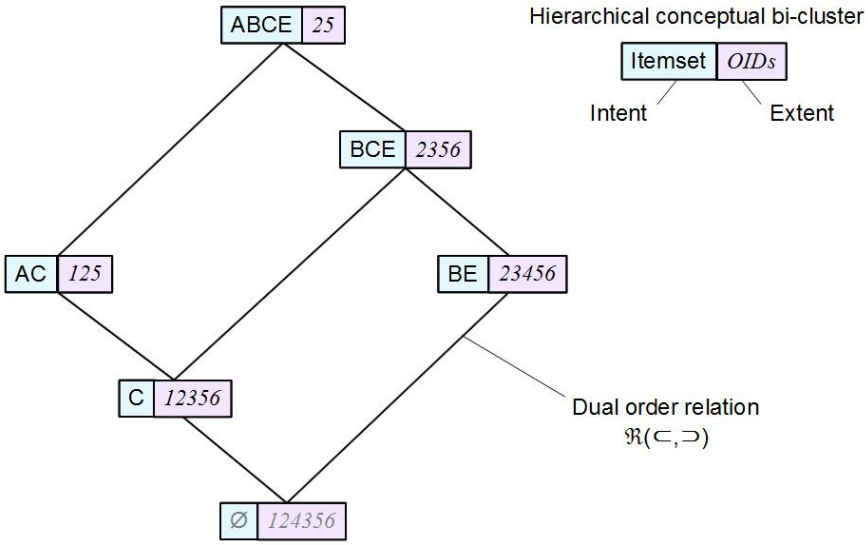


Figure 5.3: Hierarchical Conceptual Bi-clusters for Example Database D1.

From this hierarchy of conceptual bi-clusters, frequent generators are identified in level-wise manner and used to generate minimal covers of conceptual association rules. The correctness of this level-wise identification process relies on the properties that the set of frequent generators is an order ideal and that the supporting object list of a frequent generator is identical to the supporting object list of its closure.

Property 5.1 (Set of Generators is an Order Ideal) *Let* $\mathcal{G}_\mathcal{C} = \{G_1, \ldots, G_n\}$ *be the set of generators* $G_i \subseteq L$ *of a frequent closed itemset* $C \subseteq L$, $C \in \mathcal{FC}$. *The set of generators* $\mathcal{G}_\mathcal{C}$ *of the partially ordered set* $(\mathcal{G}_\mathcal{C}, \subseteq)$ *is an order ideal defined by the following conditions:*

i) For every G_i *in* $\mathcal{G}_\mathcal{C}$, $G_j \subseteq G_i$ *implies that* $G_j \in \mathcal{G}_\mathcal{C}$.

ii) For every $G_i, G_j \in \mathcal{G}_\mathcal{C}$ *with* $|G_i| > 1$ *and* $|G_j| > 1$, *there is some element* $G_k \in \mathcal{G}_\mathcal{C}$, *such that* $G_k \subseteq G_i$ *and* $G_k \subseteq G_j$.

Proof 5.1 (Property 5.1) *See* [Yahia 2006] *and* [Hamrouni 2008b].

Given property 5.1, the FIST approach optimizes the iterative search for generators of a frequent closed itemset by considered potential generators in increasing order of their size and stopping the iterations once the size of generators is identified. This corresponds to the use of the *gen_size* variable in algorithm 22 of FIST 2.0.

Proposition 5.1 (Generator Supporting Object List) *The supporting object list of a generator is identical to the supporting object list of its closure. Let $G_i \subseteq L$ be a generator of a frequent closed itemset $C \subseteq L$, $C \in \mathcal{FC}$, that is: $G_i \in \mathcal{G}_C$. The supporting object list of G_i is equal to the supporting object list of C: $\forall G_i \in \mathcal{G}_C$ we have $\psi(G_i) = \psi(C)$.*

Proof 5.2 (Proposition 5.1) *Let $\mathcal{G}_C = \{G_i \subseteq L \mid \Gamma(G_i) = C\}$ be the set of generators of the frequent closed itemset $C \subseteq L$, $C \in \mathcal{FC}$. Given definition 1.8, we have $\Gamma(C) = C$ and, since the Galois closure is monotonically increasing, we deduce that $\forall I \subseteq L$ such that $I \supset C$ we have $\Gamma(I) \supset \Gamma(C)$. Given definition 1.10, we have $\forall G_i \in \mathcal{G}_C$, $\Gamma(G_i) = \Gamma(C) = C$ and since $\forall I \supset C$ we have $\Gamma(I) \supset \Gamma(C)$, we deduce $G_i \subseteq C$. Since ψ is monotonically decreasing, we have $\psi(G_i) \supseteq \psi(C)$.*
We first show that all objects in the object list of C are present in the object list of G_i and second, that all objects in the object list of G_i are present in the object list of C:

(1) Suppose we have $\exists o \in O$ such that $o \supseteq C$ and $o \not\supseteq G_i$. We then have $o \in \psi(C)$ and $o \notin \psi(G_i)$ $\Longrightarrow \psi(G_i) \not\supseteq \psi(C)$ which is contradictory with $\psi(G_i) \supseteq \psi(C)$. □

(2) Suppose we have $\exists o \in O$ such that $o \not\supseteq C$ and $o \supseteq G_i$. Then, either we have $\exists i \in G_i \mid i \notin C$ $\Longrightarrow \Gamma(G_i) \neq \Gamma(C)$, or we have $o \subset C \Longrightarrow G_i \subseteq o \subset C \Longrightarrow \Gamma(G_i) \subset \Gamma(C) \Longrightarrow \Gamma(G_i) \neq \Gamma(C)$. □

Corollary 5.1 (Equivalence Class Supporting Object List) *The supporting object list of itemsets in an equivalence class is identical to the supporting object list of the frequent closed itemset of the class. Let $\mathcal{G}_C = \bigcup_{i=1}^{i=k} G_i$ be the set of generators of the frequent closed itemset C. The supporting object list of all itemsets I such that $I \supseteq G_i \in \mathcal{G}_C$ and $I \subseteq C$ is equal to the supporting object list of C: $\forall I \in E_C$ where E_C is the equivalence class which frequent closed itemset is C, we have $\psi(I) = \psi(C)$.*

Proof 5.3 (Corollary 5.1) *Let $\mathcal{G}_C = \bigcup_{i=1}^{i=k} G_i$ be the set of generators of the frequent closed itemset C and $I \in E_C$ where E_C is the equivalence class which frequent closed itemset is C. Given definitions 1.8 and 1.11, and proposition 5.1, we have:*

$P_0 :\ \Gamma(C) = \gamma(\psi(C)) = C$

$P_1 :\ \exists G_i \in \mathcal{G}_C \mid G_i \subseteq I \subseteq C.$

$P_2 :\ \psi(G_i) = \psi(C).$

We first show that all objects in object list of I belong to the object lists of G_i and C, and second, that all objects in object lists of G_i and C belong to the object list of I:

(1) Suppose we have $\exists o \in O$ such that $o \supseteq I$ and $o \not\supseteq G_i \Longrightarrow o \not\supseteq C$. Since $G_i \subseteq I \subseteq C$ we have $\psi(G_i) \supseteq \psi(I) \supseteq \psi(C)$. Since $o \notin \psi(G_i)$ and $o \in \psi(I)$ we have $\psi(G_i) \not\supseteq \psi(I)$ that is contradictory with $\psi(G_i) \supseteq \psi(I) \supseteq \psi(C)$. □

(2) Suppose we have $\exists o \in O$ such that $o \not\supseteq I$ and $o \supseteq G_i \Longrightarrow o \supseteq C$. Since $o \in \psi(C)$ and $o \notin \psi(I)$ we have $\psi(I) \not\supseteq \psi(C)$ that is contradictory with $\psi(G_i) \supseteq \psi(I) \supseteq \psi(C)$. □

From the frequent generators and the frequent closed patterns, bases of conceptual association rules are generated. The definitions of conceptual association rules and, of min-max exact and approximate, and proper approximate conceptual association rules are given below. These rules are defined for a database $\mathcal{D} = (O, L, R)$ and, a *minsupport* and a *minconfidence* threshold values.

Definition 5.2 (Conceptual Association Rules) *A* conceptual association rule r *is an implication rule between two frequent itemsets* $I_1, I_2 \subset L$ *in* $\mathcal{D}$ *such that* $I_1 \subset I_2$ *and* $support(I_2)/support(I_1) \geq minconfidence$, *denoted* r: $\{I_1 \longrightarrow I_2 \setminus I_1$, $support(r)$, $confidence(r)$, $object\_list(r)\}$. *The support of* r *is* $support(r) = support(I_2)$, *the confidence of* r *is* $confidence(r) = support(I_2)/support(I_1)$ *and the object list of* r *is* $object\_list(r) = \{o \in O \mid \forall i \in I_2, i \subset o\}$.

Definition 5.3 (Min-max Exact Conceptual Association Rules) *A* min-max exact conceptual association rule r *is an implication rule between a frequent generator* $G_i \in \mathcal{G}_C$ *and its closure* $C \in \mathcal{FC}$ *in* $\mathcal{D}$ *denoted* r: $\{G_i \longrightarrow C \setminus G_i$, $support(r)$, $confidence(r)$, $object\_list(r)\}$ *with* $support(r) = support(C)$, $confidence(r) = 1$ *and* $object\_list(r) = \{o \in O \mid \forall i \in C, i \subset o\}$.

Definition 5.4 (Min-max Approximate Conceptual Association Rules) *A* min-max approximate conceptual association rule r *is an implication rule between a frequent generator* $G_i \in \mathcal{G}_C$ *and a frequent closed itemset* $C' \in \mathcal{FC}$ *in* $\mathcal{D}$ *such that* $C \subset C'$ *and* $support(G_i)/support(C') \geq minconfidence$. *Such a rule is denoted* r: $\{G_i \longrightarrow C' \setminus G_i$, $support(r)$, $confidence(r)$, $object\_list(r)\}$ *with* $support(r) = support(C')$, $confidence(r) = support(G_i)/support(C')$ *and* $object\_list(r) = \{o \in O \mid \forall i \in C', i \subset o\}$.

Definition 5.5 (Proper Approximate Conceptual Association Rules) *A* proper approximate conceptual association rule r *is an implication rule between two frequent closed itemsets* $C, C' \in \mathcal{FC}$ *in* $\mathcal{D}$ *such that* $C \subset C'$ *and* $support(C)/support(C') \geq minconfidence$. *Such a rule is denoted* r: $\{C \longrightarrow C' \setminus C$, $support(r)$, $confidence(r)$, $object\_list(r)\}$ *with* $support(r) = support(C')$, $confidence(r) = support(C)/support(C')$ *and* $object\_list(r) = \{o \in O \mid \forall i \in C', i \subset o\}$.

The graphical representation of the min-max exact rules and other valid exact rules, for the example database D1 and *minsupport* = 2/6 is shown in figure 5.4. The graphical representation of the min-max approximate rules, proper approximate rules and other valid approximate rules for the three equivalence classes with top element {BCE}, {BE} and {C} and *minconfidence* = 4/5 is shown in figure 5.5. The valid rules depicted in blue color are additional rules, compared with the min-max bases, extracted by Apriori-like approaches. The proper approximate rules are depicted in dashed lines. The number of valid rules for this example is more than twice the number of rules in the bases. The object list associated to a conceptual association rule corresponds to the object list of the equivalence class of the itemset in the consequent of the rule.

5.3.2 Impact of Item Ordering on the FGIST

An important optimization of the FIST approach relies on the ordering in increasing support order of dataset variable values as items. This ordering importantly impacts the structure and the size, both in number of nodes and of information in their fields, of the FGIST. To illustrate this optimization, we consider another example database D2 which binary (A) and transactional (B) format representations are given in table 5.22.

This database is composed of five different variables {A, B, C, D, E} and five objects. We consider a *minsupport* value of 1/5, for which all items are frequent, for all examples presented in

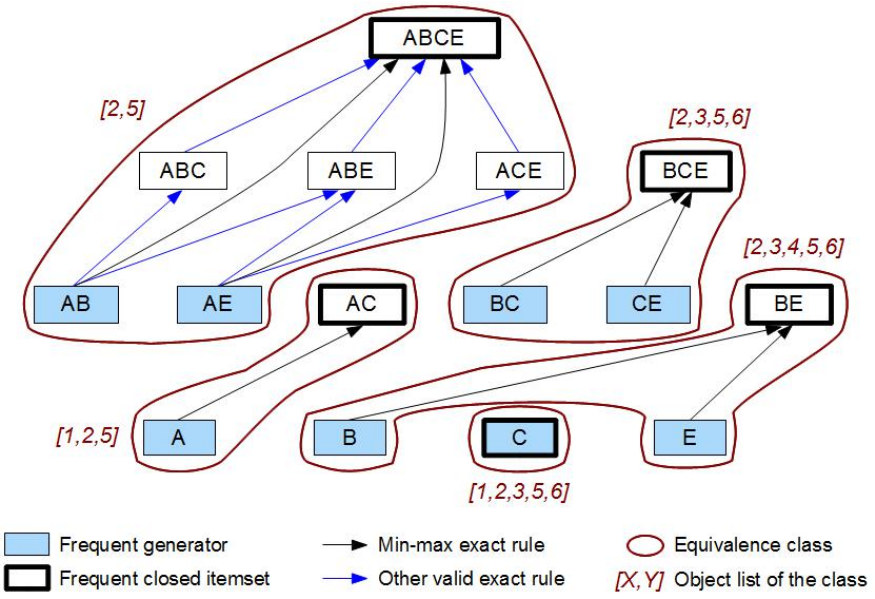


Figure 5.4: Exact Association Rules for Example Database D1

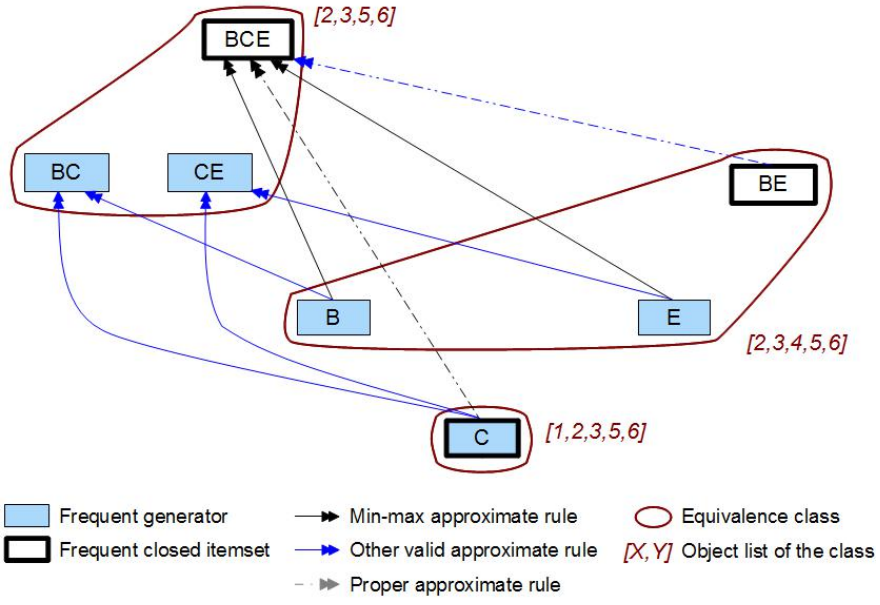


Figure 5.5: Approximate Association Rules for Example Database D1

Object	A	B	C	D	E
1	1	1	1	0	0
2	0	1	1	1	0
3	0	1	1	0	1
4	1	1	1	1	0
5	0	1	1	1	1

A. Binary Matrix Representation

Object	Items
1	A B C
2	B C D
3	B C E
4	A B C D
5	B C D E

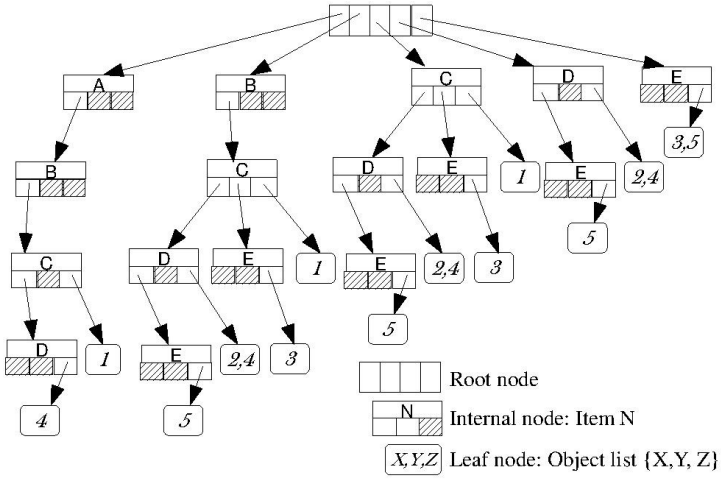
B. Transactional Representation

Table 5.22: Example Database D2

this subsection. The first possible ordering of items is the *lexicographic order*, which correspond to the order of columns in the binary format and of values in objects of the transactional format representations of D2. The corresponding item order (A), represented by the vertical order of items, and FGIST (B), after the dataset scan, for example database D2 are represented in figure 5.6. The support count of each item in the dataset is also represented in table (A).

Item	Support
A	2
B	5
C	5
D	3
E	2

A. Item Order



B. Frequent Generalized Itemsets Suffix-Tree

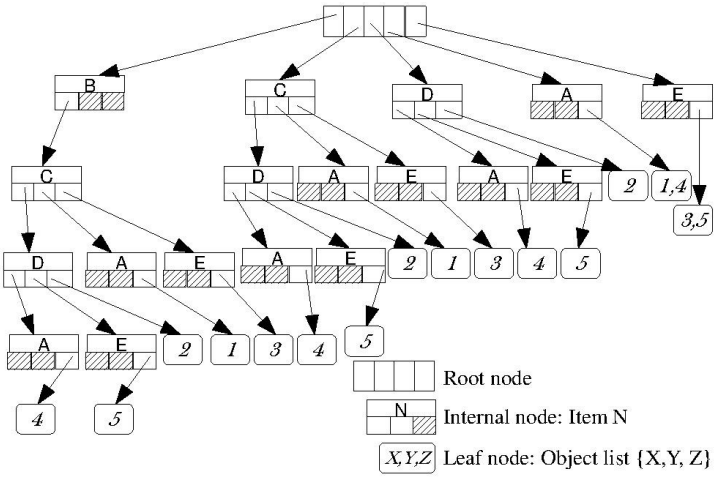
Figure 5.6: Lexicographic Ordering of Items for Example Database D2

The second ordering of items considered is the *decreasing supports order*, that is items corresponding to dataset values are sorted to have most frequent items firsts and least frequent items lasts. The corresponding item order (A) and FGIST (B) for example database D2 are represented in figure 5.7. With this ordering of items, we can see that almost all object lists of nodes, except for the two last items A and E, contain only one object identifier. This shows the information is broken-out and distributed among the nodes of the FGIST.

The last ordering of items considered is the *increasing supports order*, that is items corresponding to dataset values are sorted to have least frequent items firsts and most frequent items lasts. The corresponding item order (A) and FGIST (B) for example database D2 are represented in figure 5.8. We can see that with this ordering, the number of nodes in the

Item	Support
B	5
C	5
D	3
A	2
E	2

A. Item Order



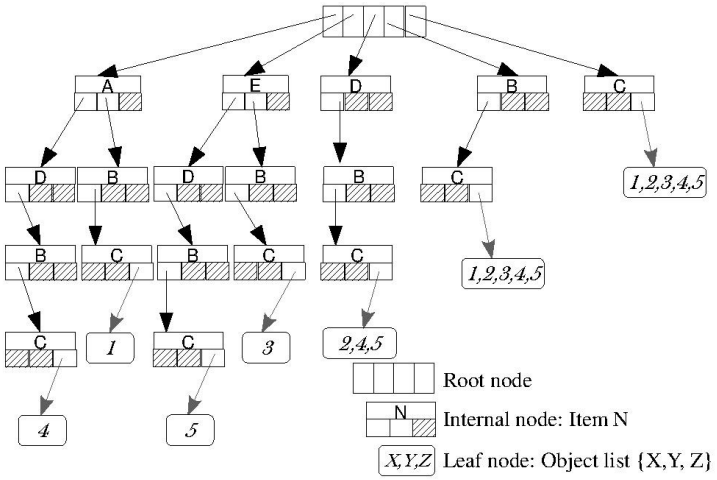
B. Frequent Generalized Itemsets Suffix-Tree

Figure 5.7: Decreasing Supports Ordering of Items for Example Database D2

FGIST is minimized and that the last itemsets in the FGIST (in the right part of the tree) have complete lists of supporting objects, contrarily to the two preceding item ordering cases. These object lists are by definition the largest ones and thus those that require the greatest number of computations for comparison and union operations.

Item	Support
A	2
E	2
D	3
B	5
C	5

A. Item Order



B. Frequent Generalized Itemsets Suffix-Tree

Figure 5.8: Increasing Supports Ordering of Items for Example Database D2

The comparison between the sizes of the three preceding FGISTs are presented in table 5.23. Sizes are compared in terms of total number of nodes in the tree, number of internal nodes and number of leaf nodes, representing frequent patterns, that are identified by their non-empty

supporting object list. The number of closed and non-closed frequent patterns in the FGIST are also presented in the last two column of the table.

Ordering	**Number of Nodes**			**Number of Patterns**	
	Total	**Internal**	**Leaves**	**Closed**	**Non-closed**
Lexicographic	29	16	13	6	7
Decreasing supports	33	18	15	5	10
Increasing supports	25	18	7	6	1

Table 5.23: Comparison of Sizes of the FGISTs for Example Database D2

For the considered *minsupport* value, all closed itemsets in the dataset are frequent and the six frequent closed itemsets are the following (in lexicographic order): {A,B,C}, {A,B,C,D}, {B,C}, {B,C,D}, {B,C,E} and {B,C,D,E}. These itemsets are all represented in the FGIST for the lexicographic and increasing supports orderings, and one ({B,C}) is missing for the decreasing supports ordering. The FGIST for the lexicographic and decreasing supports orderings contain respectively seven and ten non-closed patterns, that correspond to non-closed itemsets. The FGIST for the increasing supports ordering contains one non-closed pattern, corresponding to the non-closed itemset {C}. This peculiar case is due to the fact that C is present in all data rows, together with B; the closure of {C} and {B} is thus {B,C}.

Given the ordering of frequent items according to their supports, the sequential numeric identifiers assigned to items define a total order relation on the itemsets called *itemset lexicographic order relation* and denoted $\preceq_L$. Tests performed to compare two itemsets according to the lexicographic order relation during the object lists update and pruning of non-closed patterns have a major effect on the complexity, and thus the execution times, of the algorithm.

Definition 5.6 (Itemset Lexicographic Order Relation) *We say that for two items i and j, we have $i < j$ if i precedes j in the alphabetical order. Let $I = \{i_1, \ldots, i_p\}$ and $J = \{j_1, \ldots, j_q\}$ be two frequent itemsets $I, J \subseteq L$. We denote i_k and j_k the items in the k^{th} position in I and J. We have $I \prec_L J$ if one of the two following conditions is verified:*

i) $\exists i_k \in I$ and $j_k \in J$ such that $i_k < j_k$ and $\forall m < k$ we have $i_m \leq j_m$.

ii) $\forall i_k \in I$ and $j_k \in J$ we have $i_k = j_k$ and $|I| < [J|$.

The efficiency of the final FGIST creation depends partly on how many frequent closed patterns must be generated by intersection operations during the FGIST update step and on how many patterns must be pruned, using inclusion tests, during the FGIST pruning step. In order to minimize the number of these operations items are ordered in increasing supports order. This ordering efficiency is based on the observation that the probability for an itemset to be closed is proportional to its support in the dataset. This means that itemsets with higher support value are more likely to be closed. A frequent closed itemset corresponds to one of the two following categories: Frequent itemsets that are represented in the database as a complete object, i.e., containing no other item than those in the itemset, and frequent itemsets corresponding to the intersection of at least two itemsets in the database.

Conjecture 5.1 (Itemset Closure Property) *The probability for an itemset to be closed is proportionally relative to its support in the dataset. Let $I_1, I_2 \subseteq L$ be two frequent itemsets*

$I_1, I_2 \in \mathcal{F}$ in database $\mathcal{D}$. If support$(I_1) \leq$ support(I_2) then $P_{\mathcal{D}}(I_1 \in \mathcal{FC}) \leq P_{\mathcal{D}}(I_2 \in \mathcal{FC})$ where $P_{\mathcal{D}}(I_1 \in \mathcal{FC})$ and $P_{\mathcal{D}}(I_2 \in \mathcal{FC})$ are the probability for I_1 and I_2, respectively, to be closed itemsets in database $\mathcal{D}$.

Some partial evidence to support this conjecture is discussed in appendix A.1. It is combined with the identification of the potential most frequent itemsets to minimize the size of the initial FGIST. Informally, itemsets with the highest support value for a given set of items are those containing the most frequent items and having a minimal size. Consider for example, two itemsets $I_1 = \{i_1, \ldots, i_n\}$ and $I_2 = \{i_1, \ldots, i_n, i_{n+1}\}$ identical except for the last item i_{n+1} of I_2 that is absent of I_1. Let denote $P_{\mathcal{D}}(I)$ and $P_{\mathcal{D}}(i)$ the probabilities that itemset I and item i, respectively, are contained in an object $o \in O$. If we consider a finite set of mutually independent items L, and since $P_{\mathcal{D}}(i_{n+1}) \leq 1$, we have:

$$P_{\mathcal{D}}(I_2) = P_{\mathcal{D}}(I_1) \cdot P_{\mathcal{D}}(i_{n+1}) \Longrightarrow P_{\mathcal{D}}(I_2) \leq P_{\mathcal{D}}(I_1)$$

Consider now two itemsets $I_1 = \{i_1^1, \ldots, i_n^1\}$ and $I_2 = \{i_1^2, \ldots, i_n^2\}$ with $P_{\mathcal{D}}(i_\alpha^1) \geq P_{\mathcal{D}}(i_\alpha^2)$ for all $\alpha \in [1, n]$ and $|I_1| = |I_2|$. Each item i_α^1 in I_1 has a probability of occurrence greater than the item i_α^2 at the same position in I_2 and, I_1 and I_2 are of equal size. We have:

$$P_{\mathcal{D}}(I_1) = \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha^1) \geq P_{\mathcal{D}}(I_2) = \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha^2)$$

These elements, associated to the itemset lexicographic order relation, minimize the number of non-closed itemsets inserted in the FGIST during its initialization, while scanning the dataset. Given this ordering, items with highest support are last in the objects of the SFD. Consider for example an object $o = \{i_1, i_2, i_3\}$ in the SFD with support$(\{i_1\})$ < support$(\{i_2\})$ < support$(\{i_3\})$. From this object, the suffix of size one processed will be $\{i_3\}$ which is the most frequent 1-itemsets among $\{i_1\}$, $\{i_2\}$ and $\{i_3\}$, and thus the one most likely to be closed. The same reasoning applies for suffixes of size greater than one: For suffixes of size two, the suffix processed will be $\{i_2, i_3\}$ that is the most frequent among the three suffixes. This means that itemsets corresponding to objects of the Sorted Frequent Database that are inserted in the FGIST are the most likely to be closed among the possible suffixes.

Experiments were conducted to compare the size of the FGIST for the different item orderings and for databases with different properties (number of objects, variables and variable values, density, co-occurrence frequencies, etc.). Consequently, the FIST approach first order frequent items in the dataset in increasing supports order to create a transactional format condensed representation of the dataset: The Sorted Frequent Database. In case of equal support counts, the item order is their order of first occurrence in the dataset. This ordering of frequent items in increasing support values minimizes both the number of nodes in the FGIST and the number of patterns considered during the extraction process. Consequently, the number of operations performed, and thus the execution time, of the generation of the frequent closed patterns from the FGIST, that is strictly proportional to its number of nodes and patterns, is lessened.

5.3.3 Frequent Generalized Itemset Suffix-Tree Closure Property

The correctness and completeness of the sets of frequent conceptual patterns generated by the FIST approach relies on the closure property of the FGIST. This property can be divided into

two sub-properties. These sub-properties are defined according to patterns represented in the FGIST as branches, i.e. as a path from the root to a leaf node (node with a non empty object list). The first is that all frequent closed patterns in the dataset $\mathcal{D}$, constituted of a frequent closed itemset and its supporting object list, are represented by a branch in the final FGIST. This property is called *FGIST Completeness*.

Property 5.2 (Frequent Generalized Itemset Suffix-Tree Completeness) *The final FGIST contains all frequent closed patterns in database $\mathcal{D}$, each associating a frequent closed itemset and its supporting object list: If I is a frequent closed itemset then $\exists N \in FGIST$ such that for node N we have itemset(N) = I, where itemset(N) is the itemset corresponding to the branch from the root to the N node, and $P \neq \emptyset$ is the supporting object list of I.*

Proof 5.4 (Property 5.2) *We show by construction that $\forall I \subseteq L$, if $I \in \mathcal{FC}$ we have $I \in FGIST$. We distinguish the two categories of frequent closed itemsets defined hereafter: Those that correspond to complete objects of the database $\mathcal{D}$ and those that correspond to the intersection of objects in $\mathcal{D}$.*

- *Frequent closed itemsets that are equal to at least one complete object and included in at least minsupport objects of database $\mathcal{D}$: $\exists o \in O$ such that $I = o$ and $I \subseteq o_i$ where $o_i \in O$ for $i = \{1, \ldots, n\}$ and $n \geq minsupport$.*
- *Frequent closed itemsets corresponding to the frequent intersection of at least two objects of database $\mathcal{D}$ and included in at least minsupport objects of database $\mathcal{D}$: $I = \bigcap_{i=1}^{i=n} o_i$ where $o_i \in O$ for $i = \{1, \ldots, n\}$ and $n \geq minsupport$.*

First category frequent closed itemsets are inserted with their supporting object list in the FGIST during the scan of the dataset. This is performed by step 9 of algorithm 10 (section 4.2.2). Second category frequent closed itemsets are created in the FGIST during the FGIST update step since:

(1) Itemsets corresponding to database objects are inserted in the FGIST during dataset scan: $\forall I = o \in O$, $I \in FGIST$. □

(2) The intersection between two itemsets in the initial FGIST is inserted in the FGIST: $\forall I_1, I_2 \in FGIST$, $I_1 \cap I_2 \in FGIST$. This insertion is performed by step 6 of algorithm 12 (section 4.2.4). □

(3) All frequent intersections between itemsets in the updated FGIST is inserted in the FGIST: $\forall I = \bigcap_{i=1}^{i=n} I_i \in FGIST$, $I \in FGIST$. Successive insertions of itemsets resulting of intersections between two itemsets are performed by the HNode.intersect() *and* HNode.intersect2() *functions given in algorithms 14 and 15 respectively (section 4.2.4). These functions are called iteratively by the* HTree.Intersect() *function given in algorithm 13 until no new intersection is computed as indicated by the* change *variable.* □

The second property is that each branch from the root to a leaf node in the final FGIST represents a frequent closed pattern. This property is verified if no branch in the final FGIST represents an infrequent or non-closed pattern. An infrequent pattern corresponds to an itemset which supporting object list size is less than the *minsupport* value. A non-closed pattern corresponds to an itemset which object list is identical to the object list of another itemset that is one of its supersets. This property is called *FGIST Exactitude*.

Property 5.3 (Frequent Generalized Itemset Suffix-Tree Exactitude) *The final FGIST contains only frequent closed patterns: If I is an infrequent or non-closed itemset then $\nexists N \in FGIST$ such that for node N we have itemset(N) $= I$, where itemset(N) is the itemset corresponding to the branch from the root to the N node, and $P \neq \emptyset$ is the supporting object list of I ($P = \psi(I)$).*

Proof 5.5 (Property 5.3) *We show by construction this property, by considering the three different categories of itemsets inserted into the updated FGIST:*

- *Itemsets I that are exactly equal to at least one object of database $\mathcal{D}$: $I = o$ where $o \in O$.*
- *Itemsets I corresponding to the intersection of other itemsets in the FGIST: $I = \bigcap_{i=1}^{i=m} I_i$ where $I_i \in FGIST$ for $i = \{1, \ldots, m\}$.*
- *Itemsets corresponding to suffixes S of an itemset I included in at least one object of database $\mathcal{D}$: $S \subset I$ with $I = o$ where $o \in O$.*

The second category corresponds to itemsets resulting of successive intersections between ordered itemset pairs that are inserted in the FGIST by step 5 of algorithm 12 with the union of the object lists of the two intersected itemsets. Among itemsets in these three categories, only frequent closed itemsets are present in the FGIST after the FGIST pruning step:

(1) Itemsets in the first category are closed itemsets. The suffix corresponding to each of these itemsets is inserted in the FGIST by step 9 of algorithm 10 with the object identifier row_num *for each object of the dataset containing it (section 4.2.2). Such closed itemsets that are infrequent are then pruned from the FGIST by steps 3 to 5 of algorithm 16 (section 4.2.5) testing the size of the object list for all itemsets represented in the updated FGIST.* □

(2) Itemsets in the second category are closed itemsets but can be infrequent. These infrequent closed itemsets are pruned from the FGIST by steps 3 to 5 of algorithm 16 (section 4.2.5) testing the size of the object list for all itemsets represented in the updated FGIST. □

(3) Itemsets in the third category can be non-closed. These non-closed itemsets are pruned from the FGIST by steps 6 to 10 of algorithm 16 (section 4.2.5): $\forall I_1, I_2 \in FGIST$ with $I_1 \prec_L I_2$, if $I_1 \subset I_2$ and $\psi(I_1) = \psi(I_2)$ then I_1 is non-closed and pruned from the FGIST. Resulting closed itemsets can be infrequent. These infrequent closed itemsets are pruned from the FGIST by steps 3 to 5 of algorithm 16 (section 4.2.5) testing the size of the object list for all itemsets represented in the updated FGIST. □

Given properties 5.2 and 5.3, the final FGIST contains all and only frequent closed patterns. These patterns are then used to generate the complete sets of conceptual knowledge patterns defined in definitions 5.2, 5.3, 5.4 and 5.5. These results have been confirmed by experiments conducted on several synthetic and benchmark databases with different properties (number of objects, variables, items, density of the matrix, correlation between co-occurrences of values, etc.). During these experiments, the completeness and correctness of the sets of patterns generated by FIST were checked by comparison with results from algorithms for extracting frequent closed itemsets, generators, classical association rules and bases of association rules, and automatized checking of supporting object lists of conceptual patterns in the initial dataset.

