

Developing Efficient Algorithms for Incremental Mining of Sequential Patterns

A Thesis

*Submitted in fulfillment of the
requirements for the award of the degree of*

Doctor of Philosophy

Submitted by

Bhawna Mallick

(Registration No: 950903027)

Under the supervision of

Dr. Deepak Garg

Associate Professor,
Computer Science and Engineering
Department, Thapar University,
Patiala – 147004

Dr. P S Grover

Professor of Computer Science,
Guru Tegh Bahadur Institute of Technology,
Guru Gobind Singh Indraprastha University,
Delhi – 110006



**Computer Science and Engineering Department
Thapar University,
Patiala-147004 (Punjab), India.**

August 2013

To

My Family

CERTIFICATE

I hereby certify that the work which is being presented in this thesis entitled **DEVELOPING EFFICIENT ALGORITHMS FOR INCREMENTAL MINING OF SEQUENTIAL PATTERNS**, in fulfillment of the requirements for the award of degree of **DOCTOR OF PHILOSOPHY** submitted in Computer Science and Engineering Department (CSED), Thapar University, Patiala, Punjab, is an authentic record of my own work carried out under the supervision of Dr. Deepak Garg and Dr. P. S. Grover, and refers the work of other researchers, which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.



(Bhawna Mallick)

Registration No. 950903027

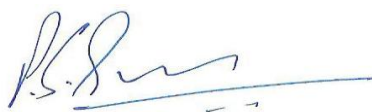
This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge and belief.



(Deepak Garg)

Associate Professor,
Computer Science and Engineering
Department, Thapar University,
Patiala – 147004.

Supervisor



(P.S. Grover)

Professor of Computer Science,
Guru Tegh Bahadur Institute of Technology,
Guru Govind Singh Indraprastha University,
Delhi – 110006.

Supervisor

ACKNOWLEDGEMENTS

It would not have been possible to write this doctoral thesis without the help and support of the kind people around me, to only some of whom it is possible to give particular mention here.

At the outset, this research endeavor has been only possible with the help, support and able guidance of my supervisors, Dr. Deepak Garg and Dr. P.S. Grover. I am highly indebted to Dr. P. S. Grover for his advice, encouragement and unsurpassed knowledge of comparative linguistics. The good support, diligence and friendship of Dr. Deepak Garg have been invaluable on both an academic and a personal level, for which I am extremely grateful. It has indeed been a privilege to carry out this research work under their guidance.

I would like to acknowledge the Director and Dean (R&D), Thapar University for the academic and technical support which has been indispensable. I express my gratitude to the Doctoral Committee comprising of Dr. Maninder Singh, Dr. R. K. Sharma and PG Coordinator, Dr. Inderveer Chana for monitoring the progress and providing valuable suggestions for improvement of my Ph. D. work from time to time. I also thank the faculty and staff of Computer Science and Engineering Department, especially the Head of Department, Dr. Maninder Singh in promoting a stimulating and welcoming academic and cooperative environment.

I am grateful to the Director and Management, Galgotias College of Engineering and Technology, Greater Noida, Uttar Pradesh for providing me resources and facilities to carry out this research work. I would also like to thank my colleagues and friends for their support and valuable suggestions throughout. I am thankful to my friends working in corporate for providing necessary feedback and suggestions required for this research work.

My hearty thanks to all the anonymous referees of several international research journals in the field of intelligent data mining and fuzzy set theory. The research work presented in this thesis report has accomplished the eminence due to the constructive comments and ideas for improvement by them.

I would like to thank many acclaimed researchers across the world for timely contributing the reprints of their valuable research papers for my reference apropos this thesis report. These include Dr. Lei Chang, Dr. C.H. Cheng, Dr T.P. Hong, Dr. Jiang Hua, Dr. S. Piramuthu, Dr. Jong Bum Lee, Dr. S. K. Gupta, Dr. M.Y. Lin, Dr. Jian Pei, Dr. V. Kumar, Dr. X. Wu, Dr. Mingming Zhou, and several others.

I express gratitude to my loving husband Nitin for his personal support and great patience at all times. I acknowledge the constant cooperation from him without which this work could not have been possible. My parents and sisters have given me their unequivocal support throughout, as always, for which my mere expression of thanks likewise does not suffice. I dedicate this thesis to my family.

Above all, I pay my reverence to the almighty God.

Date: July, 2013.

Place: Thapar University, Patiala.

A handwritten signature in dark ink, appearing to read 'Bhawna', with a long horizontal stroke extending to the right and two small dots below it.

(Bhawna Mallick)

ABSTRACT

There has been an increase in our capabilities of both generating and collecting data with the progress of humanity and development of technology. The ultimate intent of this massive data collection is to utilize it for different reasons that include process efficiency, system automation and competitive benefits among others. This is achieved by determining both implicit and explicit unidentified patterns in data that can direct the process of decision making. Data mining is a process of finding rules and patterns that describe the characteristic properties of the data. Sequential pattern mining is one of the methods of data mining. Sequential pattern is a sequence of item sets that frequently occur in a specific order; where all items in the same item set are supposed to have the same transaction time value.

Sequential pattern mining is used in wide spectrum of areas, such as financial data analysis of banks and other financial institutions, retail industry, customer shopping history, goods transportation, consumption and services, telecommunication industry, biological data analysis, scientific applications, network intrusion detection.

The sequential pattern mining algorithms, in general, result in enormous number of interesting and uninteresting patterns. Therefore, the miner may have difficulty in selecting the interesting/significant ones. The core idea behind this study is the incorporation of specific constraints into the mining process. One another important concern of sequential pattern mining methods is that they are based on assumption that the data is centralized and static. For dynamic data addition and deletion in the databases, these methods waste computational and input/output (I/O) resources. It is undesirable to mine sequential patterns from scratch each time when a small set of sequences grow, or when some new sequences are added, deleted or updated in the database.

There is consistent outdateding of the discovered sequential patterns because of the incremental evolution of databases (progressive databases) over time. This necessitates incremental mining of sequential patterns from progressive database so as to capture the dynamic nature of data addition, deletion and updating. In present study, an extensive study of the algorithms that perform discovery of constraint-based sequential patterns

and incremental mining of sequential patterns available in the literature is done. On the basis of the study; efficient algorithms for incremental mining of sequential patterns from progressive database are proposed.

This study contributes to the literature as it identifies the approach and constraints that can be used for developing algorithms for the incremental mining of sequential patterns. The constraints have been selected to keep track of customer purchasing history and further be used for customer value analysis for CRM. A COPRE (COnstraint-based PRefix) framework is developed based on pattern growth approach that can incorporate any number of constraints and used for any domain. The first three proposed algorithms, “Constraint-based sequential pattern mining algorithm incorporating compactness, length and monetary”; “CFM-PrefixSpan algorithm” and “Progressive CFM-Miner algorithm” are based on this framework. The real-world business and commercial databases consist of numerical and time-stamped data. We can get more relevant rules from these databases by making use of fuzzy set theory to minimize the sharp cut between intervals. Fuzzy sequential mining generate simple and practical patterns which are close to human reasoning. So, the fourth algorithm, “Progreslattice-Miner (PLM) algorithm” is based on incremental mining of fuzzy sequential patterns from progressive databases.

Finally, a model “Constraint-Guided Progressive Sequential Mining Waterfall” is devised to demonstrate the use of the devised algorithms for the CRM domain.

OBJECTIVES OF THE RESEARCH WORK

The objectives and scope for present research work are as below:

- i) To study and analyze various incremental mining and constraint based algorithms of sequential pattern mining.
 - Different constraints have been studied and analyzed for the customer purchasing behavior and CRM specific domain.
 - Constraints that have a crucial role in the customer value analysis which is core of CRM have been identified.
 - A framework has been designed to incorporate any number of constraints belonging to different category using a pattern growth approach.

- A systematic method for application of constraints in the algorithms has been studied and suggested.
- The review of the algorithms for the incremental mining of sequential patterns present in the literature is done for the progress made so far and the existing challenges.

ii) To develop efficient sequential pattern mining algorithms by incorporating dynamic nature of data addition and deletion for progressive and distributed databases.

- A constraint-based sequential pattern mining algorithm incorporating compactness, length and monetary apart from frequency constraints, is developed and compared with naïve approach using synthetic database. This is primarily applicable for the static databases and can be used to study and analyze customer purchasing behavior.
- A constraint-based sequential pattern mining algorithm, “CFM- PrefixSpan” is developed incorporating compactness, frequency and monetary constraints from the customer value analysis perspective. The algorithm is compared with the naïve approach using both synthetic and real databases. The experimental results well justified the efficiency and effectiveness of the proposed algorithm.
- To consider the dynamic data addition and deletion for progressive and distributed databases, an incremental algorithm for sequential pattern mining “Progressive CFM-Miner” is developed. The algorithm is further compared with a well-known incremental algorithm for its performance.
- As the real data is numerical and quantitative, that often belongs to fuzzy boundaries, like similarity, preference and uncertainty which can only be accounted with the fuzzy theory. An algorithm to mine fuzzy sequential patterns from the progressive databases (considering the dynamic data addition and deletion of data); ProgressLattice Miner (PLM) is developed. In this work, the fuzzification is done in time of arrival instead of the frequency of certain quantitative sequences.

iii) *To verify and validate the designed sequential pattern mining algorithms.*

- The four algorithms designed and developed for sequential pattern mining have been compared with their naïve or other well-known algorithms.
- The first algorithm and CFM-PrefixSpan algorithm developed is compared with the PrefixSpan algorithm using both the synthetic and real databases.
- The third algorithm “Progressive CFM-Miner” is compared with the well-known incremental algorithm IncSpan for a) the effect of compact threshold, b) the effect of support value and c) scalability analysis.
- The fourth algorithm “ProgressLattice Miner” is compared with the Nancy P. Lin et al.s’ algorithm.
- The results are obtained for three standard evaluation metrics: generated number of sequential patterns, computational time and memory usage. The developed algorithms have given better performance, in general for these metrics.

A model called “Constraint guided progressive sequential mining waterfall model for CRM” has been developed that demonstrates the application of the developed algorithms for the CRM domain.

THESIS OUTLINE

Thesis is divided into eight chapters:

Chapter 1 covers our understanding of mining of sequential patterns and its basic approaches. It discusses the limitations of the sequential pattern mining algorithms. Further the mechanisms to handle these limitations are introduced that forms the problem definition. The chapter confers the importance of constraint-based sequential pattern mining. It gives fair understanding of the need of incremental mining of sequential pattern mining. It discusses the use of fuzzy set theory along with incremental mining to give more practical and simple fuzzy sequential patterns.

Chapter 2 covers the comprehensive literature survey of algorithms for constraint-based and incremental mining of sequential patterns. Different incremental mining algorithms for sequential patterns have been studied for their characteristics. Taxonomy of these algorithms has been proposed. The concept of each of these algorithms has been analyzed for discovery of sequential patterns. This detail survey has identified the gap in the existing algorithms and concluded that the incremental mining of constraint-based sequential patterns from progressive databases require our attention. Further, the fuzzification can be done in the proposed algorithm to have more relevant and simple sequential patterns.

Chapter 3 discusses the basis of selection of approach that is used for the proposed algorithms in this study. The literature has different types of constraints. The identification of constraints is mainly based on application. We have selected the constraints based on studying the customer purchasing behavior and customer value analysis. There has been no study on selection of constraints based on customer value analysis which is important for Customer Relationship Management. This chapter focuses on how the selection of constraints can help in satisfying the important parameters of CRM. The technique to apply the constraints in the mining process is also discussed in this chapter.

Chapter 4 proposes a framework called COPRE (COnstraint- based PREfixSpan). This framework can be used to incorporate any number of constraints in projection- based pattern growth mining approach. The usefulness of the COPRE framework lies in the fact that it can be used for any domain and application. The framework is used by the proposed algorithms. A concept of inclusive and exclusive property of constraint based sequential pattern mining algorithm is introduced in this chapter. Most of the constraint based sequential pattern mining algorithms in the literature have been classified in terms of this property to help further study in this direction.

Chapter 5 introduces two algorithms “Constraint-based sequential pattern mining algorithm incorporating compactness, length and monetary” and “CFM-PrefixSpan algorithm”. Both these algorithms are for mining constraint based sequential patterns and are based on projection-based pattern growth approach. However, the first algorithm makes use of the constraints that are appropriate for the study of the customer purchasing behavior but the latter chooses for the constraints from the perspective of customer value analysis. The incorporation of the constraints is done by using the proposed COPRE frameworks for both the algorithms. The algorithms are explained with the help of a detailed example and further their efficiency is justified by comparing them with the naïve approach PrefixSpan in terms of computation time, memory usage and the number of sequential patterns generated.

Chapter 6 presents an algorithm for incremental mining of sequential patterns “Progressive CFM-Miner”. This algorithm works for progressive databases and it can handle the maintenance of CFM-PrefixSpan algorithm presented in Chapter 5. The dynamic data addition, deletion and updating in the database which requires re-mining in case of CFM-PrefixSpan algorithm is handled efficiently by this incremental mining algorithm that uses tree data structure. The stepwise execution, definitions of terms used, pseudocode and illustrative example for the algorithm are given in this chapter. The algorithm is compared for its performance with other well-known incremental mining algorithm IncSpan. They are compared on three metrics again, that is, computational time, memory usage and number of sequential patterns generated using both synthetic and real datasets.

Chapter 7 discusses the need of fuzziness in incremental mining to give more simple and practical sequential patterns. The concept of fuzzy sequential pattern summarization using member functions and knowledge of linguistics is explained. The role of lattice structure for fuzzy pattern summarization is further discussed. A data structure called ProgresLattice that is based on standard lattice structure is introduced in this chapter. An algorithm that makes use of fuzziness, proposed Progreslattice and incremental mining is proposed in Chapter 7 to mine fuzzy sequential patterns. The algorithm is explained with the help of example along with its stepwise execution. This efficiency of this algorithm is compared with another fuzzy sequential pattern mining algorithm called as Nancy, L.P. *et al.s'* algorithm.

Chapter 8 concludes this thesis by giving the application of the proposed algorithms. A model called "Constraint guided progressive sequential mining waterfall model for CRM (CGP-SMW)" is proposed in this chapter that shows the stages in CRM application where the use of these proposed algorithms can be done. The chapter further gives various application of any algorithm for incremental mining of sequential patterns from progressive databases. The limitation of this study and future research direction in this area is also suggested.

CONTENTS

ACKNOWLEDGEMENTS	ii
ABSTRACT	iv
THESIS OUTLINE	viii
CONTENTS	xi
LIST OF FIGURES	xvi
LIST OF TABLES	xx
ABBREVIATIONS	xxii
LIST OF PUBLICATIONS BY THE AUTHOR	xxv
CHAPTER 1: MINING OF SEQUENTIAL PATTERNS	1 -22
1.1 Introduction	1
1.2 Data Mining	1
1.3 Sequential Pattern Mining (SPM)	4
1.4 Basic Approaches for Sequential Pattern Mining	6
1.4.1 Apriori-Based Approach	7
1.4.2 Apriori Vertical Formatting Approach	7
1.4.3 Projection Based Pattern Growth Approach	8
1.5 Limitations of Sequential Pattern Mining	9
1.6 Handling SPM Challenges: Problem Definition of The Thesis	11
1.6.1 Sequential Pattern Mining Using Constraints	11
1.6.2 Incremental Mining of Sequential Patterns (ISPM)	17
1.6.3 Fuzzy Sequential Pattern Mining	20
1.7 Summary	21
CHAPTER 2: LITERATURE REVIEW	23 - 46
2.1 Introduction	23
2.2 Constraint-based Sequential Pattern Mining	23

2.3 Incremental Mining of Sequential Pattern (ISPM)	26
2.3.1 Apriori-based ISPM Algorithms	27
2.3.2 Projection-based ISPM Algorithms	29
2.3.3 Frequent Pattern Tree-based Algorithms	31
2.3.4 Hybrid/Other Algorithms	32
2.3.5 Analysis of ISPM Algorithms	38
2.4 ISPM from Progressive Databases	40
2.5 Fuzzy Sequential Pattern Mining	46
2.6 Summary	46

CHAPTER 3: SELECTION OF APPROACH AND CONSTRAINTS FOR PROPOSED ALGORITHMS

47 - 61

3.1 Introduction	47
3.2 Selection of Approach: Pattern-Growth	47
3.3 Selection of Constraints	48
3.3.1 Monetary	49
3.3.2 Compactness	49
3.3.3 Length	50
3.4 Selection of Domain- Specific Constraints	50
3.4.1 Customer Relationship Management	51
3.4.2 Customer Value Analysis Perspective	53
3.5 Technique to Apply Constraints in Mining Process	59
3.6 Summary	61

CHAPTER 4: PROPOSED COPRE FRAMEWORK TO INCOPORATE CONSTRAINTS IN SPM ALGORITHMS

62 - 69

4.1 Introduction	62
4.2 Need for a Framework to Incorporate Constraints in SPM	62
4.3 Exclusive and Inclusive Property	64
4.4 Proposed COPRE Framework	66
4.5 Summary	69

CHAPTER 5: PROPOSED SPM ALGORITHMS USING CONSTRAINTS	70- 86
5.1 Introduction	70
5.2 Preliminaries	70
5.2.1 Terms Used in Algorithms	71
5.3 Constraint- based Sequential Pattern Mining Algorithm Incorporating Compactness, Length and Monetary	72
5.3.1 Pseudo Code	73
5.3.2 Illustration with an Example	74
5.3.3 Experimental Setup and Dataset Description	76
5.3.4 Result Analysis	77
5.3.5 Performance Analysis	78
5.4 CFM-PrefixSpan: A Constraint –based Algorithm for Mining Sequential Patterns with CRM perspective	79
5.4.1 Pseudo Code	80
5.4.2 Illustration with an Example	81
5.4.3 Experimental Setup and Dataset Description	82
5.4.4 Result Analysis	83
5.4.5 Performance Analysis	83
5.5 Summary	86
 CHAPTER 6: PROPOSED ALGORITHM FOR INCREMENTAL MINING OF SEQUENTIAL PATTERNS	 87 - 102
6.1 Introduction	87
6.2 Progressive CFM-Miner: An Algorithm for Incremental Mining of Sequential Patterns	87
6.3 Preliminaries	88
6.3.1 Terms Used in Algorithm	89
6.3.2 Types of Update Operations in Incremental Mining	90
6.4 Stepwise Execution of the Algorithm	91
6.4.1 Mining of CFM Sequential Patterns from the Static Database	91

6.4.2 Building up the CFM-Tree from the CFM patterns	92
6.4.3 Handling the Update Operation	94
6.4.4 Handling the Node Deletion Operation in the Updated CFM-Tree	95
6.4.5 Mining of CFM patterns from the Progressive Database	96
6.5 Pseudo Code	98
6.6 Experimental Setup and Dataset Description	99
6.7 Result Analysis	100
6.8 Performance Analysis	101
6.9 Summary	102
 CHAPTER 7: INCORPORATING FUZZINESS IN INCREMENTAL MINING OF SEQUENTIAL PATTERNS	 103- 128
7.1 Introduction	103
7.2 Motivation for the Use of Fuzzy Sequential Patterns	104
7.3 Fuzzy Sequential Patterns Summarization	105
7.4 Role of Lattice Structure	108
7.5 ProgresLattice Miner: Proposed Algorithm for Incremental Mining of Fuzzy Sequential Patterns	111
7.5.1 Preliminaries	111
7.5.2 Phases of the Algorithm	113
7.5.3 Stepwise Execution	114
7.5.4 An Illustrative example	116
7.5.5 Experimental Setup and Dataset Description	123
7.5.6 Performance Analysis	124
7.6 Summary	127
 CHAPTER 8: CONCLUSIONS AND FUTURE RESEARCH	 129-138
8.1 Concluding Observations	129
8.2 Application of Proposed Algorithms	130
8.2.1 CRM key Performance Indicators	130

8.2.3 Constrained Guided Progressive Sequential Mining Waterfall (CGP-SMW) Model	132
8.3 Application of Incremental Sequential Patterns Mining Algorithms	136
8.4 Limitations of Present Research	137
8.5 Recommended Future Work	138
REFERENCES	140- 158
APPENDIX	159- 160

LIST OF FIGURES

Figure 1.1: Information extractions from databases	2
Figure 1.2: Different types of time-related patterns generated by SPM algorithms	6
Figure 1.3: Challenges faced by the Sequential pattern mining algorithms	9
Figure 1.4: Role played by constraints in Sequential Pattern mining	12
Figure 1.5: Sliding window concept implemented using window size '3'	13
Figure 1.6: Gap constraints implemented using window size '2'	14
Figure 1.7: Changes in database due to dynamic transactions	17
Figure 1.8: Process flow to obtain F^p by re-mining the database using SPM algorithms	18
Figure 1.9: Process flow to obtain F^p by ISPM algorithms	18
Figure 1.10: Evolution of SPM algorithms with time	20
Figure 1.11: Comparative study of three methods of Sequential pattern mining	21
Figure 2.1: Taxonomy of Incremental mining of Sequential Patterns Algorithms	26
Figure 2.2: ISPM algorithms that use Apriori as naïve approach	27
Figure 2.3: ISPM algorithms that use Tree Projection as naïve approach	29
Figure 2.4: Evolution of Frequent Pattern Tree for maintaining sequential patterns	31
Figure 2.5: Combined approaches used by DSPID algorithm in its implementation	33
Figure 2.6: Combined approaches used by IncSPAM algorithm in its implementation	33
Figure 2.7: Combined approaches used by IMCS algorithm in its implementation	34
Figure 2.8: Combined approaches used by BSPinc algorithm in its implementation	35
Figure 2.9: Combined approaches used by IMSP algorithm in its implementation	35
Figure 2.10: The naïve approach used by ISPM algorithms from progressive databases	42
Figure 2.11: Sample database 'S' to illustrate the PISA and Weighted algorithm	44

Figure 2.12: Diagrammatic representation giving overview of CISpan algorithm	45
Figure 3.1: Identified '17' vital parameters for Customer Value Analysis	54
Figure 3.2: 360° Customer Value Model based on Constraint based SPM	55
Figure 3.3: Three integral aspects of Customer loyalty	56
Figure 3.4: Relations between Customer Retention and Enterprise Profit	57
Figure 3.5: Relations between Customer Satisfaction and Enterprise Credibility	58
Figure 3.6: Approaches to incorporate constraints for CRM strategies	59
Figure 3.7: Three phases of proposed approach of using constraints for CRM domain	60
Figure 4.1: Constraint based sequential pattern mining in an 'Exclusive' manner	64
Figure 4.2: Constraint based sequential pattern mining in an 'Inclusive' manner	66
Figure 4.3: Proposed COPRE Framework – Stepwise execution	68
Figure 5.1: Comparison graph for number of sequential patterns generated by CFML- PrefixSpan and PrefixSpan algorithms	78
Figure 5.2: Runtime performance of CFML-PrefixSpan and PrefixSpan algorithms	79
Figure 5.3: Comparison graph for number of sequential patterns generated with various compact thresholds by CFML-PrefixSpan and PrefixSpan algorithms	79
Figure 5.4: Number of sequences generated by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values	84
Figure 5.5: Memory usage by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values	84
Figure 5.6: Runtime performance by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values	84
Figure 5.7: Number of sequences generated by PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases	85
Figure 5.8: Memory usage by PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases	85

Figure 5.9: Computation time comparison for PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases	86
Figure 6.1: CFM-Pattern Tree for the Static Database	93
Figure 6.2: An Updated CFM Tree with newly inserted items	95
Figure 6.3: Updated CFM-tree with no Incompact nodes	96
Figure 6.4: Final updated CFM-tree for progressive database	97
Figure 6.5: Mining of progressive CFM patterns from the Updated CFM-tree	98
Figure 6.6: Number of sequences generated by Progressive CFM-Miner and IncSpan with different support values for real and synthetic databases	101
Figure 6.7: Computation time for Progressive CFM-Miner and IncSpan with different support values for real and synthetic databases	102
Figure 6.8: Memory usage of Progressive CFM-Miner and IncSpan with different support values for real and synthetic databases	102
Figure 7.1: Fuzzy sets for hardware, software and accessories itemsets	107
Figure 7.2: Graph summarizing the sequences with the fuzzy sets	110
Figure 7.3: Lattice structure summarizing the sequences with fuzzy sets	111
Figure 7.4: Model for ProgresLattice Miner algorithm	113
Figure 7.5: Formula $f(x)$ to compute triangular membership function	117
Figure 7.6: Triangular membership function	117
Figure 7.7: Triangular membership functions with defined parameters values	118
Figure 7.8: Lattice structure for the sequences of T2 to T6 of Table 7.5	119
Figure 7.9: Complete lattice structure with updating in the existing customer	120
Figure 7.10: Complete lattice structure with updating of new customer sequences	121
Figure 7.11: Complete ProgresLattice structure with deletion operation in database	122
Figure 7.12: Number of sequences generated by ProgresLattice Miner and Lin N.P.	

et al's with different support values for real and synthetic databases	125
Figure 7.13: Computation time for ProgresLattice Miner and Lin N.P. et al's with different support values for real and synthetic databases	125
Figure 7.14: Memory usage of ProgresLattice Miner and Lin N.P. et al's with different support values for real and synthetic databases	126
Figure 7.15: Number of sequences generated for ProgresLattice Miner and Lin N.P. et al's with different number of records of real and synthetic databases	126
Figure 7.16: Computation time for ProgresLattice Miner and Lin N.P. et al's with different number of records of real and synthetic databases	127
Figure 7.17: Memory usage for ProgresLattice Miner and Lin N.P. et al's with different number of records of real and synthetic databases	127
Figure 8.1: CRM Key Performance Indicators along with the Desired and Minimum Priority level for an Enterprise	131
Figure 8.2: Constraint Guided Progressive –Sequential Mining Waterfall (CGP-SMW) Model	134
Figure 8.3: Proposed CGP-SMW Model in Detail	135

LIST OF TABLES

Table 1.1: Transaction Database 'S'	4
Table 1.2: Sequence Database 'D'	5
Table 2.1: Concepts used by ISPM algorithms in their implementation	37
Table 2.2: Characteristics of various ISPM algorithms	39
Table 2.3: Concepts used by ISPM algorithms on progressive databases in their implementation	41
Table 3.1: Research studies for customer classification/churn prediction in CRM using data mining techniques	52
Table 4.1: Constraint-based Sequential pattern mining algorithms classified as per Exclusive/Inclusive property	65
Table 5.1: Sequence Database 'SD1'	75
Table 5.2: Monetary Value Table 'MVT1'	75
Table 5.3: Projected Database for 1-length Compact Frequent pattern	75
Table 5.4: Comparison of patterns discovered from CFML-PrefixSpan and Prefix- Span algorithms	77
Table 5.5: Number of sequential patterns generated by CFML-PrefixSpan and Prefix- Span algorithms for various length thresholds	78
Table 5.6: Purchasing Sequence Database of Customers	81
Table 5.7: Monetary value of each item	81
Table 5.8: Projected Database of 1-length Constraint based Sequential Patterns	82
Table 5.9: Reduced number of patterns obtained using the CFM-PrefixSpan algorithm	83
Table 6.1: Sample Database 'SD2'	92
Table 6.2: Monetary Value Table 'MVT2'	92
Table 6.3: Mined CFM- Sequential Pattern	92
Table 6.4: Final progressive CFM-patterns	97
Table 6.5: Comparison of algorithms in mining of meaningful sequences	100
Table 7.1: Sequence Database 'SD3'	105
Table 7.2: Parsing results of Sequence Database 'SD3'	106

Table 7.3: Membership degree of customer sequences for fuzzy sets	108
Table 7.4: Customer transactions from Progressive database	117
Table 7.5: Fuzzy sequences transformed from progressive database	118
Table 7.6: Mined frequent fuzzy sequential patterns from incremental database	123
Table 7.7: Mined frequent fuzzy sequential patterns from progressive database	123