Part III

Experimental Results and Conclusion

Chapter 6

Experimental Data and Results

Contents

This chapter is devoted to the description of databases used for the experiments conducted and experimental results obtained for these databases. These databases were used to evaluate the performances and applicability of the FIST algorithm, compare them with respect to other state of the art pattern mining algorithms and evaluate knowledge patterns extracted by FIST from a biological interpretation viewpoint. The databases used for the experiments are described in section 6.1. Performance evaluations of the FIST approach algorithmic versions and comparison with other state of the art algorithms are presented in section 6.2. The evaluation and biological interpretation of knowledge patterns extracted by FIST are presented in section 6.3.

6.1 Experimental Databases

Experiments were conducted on five bioinformatics databases. The two first databases are related to genomics and Yeast gene expression data and contain gene expression data and, biological and bibliographic annotations of genes. These databases are presented in section 6.1.1. The three other databases are related to interaction proteomics and HIV-1 and human protein-protein interaction. They contain interaction proteomics data and, biological and bibliographic annotations of proteins. They were constructed by integrating heterogeneous data from different proteomic data and knowledge bases. These databases are presented in section 6.1.2.

The reason for using these databases is that knowledge patterns generated by the FIST approach are particularly relevant in the field of bioinformatics and biological analysis, as was shown in the literature, although they can also be relevant for any other application domain. However, contrarily to the FIST implementations, the format of the dataset needs to be transformed according to the implementation constraints (transactional format only, impossibility to process variables represented on several columns, representation of items as integer numbers only, etc.) for most available implementations of algorithms. These problems of applicability to different data formats render difficult the process of experimentations as time consuming transformation operations of datasets are required. This can also be a usage limiting factor for operational applications.

The main characteristics of these databases are summarized in table 6.1. This table show the number of rows, the number of variables (number of information types), the number of column (number of different data values, or items), and the maximal size of rows (maximal number of items on one row) for each of these databases. The number of data values corresponds to the number of columns in the case of a data matrix representation of the database. These databases and their detailed descriptions are available at `http://keia.i3s.unice.fr`.

Table 6.1: Principal Characteristics of Experimental Databases

Database	**Number of rows**	**Number of variables**	**Number of columns**	**Maximal row size**
Eisen gene expression data	2465	79	237	43
Eisen integrated data	2465	737	9918	354
HIV-1–Human binary interactions	1433	19	19	11
HIV-1–Human integrated data	1433	149	3839	136
HIV-1–Human interaction types	1433	327	327	31

6.1.1 Yeast Gene Expression Databases

One of the main objectives of gene expression analyses is to discover information about biological processes that command cell behavior. A major task in this goal is the interpretation of gene expression profiles in the light of biological knowledge represented as gene annotations in biological data and knowledge bases. This task aims at detecting gene groups that are both co-expressed, i.e. that share similar expression profiles, and co-annotated, i.e. that share the same biological annotations such as functions, regulatory mechanisms, etc.

The two Yeast gene expression datasets used for experiments are based on the well known original Eisen *et al.* data [Eisen 1998]. These data were intensively studied in the literature, allowing us to compare results obtained with the FIST approach to existing information extracted from these data that were confirmed and validated by biologists.

Yeast Gene Expression Data The first gene expression database used for the experiments corresponds to the original Eisen *et al.* Yeast dataset. This dataset contains expression measures for 2465 yeast genes under 79 biological conditions extracted from a collection of four independent microarray studies about the *Saccharomyces cerevisiae* organism during several

biological processes: cell cycle experiments, sporulation experiments, temperature shock experiments, and diauxic shift.

The dataset constructed from these data contains discretized gene expression values, in terms of over-expressed, under-expressed and unchanged expression, for the 2465 yeast genes under the 79 different biological conditions. The gene expression measures were discretized using the normalized discretization Nordi algorithm [Martinez 2009]. Implementation of the Nordi algorithm available at http://keia.i3s.unice.fr/?Logiciels_et_Impl%C3%A9mentations___Nordi. It contains 2464 rows corresponding to the yeast genes and 79 variables, represented as columns of the data matrix, corresponding to the experimental biological conditions. Detailed description of this dataset and experimental results obtained by application of association rule mining are available at http://keia.i3s.unice.fr/?Jeux_de_Donn%C3%A9es___Eisen_et_al._Dataset.

Integrated Yeast Gene Expression Data and Annotations The second dataset results from the integration with the Eisen Yeast gene expression data of biological and bibliographical annotations. Each Yeast gene was annotated with the Gene Ontology (GO) identifiers of its associated terms in Yeast GO Slim [1], that is a yeast-specific cut-down version of Gene Ontology, the PubMed identifiers representing its associations with research papers, the identifiers of the KEGG pathways in which it is involved, its phenotypes expressed in plain text and the names of the transcriptional regulator genes that bind its promoter regions. Overall, the following gene annotations were included in the data matrix as columns: 76 GO, 97 different pathways, 109 transcriptional regulators, 1776 phenotypes and 7623 PubMed identifiers. For each gene, up to 25 GO, 14 pathways, 25 transcriptional regulators, 14 phenotypes and 581 PubMed identifiers are included as annotations. To take into account the hierarchical structure of GO, each gene is associated with all its annotations, including direct and inherited annotations. The resulting dataset is a matrix of 2465 lines representing yeast genes and 737 columns representing discretized gene expression measures and gene annotations.

This dataset was used to assess the scalability of the FIST approach and compare its applicability versus execution times and memory usage with those of other algorithms for datasets with a large number of columns. Detailed description of this dataset and experimental results obtained by application of association rule mining algorithms are given in [Martinez 2007]. Knowledge patterns extracted from this dataset showed significant co-annotated and co-expressed gene patterns, depicting important biological relationships between genes and their features. Several of these relationships are supported by recent biological literature.

6.1.2 HIV-1 and Human Protein-Protein Interaction Databases

Proteomics focuses on the large scale study of proteins in complex biological systems (fluid, tissue, organ, etc.). As most of the high throughput "omics" fields, proteomics is facing a tremendous increase of the size of datasets to process (number of items as well as number of variables, but also their meta-data which structure the datasets into complex objects), leading to challenging knowledge discovery from data problems. The FIST approach was applied for the analysis of a real life dataset of protein-protein interactions (PPI) between HIV-1 and Human proteins. Discovering protein-protein interactions is a recent major challenge in computational

[1] http://www.geneontology.org/GO.slims.shtml

biology. Identifying interactions among proteins was shown to be useful for finding new drugs and preventing several kinds of diseases. The identification of interactions between HIV-1 proteins and Human proteins is a particular PPI problem whose study might lead to the discovery of important interactions responsible for AIDS and help designing drugs and treatments.

In order to extend knowledge patterns extracted from original HIV-1 and human PPI data, we constructed three new databases integrating the most recent protein-protein interaction information and, biological and bibliographical related knowledge. The first is the binary interaction database that was constructed with the most up to date literature knowledge on this topic. The second database was built by integrating biological and bibliographical protein annotations with protein-protein interaction data. The third database contains HIV-1 and human protein-protein interactions with their types represented as directed relationships between proteins of both organisms. The datasets generated from these three databases, in different data representation formats, are available at `http://keia.i3s.unice.fr/`. Experimental results for these PPI databases, and new information discovered using the FIST approach on these databases, are presented in the sections 6.2 and 6.3 of this chapter.

Binary Interaction Database Several experimental methods have been developed to identify protein interactions and different databases were created to classify the large collections of experimental data generated [Jager 2011, Shoemaker 2007a]. The most prominent source of information on HIV-1 and human PPI is the *HIV-1, Human Protein Interaction Database* of the National Institute of Allergy and Infectious Diseases (NIAID) [Fu 2009, Pinney 2009, Ptak 2008]. This database was developed, starting from year 2000, from the peer-reviewed scientific literature in collaboration with the Southern Research Institute and the National Center for Biotechnology Information. It contains information on HIV-1 and human host cell proteins known interactions based on literature reports. This database contains only previously identified interactions: the absence of an interaction between two specific HIV-1 and human proteins in the database does not imply that they do not interact but only that no interaction between them was reported.

The first PPI database used for experiments conducted with the FIST approach was constructed from the HIV-1-Human Protein Protein Interaction Database of the NIAID available at `http://www.ncbi.nlm.nih.gov/RefSeq/HIVInteractions/`. Several datasets were derived from this data repository to represent data in both transactional and, binary and unary matrix formats. These datasets contain 1433 rows corresponding to the human proteins studied and, on the whole, 2586 interactions between HIV-1 and human proteins as data values. Matrix format datasets contain 19 columns corresponding to the different HIV-1 proteins: RETROPEPSIN, TAT, NEF, MATRIX, RT, ENV_GP41, NUCLEOCAPSID, REV, VPR, ENV_GP120, INTEGRASE, GAG_PR55, VPU, ENV_GP160, CAPSID, VIF, POL, P6 and P1. In the binary matrix format dataset, each cell of the matrix contains a 1 if there is a positive interaction between the corresponding pair of proteins and a 0 if no interaction was reported in the literature. In the unary matrix format dataset, each cell of the matrix contains a 1 if there is a positive interaction between the corresponding pair of proteins and a question mark if no interaction was reported in the literature. In the transactional format dataset, each of the 1433 row contains the list of HIV-1 protein identifiers, among the 19 HIV-1 proteins, that are known to interact with the human protein corresponding to the row.

Integrated Protein Interactions and Annotations Database This database was constructed by integrating biological annotations and related publications for human proteins with the HIV1-Human protein protein interaction network data of the NCBI Reference Sequences (RefSeq) collection available at http://www.ncbi.nlm.nih.gov/projects/RefSeq/. The Reference Sequence collection objective is to provide an exhaustive, non-redundant, integrated and well-annotated set of protein sequences, including DNA, transcripts and proteins.

Annotations were collected from the Gene Ontology, PubMed and Reactome knowledge bases. Gene Ontology (GO) biological annotations of human proteins from the UniProtKB-GOA (GO Annotation@EBI) database were collected from the Gene Ontology web site at http://www.geneontology.org/GO.downloads.annotations.shtml. GO annotations with TAS evidence code [2] were integrated in the data. Gene Ontology annotations with TAS evidence code are are the most reliable biological annotations as they were manually validated by biologists and cited in published biological references. Publication annotations were collected from the NCBI web site at http://www.ncbi.nlm.nih.gov/sites/entrez and PubMed and Reactome publications related to the GO biological annotations of human proteins were also integrated in the dataset as new variables. The Reactome knowledge base [3] is a manually curated and peer-reviewed pathway database. It contains pathway annotations validated by expert biologists and cross-referenced to several bioinformatics data and knowledge bases.

These data were collected for the 1433 human proteins and the 19 HIV-1 protein. Each of these 19 HIV-1 proteins is represented by one column in the resulting data matrix. In these columns, a positive interaction between the HIV-1 protein and the human protein corresponding to the row is represented by a "1" value, and the absence of knowledge of an interaction is represented by a "?" value. Each type of annotations, i.e., GO, PubMed and Reactome annotations, is represented by several columns in the resulting data matrix. In these columns, an annotation of the human protein corresponding to the row is represented by the identifier of the annotation. Overall, this database contains 1149 distinct GO annotations and 2670 distinct publication annotations, and up to 40 GO annotations and 88 publication annotations for each protein. It was used to assess and compare the scalability and applicability of the FIST approach with those of other state of the art pattern extraction methods for datasets with a large number of variables.

Interaction Types Database The initial analysis of HIV-1 and human interaction data show that about one third of these interactions are direct physical interactions, such as "binding", and two thirds are indirect interactions, such as up-regulation through "activation of signaling pathways" [Ptak 2008]. The proteins that the HIV-1 virus targets depend on interaction relationships between human proteins, as the virus makes use of the existing communication pathways within the cell [Tastan 2009]. A better understanding of the nature of the interaction relationships between proteins of both organisms can thus contribute to determine how the virus uses the human proteins and communication pathways to infect the organism. Knowledge patterns taking into account interaction types between the virus and host proteins can give important insights in regard with this objective.

This database depicts the interactions between 19 HIV-1 proteins and 1433 human proteins with directed interaction types. Directed interaction types show the biological na-

[2] http://www.geneontology.org/GO.evidence.shtml
[3] http://www.reactome.org

ture of the interaction relationship existing between the two types of proteins. An example of such interaction types is "TAT activated by IL2" meaning that the interaction relationship between the TAT HIV-1 protein and the IL2 human protein is "activated by". Another example is "TAT activates AKT1" means that the TAT HIV-1 protein "activates" the AKT1 human protein. Overall, this database contains 327 HIV-1 protein interaction types represented as combinations of an HIV-1 protein identifier, e.g. "TAT", and an interaction type, e.g. "activated by". Interaction type data between HIV-1 and human organisms were collected from the *H*IV-1, Human Protein Interaction Database of the NIAID, provided in collaboration with the Southern Research Institute and NCBI, available at http://www.ncbi.nlm.nih.gov/projects/RefSeq/HIVInteractions/. The complete list of interaction types integrated in this database is given in Appendix B.1.

6.2 Performance Evaluations

In this section, we present the performance evaluations of the implementations of the FIST algorithms and compare them with other state of the art frequent itemset mining, frequent closed itemset mining and bi-clustering algorithms. These experiments were conducted on the datasets described in section 6.1.

6.2.1 Experimental Design

The two algorithmic versions of the FIST approach were implemented in Java language for portability reasons. Java is at present the most predominant and popular language in computer science applications as shown in numerous studies available on the Internet. For example, it represents 18.16% of projects actually developed according to the March 2013 study of TIOBE that checked more than 211 million lines of software code[4]. It also represents 26% of the 10,000 validated projects gathered since 2008 by QSM[5] that is the largest, most complete set of historically completed software projects. The first version was implemented using standard Java Collections API (version named FIST 1.0) and the second version was implemented using both standard Java Collections API (version named FIST 2.0) and the Trove for Java Collections API (version named FIST 3.0).

Experiments were achieved to compare the different version of FIST to six state of the art algorithms for frequent itemsets, or frequent closed itemsets, based association rules mining: Apriori, Charm, DCI-Closed, Eclat, JClose and Zart. The Apriori and Zart algorithms are based on the frequent itemsets framework. The Charm, DCI-Closed and Eclat algorithms are based on the frequent closed itemsets framework. Optimized Java implementations of Apriori, Charm, DCI-Closed, Eclat and Zart used for these experiments are available at http://www.philippe-fournier-viger.com/spmf/. The JClose algorithm is a frequent closed itemsets based algorithm that generates bases of association rules. The Java implementation of JClose used for these experiments is available at http://keia.i3s.unice.fr/?Logiciels_et_Impl%C3%A9mentations___JClose. As the statistical measures and minimum thresholds, only the support and confidence, and the *minsupport* and *minconfidence* thresholds, were used for defining and evaluating the outputs. This in order to make possible the

[4] http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
[5] http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

comparison with association rule mining algorithms that all have these parameters in common. Bi-clusters were thus selected according to their support (size of their extent) only, according to the *minsupport* threshold, and both support and confidence were used for generating the rules. Other parameters of FIST for selecting bi-clusters, that is, minimal and maximal sizes of the intent and the extent, were not tested during these experiments. As a consequence, the most complete possible result was generated by the FIST implementations for each experiment.

Experiments were run on two different computers to evaluate the applicability of the proposed approach in different contexts. The first, used for the primary evaluation of the approach, is a Dell portable personal computer with an Intel Core 2 Duo (T5670 Series) processor at 1.80 GHz and 4 GB of DDR2 RAM running under the 32 bits Windows 7 Professional Edition operating system. The results of this first series of experiments are presented in [Mondal 2012b]. The second is a Dell PowerEdge R710 server with 2 Intel Xeon X5675 processors at 3.06 GHz, each possessing 6 cores, 12 MB cache memory, 24 GB of DDR3 RAM at 1333 MHz and 2 Hot Plug SAS hard disks of 600 GB at 15000 rnds/min with RAID 0 running under the 64 bits CentOS Linux operating system. For each experiment, that is the execution of an algorithm for specific parameter values and a given dataset, ten runs were performed and the measures given are the averages of these series of runs.

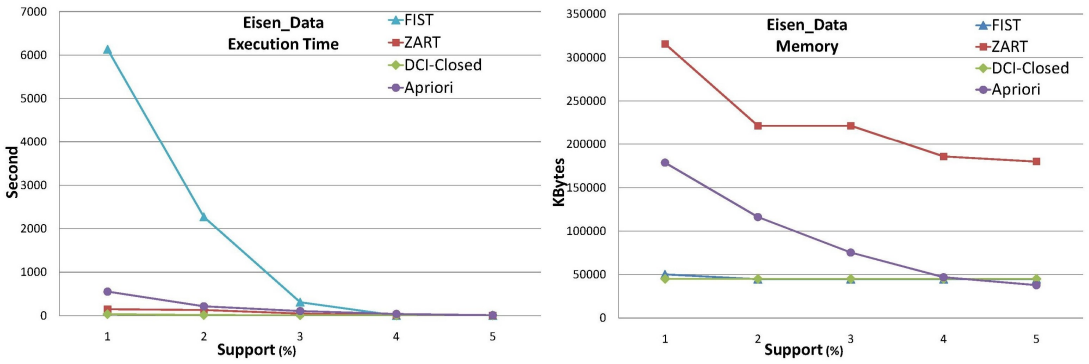
6.2.2 Results for Gene Expression Databases

In this section, we present experimental results in terms of execution times and memory usage for the two datasets derived from the Eisen *et al.* yeast data presented in section 6.1.1. The *Eisen Yeast Gene Expression* dataset is the matrix of 2465 yeast genes in rows and 79 biological conditions in columns. The *Integrated Gene Expression Data and Annotations* dataset is the matrix of 2465 rows representing yeast genes and 737 columns representing discretized gene expressions and gene biological and bibliographic annotations.

6.2.2.1 Eisen Yeast Gene Expression Data

The first series of experiments was performed on the Eisen Yeast gene expression dataset. They show the comparison between the first algorithmic version of the FIST approach and three other state of the art algorithms for association rule mining: Apriori, Zart and DCI-Closed. For these experiments, the *minconfidence* threshold was set to 0.001 (0.1%) and the *minsupport* threshold was varied from 0.01 (1%) to 0.05 (5%). These experiments were run, as a preliminary evaluation of the FIST approach, on the portable personal computer described above. Execution times (A) and memory usage (B) for the four algorithms for this series of experiments are presented in Figure 6.1.

We can see that execution times of FIST 1.0 are higher for very low *minsupport* threshold values. This is because of the exponential number of bi-clusters generated when *minsupport* is lessened. However, execution times remain acceptable in all cases, ranging from seconds to minutes. Except for these challenging cases, execution times of FIST are equivalent to those of the three other algorithms, even if it generates much more patterns. We can also see that memory consumption of FIST 1.0 are lower than those of the Apriori and Zart frequent itemsets based approaches and are equivalent to those of the DCI-Closed frequent closed itemsets based approach.



A. Execution times. B. Memory usage.

Figure 6.1: Experimental Results for the Eisen Yeast Gene Expression Dataset (series 1)

Experiments in the second series for this dataset were run on the PowerEdge server computer. For this series of experiments, we compared the three versions of FIST with the Java implementation of the Apriori, Charm, DCI-Closed, Eclat, JClose and Zart algorithms. Due to the difference of performances between the server and the personal computer, we were able to run experiments for very low *minsupport* values, with a minimal value of 0.001 (0.1%). Execution times for the nine algorithms for this series of experiments are presented in Figure 6.2. These results show the execution times of the generation of frequent itemsets for Apriori and Zart, and of frequent closed itemsets for the other seven algorithms, that are the most computationally intensive phases of the frequent pattern extraction. It should be noted that the *minsupport* values on the horizontal axis of the curves are not on a linear scale.

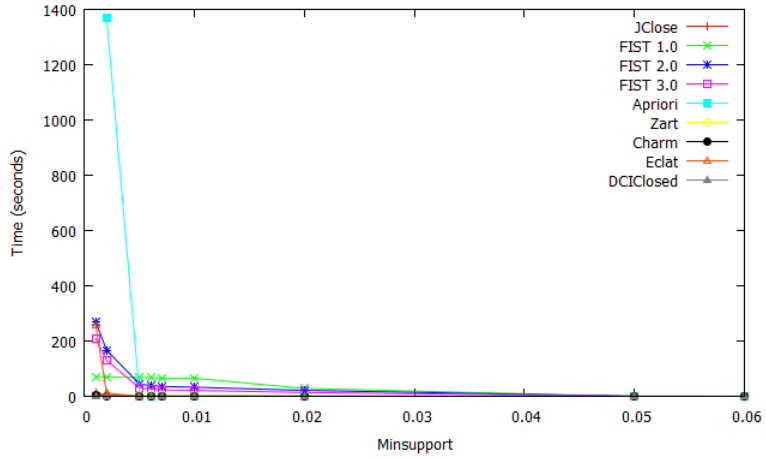


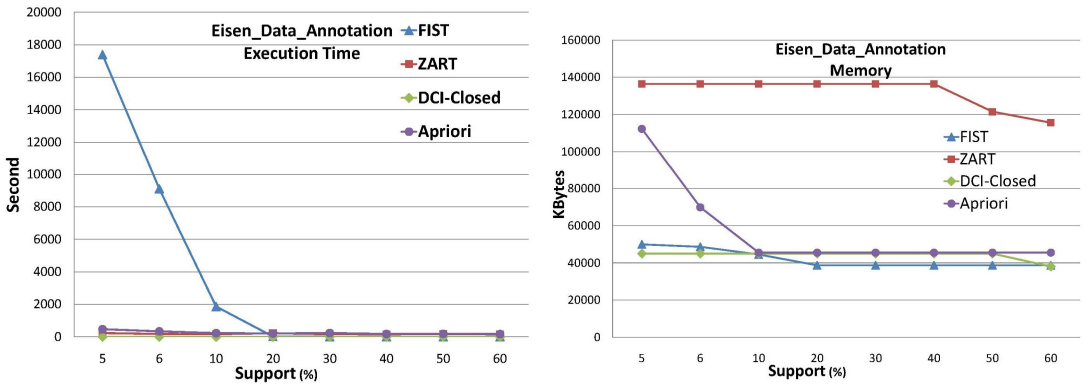
Figure 6.2: Experimental Results for the Eisen Yeast Gene Expression Dataset (series 2)

Executions of Apriori for *minsupport* = 0.001 (0.1%) and of Zart for *minsupport* = 0.001 (0.1%) and 0.002 (0.2%) were interrupted after several hours of run. The maximal execution

times obtained for this series of experiments are for the Apriori execution for *minsupport* = 0.2% with an average measure of ten runs slightly below 1400 seconds. These results show that the frequent itemsets based approaches can face efficient problems for large datasets which limit their applicability. On the contrary, frequent closed itemsets based approaches were able to process this dataset for *minsupport* = 0.001 (0.1%) with execution times ranging from tens of seconds to less than 5 minutes for Eclat and FIST 2.0 that are the most time consuming.

6.2.2.2 Integrated Gene Expression Data and Annotations

The first series of experiments on the integrated yeast gene expression data and annotations dataset was performed on the portable personal computer. They were achieved with the first version of FIST and the Apriori, Zart and DCI-Closed algorithm implementations. The *min-confidence* threshold was set to 0.001 (0.1%) and the *minsupport* threshold was varied between 0.01 (1%) and 0.5 (50%). Execution times (A) and memory usage (B) for the four algorithms for this series of experiments are presented in Figure 6.3.



A. Execution times. B. Memory usage.

Figure 6.3: Experimental Results for the Integrated Eisen Yeast Dataset (series 1)

We can see that execution times of FIST 1.0 are important for a *minsupport* threshold, due to the large number of additional knowledge patterns extracted by FIST compared to other algorithms. We can also see that memory usage of FIST 1.0 are equivalent to those of DCI-Closed and lower than both Apriori and Zart memory usage. One should note the exponential increase of the Apriori memory usage as the *minsupport* threshold is lessened.

The second series of experiments for this dataset was run on the PowerEdge server computer. Execution times for this series of experiments, performed with the three versions of FIST and the Apriori, Charm, DCI-Closed, Eclat, JClose and Zart algorithms, are presented in Figure 6.4. In this graphic also, the *minsupport* values on the horizontal axis are not on a linear scale.

These results show that FIST 1.0 is the worst performer, with a maximal execution time of about five hours, and the second worst performer is Zart, with a maximal execution time of about two hours and a half, for a *minsupport* threshold of 0.001 (0.1%). For this *minsupport* threshold, all other algorithms have execution times lower than twenty four minutes, that is nearly the execution times of Apriori and FIST 2.0, whereas FIST 3.0 required about twelve

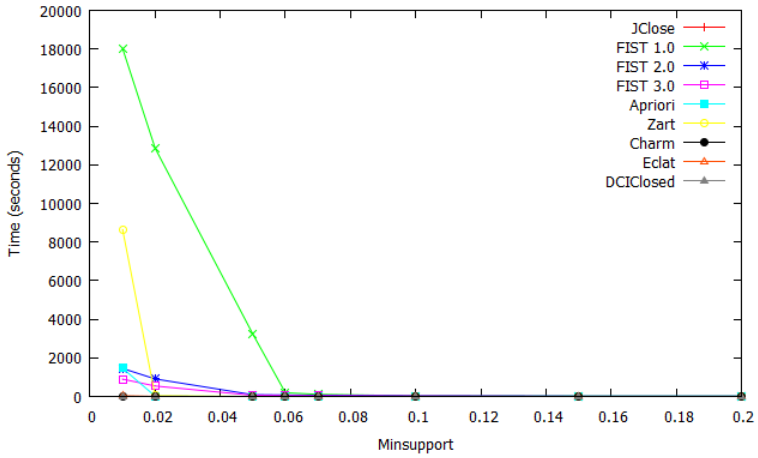


Figure 6.4: Experimental Results for the Integrated Eisen Yeast Dataset (series 2)

minutes to generate frequent patterns. For *minsupport* threshold values of 0.006 (0.6%) and higher, execution times of all algorithms are lower than three minutes.

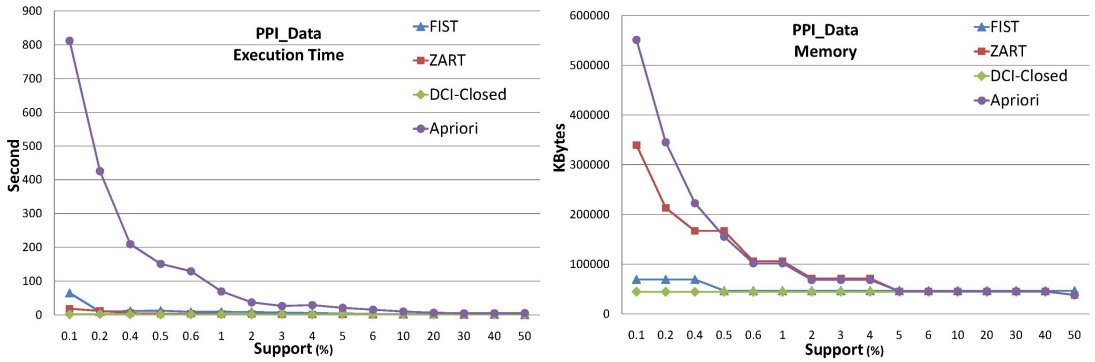
6.2.3 Results for Protein-Protein Interaction Databases

This section presents results, in terms of execution times and memory usage, for the experiments conducted for two datasets on HIV-1 and human PPI data presented in section 6.1.2. The *HIV-1 and Human Protein-Protein Interaction Data* dataset corresponds to the matrix of 1433 human proteins in rows and 19 HIV-1 proteins in columns. The *Integrated Protein-Protein Interaction Data and Annotations* dataset corresponds to the matrix of 1433 rows representing human proteins and 149 columns representing protein interactions and biological and bibliographic annotations.

6.2.3.1 HIV-1 and Human Protein-Protein Interaction Data

The first series of experiments on this dataset was performed on the portable personal computer with the first version of the FIST approach and, the Apriori, Zart and DCI-Closed algorithm implementations. The *minconfidence* threshold was set to 0.001 (0.1%) and the *minsupport* threshold was varied between 0.01 (1%) and 0.5 (50%). Execution times (A) and memory usage (B) for the four algorithms for this series of experiments are presented in Figure 6.5.

These results show that Apriori has the maximal execution times for all *minsupport* threshold values tested, with a maximal value of approximately fourteen minutes for the 0.001 (0.1%) threshold value. They also show that DCI-Closed, Zart and FIST 1.0 have equivalent execution times except for the 0.001 (0.1%) threshold value for which FIST 1.0 execution times are about one minute whereas DCI-Closed and Zart have execution times of approximately ten and twenty seconds respectively. Considering memory usage results, we can see that Apriori and Zart have the higher requirements, comprised between 350 MB and 550 MB for *minsupport* value of 0.001 (0.1%), whereas those of DCI-Closed and FIST 1.0, comprised between 50 MB



A. Execution times. B. Memory usage.

Figure 6.5: Experimental Results for the HIV-1–Human PPI Dataset (series 1)

and 70 MB for this *minsupport* value, are much lower when *minsupport* falls below 0.05 (5%).

The second series of experiments for this dataset were run on the PowerEdge server computer to compare the three versions of FIST with the Java implementations of Apriori, Charm, DCI-Closed, Eclat, JClose and Zart. For these experiments, the *minsupport* threshold values was varied betwwen 0.001 (0.1%) and 0.5 (50%). Execution times for the nine algorithms are presented in Figure 6.6 in which the *minsupport* values on the horizontal axis are not on a linear scale.

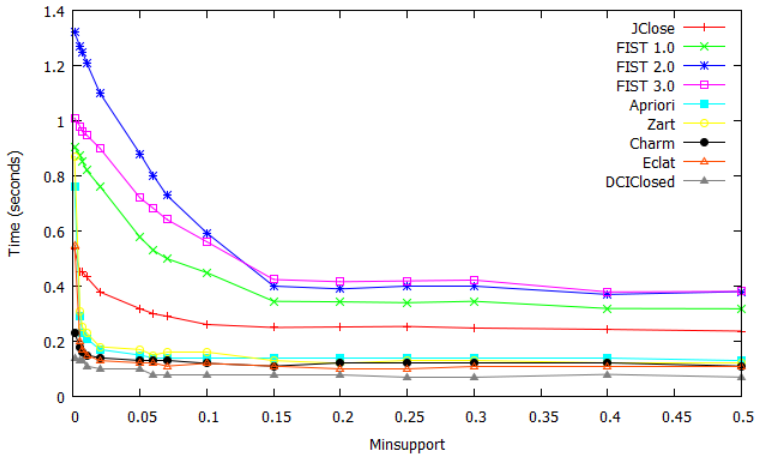


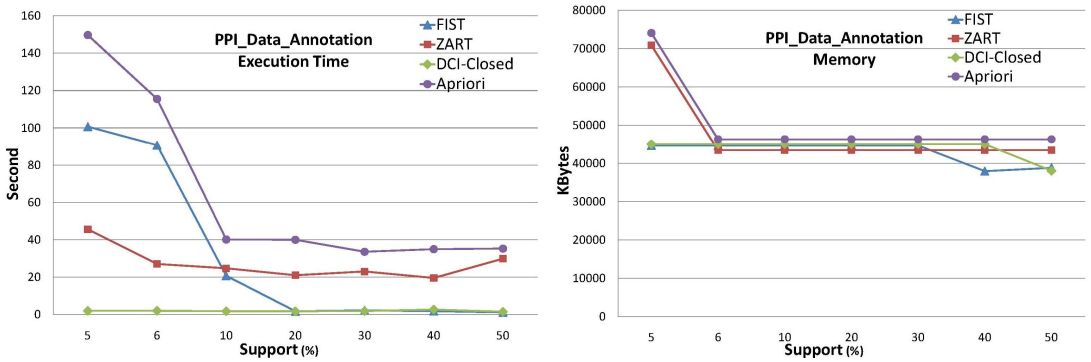
Figure 6.6: Experimental Results for the HIV-1–Human PPI Dataset (series 2)

For this series of experiments, we can see that the FIST implementations have the greatest execution times, comprised between 0.9 and 1.3 seconds approximately for *minsupport* equals to 0.001. This is due to the time required to process textual data as identifiers of objects and items, to generate user-friendly results, contrarily to other implementations that generate patterns containing integer value representations of the objects and items. For other algorithms,

we can see that Zart and Apriori, that are frequent itemsets based approaches, have the maximal execution times with approximately 0.8 and 0.9 seconds respectively for this *minsupport* value. Frequent closed itemsets based approaches have execution times comprised between 0.16 and 0.5 seconds approximately for this *minsupport* value.