ABBREVIATIONS

BSPinc	Backward SPAM for incremental mining
CAP	Constraint Algorithm for Pattern mining
CBASW	Customer Bit Vector Array with Sliding Window
CCSM	Cache-based Constrained Sequence Miner
CFM	Compactness, Frequency, Monetary
CFML	Compactness, Frequency, Monetary, Length
CGP-SMW	Constraint Guided Progressive Sequential Mining Waterfall
CISpan	Comprehensive Incremental Sequential PAttreN mining
Con_FST	Construction algorithm for Frequency Sequence Tree
COPRE	COntstraint based PREfix
CRM	Customer Relationship Management
cSPADE	Constrained Sequential Pattern, Discovery using Equivalent Classes
CSTree	Closed Sequence Tree
DELISP	DELimited Sequential Pattern Mining
DSPID	Discover Sequential Patterns in Incremental Database
FAL	Fast Accumulation Lattice
FP tree	Frequent Pattern tree
FS tree	Frequent Sequence tree
FSS	Frequent Sequence Set
FUFP tree	Fast Updated Frequent Pattern tree
FUSP tree	Fast Updated Sequential Pattern tree
GSP	Generalized Sequential Pattern
GTC	Graph for Time Constraints
ICspan	Incremental mining of Closed sequential pattern in multiple data streams
IMCS	Incremental Maintenance of Closed Sequential Pattern
IMSP	Incremental Mining of Sequential Patterns
IncSP	Incremental Sequential Pattern Update

IncSPAM	Incremental Sequential PAttern Mining
IncSpan	Incremental Sequential pattern mining
ISE	Incremental Sequence Extraction
ISM	Incremental Sequence Mining
ISPM	Incremental Sequential Pattern Mining
IU_D	Incremental updating with Deletion
IUS	Incrementally Updating Sequences
KB	Knowledge Base
KDD	Knowledge Discovery in Database
KISP	Knowledge base assisted Incremental Sequential Pattern
KPI	Key Performance Indicators
MA_D	Maintenance Algorithm when Deleting some information
MaxGap	Maximum Time Gap
MEMISP	MEMory Indexing for Sequential Pattern mining
METISP	MEMory Time Indexed Sequential Pattern mining
MFS	Mining Frequent Sequence
MILE	MIning in muLtiple strEams
min_nbd_sup	Minimum support of negative border sequences
min_sup	Minimum support threshold
MinGap	Minimum Time Gap
MSPVF	Mining Sequential Patterns using Time Constraint based on Vertical Format
MVT	Monetary Value Table
PISA	Progressive mIning of Sequential pAttern
PLM	ProgresLattice Miner
POI	Period of Interest
Progressive CFM-Miner	Progressive Compactness, Frequency, Monetary-Miner
PrefixSpan	Prefix-projected Sequential pattern mining
PS_tree	Progressive Sequential _tree
PSP	Pseudo projection Sequential Pattern

PSPM	Progressive Sequential Pattern Mining
PTAC	Sequential frequent Patterns mining with Tough Aggregate Constraints
SD	Sequence Database
SPA	Sequential Pattern Automaton
SPADE	Sequential Pattern, Discovery using Equivalent Classes
SPAM	Sequential Pattern Mining
SPAMS	Sequential Pattern Automaton for Mining Streams
SPaRSe	Sequential Pattern mining with Restricted Search
SPIRIT	Sequential Pattern mining with Regular expression constraints
SPM	Sequential Pattern Mining
Stree_PS	Sequence tree_PrefixSpan
TiWS	Time-Interval Weighted Sequential Patterns
WLPMiner	Weighted Length decreasing Pattern Miner
Wspan	Weighted sequential pattern mining

LIST OF PUBLICATIONS MADE BY AUTHOR FOR THE THESIS

- 1) "CFM- PrefixSpan: A pattern growth algorithm incorporating compactness and monetary". *International Journal of Innovative Computing, Information and Control*, ISSN 1349-4198, Volume 8, Number 7(A), pp: 4509 – 4520, July 2012. (A **SCI journal** with an impact factor of **2.932** as per 2010 JCR report and 1.9 as per 2011 JCR report).
- 2) "Progressive CFM-Miner: An Algorithm to Mine CFM- Sequential Patterns from A Progressive Database". *International Journal of Computational Intelligence System*, ISSN 1875- 6891, Volume 6, Issue 2, pp. 209-222, January 2013. (A **SCI journal** with an impact factor of **1.49** as per 2011 JCR report).
- 3) "Incremental Mining of Sequential Patterns: Progress and Challenges". *Intelligent Data Analysis- an International Journal*, ISSN 1088-467X, Volume 17, Number 3, pp. 507-530, May 2013. (A **SCI journal** with an impact factor of 0.448).
- 4) "Constraint-Based Sequential Pattern Mining: A pattern growth algorithm incorporating compactness, length and monetary". *International Arab Journal of Information Technology*, ISSN 1683-3198, Volume 11, Number1, January 2014. (A **SCI journal** with an impact factor of 0.127).
- 5) "Progreslattice: An efficient algorithm to mine fuzzy sequential patterns from the progressive databases" - has been re-submitted after revision for publication to *International Journal of Uncertainty, Fuzziness and Knowledge System*, ISSN 0218-4885. (A **SCI journal** with an impact factor of 1.78 as per JCR report).
- 6) "CRM Customer Value based on Constrained Sequential Pattern Mining". *International Journal of Computer Applications*, ISSN 0975-8887, Volume 64, Number 9, pp. 21-29, February 2013, DOI 10.5120/10663-5434.
- 7) "A framework for discovering reduced combinations of sequential patterns with categorical domain constraints" – under review with revision to *International Journal of Computer Systems Science and Engineering*, ISSN 0267- 6192. (A **SCI journal** with an impact factor of 0.319).
- 8) "Constraint guided progressive sequential mining waterfall model for CRM" - has been communicated to the *Journal of Database Management- an International Journal*, ISSN 1063-8016. (A **SCI journal** with an impact factor of 0.875).
- 9) "Fuzzy sequential pattern summarization with lattice structure". *International Journal of Hybrid Technology*, Science & Engineering Research Support Society, Korea, ISSN 1738-9968, Volume 6, Number 3, May 2013 (indexed by **EI Compendex**)

Chapter 1

Mining of Sequential Patterns¹

1.1. INTRODUCTION

The advancement in computer and technology has provided us a fountain of wisdom but at the same time also has delivered a flood of data. It has been estimated that the amount of information in the world doubles every 20 months (Gupta *et al.*, 2001; Ahmed *et al.*, 2010). The automation of commercial and business activities contributes further to this growing data. It becomes vital to analyze this huge amount of data to understand and use it further (Dunham *et al.*, 2001). The use of statistical techniques for analysis of data is being done from long time; but they are applicable for simple applications. There is requirement for advanced techniques for intelligent data analysis to provide competitive advantage. An important area called as Data Mining, or sometimes even termed as Knowledge Discovery has evolved in recent times. It refers to the nontrivial extraction of implicit, previously unknown, and potentially useful information from data (Piramuthu S, 2004b; Gupta *et al.*, 2005). The extracted information that is of interest to the user, certain and effective as per the defined criteria, is referred as knowledge. A pattern by itself is a collection of records in the database that have same attribute values. To discover useful patterns from huge and complex databases, we require both empirical and knowledge based techniques to gain from both the features of statistical regularity as well as the domain knowledge.

1.2. DATA MINING

Information or patterns can be extracted from huge data repositories with data mining. The data can be analyzed with the purpose to find rules and patterns that describe its

¹ The major discussions of this Chapter have been (i) published in *Intelligent Data Analysis*, 17(3), pp: 507-530, 2013 and (ii) communicated to *International Journal of Computer Systems Science and Engineering*.

distinguished properties by using the data mining (Agrawal *et al.*, 1993, Han and Kamber, 2001) methods. A model representing the semantic structure of the dataset can be formed with the discovered information. This representation could further be used on new data for classification or prediction. Data mining is sometimes referred as a synonym for another popular term, Knowledge Discovery in Database (KDD) (Fayyad *et al.*, 1996). However, it is considered as one of the main process of Knowledge Discovery in Database by many people. The three KDD processes are shown in Figure 1.1. The first process is called preprocessing that includes cleaning, integration, selection and transformation of data. It is executed to prepare the data for mining. The main process of KDD is the data mining process, where different algorithms are applied to produce hidden knowledge.

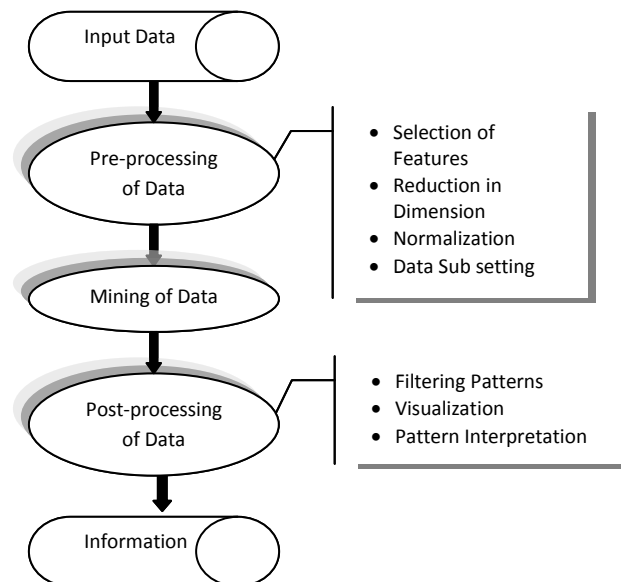


Figure 1.1 Information extractions from databases

The last process called post processing evaluates the mining result according to users' requirements and domain knowledge. As soon as the satisfactory results are obtained the results need to be visualized and presented in understandable format.

Of these, data mining is an extremely important step (Zhao and Bhowmick, 2003) that analyzes the data with the objective of finding rules and patterns to categorize the data (Agrawal *et al.*, 1993). It has wide application in all fields like medicine, finance, marketing, sales, stock market, insurance and military (Khan *et al.*, 1997; 2007; Julisch 2002; Bullo *et al.*, 2003; Piramuthu S, 2004a; Bhatia *et al.*, 2007; 2009; 2010). It can be

performed on different type of data repositories including databases as well as transient data. These data repositories include relational databases, transactional databases, object-relational databases, spatial databases, temporal databases, sequence databases, data streams, among several others. The challenges and techniques of data mining vary with the type of data repository (Han and Kamber, 2001). This study is based on the sequence database which is discussed in detail in Section 1.3.

There are two classes of data mining tasks - descriptive and prescriptive. Descriptive mining summarizes or characterizes general properties of data in data repository, while prescriptive mining performs inference on current data, to make predictions based on the historical data (Chelluri and Kumar, 2001). These two processes are sometimes referred to as supervised and unsupervised learning, respectively. Supervised learning has tagged data that is used to identify and categorize the future data. Unsupervised learning finds correlations without any external inputs other than the raw data to identify and categorize them. These techniques determine the type of patterns that are discovered from the data repositories. Among the various descriptive data mining tasks, frequent mining, association rule mining and sequential mining have been the most commonly employed tasks to facilitate business development. The patterns that occur frequently in the database are called as frequent patterns. These patterns may include itemsets that appear together, correlated and frequent; such data when extracted from the transaction databases or other data repositories is called association rule mining (Cheung *et al.*, 1996; Gupta *et al.*, 2000; Chiang 2010); for example, customer buying soap and shampoo together. However, frequently occurring subsequence like a pattern where the customer purchase a butter followed by bread is a frequent sequential pattern.

Sequential pattern mining is similar to association rule mining, with an important difference of having the events being linked by time (Antunes and Oliveria, 2003; Sobh T., 2007). Sequential patterns indicate the correlation between transactions while association rule represents intra transaction relationships (Srikant and Agrawal, 1996a). The mining results are based on the items being brought together frequently within the same transaction in case of the association rule mining (Maragatham and Lakshmi, 2011; Yafi *et al.*, 2012). While the results of sequential pattern mining are about which items are bought in a certain order, with those items coming from different transactions (Agrawal and

Srikant, 1995; Zhao and Bhowmick, 2003). Sequential patterns are generated by finding the correlation between transactions while association rule mining is based on intra transactions. To elaborate it further, association rule mining results are based on items that are purchased together frequently and belong to same transaction (Dubois *et al.*, 2003). However, results generated by sequential pattern mining are based on items that are brought in a certain order by same customer and can be from different transactions.

This research study is based on sequential pattern mining and the next section discusses other important aspects of this data mining task.

1.3. SEQUENTIAL PATTERN MINING

Sequential pattern mining [SPM] one of the imperative subjects of data mining (Frawley *et al.*, 1992), is an additional approval of association rule mining (Massegia *et al.*, 2003; Hou and Zhang, 2008). It discovers frequent sequences in a sequence database satisfying a support threshold (Mannila *et al.*, 1995; Zhao and Bhowmick, 2003). This threshold is called as the minimum support threshold for the sequential pattern mining. Since the number of sequences can be very large, and users have different interests and requirements, usually a minimum support is pre-defined by users. By using the minimum support we can prune out sequential patterns of no interest, consequently make the mining process more efficient. Obviously a higher support of sequential pattern is desired for more useful and interesting sequential patterns.

The sequence database differs from the transaction database, as illustrated by the following example, which also examines the potential sequence patterns in a sequence database. A simple transactional database includes a unique transaction identity number and a list of the items making up the transaction. Let our customer purchase transaction database be 'S' given in Table 1.1. A transaction identifier 100 refers to a unique transaction with items *bread, butter, milk*.

Transaction Identifier	Item Sets
100	bread, butter, milk
200	bread, egg, milk
300	bread, milk, cheese
400	butter, cheese, sausage

Table 1.1 Transaction Database 'S'

On the contrary, a sequence database includes a sequence identity number and a sequence, where a sequence is an ordered list of events denoted as $\langle e_1 e_2 e_3 e_4 \dots e_n \rangle$ where e_1 occurs before e_2 , which occurs before e_3 and so on. However the event by itself is an item set, that is, an unordered list of items denoted as $(x_1, x_2, \dots, x_m)$ where x_j is an item. Let our customer purchase sequence database be ' D ' given in Table 1.2 with 100, 200, 300 and 400 as identifier for different sequences. A sequence 100 has ordered list of events $\langle \text{bread (bread, butter, egg) (bread, egg) milk (egg, sausage)} \rangle$ where purchase of bread as an event e_1 has occurred before purchase of another event e_2 (bread, butter, egg). The event e_2 is an item set consisting of unordered list of items - bread, butter and egg.

Sequence Identifier	Sequences
100	$\langle \text{bread (bread, butter, egg) (bread, egg) milk (egg, sausage)} \rangle$
200	$\langle \text{(bread, milk) egg (butter, egg) (bread, cheese)} \rangle$
300	$\langle \text{(cheese, sausage) (bread, butter) (milk, sausage) egg butter} \rangle$
400	$\langle \text{cheese oats (bread, sausage) egg butter egg} \rangle$

Table 1.2 Sequence Database ' D '

So, if $I = \{i_1, i_2, \dots, i_m\}$ be a set of literals called items, then an item set is a non-empty set of items. A sequence s is a set of item sets ordered according to their time stamp. It is denoted by $[s_1, s_2, \dots, s_n]$, where $s_j, j \in 1, \dots, n$, is an item set. A k -sequence is a sequence of k items (or of length k). A sequence $[s_1, s_2, \dots, s_n]$ is a sub-sequence of another sequence $[s'_1, s'_2, \dots, s'_m]$, if there exist integers $i_1 < i_2 < \dots < i_j < \dots < i_n$ such that $s_1 \subseteq s'_{i_1}, s_2 \subseteq s'_{i_2}, \dots, s_n \subseteq s'_{i_n}$.

Each sequence corresponds to a temporally ordered list of events, where each event is a collection of items (item set) supposed to have the same transaction time value or within a time gap (Zhao and Bhowmick, 2003; Agrawal *et al.*, 1993). The problem of sequential pattern mining is to find all frequent subsequences whose occurrence frequency is no less than the user defined minimum threshold *min_sup*. For our example in Table 1.2, $\langle \text{bread (butter, egg) milk egg} \rangle$ is a subsequence of $\langle \text{bread (bread, butter, egg) (bread, egg) milk (egg, sausage)} \rangle$. Given a minimum support threshold, $\text{min_sup} = 2$, $\langle \text{(bread, butter) egg} \rangle$ is a sequential pattern.

The temporal ordering among the events is induced by the absolute timestamps associated with the events (Albert-Lorincz and Boulicaut, 2003). From the temporal aspect, the environment is dynamic and user behavior is not always the same. In different time periods, we may find different sets of rules or the significance of a rule may change. Hence sequential patterns are useful in giving us different time related patterns as shown in Figure 1.2.

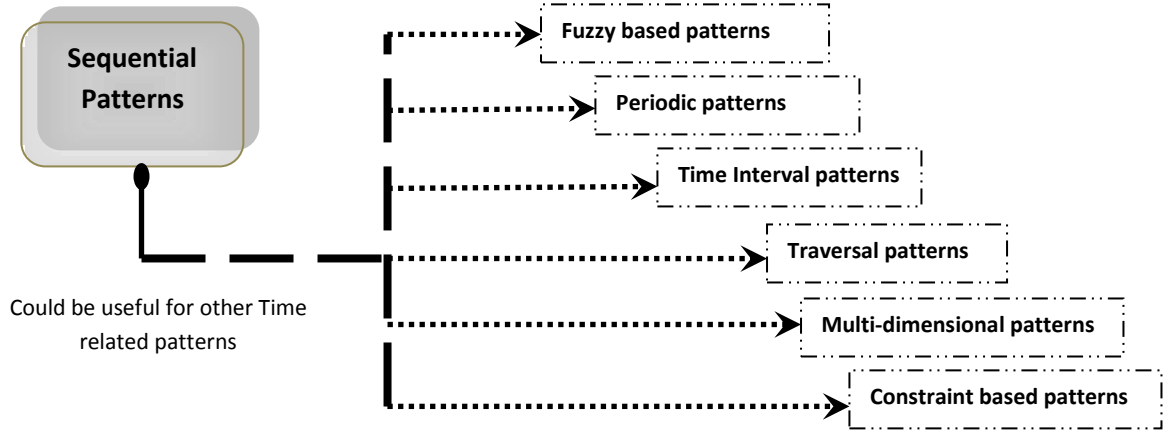


Figure 1.2 Different types of time-related patterns generated by SPM algorithms

Sequential pattern mining finds application in various areas like telecommunication industry to study customer call pattern, financial institutions like banks for analysis of financial data, retail industry to analyze the sales data (Kapoor and Piramuthu, 2009) , stock market analysis and prediction to purchase mutual funds, biological data analysis as in DNA sequence analysis (She *et al.*, 2003), scientific applications as weather prediction, intrusion detection in networks (Thampi *et al.*, 2008; 2010), world wide web as to study web traversal sequences (Myra S, 2000; Khan *et al.*, 2003; 2005; Feng *et al.*, 2007; Parmar *et al.*, 2007; Chen and Cook, 2007).

1.4. BASIC APPROACHES FOR SEQUENTIAL PATTERN MINING

It is challenging to discover all frequent sequences in huge databases as the search space is extremely large. The factors that make sequential pattern mining difficult and time-consuming are as follows;

- i. The pattern can be formed using single item as well as item sets.
- ii. The details regarding the number of item sets in a pattern or the number of items in an item set is not known beforehand.

- iii. The permutation and combination of all possible items in the database is used to form patterns.

There is plenty of literature available on different approaches used by SPM algorithms to handle these challenges (Agrawal and Srikant, 1995; Srikant and Agrawal, 1996b; Massegia *et al.*, 1998; Zaki, 1998; Zaki, 2001; Ayres *et al.*, 2002; Zhao and Bhowmick, 2003; Yan *et al.*, 2003; Pei *et al.*, 2004; Wang and Han, 2004; Antunes and Oliveira, 2004a; Luo and Chung, 2005; Han *et al.*, 2005; Lo *et al.*, 2006; Wang *et al.*, 2007; Han *et al.*, 2007; Wu *et al.*, 2007; Zhang *et al.*, 2008a; Ren and Li, 2008; Li *et al.*, 2009; Liao *et al.*, 2009; Cui and Zhong, 2009). These approaches can be classified into three basic categories as below:

1.4.1. Apriori-based approach

AprioriSome, AprioriAll and Dynamicsome algorithms were proposed by Agrawal *et al.*, (1993, 1995) based on apriori approach. The approach uses the prior knowledge of frequent sequences. It makes use of iterative approach for candidate generation. It starts with the generation of frequent 1 sequences and then generates frequent (k+1) sequences from the set of frequent k sequences (called as candidate). However to decide whether a particular sequence is frequent, the entire database is scanned. The GSP (Generalized Sequential Pattern) (Zhao and Bhowmick, 2003) algorithm is an extension of the apriori model (candidate generation approach) and is based on sliding window principle. The concept of sliding window suggests that the items are considered in the same transaction only if the distance between maximal transaction time and minimal transaction time of the items is less than the window size.

1.4.2. Apriori vertical formatting approach

SPADE (Sequential Pattern, Discovery using Equivalent classes) algorithm (Zaki, 2001) makes use of this method. The sequential database is converted into a vertical id-list database format and each id is associated with corresponding items and the time stamp. This algorithm aims to find frequent sequences using efficient lattice search techniques and simple joins. All the sequences are discovered with only three passes over the database. The frequent items are discovered in first scan, the frequent sequences of length

2 are searched in second scan and the last scan associates a table of the sequences id and item sets id in the database to the corresponding frequent sequences of length 2. The method forms the enumeration of the candidate sequences, based on their common prefixes. As AprioriAll, GSP and other algorithms depend little on main memory so SPADE outperforms these algorithms. SPAM (Sequential PAttern Mining) (Ayres *et al.*, 2002) is another method that represents the database in the main memory. A vertical bitmap representation of the database for candidate as well as support counting is generated by this algorithm.

1.4.3. *Projection based pattern growth approach*

This approach avoids candidate generation and only a restricted part of the database is searched. FreeSpan (Han *et al.*, 2005) is the first algorithm proposed that considers the projection method for SPM. This work has been continued with PrefixSpan algorithm (Pei *et al.*, 2004). The frequent items of database are first identified and then the projected databases with remaining data sequences are generated by the algorithm. So, the suffixes of sequences from original database which are grouped by prefixes form projected databases. This process is repeated till there is no frequent item left in the projected database. PrefixSpan scans a database every time which is much smaller than original database due to projection method. Unlike GSP, PrefixSpan can handle very long sequential pattern more efficiently. This method is also called level-by-level projection.

PSP algorithm (Massegia *et al.*, 1998, Han *et al.*, 2005] has implemented pseudo projection technique, which do not construct projection database physically. Each postfix is represented by a pair of pointer and offset value in the main memory. The use of the algorithm is limited for projected database and its associated pseudo-projection processing structure that can fit in the main memory. SPaRSe (Sequential PAttern mining with Restricted SEarch) (Srikant and Agrawal, 1996b) is an algorithm that makes apriori-based method as efficient as pattern-growth method under specific conditions. It gives better efficiency for high pattern density conditions as it uses both the concepts of candidate generation and projected database. The algorithm maintains a list of supporting sequences for each candidate. It then verifies the existence of support only in the subset of sequences that are super-sequences of generated candidates (Han *et al.*,

2005). MEMISP (MEMory Indexing for Sequential Pattern mining) (Zhao and Bhowmick, 2003; Han *et al.*, 2007) algorithm requires one pass over database and at most two passes for very large database. It avoids candidate generation and intermediate database projection by implementing techniques of recursive searching and indexing strategy, to generate sequential patterns from data sequences that are stored in memory.

1.5. LIMITATIONS OF SEQUENTIAL PATTERN MINING

In general, SPM algorithms either result in enormous or very few patterns based on the user specified support threshold. Very few sequential patterns are given in case of large support threshold value. On the contrary, the mining algorithms result with enormous patterns for a smaller support threshold value. So, a tradeoff is made between efficiency and effectiveness by these methods. Ignoring small differences in the problem definition, most of the studies in literature lack user controlled focus in the pattern mining process. In fact, the interaction of the user with the pattern mining system is limited to specifying a threshold value of desired support for the extracted patterns.

Although there is an improvement in the efficiency of mining complete set of sequential patterns, but still in many cases sequential pattern mining faces strong challenges [Figure 1.3]. The limitations of SPM algorithms can be summarized as below:

- i. **Efficiency:** A huge set of sequential patterns are generated especially if the database is large. Generally, only a small subset of such patterns is of an interest for the user.
- ii. **Understandability and Usefulness:** It is difficult to understand and hard to use the large number of mined sequential patterns for a real world application.

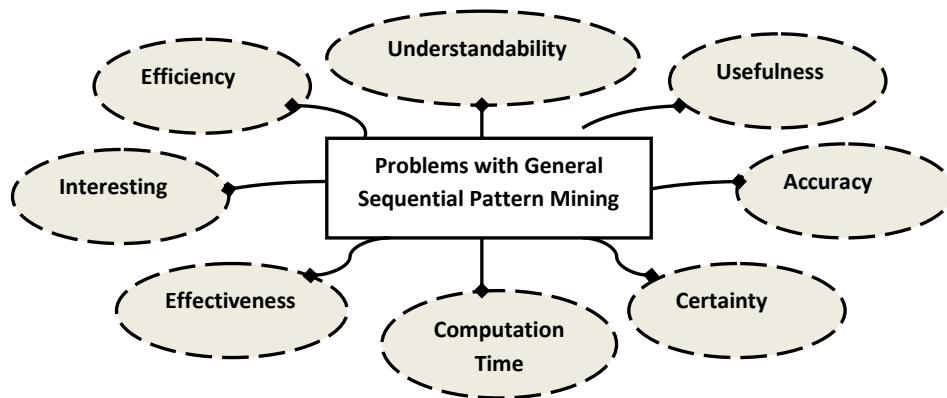


Figure 1.3 Challenges faced by the Sequential pattern mining algorithms

- iii. **Effectiveness:** In addition, the algorithms may reduce the number of patterns but unimportant patterns are still found in the result patterns. It would be better if the unimportant patterns could be pruned first, resulting in fewer but important patterns after mining.
- iv. **Certainty:** Although efficient algorithms have been developed to tackle these problems, the performance of the algorithms dramatically degrades in case of mining long sequential patterns in dense databases or using low minimum supports.
- v. **Interestingness:** In general, SPM considers subsequences and items in a sequence uniformly. The data elements that have different importance in real world applications are not differentiated, and thus general sequential pattern mining is limited in finding more interesting characteristics embedded in the real world applications. There are several applications in which specific sequences are more important or have more priority than other sequences.
- vi. **Computation Time:** Mining a large amount of sequential patterns from large sequence databases is a computationally expensive task. If we can focus on only those sequential patterns that are interesting to users, we may be able to save a lot of computation cost (Srikant and Agrawal, 1996b).
- vii. **Accuracy:** To ensure that the results of sequential pattern mining are complete and accurate is a challenge.
- viii. **Assumption that database is static:** The basic approaches used for sequential pattern mining work in a one-time manner as they are based on the postulation that database is static. However, any real time databases grow dynamically and the updates are inevitable. These algorithms re-mine the entire database to commensurate the changes and re-discover the new patterns from updated database. So they are expensive and time-consuming methods.
- ix. **Assumption that database is centralized:** The basic approaches of sequential pattern mining have no consideration for the databases that are in distributed environment. The algorithms' functionality is based on assumption that the database to be mined is centralized so waste computational and input/output (I/O) resources for excessive communication overhead for distributed data.

- x. **Assumption that database is binary-valued:** This has been noticeably observed that the real-world databases consist of numerical and time-stamped data. The sequential pattern mining is based on binary valued transaction data. This should however be extended to fuzzy sequential pattern mining for quantitative valued data. Fuzzy sequential mining generates simple and practical patterns which are close to human reasoning.

1.6. HANDLING SPM CHALLENGES: PROBLEM DEFINITION OF THE THESIS

The sequential pattern mining is based on the concept to capture typical behaviors repeated by individuals over time to be relevant for the decision maker. The anti-monotone property (Pei *et al.*, 2002) has been mainly used to prune infrequent patterns in sequential pattern mining. That is, if a sequential pattern is infrequent, all super patterns of the sequential pattern must be infrequent. But with huge number of patterns, the user may have difficulty in distinguishing the significant ones. The incorporation of additional constraints into mining is a possible and effective solution to this problem. It also handles the issues of giving useful, understandable and interesting patterns. Further the notion of incremental mining of sequential patterns both from dynamic and progressive databases is introduced in this section. The detail explanation of these concepts justify how the limitation of basic approaches of sequential pattern mining of assuming the database to be static and memory-resident can be resolved. The fuzzy sequential pattern mining, on the other hand gives flexibility in terms of treating the quantitative real time databases.

1.6.1. Sequential Pattern Mining Using Constraints

It has been recognized that frequency is not the best measure to use in determining the significance of a pattern in many applications (Han and Fu, 1996; Srikant and Agrawal, 1996b; Guha *et al.*, 1999, Pei *et al.*, 2002). When using the single frequency constraint, the traditional mining method often generates a huge number of patterns and rules, but most of them are worthless, so there is need of constraint-based sequential pattern mining [Figure 1.4].

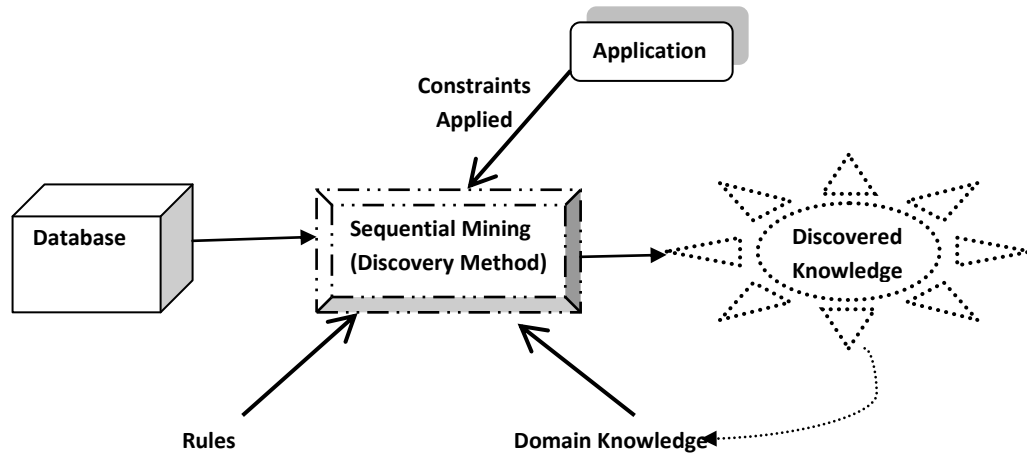


Figure 1.4 Role played by constraints in Sequential Pattern Mining

Let a constraint be represented as ‘CST’ for sequential pattern β , and a boolean function, $CST(\beta)$ on the set of all sequences. The constraint-based sequential pattern mining is a process to find the entire set of sequential patterns, represented as $SEQ(CST)$ that satisfies a given constraint ‘CST’ (Guha et al., 1999). There are different points of views to examine and further characterize constraints. There are number of constraints in the literature, however, the selection of constraint in the mining process determines the interestingness, usefulness, effectiveness, accuracy and certainty of the resultant sequential patterns. The different category of constraints based on application domains includes the following:

- i. **Time constraint:** It is defined only in sequence databases where a time-stamp is associated with each transaction in every sequence. The constraint is applied to find how a sequence matches a given pattern where the minimum support is specified by the user. It is also termed as a support-related constraint. To elaborate further we say that to find whether a sequential pattern satisfies a time constraint one needs to examine the complete sequence database not only a single sequence of transactions. Thus, the database ‘DB’ is searched completely in order to compare a pattern sequence ‘PAT’, with all data sequences ‘d’ in ‘DB’ and all the item sets in ‘d’ are grouped together with respect to the time constraint. The time constraint could further be classified as duration and gap constraint.
 - a. **Duration Constraint:** This constraint requires that the time-stamp difference between the first and the last transaction in a sequential pattern must be longer or

shorter than a given period. So application of the duration constraint gives the maximum allowed time difference between the most recent and initial occurrences of events in the entire sequence. It is implemented as a 'sliding window' across the event set for the temporal sequences. The sliding window calculates the difference in transaction-times between the transactions in the set and if it is less than the user-specified window-size then such a data-sequence contributes to the support of a sequence. Figure 1.5 illustrates an example with the window size given as '3'. The first transaction set has start transaction time (l_1) as '1' and end transaction time (u_1) as '4'. Using the sliding window concept, the second transaction set have '2' and '5' as the start (l_2) and end (u_2) transaction time value, whereas the third transaction set have '3' and '6' as the start (l_3) and end (u_3) transaction time value respectively.

For many applications it is irrelevant that the items of a sequential pattern are present in two different transactions, as long as these transactions have transaction-times within the same time window.

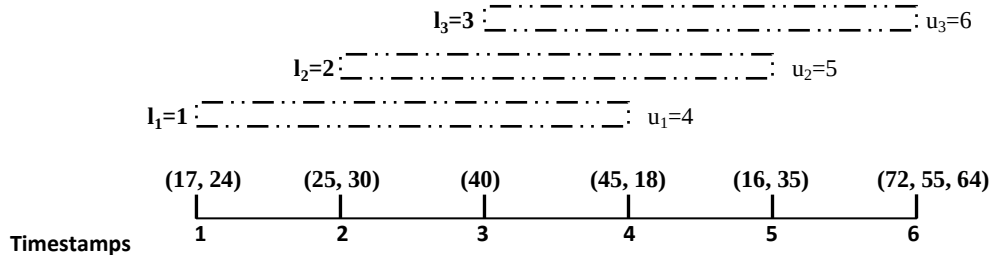


Figure 1.5 Sliding Window concept implemented using window size '3'

b. Gap Constraint: A limit on the separation of two consecutive elements of a sequence is imposed by the gap constraint [Figure 1.6]. This constraint is significant for the applications having long sequences as in case of protein sequence analysis. It could further be categorized as:

- *Maximum Time Gap (MaxGap) Constraint:* Maximum Gap is allowed as the time difference between the latest occurrence of an item in an item-set and the earliest occurrence of an item in its immediately preceding item-set.
- *Minimum Time Gap (MinGap) Constraint:* Minimum Gap is required as the time difference between the earliest occurrence of an item in an item-set and the latest occurrence of an item in its immediately preceding item-set (Zaki, 2000).

For example, the window size is defined as '2' in Figure 1.6. The gap constraints using this window size specifies two transaction sets with 'l₁' as start time and 'u₁' as end time for the first transaction, whereas 'l₂' and 'u₂' as start time and end time for the second transaction.

- The transaction time difference between l₂ (4) and u₁ (3), which is '1' in this case is the minimum gap.
- The transaction time difference between u₂ and l₁ is the maximum gap. For our example, it is '5'.

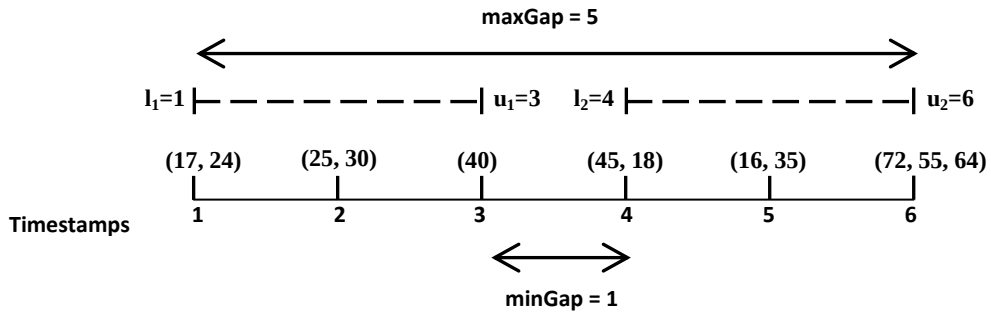


Figure 1.6 Gap constraints implemented using window size '2'

- ii. **Item constraint:** This constraint specifies which of the item or item set should be or should not be the present in the sequential pattern. Suppose we are interested in customer purchase pattern of a grocery shop for all items other than 'butter'. In this case, 'CST' is butter, and $CST_{butter}(\beta) = \text{all items purchased by particular customer other than butter, that is, } \{\forall i, 1 \leq i \leq \text{len}(\beta), \beta[i] \neq \text{butter}\}$.
- iii. **Length constraint:** This constraint imposes the condition on the length of the patterns. Both the number of distinct items and the maximum number of items per transaction can be restricted by the length constraint. So either the number of occurrences of items or the number of transactions can be specified by this constraint. For example, we are interested in all such transactions where the customer has made a purchase of 8 or more items. We can get the desired sequential patterns by imposing length constraint as $CST_{\text{len}}(\beta) = (\text{len}(\beta) \geq 8)$.
- iv. **Super pattern constraint:** This constraint when applied to a database gives all such patterns that contain a particular pattern as a sub-pattern. For example in case of customer purchasing database, we may like to know all such customers who

purchased bread followed by purchase of butter and egg. It could be represented as $CST(\beta) = \langle \text{bread butter egg} \rangle \subseteq \beta$.

- v. **Regular expression constraint:** If the constraint is specified as a regular expression over the set of items using the regular expression operators like disjunction, union, Kleen closure etc., then it is regular expression constraint CST_{RE} . A sequential pattern is said to satisfy a regular expression constraint CST_{RE} only if a deterministic finite automaton (DFA) accepts it. For example, a regular expression constraint $CST_{RE}(\beta) = [\langle a * \{bb|(bm)f|ff\} \rangle \in \beta]$.
- vi. **Sequential Interestingness constraint:** A sequential pattern includes sequential sub-patterns whose frequencies are not always high. The support of a sequential pattern evaluates its frequency. The other parameter, confidence, evaluates the frequency of the pattern when its sub-patterns are given. Usually, sequential mining methods discover patterns with high support and the discovery of all sequential patterns with high confidence is not ensured. If a smaller threshold for the support is used then the sequential patterns with high confidence can be generated. But this leads to discover many sequential patterns. The confidence parameter can be used to predict remaining item sets with high probability in a sequential pattern when their sub-patterns are known. The pattern is tied to the sub-patterns with high confidence, though the frequency of the sub-pattern is not high. The parameter α (≥ 0) represents the importance of the frequency of the pattern and termed as confidence priority. The criterion is called the sequential interestingness. The pattern that is bigger than or equal to the given minimum sequential interestingness is called the interesting pattern. The confidence can evaluate the relationships between sequential patterns and their sequential sub patterns (Sakurai *et al.*, 2008).
- vii. **Weight Constraint:** In real applications, there are some sequences more important or have priority than others. However, the sequential pattern mining approaches consider all items and sequences uniformly. For example, in customer purchasing behavior, a customer purchasing a gold ring and chain is more important than the one purchasing a silver ring and chain considering the monetary aspect. We may be more interested in first customer sequence though with low minimum support than the other customer purchasing sequence with high support. The problem of weighted

sequential pattern mining is to find the complete set of weighted sequential patterns in the database with a user specified minimum support and a weight constraint. The weights of items are given according to the priority or importance. The main consideration in applying a weight constraint to sequential pattern mining is anti-monotone which represents the downward closure property may be broken. A sequence with lower weight can be a frequent sequential pattern by combining items with higher weight in the sequence. To resolve these problems, a weight range is defined and used to give weight boundaries (Yun and Leggett, 2006).

- viii. **User-defined constraint:** The user is often interested in specific constraints that can be incorporated to get sequential patterns that are useful for a particular application. These user defined constraints may or may not be derived from the other constraints. For example, the time constraint is a more generalized constraint which may not always be directly applicable for customer purchasing sequence. If the retail store manager is interested to get the customer purchasing sequential patterns in the recent past, he can use a recency constraint that is derived from the time constraint having user specified recent time duration. Similarly for the educational learning system, pedagogical constraints like action, context constraints for generating sequential learning patterns can be used (Zhou and Xu, 2010).
- ix. **Aggregate constraint:** This is used for a collection of items that satisfy a particular pattern and user specified threshold. There are several aggregate functions that could be applied on a cumulative sequential pattern like sum, average, mean, maximum, minimum and standard deviation. For example, a retail store manager may be interested in a pattern that gives maximum profit in a customer purchasing sequence.

The constraints can also be classified as anti-monotone, monotone, succinct, convertible, and inconvertible, based on interactions between constraints and SPM algorithms. Pei *et al.* (2002) classified constraints into seven categories, which include temporal and non-temporal constraints, in addition to these five categories.

1.6.2. Incremental Mining of Sequential Patterns

Most of the basic solutions are based on the assumption that the mining is performed on static databases. But modern day databases are being continuously updated and are

dynamic in nature. In practice, databases are appended with new data sequences, either in existing sequences or as new sequences. Some of the existing patterns may become invalid as their minimum support become insufficient with respect to the currently updated database. Some new patterns may even get created due to the increase in their support value. In order to keep the current and up-to date sequential patterns, re-execution of the mining algorithm on the whole updated database is required with basic approaches of SPM. However, it takes more time than required in mining with the original database because of the new data sequences appended.

Therefore, the incremental mining of sequential patterns is required which utilizes previously discovered knowledge to solve the maintenance problem of the dynamic updated database efficiently without re-mining from scratch (Massegia *et al.*, 2003; Cheng *et al.*, 2004). This research study refers to it as **Incremental Sequential Patterns Mining (ISPM)**. It is defined as the process to compute the set of sequential patterns from the updated database by using the two operations stated as;

- i. To find all such sequential patterns those become frequent with the increment in the database, though they may not be frequent in the original database.
- ii. To examine all such transactions in the original database that can be extended to become frequent.

For the purpose of representation, let us consider an original database 'U' and an incremental database 'I' where new transactions or new customers are added to 'U' [Figure 1.7].

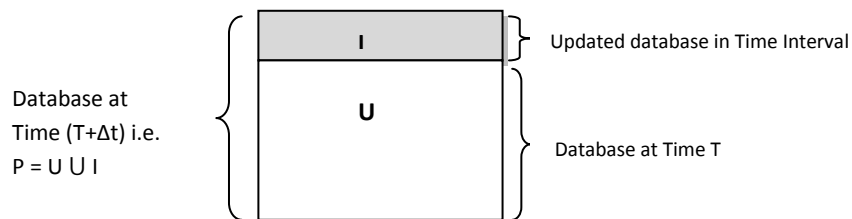


Figure 1.7 Changes in database due to dynamic transactions

The updated database containing sequences from both U and I is represented as $P = U \cup I$. If F^U is the set of frequent sequences in U, then the problem of ISPM is to find frequent sequence patterns in P, denoted as F^P , with respect to the same minimum support. Unlike

the conventional static mining algorithms, the incremental approach avoids re-running of mining algorithms from scratch when the database is updated [Figure 1.8].

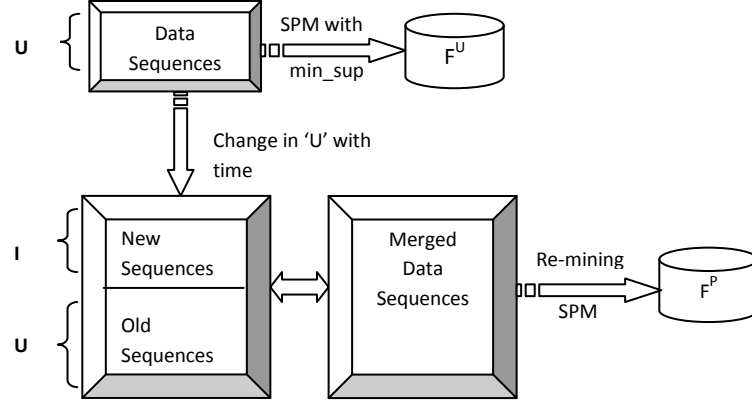


Figure 1.8 Process flow to obtain F^P by re-mining the database using SPM algorithms

ISPM algorithms reuse previously mined information and combine this information with the fresh data to efficiently compute the new set of frequent sequences (Massegia *et al.*, 2003; Lee *et al.*, 2010) as depicted in Figure 1.9. ISPM has received a great deal of attention for ensuring scalability and facilitating knowledge discovery when data is dynamic and distributed.

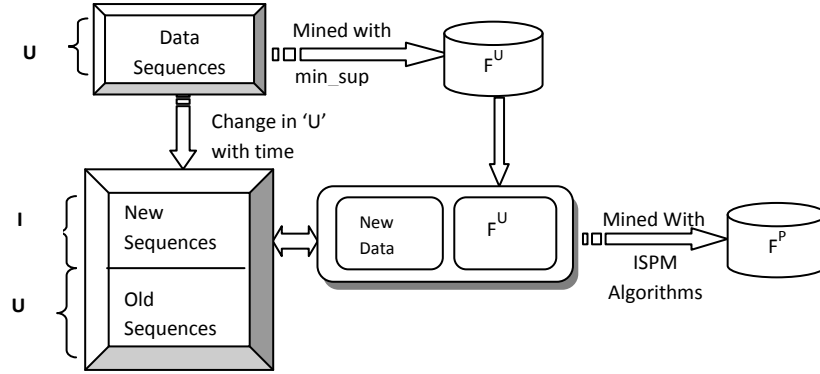


Figure 1.9 Process flow to obtain F^P by ISPM algorithms

1.6.2.1. Limitations of ISPM algorithms

Some data is often deleted from sequence databases in real time applications, like electronic commerce, web usage mining, financial banks and services. This is done for different reasons, (i) to save storage space, (ii) data may not be interesting any longer, or (iii) data may have become invalid. The insertion of sequences or appending items at the end of each sequence can only be handled by ISPM algorithms. Other modifications like removing items from a sequence or deletion of complete sequence are not considered by

ISPM algorithms. So, if a certain sequence does not have any newly arriving elements, this sequence will still stay in the database and undesirably contribute to count of frequent sequential patterns. As deletion of obsolete data from sequence database is not considered so many of these are uninteresting patterns. It is noted that users are usually more interested in recent data than old ones. Moreover, new arriving patterns may not be identified as frequent sequential patterns, due to the existence of old sequences. The obsolete sequential patterns that are not frequent recently, may be present in the reported results. The sequential patterns generated with these ISPM algorithms, hence, give incomplete or inefficient patterns.

1.6.2.2. Incremental Mining of Sequential Patterns from Progressive Databases

The solution is in form of incremental mining sequential pattern from progressive databases (Mhatre *et al.*, 2009). This type of mining not only adds new data to original database but also removes obsolete data from the database. Sequential Pattern mining over dynamically changing databases extracts sequences over various time periods, which keeps on progressing along with time. As a result the dynamic databases are called as progressive databases (Huang *et al.*, 2008). Hence, SPM could be classified into three categories based on the way it maintains the database:

- i. The SPM performed on static database finds sequential patterns in the database that do not change over time.
- ii. The ISPM done on an incremental database corresponds to the mining process where patterns are discovered without re-mining when new data arrives with time i.e., the database is incremental.
- iii. The **Progressive Sequential Pattern Mining (PSPM)** performed on progressive database is the one where new data is added into the database and obsolete data is removed simultaneously. Therefore, one can find the most up-to-date sequential patterns that are not influenced by obsolete data (Huang *et al.*, 2008). It is also termed as incremental mining of sequential patterns from progressive databases.

Thus, SPM with a static database and with an incremental database can be considered as the special cases of PSPM. When obsolete data is not deleted from database, the proposed

approach for PSPM is same as in case of ISPM. However, if database does not have new data and does not delete obsolete data, then PSPM approach can solve the static SPM problem. So, the algorithms that perform SPM on progressive database handle new sequence addition besides deletion of obsolete data. Figure 1.10 gives diagrammatic representation of three categories of SPM algorithms that are classified on the basis of how they mine patterns from databases that evolve with time and how they maintain them.

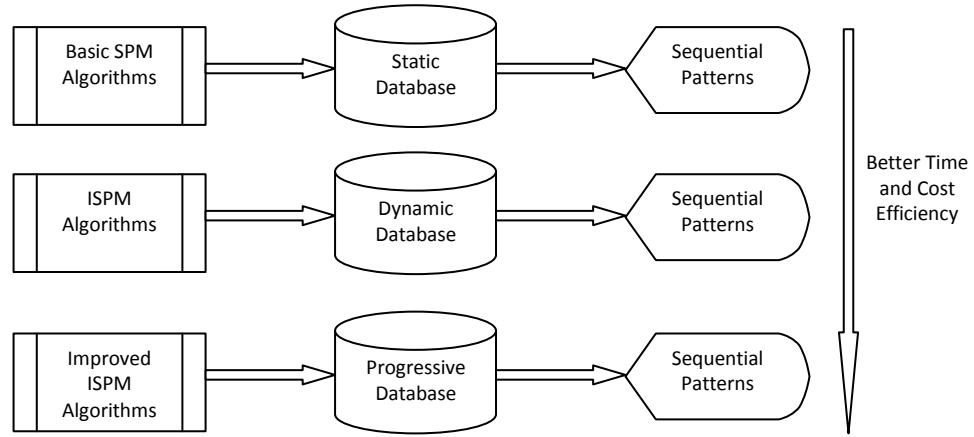


Figure 1.10 Evolution of SPM algorithms with time

The constraints can be incorporated in all these three categories of sequential pattern mining and the difference between them is summarized in Figure 1.11.

1.6.3. Fuzzy Sequential Pattern Mining

Business and real-time progressive databases comprise of numerical data that is time-stamped. Fuzzy set theory can be used to find relevant information from quantitative databases for human reasoning. When it is used to discover sequential patterns from databases then it is called as fuzzy sequential pattern mining. Within the perspective of large-scale database mining, few studies have been carried out to process this kind of data and most work is restricted to association rules. In recent times, the fuzzy set theory utilization has reduced bleak cuts over the period, and hence provides more relevant rules. However, to the best of our knowledge, there is no study made for fuzzy sequential pattern mining on progressive databases.

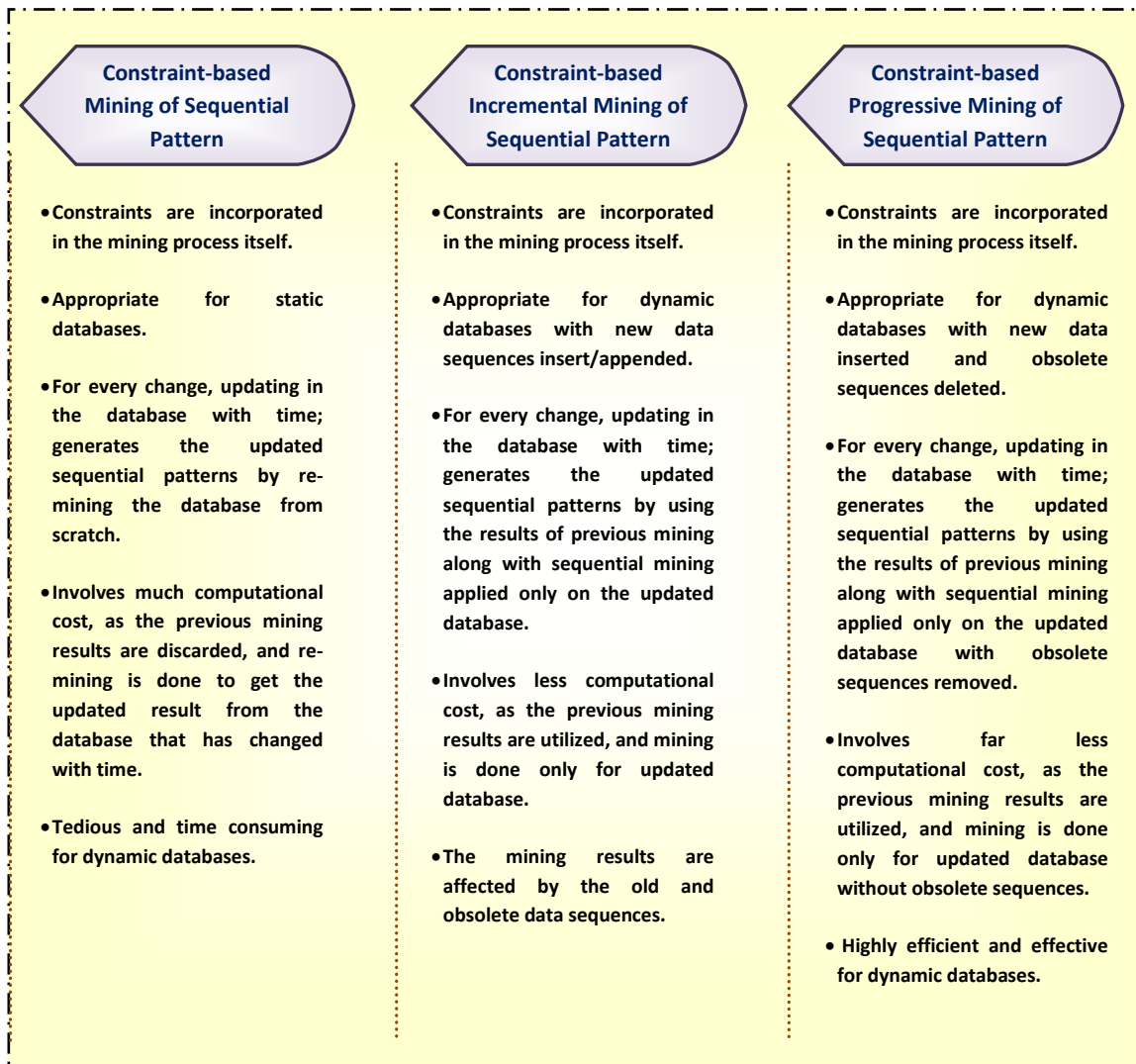


Figure 1.11 Comparative study of three methods of Sequential Pattern Mining

1.7. SUMMARY

Data mining is a significant field to discover interesting and useful knowledge from huge databases. Sequential pattern mining is a type of data mining with added timing order information with the extracted knowledge. There are three basic approaches of sequential pattern mining algorithms. However these methods, suffer from major limitations of not being efficient, effective, useful, certain and interesting from user view. They assume the data to be static, centralized and binary-valued.

Since they were defined in early 90's, mining of sequential patterns have received a great deal of attention. The efficiency of the algorithms was improved by using novel data structures, new models or managing the database in the main memory by the earlier

studies done in this area. However, the use of constraints in the mining process can generate more effective and interesting results.

In order to keep the current and up-to date sequential patterns, re-execution of the mining algorithm on the whole updated database is required with SPM which is time-consuming. Therefore, incremental mining of sequential patterns is proposed which utilizes previous discovered knowledge. It solves the maintenance problem of the dynamically updated database efficiently without re-mining from scratch. However, incremental mining of sequential patterns from progressive databases is more effective as it even takes care of old, irrelevant and obsolete data. Hence, this research study is based on constraint-based incremental mining of sequential patterns from progressive databases.

Fuzziness considered with incremental mining of sequential patterns from progressive databases give quantitative, understandable and user-friendly knowledge. For this reason, an algorithm having both the features of fuzzification and incremental mining to discover sequential patterns from progressive databases is proposed in this thesis.