6.2.3.2 Integrated Protein-Protein Interaction Data and Annotations

This dataset was used to compare the applicability and efficiency of the first version of the FIST approach with those of the Apriori, Zart and DCI-Closed algorithms on the portable personal computer. For this first series of experiments, the *minconfidence* threshold was set to 0.001 (0.1%) and the *minsupport* threshold was varied between 0.05 (5%) and 0.5 (50%). Execution times (A) and memory usage (B) for this series of experiments are presented in Figure 6.7.



A. Execution times. B. Memory usage.

Figure 6.7: Experimental Results for the Integrated HIV-1–Human PPI Dataset (series 1)

Execution time results show that Apriori gives the highest results, with a maximal execution time of approximately two minutes and a half, and that DCI-Closed is the most efficient with execution times of a few seconds for all *minsupport* values. Execution times of FIST 1.0 are equivalent to those of DCI-Closed, and lower than those of Zart, for *minsupport* values between 0.2 (20%) and 0.5 (50%). They become higher than those of Zart for *minsupport* values lower than 0.1 (10%). Memory usage results show that approaches in the two groups, i.e., Apriori and Zart for frequent itemsets based approaches and DCI-Closed and FIST for frequent closed itemsets based approaches, give identical results. The difference between the two groups become visible when the *minsupport* value is less than 0.06 (6%) with a maximal memory usage of approximately 75 MB for frequent itemsets based approaches and 45 MB for frequent closed itemsets based approaches.

As for other datasets, a second series of experiments was run on this dataset to compare efficiency of the three versions of FIST with those of Apriori, Charm, DCI-Closed, Eclat, JClose and Zart on the PowerEdge server. For this first series of experiments, the *minconfidence* threshold was set to 0.001 (0.1%) and the *minsupport* threshold was varied between 0.01 (1%) and 0.5 (50%). Execution times are given in Figure 6.8 with *minsupport* values on the horizontal axis on a non-linear scale.

For this series, several experiments were stopped after twenty four hours of execution. These

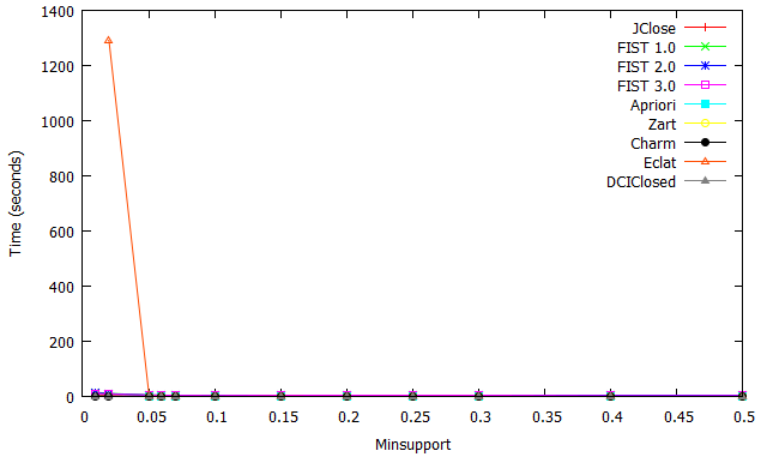


Figure 6.8: Experimental Results for the Integrated HIV-1–Human PPI Dataset (series 2)

are the executions of Apriori for *minsupport* = 0.01 and 0.02, Zart for *minsupport* = 0.01 and 0.02, and Eclat for *minsupport* = 0.01. All other algorithms were able to process this dataset for all *minsupport* values. The maximal execution time obtained was for Zart, with approximately twenty one minutes for *minsupport* = 0.02. All other algorithms have execution times less than fifteen seconds, that is the execution time of FIST 1.0 for the lowest *minsupport* value. The most efficient algorithms are DCI-Closed and Charm with execution times less than one seconds for all *minsupport* values. We can also see that the successive versions of FIST have decreasing execution times, the maximal execution time for FIST 2.0 and FIST 3.0 being of thirteen and ten seconds respectively for *minsupport* = 0.01.

6.2.4 Discussion on Results

The first series of experiments were run to validate the FIST approach and to evaluate the relevance of the conceptual patterns extracted from the end-user's viewpoint. They were run on a portable personal computer with limited capabilities to evaluate the applicability, in term of resource usages, and the efficiency, in term of acceptable execution times, of the approach in this context. Experimental results showed that the FIST approach is relevant, considering the fact that much more patterns are generated than by classical association rule mining approaches, and has limited resource usages that make possible its application to datasets with large numbers of items, even if in some cases execution times can be important. Detailed experimental results for the different phases (not given in this report) showed that important improvements can be achieved considering the method for generating patterns from the generalized suffix-tree data structure.

The second series of experiments were run to compare the efficiency of the three versions of FIST and six other state of the art algorithms for mining association rules. Differences between the results of the first and the second versions of FIST show that the improvements in both memory usage and execution times, resulting from the successive optimizations developed during this work, are important. The first version of FIST was implemented using a table

and index based representation to extract patterns from the generalized suffix-tree constructed while reading the dataset. This method poses efficiency problems in the case of a very large number of items as shown in experimental results for the integrated Eisen yeast dataset. The second algorithmic version of FIST uses the generalized suffix-tree data structure to perform all required operations to generate knowledge patterns. It was implemented with the standard Java Collections API in FIST 2.0 and with the Trove for Java Collections API in FIST 3.0. Comparing results for FIST 2.0 and FIST 3.0 shows important improvements in execution times due to the difference between Trove and standard Java collections API. Overall, experimental results show that the second algorithmic version of FIST has resource usages comparable to those of frequent closed itemsets based approaches for mining association rules, even if somewhat higher due to the generation of larger resulting sets.

From a theoretical framework viewpoint, experimental results confirm that frequent itemsets based approaches are the less efficient, and pose applicability problems, compared to the frequent closed itemsets based approaches. These lasts were all applicable for all the experiments conducted, except the Eclat algorithm for the integrated HIV-1–Human PPI dataset and *minsupport* = 0.01 (1%). Among the frequent closed itemsets based approaches, the DCI-Closed and Charm approaches are the most efficient and have globally the least resource requirements. As shown in the literature, to the best of our knowledge, DCI-Closed is actually the most efficient frequent closed itemsets based algorithm for mining association rules with resources usage slightly lower than those of Charm. Results for FIST 3.0 confirm that this approach can be applied to problems in the "omics" field even for datasets with a large number of items and very low *minsupport* values.

6.3 Extracted Knowledge Pattern Evaluations

This section presents studies on the extracted knowledge patterns and the interpretation of these patterns from a biological viewpoint. The prediction and analysis of interactions between proteins has been the subject of several studies in the literature [Shoemaker 2007b]. Most of these studies concern intra-species interactions, i.e., the analysis of interactions between proteins of a single organism like yeast or human. The approaches used for these studies are based on Bayesian networks [Jansen 2003], decision trees [Zhang 2004], random forest classifiers [Lin 2004, Qi 2005], kernel methods [Ben-Hur 2005b, Martin 2005, Yamanishi 2004], mixture-of-feature-expert classifiers [Qi 2007] and genetic algorithms [Singhal 2007]. The accuracy of predictions and analyses is improved when several sources of information are integrated and integrative approaches can also provide means to justify the confidence of inferred interactions [Dhanapalan 2007, Shoemaker 2007b]. Support Vector Machines [Qiu 2008] and Bayesian networks [Lee 2006] based approaches integrating protein interaction data with biological annotations have been proposed. These studies showed that association rule mining, classifications and bi-clustering can provide new important knowledge and relevant patterns to support protein interaction analyses. However, a limitation in these studies is that the extracted patterns contain a limited number of different types of information. The study of conceptual patterns extracted by FIST on the protein interaction datasets presented below show that integrating domain knowledge with human and HIV-1 protein interaction data, and generating data mining patterns containing all possible combination of these types of information can improve the relevance of the results.

6.3.1 Number of Knowledge Patterns Generated

We study in this section experimental results on the number of patterns (generators, rules and bi-clusters) generated by FIST and other approaches for different datasets and while varying the *minsupport* and *minconfidence* thresholds.

6.3.1.1 Comparison of Apriori-like and FIST Approaches Association Rules

We first compare the number of classical association rules, such as generated by the Apriori approach, and the number of association rules in the conceptual bases generated by the FIST approach. These experiments were conducted on the PowerEdge server computer as memory requirements to generate classical association rules are important for low *minsupport* values, for which the number of rules can be important. In the following three graphics, showing the number of association rules generated by Apriori and FIST, the blue curves correspond to Apriori results and the red curves correspond to FIST results.

Figure 6.9 shows results for the HIV-1–Human binary interactions database for *minsupport* = 0.001 (0.1%) and *minsupport* = 0.002 (0.1%), and *minconfidence* varied between 0.001 (0.1%) and 0.5 (50%). Complete and detailed results of this series of experiments are given in tables B.2 and B.3 of Annex B.2.

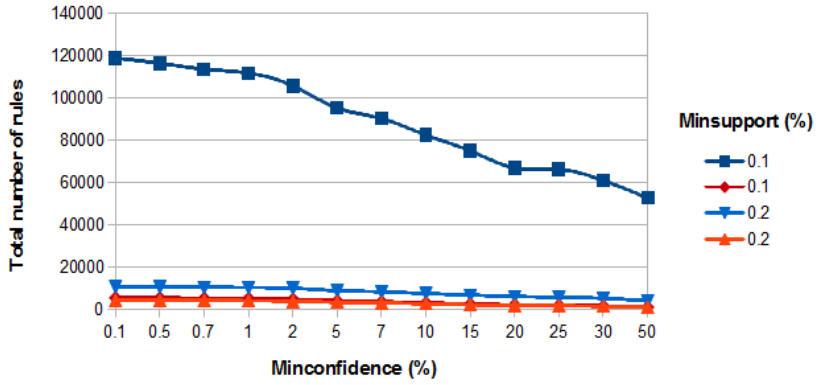


Figure 6.9: Comparing Number of Association Rules Extracted from HIV-1–Human Binary Interactions Database

We can see that the number of rules in the bases generated by FIST is reduced by a factor up to several tens compared with rules generated using the classical Apriori-like approach. As redundant rules are removed from the results, we obtain a compact set of rules, easy to manage and explore, allowing the end-user to concentrate on the most relevant rules.

Results for the HIV-1–Human interaction types database are depicted in figure 6.10 for *minsupport* = 0.002 (0.2%) and *minsupport* = 0.003 (0.3%), and *minconfidence* varied between 0.001 (0.1%) and 0.5 (50%). Up to 18 GB of primary memory were required to generate classical association rules for *minsupport* = 0.002 (0.2%). It was not possible to generate classical association rules for *minsupport* = 0.001 (0.1%) as the primary memory space required is greater than the 23 GB available. Complete and detailed results of this series of experiments are given in tables B.4 and B.5 of Annex B.2.

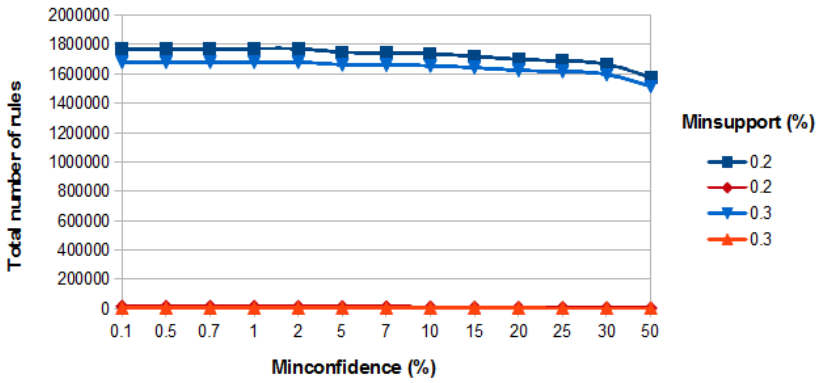


Figure 6.10: Comparing Number of Association Rules Extracted from HIV-1–Human Interaction Types Database

The number of extracted classical association rules is for both *minsupport* values and for all *minconfidence* values greater than one million. This poses the problem of the usability of the result as it is difficult to manage and explore such a huge amount of rules, even with dedicated tools. For the same *minsupport* and *minconfidence* values, the number of rules in the conceptual bases varies between 1734 and 15104. The reduction factor is thus for these experiments from the order of one hundred to one thousand, depending on the threshold values.

In figure 6.11, results for the HIV-1–Human integrated database for $minsupport = 0.035$ (3.5%) and $minsupport = 0.04$ (4%), and *minconfidence* varied between 0.001 (0.1%) and 0.5 (50%) are presented. Complete and detailed results of this series of experiments are given in tables B.6 and B.7 of Annex B.2.

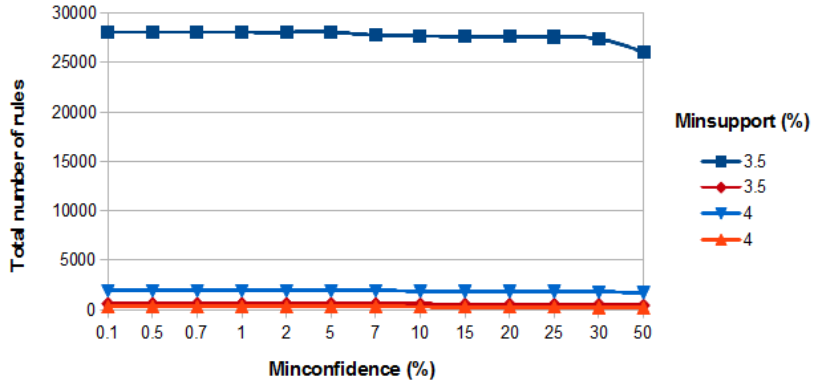


Figure 6.11: Comparing Number of Association Rules Extracted from HIV-1–Human Integrated Database

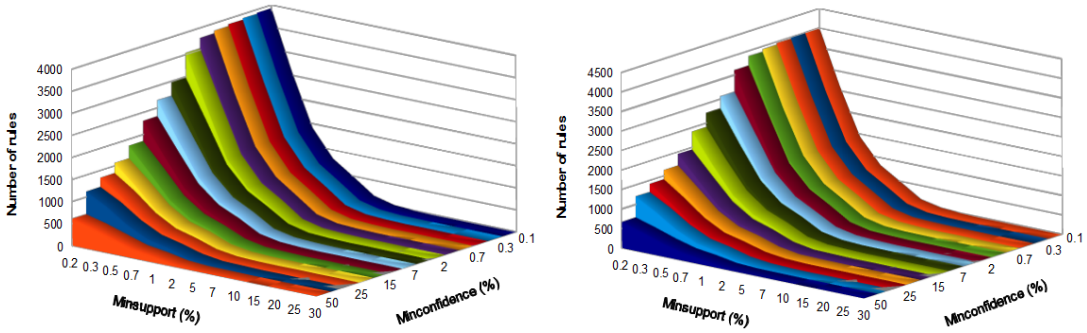
It was not possible for this database to generate Apriori association rules for *minsupport* values less than 0.035 (3.5%). This is due to the memory required during the process that

exceeds the 23 GB available for *minsupport* values of 0.03 (3%). Due to this limitation, we were not able to extract classical association rules depicting relationships supported by less than 50 proteins.

These experiments show that, even if considered as trivial from an algorithmic viewpoint, the second phase of ARM, that consists in generating association rules from frequent or frequent closed itemsets, can require important resources. This can be a major limitation in terms of applicability and usability for pattern mining implementations in an application context.

6.3.1.2 Comparison of Bases of Association Rules Extracted by FIST

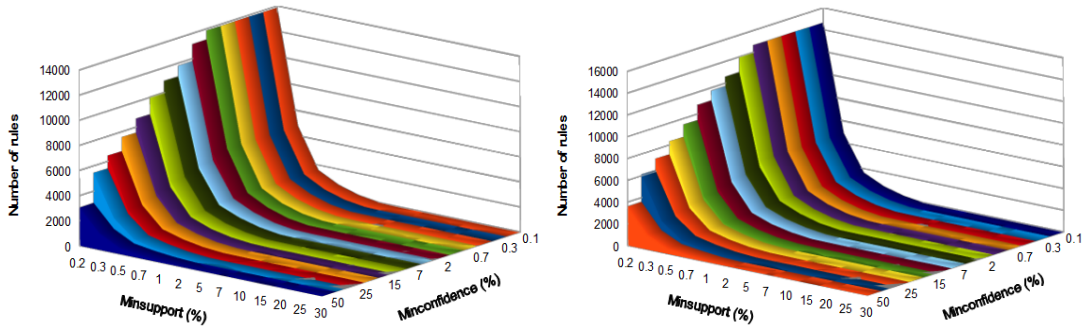
The number of proper approximate association rules and min-max approximate association rules extracted by FIST for the HIV-1–Human PPI binary interactions, interaction types and, integrated interactions and annotations databases are shown in figure 6.12, figure 6.13 and figure 6.14 respectively. In these figures, the horizontal x-axis and z-axis show variations in the *minsupport* and *minconfidence* threshold values, in percentages, respectively. The vertical y-axis shows the number of association rules generated. For these experiments, the *minsupport* threshold was varied between 0.002 (0.2%) and 0.3 (30%) and the *minconfidence* threshold was varied between 0.001 (0.1%) and 0.5 (50%). The resulting patterns obtained during this experiments are the most general possible for these datasets, as for *minsupport* = 2%, extracted patterns show relationships between items that are supported by as few as two proteins.



A. Proper base. B. Structural base.

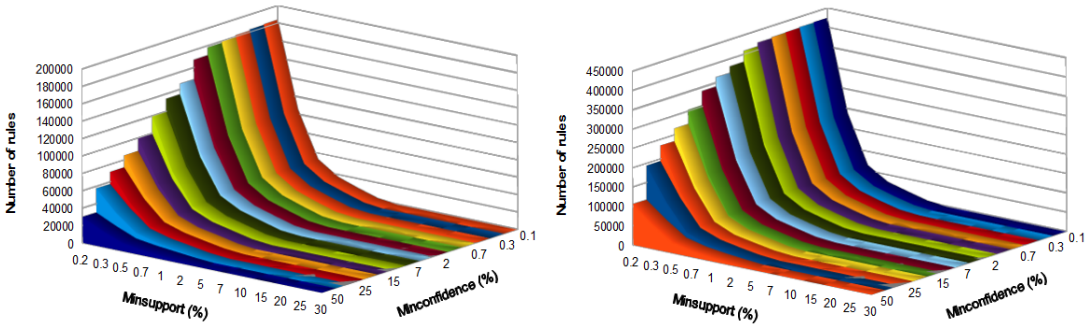
Figure 6.12: Size of Approximate Rule Bases for HIV-1–Human Binary Interactions Database

We can see differences in the sizes of the proper base of approximate rules and the structural base of min-max (minimal antecedent and maximal consequent) approximate rules. This is related to the fact that the proper base contains one rule for each pair of frequent closed itemsets related by inclusion whereas there can be several min-max approximate rules for the same pair, one rule for each generator of the frequent closed itemsets included in the other. For the two first databases, the sizes of the proper base and the base of min-max approximate rules are nearly identical. These results show that approximately 3.2% of the frequent closed itemsets have more than one generator for the HIV-1–Human PPI binary interactions database, and approximately 8% for the HIV-1–Human PPI interaction types database. Larger differences can be seen for the HIV-1–Human PPI integrated database, for which this proportion is approximately of 18.9%.



A. Proper base. B. Structural base.

Figure 6.13: Size of Approximate Rule Bases for HIV-1–Human Interaction Types Database



A. Proper base. B. Structural base.

Figure 6.14: Size of Approximate Rule Bases for HIV-1–Human Integrated Database

The number of frequent generators extracted by FIST for the three HIV-1–Human PPI databases is shown in figure 6.15. It should be noted that the values for the number of frequent generators, corresponding to the vertical y-axis of the graphic, are on a logarithmic scale.

The frequent generators represent the minimal sets of features, i.e., HIV-1 interacting proteins, HIV-1 protein interaction types or protein annotations, that are common to a set of human proteins. For each frequent generator extracted by FIST, an exact association rule is generated between the generator (antecedent of the rule) and the frequent closed itemset that is its closure (consequent of the rule) if they are not identical.

The number of min-max exact association rules extracted by FIST for the three HIV-1–Human PPI databases are shown in figure 6.16. The number of min-max exact association rules, depicted by the vertical y-axis of the graphic, are on a logarithmic scale.

For each *minsupport* threshold value, the number of min-max exact rules shows the number of generators that are different from their closure. Hence, only generators that are different from their closure produce a min-max exact rule, with the generator in antecedent and the closed itemset that is its closure in the consequent. Experimental results showed that this number is in most cases equal to zero for market basket data for instance [Brin 1997b, Pasquier 1999b].

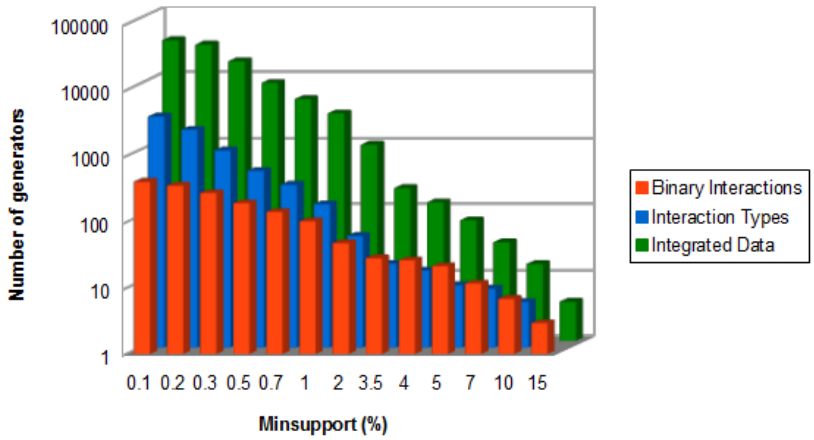


Figure 6.15: Number of Generators for HIV-1–Human PPI Databases

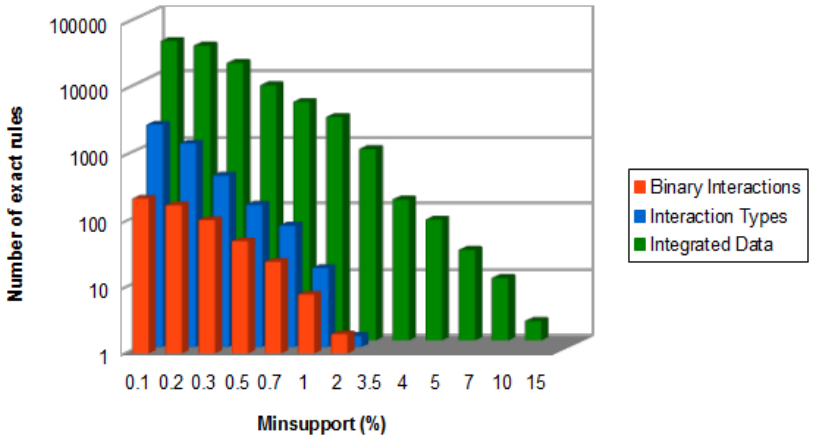


Figure 6.16: Number of Exact Rules for HIV-1–Human PPI Databases

This is related to the fact that in these data, the number of strong relationships between items is low, due to the weakly correlated and sparse nature of these data. We can see here that for the HIV-1–Human PPI data this number is high as the number and proportion of correlated items, i.e., features common to several proteins, is important.

The difference between the number of generators, given in table 6.15, and the number of min-max exact rules shows the number of generators that have an identical closure. Differences between these numbers means that for some frequent closed itemsets (consequent of rules), more than one rule is generated, each with a different generator in the antecedent of the rule. For the HIV-1–Human PPI binary interactions database, the frequent closed itemsets have on average 2.81 generators, for the HIV-1–Human PPI interaction types database, 1.72 generators, and for the HIV-1–Human PPI integrated database, 1.08 generators.

6.3.1.3 Number of Bi-clusters Extracted by the FIST Approach

The number of bi-clusters generated by FIST for the three HIV-1–Human PPI databases is shown in table 6.2. The only parameter used for determining extracted bi-clusters during these experiments is the *minsupport* threshold that was varied between 0.001 (0.1%) and 0.3 (30%).

Table 6.2: Number of Bi-clusters Extracted from HIV-1–Human PPI Databases

HIV–Human	**Minsupport (%)**											
PPI Database	0.1%	0.3%	0.5%	0.7%	1%	2%	5%	7%	10%	15%	20%	30%
Interaction data	340	256	186	141	104	49	22	12	7	3	2	2
Integrated data	9872	5361	2837	1748	1097	386	61	29	15	4	2	2
Interaction types	1736	840	417	269	146	50	9	8	5	0	0	0

We can see that the maximal numbers of bi-clusters were obtained for the integrated interaction and annotation dataset (row 2), the minimal results for the binary interaction dataset (row 1), and results for the interaction type dataset (row 3) are intermediate. These results are proportional to the size, i.e., the number of items, of the three databases. This series of experiments confirms that the number of bi-clusters grows exponentially when the value of the *minsupport* threshold is lessened. However, we can see that for the three datasets the number of bi-clusters can be efficiently managed by exploration, selection and visualization tools, even for *minsupport* values as low as 0.1% which is the least possible value for these datasets. This enables to identify for each human protein all other proteins that have at least one common property (HIV-1 interacting protein, type of interaction or, biological or bibliographical annotation) and to identify all these common properties.

The number of bi-clusters generated from the two Eisen *et al.* databases are shown in table 6.3. During these experiments, the *minsupport* threshold was varied between 0.005 (0.5%) and 0.3 (30%).

Table 6.3: Number of Bi-clusters Extracted from Eisen Yeast Gene Expression Databases

Eisen	**Minsupport (%)**											
Database	0.3%	0.5%	0.7%	1%	2%	5%	7%	10%	15%	20%	25%	30%
Expression data	7284	2393	1002	477	147	2	0	0	0	0	0	0
Integrated data	80384	29396	14116	6780	1508	198	98	43	20	10	4	2

We can see that for the second dataset, integrating expression data and gene annotations, the number of bi-clusters increases in an important manner when the *minsupport* threshold is lessened, with several thousands of bi-clusters for threshold values less than 1%. The use of other parameters than the *minsupport* threshold can reduce this number of bi-clusters by deleting from the results the less relevant bi-clusters, from an interpretability viewpoint, regarding the size of their intent and extent.

6.3.2 Biological Interpretation of Extracted Knowledge Patterns

Using bi-clustering and association rule mining can help to identify significant sets of host proteins that undergo the same types of interactions. The functional cohesiveness of the protein sets

described by the extracted knowledge patterns can be validated using human protein-protein interaction networks, gene ontology annotations and sequence similarities [MacPherson 2010].

Results obtained with the FIST approach were compared to those obtained by Tastan *et al.* [Tastan 2009] which are the most comprehensive HIV-Human PPI results available to date. These experimental results on the different PPI databases are presented in [Mondal 2012a]. We focused on the results generated for $minsupport = 0.1\%$ and $minconfidence = 0.1\%$, which are the least threshold values tested and thus, which resulting patterns contain maximal information. For each protein pair interaction predicted in [Tastan 2009], the number of bi-clusters and the numbers of antecedent and of consequent of association rules generated covering it were determined. Among the 3372 interactions predicted in [Tastan 2009] using random forest classifiers, 895 are covered by at least one bi-cluster generated by FIST. This is 26.5% of their predicted pairs. The HIV-1 proteins associated in bi-clusters, and the frequency of the associations in the database, generated by FIST during these experiments are given in Appendix B.3.

Now, the random forest classifier is reported to achieve a mean average precision (MAP) of 0.23 on this problem, meaning that around 23% of the predicted interacting pairs should be expected to be true positives. This is just a little below the percentage of predicted pairs that are "confirmed" by FIST. Since the random forest classifier has little in common with FIST, we believe the two techniques should be regarded as complementary to one another. By the same argument, there are good chances that the interacting pairs predicted by [Tastan 2009] and confirmed by FIST are indeed true interactions.

In general, it appears that proteins pairs predicted by the random forest classifier with a high score are mostly confirmed by a large number of bi-clusters, although exceptions exist, like the novel high-score predicted pair $\langle \text{ENV\_GP120}, \text{CALM1} \rangle$, which is not covered by any bi-cluster, indicating perhaps that it is a false positive. Likewise, most low-score predictions are not confirmed by FIST with some exceptions, like $\langle \text{ENV\_GP120}, \text{EP300} \rangle$ which, however, were known to be indirectly interacting (the human gene is reported in the siRNA screen in [Konig 2008]). All in all, exceedingly few (28) of the 2100 novel predictions by [Tastan 2009], or 1.3%, are confirmed by FIST. An exhaustive list thereof is given in Table 6.4, along with the number of covering bi-clusters, approximate, and exact rules. For rules, two separate counts are provided for rules that have the viral protein in the antecedent (LHS part) and in the consequent (RHS part).

On the other hand, FIST finds 451 protein pairs that are covered by at least one bi-cluster among those not included in [Tastan 2009], i.e., for which no explicit indication of possible interaction was pointed out. This is 2.2% of the pairs not included in [Tastan 2009].

The most covered of these protein pairs is $\langle \text{NEF}, \text{IFNG} \rangle$, covered by 70 bi-clusters. The NEF protein occurs in the antecedent of 755 approximate rules, in the consequent of 779 approximate rules, in the antecedent of 28 exact rules and in the consequent of 30 exact rules. Lagging far behind this pair, we find the four pairs $\langle \text{TAT}, \text{ACTG1} \rangle$, covered by 45 bi-clusters. $\langle \text{NEF}, \text{IL6} \rangle$, covered by 45 bi-clusters, $\langle \text{TAT}, \text{IL2} \rangle$, covered by 44 bi-clusters, and $\langle \text{TAT}, \text{IL6} \rangle$, covered by 44 bi-clusters. There are a number of other pairs covered by 35 or fewer bi-clusters.

The $\langle \text{NEF}, \text{IFNG} \rangle$ pair, to begin by the most covered novel suggestion, although not previously signaled, looks like a promising candidate for further investigation: NEF is the viral negative regulatory factor, associated with the early stages of HIV infection, and the IFNG gene encodes for the interferon-γ protein, an important immune response stimulator and modulator. Interferon proteins are produced and released by host cells in response to the presence of pathogens (viruses, bacteria, parasites or tumor cells). They allow the cell to warn neigh-

HIV-1	Human	# bi-clusters	# approximate rules		# exact rules	
			LHS	RHS	LHS	RHS
ENV_GP160	APOBEC3G	1	0	0	0	0
REV	CXCR4	4	5	5	0	0
ENV_GP120	FURIN	1	0	0	0	0
VPR	MAPK3	34	186	197	7	0
ENV_GP120	PAK1	1	0	0	0	0
TAT	PAK2	2	1	1	0	0
NEF	PIK3R2	8	19	19	0	0
TAT	PPARG	1	0	0	0	0
NEF	PRKCD	34	275	300	17	5
NEF	PRKCG	34	275	300	17	5
NEF	PRKCZ	18	77	83	4	0
TAT	RAF1	3	2	2	0	0
VPR	RAF1	3	2	2	0	0
ENV_GP120	RAN	2	1	1	0	0
TAT	RPA2	4	5	5	0	0
TAT	SDCBP	2	1	1	0	0
GAG_PR55	SHC1	1	0	0	0	0
ENV_GP120	SLC3A2	1	0	0	0	0
TAT	SREBF2	2	1	1	0	0
NEF	STAT5A	4	5	5	0	0
NEF	SUMO1	1	0	0	0	0
TAT	TCEB1	1	0	0	0	0
ENV_GP120	TUBB1	1	0	0	0	0
ENV_GP120	UBB	2	1	1	0	0
NEF	UBB	2	1	1	0	0
TAT	UBE2I	1	0	0	0	0
TAT	WT1	1	0	0	0	0
REV	XRCC5	3	2	2	0	0

Table 6.4: New Predicted Interacting Pairs Confirmed by FIST

boring cells to trigger the immune system protective defenses that eradicate pathogens. The suggestion of some kind of relationships between NEF and IFNG proteins may be corroborated by recent research on HIV vaccines [Gahery 2007]. Examining bi-clusters extracted from the interaction types database, we found 194 bi-clusters involving both the NEF and IFNG proteins; these bi-clusters contain both the "NEF downregulates" and "NEF upregulates" interaction types. Conceptual pattern extracted from the integrated database reinforce this view as the selection of bi-clusters containing the NEF and IFNG proteins also contain annotations on the functions associated to the proteins. For instance, the GO:0060333, REACT:25078 and REACT:25229 annotations are involved in the bi-cluster associating the {GO:0060333, REACT:25078, REACT:25229, GO:0019221, NEF} itemset and the {FCGR1A, HLA-A, HLA-B, HLA-C, HLA-DRA, HLA-DRB1, HLA-E, HLA-F, HLA-G, ICAM1, IFNG, IRF1, IRF2, IRF3, STAT1, SOCS1} human proteins. The GO:0060333 annotation is reported as "interferon-

gamma-mediated signaling pathway" in the AmiGO advance search engine for biological annotations[6]. The REACT:25078 and REACT:25229 are reported to belong to "interferon gamma signaling" and "interferon signaling" pathways in the Reactome knowledge base. These pathways are related to interferon-γ that are cytokines secreted by activated immune cells, and also B-cells and APC, that play a central role in initiating immune responses, especially antiviral and anti-tumor effects [Gough 2008]. The NEF HIV-1 protein might then be responsible of regulating IFNG, thus inhibiting one of the natural immune defenses of the host cell. A possible publication corroborating this hypothesis, that needs further investigation, is [Gahery 2007].