Chapter 2

Literature Review²

2.1. INTRODUCTION

Sequential pattern mining is an active area of research since the time it has been introduced. A handful of researches are available in the literature for effective mining of sequential patterns from databases. In this study, the literature has been reviewed in four major directions. All four directions are associated with each other but at the same time they can be distinguished based on the nature of their study. The constraint based sequential pattern mining has been studied for different applications. Some of the recent researches are portrayed here in Section 2.2. A detailed discussion of the incremental mining algorithms for sequential patterns is done in Section 2.3. These algorithms have limitations which are addressed, in general, by other set of algorithms termed in thesis as Progressive sequential pattern mining algorithms or Incremental mining of sequential patterns from progressive databases. Such algorithms are reviewed in Section 2.4 and they formulate the basis of design and implementation of the proposed algorithm. To conclude, the fuzzy sequential mining algorithms are depicted in Section 2.5; as the scope of this study comprises to analyze the role of fuzziness along with incremental mining of sequential patterns.

2.2. CONSTRAINT-BASED SEQUENTIAL PATTERN MINING

The concept of incorporation of constraints with mining algorithms is as old as sequential pattern mining itself. There are many constraint based algorithms that have been proposed and their glimpse is given in Table 4.1, Chapter 4. They suffer from major limitation of not considering user interestingness and focus. It is vital for any application

² The major findings of this Chapter have been (i) published in *Intelligent Data Analysis*, 17(3), pp: 507-530, 2013 and (ii) communicated to *International Journal of Computer Systems Science and Engineering*.

that the algorithm should be specific to its requirement and give resultant patterns accordingly. In this section, some of the recent constraint based sequential pattern mining algorithms, and their distinguished features are presented.

A DELISP (Delimited sequential pattern) technique has been proposed by Lin and Lee (2005), based on pattern-growth methodology. DELISP has utilized bounded and windowed projection methods to diminish the size of the proposed databases. The valid time-gap valid subsequences have been maintained by bounded projection. On the other hand non-redundant subsequences satisfying the sliding time-window constraint are preserved by windowed projection. The delimited growth technique has directly discovered constraint-satisfactory patterns and increased the pace of the pattern growing process. It has been found that the DELISP has excellent scalability and performed better than the eminent GSP algorithm in discovering sequential patterns with time constraints.

Numerous applications necessitate approaches for temporal knowledge discovery. Few of those approaches deal with time constraints among events. But, such constraints have often been too crisp or required a very accurate assessment to evade flawed information. Hence, Fiot *et al.* (2007) presented an algorithm to manage the temporal constraints while data mining on the basis of sequence graphs. In addition, as these unstrained constraints may discover more generalized patterns, a temporal accuracy measure has been proposed for supporting the analysis of several mined patterns. For constraint based frequent-pattern mining, Pei *et al.* (2007) have designed a framework on the basis of a sequential pattern growth technique. Here, the constraints were effectively pushed deep into the sequential pattern mining under this proposed framework. Also, the framework has been extended to constraint-based structured pattern mining.

Chen *et al.* (2008) have presented robust approaches to cope with tough aggregate constraints. By a theoretical assessment of the tough aggregate constraints on the basis of the concept of total contribution of sequences, two typical types of constraints have been converted into the same form and thus processed in a consistent manner. Subsequently, a PTAC (sequential frequent Patterns mining with Tough Aggregate Constraints) algorithm has been proposed to diminish the cost of using tough aggregate constraints by integrating two efficient approaches. One shuns checking the data items one by one by

using the promising features revealed by some other items and validity of the respective prefix. The other evades building a superfluous projected database by successfully eliminating those bleak new patterns, which may otherwise function as new prefixes. Experimental studies performed on the synthetic datasets produced by the IBM sequence generator as well as a real dataset have revealed that the proposed algorithm has gained better performance in speed and space by means of these approaches.

Masseglia *et al.* (2009) have addressed the problem of mining sequential patterns by handling the time constraints as specified in the GSP algorithm. Sequential patterns were seen as temporal relationships between data present in the database where the considered data was simply the features of individuals or observations of individual behavior. The intent of generalized sequential patterns is to provide the end user with a more flexible handling of the transactions embedded in the database. A proficient GTC (Graph for Time Constraints) algorithm has been proposed to discover such patterns in giant databases. It was based on the idea that handling the time constraints in the initial phase of the data mining process would be highly advantageous. One of the most vital features of the proposed approach is that the handling of time constraint can be easily taken into consideration in conventional level-wise approaches because it is carried out prior to and independently from the counting step of a data sequence. Experiments have shown that the performance of proposed algorithm was substantially faster than the existing sequence mining algorithm.

Chen, Yen-Liang, *et al.* (2009) have proposed an algorithm for mining user specified sequential patterns from customers' purchasing data. Also, a pattern segmentation framework has been designed by using this algorithm to obtain significant information regarding customer purchasing behavior for managerial decision-making. Moreover, Jigyasa Bisaria *et al.* (2009a) have proposed a rough set perspective to the problem of constraint driven mining of sequential pattern. The search space of sequential patterns has been partitioned using indiscernibility relation from theory of rough sets and an algorithm has been developed, which allows pre-visualization of patterns and imposition of different kinds of constraints in the mining task. The C-Rough Set Partitioning algorithm was at least ten times faster than the naïve algorithm SPIRIT that was based on the diverse types of regular expression constraints.

Jigyasa Bisaria *et al.* (2009b) have investigated the sequential pattern mining problem by two perspectives, one the computational feature of the problem and the other was the integration and adjustability of time constraint. The search space of sequential patterns has been partitioned by indiscernibility relation from theory of rough sets and a robust algorithm has been proposed that allows pre-visualization of patterns and amendment of time constraints before the implementation of mining task. The Rough Set Partitioning algorithm was at least ten times faster than the naive time constraint based sequential pattern mining algorithm GSP. As well, an extra knowledge regarding the time interval of sequential patterns has been determined using the technique.

2.3. INCREMENTAL MINING OF SEQUENTIAL PATTERNS (ISPM)

A concise review of the research work related to the ISPM is presented here. Based on the typical naïve approach each of these algorithms implement and the characteristics they possess, we have classified them in four categories [Figure 2.1] as given in the following sub-sections.

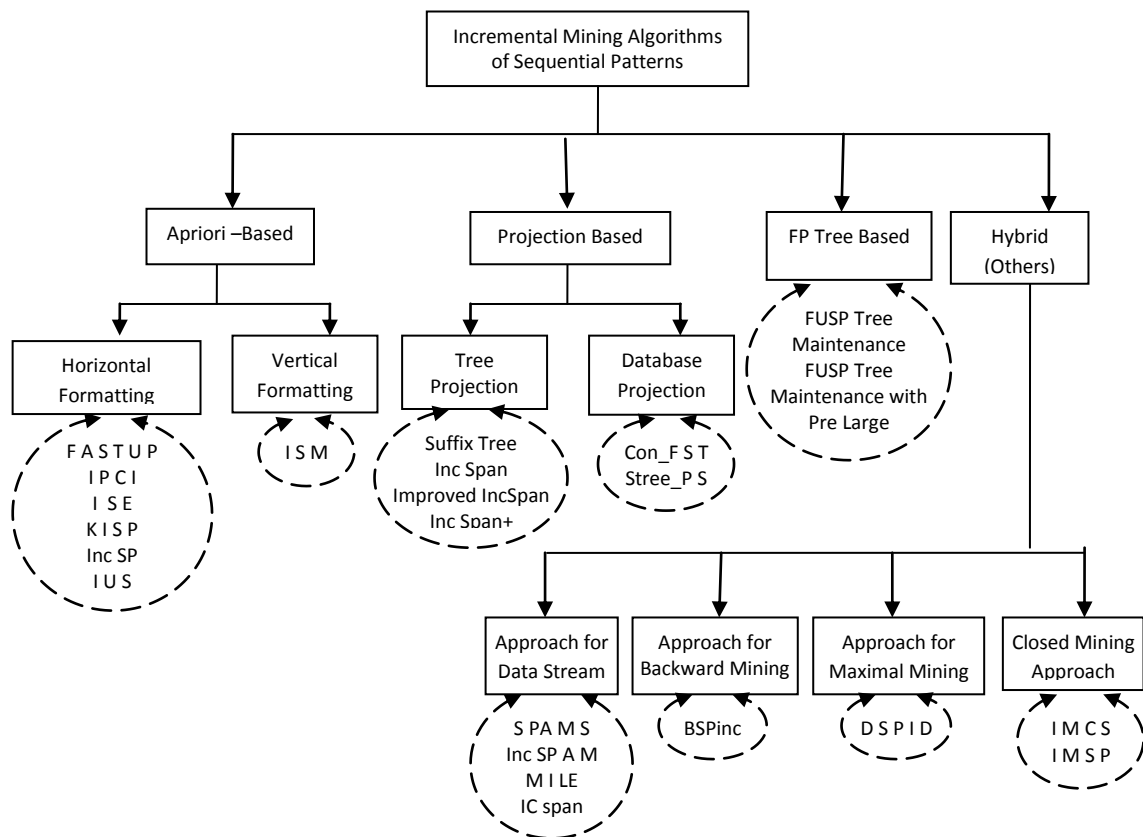


Figure 2.1 Taxonomy of Incremental Sequential Pattern Mining

2.3.1 Apriori-based ISPM Algorithms

The ISPM algorithms discussed in this section are those that have used Apriori-based (candidate generate-and-test) approach. Figure 2.2 gives the diagrammatic representation of these set of ISPM algorithms.

FastUP approach (Lin and Lee, 1998) is one among the initial work done in the field of the ISPM. It is based on GSP approach with improvements performed on candidate generation and support counting. The algorithm uses the pruning method before candidate generation and validation. This pruning approach uses previous mining results to get information about the sequence thresholds. Hence it can avoid generating some sequence based on their support. It further applies techniques of fast filtering during support counting and successive candidate sequence reduction. This improves overall space and time efficiency of the algorithm over conventional SPM algorithm.

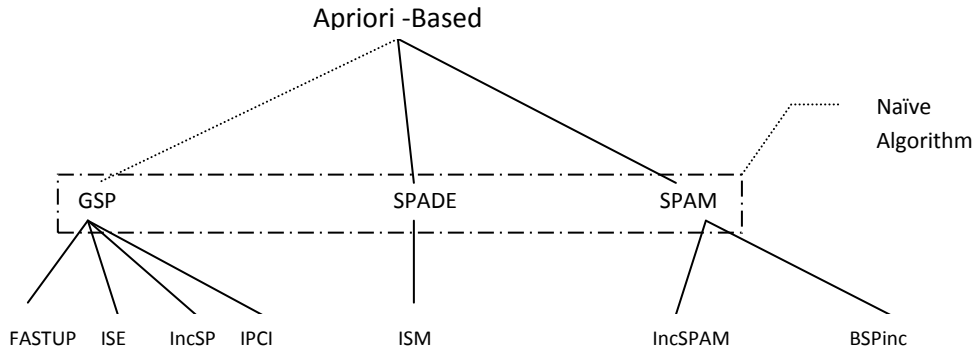


Figure 2.2 ISPM algorithms that use Apriori as naïve approach

Then, an algorithm with an approach similar to SPADE was proposed called ISM (Incremental Sequence Mining) (Parthasarathy *et al.*, 1999), which maintains sequence lattice of an old database. This lattice includes all the sequences that are frequent or are in the negative border. The negative border concept helps algorithm to maintain the updating of the database. ISM performs pruning of lattice and generates negative border sequences in one database scan initially. In next step, the new frequent sequences are generated using the SPADE approach. As the number of sequences in the negative border can be very huge, this approach could be expensive in terms of memory and time. However, it is unlikely to have sequences in the negative border to become frequent in the updated database if they have low support.

IUS (Incrementally Updating Sequences) (Zheng *et al.*, 2002) is an approach to reduce the computational cost by reusing frequent and negative border sequences in the

original database. A new threshold for negative border sequences called the minimum support of negative border sequences; min_nbd_sup is defined by the algorithm to control the memory and time consumed otherwise by these sequences [as observed in ISM algorithm]. Hence the frequent sequences whose support is less than min_sup and greater than min_nbd_sup are treated as negative border sequences. IUS generates candidates from original database by extending prefix and suffix of frequent sequences.

Another well recognized ISPM algorithm ISE (Incremental Sequence Extraction) (Masseglia *et al.*, 2003) minimizes computational effort by using old frequent sequences in an iterative manner and then applying optimization techniques. ISE algorithm though significantly outperforms the naïve approach, GSP algorithm but suffers from major limitations. These drawbacks include a large set of the candidate set slowing the test-phase and multiple scans of database due to its level-wise functioning. The algorithm is more expensive especially when there are long sequences in database.

IncSP (Incremental Sequential Pattern Update) (Lin and Lee, 2004a) algorithm is based on combined approaches of effective implicit merging, early candidate pruning and efficient separate counting. Implicit merging of updated sequences with previous useful sequences is done. Candidate pruning is performed by checking counts in increment database to generate fewer but better candidates. However, support counting of promising candidates over original database gives new updated patterns. IncSP not only updates the support of existing patterns but also find out new patterns from the updated database in continuous manner. IncSP was found to be fast and scalable as compared to GSP and ISM in most cases.

The concept of partitioning the search space using theory of rough sets is used by the ISPM algorithm, IPCI (Bisaria *et al.*, 2010). The knowledge of the minimal generators (one that do not have any sub patterns), frequent closed patterns (the one with no super pattern with the same support) and maximal patterns (sequence set with the highest cardinality) is used. All these types of patterns with their occurrence frequency are utilized to reduce the space of frequent sequential patterns in the updated database. IPCI looks for patterns in only the specified equivalence classes and require no buffering of patterns. This gives better time and space efficiency. The database scans are limited to one or two with no huge candidate generation.

The SPM process is typically iterative and interactive. As large or very few patterns could be mined from database depending on the minimum support (min_sup) value. So generally the user tries various min_sup values until the result is satisfactory. Most ISPM approaches are not designed to deal with repeated mining so each min_sup value invokes a re-mining from scratch. Lin and Lee (2004b) proposed a simple method, KISP (Knowledge base assisted Incremental Sequential Pattern) to improve efficiency of sequential pattern mining with changing supports. KISP utilizes information obtained from prior mining performed using an assumed least min_sup. A knowledge base is generated by the method to handle further queries about sequential patterns of various min_sup. When the results cannot be directly generated from the knowledge base, KISP can perform fast sequence mining by removing candidates before counting support. The algorithm can mine huge databases with any support value even with a small main memory. Two optimizations are done to speed up KISP. The direct generation of new candidates eliminate candidate searching in KB and concurrent support counting to reduce database scanning.

2.3.2. Projection-based ISPM Algorithms

ISPM algorithms that are based on tree projection approach (Figure 2.3) are discussed in this section. These algorithms do not generate candidate set followed by testing phase and hence avoid rescanning of the database, as is the case with the apriori-based methods.

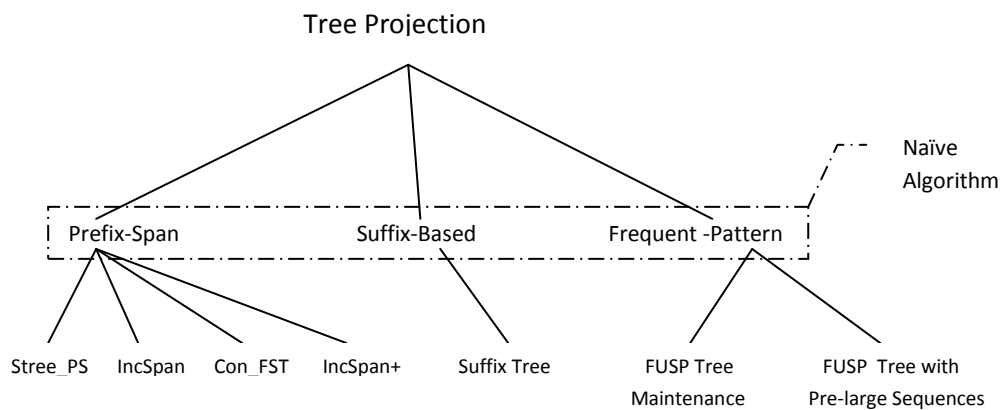


Figure 2.3 ISPM algorithms that use Tree Projection as naïve approach

Suffix Tree algorithm (Wang and Tan, 1996), a very preliminary concept that use projection approach was first introduced for ISPM. It maintains suffix tree to store the

frequent sequences of the database. The algorithm suggests that for every addition of the data sequence in the database, the tree could be modified by inserting a node appropriately. The algorithm suffers from major limitation that though it can work for multiple sequences of item sets, but is inefficient for single long sequence of items.

Then, a popular algorithm IncSpan (Cheng *et al.*, 2004), based on PrefixSpan approach was proposed that has interesting properties. IncSpan buffers both the frequent sequences and semi-frequent sequences, which are likely to become frequent in the next iteration, while mining the original database. Later, when mining the new version of database, only sequences with support over a certain threshold are likely to become frequent, out of un-buffered sequences. The better performance is achieved by the algorithm as it utilizes implicit merging, candidate pruning, and separate counting. However, it was found that IncSpan is not general enough to handle real life database evolution and its performance is far from optimal.

IncSpan algorithm fails to mine the complete set of sequential patterns from an updated database, as discovered by Nguyen, Son N., *et al.* (2005). They clarified the weakness of the algorithm, by proving the incorrectness of the basic properties in the IncSpan algorithm. Also, they rectified the observed shortcomings by giving their solution as Improved IncSpan.

Another better solution to avoid the drawback of IncSpan algorithm was given by Chen, Yue *et al.* (2007). IncSpan+, an ISPM algorithm constructs a prefix tree to represent the sequential patterns. Then it continuously scans the incremental item set in the updated database and maintains the tree structure accordingly. To improvise the performance and eliminate the search space, the techniques of width pruning and depth pruning are used by this algorithm.

Further enhancement in this direction was made by introducing a new storage structure called frequent sequence (FS) tree (Liu *et al.*, 2010). The data storage structure used by most of the ISPM algorithms cannot cope up with the change in the minimum support threshold given by the user. In such cases, the algorithms need to mine the database again. The FS tree stores all the sequential patterns that satisfy the minimum support threshold along with their support. An algorithm called Con_FST (Construction algorithm for Frequency Sequence Tree) can find all the sequential patterns with change

in support, without re-mining the database, using this tree. The path from the root node to any leaf node represent sequential pattern in the database. The root node of the FST stores the support threshold of the frequent sequence tree. To enhance the performance of the algorithm, the pruning strategy could be further applied. However, for high support threshold, the number of sequential pattern is limited and the storage requirement of projected database is less than that of the sequence tree.

In general, the ISPM algorithms that are based on tree projection, maintain structure to store frequent sequential patterns with their minimum support threshold. With every update in the original database, these algorithms re-construct the storage structure. Liu J. *et al.* (2011) proposed an algorithm Stree_PS (Sequence tree_PrefixSpan) to construct and maintain a storage structure called sequence tree. This tree is similar to prefix tree which stores all sequences in the original database i.e. all frequent and infrequent sequences along with their support. Hence the path from the root node to leaf node represents a sequence in the database. Stree_PS adds the incremental sequences to the sequence tree and dynamically maintain its structure. It uses depth first search strategy to traverse the sequence tree and find all sequential patterns in updated database. This avoids re-constructing of projected database with every updating of original database. Hence, the algorithm has better time and space efficiency for ISPM.

2.3.3. Frequent Pattern Tree-based Algorithms

Another set of ISPM algorithms, are those that use frequent pattern (FP) tree (different from frequent sequence tree) (Lin *et al.*, 2008b) as storage structure. A FP tree was extended to fast updated frequent pattern (FUFP) tree which further was upgraded to fast updating sequential pattern (FUSP) tree as storage structure (Figure 2.4).



Figure 2.4 Evolution of Frequent Pattern Tree for maintaining sequential patterns

The proposed FUSP tree structure extends the concept of FUFP – tree and IncSpan tree to store only frequent sequential patterns. This makes the FUSP tree compact and complete to avoid the rescanning of original database. An incremental FUSP –tree

maintenance algorithm (Lin *et al.*, 2008b) was developed to handle and maintain the FUSP – tree for recursive ISPM. This can handle new transactions from old customers and new customers. The FUSP – tree is build initially with original database. When new transactions are added, the algorithm processes them to maintain the FUSP tree. The entire FUSP tree is re-constructed in a batch manner when a sufficiently large number of transactions have been inserted. As the frequency of sequences is not maintained by FUSP-tree, the tree complexity is not optimal.

Hong, Tzung –Pei, *et al.* (2008) have proposed an algorithm, called Incrementally Fast Updated Sequential Pattern tree with pre-large sequences. It modifies the FUSP tree maintenance algorithm, based on pre-large concept (Hong *et al.*, 2001b). This reduces rescanning of original database, till a sufficiently large number of sequences are added. This modified FUSP tree structure is extended to FUSP-tree to store frequent sequential patterns. The FUSP-tree is initially built from the original database. The algorithm divides the newly added customer sequences into three parts as large, pre-large or small sequences of the original database and processes them accordingly. Then these sequences are sorted in descending order of their frequency. Finally they are scanned again to reconstruct the FUSP tree according to their sequence order. The algorithm also makes use of index table and header table to store the index and frequency of these sequences.

2.3.4. Hybrid/Other Algorithms

This section studies those ISPM algorithms that are based on combination of typical approaches [Figure 2.5, 2.6, 2.7, 2.8, and 2.9] and hence have interesting properties.

The algorithm DSPID (Discover Sequential Patterns in Incremental Database) (Lin *et al.*, 2007a) uses apriori and maximal sequence pattern approach [Figure 2.5] with some variations. It utilizes the information obtained from previous mining results unlike apriori-based approaches. It performs mining of maximal sequence patterns (compact representation of frequent sequential patterns) that reduce number of patterns, without losing any information. DSPID then builds a data model called Frequent Sequence Set (FSS), to represent original sequence database along with frequency. The algorithm compares the new sequence appended in the database with the existing sequences in the FSS. Every new sequence is processed as per the relationship; it has with the original

sequences as defined by the algorithm. It saves a lot of memory because no candidate sets are generated during mining process.

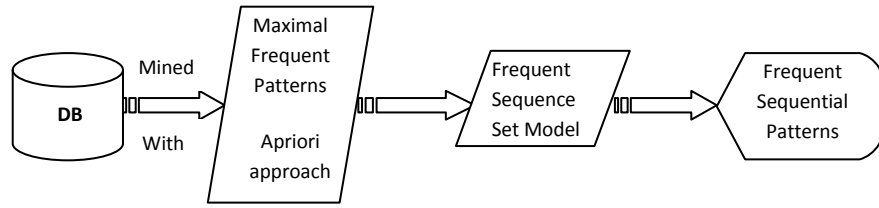


Figure 2.5 Combined approaches used by DSPID algorithm in its implementation

The algorithm IncSPAM (Ho *et al.*, 2006) is typically used for SPM over a stream of item set sequences. It utilizes the sliding window model over data stream, that is, transaction sensitive. The model receives the transactions from the data stream and uses an effective bit sequence representation of item set. This bit vector called Customer Bit Vector Array with Sliding Window (CBASW) can analyze each transaction effectively. The algorithm then makes use of weight function, to evaluate the importance of a sequence [Figure 2.6] and also ensure the correctness of the sliding window model. IncSPAM is a single pass algorithm with efficient execution time and memory requirement.

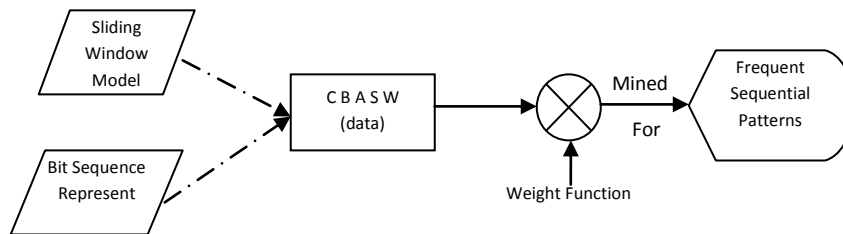


Figure 2.6 Combined approaches used by IncSPAM algorithm in its implementation

Another algorithm, MILE (Mining in multiple strEams) (Chen *et al.*, 2005) was proposed for mining sequential patterns in multiple data streams. The performance of the algorithm is improvised, by having the prior information regarding the data distribution in data streams. This avoids re-scanning of the database. MILE though based on PrefixSpan, append suffix with updated sequences to enhance the speed of discovering the new frequent sequential patterns. It emphasizes on generating sequential patterns from periodical data and uses a hash method to reduce time requirement. So the sequential patterns generated are not complete and accurate.

Chang, Lei *et al.* (2007) have examined the problem of mining closed sequential patterns incrementally. The idea is to mine patterns which have no super-patterns with same support in the database (Ren and Li, 2008; Li *et al.*, 2009). This reduces redundant patterns generation and subsequently storage requirement, as compared to minimum support value is low. A compact structure CSTree (closed sequence tree) was designed to keep closed sequential patterns of the database. They proposed an efficient algorithm IMCS, (Incremental Maintenance of Closed Sequential Pattern) that maintains the CSTree for sequence database, that is updated incrementally. The nice properties of CSTree are used to find obsolete sequences and to change the state of other sequences (represented as nodes), when the database is updated [Figure 2.7].



Figure 2.7 Combined approaches used by IMCS algorithm in its implementation

ICspan (Incremental mining of closed sequential patterns in multiple data streams) (Yang *et al.*, 2011) algorithm has also implemented closed sequential pattern mining to resolve the drawback of MILE algorithm. MILE algorithm with every new stream data input, mines only the new data rather than integrating new mining results with old ones. ICspan algorithm, hence proposed the use of sliding window model that sample the data from the input stream into units. It integrates the mining result obtained earlier with new result by implementing closed SPM in an incremental manner. So, the memory and time requirement of the algorithm is better than that of MILE.

BSPinc (Backward SPAM for incremental mining) algorithm is based on backward mining concept, for efficient incremental discovery of sequential patterns (Lin *et al.*, 2009b). Candidates are generated as $(k+1)$ sequences from k patterns, within the shrinking space of the projected database in the backward manner. The bitmapped representation is used for sequences. The patterns whose support count does not change in the updated database are mined with forward mining approach followed by apriori based method. BSPinc makes use of stable sequence property [Figure 2.8] for candidate pruning and hence substantially improves the most time consuming step of support counting.

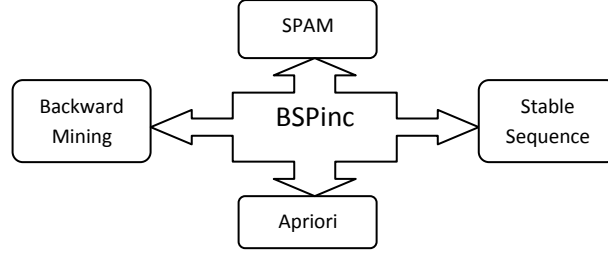


Figure 2.8 Combined approaches used by BSPinc algorithm in its implementation

There is another algorithm IMSP that performs only one database scan to find the frequent sequential patterns and generates no candidate sets. This is done by transforming the original sequence database into a closed frequency data model. It mines frequent sequences in form of maximal sequential pattern instead of mining the full set of frequent sequences. IMSP has compact searching space compared to other maximal pattern mining algorithm and adjustable minimum support threshold [Figure 2.9]. The approach requires less memory space.