The same negative regulatory factor is involved in the $\langle \text{NEF}, \text{IL6} \rangle$ pair: IL6 is the gene encoding for interleukin-6, a pro-inflammatory cytokine secreted by T-cells and macrophages to stimulate immune response. Indeed, the interaction between NEF and interleukin-6 has been recognized quite early in the study of AIDS [Chirmule 1994]. This relationship is supported by 25 bi-clusters extracted from the interaction types database that involve the "NEF downregulates" interaction type and IL6 human protein.

Other two novel pairs suggested by FIST, namely $\langle \text{TAT}, \text{IL2} \rangle$ and $\langle \text{TAT}, \text{IL6} \rangle$, involve interleukins. The TAT and IL2 pair is supported by 187 bi-clusters extracted from the interaction types database that involve the "TAT activated by", "TAT downregulates" and "TAT upregulates" interaction type. The TAT and IL6 pair is supported by 32 bi-clusters involving the "TAT activated by" and "TAT upregulates" interaction type. IL2 is the gene of interleukin-2, a signaling molecule normally produced during an immune response: an antigen binding to a T-cell receptor stimulates the secretion of interleukin-2, which in turn stimulates the growth of antigen-selected cytotoxic T-cells. TAT, for trans-activator of transcription, is a key protein of HIV-1, the first to be transcribed, causing the subsequent massive increase in the transcription levels of the HIV dsRNA. Both interactions are mentioned in the literature: The interaction between TAT and interleukin-6 in [Scala 1994] and the one between TAT and interleukin-2 in [Westendorp 1994].

As for pair $\langle \text{TAT}, \text{ACTG1} \rangle$ suggested by FIST, we are not aware of any work in the literature mentioning it. However, the suggestion does not look completely implausible, for TAT functions also as a cell-penetrating peptide that acts as a toxine, causing the apoptosis of uninfected T-cells, and the γ-actin 1, encoded for by gene ACTG1, is a component of the cytoskeleton of T-cells. The TAT and ACTG1 pair is present in 4 bi-clusters extracted from the interaction types database that involve the "TAT induces rearrangement of" and "TAT downregulates" interaction type.

[6] http://amigo.geneontology.org/

Chapter 7

Conclusion and Perspectives

Contents

In this chapter, we present the conclusion of the report, summarizing the work conducted during the thesis and presenting a critical analysis of this work, in section 7.1. Perspectives of future research works and open research problems on related topics are presented in section 7.2.

7.1 Conclusion

We present in the two first subsections conclusions drawn from the work from: First, a theoretical and technical viewpoint; Second, an application and experimental viewpoint. These conclusions are based on the algorithmic and experimental results obtained, including both performance evaluation tasks and biological knowledge discovery from protein interaction databases. Limitations of the actual FIST algorithm are also mentioned in the third subsection, as well as possible solutions as open problems.

7.1.1 Extraction of Frequent Conceptual Patterns

We propose a new integrated approach for finding bases of conceptual association rules and bi-clusters together. This approach, named FIST, is based on the frequent closed itemsets theoretical framework. It extracts frequent closed itemsets, with the generators and the supporting object list of each. These are the minimal sets of information required to construct the conceptual frequent patterns conjointly, without extra database access. For simplicity, reproducibility and extensibility, the FIST approach uses a suffix-tree based data structure instead of the traditional FP-Tree or Prefix-Tree used in most approaches, which is new in frequent closed pattern mining to the best of our knowledge. This data structure is called frequent generalized suffix-tree. It does not require complex procedures, like maintaining a transverse chained list of items or suffix links, and allows the parallel processing of the tree branches in multi-threaded

environments. This data structure used in combination with the ordering of items according to their support optimizes the extraction of the frequent closed itemsets and limits memory usage of the data processing. Contrarily to most existing implementations of frequent pattern extractions, FIST can generate a user-friendly result in which information on elements (variables, objects, items, etc.) in extracted patterns are semantic descriptors (names, labels, etc.) that can be directly used by the end-user without requirement for other post-processing operations. As shown in the experiments, the generation of user-friendly patterns can be a major limiting factor for the applicability of methods because of the resources required for this post-processing phase of patterns.

Another important feature of FIST is that it extends the classical association rules by associating with each the lists of objects supporting the rule. This extra information provides a more detailed knowledge on the relationship depicted by the rule and can be used to relate the rules with bi-clusters concerning the same elements. The bases of rules generated by FIST are small sets of rules, called minimal non-redundant cover of association rules, from which all classical association rules, such as generated by Apriori, can be deduced if required. Two bases of approximate association rules can be generated by FIST. The first is the proper base, containing rules between two frequent closed itemsets in antecedent and consequent, that are the maximal antecedent and consequent (max-max) rules. The second is the structural base, containing rules between a generator in antecedent and its closure in consequent, that are the minimal antecedent and maximal consequent (min-max) rules. From these rules, conceptual classification rules, containing a class value in the consequent and associated with the list of objects supporting the rule, can also be generated.

The bi-clusters generated by FIST are hierarchical conceptual bi-clusters. They are represented as concepts, i.e., intension and extension, and they form a dual lattice structure defined by the inclusion relation. In these bi-clusters, the intension represents a maximal set of items (data values) that are common to some objects (rows) and the extension represents the maximal set of objects that have the items in common. The hierarchical lattice structure can provide the user with a bi-clustering result at different level of precision. The highest bi-clusters in the lattice show the largest sets of items shared by some objects and the lowest bi-clusters show the largest sets of objects that have some common items.

Two algorithmic versions of the FIST approach were developed and implemented in Java language. Java was chosen for its portability and accessibility on all computers. The second algorithmic version was implemented with the Standard Java Collections API and the Trove for Java Collections. Experiments were conducted on four bioinformatics datasets to evaluate the efficiency and applicability of the approach and compare them with state of the art algorithms. Experimental results showed that FIST clearly outperforms frequent itemsets based approaches, like Apriori and Zart, in both execution times and memory usage. They also showed that FIST gives results that are comparable to those of frequent closed itemsets based approaches in execution times and memory usage. From these results, we also deduced that Trove for Java Collections improves the efficiency compared with the Standard Java Collections API.

7.1.2 Application to Interaction Proteomics

Bioinformatics presents important challenges for knowledge extraction and data mining, and association rule mining, bi-clustering and classification have been used for several genomics and proteomics problems. Interaction proteomics is a recent new bioinformatics problem that aims

at discovering and analyzing interactions between proteins to understand underlying biological processes. This understanding can help in the design of medical protocols and cures for diseases such as the AIDS syndrome that is one of the major world-wide health problems actually. For an application purpose of the FIST approach, three databases were constructed from HIV-1 and human protein protein interaction data and knowledge bases. The first database contains protein-protein interaction binary data, the second database integrates biological and bibliographic annotations of proteins with these interaction data, and the third contains protein-protein interaction types data.

The FIST approach was validated by its application for the analysis of HIV-1 and human protein interactions. The results obtained for the binary interaction database confirmed and improved the predictions made by existing methods, and suggested new possible interactions to be further investigated. Additional background knowledge integrated in the second database are Gene Ontology biological annotations, phenotype annotations and PubMed and Reactome publication annotations. Extracting conceptual patterns with FIST from this database, integrating protein interaction data with corresponding protein annotations, demonstrated its capability to extract meaningful relationships between protein interactions and annotations. The patterns extracted from this database can help the user to understand the nature of relationships between interacting proteins by using background biological knowledge represented as annotations. Furthermore, it showed the potential of integrating several heterogeneous sources of biological information. The analysis of the third database, containing interactions described as directed relationships with type, enabled to further precise the relationships between interacting proteins. Compared with results for the first database, groups of interacting proteins are partitioned into subgroups of proteins with identical influential relationships that are described by interaction types. The next step in this study is to integrate interaction types with protein annotations to precise the nature of relationships between interacting proteins.

These experiments showed that FIST can efficiently extracts bases of conceptual association rules and bi-clusters even for a very large number of items such as the integrated database that contains several thousands of variables (data matrix columns). These performance results were confirmed by experiments conducted on two other datasets on yeast gene expression data from Eisen *et al.* [Eisen 1998]. The first contains gene expression levels after normalized discretization of numerical expression values. The second integrates biological annotations on genes, from the popular GO, KEGG and PubMed knowledge bases, with these expression data. This last data matrix is a challenge for pattern extraction, as it contains more than 2 400 rows and more than 9 900 columns. FIST was shown to be able to efficiently process all these datasets.

7.1.3 Limitations of the FIST Approach

According to the conclusions drawn from the experimental results, we point out here some limitations of the approach.

Even if the FIST application was conceived to process a large panel of data representations (unary, binary, multi-valued, transactional, matrix, duplicated variables, etc.), considering all application areas and all data formats is nearly infeasible. Processing multimedia or text data for instance requires the application of methods from other domains, such as signal or language processing. As for all itemset based pattern mining methods, the FIST approach cannot be directly applied to databases containing numerical continuous values, if close values need to be

considered as similar. Such variables need to be discretized, by regrouping sufficiently closed values considered as similar, to convert numerical continuous values to discrete values. This discretization phase can also improve the relevance of extracted patterns as the regrouping of numerical continuous values can take into account the semantic of the application. This is for instance the case when numerical gene expression values are discretized into "under-expressed", "unchanged" and "over-expressed" values depicting different significant expression levels of genes. Fuzziness based approaches can also be integrated in the FIST approach to define the items. These approaches allow to consider several grouping of numerical data values at the same time and, thus, improve the discretization results in application domains where the criteria for grouping values are not formally defined or are previously unknown [Hong 2008].

Implementations of the second algorithmic version of the FIST approach have shown to be much more efficient than frequent itemsets based approaches and to have performances close to those of state-of-art frequent closed itemsets based approaches. However, the amount of information generated by FIST, i.e., bi-clusters and conceptual association rules, can induce large memory requirements for the generation of the user-friendly resulting patterns when very large datasets, such as image data descriptors for example, are processed. Memory usage efficiency can be improved by applying algorithmic procedures for regrouping similar subtrees in the frequent generalized suffix-tree and for on the fly generation of searched patterns from the generalized frequent suffix-tree. This on the fly generation can be combined with the use of user-defined templates determining which elements (variables, items or objects) are searched for and in which part of the pattern.

7.2 Perspectives

This work can be extended in several ways and the perspective of future research works lie in two directions. The first is the extension of the work done from technical and theoretical viewpoints. The second concerns experiments and analysis of resulting patterns from an application viewpoint. These two perspectives are presented in the next two subsections.

7.2.1 Scope of Technical Enhancement

Different extensions and improvements of the FIST approach and implementations can be achieved. These can improve the efficiency and applicability of FIST, and the usefulness of extracted sets, and adapt the approach to other domains of application with specific requirements.

A first improvement is the one cited at the end of section 8.1.3 for reducing memory usage and improving efficiency of the generalized frequent suffix-tree processing and on the fly generation of patterns. This on the fly generation can be combined with the integration of other objective and subjective measures of interestingness to select and filter mined patterns according to the end-user's interests. This can improve the relevance and usefulness of extracted patterns. From an executional context viewpoint, implementing a multi-threaded version can improve in an important manner performances as branches of the generalized suffix-tree can be processed separately and in parallel. This simple extension can be a major enhancement of the implementation, particularly for processing very large datasets such as image data descriptors.

The FIST approach parameters allow to generate conceptual classification association rules that are rules with values of a class variable in the consequence. However, the application

context of this work did not allow us to test intensively this feature and, further studies and experiments must be conducted to evaluate these patterns and compare the obtained classifier to state of the art classification methods. A theoretical analysis and the development of an associative conceptual classifier based on the generalized suffix-tree approach can be a promising path to extend the possibilities of the FIST approach. Interesting results on association rule based classification were obtained in the literature, particularly with non-redundant bases of association rules that can limit the impact of over-fitting, i.e., obtaining rules too specific to a particular dataset, the training set [Ras 2010].

Another scope of enhancement is to extend FIST for the generation of conceptual sequential patterns from time series data. The suffix-tree data structure was shown to be able to efficiently process such data with for example the CCC approach (Contiguous Column Coherent clusters) proposed for extracting bi-clusters in [Madeira 2009]. The hierarchical structure of conceptual bi-clusters extracted by FIST can improve the relevance of such results as different levels of abstraction of bi-clusters can be studied.

These are some hints of enhancement and other possible further optimizations can be devised as FIST was implemented in a straightforward way to be easily reproducible, adapted and extended for other programming languages or contexts of application, with different constraints for instance.

7.2.2 Scope of Application Enhancement

The full exploitation of the numerous conceptual patterns extracted from the HIV-1 and human PPI databases requires further investigations. The analysis of patterns containing interaction data and protein annotations, and of patterns containing interaction types, is a long task that can provide detailed knowledge on protein interactions. Indeed, the integration of different kinds of biological information is an essential consideration to fully understand the underlying biological processes [Bell 2011]. In near future, we also plan to integrate additional information about proteins, like structural and sequence similarities, with protein-protein interactions and annotations to improve the results.

In the future, we also plan to apply the actual version of FIST to other domains of application and different kinds of datasets. An interesting perspective of application is the experimentation on image data descriptors, such as SIFT (Scale-Invariant Feature Transform), in order to identify relevant criteria required to classify images. The conceptual patterns extracted by FIST were shown to be well adapted to this task in preliminary theoretical studies. Identifying frequent conceptual patterns in image descriptors such as the standard SIFT can improve the image classification process. But this is a difficult case from a performance viewpoint because of the data scale, as an image descriptor data file usually contains several tens of thousands of descriptors (rows) with for each several hundred of values (columns). This application is an ongoing work in the image domain. Another ongoing application of the FIST approach concerns the biodiversity domain, that aims at analyzing biological diversity. Preliminary results on a new botanical database constructed by integrating phenotypic, biological and medical features of plants showed that conceptual patterns extracted by FIST can help the process of identification of families, genius and species. Further investigations of these results are actually conducted.

All these results and promising perspectives of research works confirm that lattice and formal concept analysis based bi-clustering, itemset search and association or classification rule extrac-

tion are suitable paradigms for knowledge pattern extraction that may be used for real-sized applications integrating domain knowledge for improving the data mining process [Lieber 2008].

Chapter 8

Conclusion et Perspectives

Contents

Dans ce chapitre, nous présentons la conclusion du rapport, résumant les travaux réalisés au cours de la thèse et présentant une analyse critique du travail mené, et des perspectives de travaux ultérieurs ainsi que les problèmes de recherche ouverts concernant les sujets connexes.

8.1 Conclusion

Nous présentons tout d'abord les conclusions tirés des travaux mené, premièrement d'un point de vue théorique et technique, et deuxième d'un point de vue expérimental et applicatif. Ces conclusions sont basées sur les résultats algorithmiques et expérimentaux obtenus, incluant les évaluations des performances et des modèles de connaissances biologiques extraits à partir des bases de données d'interactions protéomiques. Les limitations de l'approche FIST sont également mentionnées, ainsi que leurs solutions possibles et problèmes restant ouverts.

8.1.1 Extraction de Motifs Conceptuels Fréquents

Nous proposons une nouvelle approche intégrée pour l'extraction simultanée de *bases de règles d'association conceptuelles* et *bi-clusters conceptuels*. Cette approche, nommée FIST, se base sur la théorie des motifs fermés fréquents pour extraire les itemsets fermés fréquents, avec leurs générateurs les listes d'objets support de chacun. Ces ensembles constituent l'information minimale nécessaire pour construire ces motifs fréquents conceptuels conjointement, sans accès supplémentaire à l'ensemble de données. Pour des raisons de simplicité, reproductibilité et extensibilité, l'approche FIST utilise une structure de données basée sur les arbres suffixés au lieu des FP-Tree ou arbre préfixé utilisés par la plupart des approches existantes. L'utilisation d'une telle structure de données, appelée arbre suffixé généralisé, est nouveau dans le domaine de l'extraction de motifs fréquents. Elle ne nécessite pas de procédures complexes, telle que le

maintien d'une liste chaînée transversale d'items ou de liens de préfixes, et permet le traitement en parallèle des branches de l'arbre dans des environnements multi-tâches. Cette structure de données utilisée en combinaison avec l'ordonnancement des items en fonction de leur support permet d'optimiser l'extraction des motifs fermés fréquents en limitant l'utilisation mémoire durant le traitement des données. Contrairement à la plupart des implémentations existantes d'extraction de motifs fréquents, FIST peut générer un résultat complet et interprétable dans lequel les informations sur les éléments (variables, objets, items, etc.) dans les motifs extraits sont des descripteurs sémantiques (noms, libellés, etc.) qui peuvent être directement utilisé par l'utilisateur final, sans nécessiter d'autres opérations de post-traitement. Comme le montrent les expériences, la génération de modèles complets et interprétables peut être un facteur limitant important pour l'applicabilité de ces méthodes en raison de la nécessité de post-traitements des motifs extraits.

Une autre caractéristique importante de l'approche FIST est qu'elle étend les règles d'association classiques en leur associant les listes d'objets supportant la règle. Cette information supplémentaire apporte une connaissance plus détaillée de la relation représentée par la règle et peut être utilisée pour relier les règles et bi-clusters portant sur les mêmes éléments. Les bases de règles générées par FIST sont des ensembles condensés de règles, appelées couvertures minimales non-redondantes de règles d'association, à partir desquelles toutes les règles d'association classiques, telles que générées par Apriori, peuvent être déduites si nécessaire. Deux bases de règles approximatives association peuvent être générées par FIST. La première est la base propre, contenant les règles valides entre deux itemsets fermés fréquents, en antécédent et en conséquence, qui sont les règles d'antécédent et conséquence maximaux (règles "max-max"). La seconde est la base structurelle, contenant les règles valides entre un générateur, en antécédent, et sa fermeture, en conséquence, qui sont les règles d'antécédents minimaux et conséquences maximales (règles "min-max"). A partir de ces règles, les règles de classification conceptuelle, contenant une valeur de classe dans la conséquence et associées chacune à la liste des objets support, peuvent également être générés.

Les bi-clusters conceptuels hiérarchiques générés par FIST sont représentés comme des concepts, définis par une intention et une extension, et forment une hiérarchie duale en treillis définie par la relation d'inclusion. Dans ces bi-clusters, l'intention représente un ensemble maximal d'items (valeurs des données) qui sont communs à certains objets (lignes des données) et l'extension représente l'ensemble maximal d'objets qui ont des éléments en commun. La structure hiérarchique de treillis peut fournir à l'utilisateur un résultat de bi-clustering à différents niveaux de précision. Les bi-clusters les plus hauts dans le treillis décrivent les plus grands ensembles de valeurs partagés par certains objets et les bi-clusters les plus bas affichent les plus grands ensembles d'objets qui ont des valeurs communes.

Deux versions algorithmiques de l'approche FIST ont été développées et implémentées en langage Java choisi pour sa portabilité et sa disponibilité sur tous systèmes. La deuxième version algorithmique a été implémentée avec les collections standards de l'API Java et les collections de l'API Trove. Les expérimentations ont été menées sur quatre ensembles de données bioinformatiques pour évaluer l'efficacité et l'applicabilité de l'approche et de les comparer avec l'état des algorithmes d'art. Les résultats montrent que l'approche FIST est plus performante que les approches basées sur l'extraction des itemsets fréquents, telles que Apriori et Zart, tant en termes de temps d'exécution que d'utilisation mémoire. Ils montent également que ses performances sont comparables à celles des approches basées l'extraction d'itemsets fermés fréquents en temps d'exécution et utilisation mémoire, et que l'utilisation des collections de

l'API Trove améliorent l'efficacité par rapport aux collections standards de l'API Java.

8.1.2 Application aux Interactions Protéomiques

La bio-informatique présente des défis importants pour l'extraction de connaissances à partir des données et la fouille de données, et l'extraction de règles d'association, le bi-clustering et la classification supervisée ont été largement appliqués en génomique et protéomique. L'analyse d'interactions protéomiques est un problème récent en bio-informatique qui vise à découvrir et analyser les interactions entre protéines afin de comprendre les processus biologiques sous-jacents. Cette compréhension peut contribuer à la conception de protocoles médicaux et remèdes pour le traitement de maladies telles que, par exemple, le virus du SIDA qui est l'un des principaux problèmes mondial de santé actuel. Afin de valider expérimentalement l'approche FIST, trois bases de données ont été construites à partir de diverses données d'interactions protéomiques et bases de connaissances sur les protéines du virus VIH-1 et de l'organisme humain. La première base de données contient des données binaires d'interactions protéomiques, la seconde intègre annotations biologiques et bibliographiques des protéines avec les données binaires d'interactions, et la troisième contient des données sur les types d'interactions protéomiques.

Les deux versions algorithmiques de l'approche FIST ont été appliquées à l'analyse des interactions entre protéines du VIH-1 et de l'organisme humain. Les résultats obtenus pour la base de données d'interactions binaires a permis de confirmer et étendre les prédictions faites lors des précédentes analyses, et de suggérer de nouvelles interactions potentielles. Les motifs conceptuels extrait pour la seconde base de données, intégrant données d'interactions protéomiques et annotations des protéines correspondantes, ont démontré la capacité de FIST à extraire des relations significatives entre les interactions et les annotations. Les annotations intégrées dans cette base de données sont issues des annotations biologiques de Gene Ontology et des annotations phénotypiques et bibliographiques des bases PubMed et Reactome. Ces motifs extraits peuvent contribuer à la compréhension de la nature des relations entre les protéines qui interagissent en utilisant les connaissances biologiques initiales représentées par les annotations. En outre, ces expérimentations ont démontré le potentiel important de l'intégration de plusieurs sources d'informations biologiques hétérogènes. L'analyse de la troisième base de données, contenant des interactions décrites par des relations typées et orientées, a permis de préciser les relations entre les protéines interagissant. Comparativement aux résultats obtenus pour la première base de données, les groupes de protéines interagissant sont partitionnés en sous-groupes de protéines d'influences identiques décrites par les types d'interactions. La prochaine étape de cette étude est l'intégration des types d'interactions et des annotations des protéines afin de préciser la nature des relations et les propriétés communes aux protéines des sous-groupes.

Ces expérimentations ont démontré la capacité de l'approche FIST à extraire les bases de règles d'association conceptuelles et bi-clusters conceptuels à partir d'ensembles de données contenant un grand nombre d'items et de variables. Elles ont été confirmées par les expérimentations menées sur deux autres ensembles de données génomiques construits à partir des données de Eisen *et al.* [Eisen 1998]. Le premier de ces ensembles contient les niveaux d'expression des gènes de *Saccharomyces cerevisiae* obtenus par la discrétisation normalisée des valeurs numériques d'expression. Le second intègre avec ces données d'expression les annotations biologiques et bibliographiques des gènes concernés issues des bases de référence GO, KEGG et PubMed. Même si ce dernier ensemble de données constitue un cas difficile pour

l'extraction de motifs fréquents, du fait de la grande taille de l'espace de recherche induite par les plus de 2 400 objets et plus de 9 900 variables contenus, l'approche FIST a été capable de traiter efficacement tous ces ensembles de données.

8.1.3 Limitations de l'Approche FIST

Selon les conclusions tirées des expérimentations, nous pointons ici certaines limitation de l'approche.

Même si l'approche FIST a été conï¿½ue pour traiter un large panel de formats de représentation des données (unaires, binaires, multi-valuées, transactionnelles, matrices, variables dupliqués, etc.), il est impossible de prendre en considération l'ensemble des domaines applicatifs et formats de données potentiels. Le traitement de données multimédia ou données textuelles par exemple nécessite l'application de méthodes d'autres domaines, telles que le traitement du signal ou du langage naturel. Comme toutes les méthodes d'extraction de connaissances basées sur les itemsets, l'approche FIST ne peut être appliquée directement aux données constituées de valeurs numériques continues si les valeurs proches doivent être considérés comme similaires. Dans ce contexte, les variables doivent être discrétisées, en regroupant les valeurs suffisamment proches pour être considérées comme similaire, afin de convertir les valeurs numériques continues en valeurs discrètes. Cette phase de discrétisation peut également améliorer la pertinence des motifs extraits par la prise en compte de la sémantique de l'application dans le regroupement des valeurs numériques continues. C'est par exemple le cas lorsque les valeurs numériques d'expression génomiques sont discrétisées en valeurs "sous-exprimé", "inchangé" et "sur-exprimé" qui représentent les différents niveaux d'expression de gènes significatifs. Les méthodes basées sur les ensembles flous peuvent également être intégrés dans l'approche FIST afin de définir les items considérés. Ces approches permettent de considérer plusieurs groupements de valeurs des données numériques simultanément et, par conséquent, d'améliorer les résultats de la discrétisation dans des domaines d'application oïï¿½ les critères de regroupement de valeurs ne sont pas formellement définies ou sont inconnues [Hong 2008].

Les implémentations de la deuxième version algorithmique de l'approche FIST se sont avérées être beaucoup plus efficaces que les approches basées sur les itemsets fréquents et avoir des performances proches de celles des approches basées sur les itemsets fermés fréquents d'état de l'art. Toutefois, la quantité d'informations générées par FIST, soit les bi-clusters conceptuel et les règles d'association conceptuelles, peut induire d'importants besoins en mémoire pour la génération de motifs complets et interprétables lorsque de très grands ensembles de données, tels que les descripteurs d'images par exemple, sont traités. L'efficacité d'utilisation de la mémoire peut être améliorée par l'application de procédures algorithmiques afin de regrouper les sous-arbres semblables dans l'arbre suffixé généralisé et pour la génération à la volée des motifs recherchés à partir de l'arbre suffixé généralisé. Cette génération à la volée peut être combinée avec l'utilisation de templates, ou patrons, définis par l'utilisateur pour déterminer quels éléments (variables, items ou objets) sont recherchés dans quelle partie du motif.

8.2 Perspectives

Ce travail peut être étendu de plusieurs manières et les perspectives de travaux de recherche ultérieurs se situent dans deux directions. La première est le prolongement du travail effectué, des points de vue techniques et théoriques. La seconde concerne les expérimentations

et l'analyse des motifs extraits du point de vue de l'application. Ces deux perspectives sont présentées dans les deux sections suivantes.

8.2.1 Perspectives d'Extensions Techniques

Différentes extensions et améliorations de l'approche FIST et des implémentations peuvent être réalisées. Celles-ci peuvent améliorer l'efficacité et l'applicabilité de l'approche, et l'utilité des ensembles extraits, et adapter l'approche à d'autres domaines d'application ayant des besoins spécifiques.

La première amélioration, évoquée dans la section 8.1.3, a pour objet la réduction de la consommation mémoire et l'amélioration de l'efficacité du traitement de l'arbre suffixé généralisé ainsi que la génération à la volée des motifs. Cette phase de génération à la volée peut être combinée avec l'intégration d'autres mesures d'intérêt objectives et subjectives afin de sélectionner et filtrer les motifs extraits en fonction des intérêts de l'utilisateur final afin d'en améliorer la pertinence et l'utilité. D'un point de vue du contexte d'exécution, l'implémentation d'une version multi-tâches peut améliorer de manière importante les performances, les branches de l'arbre suffixé généralisé pouvant être traitées séparément en parallèle. Cette extension simple peut être une amélioration majeure de l'implémentation, en particulier pour le traitement de très grands ensembles de données tels que les descripteurs d'images.

Les paramètres de l'approche FIST permettent de générer des règles d'association conceptuelles contenant les valeurs d'une variable de classe en conséquence pour la classification. Cependant, le contexte d'application de ce travail ne nous a pas permis de tester intensivement cette fonction et d'autres études et expérimentations doivent être menées afin d'évaluer ces règles et comparer le classifieur obtenu aux méthodes de classification de l'état de l'art. Une analyse théorique et l'élaboration d'un classifieur conceptuel associatif basé sur l'approche par arbre suffixé généralisé peut constituer une voie prometteuse pour étendre les possibilités de l'approche FIST. Des résultats intéressants sur la classification basée sur des règles d'association ont été obtenus et présentés dans la littérature du domaine, en particulier concernant les bases non-redondantes de règles d'association qui peuvent permettre de limiter l'impact du sur-apprentissage, c'est à dire l'obtention de règles trop spécifiques à un ensemble de données particulier [Ras 2010].

Un autre champ d'amélioration possible concerne l'extension de FIST pour la génération de motifs séquentiels conceptuels à partir de données de séries chronologiques. Les structures de données à base d'arbres de suffixes ont montré leur capacité à traiter efficacement ces données, avec par exemple l'approche CCC (Contiguous Column Coherent clusters) proposée pour l'extraction de bi-clusters de valeurs contigües dans [Madeira 2009]. La structure hiérarchique des bi-clusters conceptuels extraits par FIST peut améliorer la pertinence de ces résultats en permettant l'étude des bi-clusters à différents niveaux d'abstraction.

De nombreuses autres optimisations et extensions de l'approche FIST sont possibles du fait de la simplicité de sa mise en oeuvre et sa reproductibilité aisée. Elle peut facilement être adaptée et étendue à d'autres langages de programmation ou contextes d'application, avec des contraintes différentes, par exemple.

8.2.2 Perspectives d'Extensions Applicatives

La pleine exploitation des nombreux motifs conceptuels extraits des bases de données d'interactions protéomiques du VIH-1 et de l'organisme humain nécessite des investigations complémentaires. L'analyse des motifs contenant des données sur les interactions et les annotations de protéines et des motifs contenant les types d'interaction, est une tâche longue qui peut apporter des connaissances détaillées sur les interactions entre protéines. En effet, l'intégration des différents types d'informations biologiques est un élément essentiel pour une bonne compréhension des processus biologiques sous-jacents [Bell 2011]. Dans un futur proche, nous prévoyons également d'intégrer des informations supplémentaires concernant les protéines, telles que les similarités structurelles et de séquences, avec les interactions et annotations protéomiques afin d'améliorer les résultats.

A l'avenir, nous prévoyons également d'appliquer l'approche FIST à d'autres domaines d'application et différents types de jeux de données. Une perspective intéressante d'application est l'expérimentation sur les descripteurs de données d'images, tels que les SIFT (Scale-Invariant Feature Transform), afin d'identifier les critères pertinents requis pour classifier ces images. Les modèles conceptuels extraites par FIST se sont avérés particulièrement adaptés à cette tâche dans les études théoriques préliminaires. L'identification des motifs conceptuels fréquents dans les descripteurs d'images tels que les SIFT peut améliorer le processus de classification des images. Il s'agit toutefois d'un cas applicatif complexe du point de vue des performances en raison des volumes de données, les matrices de données de description d'une image contenant généralement plusieurs dizaines de milliers de descripteurs (lignes) avec pour chacun plusieurs centaines de valeurs (colonnes). En sus de cette application en cours dans le domaine de l'imagerie, une autre application en cours concerne le domaine de la biodiversité et vise à l'analyse de la diversité biologique. Les résultats préliminaires obtenu avec une nouvelle base de données botaniques construite en intégrant des caractéristiques phénotypiques, biologiques et médicales de plantes ont montré que les motifs conceptuels extraits par FIST peuvent aider le processus d'identification des familles, groupes et espèces de plantes.

Ces résultats et ces perspectives prometteuses de travaux de recherche confirment que le bi-clustering, la recherche d'itemsets fréquents et l'extraction de règles d'association et de classification basés sur les théories des treillis et de l'analyse de concepts formels sont des paradigmes adaptés à l'extraction de connaissances qui peuvent être utilisés pour des applications de taille importante intégrant des connaissances initiales du domaine afin d'améliorer le processus de fouille [Lieber 2008].

Part IV

Appendix and Bibliography

Appendix A

Theoretical Considerations

A.1 Discussion on Itemset Closure Property

Let $I = \{i_1, \ldots, i_n\}$ be a frequent itemset $I \subseteq L$ and $I \in \mathcal{F}$. I is a frequent closed itemset if one of the two following conditions is verified:

1. I corresponds exactly to at least one object in database $\mathcal{D}$: $\exists o \in O$ such as $\forall i \in I$ we have $i \in o$ and $\nexists j \in o$ such that $j \notin I$. The probability for this event is denoted $P_{\mathcal{D}}(I = o)$.
2. I corresponds to the intersection of at least two objects of the database $\mathcal{D}$: $I = \bigcap_{k=1}^{k=N} o_k$ with $N \geq 2$. That means the intersection of objects o_k contains all and only the items contained in I: $\nexists i \in L$ such as $i \in \bigcap_{k=1}^{k=N} o_k$. The probability for this event is denoted $P_{\mathcal{D}}(I \sqcap S)$ where $S = \bigcup_{k=1}^{k=N} o_k$.