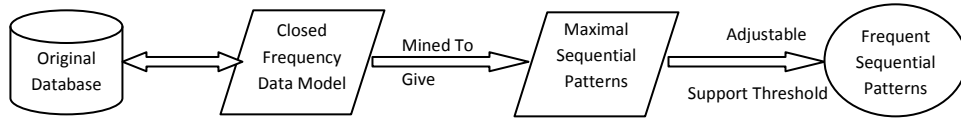


Figure 2.9 Combined approaches used by IMSP algorithm in its implementation

SPAMS (Sequential Patterns Automaton for Mining Streams), is an online algorithm proposed by Vincelas *et al.* (2009; 2010) for incremental sequential patterns mining in data streams. This algorithm uses SPA (Sequential Pattern Automaton), an automaton-based structure to maintain frequent sequential patterns. The automaton indexes all frequent sequential patterns from a data stream. The number of these patterns is smaller than set of otherwise generated frequent sequential patterns by two or more orders of magnitude. This helps to overcome problem of combinatorial high number of sequential patterns, usually in case of other algorithms for data streams. The proposed algorithm, SPAMS, is designed by considering the characteristics of data streams. The results are based on the users' specified threshold. Experimental studies performed, showed the relevance of the SPA data structure and the efficiency of the SPAMS algorithm, on various datasets.

The pre-large concept (Hong *et al.*, 2001b) is generally used to postpone original small sequences, becoming large directly and vice versa, when new transactions are

added. This basic idea was used for ISPM very recently. A lower support threshold and an upper support threshold are defined for the pre-large sequences. The upper support threshold is same as the minimum support threshold of conventional mining algorithms. The support ratio of a sequence must be larger than the upper support threshold to be considered large. A sequence with a support ratio below the lower threshold is treated as a small sequence. Hence, pre-large sequence acts like buffer to reduce the movement from large to small and small to large in incremental mining. Therefore when few new transactions are added in database, the original small sequences at most become pre-large and never large. This reduces rescanning of original database. Both the threshold – upper and lower along with size of the database define the safety bound for new transactions.

Zhang *et al.* (2008b) have used Galois lattice to discover web sequential access patterns by revealing the paths traversed via graphs. An incremental and robust mining algorithm has been proposed for increasing specific day' web log and changing transaction base. Peng and Liao (2007; 2009) have proposed two algorithms, IndividualMine and PropagatedMine. IndividualMine algorithm initially derives sequential patterns for each domain. Then the discovered patterns from sequence databases of multiple domains are combined to obtain candidate multi-domain sequential patterns. To shun the mining cost acquired by IndividualMine, they preferred the algorithm PropagatedMine that first mine sequential patterns from one sequence database. Then these mined sequential patterns are propagated to other sequence databases using lattice structure.

To accommodate problems like creation of many candidate sequences during the mining process, increase in computing power and runtime, Lin, Nancy P. *et al.* (2007b) have developed a Fast Accumulation Lattice (FAL) algorithm. The sequential database has been scanned only once by FAL to build lattice structure, which is a quasi-compressed data representation of original sequential database. Aseervatham *et al.* (2006) have proposed a bitSPADE algorithm, which combines best features of SPAM, one of the fastest algorithm, and SPADE, one of the most memory efficient algorithms. A pruning approach is used, which enables bitSPADE algorithm to achieve high performance.

The theoretical basis of the approaches used by various authors to investigate ISPM algorithms have been summarized in Table 2.1 for the quick reference.

Year	ISPM algorithm	Theoretical Basis
1996	Suffix tree [Wang et al.]	Use a suffix tree technique to store the frequent sequential patterns.
1998	FASTUP [Lin et al.]	Use a candidate generation and support counting approach with candidate pruning.
1999	ISM [Parthasarthy et al.]	SPADE based approach that maintains sequence lattice of frequent sequences and sequences in negative border.
2002	IUS [Zheng et al.]	Based on negative border approach and uses a minimum support of negative border sequences threshold along with min_sup.
2003	ISE [Massegli et al.]	Use a candidate-generate and test approach of Apriori based algorithms.
2003	KISP [Lin et al.]	Interactive mining to obtain results for different values of minimum support. It maintains knowledge database for this purpose.
2004	IncSpan [Cheng et al.]	PrefixSpan based approach and uses implicit merging, early candidate pruning and efficient separate counting.
2004	IncSP [Lin et al.]	Implicit merging, separate counting over appended sequences and early candidate pruning for incrementally updated patterns
2005	Improved IncSpan [Nguyen et al.]	Rectified the incorrectness of the basic properties of IncSpan.
2005	MILE [Chen et al.]	Based on prefixSpan but also append suffix to discover patterns in multiple data streams.
2006	2006 IncSPAM [Ho et al.]	Uses the sliding window model over data streams and bit vector representation for the customer sequential item sets.
2007	IncSpan+ [Chen et al.]	Maintains prefix tree to represent sequential patterns and implement width and depth pruning to eliminate search space.
2007	DSPID [Lin et al.]	Based on maximal sequential pattern mining and uses a breadth first technique to extract the frequent maximal patterns.
2007	IMCS [Chang et al.]	Mine closed sequential patterns incrementally using CSTree as the database is updated in an incremental manner.
2008	FUSP tree maintenance [Lin]	Use FUSP tree maintenance approach to store only frequent sequential patterns to build FUSP tree.
2008	FUSP tree with pre-large [Tzung-Pei]	Use FUSP tree maintenance approach to build FUSP tree along with the concept of pre-large sequences.
2008	[Zhang et al.]	Use Galois lattice to incrementally discover web sequential access pattern
2009	BSPinc [Lin et al.]	Backward mining strategy to eliminate the stable sequences whose support count does not change in the updated database.
2009	IndividualMine PropagatedMine [Peng]	Used for incrementally discover sequential patterns from multi-domain databases without using join operators.
2010	SPAMS [Vincaslas et al.]	Used for sequential patterns mining in data streams maintaining automation based structure for frequent sequential patterns.
2010	Stree_PS [Liu et al.]	Maintain sequence tree as the storage structure to keep all sequences of the original database along with their supports.
	IMSP [Hao et al.]	Generates a closed maximal frequency data model for the original sequence database instead of generating candidate sets.
2010	IPCI [Bisaria et al.]	Based on partitioning concept on theory of rough sets.
2011	Con_FST [Liu et al.]	Uses the frequent sequence tree as the storage structure to keep sequential patterns along with their supports.
2011	Algorithm using Pre-large sequences	Based on the pre-large sequence concept to avoid the rescanning of the database till a defined safety bound.
2011	ICspan [Yang et al.]	It uses sliding window model for data sampling and closed sequential pattern mining.

Table 2.1 Concepts used by ISPM algorithms in their implementation

2.3.5. Analysis of ISPM Algorithms

The study done on these ISPM algorithms helps us to draw important inferences and specify characteristic features of each one of these, as given in Table 2.2.

IncSpan+, BSPinc, SPAMS, IPCI, ISM are the algorithms whose performance depends on size of frequent patterns. They become more useful in application where the size of sequential pattern is small, like to study biological/ medical application where protein, DNA size is restricted or in case of study of disease with limited symptoms. FASTUP, IUS, IncSP, FUSP-tree based algorithms, pre-large concept based algorithms generate efficient sequential patterns based on size of new transaction sequences added in the database. Applications, like study and analysis of software version change monitoring, where the databases do not grow continuously, these algorithms can be of more useful.

Stree_PS, IMSP, Con_FST, DSPID, IncSPAM algorithm are more suitable for applications where there is huge generation of sequences and continuous updating in the database. These algorithms do not have huge memory requirement, no multiple scanning of database is performed, either less or no candidate generation; search space is less and they do not have any impact of size of frequent / new transaction patterns. Market basket analysis, stock market analysis, web usage analysis are the applications, where these algorithms can contribute better than the other ISPM algorithms.

SPAMS, Stree_PS, IMSP, Con_FST, Pre-large sequence are the algorithms whose performance is affected by value of minimum support threshold. So, these algorithms are not suitable for applications with changing and possibly high threshold values, like source code mining / text mining where the user is interested in studying the text patterns with different values or for the study of customer purchase behavior with varying attributes that can act as support values.

IUS, ISE, IncSP are the algorithms that require multiple scans of database and generate huge candidate sets. These algorithms are hence useful in applications where the numbers of database sequences are less like to study intrusion detection in a very small network. IncSpan, Improved IncSpan, IncSPAM, IncSpan+, DSPID, IMCS, FUSP-tree based algorithms, IMSP, IPCI are the algorithms, that would give fast mining result as they are based on partitioning the search space of the database. So they could be

I SPM	Huge Memory Overhead	Multiple Scans of Database	Huge Candidate Sets	Search Space Partition	Candidate Sequence Pruning	Depend on Size of Frequent Patterns	Depend on Size of New Transaction Sequences	Generate Complete Set of Sequence Patterns	Time Consuming Support Counting	Effect of Minimum Support Threshold	Use of Negative Border
Suffix Tree	X					X		X			
FASTUP							X	X			
ISM	X					X		X			X
IUS		X	X			X	X	X	X		X
ISE		X	X						X		
KISP				X				X		X	
IncSpan	X			X	X	X			X		
IncSP		X	X		X		X	X	X		
Improved IncSpan				X				X			
MILE	X						X				
ICspan				X				X			
IncSPAM				X				X			
IncSpan+				X	X	X		X			
DSPID				X				X			
IMCS	X			X		X	X	X			
FUSPTree				X		X	X	X			
FUSP Tree with Pre-				X		X	X	X			
BSPinc	X				X	X		X		X	
SPAMS	X					X				X	
Stree_PS				X				X		X	
IMSP				X				X		X	
IPCI				X		X		X			
Con_FST				X	X			X		X	
Pre-large sequence						X	X			X	

Table 2.2 Characteristics of various ISPM algorithms

significant when time efficiency is more important, like in case of scientific applications, weather forecasting etc. However, IncSpan, IMCS has huge memory requirement so they could be useful only if the tradeoff between time and memory is feasible.

It has also been realized, that the ISE, IncSpan, SPAMS, Prelarge sequences based algorithms do not generate complete set of frequent sequential patterns. The analysis inferences and decision made on the basis of mining results given by these algorithms for any application would be incorrect and incomplete. These algorithms may even lead to loss of important information.

In general, the above discussion only gives an idea that how different ISPM algorithms' characteristics could be exploited for the use in any application. However, the selection of a particular ISPM algorithm or a combination of these algorithms in real – time, depends on various factors like performance parameters, hardware configuration, time and space requirement parameters etc as required for any application.

2.4. ISPM FROM PROGRESSIVE DATABASES

These algorithms have been termed as PSPM algorithms (as discussed in Chapter 1) and briefly a review of these is given in Table 2.3. The three algorithms PISA, Weighted algorithm and CISpan has been discussed in detail with the illustrations, as we find these algorithms to address most of the limitation of the ISPM algorithms. However, other relevant work done for PSPM is also discussed in this section.

An algorithm (Hong *et al.*, 2001c) based on the concept of pre- large sequence is proposed to maintain databases with record deletion. Pre-large sequence as defined by a lower support threshold and an upper support threshold behave like a buffer (Hong *et al.*, 2001b). This prevents the transition of sequential patterns directly from large to small and vice versa. The algorithm maintains the large and pre-large sequences with their counts from original database. The deleted customer sequences are compared with them. If found they are categorized as small, pre-large and large. Accordingly the support ratio is changed, otherwise no action is required. The algorithm does not require re-scanning the original database until the deleted customer sequences exceeds a threshold. This threshold value depends on database size. This proposed approach is hence more efficient for large databases.

Zhang, M. *et al.* (2002) developed two algorithms for PSPM, based on static algorithms one based on GSP, and the other based on MFS. To handle the deletion and the insertion of data, they employed two pruning lemmas that reduce the number of candidates with iterations. This further improves the scanning time of the database. However, MFS+ and to some extent GSP+ have to mine sequential patterns separately from each sub database for all combinations of Period of Interest (POIs). Thus, the performance improvement of GSP+ and MFS+ over GSP and MFS is not much significant.

Year	ISPM algorithms on progressive databases	Theoretical Basis
2001	Pre-large sequence algorithm for record deletion [Wang et al.]	Use Pre-large sequences to maintain the database for customer sequences deletion.
2002	GSP+ [Zhang et al.]	Based on static database algorithm GSP and implement two pruning lemmas to reduce candidate in iterations.
2002	MFS+ [Zhang et al.]	Based on static database algorithm MFS and implement two pruning lemmas to reduce candidate in iterations.
2006	SPAM+ [Huang et al.]	Based on static database algorithm SPAM.
2006	MA_D [Ren and Zhou]	Maintains the sequential patterns obtain from the updating of database. It uses the information of previous mining results.
2006	IU_D [Ren and Zhou]	Maintains the incremental updating of the database with every deletion by using the information of previous mining results.
2008	PISA [Huang et al.]	Provides a general model to find the sequential pattern in defined period of interest (POI). It uses the sliding window concept. and maintains tree to store latest data sequence.
2008	CISpan [Yuan et al.]	Used for closed sequential pattern by building tree- like structure, incremental lattice to store frequent sequences.
2009	Weighted approach to extract sequential pattern from progressive database [Mhatre et al.]	It provides weights to individual items or timestamps of sequential pattern so that the importance of the extracted sequential pattern from the progressive database could be measured.
2009	FUSP tree update algorithm [Lin, 2009]	An FUSP tree is maintained to incrementally mine the progressive database for sequential patterns.

Table 2.3 Concepts used by ISPM algorithms on progressive databases in their implementation

MA_D (Maintenance Algorithm when Deleting some information) (Ren and Zhou, 2006a) algorithm uses the information that is obtained from previous mining processes. A new method was developed to generate the minimum sets of candidate sequences that can be used for further mining and gives better results. When some information is deleted from an original sequence database, MA_D algorithm generates frequent sequences from the updated database in an incremental manner for the same

minimum support threshold. This minimizes the overall runtime of the whole process. However, the multiple pass performed on the database is the limitation of the *MA_D* algorithm.

Ren and Zhou (2006b) have presented an algorithm, called *IU_D*, for mining frequent sequences when some transactions and/or data sequences are deleted from the original sequence database. This algorithm also uses information obtained during earlier mining process to reduce the cost of finding new sequential patterns in the updated database. Jen-Wei Huang *et al.* employed a more recent method SPAM, as it is known as a better algorithm than other static sequential pattern algorithms. They modified this algorithm to SPAM+ and experimentally proved that the proposed algorithm outperforms the static SPAM algorithm with a considerable factor. Figure 2.10 gives the naïve approaches implemented by some of these PSPM algorithms.

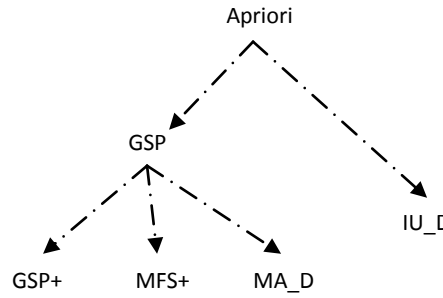


Figure 2.10 The naïve approach used by ISPM algorithms from progressive databases

FUSP-Tree Update Algorithm (Lin *et al.*, 2009a) uses FUSP- tree to maintain the incremental update in the sequences. A FUSP tree is built initially from original database. Later, tree is maintained with every update and deletion of customer sequences. The FUSP tree is based on concept of FUFPTree and IncSpan tree. The FUSP tree maintains the complete sequences that avoids and reduce the cost of rescanning the original database. This algorithm based on tree update give better result than the batch FUSP-tree algorithm (discussed in Section 2.3) to handle the deletion of customer sequences.

An efficient approach based on a general model of SPM in a progressive database, PISA (Progressive mIning of Sequential pAttern) (Huang *et al.*, 2008) is proposed. It is based on the concept of sliding window which is continuously progressing along with time. The window length is specified by the user and corresponds to the particular timestamp called as POI (Period of Interest). The concept behind Pisa algorithm is

explained with a sample database 'S' [Figure 2.11]. 'W' represents the POI with sequences of item/item set whose timestamp falls in the interval $[p, q]$, that is, ' $W_{p,q}$ ' is the window to represent time period starting with timestamp 'p' and end with timestamp 'q'. $T_1, T_2, T_3, T_4, \dots, T_{10}$ represent the timestamps and 101, 102, 103-----106 are the sequence identifiers. a, b, c, d, e, f, g and h are the items in the database. The minimum support threshold, min_sup be 0.3 and POI be five timestamps. The minimum frequency for a sequential pattern to be considered as frequent, for given example would be 5×0.3 ($W_{p,q} \times \text{min_sup}$) = 1.5. So for $W_{1,5}$ in the database 'S', 'ab' is discovered as frequent sequential pattern, whose occurrence frequency is 3 as in sequence 101, 103, 105. However, for all timestamps further till T_{10} , 'ab' is not a frequent pattern in any POI. The Pisa algorithm can delete this pattern at timestamp T_6 as obsolete. The algorithm incrementally discovers patterns in defined time of POI. It utilizes a PS_tree (progressive sequential tree) to maintain latest data sequences, discover complete set of up-to-date sequential patterns, and delete obsolete data. PS_tree stores element, timestamps of sequences of POIs and occurrence frequency of every sequential pattern at any time. The height of tree depends on the length of POI. Hence, memory requirement of Pisa is limited as compared to other methods. It requires only one scan of the candidate sequences maintained by the PS-tree to discover frequent sequential pattern at each timestamp.

Mhatre *et al.* (2009) contribute further to this work, by proposing a method to prioritize between any two patterns mined at any instance of time. The method presented by them considers time intervals between successive items, in addition to the item occurrence order. It is obvious, that an occurrence of a sequence with more time intervals in between its items would have lesser participation in determining the total support count for that sequence; as compared to an occurrence with no or lesser time intervals with the same POI. So this method provides a metric to measure the importance of the extracted sequential pattern from the progressive database. Using the same example database 'S', given in Figure 2.11, the PISA algorithm for POI, $W_{1,5}$ will give 'ab', 'bc' and 'df' as frequent sequential pattern with min_sup value as 0.3. On the contrary, this method would assign weight/priority to the item sets according to the time gaps within the items of the item sets. So, for our example, the sequence pattern 'bc' with a time gap of 2 timestamps in sequence 101 and 105 is not considered to be frequent sequential

pattern by the weighted approach. On the other hand, it considers 'ab' as a frequent sequential pattern as the time gap between 'a' and 'b' is 1 in sequence 101, 103 and 105. The weighted approach maintains Weighted M-ary tree to keep the information regarding, a) the item/itemset of the sequence, b) list of sequence identifiers, c) list of timestamps corresponding to each sequence, d) original timestamp that is the actual timestamp when the element occurs in a particular sequence and e) the time gap field to denote the number of inter-item time gaps in the pattern. The weighted algorithm insert, update and delete the nodes in the WM-ary tree as per the changes in the database. So, all candidate sequential patterns can be obtained from the tree for a particular POI.

Seq ID	T ₁	T ₂	T ₃	T ₄	T ₅	T ₆	T ₇	T ₈	T ₉	T ₁₀
101	a	b	d	ca	e		b	f	bd	
102	d	b	f	a				c	d	
103	fa	b	c	df	cb		h		bc	
104		g	af	c	b				bc	
105		a	b		cb	e		f	h	
106			d	b	f		eb		b	d

Figure 2.11 Sample database 'S' to illustrate the PISA and Weighted algorithm

CISpan, (Comprehensive Incremental Sequential PAtterN mining) (Yuan *et al.*, 2008) algorithm provides the solution in a divide and conquer manner. This separates the insertion and removal case apart from each other. The algorithm builds an incremental lattice, a tree-like data structure to store all frequent sequences appearing in the inserted sequences. However, removal of sequences is done directly by updating the intermediate mining data structure, namely prefix lattice, of original database. It is compact representation of all frequent sequence within the database. For incremental lattice, the number of nodes is confined only to items that are involved within the inserted

sequences. However, the removed sequences are handled by updating the original prefix lattice. CISpan algorithm then merges the original prefix lattice with the incremental lattice to give the updated lattice. A non- closed sequence elimination algorithm is applied to the updated lattice to give closed frequent sequences.

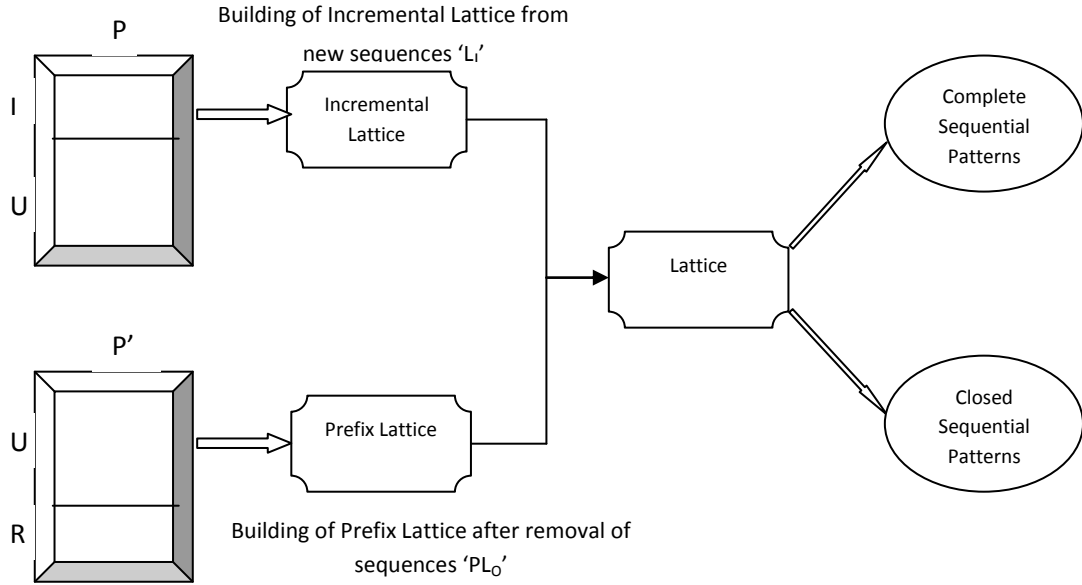


Figure 2.12 Diagrammatic representation giving overview of CISpan algorithm

The algorithm can also be used for generating complete frequent sequences. Let us illustrate this with an example. We use 'U' to represent the unchanged sequence set in our database, 'I' for the inserted sequence set and 'R' for the removed sequence set from the database [Figure 2.12]. 'P' is the updated database given by $P = U \cup I$ and P' is the database obtained after removal of sequences. The incremental lattice L_i is generated only from items in I. L_i calculate the support of sequences by using the database P. PL_o is the prefix lattice that is build from the original database. PL_o' is the prefix lattice derived from PL_o after the removal of sequences. The CISpan algorithm handle the insertion and removal of sequences in the database by performing the two activities of building L_i and PL_o' separately. It then merges these two lattices to give the updated lattice denoted as L. We can obtain frequent closed sequential pattern from this by applying closed sequential mining. The lattice L can also give complete frequent sequential pattern by performing basic SPM algorithms.

2.5 FUZZY SEQUENTIAL PATTERN MINING

Real world applications usually consist of quantitative data. Fuzzy set theory is being used for quantifying and reasoning. It partitions quantitative values into fuzzy sets which are further represented with linguistic terms. Several fuzzy learning algorithms have been designed. Many of these are based on finding fuzzy association rules or other fuzzy mining task (Kuok *et al.*, 1998; Hong *et al.*, 2003; Kaur and Gosain, 2009; Kuo *et al.*, 2009b; Dubois *et al.*, 2006; Mangalampall and Pudi, 2011; Kaur *et al.*, 2013). These association based approaches were not competent to consider the temporal ordering of data.

Their extension to fuzzy sequential pattern mining include studies (Hong *et al.*, 2001a; Chen *et al.*, 2001; Chen and Hu, 2002; Hu *et al.*, 2003; Fiot *et al.*, 2006a; Ouyang *et al.*, 2008; Hong *et al.*, 2010; Barik *et al.*, 2010; Hong *et al.*, 2010b; Cheng, 2011; Aggarwal and Bhatia, 2012). The fuzzy theory based algorithm proposed by Hong *et al.* (2001a) is based on combining the concepts of both fuzzy set and AprioriAll algorithm. Hu *et al.* (2003) have adopted a theoretical approach to present an algorithm without explaining the data structure used. The work is not even supported by experiments. SpeedyFuzzy, MiniFuzzy and TotallyFuzzy (Fiot *et al.*, 2006) algorithms were given to support different level of fuzzification to assess the frequency of the sequence. These methods can be generalized and improved as they primarily implement on static or incremental databases.

2.6 SUMMARY

An extensive study of different constraint-based and incremental mining algorithms to discover sequential patterns from databases has been done in this chapter. It is quite apparent that there is scope of selection of domain specific constraints that could be incorporated with algorithms. Such algorithm can further perform incremental mining from progressive databases to extract sequential patterns. The concept of fuzzy theory can also be incorporated in such mining algorithms to discover patterns that are more practical, simple and close to human reasoning. With the literature review, *problems for this study are identified as (i) need for an efficient algorithm for incremental mining of constraint based sequential patterns from progressive databases and (ii) applying fuzzification in incremental mining of sequential patterns from progressive databases.*

Chapter 3

Selection of Approach and Constraints for Proposed Algorithms³

3.1. INTRODUCTION

The goal of constraint-based sequential pattern mining is to determine entire set of sequential patterns that satisfy constraint(s). These constraints limit number and range of mined patterns that are utilized by sequential pattern mining algorithms to reduce intricacy (Antunes and Oliveira, 2004b). They are essentially incorporated in algorithm to make them application- specific. This would increase the usability of algorithm and interestingness of result for user. Hence it becomes important to opt for constraints based on their practical implementation and utilization for specific domain. The approach to design the algorithm is to be identified first followed by selection of constraints. In this chapter, the approach used in proposed algorithms is recognized and then the selection of constraints is made to suffice the study of customer value analysis for CRM. The technique to incorporate the selected constraints in the identified approach for the design and implementation of the algorithm is also discussed to conclude this chapter.

3.2. SELECTION OF APPROACH: PATTERN-GROWTH

On the basis of two prominent approaches, Apriori and Pattern-growth, quite a few algorithms have been proposed for successful sequential pattern mining (as discussed in Chapter 1 and 2). The apriori-like sequential pattern mining approach has come across a lot of difficulties such as,

- i. A large set of candidate sequences could be created in a giant sequence database. Since the apriori-based methods generate candidate sequences by making possible permutations of all items in a sequence, so a large set of candidate sequences are formulated even for moderate seed sequence.

³ The major findings of this Chapter have been published in International Journal of Computer Applications, 64(9), pp: 21-29, 2013.

- ii. Costly method as candidate sequence generation, candidate testing, candidate pruning and support counting is involved in all techniques based on apriori-based methods, irrespective of how it is implemented.
- iii. Requires scanning of database multiple times - to find a sequential pattern of length 'l', the database has to be scanned 'l' times.
- iv. An explosive number of candidates get generated during mining of long sequential patterns. Further to complicate the issue, the length of each candidate sequence is increased by one with each database scan.

In order to overcome such problems, a pattern growth approach was introduced to effectively discover sequential patterns. It uses divide and conquer method, instead of scanning the database repeatedly and generating-testing the candidate sequences. The approach is based on projection of sequence database into a few numbers of smaller databases with a set of patterns and then mine frequent patterns from each projected database. PrefixSpan (Pei *et al.*, 2004) is the most propitious pattern-growth approach, which is based on constructing the patterns recursively. It examines the database to identify frequent 1-sequences. Then, as per these frequent items, sequence database is projected into different groups, where each group is the projection of sequence database with respect to the parallel 1-sequence. For these projected databases, the PrefixSpan algorithm continues to find frequent 1-sequences to form frequent 2-sequences with the same respective prefix. Repetitively, the PrefixSpan algorithm produces a projected database for all frequent k-sequences to discover frequent (k+1)-sequences.

3.3. SELECTION OF CONSTRAINTS

As already discussed, researchers have found that the frequency is not the best measure that can be used to determine the significance of a pattern in different applications. When a single frequency constraint is employed, the conventional mining approaches normally produce a large number of patterns and rules, but majority of them are futile. Due to its ineptitude, the significance of constraint-based pattern mining has increased (Orlando *et al.*, 2004; Hu Y., 2007). Constraints can be evaluated and distinguished from diverse point of view. The literature presents various constraints (Pei *et al.*, 2007) that are utilized in sequential pattern mining (Chapter 1, Section 1.6.1).

In this study, the constraints are initially identified to discover most significant patterns in order to forecast customer purchasing behavior. Then these constraints are reviewed with the perspective of analyzing the customer value for CRM as discussed in Section 3.4. The customer purchasing behavior is not always static and change with time and purchasing cost. Accordingly length of the sequence or transaction length may also differ. With the aim of facing these challenges in the mining process, three new constraints namely, monetary, length and compactness, along with the frequency are selected to get incorporated into the identified pattern growth approach. Of these monetary and compactness constraints are derived from aggregate and duration constraints, respectively. These constraints are discussed in detail as below;

3.3.1. Monetary

In business point of view, there is always a need to consider the cost of an item. This is primarily because there are some patterns that are frequently occurring in the sequential database and are not providing much income. Moreover, the purchasing behavior of the user will be changed based on the cost of an item. For example, daily required items such as, milk, bread, butter, and cheese are frequently bought by customers, but valuable goods like gold and diamond are not purchased often. It has been though observed the latter items give better profit compared to frequently purchased items. To consider it, a monetary threshold ' T_m ' is specified by the user and the average cost of purchased items by the customer should be greater than or equal to T_m . It is in spirit with an aggregate constraint where the aggregate of items in a sequence should be above or below a given threshold value, represented as, $C_{agg} \equiv Agg(\alpha) \omega \Delta T$, where, $\omega \in \{\leq, \geq\}$, $Agg(\alpha)$ may be sum, average, max, min, standard deviation, and ΔT is a given integer.

3.3.2. Compactness

The customer's purchasing behavior can be varied over time in customer purchasing database. Hence, there is a necessity to consider the time, so that the decision makers who are attempting to find the user sequence behavior can develop better marketing and product strategies. The benefit of compactness is that, it allows drawing out sequential patterns that occur within a reasonable time span. It is based on duration constraint that

describes the time difference between the first and last items in a sequence should be greater than or less than a predefined threshold value. The duration constraint is represented as, $C_{dur} \equiv Dur(\alpha) \omega \Delta T$ where, $\omega \in \{\leq, \geq\}$ and ΔT is an integer value. A sequence α satisfies the duration constraint if and only if

$$\left| \left\{ \beta \in SDB \mid \exists 1 \leq i_1 < \dots < i_{len(\alpha)} \leq len(\beta) \text{ s.t. } \right. \right. \\ \left. \left. \begin{array}{l} \alpha[1] \subseteq \beta[i_1], \dots, \alpha[len(\alpha)] \subseteq \beta[i_{len(\alpha)}] \text{ and} \\ (\beta[i_{len(\alpha)}] \text{ time} - \beta[i_1] \text{ time}) \omega \Delta T \end{array} \right\} \right| \geq \text{min_sup}$$

where, β is a super sequence of α and belong to SDB, that is sequence database. $i_1, i_2, \dots, i_{len(\alpha)}$ are items of sequence α .

3.3.3. Length constraint

It details the requirement on length of the patterns, where it can be either the number of occurrences of items or the number of transactions. For instance, a user may desire to find only the long patterns (for example, the patterns consisting of at least 20 transactions) in market-basket analysis. Such a requirement can be expressed by a length constraint, which is defined as, $C_{len} \equiv (len(\alpha) \geq 20)$ Length constraint for sequential pattern mining is crucial in supermarket data to obtain the interesting patterns.

3.4. SELECTION OF DOMAIN-SPECIFIC CONSTRAINTS

The role of sequential pattern mining has become increasingly important for an organization that has large customer databases. Customer Relationship Management (CRM) systems are implemented to identify most profitable customers and manage relationship of company with them. CRM is a kind of business strategy of enterprise which centers on customers (Huang *et al.*, 2009). Intelligent data mining tools and techniques are used as backbone to the whole CRM initiative taken by the companies. Mining algorithms search data warehouse maintained by the companies and predict the hidden patterns and present them in form of a model. Strategic decisions about customers can be taken based on the outcomes of these models. To focus on specific category of customers for business profitability, constraints based mining algorithms are used. The identified constraints, monetary, compactness, length along with frequency need to be gauged for satisfying the objective of CRM as discussed in next section.

3.4.1 Customer Relationship Management

CRM is very important for any organization to know the relationship between customers and their organization (Collins, 2001; Kracklauer *et al.*, 2004). Any focused company need to identify the problems of customers and enhance their cohesion with them. CRM is the utilization of customer related information or knowledge to deliver relevant products or services to customers. Customer satisfaction helps in achieving the bottom-line business performance. This could be in the form of increased purchased volumes or repetitive purchases. It even helps in generation of new business in the form of references and prospect identification.

3.4.1.1. Related Work

The concept of customer relationship management has gained its importance since early 1980s. Many CRM-related researches have used data mining techniques to analyze and understand customer behavior and characteristics (Langley and Simon, 1995; Parvatiyar and Sheth, 2001; Lau *et al.*, 2003; Liao and Chen, 2004; Tsai and Lu, 2009; Ling *et al.*, 2010). To analyze and predict the customer purchasing behavior and accordingly formulate the marketing and sales strategies most researches utilize classification of customers in CRM domain. Classification tasks have been carried out for various purposes in CRM domain as given in Table 3.1. These studies use data mining techniques of neural networks, K-means, logistic regression for customer classification.

However, there are other data mining techniques like association rules and/or sequential patterns mining which are useful for the analysis of customer data. There has been some study done using them as well in CRM domain. The sequential pattern is used to predict future complaint (Lariviere and Poel, 2005) and to predict network banking churn (Chiang *et al.*, 2003). Market basket analysis to identify association rules (Aggarwal *et al.*, 2002) and use of both artificial neural networks and association rules to develop online personalized sales promotion (Changchien *et al.*, 2004) has been discussed. In recent time, the use of association rules to select variables for churn prediction (Tsai and Chen, 2010) and study of the sequential pattern mining for discovering customers' purchasing patterns over time (Chen and Hu, 2006; Cheng *et al.*, 2009) has been done. However, the customer segmentation/classification based on constraint incorporated sequential mining methods is underdeveloped. CRM through sequential pattern mining

can get more efficiency in acquiring new customers, increasing value of existing customers and retaining good customers.

Purpose	Technique	Contributors	Year
Customer knowledge management framework	K-means	Dennis <i>et al.</i>	2001
Customer targeting	Genetic algorithm and neural networks	Kim and Street	
Identifying slope in customer life cycle	Bayesian network classifier	Baesens <i>et al.</i>	2004
Formulate CRM strategies	Logistic regression	Hwang <i>et al.</i>	
Formulate CRM strategies	Decision tree	Yu <i>et al.</i>	2005
Churn prediction	Logistic regression and neural networks	Kim Y.S.	2006
Formulate CRM strategies	Decision tree	Kim <i>et al.</i>	
Churn prediction	Decision tree and logistic regression	Sinha and Zhao	2008
Churn prediction	Artificial neural networks	Tsai and Lu	2009
Formulate CRM strategies	Rough set theory	Hua, Jiang <i>et al.</i>	
Customer targeting	K-means	Ling, Xu <i>et al.</i>	2010

Table 3.1 Research studies for customer classification/churn prediction in CRM using data mining techniques

Although there has been a great deal of effort put forth on sequential pattern mining in recent years, its performance is still far from satisfactory. CRM applications based on sequential pattern mining techniques (Eirinaki and Vazirgiannis, 2003; Liu and Shih, 2005a) are based on assumption that the importance of each customer is the same. Many studies in CRM show that not all customers have the same contribution to business, and, to maximize business profit, it is necessary to evaluate customer value before the design of effective marketing strategies (Malhotra and Gosain, 2006; Luming *et al.*, 2007). Therefore, the consideration to customer value is essential in the process of sequential pattern discovery. This could be achieved by incorporation of additional constraints into mining process.

3.4.2 Customer Value Analysis Perspective

The selection of appropriate constraints for the CRM can be done with the help of customer value analysis. In general, any enterprise in order to develop new customers retains old customers. However not all customers have the same value for the enterprises. Usually, any enterprise would have limited resources, which could not be shared equally among all its customers. So, it is very important to perform customer value analysis. The evaluation of customer and classifying them accordingly can help the company to identify and focus on its real valuable customers. The main objectives of any enterprise for customer relationship management are customer value analysis and to improve customer retention rate.

To achieve these goals and further develop marketing strategies, enterprise can incorporate appropriate constraints in the sequential purchasing patterns developed over time. A related concept is that of cross selling which refers to a process where the enterprise sells the new products or services to the existing customers. The advantage of cross selling is that, for the existing customers, the enterprise can easily get relatively abundant information about them. In fact, the sequential purchasing patterns of the existing customer held by the enterprise helps in determining this customer's next purchasing behavior. Hence if the constraints are selected intelligently and applied on these sequential patterns then the customer value analysis and customer retention both the vital pillars of CRM can be accomplished. There are seventeen (17) most important parameters from the customer value analysis perspective which are identified (given in Figure 3.1). Let us now discuss how the selection of constraints could be made to meet particular objective/s of customer value.

3.4.2.1. To acquire new customers by studying the characteristics of existing customers over a particular duration: Compactness constraint

The major indexes of business development for any commercial area include the ability of acquiring new customers (He *et al.*, 2004). In fact, the CRM initiate with the activity of identifying prospects and converting them to customers. It is vital to understand the potential demand of product/s or items for a particular duration of period to acquire new customers. The division of the customers using certain product/s with the

aid of compactness constraint is the basis of enterprise's strategy formulation for selling effectively and market to new customers. A customer can be more interested in certain set of products during a festive season as compared to any other time in a year and so on. For example, by analyzing which items/products are purchased by the existing customers during Christmas occasion, strategy could be formulated to acquire new customers.

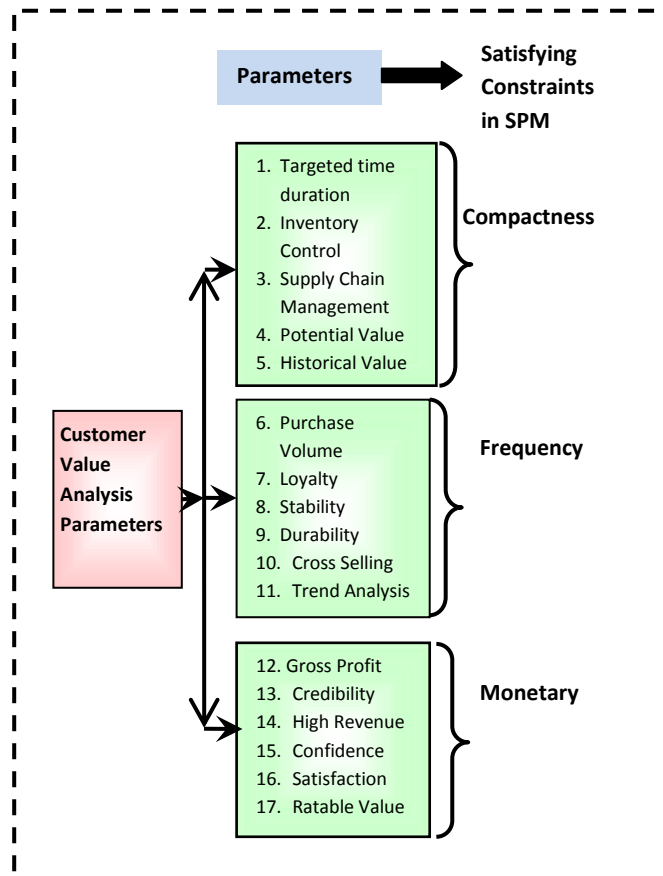


Figure 3.1 Identified '17' vital parameters for Customer Value Analysis

Compactness constraint incorporated in sequential pattern mining over the existing customers helps to identify these potential customer groups, increase the response rate of market activities and finally to enable the enterprise to improve and enhance its business. The important goals of CRM like maintaining appropriate inventory and manage supply chain (given in Figure 3.1) could be further improved using this constraint. As given in Figure 3.2, the historical value of existing customers could be low, medium or high, but the potential value of such customers is often very high as they have shown interest in the enterprise by making purchases during some time span. Customer value analysis using compactness constraint in the sequential mining process can not

only acquire new customers, but also cultivate the existing customers for further purchases.

3.4.2.2. To focus on existing customers and improve the retention rate: Frequency constraint

The other criteria to select the set of customers that any enterprise should pay attention for improving sales, could be to identify those customers making frequent purchases by analyzing the respective transaction databases. Customer retention is a process to keep old customers and to prevent them from losing. In this process, we could come across a huge set of customers again if the enterprise is doing well.

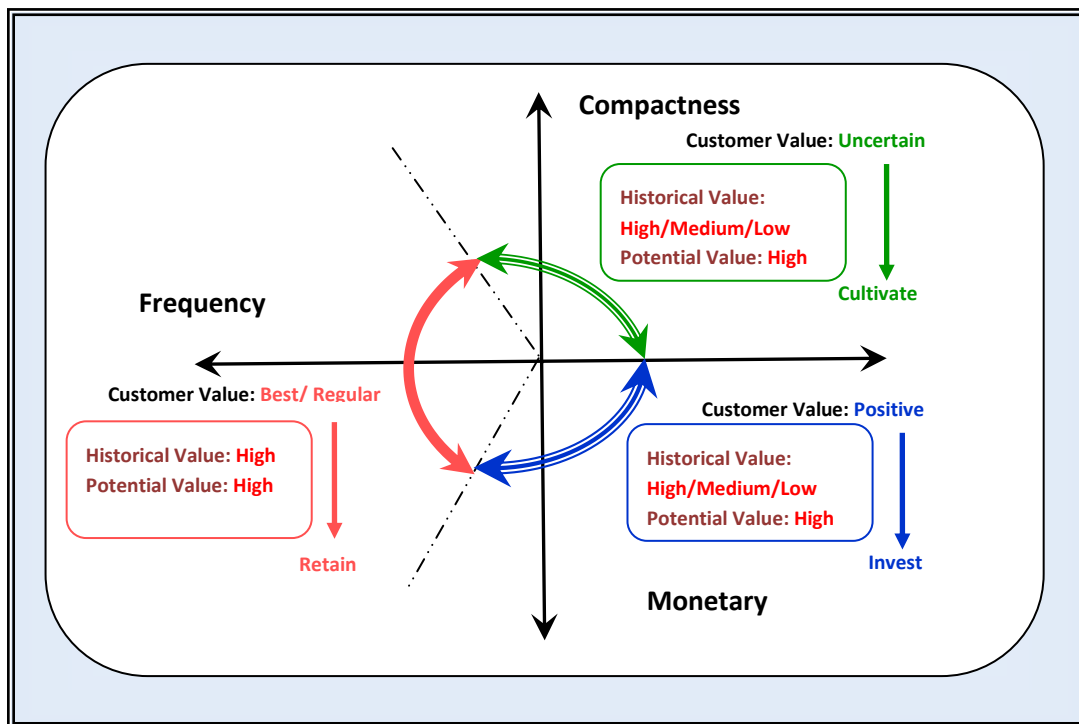


Figure 3.2 360° Customer Value Model based on Constraint based SPM

However, to maintain and improve further, the business organization should particularly focus on those set of customers who have been loyal and stable. A loyal customer is the one who is durable, that is, making continuous consumption in the enterprise. The stability of the customer can be gauged by his consumption cycle and frequency (as given in Figure 3.1). Such loyal, stable and durable customers are often found to be immobile, that are unaffected by other factors. If we incorporate frequency constraint in the sequential purchasing pattern of customer's transaction data, we could

discover the customers who purchase products/items of the enterprise at short intervals. Hence such customers satisfy the important parameters of stability, loyalty and durability of customer value. In fact, stability, durability and immobility are referred as three integral pillars of the customers' loyalty (Hosseini *et al.*, 2010) (Figure 3.3). Thus, the groups of frequent customers are the one that the business organization should concentrate on.

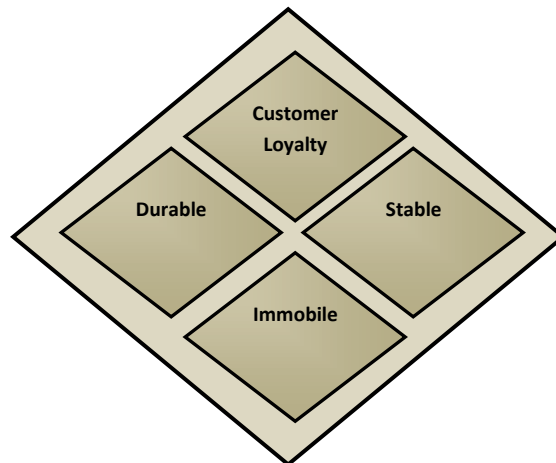


Figure 3.3 Three integral aspects of Customer loyalty

The mining of sequential patterns of customers can be used to analyze customers' purchasing trends. Trend analysis includes study of trending, cycle trend and changes, seasonal trend and changes, irregular random trend. It makes enterprise to understand customers' consumption trends over a period. This could be accomplished with the incorporation of frequency constraint in mining process. Cross selling by an enterprise could also be achieved by using frequency constraint within the mining process. This refers to a process where the enterprises sell the new products or provide services to the existing customers. The customer have benefit of getting more easy service of meeting his demand, enterprises make a profit because of growth of the sales amount. The constraint based mining can realize cross-selling to excavate effective marketing and realize buying recommendation and upgrade sale. As shown in Figure 3.4, the enterprise profit maintain a linear relationship with the customer retention, that is more the enterprise is able to sustain the existing customers, more regular profit it can generate. This is simply because the cost of acquiring new customers is saved in this case.

As given in Figure 3.2, the frequent customers have high potential and historical value so they are the best for the enterprise to retain for maintaining regular sales. For

any enterprise, it is much easier to get abundant information about these existing customers comparatively.

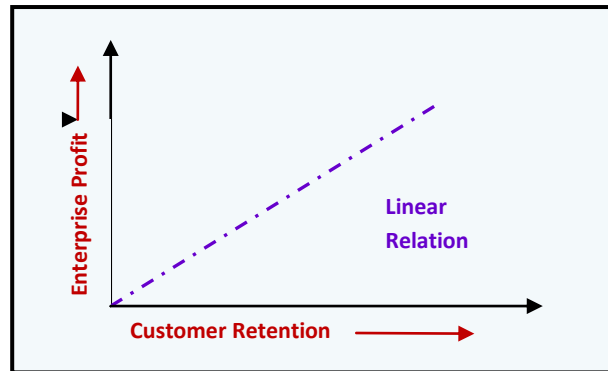


Figure 3.4 Relations between Customer Retention and Enterprise Profit

3.4.2.3. To preserve valuable customers: Monetary constraint

As the competition of each type of business, is getting fiercer and fiercer. The cost that enterprises to obtain the new customer is rising constantly. Hence, it is only obvious for the enterprises to pay more attention to retain existing customers. As discussed in previous section, for the high loyalty customer, enterprises make every effort to keep its good relation; however to the low loyalty customer, enterprises do not waste the financial resources to give up directly. However, using the frequency constraint alone cannot suffice the need of enterprise to obtain the data of valuable customer. For example, a customer giving a daily purchasing pattern of milk and bread from a particular grocery shop may be loyal. However, when it comes to analyze customer value in terms of generating high revenue and gross profit to the grocery shop, the customer who is making periodic purchase of complete household grocery from the shop could be more valuable.

Monetary denotes the amount of money the customer spent on the product/s or item/s. A customer with high monetary score means that it contributes higher revenue to the business. The customer profitability is also an indicator for the quantitative evaluation of customer purchases. The customer profitability analysis is intended to find the high-value customers, and to provide deeper and more detailed customer relationship management towards them. Such customers may have low or medium or high historical value but offer high potential value (as given in Figure 3.2). This is because such customers have high confidence, satisfaction and creditability towards the enterprise, that

make them invest high in the product purchases. All these are important parameters in the customer value analysis (Figure 3.1). This is the category of customers who have very positive opinion towards the enterprise's products. Due to these factors and the high ratable value of these customers, the enterprise should invest to preserve them. They can be obtained by implementing the monetary constraint in the sequential pattern mining process. There is no doubt, as the customer satisfaction towards the enterprises' product purchased by him on a certain price increases, the credibility of the enterprise grows exponentially, due to this particular customer as well as because of the word of mouth spread by him (Figure 3.5).

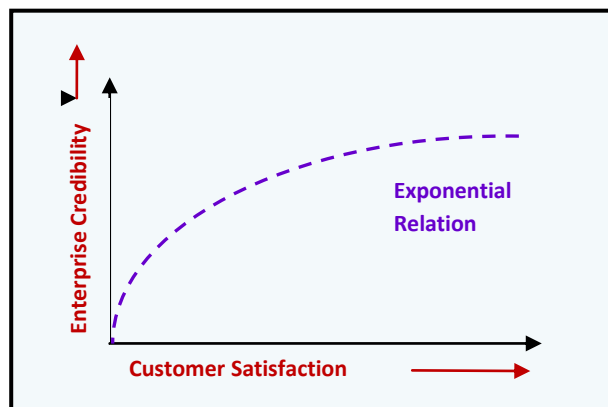


Figure 3.5 Relations between Customer Satisfaction and Enterprise Credibility

Enterprise can handle the customer's demand, adopt the individualized marketing tactics, and offer service as per customer classification, providing improved customer's satisfaction for all kinds of customers using constraint based sequential mining. It helps in evaluating the potential value of customers, and accordingly enables enterprises to make more efficient use of its limited marketing resources. For low-value customers, since their ratable and potential values are lower, so limited business resources should be used for them. The potential customers (Figure 3.2), one giving high revenue according to the business resources utilized but total purchase volume is smaller, they have higher index of credit and satisfaction with the enterprise. For such customers, companies should implement a strategy to attract them to buy, they are much easy to upgrade to loyal customers with certain investment, and tap their potential to establish long-term friendly relations. Such customers may have smaller proportion, but for

enterprises, they are very important for enterprises' large profits. If the customer has low potential value its better for an enterprise to reduce its investment in such customers.

3.5. TECHNIQUE TO APPLY CONSTRAINTS IN MINING PROCESS

With the identification of constraints for customer value analysis for CRM, a very important next step is how and when to apply them in the mining process. It could be implemented by two identified approaches [Figure 3.6] and explained as follows:

- i. Perform customer segmentation based on constraints and then discovering patterns from valuable customer group (Miglautsch, 2002; Liu and Shih, 2005b; Liu *et al.*, 2009).
- ii. Incorporating constraints in the sequential mining process and segmenting the customers based on the sequential patterns discovered. This is more complicated as it involves the redesigning of the sequential pattern mining technique to incorporate the constraints.

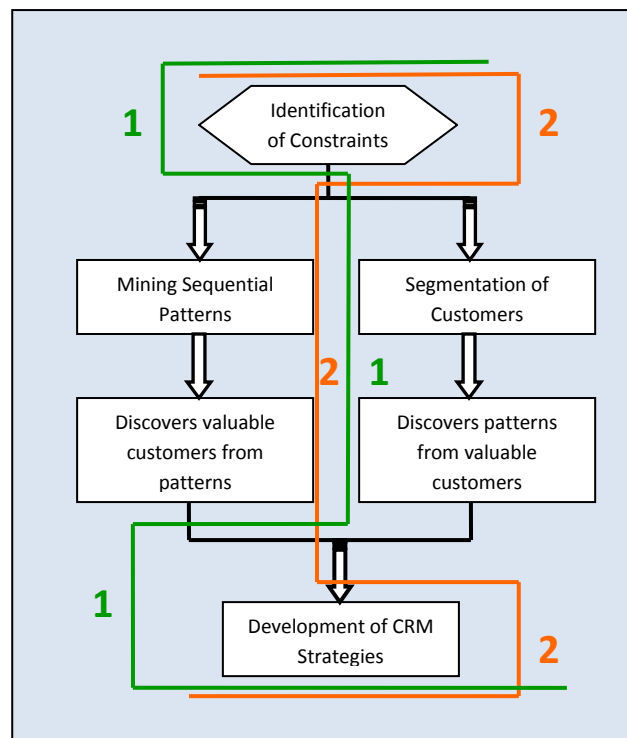


Figure 3.6 Approaches to incorporate constraints for CRM strategies

The first approach has the following shortcomings:

- i. This would require the fair knowledge of both the segmentation/clustering and sequential pattern mining techniques.

- ii. It works well when we have patterns of single item or items with similar constraint value to be used for customer segmentation.
- iii. Loss of useful and interesting patterns during constraint based customer segmentation. For example, an interesting purchasing pattern of egg and milk by a group of customers would be overlooked for a large grocery shop in compared to a pattern of expensive cosmetic item-sets, if we have to segment the customers in terms of purchase worth.

On the other hand the second approach requires a one-time effort. We need to redesign the sequential pattern mining approach using constraints for the first time and then it could be used to get the useful patterns. These discovered constraints based patterns could be analyzed to segment the customers. By setting adequate thresholds of the business specific constraints the mining algorithms search complete customer sequence database and filter out worthless patterns.

Recently, the concept of constraints has been integrated into the mining of valuable sequential patterns (Chen and Hu, 2006; Chen *et al.*, 2009; Liu, 2009; Cheng and Chen, 2009; Jiang and Tuzhilin, 2006). Most of these are based on using the first approach. However, our proposed method (as given in Figure 3.7) suggests the use of three phases for the second approach.

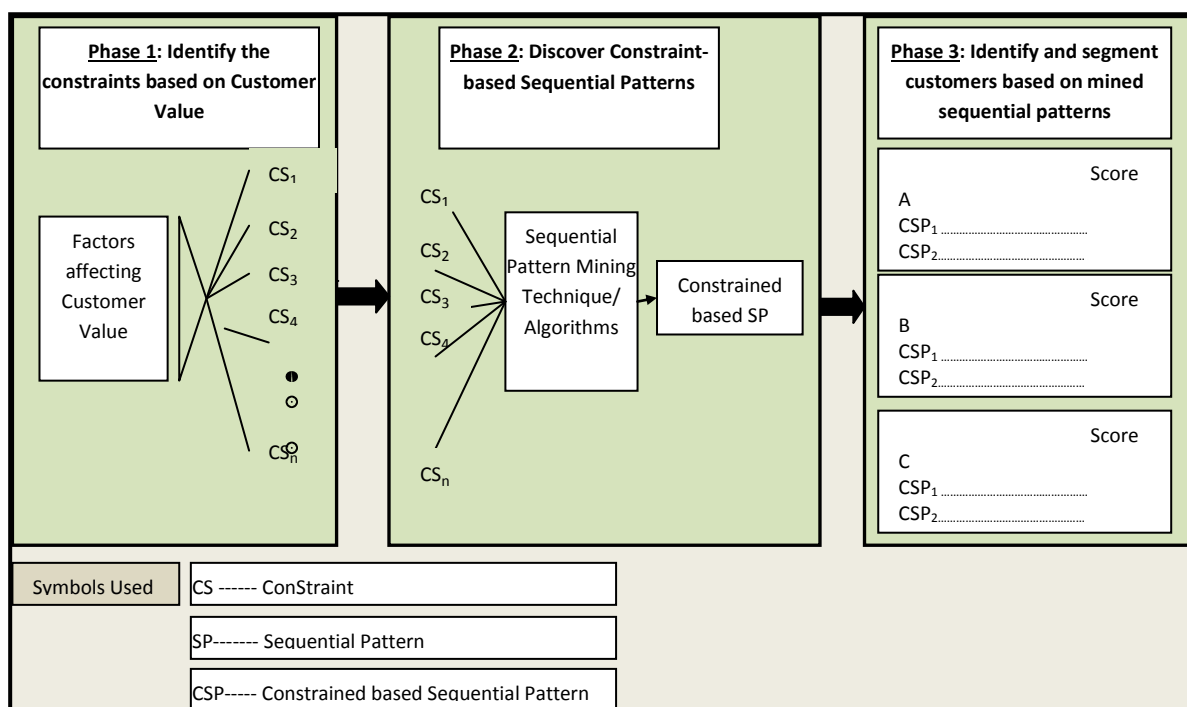


Figure 3.7 Three phases of proposed approach of using constraints for CRM domain

Phase 1 identifies the constraints as per the parameters of customer value. In next Phase, selected constraints in sequential pattern mining are applied on customers' transaction database. In last Phase, customers are identified and segmented based on the discovered patterns. This could even be done by giving appropriate score to each customer based on the number of constraints satisfied by him in each purchased sequences. There is some work done for last two phases like for phase 2, (Chen and Hu, 2006) and phase 3 (Chen *et al.*, 2009). However as far as our knowledge no study is made on how the constraints could be appropriately be selected from the customer value analysis perspective, which has been discussed in Section 3.4.