Consider a discrete uniform distribution of occurrence probabilities of independent items in L in objects of O over database $\mathcal{D}$. We then have a probability distribution whereby the finite number of objects are equally likely to contain an item i: All items $i \in L$ have an identical probability, denoted $P_{\mathcal{D}}(i \in o)$ or $P_{\mathcal{D}}(i)$ for short, equals to support(i) to be contained in an object o for all $o \in O$. Conversely, all items $i \in L$ have an identical probability, denoted $P_{\mathcal{D}}(i \notin o)$ or $P_{\mathcal{D}}(\overline{i})$ for short, equals to 1 - support(i) not to be contained in an object o for all $o \in O$. The probability for itemset $I = \{i_1, \ldots, i_n\}$ to be contained in an object o is then:

$$P_{\mathcal{D}}(I \subseteq o) = \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha) = \prod_{\alpha=1}^{\alpha=n} support(i_\alpha)$$

and the probability for itemset $I = \{i_1, \ldots, i_n\}$ with $L \setminus I = \{i_p, \ldots, i_q\}$ to be equal to an object o is:

$$P_{\mathcal{D}}(I = o) = \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha) \cdot \prod_{\beta=p}^{\beta=q} P_{\mathcal{D}}(\overline{i_\beta}) = \prod_{\alpha=1}^{\alpha=n} support(i_\alpha) \cdot \prod_{\beta=p}^{\beta=q} (1 - support(i_\beta))$$

Now, consider two frequent itemsets $I_1 = \{i_1^1, \ldots, i_n^1\}$ and $I_2 = \{i_1^2, \ldots, i_n^2\}$ such that $support(I_1) \geq support(I_2) \iff \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha^1) \geq \prod_{\beta=1}^{\beta=n} P_{\mathcal{D}}(i_\beta^2)$. Let denote $I_3 = L \setminus I_1 = \{i_1^3, \ldots, i_p^3\}$ the set of items not contained in I_1 and, $I_4 = L \setminus I_1 = \{i_1^4, \ldots, i_p^4\}$ the set of items not contained in I_2 We have:

$$\prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha^1) \geq \prod_{\beta=1}^{\beta=n} P_{\mathcal{D}}(i_\beta^2) \Longrightarrow \prod_{\alpha=1}^{\alpha=n} P_{\mathcal{D}}(i_\alpha^1) \cdot \prod_{\mu=1}^{\mu=p} (1 - P_{\mathcal{D}}(i_\mu^3)) \geq \prod_{\beta=1}^{\beta=n} P_{\mathcal{D}}(i_\beta^2) \cdot \prod_{\nu=n}^{\nu=p} (1 - P_{\mathcal{D}}(i_\nu^4))$$
$$\Longrightarrow P_{\mathcal{D}}(I_1 = o) \geq P_{\mathcal{D}}(I_2 = o)$$

The probability that I_1 is equal to an object in $\mathcal{D}$ is greater than or equal to the probability that I_2 is equal to an object in $\mathcal{D}$. We also have $P_{\mathcal{D}}(I_1 \subseteq o) \geq P_{\mathcal{D}}(I_2 \subseteq o)$ and by extension to all objects in $S \subseteq O$, $S = \bigcup_{k=1}^{k=N} o_k$ we have:

$$\prod_{k=1}^{k=N} P_{\mathcal{D}}(I_1 \subseteq o_k) \geq \prod_{k=1}^{k=N} P_{\mathcal{D}}(I_2 \subseteq o_k)$$

Let denote $P_{\mathcal{D}}(i_\beta \in S)$ the probability that item i_β is contained in all object $o_k \in S$, that is $\forall o_k \in S$ we have $i_\beta \in o_k$. Let denote $P_{\mathcal{D}}(I_1 \sqsubseteq S)$ the probability that I_1 is contained in all object $o_k \in S$. We deduce:

$$P_{\mathcal{D}}(I_1 \sqsubseteq S) - \prod_{\mu=1}^{\mu=p} P_{\mathcal{D}}(i_\mu^3 \in S) \geq P_{\mathcal{D}}(I_2 \sqsubseteq S) - \prod_{\nu=1}^{\nu=p} P_{\mathcal{D}}(i_\nu^4 \in S) \Longrightarrow P_{\mathcal{D}}(I_1 \sqcap S) \geq P_{\mathcal{D}}(I_2 \sqcap S)$$

The probability that I_1 is equal to the intersection of objects in S is greater than or equal to the probability that I_2 is equal to the intersection of objects in S.

Appendix B

Experimental Results

B.1 HIV-1 and Human Protein Interaction Types

Table B.1: List of HIV-1 and Human Protein Protein Interaction Types.

Interaction Type	**Interaction Type**	**Interaction Type**
CAPSID binds	INTEGRASE binds	RT enhanced by
CAPSID co-localizes with	INTEGRASE co-localizes with	RT fractionates with
CAPSID complexes with	INTEGRASE degraded by	RT imported by
CAPSID degraded by	INTEGRASE imported by	RT inhibited by
CAPSID downregulates	INTEGRASE incorporates	RT interacts with
CAPSID induces release of	INTEGRASE inhibited by	RT phosphorylated by
CAPSID inhibited by	INTEGRASE interacts with	RT stimulated by
CAPSID inhibits	INTEGRASE recruits	TAT acetylated by
CAPSID interacts with	INTEGRASE regulated by	TAT acetylates
CAPSID isomerized by	INTEGRASE requires	TAT activated by
CAPSID modulates	INTEGRASE stimulated by	TAT activates
CAPSID phosphorylated by	INTEGRASE stimulates	TAT associates with
CAPSID stabilizes	MATRIX activates	TAT binds
CAPSID ubiquitinated by	MATRIX associates with	TAT cleaved by
CAPSID upregulates	MATRIX binds	TAT competes with
ENV_GP120 activated by	MATRIX co-localizes with	TAT complexes with
ENV_GP120 activates	MATRIX complexes with	TAT cooperates with
ENV_GP120 associates with	MATRIX downregulates	TAT decreases phosphorylation of
ENV_GP120 binds	MATRIX enhances	TAT degraded by
ENV_GP120 cleavage induced by	MATRIX exported by	TAT degrades
ENV_GP120 cleaved by	MATRIX fractionates with	TAT disrupts
ENV_GP120 cleaves	MATRIX imported by	TAT downregulates
ENV_GP120 competes with	MATRIX incorporates	TAT enhanced by
ENV_GP120 complexes with	MATRIX inhibits	TAT enhances
ENV_GP120 cooperates with	MATRIX interacts with	TAT enhances polymerization of
ENV_GP120 decreases phosphorylation of	MATRIX myristoylated by	TAT imported by
ENV_GP120 deglycosylates	MATRIX phosphorylated by	TAT inactivates
ENV_GP120 degrades	MATRIX regulated by	TAT induces cleavage of
ENV_GP120 downregulated by	MATRIX stimulated by	TAT induces complex with
ENV_GP120 downregulates	MATRIX ubiquitinated by	TAT induces phosphorylation of
ENV_GP120 enhanced by	MATRIX upregulates	TAT induces rearrangement of
ENV_GP120 enhances	NEF Co-localizes with	TAT induces release of
ENV_GP120 glycosylated by	NEF Induces release of	TAT inhibited by
ENV_GP120 inactivates	NEF Inhibits	TAT inhibits
ENV_GP120 incorporates	NEF Interacts with	TAT inhibits acetylation of
ENV_GP120 induces accumulation of	NEF Modulates	TAT interacts with
ENV_GP120 induces acetylation of	NEF Requires	TAT methylated by
ENV_GP120 induces cleavage of	NEF Upregulates	TAT modified by

Continued on next page ...

Interaction Type	Interaction Type	Interaction Type
ENV_GP120 induces phosphorylation of	NEF activates	TAT modulated by
ENV_GP120 induces release of	NEF associates with	TAT modulates
ENV_GP120 inhibited by	NEF binds	TAT phosphorylated by
ENV_GP120 inhibits	NEF co-localizes with	TAT phosphorylates
ENV_GP120 interacts with	NEF degrades	TAT recruited by
ENV_GP120 mediated by	NEF downregulates	TAT recruits
ENV_GP120 modified by	NEF inactivates	TAT regulated by
ENV_GP120 modulated by	NEF induces cleavage of	TAT regulates
ENV_GP120 modulates	NEF induces complex with	TAT requires
ENV_GP120 processed by	NEF induces phosphorylation of	TAT stabilizes
ENV_GP120 regulated by	NEF induces rearrangement of	TAT stimulated by
ENV_GP120 relocalizes	NEF inhibits	TAT stimulates
ENV_GP120 requires	NEF interacts with	TAT synergizes with
ENV_GP120 sensitizes	NEF modulates	TAT ubiquitinated by
ENV_GP120 stimulates	NEF myristoylated by	TAT upregulated by
ENV_GP120 upregulates	NEF phosphorylated by	TAT upregulates
ENV_GP160 activates	NEF polarizes	VIF associates with
ENV_GP160 associates with	NEF regulated by	VIF binds
ENV_GP160 binds	NEF relocalizes	VIF co-localizes with
ENV_GP160 cleaved by	NEF requires	VIF complexes with
ENV_GP160 co-localizes with	NEF stabilizes	VIF degrades
ENV_GP160 complexes with	NEF synergizes with	VIF inhibited by
ENV_GP160 decreases phosphorylation of	NEF upregulates	VIF inhibits
ENV_GP160 downregulated by	NUCLEOCAPSID activates	VIF interacts with
ENV_GP160 downregulates	NUCLEOCAPSID binds	VIF modulates
ENV_GP160 inactivates	NUCLEOCAPSID enhances	VIF phosphorylated by
ENV_GP160 induces phosphorylation of	NUCLEOCAPSID imported by	VIF relocalizes
ENV_GP160 induces release of	NUCLEOCAPSID incorporates	VPR activates
ENV_GP160 inhibited by	NUCLEOCAPSID interacts with	VPR binds
ENV_GP160 inhibits	NUCLEOCAPSID ubiquitinated by	VPR co-localizes with
ENV_GP160 interacts with	P1 ubiquitinated by	VPR competes with
ENV_GP160 methylated by	P6 binds	VPR complexes with
ENV_GP160 palmitoylated by	P6 incorporates	VPR cooperates with
ENV_GP160 processed by	P6 inhibited by	VPR decreases phosphorylation of
ENV_GP160 regulated by	P6 interacts with	VPR downregulates
ENV_GP160 relocalizes	P6 phosphorylated by	VPR enhanced by
ENV_GP160 upregulated by	P6 ubiquitinated by	VPR enhances
ENV_GP160 upregulates	POL binds	VPR enhances polymerization of
ENV_GP41 activates	RETROPEPSIN activated by	VPR exported by
ENV_GP41 binds	RETROPEPSIN activates	VPR imported by
ENV_GP41 complexes with	RETROPEPSIN cleaves	VPR inactivates
ENV_GP41 decreases phosphorylation of	RETROPEPSIN degrades	VPR incorporates
ENV_GP41 downregulated by	RETROPEPSIN downregulates	VPR induces accumulation of
ENV_GP41 downregulates	RETROPEPSIN induces cleavage of	VPR induces cleavage of
ENV_GP41 inactivates	RETROPEPSIN induces release of	VPR induces phosphorylation of
ENV_GP41 incorporates	RETROPEPSIN inhibited by	VPR induces release of
ENV_GP41 induces phosphorylation of	RETROPEPSIN phosphorylated by	VPR inhibited by
ENV_GP41 induces release of	REV activates	VPR inhibits
ENV_GP41 inhibited by	REV associates with	VPR interacts with
ENV_GP41 inhibits	REV binds	VPR isomerized by
ENV_GP41 interacts with	REV co-localizes with	VPR mediated by
ENV_GP41 modified by	REV competes with	VPR modulates
ENV_GP41 processed by	REV depolymerizes	VPR recruits
ENV_GP41 upregulates	REV enhanced by	VPR regulated by

Continued on next page ...

Interaction Type	**Interaction Type**	**Interaction Type**
GAG_PR55 associates with	REV exported by	VPR regulates
GAG_PR55 binds	REV inhibited by	VPR relocalizes
GAG_PR55 co-localizes with	REV inhibits	VPR requires
GAG_PR55 downregulated by	REV interacts with	VPR stimulates
GAG_PR55 incorporates	REV methylated by	VPR upregulates
GAG_PR55 inhibited by	REV modulated by	VPU activates
GAG_PR55 inhibits	REV phosphorylated by	VPU binds
GAG_PR55 interacts with	REV recruits	VPU degrades
GAG_PR55 modulated by	REV relocalized by	VPU downregulates
GAG_PR55 regulated by	REV requires	VPU inhibited by
GAG_PR55 relocalized by	REV stimulated by	VPU inhibits
GAG_PR55 relocalizes	REV stimulates	VPU interacts with
GAG_PR55 upregulated by	REV synergizes with	VPU phosphorylated by
GAG_PR55 upregulates	REV ubiquitinated by	VPU recruits
INTEGRASE acetylated by	RT binds	VPU regulates
INTEGRASE activated by	RT co-localizes with	VPU stabilizes
INTEGRASE associates with	RT degraded by	VPU upregulates

B.2 Number of Association Rules Extracted from HIV-1 and Human PPI Databases

Table B.2: Apriori Association Rules for HIV-1–Human PPI Binary Interactions Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.2	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	118762	117342	116306	113516	111619	105640	95162	90152	82343	74897	66826	66114	60878	52640
0.2	10508	10508	10411	10292	10171	9848	8681	8165	7347	6592	5963	5554	5038	3872
0.3	6562	6562	6562	6486	6365	6198	5411	4951	4394	3876	3447	3188	2851	1852
0.5	2440	2440	2440	2440	2420	2303	2103	1950	1757	1489	1316	1225	1130	749
0.7	1066	1066	1066	1066	1066	1022	941	875	790	656	562	511	461	309
1	546	546	546	546	546	537	493	473	428	371	323	286	263	162
2	158	158	158	158	158	158	153	143	136	123	111	101	92	51
5	36	36	36	36	36	36	36	36	34	25	23	22	22	12
7	12	12	12	12	12	12	12	12	12	10	8	7	7	5
10	2	2	2	2	2	2	2	2	2	2	2	2	2	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table B.3: FIST Association Rules for HIV-1–Human PPI Binary Interactions Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.3	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	5347	5295	5252	5161	5046	4666	3899	3454	2967	2347	2001	1718	1453	892
0.2	4106	4086	4056	4016	3954	3715	3161	2853	2491	2034	1745	1515	1309	794
0.3	2689	2689	2689	2659	2611	2523	2211	1985	1737	1453	1255	1125	989	602
0.5	1459	1459	1459	1459	1448	1380	1254	1164	1035	865	749	683	617	399
0.7	830	830	830	830	830	798	732	683	619	514	439	394	356	229
1	495	495	495	495	495	488	445	430	386	333	289	254	233	142
2	154	154	154	154	154	154	148	138	131	119	107	97	89	48
5	38	38	38	38	38	38	38	38	36	27	25	24	24	12
7	12	12	12	12	12	12	12	12	12	10	8	7	7	5
10	2	2	2	2	2	2	2	2	2	2	2	2	2	0
15	2	2	2	2	2	2	2	2	2	2	2	2	2	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table B.4: Apriori Association Rules for HIV-1–Human PPI Interaction Types Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.2	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
0.2	1771450	1771450	1771450	1771450	1771450	1769903	1747881	1743003	1735279	1719727	1698149	1688560	1663948	1575064
0.3	1680114	1680114	1680114	1680114	1680114	1680114	1662381	1659764	1654436	1642234	1623851	1616312	1595318	1515236
0.5	81012	81012	81012	81012	81012	81012	80845	78830	78428	77432	76899	76149	75485	71251
0.7	21234	21234	21234	21234	21234	21234	21234	20868	20497	20159	19821	19696	19438	18311
1	1086	1086	1086	1086	1086	1086	1086	1086	1032	957	933	904	861	743
2	194	194	194	194	194	194	194	194	194	194	192	172	168	152
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table B.5: FIST Association Rules for HIV-1–Human PPI Interaction Types Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.3	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	22454	22454	22454	22214	22117	21191	19188	17842	16393	14014	12512	10862	9208	6095
0.2	15104	15104	15104	15104	15104	14395	13041	12188	11289	9905	8796	7694	6684	4305
0.3	5195	5195	5195	5195	5195	5195	4610	4326	4038	3586	3260	2898	2581	1734
0.5	1762	1762	1762	1762	1762	1762	1682	1530	1433	1300	1189	1060	955	634
0.7	839	839	839	839	839	839	839	769	687	622	579	521	482	328
1	286	286	286	286	286	286	286	286	264	211	190	173	161	109
2	33.5	33.5	33.5	33.5	33.5	33.5	33.5	33.5	33.5	33.5	29.5	19.5	15.5	11.5
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table B.6: Apriori Association Rules for HIV-1–Human PPI Integrated Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.2	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
0.2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
0.3	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
0.5	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
0.7	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
1	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
3	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
3.5	28054	28054	28054	28054	28054	28054	28054	27751	27665	27623	27617	27608	27367	26066
4	1954	1954	1954	1954	1954	1954	1954	1954	1878	1846	1840	1832	1809	1726
5	208	208	208	208	208	208	208	208	202	176	170	169	169	139
7	38	38	38	38	38	38	38	38	38	32	28	27	27	25
10	8	8	8	8	8	8	8	8	8	8	7	7	7	5
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table B.7: FIST Association Rules for HIV-1–Human PPI Integrated Database.

Minsupport (%) \ Minconfidence (%)	0.1	0.3	0.5	0.7	1	2	5	7	10	15	20	25	30	50
0.1	276432	274666	272693	270133	266831	253227	220971	204181	184180	154825	136668	119164	101151	69130
0.2	213732	212838	211423	210077	207900	198808	176506	164516	150234	130082	114971	101292	87428	59918
0.3	101731	101731	101731	100921	99863	97807	89012	83701	77581	69128	62456	56363	50275	35197
0.5	40293	40293	40293	40293	40113	39154	36945	35274	33106	30126	27827	25586	23354	17128
0.7	21445	21445	21445	21445	21445	21035	20221	19527	18557	16995	15843	14744	13622	10236
1	11903	11903	11903	11903	11903	11827	11400	11176	10709	10026	9421	8817	8170	6294
2	3297	3297	3297	3297	3297	3297	3255	3153	3119	3061	2960	2895	2787	2393
3.5	623	623	623	623	623	623	623	610	586	561	555	550	538	454
4	322	322	322	322	322	322	322	322	303	278	272	268	261	217
5	117	117	117	117	117	117	117	117	113	93	87	86	86	64
7	34	34	34	34	34	34	34	34	34	29	25	24	24	22
10	7	7	7	7	7	7	7	7	7	7	6	6	6	4
15	1	1	1	1	1	1	1	1	1	1	1	1	1	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0

B.3 HIV-1 and Human Protein Interaction Bi-clusters

Table B.8: Bi-clusters Extracted from the HIV-1 and Human Protein Binary Interactions Database.

HIV-1 Proteins	**Support**
Minsupport=0.20%	
TAT	774
ENV_GP120	532
Minsupport=0.15%	
TAT	774
ENV_GP120	532
ENV_GP120, TAT	234
Minsupport=0.10%	
TAT	774
ENV_GP120	532
ENV_GP120, TAT	234
NEF	200
VPR	179
ENV_GP160	176
ENV_GP41	156
Minsupport=0.05%	
TAT	774
ENV_GP120	532
ENV_GP120, TAT	234
NEF	200
NEF, TAT	101
NEF, ENV_GP120	101
NEF, ENV_GP120, TAT	79
VPR	179
VPR, TAT	108
VPR, ENV_GP120	73
ENV_GP160	176
ENV_GP160, TAT	81
ENV_GP160, ENV_GP120	106
ENV_GP160, ENV_GP120, TAT	72
ENV_GP41	156
ENV_GP41, TAT	89
ENV_GP41, ENV_GP120	117
ENV_GP41, ENV_GP120, TAT	76
ENV_GP41, ENV_GP160	71
RETROPEPSIN	83
INTEGRASE	80
MATRIX	78
Minsupport=0.01%	
TAT	774
VPU	22
P6	14
ENV_GP120	532
NEF	200
VPR	179
ENV_GP160	176
ENV_GP41	156
RETROPEPSIN	83
CAPSID	33
NUCLEOCAPSID	25

Continued on next page ...

HIV-1 Proteins	**Support**
REV	65
RT	42
GAG_PR55	55
INTEGRASE	80
MATRIX	78
VIF	68
CAPSID, ENV_GP120	18
CAPSID, TAT	17
NUCLEOCAPSID, VPR	19
NUCLEOCAPSID, TAT	22
ENV_GP41, TAT	89
ENV_GP41, ENV_GP120	117
VPR, TAT	108
ENV_GP120, TAT	234
NEF, TAT	101
NEF, ENV_GP120	101
VPR, ENV_GP120	73
VPR, NEF	36
ENV_GP160, TAT	81
ENV_GP160, ENV_GP120	106
ENV_GP160, NEF	48
ENV_GP160, VPR	19
RT, TAT	28
RT, RETROPEPSIN	17
RT, MATRIX	15
RT, ENV_GP41	21
VIF, TAT	48
MATRIX, ENV_GP120	28
RETROPEPSIN, ENV_GP41	20
INTEGRASE, TAT	53
MATRIX, TAT	42
MATRIX, NEF	24
MATRIX, VPR	26
MATRIX, ENV_GP160	15
MATRIX, ENV_GP41	18
MATRIX, INTEGRASE	15
REV, TAT	33
GAG_PR55, ENV_GP41	16
REV, MATRIX	15
GAG_PR55, TAT	25
GAG_PR55, ENV_GP120	22
RT, ENV_GP120	24
ENV_GP41, NEF	57
ENV_GP41, ENV_GP160	71
RETROPEPSIN, NEF	16
RETROPEPSIN, ENV_GP160	15
RETROPEPSIN, TAT	29
RETROPEPSIN, ENV_GP120	22
RETROPEPSIN, ENV_GP120, TAT	19
RETROPEPSIN, NEF, TAT	14
RETROPEPSIN, ENV_GP41, TAT	19
ENV_GP41, ENV_GP120, TAT	76
ENV_GP41, NEF, TAT	48
ENV_GP41, ENV_GP160, NEF	26
ENV_GP41, NEF, ENV_GP120	45
ENV_GP41, ENV_GP160, ENV_GP120	63

Continued on next page ...

HIV-1 Proteins	**Support**
GAG_PR55, ENV_GP120, TAT	17
MATRIX, ENV_GP120, TAT	25
MATRIX, NEF, ENV_GP120	14
MATRIX, NEF, TAT	14
MATRIX, VPR, ENV_GP120	17
MATRIX, VPR, TAT	20
VIF, INTEGRASE, TAT	42
GAG_PR55, ENV_GP41, TAT	14
RT, ENV_GP120, TAT	20
RT, ENV_GP160, ENV_GP120	17
NUCLEOCAPSID, VPR, TAT	18
RT, ENV_GP41, TAT	18
RT, ENV_GP41, ENV_GP120	15
ENV_GP160, VPR, TAT	17
ENV_GP160, VPR, ENV_GP120	17
ENV_GP160, NEF, TAT	38
ENV_GP160, ENV_GP120, TAT	72
VPR, NEF, TAT	32
VPR, ENV_GP120, TAT	63
NEF, ENV_GP120, TAT	79
VPR, NEF, ENV_GP120	30
ENV_GP160, VPR, NEF	17
ENV_GP160, NEF, ENV_GP120	40
VPR, NEF, ENV_GP120, TAT	27
ENV_GP160, NEF, ENV_GP120, TAT	35
ENV_GP160, VPR, ENV_GP120, TAT	16
ENV_GP160, VPR, NEF, TAT	15
ENV_GP160, VPR, NEF, ENV_GP120	15
ENV_GP41, NEF, ENV_GP120, TAT	44
ENV_GP41, ENV_GP160, ENV_GP120, TAT	42
ENV_GP41, ENV_GP160, NEF, ENV_GP120	23
MATRIX, VPR, ENV_GP120, TAT	16
RT, ENV_GP160, ENV_GP120, TAT	14
RT, RETROPEPSIN, ENV_GP41, TAT	15
RT, ENV_GP41, ENV_GP160, ENV_GP120	14
ENV_GP160, VPR, NEF, ENV_GP120, TAT	14
ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	22
Minsupport=0.005%	
TAT	774
ENV_GP120	532
NEF	200
VPR	179
ENV_GP160	176
ENV_GP41	156
RETROPEPSIN	83
INTEGRASE	80
MATRIX	78
GAG_PR55	55
VIF	68
REV	65
NUCLEOCAPSID	25
VPU	22
CAPSID	33
P6	14
RT	42
MATRIX, TAT	42

Continued on next page ...

HIV-1 Proteins	**Support**
MATRIX, ENV_GP120	28
INTEGRASE, TAT	53
INTEGRASE, VPR	12
ENV_GP41, TAT	89
ENV_GP41, ENV_GP120	117
ENV_GP160, TAT	81
ENV_GP160, ENV_GP120	106
VPR, TAT	108
VPR, ENV_GP120	73
ENV_GP120, TAT	234
NEF, TAT	101
NEF, ENV_GP120	101
VPR, NEF	36
ENV_GP160, NEF	48
ENV_GP160, VPR	19
ENV_GP41, NEF	57
ENV_GP41, ENV_GP160	71
RETROPEPSIN, TAT	29
RETROPEPSIN, ENV_GP120	22
RETROPEPSIN, NEF	16
RETROPEPSIN, VPR	9
RETROPEPSIN, ENV_GP160	15
RETROPEPSIN, ENV_GP41	20
MATRIX, NEF	24
MATRIX, VPR	26
MATRIX, ENV_GP160	15
MATRIX, ENV_GP41	18
MATRIX, RETROPEPSIN	11
MATRIX, INTEGRASE	15
VIF, ENV_GP120	9
VIF, TAT	48
REV, TAT	33
REV, ENV_GP120	12
REV, VPR	10
REV, MATRIX	15
GAG_PR55, TAT	25
GAG_PR55, ENV_GP120	22
GAG_PR55, NEF	9
GAG_PR55, VPR	12
GAG_PR55, ENV_GP41	16
RT, ENV_GP120	24
GAG_PR55, MATRIX	11
RT, TAT	28
RT, VPR	11
RT, RETROPEPSIN	17
RT, MATRIX	15
RT, ENV_GP41	21
RT, VIF	8
CAPSID, ENV_GP120	18
CAPSID, TAT	17
CAPSID, NEF	10
CAPSID, MATRIX	8
CAPSID, GAG_PR55	7
NUCLEOCAPSID, VPR	19
NUCLEOCAPSID, TAT	22
VPU, TAT	11

Continued on next page ...

HIV-1 Proteins	Support
MATRIX, ENV_GP41, TAT	13
MATRIX, ENV_GP41, ENV_GP120	8
MATRIX, ENV_GP41, NEF	12
ENV_GP41, ENV_GP160, ENV_GP120	63
ENV_GP41, NEF, TAT	48
ENV_GP160, VPR, TAT	17
ENV_GP160, VPR, ENV_GP120	17
ENV_GP160, NEF, TAT	38
ENV_GP160, NEF, ENV_GP120	40
VPR, NEF, TAT	32
VPR, NEF, ENV_GP120	30
NEF, ENV_GP120, TAT	79
ENV_GP160, ENV_GP120, TAT	72
VPR, ENV_GP120, TAT	63
ENV_GP160, VPR, NEF	17
ENV_GP41, ENV_GP120, TAT	76
ENV_GP41, NEF, ENV_GP120	45
ENV_GP41, ENV_GP160, NEF	26
RETROPEPSIN, ENV_GP120, TAT	19
RETROPEPSIN, NEF, TAT	14
RETROPEPSIN, NEF, ENV_GP120	12
RETROPEPSIN, VPR, NEF	7
RETROPEPSIN, VPR, TAT	7
RETROPEPSIN, ENV_GP160, NEF	8
RETROPEPSIN, ENV_GP41, TAT	19
RETROPEPSIN, ENV_GP41, NEF	10
RETROPEPSIN, ENV_GP41, ENV_GP160	12
MATRIX, ENV_GP120, TAT	25
MATRIX, NEF, ENV_GP120	14
MATRIX, NEF, TAT	14
MATRIX, VPR, TAT	20
MATRIX, VPR, NEF	10
MATRIX, VPR, ENV_GP120	17
MATRIX, ENV_GP160, NEF	13
REV, VPR, TAT	7
REV, ENV_GP120, TAT	11
MATRIX, RETROPEPSIN, ENV_GP41	8
MATRIX, INTEGRASE, VPR	7
VIF, ENV_GP160, ENV_GP120	7
VIF, INTEGRASE, TAT	42
REV, MATRIX, TAT	10
GAG_PR55, ENV_GP120, TAT	17
GAG_PR55, VPR, ENV_GP120	10
GAG_PR55, ENV_GP41, TAT	14
RT, ENV_GP120, TAT	20
RT, NEF, TAT	13
RT, VPR, TAT	8
RT, ENV_GP160, ENV_GP120	17
RT, MATRIX, RETROPEPSIN	9
RT, ENV_GP41, TAT	18
RT, MATRIX, TAT	10
RT, ENV_GP41, ENV_GP120	15
CAPSID, ENV_GP120, TAT	13
CAPSID, MATRIX, TAT	7
CAPSID, NEF, TAT	9
CAPSID, NEF, ENV_GP120	9

Continued on next page ...

HIV-1 Proteins	**Support**
CAPSID, ENV_GP41, ENV_GP120	11
NUCLEOCAPSID, VPR, TAT	18
VPU, ENV_GP120, TAT	10
VPU, NEF, TAT	9
ENV_GP41, VPR, NEF, TAT	13
VPR, NEF, ENV_GP120, TAT	27
ENV_GP160, NEF, ENV_GP120, TAT	35
ENV_GP160, VPR, ENV_GP120, TAT	16
ENV_GP160, VPR, NEF, TAT	15
ENV_GP41, NEF, ENV_GP120, TAT	44
ENV_GP160, VPR, NEF, ENV_GP120	15
ENV_GP41, ENV_GP160, ENV_GP120, TAT	42
ENV_GP41, ENV_GP160, NEF, ENV_GP120	23
RETROPEPSIN, NEF, ENV_GP120, TAT	11
RETROPEPSIN, ENV_GP160, ENV_GP120, TAT	13
RETROPEPSIN, ENV_GP41, ENV_GP120, TAT	13
RETROPEPSIN, ENV_GP41, NEF, TAT	9
RETROPEPSIN, ENV_GP41, ENV_GP160, NEF	7
MATRIX, NEF, ENV_GP120, TAT	12
MATRIX, VPR, ENV_GP120, TAT	16
MATRIX, VPR, NEF, ENV_GP120	9
MATRIX, ENV_GP160, ENV_GP120, TAT	12
MATRIX, VPR, NEF, TAT	9
MATRIX, ENV_GP41, NEF, TAT	9
MATRIX, ENV_GP41, ENV_GP160, NEF	9
MATRIX, RETROPEPSIN, ENV_GP41, TAT	7
GAG_PR55, MATRIX, VPR, ENV_GP120	9
GAG_PR55, ENV_GP160, ENV_GP120, TAT	9
RT, NEF, ENV_GP120, TAT	11
RT, ENV_GP160, ENV_GP120, TAT	14
RT, RETROPEPSIN, ENV_GP41, TAT	15
RT, ENV_GP41, ENV_GP160, ENV_GP120	14
RT, ENV_GP41, ENV_GP120, TAT	12
RT, MATRIX, ENV_GP41, TAT	8
RT, ENV_GP41, NEF, TAT	8
CAPSID, NEF, ENV_GP120, TAT	8
CAPSID, ENV_GP41, ENV_GP120, TAT	10
VPU, ENV_GP160, NEF, TAT	7
VPU, NEF, ENV_GP120, TAT	8
ENV_GP160, VPR, NEF, ENV_GP120, TAT	14
ENV_GP41, VPR, NEF, ENV_GP120, TAT	12
ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	22
RETROPEPSIN, ENV_GP160, NEF, ENV_GP120, TAT	7
RETROPEPSIN, ENV_GP41, ENV_GP160, ENV_GP120, TAT	11
RETROPEPSIN, ENV_GP41, NEF, ENV_GP120, TAT	7
MATRIX, ENV_GP160, NEF, ENV_GP120, TAT	10
MATRIX, ENV_GP160, VPR, ENV_GP120, TAT	8
MATRIX, VPR, NEF, ENV_GP120, TAT	8
MATRIX, ENV_GP41, NEF, ENV_GP120, TAT	7
GAG_PR55, ENV_GP41, ENV_GP160, ENV_GP120, TAT	8
GAG_PR55, MATRIX, VPR, ENV_GP120, TAT	8
RT, ENV_GP41, ENV_GP160, ENV_GP120, TAT	11
RT, ENV_GP160, NEF, ENV_GP120, TAT	7
RT, MATRIX, RETROPEPSIN, ENV_GP41, TAT	7
CAPSID, ENV_GP41, NEF, ENV_GP120, TAT	7
MATRIX, ENV_GP160, VPR, NEF, ENV_GP120, TAT	7

Continued on next page ...