3.6. SUMMARY

Customer relationship management is vital to survive in the competitive marketplace today. The more effectively enterprises can use the information about its customers to meet their needs the more profitable enterprises will be. Sequential pattern mining is used to identify useful and interesting patterns in the customer sequence databases. However, there is a severe lack of research that takes a broader strategic focus on the relationship between CRM and sequential pattern mining. Of the few study made in this area, there is hardly any work for the realization of constraints from the customer value analysis. There are certain parameters that should be satisfied for effective implementation of customer value analysis.

Monetary, compactness, length and frequency are constraints that are useful to study customer purchasing behavior. However, for customer value analysis, length constraint seems to be insignificant. The use of these identified constraints in the sequential pattern mining applied on customers' transaction database can help to improve customer retention, acquire new customers and preserve the valuable customers as well. Two approaches to incorporate the constraints and segment the customers based on their purchasing patterns are suggested for CRM. But, the use of one approach over the other is recommended due to the shortcomings of the later approach. The selected pattern-growth approach, identified constraints and technique to apply these constraints in this Chapter are used to design and implement the proposed algorithms in Chapter 5 and 6.

Chapter 4

Proposed COPRE Framework to Incorporate Constraints in SPM Algorithm⁴

4.1. INTRODUCTION

The mining of sequential patterns addresses the problem of discovering the existent frequent sequences in a given database (Frawley *et al*, 1992). The study done here is to exploit pattern constraints to prune the computational cost and enhance the performance of the system that is commensurate with the level of user focus and requirement. The concept of the inclusive property to incorporate more than one constraints of different types is introduced. The property is used to give CONstraint based PREFIX (COPRE) framework based on pattern growth approach. The framework can be used to give reduced combination of sequential patterns and is used for implementation of proposed algorithms for constraint based mining of sequential patterns in Chapter 5 and incremental mining of constraint sequential patterns in Chapter 6.

4.2. NEED FOR A FRAMEWORK TO INCORPORATE CONSTRAINTS IN SPM

It is obvious that the constraints can be verified in a post-processing or pre-processing step, after discovering all patterns exceeding a given minimum support threshold. However, it has been shown that pushing constraints deep into the mining process can reduce processing time by more than an order of magnitude (Massegia, F, *et al.*, 1998, Garofalakis MN *et al*, 1999). An apriori-based improved algorithm, GSP (Generalized Sequential Patterns) to include time constraints, sliding time window, and user defined taxonomy was presented by Srikant and Agrawal (1996b).

SPIRIT (Sequential Pattern mining with Regular expression constraints) (Garofalakis *et al*, 1999) is a popular method of mining user specified sequential patterns

⁴ The major findings of this Chapter have been communicated to International Journal of Computer Systems Science and Engineering.

by using regular expression constraints. It proposes a regular expression and a series of operators that can read and interpret the regular expression of constraints. The concept of weighted sequential pattern mining was introduced (Yun and Leggett, 2006, Yun, 2008, Chang, 2011) which require to pre-assign weights based on application. For example, the quantity and price of an item being sold in a retail store's database are considered as its weight. On the other hand, some algorithms find the weights of data elements from the relationship or status of data elements without any pre-assigned weights. The main problem in weight based sequential pattern mining is that the downward closure property is broken. A sequence with lower weight cannot be pruned because super sequences may have higher weight. WSpan (Yun and Leggett, 2006) algorithm pushes weight constraints into the sequential pattern growth algorithm while keeping the downward closure property. The algorithm defines a weight range and items are given different weights within the weight range to reflect characteristics of the real dataset. The weights of sequences are calculated accordingly.

A framework of mining time-interval weighted sequential (TiWS) patterns (Chang, 2011) has been proposed. It contains a basic approach to get a time-interval weight of a sequence which is based on time period and strength of each pair of data elements in the sequence. Currently, several algorithms have been proposed for mining sequential patterns with constraints (Srikant and Agrawal, 1996b; Ng *et al.*, 1998; Masseglia *et al.*, 1998; Guha *et al.*, 1999; Gupta *et al.*, 1999; Garofalakis *et al.*, 1999; Zaki, 2000; Pei *et al.*, 2002; Garofalakis *et al.*, 2002; Antunes and Oliveira, 2003; Albert-Lorincz and Boulicaut, 2003; Leleu *et al.*, 2003; Antunes and Oliveira, 2004b; Orlando *et al.*, 2004; Lin *et al.*, 2005; Yun and Legget, 2006; Lin *et al.*, 2006; Pei *et al.*, 2007; Chen *et al.*, 2008; Lin *et al.*, 2008; Sakurai *et al.*, 2008; Yun, 2008; Masseglia *et al.*, 2009; Bisaria *et al.*, 2009(a and b); Kuo *et al.*, 2009a; Wu *et al.*, 2010; Zhou and Xu, 2010; Chang, 2011). These research studies are inadequate as they handle limited constraints. Some of these algorithms have been discussed in Chapter 2.

However, the following problems remain with the studies done related with constraint-based sequential pattern mining:

- i. Many application-specific and practical constraints are not covered.

- ii. There lacks a systematic method to push various constraints into the mining process.
- iii. Most of the algorithms focus on one or two type of constraint.
- iv. The method is not generalized and usually application specific.
- v. The approach used by most of the algorithms is exclusive, that is can give sequential patterns mining results based on one constraint at a time.

4.3. EXCLUSIVE AND INCLUSIVE PROPERTY

As being discussed, most of the constraint-based sequential pattern mining algorithms can give mining results considering one constraint at a time. The algorithms available in the literature like Generalized Prefix-Span, GTC, CCSM, DELISP, METISP [refer Table 4.1 for details] are generating sequential patterns by incorporating one constraint at a time. Many of these algorithms are based on incorporating time constraints in one or the other naïve approach of sequential pattern mining. We are henceforth using a term “**exclusive**” for such algorithms that apply only one constraint or another at a time and generate target database accordingly [Figure 4.1].

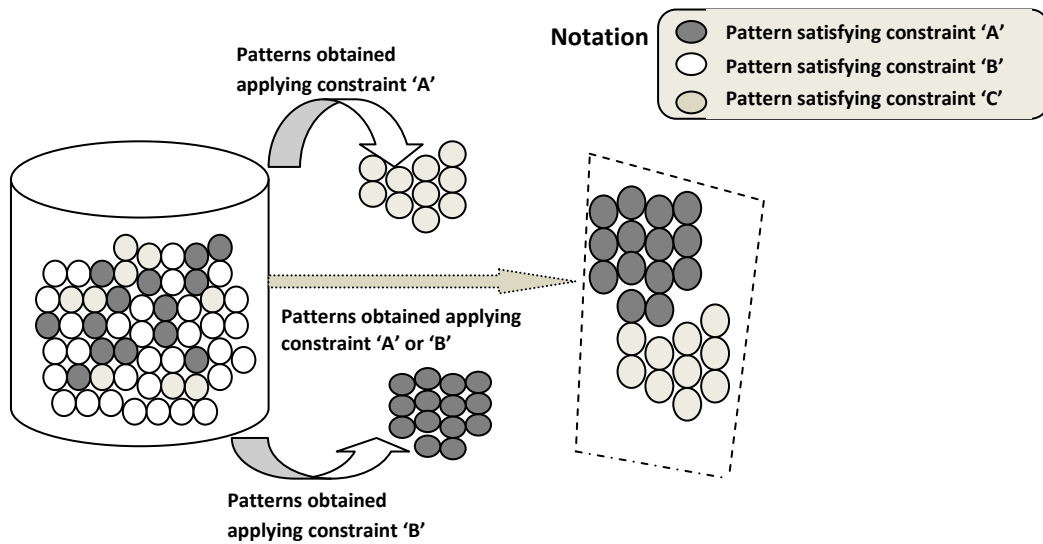


Figure 4.1 Constraint based sequential pattern mining in an ‘Exclusive’ manner

However, in real time applications, we have seen it is often required to incorporate more than one constraint in the sequential pattern mining to get the desired results. There is a regular expression constraint which helps in achieving this. Some of the

Algorithm	Naïve Approach	Constraint	Author	Year	Property
GSP	Apriori-based	Time, User defined taxonomy	Srikant R <i>et al</i>	1996	Exclusive
PSP	Apriori-based	Time	Masseglia F <i>et al</i>	1998	Exclusive
CAP	Apriori-based	Aggregate	Ng <i>et al</i>	1998	Exclusive
SPIRIT	Apriori-based	Regular Expression	Garofalakis M <i>et al</i>	1999	Inclusive, (with limitation)
cSPADE	Apriori-based (SPADE)	Time, Length and Item	Zaki MJ	2000	Inclusive (with limitation)
Generalized PrefixSpan	Pattern growth (PrefixSpan)	Gap	Antunes and Oliveira	2003	Exclusive
RE-Hackle	Apriori-based	Regular Expression	Albert-Lorincz H <i>et al</i>	2003	Inclusive
GTC	Apriori-based (Preprocessing of PSP and GSP)	Time	Masseglia <i>et al</i>	2004	Exclusive
CCSM	Apriori-based (GSP)	Maximum Gap	Salvatore Orlando <i>et al</i>	2004	Exclusive
DELISP	Pattern growth	Gap and Sliding Window	Lin <i>et al</i>	2005	Exclusive
WSpan	Pattern growth	Minimum-Maximum Weight	Yun U <i>et al</i>	2006	Exclusive
Prefix-growth	Pattern growth	Time (excluding sliding window) and regular expression	Pei J <i>et al</i>	2007	Inclusive
Sequential Interestingness	Apriori- based	Interestingness	Sakurai S <i>et al</i>	2008	Exclusive
WLPMiner	Pattern growth	Weight and Length decreasing support	Yun Unil	2008	Inclusive
METISP	Pattern growth	Time	Lin <i>et al</i>	2008	Inclusive only for time constraints, otherwise exclusive
PTAC	Pattern-growth	Aggregate	Chen E <i>et al</i>	2008	Exclusive
Rough Set Partitioning Algorithm	Apriori -based	Time	Bisaria J <i>et al</i>	2009	Inclusive only for time constraints, otherwise exclusive
MSPVF	Apriori-based	Time	Wu Di <i>et al</i>	2010	Exclusive
TiWS	Pattern growth	Time Interval Weighted	Chang <i>et al</i>	2011	Exclusive

Table 4.1 Constraint-based Sequential pattern mining algorithms classified as per Exclusive/ Inclusive property

proposed algorithms like Re-HACKLE, SPIRIT (Garofalakis *et al*, 1999, Albert-Lorincz and Boulicaut, 2003) are based on regular expression where the expression is formulated based on user defined constraints. But these algorithms allow the user to express the specific family of sequential patterns in which they are interested. At the same time the algorithm requires the user to specify the explicit ordering of the constraints so that it could be expressed in regular expression language using operator. In cases where the user specified constraints are from different family or types and the user is not technically competent to specify the ordering and formulate the regular expression then the constraint based sequential pattern mining is a challenge.

We have termed such a constraint based sequential pattern mining as “**inclusive**” that incorporated more than one constraint belonging to different family altogether [Figure 4.2]. There is a lot of work that is required to be done in this area. A classification of the constraint based sequential pattern mining algorithms based on inclusive and exclusive property in given in Table 4.1.

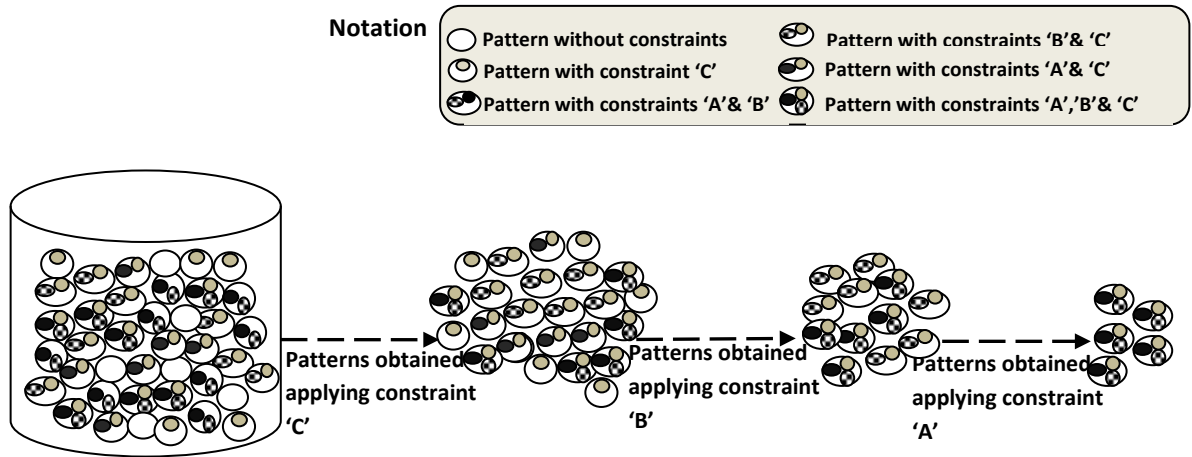


Figure 4.2 Constraint- based Sequential pattern mining in an ‘Inclusive’ manner

4.4. PROPOSED COPRE FRAMEWORK

The framework proposed is based on the inclusive property, introduced in Section 4.3. It is used for the incorporation of the user specified constraints deep into the mining process itself to get the best performance.

We have used a figure to demonstrate the complete flow of discovering the constraint based sequential patterns using the proposed COPRE (CONstraint based PREfix) framework (as shown in Figure 4.3). The suggested framework can incorporate

the user-specified constraints. For the sequence database 'DB', sequential patterns in S can be mined by a prefix-projection method incorporated with the constraints as in the following steps:

- i. **Discover length-1 constraint sequential patterns-** The database 'DB' is scanned once to find all the frequent items in sequences. The length of these frequent sequential patterns is 1. For each such frequent item, check for the required constraints threshold (user specific). This will generate length-1 constraints specific sequential pattern. In the Figure 4.3, we have taken three constraints CT_1 , CT_2 , CT_3 , in addition to frequency.
- ii. **Generate projected database -** The mined 1-length frequent, CT_1 , CT_2 , CT_3 constraint based sequential patterns are then employed to create a projected database, which is the collection of postfixes of sequence with regard to the prefix (1-Frequent, CT_1 , CT_2 , CT_3 pattern). For example, if the projection set contains k number of patterns, then k disjoint subsets can be obtained from the sequential database using the whole set of 1-length Frequent, CT_1 , CT_2 , CT_3 patterns. In the Figure 4.3, we have obtained projected database 'P₁', from the 1-length Frequent, CT_1 , CT_2 , CT_3 constraint based pattern obtained in previous step.
- iii. **Find the n-length constraint based sequential patterns-** The projected database 'P₁' is scanned to discover 2-length sequential patterns which further satisfy all the user specified constraint thresholds. Again, the projected database is created using the mined 2-length Frequent, CT_1 , CT_2 , CT_3 patterns to give 'P₂'. This process is repeated recursively until all the n-length constraint based sequential patterns are determined for the given threshold value of the user specified constraints.

The above framework is able to work with any number of user specified constraints. Moreover, the constraints need not to be belonging to any specific family unlike regular expression constraint based algorithms (Garofalakis *et al*, 1999, Garofalakis, 2002, Albert-Lorincz and Boulicaut, 2003). The user is not required to even define a specific ordering among these constraints as required to formulate a regular expression. In fact, the regular expression constraints can be more handled more efficiently with this pattern –growth framework. There is some work done to incorporate user specified constraints in the mining process itself using algorithms (Hu YH, 2007,

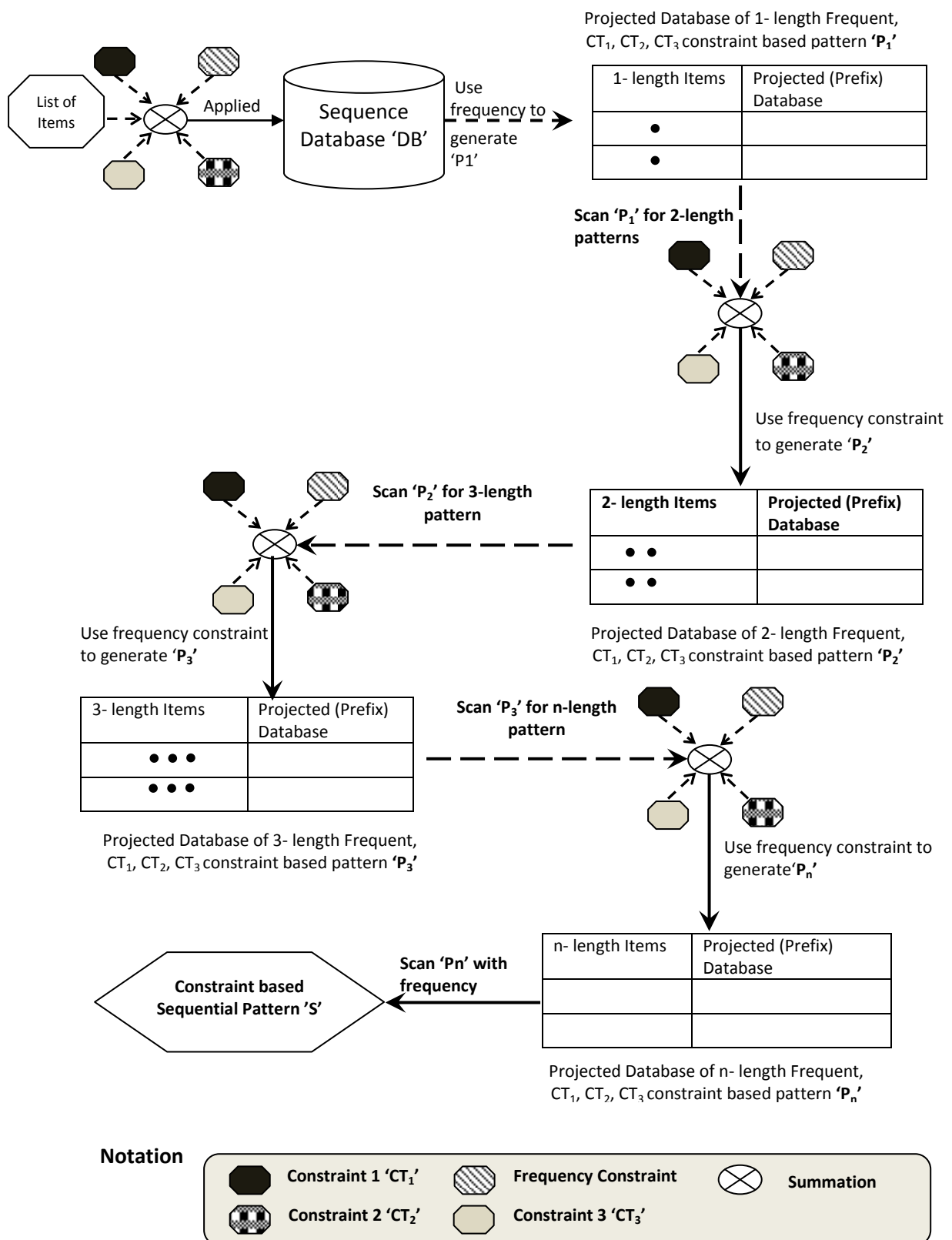


Figure 4.3 Proposed COPRE Framework – Stepwise execution

Chen YL *et al*, 2009). These cited studies are based on Apriori-based approach, just like algorithms using regular expression constraint. The framework proposed by us is based on generating projected databases which is the collection of postfixes of sequence with regard to the given prefix (obtained as sequential pattern in the previous iterations), that is divide and conquer approach. The framework is more efficient, as it does not require candidate generation, counting and testing. Even the sizes of projected databases keep on shrinking with iterations. It can incorporate any number of constraints belonging to same or different types. The framework is based on the general study and is not limited to specific application or domain.

The framework is verified and validated further with three proposed algorithms (discussed in detail in Chapter 5 and 6) which are designed based on this. The performance analysis of these algorithms compared with the naïve and other well-known algorithms justifies the usefulness of the COPRE framework.

4.5. SUMMARY

There is always a need of incorporating user specified constraints in the sequential pattern mining technique for a specific application. There is very less work in the literature where more than two constraints can be used to generate the required sequential patterns. Those limited available work are either based on the Apriori-based technique or have shortcoming of being using constraints of a specific category. We have proposed COPRE framework in this chapter which is based on pattern growth technique. The performance of this framework is evaluated by using three algorithms (discussed in next two chapters) which are using more than two constraints. The proposed COPRE framework is a generalized framework based on inclusive property and can be used to incorporate any number of user-specified constraint required for any application or domain.

Chapter 5

Proposed SPM Algorithms Using Constraints⁵

5.1. INTRODUCTION

Two algorithms, “Constraint-based sequential pattern mining algorithm incorporating compactness, length and monetary” and CFM-PrefixSpan (Compactness, Frequency, Monetary) have been designed and developed in this Chapter. Both of them are constraint-based sequential pattern mining algorithms that are implemented to study customer purchasing behavior. However, the latter one, CFM-PrefixSpan is more specific for customer value analysis for CRM. The basis of selection of constraints has been justified in detail in Chapter 3. The COPRE framework proposed in Chapter 4 is used to design both the algorithms that incorporate more than two constraints. These constraint-based SPM algorithms work well with static databases so re-mine the entire database for changes and modification with time. However, both these algorithms form basis for our proposed incremental mining algorithm of sequential patterns, Progressive CFM-Miner (discussed in next Chapter). The common conventions and terms used in these algorithms are explained in Section 5.2 before the proposed algorithms in Section 5.3 and Section 5.4 respectively.

5.2. PRELIMINARIES

Let $S = \langle (p_1, t_1, M_1), (p_2, t_2, M_2), \dots, (p_n, t_n, M_n) \rangle$ be a data sequence of database D , where p_j is an item, m_j is a purchasing money and t_j signifies the time at which p_j occurs, $1 \leq j \leq n$ and $t_{j-1} \leq t_j$ for $2 \leq j \leq n$. P denotes a set of items in the database D . $I = \{ i_1, i_2, \dots, i_m \}$ be a set of literals, called items. An item set ‘ X ’ is a set of

⁵ The major findings of this Chapter have been published in (i) International Arab Journal of Information Technology, 11(1), January 2014 and (ii) International Journal of Innovative Computing, Information and Control, 8, Number 7(A), pp: 4509 – 4520, July 2012.

items hence, $X \subseteq I$. A sequence $S = (s_1, s_2, \dots, s_n)$ is an ordered set of item sets. Consider two sequences $S_1 = (a_1, a_2, \dots, a_k)$ and $S_2 = (b_1, b_2, \dots, b_l)$. We say that S_1 contains S_2 , or equivalently, S_2 is a subsequence of S_1 if there exist integers $j_1, j_2, \dots, j_l$, such that $1 < j_1 < j_2 < \dots < j_l < k$ and $b_1 \subseteq a_{j_1}, b_2 \subseteq a_{j_2}, \dots, b_l \subseteq a_{j_l}$, represented as $S_2 \subseteq S_1$. A sequence 'S' is said to be constrained if a sequence 'S' contain a specified constraint, C.

The algorithms considers user-specified compact threshold ' T_c ', monetary threshold ' T_m ', length threshold l_s and a user defined minimum support threshold ' $\min_sup$ ' to consider the compactness, monetary, length and frequency constraints in database 'D'.

5.2.1. Terms Used In Algorithms

- i. **Subsequence:** A sequence $S_s = \langle (q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m) \rangle$ is said to be a sub sequence of S only if, (a) itemset S_s is a subsequence of S , $S_s \in S$ (b) $t_1 < t_2 < \dots < t_m$ where, t_1 is the time at which q_1 occurred in S_s , $1 \leq r \leq m$.
- ii. **Compact subsequence:** Let $S_s = \langle (q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m) \rangle$ be a sequence of itemsets, where, $t_1 < t_2 < \dots < t_m$ and T_c be the predefined compact threshold. S_s is known to be a compact subsequence of S if and only if (a) S_s is a subsequence of S , and (b) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_c$.
- iii. **Length constrained Subsequence:** $S_s = \langle (q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m) \rangle$ is said to be a length constrained subsequence of S only if, (a) itemset S_s is a subsequence of S , $S_s \in S$ and (b) the number of items in S should be equal to l_s .
- iv. **Compact Frequent subsequence:** Let D be a sequential database containing itemsets, I and T_c be the predefined compact threshold. S_s is said to be a compact frequent subsequence of D if and only if (a) S_s is a subsequence of D , (b) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_c$, and (c) S_s is a frequent subsequence of database, D .

- v. **Monetary subsequence:** Let $S_s = \langle (q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m) \rangle$ be a sequence of itemsets, where, $t_1 < t_2 < \dots < t_m$ and T_m be the predefined monetary threshold. S_s is said to be the monetary subsequence of S if and only if (a) S_s is a subsequence of S , and (b) the monetary constraint is satisfied, i.e.
- $$\left(\frac{M_1 + M_2 + \dots + M_m}{m} \right) \geq T_m.$$
- vi. **Compact Frequent Monetary subsequence:** Let D be a sequential database containing itemsets (I) , T_C be the predefined compact threshold and T_m be the predefined monetary threshold. S_s is said to be a compact frequent monetary subsequence of D if and only if, (a) S_s is a subsequence of D , (b) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_C$, (c) S_s is a frequent subsequence of database, D and (d) the monetary constraint is satisfied, i.e.
- $$\left(\frac{M_1 + M_2 + \dots + M_m}{m} \right) \geq T_m.$$
- vii. **Compact Frequent Monetary Length subsequence:** Let, D be a sequential database containing itemsets (I) , T_C be the predefined compact threshold, and T_m be the predefined monetary threshold. S_s is said to be a compact frequent monetary subsequence of D if and only if, (a) S_s is a subsequence of D , (b) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_C$, (c) S_s is a frequent subsequence of database D , (d) the monetary constraint is satisfied, i.e.
- $$\left(\frac{M_1 + M_2 + \dots + M_m}{m} \right) \geq T_m,$$
- and (e) the number of items in S should be equal to l_s .