HIV-1 Proteins	**Support**
RT, RETROPEPSIN, ENV_GP41, ENV_GP160, ENV_GP120, TAT	9
Minsupport=0.001%	
TAT	774
ENV_GP120	532
ENV_GP120, TAT	234
NEF	200
NEF, TAT	101
NEF, ENV_GP120	101
NEF, ENV_GP120, TAT	79
VPR	179
VPR, TAT	108
VPR, ENV_GP120	73
VPR, ENV_GP120, TAT	63
VPR, NEF	36
VPR, NEF, TAT	32
VPR, NEF, ENV_GP120	30
VPR, NEF, ENV_GP120, TAT	27
ENV_GP160	176
ENV_GP160, TAT	81
ENV_GP160, ENV_GP120	106
ENV_GP160, ENV_GP120, TAT	72
ENV_GP160, NEF	48
ENV_GP160, NEF, TAT	38
ENV_GP160, NEF, ENV_GP120	40
ENV_GP160, NEF, ENV_GP120, TAT	35
ENV_GP160, VPR	19
ENV_GP160, VPR, TAT	17
ENV_GP160, VPR, ENV_GP120	17
ENV_GP160, VPR, ENV_GP120, TAT	16
ENV_GP160, VPR, NEF	17
ENV_GP160, VPR, NEF, TAT	15
ENV_GP160, VPR, NEF, ENV_GP120	15
ENV_GP160, VPR, NEF, ENV_GP120, TAT	14
ENV_GP41	156
ENV_GP41, TAT	89
ENV_GP41, ENV_GP120	117
ENV_GP41, ENV_GP120, TAT	76
ENV_GP41, NEF	57
ENV_GP41, NEF, TAT	48
ENV_GP41, NEF, ENV_GP120, TAT	44
ENV_GP41, NEF, ENV_GP120	45
ENV_GP41, VPR, NEF, TAT	13
ENV_GP41, VPR, NEF, ENV_GP120, TAT	12
ENV_GP41, ENV_GP160	71
ENV_GP41, ENV_GP160, ENV_GP120	63
ENV_GP41, ENV_GP160, ENV_GP120, TAT	42
ENV_GP41, ENV_GP160, NEF	26
ENV_GP41, ENV_GP160, NEF, ENV_GP120	23
ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	22
ENV_GP41, ENV_GP160, VPR, NEF, ENV_GP120, TAT	6
RETROPEPSIN	83
RETROPEPSIN, TAT	29
RETROPEPSIN, ENV_GP120	22
RETROPEPSIN, ENV_GP120, TAT	19
RETROPEPSIN, NEF, TAT	14
RETROPEPSIN, NEF, ENV_GP120	12

Continued on next page ...

HIV-1 Proteins	**Support**
RETROPEPSIN, NEF, ENV_GP120, TAT	11
RETROPEPSIN, NEF	16
RETROPEPSIN, VPR	9
RETROPEPSIN, VPR, NEF	7
RETROPEPSIN, VPR, NEF, TAT	6
RETROPEPSIN, VPR, TAT	7
RETROPEPSIN, VPR, NEF, ENV_GP120	5
RETROPEPSIN, VPR, ENV_GP120	6
RETROPEPSIN, VPR, ENV_GP120, TAT	5
RETROPEPSIN, VPR, NEF, ENV_GP120, TAT	4
RETROPEPSIN, ENV_GP160	15
RETROPEPSIN, ENV_GP160, VPR, ENV_GP120, TAT	2
RETROPEPSIN, ENV_GP160, ENV_GP120, TAT	13
RETROPEPSIN, ENV_GP160, NEF	8
RETROPEPSIN, ENV_GP160, NEF, ENV_GP120, TAT	7
RETROPEPSIN, ENV_GP41, TAT	19
RETROPEPSIN, ENV_GP41	20
RETROPEPSIN, ENV_GP41, ENV_GP120, TAT	13
RETROPEPSIN, ENV_GP41, NEF, TAT	9
RETROPEPSIN, ENV_GP41, NEF	10
RETROPEPSIN, ENV_GP41, VPR, NEF, TAT	2
RETROPEPSIN, ENV_GP41, ENV_GP160	12
RETROPEPSIN, ENV_GP41, ENV_GP160, ENV_GP120, TAT	11
RETROPEPSIN, ENV_GP41, ENV_GP160, NEF	7
RETROPEPSIN, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	6
RETROPEPSIN, ENV_GP41, NEF, ENV_GP120, TAT	7
INTEGRASE	80
INTEGRASE, TAT	53
INTEGRASE, NEF	3
INTEGRASE, NEF, TAT	2
INTEGRASE, VPR	12
INTEGRASE, VPR, TAT	5
MATRIX	78
MATRIX, TAT	42
MATRIX, ENV_GP120	28
MATRIX, ENV_GP120, TAT	25
MATRIX, NEF	24
MATRIX, NEF, ENV_GP120	14
MATRIX, NEF, ENV_GP120, TAT	12
MATRIX, NEF, TAT	14
MATRIX, VPR	26
MATRIX, VPR, ENV_GP120	17
MATRIX, VPR, ENV_GP120, TAT	16
MATRIX, VPR, TAT	20
MATRIX, VPR, NEF, ENV_GP120	9
MATRIX, VPR, NEF	10
MATRIX, ENV_GP160, ENV_GP120, TAT	12
MATRIX, ENV_GP160	15
MATRIX, ENV_GP160, NEF, ENV_GP120, TAT	10
MATRIX, ENV_GP160, NEF	13
MATRIX, ENV_GP160, VPR, ENV_GP120, TAT	8
MATRIX, VPR, NEF, ENV_GP120, TAT	8
MATRIX, ENV_GP160, VPR, NEF, ENV_GP120, TAT	7
MATRIX, VPR, NEF, TAT	9
MATRIX, ENV_GP41	18
MATRIX, ENV_GP41, ENV_GP120	8

Continued on next page ...

HIV-1 Proteins	Support
MATRIX, ENV_GP41, NEF	12
MATRIX, ENV_GP41, NEF, ENV_GP120, TAT	7
MATRIX, ENV_GP41, VPR, NEF, ENV_GP120, TAT	5
MATRIX, ENV_GP41, NEF, TAT	9
MATRIX, ENV_GP41, TAT	13
MATRIX, ENV_GP41, VPR, NEF, TAT	6
MATRIX, ENV_GP41, ENV_GP160, NEF	9
MATRIX, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	6
MATRIX, ENV_GP41, ENV_GP160, VPR, NEF, ENV_GP120, TAT	4
MATRIX, RETROPEPSIN	11
MATRIX, RETROPEPSIN, ENV_GP41, NEF	4
MATRIX, RETROPEPSIN, ENV_GP41, TAT	7
MATRIX, RETROPEPSIN, ENV_GP41, NEF, TAT	3
MATRIX, RETROPEPSIN, ENV_GP41	8
MATRIX, RETROPEPSIN, ENV_GP41, ENV_GP160, NEF	2
MATRIX, INTEGRASE	15
MATRIX, INTEGRASE, TAT	4
MATRIX, INTEGRASE, VPR	7
VIF	68
VIF, TAT	48
VIF, NEF	6
VIF, ENV_GP41, ENV_GP160, ENV_GP120	5
VIF, ENV_GP120	9
VIF, ENV_GP160, ENV_GP120	7
VIF, ENV_GP41, ENV_GP160, ENV_GP120, TAT	2
VIF, ENV_GP160, ENV_GP120, TAT	4
VIF, INTEGRASE, TAT	42
VIF, NEF, TAT	4
VIF, MATRIX, VPR, NEF, ENV_GP120	2
VIF, VPR, NEF, ENV_GP120	3
VIF, MATRIX, NEF, ENV_GP120	3
REV	65
REV, TAT	33
REV, ENV_GP120, TAT	11
REV, ENV_GP120	12
REV, VPR	10
REV, VPR, TAT	7
REV, VPR, ENV_GP120, TAT	3
REV, ENV_GP160, TAT	5
REV, NEF, TAT	5
REV, ENV_GP160, NEF, ENV_GP120, TAT	4
REV, ENV_GP160, VPR, NEF, ENV_GP120, TAT	2
REV, ENV_GP41	3
REV, ENV_GP41, ENV_GP120	2
REV, RETROPEPSIN	6
REV, INTEGRASE, VPR	4
REV, MATRIX	15
REV, MATRIX, TAT	10
REV, MATRIX, VPR	5
REV, MATRIX, RETROPEPSIN	5
REV, MATRIX, VPR, TAT	4
GAG_PR55	55
GAG_PR55, TAT	25
GAG_PR55, ENV_GP120	22
GAG_PR55, ENV_GP120, TAT	17
GAG_PR55, NEF	9

Continued on next page ...

HIV-1 Proteins	**Support**
GAG_PR55, VPR	12
GAG_PR55, VPR, ENV_GP120	10
GAG_PR55, ENV_GP41, TAT	14
GAG_PR55, ENV_GP41	16
GAG_PR55, ENV_GP41, NEF, TAT	6
GAG_PR55, ENV_GP41, ENV_GP160, ENV_GP120, TAT	8
GAG_PR55, ENV_GP160, ENV_GP120, TAT	9
GAG_PR55, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	4
GAG_PR55, NEF, ENV_GP120	5
GAG_PR55, RETROPEPSIN	2
GAG_PR55, INTEGRASE	2
GAG_PR55, MATRIX, VPR, ENV_GP120, TAT	8
GAG_PR55, MATRIX	11
GAG_PR55, MATRIX, VPR, ENV_GP120	9
GAG_PR55, VIF	4
RT	42
RT, ENV_GP120	24
RT, ENV_GP120, TAT	20
RT, TAT	28
RT, NEF, ENV_GP120, TAT	11
RT, NEF, TAT	13
RT, VPR	11
RT, VPR, TAT	8
RT, VPR, ENV_GP120, TAT	6
RT, VPR, NEF, ENV_GP120, TAT	5
RT, VPR, NEF, TAT	6
RT, ENV_GP160, ENV_GP120, TAT	14
RT, ENV_GP160, ENV_GP120	17
RT, RETROPEPSIN, ENV_GP41, ENV_GP160, ENV_GP120, TAT	9
RT, RETROPEPSIN, ENV_GP41, TAT	15
RT, ENV_GP41, ENV_GP160, ENV_GP120	14
RT, RETROPEPSIN	17
RT, ENV_GP41, ENV_GP160, ENV_GP120, TAT	11
RT, ENV_GP41, ENV_GP120, TAT	12
RT, RETROPEPSIN, ENV_GP41, NEF, TAT	6
RT, RETROPEPSIN, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	4
RT, ENV_GP41, NEF, ENV_GP120, TAT	6
RT, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	5
RT, ENV_GP160, NEF, ENV_GP120, TAT	7
RT, MATRIX	15
RT, ENV_GP41	21
RT, MATRIX, RETROPEPSIN	9
RT, MATRIX, RETROPEPSIN, ENV_GP41, TAT	7
RT, ENV_GP41, TAT	18
RT, MATRIX, ENV_GP41, TAT	8
RT, MATRIX, TAT	10
RT, INTEGRASE	3
RT, MATRIX, VPR	5
RT, MATRIX, INTEGRASE, VPR	2
RT, VIF	8
RT, VIF, ENV_GP41, ENV_GP160, ENV_GP120	4
RT, ENV_GP41, ENV_GP120	15
RT, VIF, ENV_GP160, ENV_GP120	6
RT, VIF, TAT	4
RT, REV, MATRIX, RETROPEPSIN	4
RT, REV, MATRIX	6

Continued on next page ...

HIV-1 Proteins	Support
RT, REV, MATRIX, RETROPEPSIN, ENV_GP41, NEF, TAT	2
RT, ENV_GP41, NEF, TAT	8
RT, MATRIX, ENV_GP41, NEF, TAT	4
RT, MATRIX, RETROPEPSIN, ENV_GP41, NEF, TAT	3
RT, REV, MATRIX, NEF, TAT	4
RT, MATRIX, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	2
RT, REV, MATRIX, ENV_GP160, NEF, ENV_GP120, TAT	3
GAG_PR55, VIF, ENV_GP120	2
RT, VIF, ENV_GP160, ENV_GP120, TAT	3
CAPSID	33
CAPSID, ENV_GP120	18
CAPSID, GAG_PR55, ENV_GP41, ENV_GP160, ENV_GP120, TAT	5
CAPSID, GAG_PR55	7
CAPSID, GAG_PR55, ENV_GP120	6
CAPSID, ENV_GP41, ENV_GP120, TAT	10
CAPSID, ENV_GP41, ENV_GP160, ENV_GP120, TAT	6
CAPSID, TAT	17
CAPSID, ENV_GP41, ENV_GP120	11
CAPSID, ENV_GP120, TAT	13
CAPSID, GAG_PR55, NEF, ENV_GP120	3
CAPSID, ENV_GP41, NEF, ENV_GP120, TAT	7
CAPSID, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	3
CAPSID, NEF, TAT	9
CAPSID, NEF, ENV_GP120, TAT	8
CAPSID, GAG_PR55, VPR	2
CAPSID, VPR	4
CAPSID, VPR, NEF, ENV_GP120	3
CAPSID, MATRIX, VPR, NEF, ENV_GP120	2
CAPSID, NEF	10
CAPSID, NEF, ENV_GP120	9
CAPSID, MATRIX, ENV_GP120	5
CAPSID, MATRIX, NEF, ENV_GP120	3
CAPSID, MATRIX	8
CAPSID, RT, ENV_GP41, VPR, NEF, ENV_GP120, TAT	2
RT, ENV_GP41, VPR, NEF, TAT	3
CAPSID, MATRIX, ENV_GP120, TAT	4
CAPSID, MATRIX, NEF, ENV_GP120, TAT	2
RT, MATRIX, ENV_GP41, VPR, NEF, TAT	2
RT, MATRIX, ENV_GP160, NEF, ENV_GP120, TAT	4
RT, MATRIX, ENV_GP160, VPR, NEF, ENV_GP120, TAT	2
CAPSID, MATRIX, TAT	7
NUCLEOCAPSID, RT	5
NUCLEOCAPSID	25
NUCLEOCAPSID, RT, MATRIX, RETROPEPSIN, ENV_GP41, NEF, TAT	2
NUCLEOCAPSID, RT, MATRIX	3
NUCLEOCAPSID, TAT	22
NUCLEOCAPSID, RETROPEPSIN, NEF, TAT	3
NUCLEOCAPSID, REV, MATRIX, TAT	4
NUCLEOCAPSID, RT, MATRIX, VPR	2
NUCLEOCAPSID, VPR, TAT	18
NUCLEOCAPSID, RETROPEPSIN, VPR, NEF, TAT	2
RT, MATRIX, NEF, TAT	6
RT, MATRIX, VPR, NEF, TAT	3
NUCLEOCAPSID, MATRIX, TAT	5
NUCLEOCAPSID, VPR	19
NUCLEOCAPSID, MATRIX	6

Continued on next page ...

HIV-1 Proteins	**Support**
NUCLEOCAPSID, VPR, ENV_GP120, TAT	2
VPU	22
VPU, ENV_GP120, TAT	10
VPU, TAT	11
VPU, VPR, TAT	6
VPU, VPR, ENV_GP120, TAT	5
VPU, ENV_GP160, NEF, TAT	7
VPU, ENV_GP160, NEF, ENV_GP120, TAT	6
VPU, NEF, ENV_GP120, TAT	8
VPU, VPR, NEF, TAT	5
VPU, ENV_GP160, VPR, NEF, TAT	3
VPU, ENV_GP41, NEF, ENV_GP120, TAT	5
VPU, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	4
VPU, ENV_GP41, VPR, NEF, ENV_GP120, TAT	2
VPU, VPR, NEF, ENV_GP120, TAT	4
VPU, RETROPEPSIN	5
VPU, RETROPEPSIN, VPR, NEF, ENV_GP120, TAT	2
VPU, ENV_GP160, VPR, NEF, ENV_GP120, TAT	2
VIF, ENV_GP160, NEF, ENV_GP120, TAT	3
VIF, ENV_GP160, VPR, NEF, ENV_GP120, TAT	2
VPU, REV	4
VPU, REV, MATRIX, RETROPEPSIN	3
VPU, GAG_PR55	2
P6	14
P6, NEF	3
P6, RT, REV, VIF, MATRIX, ENV_GP160, NEF, ENV_GP120, TAT	2
P6, RT, VIF	3
P6, VIF	4
P6, REV, MATRIX, TAT	5
P6, NUCLEOCAPSID	4
RETROPEPSIN, ENV_GP160, VPR, NEF, ENV_GP120, TAT	1
INTEGRASE, RETROPEPSIN, VPR, NEF, ENV_GP120, TAT	1
MATRIX, INTEGRASE, NEF	1
VIF, RETROPEPSIN, ENV_GP120	1
VIF, INTEGRASE, NEF, TAT	1
REV, MATRIX, INTEGRASE, VPR, TAT	3
GAG_PR55, RETROPEPSIN, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	1
GAG_PR55, INTEGRASE, ENV_GP41	1
GAG_PR55, MATRIX, ENV_GP160, VPR, ENV_GP120, TAT	1
GAG_PR55, MATRIX, ENV_GP41	1
GAG_PR55, MATRIX, INTEGRASE	1
GAG_PR55, REV	1
RT, VIF, INTEGRASE, TAT	1
RT, REV, MATRIX, RETROPEPSIN, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	1
RT, GAG_PR55, VIF, ENV_GP41, ENV_GP160, ENV_GP120, TAT	1
CAPSID, GAG_PR55, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	2
CAPSID, GAG_PR55, RETROPEPSIN, VPR	1
CAPSID, GAG_PR55, VIF, MATRIX, VPR, NEF, ENV_GP120	1
CAPSID, RT, MATRIX, ENV_GP41, ENV_GP160, VPR, NEF, ENV_GP120, TAT	1
NUCLEOCAPSID, RT, REV, MATRIX, RETROPEPSIN, ENV_GP41, NEF, TAT	1
NUCLEOCAPSID, RT, MATRIX, RETROPEPSIN, ENV_GP41, VPR, NEF, TAT	1
NUCLEOCAPSID, RT, MATRIX, INTEGRASE, VPR	1
VPU, RETROPEPSIN, ENV_GP41, VPR, NEF, ENV_GP120, TAT	1
VPU, VIF, ENV_GP41, ENV_GP160, VPR, NEF, ENV_GP120, TAT	1
VPU, REV, ENV_GP160, VPR, NEF, ENV_GP120, TAT	1
VPU, RT, REV, MATRIX, RETROPEPSIN	2

Continued on next page ...

HIV-1 Proteins	**Support**
VPU, NUCLEOCAPSID, RETROPEPSIN, VPR, NEF, ENV_GP120, TAT	1
P6, GAG_PR55, NEF	1
P6, RT, REV, VIF, MATRIX, ENV_GP160, VPR, NEF, ENV_GP120, TAT	1
P6, NUCLEOCAPSID, RT, VIF	1
p1, P6, NUCLEOCAPSID, CAPSID, REV, MATRIX, TAT	3
Pol, VPU, GAG_PR55, ENV_GP41, ENV_GP160, NEF, ENV_GP120, TAT	1
VIF, NEF, ENV_GP120	4
VPU, NEF, TAT	9

Bibliography

[Abbass 2001] H. A. Abbass, R. Sarker and C. Newton. *PDE, a Pareto-frontier differential evolution approach for multi-objective optimization problems.* In Proceedings of the IEEE Congress on Evolutionary Computation (CEC), volume 2, pages 971–978, 2001. (Cited on page 53.)

[Agarwal 1999] R. C. Agarwal, C. C. Aggarwal and V. V. Prasad. *A tree projection algorithm for generation of frequent itemsets.* In Proceedings of the HPDM workshop, 1999. (Cited on page 46.)

[Aggarwal 1998] C. C. Aggarwal and P. S. Yu. *A new framework for itemset generation.* In Proceedings of the ACM Symposium on Principles of Database Systems (PODS), pages 18–24, 1998. (Cited on page 42.)

[Agrawal 1993] R. Agrawal, T. Imielinski and A. N. Swami. *Mining association rules between sets of items in large databases.* In Proceedings of ACM International Conference on Management of Data (SIGMOD), pages 207–216, May 1993. (Cited on pages 2, 42 and 44.)

[Agrawal 1994] R. Agrawal and R. Srikant. *Fast algorithms for mining association rules in large databases.* In Proceedings of International Conference on Very Large Data Bases (VLDB), pages 487–499, September 1994. (Cited on pages 3 and 45.)

[Agrawal 1996] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. I. Verkamo. *Fast Discovery of Association Rules.* In Advances in Knowledge Discovery and Data Mining, pages 307–328. AAAI/MIT Press, 1996. (Cited on page 8.)

[Agresti 1990] A. Agresti. Categorical data analysis. John Wiley & Sons, New York, 1990. (Cited on page 42.)

[Agresti 1996] A. Agresti. An introduction to categorical data analysis. John Wiley & Sons, New York, 1996. (Cited on page 42.)

[Alatas 2008] B. Alatas, E. Akin and A. Karci. *MODENAR: Multi-objective differential evolution algorithm for mining numeric association rules.* Applied Soft Computing, vol. 8, pages 646–656, 2008. (Cited on page 53.)

[Allocco 2004] D. J Allocco, I. S Kohane and A. J Butte. *Quantifying the relationship between co-expression, co-regulation and gene function.* BMC Bioinformatics, vol. 5, no. 18, 2004. (Cited on pages 24 and 25.)

[Anandhavalli 2010] M. Anandhavalli, M. K. Ghose and K. Gauthaman. *Interestingness measure for mining spatial gene expression data using association rule.* Journal of Computing, vol. 2, no. 1, pages 110–114, 2010. (Cited on pages 25 and 41.)

[Anandhavilli 2010] M. Anandhavilli, M. K. Ghose and K. Gauthaman. *Association rule mining in genomics.* International Journal of Computer Theory and Engineering, vol. 2, no. 2, pages 269–273, 2010. (Cited on page 25.)

[Anderson 1997] L. Anderson and J. Seilhamer. *A comparison of selected mRNA and protein abundances in human liver*. Electrophoresis, vol. 18, pages 533–537, 1997. (Cited on page 26.)

[Angiulli 2001] F. Angiulli, G. Ianni and L. Palopoli. *On the complexity of mining association rules*. In Proceedings of the Italian Symposium on Advanced Database Systems (SEBD), pages 177–184, 2001. (Cited on page 45.)

[Arkin 2004] M. R. Arkin and J. A. Wells. *Small-molecule inhibitors of protein-protein interactions: Progressing towards the dream*. Nat. Rev. Drug Discov., vol. 3, pages 301–317, 2004. (Cited on pages 5 and 13.)

[Aze 2002] J. Aze and Y. Kodrato. *A study of the e_ect of noisy data in rule extraction systems*. In Proceedings of the European Meeting on Cybernetics and Systems Research, volume 2, pages 781–786, 2002. (Cited on page 42.)

[Barker 2001] W. C. Barker, J. S. Garavelli, Z. Hou, H. Huang, R. S. Ledley, P. B. McGarvey, H. W. Mewes, B. C. Orcutt, F. Pfeiffer, A. Tsugita, C. R. Vinayaka, C. Xiao, L. S. Yeh and C. Wu. *Protein information resource: A community resource for expert annotation of protein data*. Nucleic Acids Research, vol. 29, pages 29–32, 2001. (Cited on page 26.)

[Bastide 2000] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme and L. Lakhal. *Mining frequent patterns with counting inference*. SIGKDD Explorations, vol. 2, no. 2, pages 66–75, 2000. (Cited on pages 8, 10 and 47.)

[Bayardo 1998] R. J. Bayardo. *Efficiently mining long patterns from databases*. In Proceedings of the ACM SIGMOD conference, pages 85–93, 1998. (Cited on page 46.)

[Bayardo 1999] R. J. Bayardo and R. Agrawal. *Mining the most interesting rules*. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 145–153, 1999. (Cited on page 52.)

[Becquet 2002] C. Becquet, S. Blachon, B. Jeudy, J.-F. Boulicaut and O. Gandrillon. *Strong-association-rule mining for large-scale gene-expression data analysis: A case study on human sage data*. Genome Biology, vol. 3, no. 12, 2002. (Cited on page 24.)

[Bell 2011] L. Bell, R. Chowdhary, J.S. Liu, X. Niu and J. Zhang. *Integrated Bio-Entity Network: A system for biological knowledge discovery*. PLoS ONE, vol. 6, 2011. (Cited on pages 173 and 180.)

[Ben-Hur 2005a] A. Ben-Hur and W. S. Noble. *Kernel methods for predicting protein-protein interactions*. Bioinformatics, vol. 21, no. 1, pages 38–46, 2005. (Cited on page 5.)

[Ben-Hur 2005b] A. Ben-Hur and W. S. Noble. *Kernel methods for predicting protein-protein interactions*. Bioinformatics, vol. 21, no. Suppl 1, pages i38–i46, 2005. (Cited on page 158.)

[Berardi 2005] M. Berardi, M. Lapi, P. Leo and C. Loglisci. *Mining generalized association rules on biomedical literature*. In Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence, IEA/AIE'2005, pages 500–509, London, UK, UK, 2005. Springer-Verlag. (Cited on page 19.)

[Besemann 2004] C. Besemann, A. Denton, A. Yekkirala, R. Hutchison and M. Anderson. *Differential association rule mining for the study of protein-protein interaction networks.* In Proceedings of the ACM SIGKDD Workshop on Data Mining in Bioinformatics (BioKDD), pages 72–80, 2004. (Cited on page 33.)

[Bhasin 2003] M. Bhasin, H. Singh and G. P. S. Raghava. *MHCBN: A comprehensive database of MHC binding and non-binding peptides.* Bioinformatics, vol. 19, no. 5, pages 665–666, 2003. (Cited on page 32.)

[Birkhoff 1995a] G. Birkhoff. Lattice theory, volume 25 of *Colloquium Publications.* American Mathematical Society, 1995. (Cited on page 3.)

[Birkhoff 1995b] G. Birkhoff. Lattice theory, 3rd edition, volume 25. American Mathematical Society, 1995. (Cited on page 47.)

[Boeckmann 2003] B. Boeckmann, A. Bairoch, R. Apweiler, M.-C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout and M. Schneider. *The SWISS-PROT protein knowledge base and its supplement TrEMBL.* Nucleic Acids Research, vol. 31, pages 365–370, 2003. (Cited on page 26.)

[Boulicaut 2000] J.-F. Boulicaut, A. Bykowski and C. Rigotti. *Approximation of frequency queries by means of free-sets.* In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), pages 75–85, 2000. (Cited on page 48.)

[Boulicaut 2001] J.-F. Boulicaut and B. Jeudy. *Mining free itemsets under constraints.* In Proceedings of the International Database Engineering and Application Symposium (IDEAS), pages 322–329, 2001. (Cited on page 50.)

[Boulicaut 2003] J. F. Boulicaut, A. Bykowski and C. Rigotti. *Free-sets: A condensed representation of boolean data for the approximation of frequency queries.* Data Mining and Knowledge Discovery, vol. 7, no. 1, pages 5–22, 2003. (Cited on pages 48 and 49.)

[Boulicaut 2008] J.-F. Boulicaut and J. Besson. *Actionability and Formal Concepts: A data mining perspective.* In Proceedings of the International Conference on Formal Concept Analysis (ICFCA), pages 14–31, 2008. (Cited on page 43.)

[Bras 2010] Y. Le Bras, P. Lenca and S. Lallich. *Mining interesting rules without support requirement: A general universal existential upward closure property.* Annals of Information Systems, vol. 8, pages 75–98, 2010. (Cited on page 42.)

[Brass 2008] A. L. Brass, D. M. Dykxhoorn, Y. Benita, N. Yan, A. Engelman, R. J. Xavier, J. Lieberman and S. J. Elledge. *Identification of host proteins required for HIV infection through a functional genomic screen.* Science, vol. 319, no. 5865, pages 921–926, 2008. (Cited on page 13.)

[Breiman 1984] L. Breiman, J. Freidman, R. Olshen and C. Stone. *Classification and regression trees.* Wadsworth and Brooks, 1984. (Cited on page 42.)

[Brin 1997a] S. Brin, R. Motwani and C. Silverstein. *Beyond market baskets: Generalizing association rules to correlations.* In Proceedings of the ACM International Conference on Management of Data (SIGMOD), pages 265–276, 1997. (Cited on page 42.)

[Brin 1997b] S. Brin, R. Motwani, J. D. Ullman and S. Tsur. *Dynamic itemset counting and implication rules for market basket data.* In Proceedings of ACM International Conference on Management of Data (SIGMOD), pages 255–264, 1997. (Cited on pages 42, 45 and 162.)

[Brin 1998] S. Brin, R. Motwani and C. Silverstein. *Beyond market baskets: Generalizing association rules to dependence rules.* Data Mining and Knowledge Discovery, vol. 2, no. 1, pages 39–68, 1998. (Cited on page 42.)

[Brisson 2004] L. Brisson, N. Pasquier, C. Hebert and M. Collard. *HASAR: Mining sequential association rules for atherosclerosis risk factor analysis.* In Proceedings of the Discovery Challenge of the International Conference on Principles of Knowledge Discovery in Databases (PKDD), 2004. (Cited on page 18.)

[Calders 2005] T. Calders, C. Rigotti and J.-F. Boulicaut. *A survey on condensed representations for frequent sets.* Constraint-Based Mining and Inductive Databases, vol. 3848, pages 64–80, 2005. (Cited on page 50.)

[Calero 2003] J. Calero, G. Delgado, M. Sanchez-Maranon, D. Sanchez, J. M. Serrano and M. A. V. Miranda. *Helping User to Discover Association Rules: A Case in Soil Color as Aggregation of Other Soil Properties.* In ICEIS-2003, pages 533–540, 2003. (Cited on page 22.)

[Cao 2006] L. Cao and C. Zhang. *Domain-Driven Actionable Knowledge Discovery in the Real World.* In Advances in Knowledge Discovery and Data Mining, volume 3918 of *Lecture Notes in Computer Science*, pages 821–830. Springer-Verlog, 2006. (Cited on page 43.)

[Cao 2007] L. Cao, D. Luo and C. Zhang. *Knowledge actionability: Satisfying technical and business interestingness.* International Journal of Business Intelligence and Data Mining, vol. 2, no. 4, pages 496–514, 2007. (Cited on page 43.)

[Cao 2010] L. Cao. *Flexible frameworks for actionable knowledge discovery.* IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 9, pages 1299–1312, 2010. (Cited on page 43.)

[Cao 2012] L. Cao. *Actionable knowledge discovery and delivery.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, no. 2, pages 149–163, 2012. (Cited on page 43.)

[Carmona-Saez 2006] P. Carmona-Saez, M. Chagoyen, A. Rodriguez, O. Trelles, J. M. Carazo and A. Pascual-Montano. *Integrated analysis of gene expression by association rules discovery.* BMC Bioinformatics, vol. 7, no. 54, 2006. (Cited on page 24.)

[Ceglar 2006] A. Ceglar and J.F. Roddick. *Association mining.* ACM Computing Surveys, vol. 38, 2006. (Cited on pages 2, 8 and 45.)

[Chae 2001] Y. M. Chae, S. H. Ho, K. W. Cho, D. H. Lee and S. H. Ji. *Data Mining approach to policy analysis in a health insurance domain.* International Journal of Medical Informatics, vol. 62, pages 103–111, 2001. (Cited on page 20.)

[Chen 2001] G. Chen, D. Liu and J. Li. *Influence and conditional influence - New interestingness measures in association rule mining.* In Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pages 1440–1443, 2001. (Cited on page 42.)

[Chen 2009] B. Chen and S. Kockara. *Mining Positional Association Super-Rules on Fixed-Size Protein Sequence motifs.* In Proceedings of IEEE BIBE 2009, pages 1–8, 2009. (Cited on page 29.)

[Chen 2010] B. Chen, M. Miller, T. Montgomery and T. Griffin. *Clustering Using Positional Association Rule Algorithm on Protein Sequence Motifs.* In Proceedings of BIOCOMP2010, pages 75–80, 2010. (Cited on page 29.)

[Chen 2011] B. Chen, B. Nordin, S. Bobba, D. Singireddy, B. Taylor, S. Kockara and M. Mete. *Clustering on Protein Sequence Motifs using SCAN and Positional Association Rule Algorithms.* In Proceedings of WORLDCOMP2011, 2011. (Cited on page 29.)

[Chirmule 1994] N. Chirmule, N. Oyaizu, C. Saxinger and S. Pahwa. *Nef protein of HIV-1 has B-cell stimulatory activity.* AIDS, vol. 6, no. 8, pages 733–4, 1994. (Cited on page 167.)

[Chiu 2008] H.-W. Chiu and F.-H. Hung. *Association rule mining from yeast protein interaction to assist protein-protein interaction prediction.* Biomedical Soft Computing and Human Sciences, vol. 13, no. 1, pages 3–6, 2008. (Cited on page 33.)

[Cho 1998] R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart and R. W. Davis. *A genome-wide transcriptional analysis of the mitotic cell cycle.* Molecular Cell, vol. 2, no. 1, pages 65–73, 1998. (Cited on page 25.)

[Clark 1991] P. Clark and R. Boswell. *Rule induction with CN2: Some recent improvements.* In Proceedings of the European Working Session on Learning (EWSL), pages 151–163, 1991. (Cited on page 42.)

[Cohen 1960] J. Cohen. *A coefficient of agreement for nominal scales.* Educational and Psychological Measurement, vol. 20, pages 37–46, 1960. (Cited on page 42.)

[Colantonio 2005] D. A. Colantonio and D. W. Chan. *The clinical application of proteomics.* Clinica Chimica Acta, vol. 357, pages 151–158, 2005. (Cited on page 33.)

[Cong 2004] G. Cong, A. K. H. Tung, X. Xu, F. Pan and J. Yang. *FARMER: Finding interesting rule groups in microarray datasets.* In Proceedings of the ACM International Conference on Management of Data (SIGMOD), 2004. (Cited on page 25.)

[Couchet 2007] J. Couchet, D. Manrique, J. Rios and A. Rodriguez-Paton. *Crossover and mutation operators for grammar-guided genetic programming.* Soft Computing, vol. 11, no. 10, pages 943–955, 2007. (Cited on page 53.)

[Creighton 2003] C. Creighton and S. Hanash. *Mining gene expression databases for association rules.* Bioinformatics, vol. 19, pages 79–86, 2003. (Cited on page 24.)

[Cunningham 1999] S. J. Cunningham and G. Holmes. *Developing Innovative Applications in Agriculture Using Data Mining*, 1999. (Cited on page 22.)

[Davey 1994] B. A. Davey and H. A. Priestley. Introduction to lattices and order. Cambridge University Press, 1994. (Cited on page 3.)

[Davis 1991] L. Davis. Handbook of genetic algorithms. Van Nostrand Reinhold, New York, 1991. (Cited on page 52.)

[Deepika 2011] N. Deepika, K. C. Shekar and D. Sujatha. *Association rule for classification of Heart-attack patients.* International Journal of Advanced Engineering Sciences and Technologies, vol. 11, no. 2, pages 253–257, 2011. (Cited on page 21.)

[Dehuri 2004] S. Dehuri and R. Mall. *Mining predictive and comprehensible rules using a multi-objective genetic algorithm.* Proceedings of the International Conference on Advanced Computing and Communication, 2004. (Cited on page 52.)

[Delgado 2009] G. Delgado, V. Aranda, J. Calero, M. S. Maranon, J. M. Serrano, D. Sanchez and M. A. Vila. *Using fuzzy data mining to evaluate survey data from olive grove cultivation.* Computers and Electronics in Agriculture, vol. 65, pages 99–113, 2009. (Cited on page 22.)

[Deng 2003] M. Deng, F. Sun and T. Chen. *Assessment of the reliability of protein-protein interactions and protein function prediction.* In Proceedings of the Pacific Symposium on Biocomputing (PSB), pages 140–151, 2003. (Cited on page 33.)

[Dhanapalan 2007] L. Dhanapalan and J. Y. Chen. *A case study of integrating protein interaction data using semantic web technology.* International Journal of Bioinformatics Research and Applications, vol. 3, no. 3, pages 286–302, 2007. (Cited on page 158.)

[Doddi 2001] S. Doddi, A. Marathe, S. S. Ravi and D. C. Torney. *Discovery of association rules in medical data.* Medical Informatics and the Internet in Medicine, vol. 26, pages 25–33, 2001. (Cited on page 21.)

[Eisen 1998] M. Eisen, P. Spellman, P. O. Brown and D. Botstein. *Cluster analysis and display of genome wide expression patterns.* Proc. Natl. Acad. Sci. USA, vol. 95, no. 25, pages 14863–14868, 1998. (Cited on pages 146, 171 and 177.)

[El-Houby 2010] E. M. F. El-Houby. *Mining protein structure class using one database scan.* International Journal of The Computer, Internet and Management, vol. 18, no. 2, pages 8–16, August 2010. (Cited on page 30.)

[Fhar 1993] V. Fhar and A. Tuzhilin. *Abstract-driven pattern discovery in databases.* IEEE Transactions on Knowledge and Data Engineering, vol. 5, pages 926–938, 1993. (Cited on page 42.)

[Freitas 2001] A. A. Freitas. *A survey of evolutionary algorithms for data mining and knowledge discovery.* Advances in Evolutionary Computation, pages 819–845, 2001. (Cited on page 53.)

[Fu 2009] W. Fu, B.E. Sanders-Beer, K.S. Katz, D.R. Maglott, K.D. Pruitt and R.G. Ptak. *Human immunodeficiency virus type 1, human protein interaction database at NCBI.* Nucleic Acids Research, vol. 37, pages 417–422, 2009. (Cited on pages 8 and 148.)

[Fung 2005] E. T. Fung, S. R. Weinberger, E. Gavin and F. Zhang. *Bioinformatics approaches in clinical proteomics.* Expert Rev. Proteomics, vol. 2, no. 6, pages 847–862, 2005. (Cited on page 33.)

[Fung 2012] K. Y. Fung, C. K. Kwong, K. W. M. Siu and K. M. Yu. *A multi-objective genetic algorithm approach to rule mining for affective product design.* Expert Systems with Applications, vol. 39, no. 8, pages 7411–7419, 2012. (Cited on page 53.)

[Gahery 2007] H. Gahery, S. Figueiredo, C. Texier, S. Pouvelle-Moratille, L. Ourth, C. Igea, M. Surenaud, J.G. Guillet and B. Maillere. *HLA-DR-restricted peptides identified in the Nef protein can induce HIV type 1-specific IL-2/IFN-gamma-secreting CD4+ and CD4+ /CD8+ T cells in humans after lipopeptide vaccination.* AIDS Res. and Hum. Retroviruses, vol. 3, no. 23, pages 427–437, 2007. (Cited on pages 166 and 167.)

[Gamberger 1999] D. Gamberger, N. Lavrac and V. Jovanoski. *High confidence association rules for medical diagnosis.* In Proceedings of IDAMAP99, pages 42–51, 1999. (Cited on page 21.)

[Ganascia 1991] J. G. Ganascia. *Deriving the learning bias from rule properties.* Machine Intelligence, vol. 12, pages 151–167, 1991. (Cited on page 42.)

[Ganter 1999] B. Ganter and R. Wille. Formal concept analysis: Mathematical foundations. Springer, 1999. (Cited on pages 3, 4, 6 and 8.)

[Ganter 2005] B. Ganter, G. Stumme and R. Wille. *Formal Concept Analysis: Foundations and applications.* Lecture Notes in Computer Science, vol. 36, no. 26, 2005. (Cited on page 47.)

[Garbis 2005] S. Garbis, G. Lubec and M. Fountoulakis. *Limitations of current proteomics technologies.* Journal of Chromatography A, vol. 1077, pages 1–18, May 2005. (Cited on page 28.)

[Gardarin 1998] G. Gardarin, P. Pucheral and F. Wu. *Bitmap based algorithms for mining association rules.* In Proceedings of the BDA conference, pages 157–176, 1998. (Cited on page 36.)

[Gauthaman 2008] A. Gauthaman. *Analysis of DNA microarray data using association rules: A selective study.* World Academy of Science, Engineering and Technology, vol. 42, pages 12–16, 2008. (Cited on pages 24 and 25.)

[Geng 2006] L. Geng and H. J. Hamilton. *Interestingness measure for data mining: A survey.* ACM Computing Survey, vol. 38, no. 3, 2006. (Cited on pages 38, 39, 40, 41 and 43.)

[Georgii 2005] E. Georgii, L. Richter, U. Ruckert and S. Kramer. *Analyzing microarray data using quantitative association rules.* Bioinformatics, pages ii123–ii129, 2005. (Cited on page 25.)

[Ghosh 2004] A. Ghosh and B. T. Nath. *Multi-objective rule mining using genetic algorithms.* Information Sciences, vol. 163, pages 123–133, 2004. (Cited on pages 42, 52 and 53.)

[Giannopoulou 2008] E. G. Giannopoulou and S. Kossida. Data mining applications in the post-genomic era, chapitre 14, pages 237–252. Data Mining in Medical and Biological Research, InTech, 2008. (Cited on page 28.)

[GO 2000] The Gene Ontology Consortium GO. *Gene Ontology: Tool for the unification of biology.* Nat. Genet., vol. 25, no. 1, pages 25–9, May 2000. http://www.geneontology.org. (Cited on page 14.)

[Gonzales 2009] E. Gonzales, K. Taboada, S. Mabu, K. Shimada and K. Hirasawa. *Combination of two evolutionary methods for mining association rules in large and dense databases.* Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 13, no. 5, pages 561–569, 2009. (Cited on pages 52 and 53.)

[Goodman 1968] L. A. Goodman and W. H. Kruskal. *Measures of association for cross-classifications.* Journal of the American Statistical Association, vol. 49, pages 732–764, 1968. (Cited on page 42.)

[Gopalakrishnan 2006] V. Gopalakrishnan, P. Ganchev, S. Ranganathan and R. Bowser. *Rule Learning for Diseas-Specific Biomarker Discovery from Clinical Proteomic Mass Spectra.* In J. Li et. al., editeur, LNBI 3916, pages 93–105, Berlin, 2006. BioDM 2006, Springer-Verlag. (Cited on page 34.)

[Gouda 2001] K. Gouda and M. J. Zaki. *Efficiently mining maximal frequent itemsets.* In Proceedings of the IEEE ICDM conference, pages 163–170, 2001. (Cited on page 36.)

[Gough 2008] D. J. Gough, D. E. Levy, R. W. Johnstone and C. J. Clarke. *IFNgamma signaling-does it mean JAK-STAT?* Cytokine & Growth Factor Reviews, vol. 19, no. 5-6, pages 383–394, October 2008. (Cited on page 167.)

[Gras 1979] R. Gras. *Contribution de l'étude expérimentale et de l'analyse de certaines acquisitions cognitives et de certains objectifs didactiques en mathématiques.* PhD thesis, Université de Rennes I, 1979. (Cited on page 42.)

[Gras 2001] R. Gras, P. Kuntz, R. Couturier and F. Guillet. *Une version entropique de l'intensité d'implication pour les corpus volumineux.* In Actes de la Conférence Internationale sur l'Extraction et la Gestion des Connaissances (EGC), pages 69–80, 2001. (Cited on page 42.)

[Gray 1998] B. Gray and M. E. Orlowska. *CCAIIA: Clustering categorical attributes into interesting association rules.* In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pages 132–143, 1998. (Cited on page 42.)

[Grigoriev 2001] A. Grigoriev. *A relationship between gene expression and protein interactions on the proteome scale: Analysis of the bacteriophage T7 and the yeas Saccharomyces Cerevisiae.* Nucleic Acid Research, vol. 29, pages 3513–3519, July 2001. (Cited on page 26.)

[Gupta 2005] A. Gupta, N. Kumar and V. Bhatnagar. *Analysis of Medical Data using Data Mining and Formal Concept Analysis.* World Academy of Science, Engineering and Technology, pages 61–64, June 2005. (Cited on page 19.)

[Gupta 2006] N. Gupta, N. Mangal, K. Tiwari and P. Mitra. *Mining quantitative association rules in protein sequences.* In Proceedings of Australasian Conference on Knowledge Discovery and Data Mining (AUSDM), pages 273–281, 2006. (Cited on page 29.)

[Gygi 1999] S.P. Gygi, Y. Rochon, B.R. Franza and R. Aebersold. *Correlation between protein and mRNA abundance in Yeast.* Molicular Cell Biology, vol. 19, pages 1720–1730, 1999. (Cited on page 26.)

[Hamrouni 2008a] T. Hamrouni, S. Ben Yahia and E. Mephu Nguifo. *Succinct System of Minimal Generators: A Thorough Study, Limitations and New Definitions.* In Concept Lattices and Their Applications, volume 4923 of *Lecture Notes in Computer Science*, pages 80–95. Springer, 2008. (Cited on page 9.)

[Hamrouni 2008b] T. Hamrouni, S. Ben Yahia and E. M. Nguifo. *Succinct system of minimal generators: Theoretical Foundations and Applications.* International Journal of Foundations of Computer Science, vol. 19, no. 2, pages 271–296, 2008. (Cited on page 133.)

[Han 2000a] J. Han and J. Pei. *Mining frequent patterns by pattern-growth: Methodology and implications.* SIGKDD Explor. Newsl., vol. 2, no. 2, pages 14–20, 2000. (Cited on page 3.)

[Han 2000b] J. Han, J. Pei and Y. Yin. *Mining frequent patterns without candidate generation.* In Proceedings of ACM International Conference on Management of Data (SIGMOD), pages 1–12, 2000. (Cited on page 46.)

[Han 2009] F. Han and N. Rao. *Mining co-regulated genes using association rules combined with hast-tree and genetic algorithms.* IEEE Xplore, pages 858–862, 2009. (Cited on page 25.)

[Han 2011] J. Han, M. Kamber and J. Pei. Data mining: Concepts and techniques. Morgan Kaufmann Series in Data Management Systems. IEEE Computer Society, 3rd édition, 2011. (Cited on page 1.)

[He 2005] Z. He, X. Xu and S. Deng. *Data mining for actionable knowledge: A survey.* CoRR, vol. abs/cs/0501079, 2005. (Cited on page 43.)

[He 2009] Y. He and S. C. Hui. *Exploring ant-based algorithms for gene expression data analysis.* Artificial Intelligence in Medicine, vol. 47, no. 2, pages 105–119, 2009. (Cited on page 25.)

[Hetland 2002] M. L. Hetland and P. Saetrom. *Temporal rule discovery using genetic programming and specialized hardware.* In Proceedings of the International Conference on Recent Advances in Soft Computing (RASC), pages 182–188, 2002. (Cited on page 53.)

[Hipp 2000] J. Hipp, U. Guntzer and G. Nakhaeizadeh. *Algorithms for association rule mining - a general survey and comparison.* SIGKDD Explor. Newsl., vol. 2, no. 1, pages 58–64, 2000. (Cited on page 46.)

[Hirji 2001] Karim K. Hirji. *Exploring data mining implementation.* Communication of the ACM, vol. 44, no. 7, pages 87–93, July 2001. (Cited on page 12.)

[Hong 2008] T.-P. Hong and Y.-C. Lee. *An Overview of Mining Fuzzy Association Rules.* In H. Bustince, F. Herrera and J. Montero, editeurs, Fuzzy Sets and Their Extensions: Representation, Aggregation and Models, volume 220 of *Studies in Fuzziness and Soft Computing*, pages 397–410. Springer, 2008. (Cited on pages 172 and 178.)

[Hu 2010] X. Hu, X. Zhang, L. Yoo, X. Wang and J. Feng. *Mining hidden connections among biomedical concepts from disjoint biomedical literature sets through semantic-based association rule.* Int. J. Intell. Syst., vol. 25, no. 2, pages 207–223, February 2010. (Cited on page 19.)

[Huang 2007] Z. Huang, J. Li, H. Su, G. Watts and H. Chen. *Large-scale regulatory network analysis from microarray data: Modified bayesian network learning and association rule mining.* Decision Support System, vol. 43, no. 4, pages 1207–1225, 2007. (Cited on page 24.)

[Icev 2003] A. Icev, C. Ruiz and E. F. Ryder. *Distance-based association rule mining.* In Proceedings of the ACM SIGKDD Workshop on Data Mining in Bioinformatics (BIOKDD), pages 34–40, 2003. (Cited on page 25.)

[Ishibuchi 1997] H. Ishibuchi, T. Murata and I. B. Turksen. *Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems.* Fuzzy Sets and Systems, vol. 89, pages 135–150, 1997. (Cited on page 53.)

[Ishibuchi 2004] H. Ishibuchi and T. Yamamoto. *Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining.* Fuzzy Sets and Systems, vol. 141, pages 59–88, 2004. (Cited on page 53.)

[Ishibuchi 2005] H. Ishibuchi and Y. Nojima. *Accuracy-complexity tradeoff analysis by multiobjective rule selection.* In Proceedings of the ICDM Workshop on Computational Intelligence in Data Mining (CIDM), pages 39–48, 2005. (Cited on page 52.)

[Ishibuchi 2006] H. Ishibuchi, I. Kuwajima and Y. Nojima. *Multi-objective association rule mining.* In Proceedings of the PPSN Workshop on Multiobjective Problem Solving from Nature, 2006. (Cited on page 53.)

[Jager 2011] Stefanie Jager, Peter Cimermancic, Natali Gulbahce, Jeffrey R. Johnson, Kathryn E. McGovern, Starlynn C. Clarke, Michael Shales, Gaelle Mercenne, Lars Pache, Kathy Li, Hilda Hernandez, Gwendolyn M. Jang, Shoshannah L. Roth, Eyal Akiva, John Marlett, Melanie Stephens, Ivan D'Orso, Jason Fernandes, Marie Fahey, Cathal Mahon, Anthony J. O'Donoghue, Aleksandar Todorovic, John H. Morris, David A. Maltby, Tom Alber, Gerard Cagney, Frederic D. Bushman, John A. Young, Sumit K. Chanda, Wesley I. Sundquist, Tanja Kortemme, Ryan D. Hernandez, Charles S. Craik, Alma Burlingame, Andrej Sali, Alan D. Frankel and Nevan J. Krogan. *Global landscape of HIV-human protein complexes.* Nature, 2011. (Cited on page 148.)

[Jansen 2003] R. Jansen, D. Greenbaum H. Yu, Y. Kluger, N. J. Krogan, S. Chung, A. Emili, M. Snyder, J. F. Greenblatt and M. Gerstein. *A Bayesian networks approach for predicting protein-protein interactions from genomic data.* Science, vol. 302, pages 449–453, 2003. (Cited on pages 5 and 158.)

[Jefreys 1935] H. Jefreys. *Some test of significance treated by theory of probability.* In Proceedings of the Cambridge Philosophical Society, pages 203–222, 1935. (Cited on page 42.)

[Ji 2011] Y. Ji, H. Ying, P. Dews, A. Mansour, J. Tran, R.E. Miller and R. M. Massanari. *A Potential Causal Association Mining Algorithm for Screening Adverse Drug Reactions in Postmarketing Surveillance.* IEEE Transactions on Information Technology in Biomedicine, vol. 15, no. 3, pages 428–437, May 2011. (Cited on page 21.)

[Jimenez 2010] A. Jimenez, F. Berzal and J.-C. Cubero. *Interestingness measure for association rules within groups.* Knowledge and Information System, vol. 80, pages 298–307, 2010. (Cited on page 43.)

[Jin 2006] H. Jin, J. Chen, C. Kelman, H. He, D. McAullay and C. M. O'Keefe. *Mining Unexpected Associations for Signalling Potential Adverse Drug Reactions from Administrative Health Databases.* In PAKDD 2006, pages 867–876. LNAI, Springer-Verlag, 2006. (Cited on page 20.)

[Jin 2008] H. Jin, J. Chen, H. He, G. Williams, C. Kelman and C. O'keefe. *Mining unexpected temporal associations: Applications in detecting adverse drug reactions.* IEEE Trans. Inf. Technol. Biomed., vol. 12, no. 4, pages 488–500, 2008. (Cited on page 20.)

[Joshi-Tope 2003] G. Joshi-Tope, I. Vastrik, G. Gopinathrao, L. Matthews, E. Schmidt, M. Gillespie, P. D'Eustachio, B. Jassal, S. Lewis, G. Wu, E. Birney and L. Stein. *The Genome Knowledgebase: A Resource for Biologists and Bioinformaticists.* In Cold Spring Harb Symp Quant Biol, volume 68, pages 237–43, 2003. (Cited on page 14.)

[Kam 2003] H. J. Kam, D. Lee and K. H. Lee. *Mining and Interpretation of Association Rules among Protein Sequence Motifs.* In Proceedings of the 25th Annual International Conference of the IEEE EMBS, pages 3551–3554, 2003. (Cited on page 29.)

[Kamber 1996] M. Kamber and R. Shinghal. *Evaluating the interestingness of characteristic rules.* In Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD), pages 263–266, 1996. (Cited on page 40.)

[Karel 2007] F. Karel and J. Klema. *Quantitative association rule mining in genomics using apriori knowledge.* In Proceedings of the ECML/PKDD Workshop on Prior Conceptual Knowledge in Machine Learning and Data Mining, pages 53–64, September 2007. (Cited on page 25.)

[Kaya 2006] M. Kaya. *Multi-objective genetic algorithm based approaches for mining optimized fuzzy association rules.* Soft Computing, vol. 10, pages 578–586, 2006. (Cited on page 52.)

[Kell 2004] D. B. Kell and S. G. Oliver. *Here is the evidence, now what is the hypothesis? The complementary roles of inductive and hypothesis-driven science in the post-genomic era.* BioEssays, vol. 26, no. 1, pages 99–105, December 2004. (Cited on page 18.)

[Kloetzel 2004] P. M. Kloetzel. *The proteasome and MHC class I antigen processing.* Biochimica et Biophysica Acta, vol. 1695, pages 217–225, 2004. (Cited on page 31.)

[Klosgen 1996] W. Klosgen. *Explora: A multipattern and multistrategy discovery assistant.* Advances in Knowledge Discovery and Data Mining, pages 249–271, 1996. (Cited on page 42.)

[Kohn 2007] E. C. Kohn, N. Azad, C. Annunziata, A. S. Dhamoon and G. Whiteley. *Proteomics as tools for biomarker discovery.* Dis. Markers, vol. 23, pages 411–417, 2007. (Cited on page 33.)

[Konig 2008] R. Konig, Y. Zhou, D. Elleder, T. L. Diamond, G.M.C. Bonamy, J. T. Irelan, C. Chiang, B. P. Tu, P.D. De Jesus, C.E. Lilley, S. Seidel, A. M. Opaluch, J. S. Caldwell, M. D. Weitzman, K. L. Kuhen, S. Bandyopadhyay, T. Ideker, A. P. Orth, L. J. Miraglia, F. D. Bushman, J. A. Young and S. K. Chanda. *Global analysis of host-pathogen interactions that regulate early-stage HIV-1 replication.* Cell, vol. 135, pages 49–60, 2008. (Cited on page 165.)

[Kotlyar 2006] M. Kotlyar and I. Jurisica. *Predicting protein-protein interactions by association mining.* Information Systems Frontiers, vol. 8, no. 1, pages 37–47, 2006. (Cited on page 33.)

[Krawczak 2000] M. Krawczak, E. V. Ball, I. Fenton, P. D. Stenson, S. Abeysinghe, N. Thomas and D. N. Cooper. *Human gene mutation database - A biomedical information and research resource.* Human Mutation, vol. 15, no. 1, pages 45–51, 2000. (Cited on pages 23 and 25.)

[Kusiak 2005] A. Kusiak, B. Dixon and S. Shah. *Predicting survival time for kidney dialysis patients: a data mining approach.* Computers in Biology and Medicine, vol. 35, pages 311–327, 2005. (Cited on page 22.)

[Kwasnicka 2005] H. Kwasnicka and K. Switalski. *Discovery of association rules from medical data - Classical and evolutionary approaches.* In Proceedings of the Meeting of Polish Information Processing Society (PIPS), pages 163–177, 2005. (Cited on page 19.)

[Lallich 2007] S. Lallich, O. Teytaud and E. Prudhomme. *Association rule interestingness: Measure and statistical validation.* Quality Measures in Data Mining, pages 251–275, 2007. (Cited on page 42.)

[Latterich 2008] M. Latterich, M. Abramovitz and B. L. Jones. *Proteomics: New technologies and clinical applications.* European Journal of Cancer, vol. 44, pages 2737–2741, 2008. (Cited on page 33.)

[Laxmi 2011] P. Laxmi, A. Poongodai and D. Sujatha. *Extended Apriori for association rule mining: Diminution based utility weightage measuring approach.* Journal of Computer Science and Technology, vol. 11, no. 22, pages 25–30, 2011. (Cited on page 41.)

[Lee 2006] H. Lee, M. Deng, F. Sun and T. Chen. *An integrated approach to the prediction of domain-domain interactions.* BMC Bioinformatics, vol. 7, no. 269, 2006. (Cited on page 158.)

[Lenca 2004] P. Lenca, P. Meyer, B. Vaillant and S. Lallich. *A multicriteria decision aid for interestingness measure selection.* Technical Report LUSSI-TR-2004-01-EN, GET/ENST, Bretagne, France, 2004. (Cited on page 40.)

[Lenca 2007] P. Lenca, B. Vaillant, P. Meyer and S. Lallich. *Association rule interestingness measures: Experimental and theoretical studies.* In Quality Measures in Data Mining, volume 43 of *Studies in Computational Intelligence*, pages 51–76. Springer, 2007. (Cited on page 41.)

[Lenca 2008] P. Lenca, P. Meyer, B. Vaillant and S. Lallich. *On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid.* European Journal of Operational Research, vol. 184, no. 2, pages 610–626, 2008. (Cited on page 41.)

[Lerman 1981] I. C. Lerman, R. Gras and H. Rostam. *Elaboration d'un indice d'implication pour les donnees binaires, i et ii.* Mathématiques et Sciences Humaines, vol. 74, no. 75, 1981. (Cited on page 42.)

[Lerman 2003] I. Lerman and J. Aze. *Une mesure probabiliste contextuelle discriminante de qualite des règles d'association.* In Actes de la Conférence Internationale sur l'Extraction et la Gestion des Connaissances (EGC), pages 247–262, 2003. (Cited on page 42.)

[Leung 2010] K.-S. Leung, K.-C. Wong, T.-M. Chan, M.-H. Wong, K.-H. Lee, C.-K. Lau and S. K. W. Tsui. *Discovering protein-DNA binding sequence patterns using association rule mining.* Nucleic Acids Research, vol. 38, no. 19, pages 6324–6337, June 2010. (Cited on page 31.)

[Lewis 2000] P. D. Lewis, J. S. Harvey, E. M. Waters and J. M. Parry. *The mammalian gene mutation database.* Mutagenesis, vol. 15, no. 5, pages 411–414, 2000. (Cited on pages 23 and 25.)

[Li 2010] J. Li. *A survey on actionable knowledge discovery applications.* In Proceedings of the International Workshop on Intelligent Systems and Applications (ISA), pages 1–3, 2010. (Cited on page 43.)

[Lieber 2008] J. Lieber, A. Napoli, L. Szathmary and Y. Toussaint. *First Elements on Knowledge Discovery Guided by Domain Knowledge (KDDK).* In Concept Lattices and Their Applications, volume 4923 of *Lecture Notes in Computer Science*, pages 22–41. Springer, 2008. (Cited on pages 174 and 180.)

[Liebler 2002] D. C. Liebler. Introduction to proteomics. Humana Press, 2002. (Cited on page 26.)

[Lin 1998] D.-I. Lin and Z. M. Kedem. *PincerSearch: A new algorithm for discovering the maximum frequent set.* In Proceedings of the EDBT conference, pages 105–119, 1998. (Cited on page 46.)

[Lin 2004] N. Lin, B. Wu, R. Jansen, M. Gerstein and H. Zhao. *Information assessment on predicting protein-protein interactions.* BMC Bioinformatics, vol. 5, no. 154, 2004. (Cited on pages 5 and 158.)

[Lippolis 2010] J. D. Lippolis and T. A. Reinhardt. *Utility, limitations, and promise of proteomics in animal science.* Veterinary Immunology and Immunopathology, vol. 138, pages 241–251, 2010. (Cited on page 28.)

[Liu 2000] B. Liu, W. Hsu, S. Chen and Y. Ma. *Analyzing subjective interestingness of association rules.* IEEE Intelligent Systems, vol. 15, no. 5, pages 47–55, 2000. (Cited on page 43.)

[Liu 2011] Y. Liu, Z. Guo, X. Ke and O. R. Zaiane. *Protein sub-cellular localization prediction with associative classification and multi-class SVM.* In Proceedings of the ACM International Conference On Bioinformatics and Computational Biology (ACM-BCB), pages 493–495, August 2011. (Cited on page 33.)

[Loevinger 1947] J. Loevinger. *A systemic approach to the construction and evaluation of tests of ability.* Psychological Monographs, vol. 61, no. 4, 1947. (Cited on page 42.)

[Lopez 2007] E. J. Lopez, A. Blanco, F. Garcia and A. Marin. *Extracting biological knowledge by fuzzy association rule mining.* In Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pages 1–6, 2007. (Cited on page 25.)

[Lopez 2012] F. J. Lopez, M. Cuadros, C. Cano, A. Concha and A. Blanco. *Biomedical application of fuzzy association rules for identifying breast cancer biomarkers.* Med. Biol. Eng. Comput., vol. 50, no. 5, 2012. (Cited on page 34.)

[Lu 2004] Z. Lu, D. Szafron, R. Greiner, P. Lu, D. S. Wishart, B. Poulin, C. Macdonell and R. Eisner. *Predicting subcellular localization of proteins using machine-learned classifiers.* Bioinformatics, vol. 20, no. 4, pages 547–556, 2004. (Cited on page 33.)

[Luna 2010] J. M. Luna, J. Raul Romero and S. Ventura. *G3PARM: A grammar guided genetic programming algorithm for mining association rules.* Computational Intelligence, pages 18–23, 2010. (Cited on page 53.)

[Luscombe 2000] N. M. Luscombe, S. E. Austin, H. M. Berman and J. M. Thornton. *An overview of the structures of protein-DNA complexes.* Genome Biology, vol. 1, 2000. (Cited on page 31.)

[Luscombe 2002] N. M. Luscombe and J. M. Thornton. *Protein-DNA interactions: Amino acid conservation and the effects of mutations on binding specificity.* Journal for Molecular Biology, vol. 320, pages 991–1009, 2002. (Cited on page 31.)

[MacPherson 2010] Jamie I. MacPherson, Jonathan E. Dickerson, John W. Pinney and David L. Robertson. *Patterns of HIV-1 protein interaction identify perturbed host-cellular subsystems.* PLoS Comput. Biol., vol. 6, no. 7, page e1000863, July 2010. (Cited on page 165.)

[Madeira 2004] S.C. Madeira and A.L. Oliveira. *Biclustering algorithms for biological data analysis: A survey.* IEEE/ACM Trans. Comput. Biol. Bioinformatics, vol. 1, pages 24–45, 2004. (Cited on pages 2 and 12.)

[Madeira 2009] S.C. Madeira and A.L. Oliveira. *A polynomial time biclustering algorithm for finding approximate expression patterns in gene expression time series.* Algorithms for Molecular Biology, vol. 4, no. 8, 2009. (Cited on pages 4, 10, 11, 173 and 179.)

[Mannila 1996] H. Mannila and H. Toivonen. *Multiple uses of frequent sets and condensed representations.* In Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (KDD), pages 189–194, 1996. (Cited on pages 45 and 48.)

[Mannila 1997] H. Mannila and H. Toivonen. *Levelwise Search and Borders of Theories in Knowledge Discovery.* Data Mining and Knowledge Discovery, vol. 1, pages 241–258, 1997. (Cited on page 48.)

[Marcotte 1999] E. M. Marcotte, M. Pellegrini, H. L. Ng, D. W. Rice, T. O. Yeates and D. Eisenberg. *Detecting protein function and protein-protein interactions from genome sequences.* Science, vol. 285, pages 751–753, 1999. (Cited on pages 26 and 33.)

[Martin 2005] S. Martin, D. Roe and J.-L. Faulon. *Predicting protein-protein interactions using signature products.* Bioinformatics, vol. 21, no. 2, pages 218–226, 2005. (Cited on page 158.)

[Martinez 2007] Ricardo Martinez, Nicolas Pasquier and Claude Pasquier. *GenMiner: Mining informative association rules from genomic data.* In IEEE BIBM International Conference on Bioinformatics and Biomedicine (IEEE BIBM'2007), pages 15–22. IEEE Computer Science, November 2007. (Cited on page 147.)

[Martinez 2009] Ricardo Martinez, Nicolas Pasquier and Claude Pasquier. *Mining association rule bases from integrated genomic data and annotations.* Lecture Notes in Bioinformatics (LNBI), vol. 5488, pages 78–90, April 2009. (Cited on page 147.)

[Matthews 2007] L. Matthews, P. D'Eustachio, M. Gillespie, D. Croft, B. de Bono, G. Gopinath, B. Jassal, S. Lewis, E. Schmidt, I. Vastrik, G. Wu, E. Birney and L. Stein. *An Introduction to the Reactome Knowledgebase of Human Biological Pathways and Processes.* Bioinformatics Primer, 2007. (Cited on page 14.)

[Matthews 2008] L. Matthews, G. Gopinath, M. Gillespie, M. Caudy, D. Croft, B. de Bono, P. Garapati, J. Hemish, H. Hermjakob, B. Jassal, A. Kanapin, S. Lewis, S. Mahajan, B. May, E. Schmidt, I. Vastrik, G. Wu, E. Birney, L. Stein and P. D'Eustachio P. *Reactome knowledgebase of biological pathways and processes.* Nucleic Acids Res., vol. 3, November 2008. (Cited on page 14.)

[McGarry 2005] K. McGarry. *A survey of interestingness measure for knowledge discovery.* Knowledge Engineering Review, vol. 20, no. 1, pages 39–61, 2005. (Cited on pages 39 and 41.)

[Mendes 2001] R. R.F. Mendes, F. de B. Voznika, A. A. Freitas and J. C. Nievola. *Discovering fuzzy classification rules with genetic programming and co-evolution.* In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), pages 314–325, 2001. (Cited on page 53.)

[Merceron 2007] A. Merceron and K. Yacef. *Revisiting interestingness of strong symmetric association rules in educational data.* In Proceedings of the International Workshop on Applying Data Mining in e-Learning (ADML), 2007. (Cited on page 39.)

[Mikos 2005] T. Mikos, K. Pantazis, D. G. Goulis, N. Maglaveras, J. N. Bontis and J. Papadimas. *The use of Data Mining in the categorisation of patients with Azoospermia.* Hormones, vol. 4, no. 4, pages 214–218, 2005. (Cited on page 22.)

[Milledge 2004] T. Milledge, G. Zheng and G. Narasimhan. *An application of association rule mining to HLA-A*0201 epitope prediction.* In Proceedings of international conference on biometric authentication (ICBA), 2004. (Cited on page 32.)

[Mondal 2012a] K. C. Mondal, N. Pasquier, A. Mukhopadhyay, C. da Costa Pereira, U. Maulik and A. G. B. Tettamanzi. *Prediction of protein interactions on HIV-1-human PPI data using a novel closure-based integrated approach.* In Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms (BIOINFORMATICS), pages 164–173, 2012. (Cited on pages 33 and 165.)

[Mondal 2012b] K. C. Mondal, N. Pasquier, A. Mukhopadhyay, U. Maulik and S. Bandyopadhyay. *A new approach for association rule mining and bi-clustering using Formal Concept Analysis.* In International Conference on Machine Learning and Data Mining (MLDM), volume 7376 of *LNAI*, pages 86–101, Berlin, Germany, July 2012. Springer. (Cited on page 151.)

[Mondal 2013] K. C. Mondal and N. Pasquier. *Galois closure based association rule mining from biological data* in Biological Knowledge Discovery Handbook: Preprocessing, Mining and Postprocessing of Biological Data, chapitre 35. Wiley-Blackwell, John Wiley & Sons, New Jersey, USA, to appear in August 2013. (Cited on page 18.)

[Monteoliva 2004] L. Monteoliva and J. P. Albar. *Differential proteomics: An overview of gel and non-gel based approaches.* Briefings in Functional Genomics and Proteomics, vol. 3, pages 220–239, 2004. (Cited on page 32.)

[Mosteller 1968] F. Mosteller. *Association and estimation in contingency tables.* Journal of the American Statistical Association, vol. 63, pages 1–28, 1968. (Cited on page 42.)

[Mukhopadhyay 2010] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay and R. Eils. *Mining association rules from HIV-human protein interactions.* In Proceedings of International Conference on Systems in Medicine and Biology (ICSMB), pages 344–348, December 2010. (Cited on pages 5, 13 and 33.)

[Nabieva 2008] E. Nabieva and M. Singh. *Protein function prediction via analysis of interactomes.* In Prediction of Protein Structures, Functions, and Interactions, chapitre 10. John Wiley & Sons, 2008. (Cited on page 33.)

[Nafar 2006] Z. Nafar and A. Golshani. *Data mining methods for protein-protein interactions.* Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), pages 991–994, 2006. (Cited on page 33.)

[Nahar 2008] J. Nahar. *Significant cancer risk factor extraction: An association rule discovery approach.* In Proceedings of the IEEE International Conference on Computer and Information Technology (ICCIT), pages 108–114, 2008. (Cited on pages 18 and 21.)

[Nahar 2011] J. Nahar, K.S. Tickle, A. B. M. S. Ali and Y.P. P. Chen. *Significant Cancer Prevention Factor Extraction: An Association Rule Discovery Approach.* Journal of Medical System, vol. 35, pages 353–367, 2011. (Cited on page 21.)

[Naidu 2012] P. M. Naidu and C. Rajendra. *Detection of Health Care using Data Mining Concepts Through Web.* International Journal of Advanced Research in Computer Engineering and Technology, vol. 1, no. 4, pages 45–50, 2012. (Cited on page 21.)

[Natarajan 2005] R. Natarajan and B. Shekar. *Interestingness of association rules in data mining: Issues relevant to e-commerce.* SADHANA Academy Proceedings in Engineering Sciences, vol. 30, pages 2991–309, 2005. (Cited on page 41.)

[Ohsaki 2004] M. Ohsaki, S. Kitaguchi, K. Okamoto, H. Yokoi and T. Yamaguchi. *Evaluation of rule interestingness measures with a clinical dataset on hepatitis.* In Proceedings of the International Conference on Principles of Knowledge Discovery in Databases (PKDD), pages 362–373, 2004. (Cited on page 41.)

[Omiecinski 2003] E. R. Omiecinski. *Alternative interest measures for mining associations in databases.* IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 1, pages 57–69, 2003. (Cited on page 42.)

[Ordonez 2000] C. Ordonez, C. Santana and L. de Braal. *Discovering interesting association rules in medical data.* In Proceedings of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD), pages 78–85, 2000. (Cited on page 21.)

[Ordonez 2001] C. Ordonez, E. Omiecinski, L. Braal, C. A. Santana, N. Ezquerra, J. A. Taboada, D. Cooke, E. Krawczynska and E. V. Garcia. *Mining Constrained Association Rules to Predict Heart Disease.* In Proceedings of the 2001 IEEE International Conference on Data Mining, ICDM '01, pages 433–440, Washington, DC, USA, 2001. IEEE Computer Society. (Cited on page 21.)

[Ordonez 2006a] C. Ordonez. *Comparing Association Rules and Decision Trees for Disease Prediction.* In HIKM'2006, pages 17–24. ACM, 2006. (Cited on page 21.)

[Ordonez 2006b] C. Ordonez, N. Ezquerra and C. A. Santana. *Constraining and summarizing association rules in medical data.* Knowledge and Information Systems, vol. 9, no. 3, pages 259–283, 2006. (Cited on page 21.)

[Osl 2008] M. Osl, S. Dreiseitl, B. Pfeifer, K. Weinberger, H. Klocker, G. Bartsch, G. Schafer, B. Tilg, A. Graber and C. Baumgartner. *A new rule-based algorithm for identifying metabolic markers in prostate cancer using tandem mass spectrometry.* Bioinformatics, vol. 24, no. 24, pages 2908–2914, 2008. (Cited on page 34.)

[Oyama 2002] T. Oyama, K. Kitano, K. Satou and T. Ito. *Extraction of knowledge on protein-protein interaction by association rule discovery.* Bioinformatics, vol. 18, no. 5, pages 705–714, 2002. (Cited on pages 32 and 33.)

[Ozbek 2005] Ahmet Caghan Ozbek. *A position sensitive association rule mining for mhc/peptide binding problem.* PhD thesis, Istanbul Bilgi University, June 2005. (Cited on page 32.)

[Pagallo 1990] G. Pagallo and D. Haussler. *Boolean feature discovery in empirical leaning.* Machine Learning, vol. 5, no. 1, pages 71–99, 1990. (Cited on page 42.)

[Pan 2003] F. Pan, G. Cong and A. K. H. Tung. *CARPENTER: Finding closed patterns in long biological datasets.* In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 637–642. ACM, 2003. (Cited on page 25.)

[Pandey 2007] G. Pandey, M. Steinbach, R. Gupta, T. Garg and V. Kumar. *Association analysis-based transformations for protein interaction networks: A function prediction case study.* In Proceedings ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 540–549, 2007. (Cited on page 33.)

[Pasquier 1998] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal. *Pruning closed itemset lattices for associations rules.* In Actes des 14ème Journées Bases de Données Avancées, pages 177–196, october 1998. (Cited on pages 3 and 47.)

[Pasquier 1999a] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal. *Discovering frequent closed itemsets for association rules.* In Proceedings of the International Conference on Database Theory (ICDT), pages 398–416, 1999. (Cited on page 46.)

[Pasquier 1999b] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal. *Efficient mining of association rules using closed itemset lattices.* Information Systems, vol. 24, no. 1, pages 25–46, 1999. (Cited on pages 3, 6, 8, 9, 46, 47 and 162.)

[Pasquier 2005] N. Pasquier, R. Taouil, Y. Bastide, G. Stumme and L. Lakhal. *Generating a condensed representation for association rules.* Journal of Intelligent Information Systems, vol. 24, no. 1, pages 29–60, 2005. (Cited on pages 8 and 10.)

[Patil 2009] S. B. Patil and Y. S. Kumaraswamy. *Intelligent and Effective Heart Attack Prediction System using Data Mining and Artificial Neural Network.* European Journal of Scientific Research, vol. 31, no. 4, pages 642–656, 2009. (Cited on page 21.)

[Pavlov 2000] D. Pavlov, H. Mannila and P. Smyth. *Probabilistic models for query approximation with large data sets.* Technical Report 2000-07, University of California, Department of Information and Computer Science, 2000. (Cited on page 49.)

[Peeters 2003] R. Peeters. *The maximum edge biclique problem is NP-complete.* Discrete Applied Mathematics, vol. 131, no. 3, pages 651–654, 2003. (Cited on page 4.)

[Pei 2002] J. Pei and J. Han. *Constrained frequent pattern mining: A pattern-growth view.* ACM SIGKDD Explorations, vol. 4, no. 1, pages 31–39, 2002. (Cited on page 47.)

[Petricoin 2003] E. F. Petricoin and L. A. Liotta. *Clinical Application of Proteomic.* The Journal of Nutrition, vol. 133, pages 2476S–2484S, 2003. (Cited on page 33.)

[Piatetsky-Shapiro 1991a] G. Piatetsky-Shapiro. *Discovery, analysis and presentation of strong rules.* Knowledge Discovery in Databases, pages 229–248, 1991. (Cited on pages 40 and 42.)

[Piatetsky-Shapiro 1991b] G. Piatetsky-Shapiro and W. J. Frawley. Knowledge discovery in databases. AAAI Press/MIT Press, 1991. (Cited on page 1.)

[Pinney 2009] J. W. Pinney, J. E. Dickerson, W. Fu, B. E. Sanders-Beer, R. G. Ptak and D. L. Robertson. *HIV-host interactions: A map of viral perturbation of the host system.* AIDS, vol. 23, no. 5, pages 549–554, 2009. (Cited on page 148.)

[Pramod 2010] S. Pramod and O. P. Vyas. *Survey on frequent item set mining algorithms.* International Journal of Computer Applications, pages 86–91, 2010. (Cited on page 45.)

[Ptak 2008] R.G. Ptak, W. Fu, B.E. Sanders-Beer, J.E. Dickerson, J.W. Pinney, D.L. Robertson, M.N. Rozanov, K.S. Katz, D.R. Maglott, K.D. Pruitt and C.W. Dieffenbach. *Cataloguing the HIV-1 Human protein interaction network.* AIDS Research and Human Retroviruses, vol. 4, no. 12, pages 1497–1502, 2008. (Cited on pages 8, 148 and 149.)

[PubMed] NCBI PubMed. *Publications from medicine, biology and life science journals.* National Center for Biotechnology Information, U.S. National Library of Medicine. http://www.ncbi.nlm.nih.gov/pubmed. (Cited on page 14.)

[Qi 2005] Y. Qi, J. Klein-Seetharaman and Z. Bar-Josep. *Random forest similarity for protein-protein interaction prediction from multiple sources.* In Proceedings of the Pacific Symposium on Biocomputing (PSB), pages 531–542, 2005. (Cited on page 158.)

[Qi 2007] Y. Qi, J. Klein-Seetharaman and Z. Bar-Joseph. *A mixture of feature experts approach for protein-protein interaction prediction.* BMC Bioinform., vol. 8, 2007. (Cited on pages 5 and 158.)

[Qiu 2008] J. Qiu and W. S. Noble. *Predicting co-complexed protein pairs from heterogeneous data.* PLoS Computational Biology, vol. 4, no. 4, page e1000054, 2008. (Cited on page 158.)

[Rajak 2008] A. Rajak and M. K. Gupta. *Association Rule Mining: Applications in Various Areas.* In Proceedings of International Conference on Data Management, pages 3–7, India, 2008. (Cited on page 21.)

[Ramaraj 2008] E. Ramaraj and N. Venkatesan. *Positive and Negative Association Rule Analysis in Health Care Database.* International Journal of Computer Science and Network Security, vol. 8, no. 10, pages 325–330, 2008. (Cited on page 20.)

[Ras 2010] Z. W. Ras and L.-S. Tsay, editeurs. Advances in intelligent information systems, volume 265 of *Studies in Computational Intelligence.* Springer, 2010. (Cited on pages 173 and 179.)

[Rattanakronkul 2002] N. Rattanakronkul, J. Yuvaniyama and K. Waiyamai. *Combining association rule discovery and data classification for protein structure prediction.* In Proceedings of the International Conference on Bio-informatics (INCOB), 2002. (Cited on page 30.)

[Ressom 2006] H. W. Ressom. *Biomarker Identification and Rule Extraction from Mass Spectral Serum Profiles.* In R. S. Varghese, E. Orvisky, S. K. Drake, G. L. Hortin, M. Abdel-Hamid, C. A. Loffredo and R. Goldman, editeurs, IEEE Symposium on Computational

Intelligence and Bioinformatics and Computational Biology (CIBCB'06), pages 1–7. IEEE Explorer, 2006. (Cited on page 34.)

[Rezende 2009] S. O. Rezende, E. A. Melanda, M. L. Fujimoto, R. A. Sinoara and V. O. de Carvalho. Combining data-driven and user-driven evaluation measures to identify interesting rules, chapitre III, pages 38–55. Post-Mining of Association Rules: Techniques for Effective Knowledge Extraction, Information Science Reference, 2009. (Cited on page 41.)

[Rifai 2006] N. Rifai and R. E. Gerszten. *Biomarker discovery and validation.* Clinical Chemistry, vol. 52, pages 1635–1637, 2006. (Cited on page 33.)

[Ruggieri 2010] S. Ruggieri. *Frequent regular itemset mining.* In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 263–272, 2010. (Cited on pages 50 and 51.)

[Sahar 1999] S. Sahar and Y. Mansour. *An empirical evaluation of objective interestingness criteria.* In Proceedings of the SPIE Conference on Data Mining and Knowledge Discovery, pages 63–74, 1999. (Cited on page 42.)

[Sandhu 2011] P. S. Sandhu, D. S. Dhaliwal and S. N. Panda. *Mining utility-oriented association rules: An efficient approach based on profit and quantity.* International Journal of the Physical Sciences, vol. 6, no. 2, pages 301–307, 2011. (Cited on page 41.)

[Sarker 2004] R. Sarker and H. A. Abbass. *Differential evolution for solving multi-objective optimization problems.* Asia-Pacific Journal of Operational Research, vol. 21, no. 2, pages 225–240, 2004. (Cited on page 53.)

[Scala 1994] G. Scala. *The expression of the interleukin 6 gene is induced by the human immunodeficiency virus 1 TAT protein.* J. Exp. Medicine, vol. 3, no. 179, pages 961–971, 1994. (Cited on page 167.)

[Schwikowski 2000] B. Schwikowski, P. Uetz and S. Fields. *A network of protein-protein interactions in yeast.* Nature Biotechnology, vol. 18, no. 12, pages 1257–1261, 2000. (Cited on page 33.)

[Sebag 1988] M. Sebag and M. Schoenauer. *Generation of rules with certainty and confidence factors from incomplete and incoherent learning bases.* In Proceedings of the European Knowledge Acquisition Workshop (EKAW), pages 28–1–28–20, 1988. (Cited on page 42.)

[Segal 2003] E. Segal, H. Wang and D. Koller. *Discovering molecular pathways from protein interaction and gene expression data.* Bioinformatics, vol. 19, no. 1, pages i264–i272, 2003. (Cited on page 33.)

[Semenova 2001] T. Semenova, M. Hegland, W. Graco and G. Williams. *Effectiveness of mining association rules for identifying trends in large health databases.* In Workshop on Integrating Data Mining and Knowledge Management. ICDM'01. The 2001 IEEE International Conference on Data Mining, 2001, 2001. (Cited on page 19.)

[Shang 2009] X. Shang, Q. Zhao and Z. Li. *Mining high-correlation association rules for inferring gene regulation networks.* Data Warehousing and Knowledge Discovery, pages 244–255, 2009. (Cited on page 24.)

[Shekofteh 2010] M. Shekofteh. *A survey of algorithms in FCIM.* In Proceedings of the International Conference on Data Storage and Data Engineering (DSDE), pages 29–33, Febuary 2010. (Cited on pages 3 and 48.)

[Shenoy 2000] P. Shenoy, J. R. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa and D. Shah. *Turbo-charging vertical mining of large databases.* In Proceedings of the ACM SIGKDD conference, pages 22–33, 2000. (Cited on page 36.)

[Shimada 2006] K. Shimada, K. Hirasawa and J. Hu. *Class association rule mining with Chi-squared test using genetic network programming.* In Proceedings of the IEEE Conference on Systems, Man, and Cybernetics (SMC), pages 5338–5344, 2006. (Cited on page 53.)

[Shimada 2008] K. Shimada and K. Hirasawa. *Exceptional association rule mining using genetic network programming.* In Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV), 2008. (Cited on page 53.)

[Shoemaker 2007a] B. A. Shoemaker and A. R. Panchenko. *Deciphering protein-protein interactions. Part I. Experimental techniques and databases.* PLoS Computational Biology, vol. 3, no. 3, page e42, 2007. (Cited on page 148.)

[Shoemaker 2007b] B. A. Shoemaker and A. R. Panchenko. *Deciphering protein-protein interactions. Part II. Computational methods to predict protein and domain interaction partners.* PLoS Computational Biology, vol. 3, no. 4, page e43, 2007. (Cited on page 158.)

[Shortliffe 1975] E. Shortliffe and B. Buchanan. *A model of inexact reasoning in medicine.* Mathematical Biosciences, vol. 23, pages 351–379, 1975. (Cited on page 42.)

[Singhal 2007] M. Singhal and H. Resat. *A domain-based approach to predict protein-protein interactions.* BMC Bioinformatics, vol. 8, no. 199, 2007. (Cited on page 158.)

[Smyth 1991] P. Smyth and R. M. Goodman. *Rule induction using information theory.* Knowledge Discovery in Databases, pages 159–176, 1991. (Cited on page 42.)

[Soni 2010] S. Soni and O. P. Vyas. *Using Associative Classifiers for Predicting Analysis in Health Care Data Mining.* International Journal of Computer Applications, vol. 4, no. 5, pages 33–37, 2010. (Cited on page 20.)

[Spruit 2007] M. R. Spruit. *Discovery of association rules between syntactic variables.* In Proceedings of the Meeting of Computational Linguistics in the Netherlands (CLIN), pages 83–98, 2007. (Cited on page 42.)

[Steinbach 2007] M. Steinbach, P.-N. Tan, H. Xiong and V. Kumar. *Objective measure for association pattern analysis.* Contemporary Mathematics, vol. 443, pages 205–226, 2007. (Cited on page 41.)

[Stelle 2011] D. Stelle, M. C. Barioni and L. P. Scott. *Using data mining to identify structural rules in proteins.* Applied Mathematics and Computation, vol. 218, pages 1997–2004, 2011. (Cited on page 30.)

[Stilou 2001] S. Stilou, P. D. Bamidis, N. Maglaveras and C. Pappas. *Mining association rules from clinical databases: An intelligent diagnostic process in healthcare.* Medinfo, vol. 10, pages 1399–1403, 2001. (Cited on page 22.)

[Storn 1996] R. Storn. *On the use of differential evolution for function optimization.* Technical report, ICSI, Berkeley, 1996. (Cited on page 53.)

[Street 2010] J. M. Street and J. W. Dear. *The application of mass-spectrometry-based protein biomarker discovery to theragnostics.* Br. J. Clin. Pharmacol, vol. 69, pages 367–378, 2010. (Cited on page 33.)

[Suzuki 2008] E. Suzuki. *Pitfalls for Categorizations of Objective Interestingness Measures for Rule Discovery.* In Statistical Implicative Analysis, volume 127 of *Studies in Computational Intelligence*, pages 383–395. Springer, 2008. (Cited on page 41.)

[Taboada 2007] K. Taboada, K. Shimada, S. Mabu, K. Hirasawa and J. Hu. *Association rule mining for continuous attributes using genetic network programming.* In Proceedings of the International Conference on Genetic and Evolutionary Computation (GECCO), 2007. (Cited on page 53.)

[Tamir 2006] R. Tamir and Y. Singer. *On a confidence gain measure for association rule discovery and scoring.* VLDB Journal, vol. 15, no. 1, pages 40–52, 2006. (Cited on page 42.)

[Tan 2000] P. N. Tan and V. Kumar. *Interestingness measures for association patterns: A perspective.* In Proceedings of the ACM SIGKDD Workshop on Post-processing in Machine Learning and Data Mining, 2000. (Cited on page 42.)

[Tan 2002] P. Tan, V. Kumar and J. Srivastava. *Selecting the right interestingness measure for association pattern.* In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 32–41, 2002. (Cited on pages 39 and 42.)

[Tan 2004] P.-N. Tan, V. Kumar and J. Srivastava. *Selecting the right objective measure for association analysis.* Information Systems, vol. 29, no. 4, pages 293–313, June 2004. (Cited on page 41.)

[Tang 2005] Y. Tang, B. Jin and Y.-Q. Zhang. *Granular support vector machines with association rules mining for protein homology prediction.* Artificial Intelligence in Medicine, vol. 35, pages 121–134, 2005. (Cited on page 31.)

[Tastan 2009] O. Tastan, Y. Qi, J. Carbonell and J. Klein-Seetharaman. *Prediction of interactions between HIV-1 and human proteins by information integration.* In Proceedings of the Pacific Symposium on Bioinformatics (PSB), pages 516–527, 2009. (Cited on pages 5, 149 and 165.)

[Thorton 2001] J. M. Thorton. *From genome to function.* Science, vol. 292, pages 2095–2097, 2001. (Cited on pages 26 and 29.)

[Tremeaux 2006] J.-M. Tremeaux and Y. Liu. *Mining for association rules in medical data.* Technical report, University Lumiere Lyon 2, 2006. (Cited on page 21.)

[Tuzhilin 2002] A. Tuzhilin and G. Adomavicius. *Handling very large numbers of association rules in the analysis of microarray data.* In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 396–404, 2002. (Cited on page 24.)

[Tyers 2003] M. Tyers and M. Mann. *From genomics to proteomics.* Nature, vol. 422, pages 193–197, March 2003. (Cited on page 26.)

[Vastrik 2007] I. Vastrik, P. D'Eustachio, E. Schmidt, G. Joshi-Tope, G. Gopinath, D. Croft, B. de Bono, M. Gillespie, B. Jassal, S. Lewis, L. Matthews, G. Wu, E. Birney and L. Stein. *Reactome: A knowledge base of biologic pathways and processes.* Genome Biology, vol. 8, no. R39, 2007. (Cited on page 14.)

[Wang 2002] K. Wang, L. Tang, J. Han and J. Liu. *Top-Down FP-Growth for association rule mining.* In Proceedings of the PAKDD conference, pages 334–340, 2002. (Cited on page 47.)

[Wang 2003] G. Wang and R. L. Dunbrack. Pisces: a protein sequence culling server in bioinformatics. Oxford Univ. Press, 2003. (Cited on page 29.)

[Westendorp 1994] M. O. Westendorp, M. Li-Weber, R. W. Frank and P. H. Krammer. *Human immunodeficiency virus type 1 Tat upregulates interleukin-2 secretion in activated T cells.* J. of Virology, vol. 7, no. 68, pages 4177–4185, 1994. (Cited on page 167.)

[Whigham 1995] P. A. Whigham. *Grammatically-based genetic programming.* In Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications (GP), pages 33–41, 1995. (Cited on page 53.)

[White 2001] R. White. Gene transcription, mechanisms and control. Blackwell science, 2001. (Cited on page 25.)

[Wilkins 1994] M. Wilkins, A. Gooley and K. Williams. *Towards the protein genome: rapid identification of 2D spots by amino acid analysis.* In Siena Meeting on 2D Electrophoresis, pages 35–36, Siena, Italy, 1994. (Cited on page 26.)

[Wille 1992] R. Wille. *Concept lattices and conceptual knowledge systems.* Computers Mathematics and Applications, vol. 23, no. 6-9, pages 493–515, 1992. (Cited on page 3.)

[Xion 2006] H. Xion, P.-N. Tan and V. Kumar. *Hyperclique pattern discovery.* Data Mining and Knowledge Discovery, vol. 13, no. 2, pages 219–242, 2006. (Cited on page 42.)

[Xu 2004] X. Xu, G. Cong, B. C. Ooi, K.-L. Tan and A. K. H. Tung. *Semantic mining and analysis of gene expression data.* In Proceedings of International Conference on Very Large Data Bases (VLDB), pages 1261–1264, 2004. (Cited on page 25.)

[Yahia 2006] S. B. Yahia, T. Hamrouni and E. M. Nguifo. *Frequent closed itemset based algorithms: A thorough structural and analytical survey.* ACM SIGKDD Explorations Newsletter, pages 93–104, 2006. (Cited on pages 3, 46, 48 and 133.)

[Yamanishi 2004] Y. Yamanishi, J. P. Vert and M. Kanehisa. *Protein network inference from multiple genomic data: A supervised approach.* Bioinformatics, vol. 20, pages 363–370, 2004. (Cited on pages 5 and 158.)

[Yang 2009] B. Yang, W. Hou, Z. Zhou and H. Quan. *KAAPRO: An approach of protein secondary structure prediction based on KDD* in the compound pyramid prediction model.* Expert Systems with Applications, vol. 36, pages 9000–9006, 2009. (Cited on page 30.)

[Yang 2010] Y. Yang, G. Webb and X. Wu. *Discretization Methods.* In Data Mining and Knowledge Discovery Handbook, Part 1, pages 101–116, 2010. (Cited on page 9.)

[Yang 2011a] B. Yang, W. Qu, Y. Xie and Y. Zhai. *Predicting protein second structure using a novel hybrid method.* Expert Systems with Applications, vol. 38, pages 11657–11664, 2011. (Cited on page 29.)

[Yang 2011b] B. Yang, Q. Wu, Z. Ying and H. Sui. *Predicting protein secondary structure using a mixed-modal SVM method in a compound pyramid model.* Knowledge-Based Systems, vol. 24, pages 304–313, 2011. (Cited on page 30.)

[Yang 2011c] G. Yang, Y. Dang, S. Mabu, K. Shimada and K. Hirasawa. *Searching interesting association rules based on evolutionary computation.* In Proceedings of the PAKDD Workshop on Quality Issues, Measures of Interestingness and Evaluation of Data Mining Models (QIMIE), numéro 7104 de LNAI, pages 243–253, 2011. (Cited on page 53.)

[Yao 1995] Y. Y. Yao. *Measuring retrieval performance based on user preference of documents.* Journel of the American Society for Information Science, vol. 46, no. 2, pages 133–145, 1995. (Cited on page 42.)

[Yao 1999] Y. Y. Yao and N. Zhong. *An analysis of quantitative measures associated with rules.* In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pages 479–488, 1999. (Cited on page 42.)

[Yao 2006a] H. Yao and H. J. Hamilton. *Mining itemset utilities from transaction databases.* Data and Knowledge Engineering, vol. 59, no. 3, pages 603–626, December 2006. (Cited on page 41.)

[Yao 2006b] H. Yao, H. J. Hamilton and L. Geng. *A unified framework for utility based measures for mining itemsets.* In Proceedings of the ACM SIGKDD Workshop on Utility-Based Data Mining (UBDM), pages 28–37, 2006. (Cited on page 41.)

[Yardimci 2006] G. G. Yardimci, A. Kucukural, Y. Saygin and U. Sezerman. *Modified Association Rule Mining Approach for the MHC-Peptide Binding Problem.* In A. Levi et. al., editeur, LNCS 4263, pages 165–173. ISCIS 2006, Springer-Verlag Berlin Heidelberg, 2006. (Cited on page 32.)

[Yeh 2011] J.Y. Yeh, T.H. Wu and C.W. Tsao. *Using data mining techniques to predict hospitalisation of hemodialysis patients.* Decision Support Systems, vol. 50, pages 439–448, 2011. (Cited on page 22.)

[Yeung 2004] K. Y. Yeung, M. Medvedovic and R. E. Bumgarner. *From coexpression to coregulation: How many microarray experiments do we need?* Genome Biology, vol. 5, no. R48, 2004. (Cited on pages 24 and 25.)

[Yin 2003] X. Yin and J. Han. *CPAR: Classification based on predictive association rules.* In Proceedings of the SIAM International Conference on Data Mining (SDM), pages 331–335, 2003. (Cited on page 31.)

[Y.Koh 2005] J. L. Y.Koh, M. L. Lee, A. M. Khan, P. T. J. Tan and V. Brusic. *Duplicate detection in biological data using association rule mining.* In Proceedings of the ECML/PKDD European Workshop on Data Mining and Text Mining in Bio-informatics (DTMBio), pages 34–41, 2005. (Cited on page 25.)

[Yu 2008] G. Yu, K. Li and S. Shao. *Mining high utility itemsets in large high dimensional data.* In Proceedings of the IEEE International Workshop on Knowledge Discovery and Data Mining (WKDD), pages 17–20, 2008. (Cited on page 41.)

[Yule 1900] G.U. Yule. *On the association of attributes in statistics.* Philosophical Transactions of the Royal Society of London, vol. A, no. 194, pages 257–319, 1900. (Cited on page 42.)

[Yule 1912] G.U. Yule. *On the methods of measuring association between two attributes.* Journal of the Royal Statistical Society, vol. 75, pages 579–642, 1912. (Cited on page 42.)

[Zadzic 2006] F. Zadzic, T. S. Dillon, A. S. Sidhu, E. Chang and H. Tan. *Mining Substructure in Protein Data.* In Sixth IEEE International Conference on Data Mining (ICDMW'06), 2006. (Cited on page 30.)

[Zaki 1997] M. J. Zaki, S. Parthasarathy, M. Ogihara and W. Li. *New algorithms for fast discovery of association rules.* In Proceedings of the KDD conference, pages 283–286, 1997. (Cited on pages 36 and 46.)

[Zaki 2000] M. Zaki, S. Jin and C. Bystroff. *Mining residue contacts in proteins using local structure predictions.* In IEEE International Symposium on Bioinformatics and Biomedical Engineering, November 2000. (Cited on page 30.)

[Zaki 2002] M. J. Zaki and C. J. Hsiao. *CHARM: An efficient algorithm for closed itemset mining.* In Proceedings of the SIAM International Conference on Data Mining (SDM), pages 457–473, 2002. (Cited on pages 11 and 36.)

[Zaki 2004] M. J. Zaki. *Mining non-redundant association rules.* Data Mining and Knowledge Discovery, vol. 9, no. 3, pages 223–248, 2004. (Cited on page 8.)

[Zeng 2001] J. Zeng, H. R. Treutlein and G. B. Rudy. *Predicting sequences and structures of MHC-binding peptides: a computational combinatorial approach.* Journal of Computer-Aided Molecular Design, pages 573–576, 2001. (Cited on page 32.)

[Zhang 2000] T. Zhang. *Association rules.* In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pages 245–256, 2000. (Cited on pages 3, 8, 10 and 42.)

[Zhang 2004] L. V. Zhang, S. L. Wong, O. D. King and F. P. Roth. *Predicting co-complexed protein pairs using genomic and proteomic data integration.* BMC Bioinformatics, vol. 5, no. 38, 2004. (Cited on pages 5, 26 and 158.)

[Zhang 2009a] X. Zhang, S. Orcun, M. Ouzzani and C. Oh. *Mass informatics in differential proteomics*. In Encyclopedia of Data Warehousing and Mining, Second Edition, pages 1176–1181. IGI Global, 2009. (Cited on pages 28 and 32.)

[Zhang 2009b] Y. Zhang, L. Zhang, G. Nie and Y. Shi. *A survey of interestingness measures for association rules*. In Proceedings of the International Conference on Business Intelligence and Financial Engineering (BIFE), pages 460–463, 2009. (Cited on page 41.)

[Zhang 2010] M. Zhang and C. He. *Survey on association rules mining algorithms*. Advancing Computing, Communication, Control and Management, vol. 56, pages 111–118, 2010. (Cited on pages 45 and 53.)

[Zhou 1991] X. Zhou and T. S. Dillon. *A statistical-heuristic feature selection criterion for decision tree induction*. IEEE Transactions on Pattern Analysis and Machine Intellegence, vol. 13, no. 8, pages 834–841, 1991. (Cited on page 42.)

[Zhou 2008] Q. Zhou and J. S. Liu. *Extracting sequence features to predict protein-DNA interactions: A comparative study*. Nucleic Acids Research, vol. 36, pages 4137–4148, 2008. (Cited on page 31.)

[Zhou 2010] Z. Zhou, B. Yang and W. Hou. *Association classification algorithm based on structure sequence in protein secondary structure prediction*. Expert System with Applications, vol. 37, pages 6381–6389, 2010. (Cited on pages 29 and 30.)