5.3. CONSTRAINT-BASED SEQUENTIAL PATTERN MINING ALGORITHM INCORPORATING COMPACTNESS, LENGTH AND MONETARY (CFML-PREFIXSPAN)

The customer purchasing behavior sequences do fluctuate with change in time, purchasing cost and other parameters. In this algorithm, monetary and compactness constraints in addition to frequency and length are included in the sequential mining

process. The algorithm is proposed by integrating these constraints using COPRE framework (refer Chapter 4). The proposed algorithm is devised from the conventional sequential pattern mining algorithm, PrefixSpan (Pei *et al.*, 2004) and has been validated on synthetic sequential databases. Initially, the proposed algorithm mines 1-length compact frequent patterns (1-CF) by considering compactness threshold and support threshold. Subsequently, the 1-length compact frequent monetary sequential patterns (1-CFML) are filtered from mined 1-CF patterns by inputting the monetary constraint. Then, a projected database corresponding to the mined 1-CF patterns is constructed and then the 2-CF patterns are generated using this database. Again, 2-CFML sequential patterns are determined from 2-CF patterns by integrating the monetary constraint and the process is applied repeatedly until all length constrained-CFML sequential patterns are discovered.

In order to discover most relevant patterns constraints are identified from the perspective of studying customer purchasing behavior. The number of purchases made within a certain period, where a higher frequency specifies higher loyalty is called Frequency constraint. Monetary constraint defines amount of cost spent during a certain period, and a higher value discloses that company should pay more attention to customer. Compactness defines that number of purchases made by customer should be within a reasonable time period. The number of items in a sequence or the number of transactions defines the Length constraint. If the mining process includes the above four concepts, then the decision makers can clearly categorize their customers, and provide a specific score to their customers (as discussed in Section 3.5, Chapter 3). As well, the mined patterns can help the company to find out customers who are more significant.

5.3.1. Pseudo code

Input: Sequence database D , minimum support threshold $\min_sup$, monetary table M_T , predefined compact threshold T_C , and predefined monetary threshold T_m .

Output: Complete set of CFML-sequential patterns β .

Method: Call $CFML\text{-}PrefixSpan(\langle \rangle, 0, D, M_T)$

Subroutine: $CFML\text{-}PrefixSpan(\alpha, l, D|_\alpha, M_T)$

Parameters: α is a sequential pattern; l is the length of α ; $D|_{\alpha}$ is the α -projected database if $\alpha \neq \langle \rangle$ (null) otherwise, the sequence database D ; M_T is the monetary table.

Method:

1. Scan $D|_{\alpha}$ once and find the set of compact frequent items f such that,
 - a) f can be assembled to the last element of α to generate a sequential pattern or
 - b) $\langle f \rangle$ can be appended to α to generate a sequential pattern.
2. For each compact frequent item f , append it to α to form a sequential pattern α' .
3. For each α' ,
 - a) Check monetary using M_T .
 - b) Check length threshold l_s .
4. Create a set β from α' by substituting the findings of step 3.
5. For each α' , create α' -projected database $D|_{\alpha'}$, and call CFML-PrefixSpan ($\alpha', l+1, D|_{\alpha'}, M_T$).

5.3.2. Illustration with an example

5.3.2.1. Step 1: Finding 1- CFML patterns

At first, sequential database D and monetary table M_T are given to the proposed CFML-PrefixSpan algorithm. 1- CFML sequential patterns are mined from sequential database by scanning database once. The 1-CF patterns (compact frequent) which satisfy the predefined compact threshold and support threshold are mined from the sequential database by simply scanning database. Then, we apply the monetary constraint on the 1-CF patterns so that we can obtain a set of 1- CFML patterns.

Example: Let D be the sequential database given in Table 5.1 and M_T be the monetary table given in Table 5.2. We scan the database once and find the set of items that satisfy the predefined compact threshold ($T_C = 4$) and predefined support ($\min_sup = 2$) as follows $[(a \rightarrow 2), (b \rightarrow 3), (c \rightarrow 3), (d \rightarrow 2), (f \rightarrow 1)]$. In this set, the patterns that satisfy the compact threshold and support threshold known as 1-CF patterns are as follows $[(a \rightarrow 2), (b \rightarrow 3), (c \rightarrow 3), (d \rightarrow 2)]$. Next, we compute the monetary of the 1-CF patterns obtained from the previous step, $[a \rightarrow (2, 5), b \rightarrow (3, 10), c \rightarrow (3, 15), d \rightarrow (2, 2)]$. Based on the monetary threshold ($T_m = 10$), we obtain the following set of 1-CFML patterns as follows

{b and c}. $\langle \rangle$ represent a transaction, (item, number) represent the data item and the number represent the time stamp value. If there are more than one item purchased together will have the same time stamp value.

Customer ID	Sequence
1	$\langle (a,1), (b,3), (c,4), (d,4), (f,5) \rangle$
2	$\langle (b,1), (c,2), (d,3) \rangle$
3	$\langle (a,3), (b,4), (c,4) \rangle$

Table 5.1 Sequence Database ‘SD1’

Item	Monetary value
a	5
b	10
c	15
d	2
f	10

Table 5.2 Monetary Value Table ‘MVT1’

5.3.2.2. Step 2: Dividing search space

The mined 1-CF patterns are then used to construct projected database that is the collection of postfixes of sequence with regard to the prefix (1-CF pattern). Suppose, if the projection set contains k number of patterns, then we can obtain k disjoint subsets from the sequential database using the complete set of 1-length compact frequent patterns.

Example: Here, we form the projected database for the 1-CF patterns such as {a, b, c and d}. The steps used for constructing the projected database of the pattern $\langle a \rangle$ are as follows: By looking at the first sequence in the database, $\langle a \rangle$ has a time stamp value of 1. So, the projection based on the first sequence is obtained by taking the postfixes of pattern $\langle a \rangle$ (sequences after the time stamp 1) in the first sequence. In a similar way, we obtain the projection for the rest of sequences present in the sequential database. The projected database for the pattern $\langle a \rangle$ contains $\langle (b, 3), (c, 4), (d, 4), (f, 5) \rangle$ and $\langle (b, 4), (c, 4) \rangle$. Similarly, the projection is done for other 1-CF patterns. Table 5.3 shows the projected database of all one length CF patterns in the projection set.

$\langle a \rangle$	$\langle (b,3), (c,4), (d,4), (f,5) \rangle \langle (b,4), (c,4) \rangle$
$\langle b \rangle$	$\langle (c,4), (d,4), (f,5) \rangle \langle (c,2), (d,3) \rangle \langle (c,4) \rangle$
$\langle c \rangle$	$\langle (d,4), (f,5) \rangle \langle (d,3) \rangle$
$\langle d \rangle$	$\langle (f,5) \rangle$

Table 5.3 Projected Database for 1-length Compact Frequent pattern

5.3.2.3. Step 3: Finding subsets of sequential patterns

Here, a set of 2-length compact frequent patterns are mined by scanning the projected database once. Subsequently, a set of 2-CF patterns are obtained by applying the monetary constraint on the 2-length compact frequent patterns. Again, the projected database is created using the mined 2-CF patterns and this process is repeated recursively until all CFML patterns are determined for the given threshold l_s .

Example: The projected database formed by the 1-length CF sequential patterns is utilized for mining all 2-length CFML sequential patterns. The procedure employed for mining 2-length CFML sequential patterns having prefix $\langle a \rangle$ is as follows: By scanning the projected database once, we obtain the count of the compact frequent items which is represented as, $[(a \rightarrow 0), (b \rightarrow 2), (c \rightarrow 1), (d \rightarrow 0), (f \rightarrow 0)]$. In this set, the patterns that are satisfying the compact threshold and support threshold are given as, $[(b \rightarrow 2)]$. The mined 2-CF sequential pattern is $\{ab\}$. Then, we apply the monetary constraint on the mined 2-CF sequential pattern so that, we can obtain the 2-CFML pattern $[\langle ab \rangle \rightarrow (2, 7.5)]$. There is no 2-length CFML sequential pattern having a prefix $\langle a \rangle$ since the pattern $\langle ab \rangle$ has not satisfied the given monetary threshold. Again, we form the projected database based on the 2-CF sequential patterns and the 3-CF patterns are obtained by scanning the projected database. Then, we mine all length CFML patterns with prefix $\langle a \rangle$ recursively. The aforementioned procedure is repeated for other 1-CF patterns $\langle b \rangle$, $\langle c \rangle$ and $\langle d \rangle$. The mined CFML-patterns are $\{\langle b \rangle, \langle bc \rangle, \langle c \rangle\}$ for the threshold length, $l_s = 2$.

5.3.3. Experimental Setup and Dataset Description

CFML-PrefixSpan and the naïve PrefixSpan algorithms are programmed using JAVA (jdk 1.6) through the help of Netbeans IDE. The experiment setup includes the PC running on core i3 processor with 4 GB RAM. The performance comparison is done using the synthetic dataset. A set of synthetic data sequence (Appendix) is generated by a data generator similar in spirit to the IBM data generator designed for testing sequential pattern mining algorithms. Each sequence contains item sets. However, different time values are assigned to the items in different item sets but the same time values to those in

the same item sets. A sequential database that contains 10,000 sequences of 10 items is generated.

5.3.4. Result Analysis

The experimental results of the proposed CFML-PrefixSpan algorithm for efficacious mining of CFML patterns is described in this section. The proposed CFML-PrefixSpan algorithm is programmed by means of JAVA (jdk 1.6). The sample database taken for experiment is given in Table 5.1 and the monetary constraint is given in Table 5.2. Such database and monetary table are given as an input to the proposed CFML-PrefixSpan algorithm for successful mining of CFML sequential patterns. Originally, we mined the 1-CFML patterns based on the thresholds, $T_c = 4$, $\min_sup = 2$, $T_m = 10$, $l_s = 2$. Subsequently, the projection was done based on the mined 1-length compact frequent patterns. The projected database for the 1-CF pattern is shown in the Table 5.3. Eventually, we obtained a complete set of CFML patterns for the given input sequential database. The obtained complete set of CFML pattern set is $\{, <bc>, <c>\}$.

The comparative results of the PrefixSpan with our proposed CFML-PrefixSpan algorithm are given in Table 5.4. It clearly ensures that the proposed algorithm provides lesser number of sequential patterns compared to PrefixSpan algorithm. The PrefixSpan algorithm contains all the less profitable and longer time length sequential patterns ($<a>$, $<ab>$, $<ac>$, $<d>$ and $<bd>$) but the proposed algorithm generates CFML sequential patterns that contain only the profitable and valuable sequential patterns ($$, $<bc>$ and $<c>$). So from a business perspective, CFML-PrefixSpan algorithm is more suitable for developing better business strategies compared to PrefixSpan algorithm.

CFML-sequential Pattern		Sequential Pattern
$<a>$		$<a>, <ab>, <ac>$
$$	$, <bc>$	$, <bc>, <bd>$
$<c>$	$<c>$	$<c>$
$<d>$		$<d>$

Table 5.4 Comparison of patterns discovered from CFML-PrefixSpan and PrefixSpan algorithms

5.3.5. Performance Analysis

5.3.5.1. Various Length- Threshold

The synthetic dataset is given to the proposed CFML-PrefixSpan algorithm for mining the CFML sequential patterns. The predefined threshold values given to our algorithm are, $T_C = 4$, $\min_sup = 1000$ and $T_m = 10$. Based on the given database and other parameters, the algorithm produced a complete set of CFML sequential patterns for the given length-threshold l_s . Then, the same sequential database is given to the PrefixSpan algorithm for mining the sequential patterns. The results obtained from both the algorithms are shown in Table 5.5. The plotted graph is illustrated in Figure 5.1 that proves less number of useful, desired and interesting patterns (as discussed in Section 3.3 and Section 3.4) is generated by the proposed algorithm than the PrefixSpan algorithm.

5.3.5.2. Computation Time

As computation time is considered one of the important parameters to find the intricacy of the algorithm, this parameter is considered for comparative study of two algorithms. Initially, by inputting the $\min_sup = 1000$ and $T_m = 10$, a set of sequential patterns is discovered with time taken by the algorithms for various threshold T_C (for CFML-PrefixSpan) and length (PrefixSpan). The results obtained are plotted in a graph, which is shown in Figure 5.2. While comparing the computational complexity, it has been found that the proposed algorithm has taken less computation time than the PrefixSpan algorithm for higher threshold values.

Length threshold	Number of CFML sequential patterns	Number of sequential patterns
$l_s = 1$	8	10
$l_s = 2$	143	200
$l_s = 3$	24	1000

Table 5.5 No. of sequential patterns generated by CFML-PrefixSpan and PrefixSpan algorithms for various length thresholds

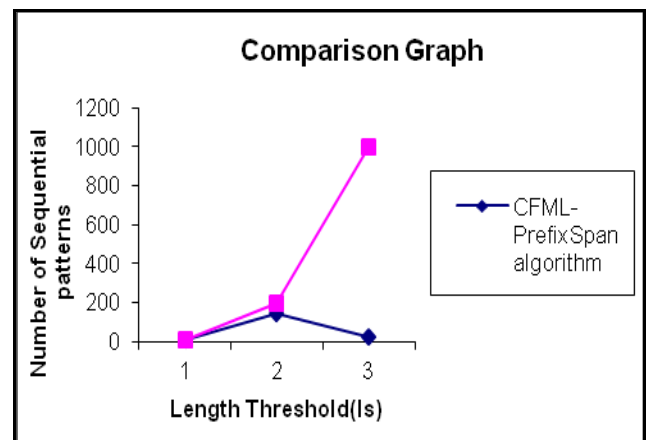


Figure 5.1 Comparison graph for number of sequential patterns generated by CFML-PrefixSpan and PrefixSpan algorithms

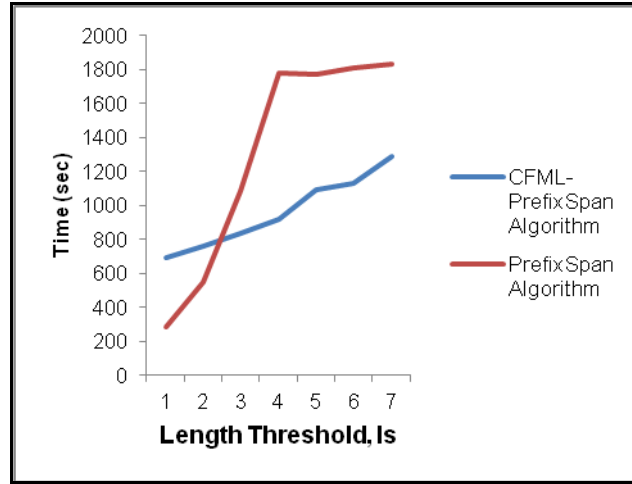


Figure 5.2 Runtime performances of CFML-PrefixSpan and PrefixSpan algorithms

5.3.5.3. Various Compact Thresholds

The two algorithms are compared in terms of the number of useful sequential patterns obtained for diverse compact support thresholds. By inputting $\min_sup = 1000$ and $T_m = 10$, the results are computed by varying T_C and the obtained results are shown in the Figure 5.3. From the graph, it is clear that the sequential patterns obtained by the proposed algorithm are considerably less compared to the PrefixSpan algorithm because the proposed algorithm is capable of discovering more relevant sequential patterns.

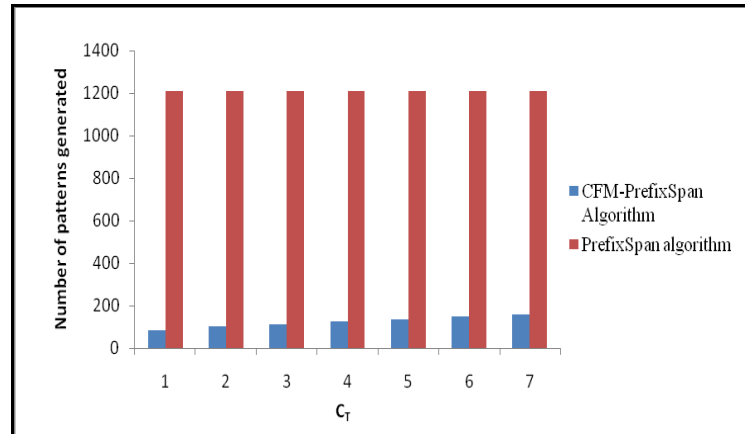


Figure 5.3 Comparison graph of the number of patterns generated for various compact thresholds by CFML-PrefixSpan and PrefixSpan algorithms

5.4. CFM-PREFIXSPAN: A CONSTRAINT-BASED ALGORITHM FOR MINING SEQUENTIAL PATTERNS WITH CRM PERSPECTIVE

This algorithm incorporates monetary, frequency and compactness constraints using COPRE framework, with the naïve approach of PrefixSpan algorithm. As supported by the arguments in Chapter 3, customer value analysis require crucial seventeen parameters. The selected constraints not only satisfy these parameters but also help to

acquire new customers, retain loyal and valuable customers for CRM. Since the number of items or the number of transactions accounted by length constraint is not a significant factor to be considered for customer value analysis for CRM, the constraint can be avoided for this domain. The study further makes use of this algorithm for considering the dynamic and distributed environment of CRM and giving more appropriate Progressive CFM-Miner, an incremental mining algorithm of sequential patterns in Chapter 6.

5.4.1. Pseudo Code

The CFM-PrefixSpan algorithm is outlined as follows.

Input : A sequence database D , and the minimum support threshold $\min_sup$, monetary table M_T , Predefined compact threshold T_C , Predefined monetary threshold T_m .

Output: The complete set of CFM-sequential patterns β .

Method: Call $CFM_PrefixSpan(\langle \rangle, 0, D, M_T)$.

Subroutine: $CFM_PrefixSpan(\alpha, l, D|_\alpha, M_T)$

Parameters: α is sequential pattern; l is the length of α ; $D|_\alpha$ is the α -projected database, if $\alpha \neq \langle \rangle$ (null); otherwise, the sequence database D ; M_T is the monetary table.

Method:

1. Scan $D|_\alpha$ once, find the set of compact frequent items f such that
 - a) f can be assembled to the last element of α to form a sequential pattern; or
 - b) $\langle f \rangle$ can be appended to α to form a sequential pattern.
2. For each compact frequent item f , append it to α to form a sequential pattern α' .
3. For each α' ,
 - a) check monetary using M_T ;
4. Create a set β from α' by substituting the findings of step 3.
5. For each α' , construct α' -projected database $D|_{\alpha'}$, and call $CFM_PrefixSpan(\alpha', l+1, D|_{\alpha'}, M_T)$.

5.4.2. Illustration with an example

The purchasing sequence of 5 customers is being considered for an illustration, which consists of five data items – a, b, c, d and e [Table 5.6]. The monetary value of each item is as given in Table 5.7.

Notations

- i. $\langle \rangle$ represent a transaction.
- ii. (item, number) represent the data item and the number represent the time stamp. If there are more than one item purchased together will have the same time stamp.

Customer ID	Purchasing Sequences along with the timestamps
101	$\langle (a,1),(b,2),(c,3),(d,4), (e, 5) \rangle$
102	$\langle (b,1),(c,2),(d,3) \rangle$
103	$\langle (a,3),(b,4),(c,5) \rangle$
104	$\langle (a,2),(d,3),(e, 4) \rangle$
105	$\langle (c,1),(b,2),(d,3) \rangle$

Table 5.6 Purchasing Sequence Database of Customers

Item	Monetary Value
a	40
b	35
c	20
d	50
e	5

Table 5.7 Monetary value of each item

For simplicity we have considered single transaction for each customer. However the sequential patterns are generated using inter- transactions may be of different customers.

5.4.2.1. Step 1: Finding 1- CFM patterns

Scan the database with the compact threshold $T_c = 4$, minimum support threshold $\min_sup = 2$ (to count frequency) and monetary threshold $T_m = 35$. We get 1-CF patterns with the frequency count as $[(a \rightarrow 3), (b \rightarrow 4), (c \rightarrow 4), (d \rightarrow 4), (e \rightarrow 2)]$ and 1- CFM patterns as $\{a, b, d\}$.

5.4.2.2 Step 2: Generating projected database

The projected database is generated using the 1-length frequent patterns, which is the collection of postfixes of sequence with regard to the prefix of frequent 1-length pattern, as shown in Table 5.8. The first sequence in the database has 'a' as an item with time stamp value of 1. So, the projection based on 'a' is obtained by taking the postfixes of 1-length frequent 'a' (after the time stamp 1) in the first sequence. We can obtain the projection for the rest of sequences present in the sequential database in a similar way.

The projected database for the pattern 'a' contains $\langle(b,2),(c,3),(d,4),(e,5)\rangle$, $\langle(b,4),(c, 4)\rangle$ and $\langle(d,3),(e, 4)\rangle$. Similarly, the projection is done for other 1-length frequent patterns – b, c and d.

a	$\langle(b,2),(c,3),(d,4),(e, 5)\rangle$ $\langle(b,4),(c, 4)\rangle$ $\langle(d,3),(e, 4)\rangle$
b	$\langle(c,3),(d,4),(e, 5)\rangle$ $\langle(c,4)\rangle$ $\langle(c, 2),(d, 3)\rangle$ $\langle(d, 3)\rangle$
c	$\langle(d,4),(e, 5)\rangle$ $\langle(d,3)\rangle$ $\langle(b,2),(d,3)\rangle$
d	$\langle(e,4)\rangle$ $\langle(e,5)\rangle$
e	nil

Table 5.8 Projected Database of 1-length Constraint based Sequential Patterns

5.4.2.3 Step 3: Formulating sequential patterns

This projected database given in Table 5.8 is used to generate 2-length CFM sequential patterns. The projected database is scanned to obtain the count of the frequent items. For 'a', the count is given as, $[a \rightarrow 0)$, $(b \rightarrow 2)$, $(c \rightarrow 0)$, $(d \rightarrow 1)$, $(e \rightarrow 0)]$. In this set, the patterns that are satisfying the compact threshold and support threshold are given as, $[(b \rightarrow 2)]$. The mined 2-CF sequential pattern is {a, b}. Applying the monetary constraint on the mined 2-CF sequential pattern to obtain the 2-CFM pattern $[\{a, b\} \rightarrow (2, 37.5)]$. That satisfies the given monetary threshold. Again, the projected database is formulated based on the 2-length frequent sequential patterns and the 3-CF patterns are obtained by scanning the projected database. Then, all length CFM patterns is mined with prefix 'a' recursively. The same process is repeated for other 1-length frequent patterns.

5.4.3. Experimental Setup and Dataset Description

The CFM-PrefixSpan and PrefixSpan algorithms are programmed using JAVA (jdk 1.6) through the help of Netbeans IDE. These algorithms are executed on machine having core i3 processor and 4 GB RAM. The performances of the algorithms have been evaluated using the synthetic datasets as well as real life datasets. *Synthetic dataset (Appendix)*: The dataset is generated containing a set of sequences of item sets. However, different time values are assigned to the items in different item sets but the same time values to those in the same item sets. A sequential database that contains 10,000 sequences of 10

items is generated. **Real life datasets:** The UCI machine learning repository (<http://archive.ics.uci.edu/ml/datasets>) is used for the implementation of all algorithms. This data describes the page visits of users who visited msnbc.com on September 28, 1999. Visits are recorded at the level of URL category ("frontpage", "news", "tech", "local", "opinion", "on-air", "misc", "weather", "health", "living", "business", "sports", "summary", "bbs" (bulletin board service), "travel", "msn-news", and "msn-sports") and are recorded in time order. We have utilized this real dataset with 1000 records of 10 transactions.

5.4.4. Result Analysis

The experimental results of the proposed CFM-PrefixSpan algorithm for mining of CFM patterns is described in this section. The proposed CFM-PrefixSpan algorithm is programmed by means of JAVA (jdk 1.6). The sample database taken for experiment is given in Table 5.6 and the monetary constraint is given in Table 5.7. CFM patterns based are mined from the database using the thresholds, $T_C = 4$, $\min_sup = 2$, $T_m = 10$. The projected database for the 1-CF pattern is shown in the Table 5.8. Eventually, we obtained a complete set of CFML patterns for the given input sequential database.

The complete set of CFM-sequential patterns is discovered using the algorithm as given in Table 5.9. The sequential patterns generated using PrefixSpan algorithm is also given in the same table. It is obvious that the patterns obtained using the proposed algorithm is far less than otherwise.

List of Items	CFM-PrefixSpan algorithm	PrefixSpan algorithm
a	<(a)> <(a, b)>	<(a)> <(a, b)> <(a, b, c)>
b	<(b, c, d)> <(b)>	<(b, c, d)> <(b)> <(b, c)>
c	<(c, d)>	<(c)> <(c, d)>
d	<(d)>	<(d)> <(d, e)>
e	nil	<(e)>

Table 5.9 Reduced number of patterns obtained using the CFM-PrefixSpan algorithm

5.4.5. Performance Analysis

5.4.5.1. Effect of support values

In order to analyze the effects of the algorithms in terms of support value, the synthetic and real datasets are given to the PrefixSpan and CFM-Prefixspan algorithm.

These algorithms are compared for the number of sequences obtained, memory usage and computation time. The values obtained through the experiments are plotted as graphs that are shown in Figure 5.4, Figure 5.5 and Figure 5.6 respectively. From the graphs, we can understand that the number of sequence generated from the CFM-Prefixspan algorithm is less compared with Prefixspan algorithm. This signifies that the most important rules are only mined by the CFM-Prefixspan algorithm. When analyzing the run time performance, the proposed algorithm outperformed the previous algorithm in both synthetic and real dataset. For synthetic datasets, the proposed algorithm achieved six times more efficiency in computation time and for real datasets also, the proposed algorithm achieved comparable results.

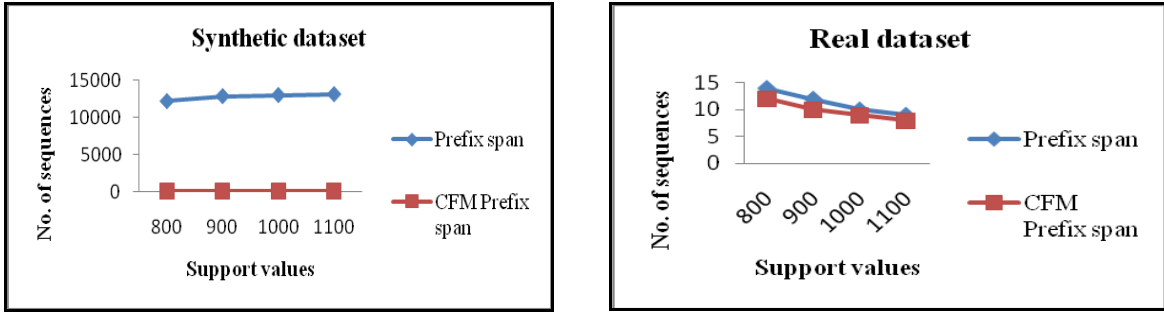


Figure 5.4 Number of sequences generated by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values.

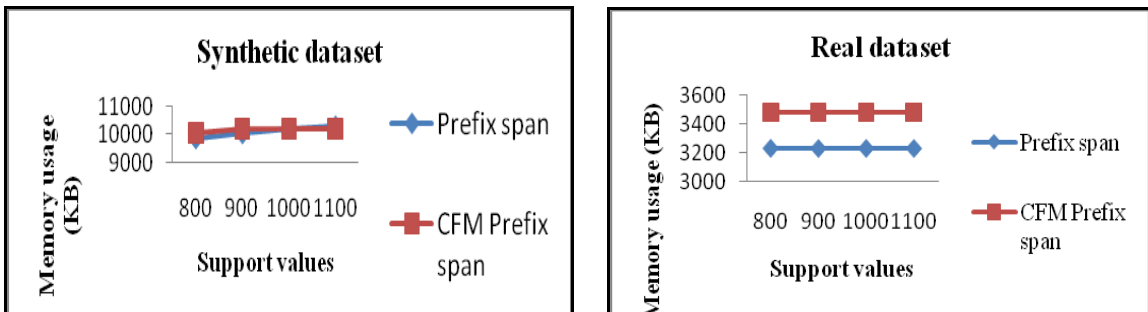


Figure 5.5 Memory usage by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values.

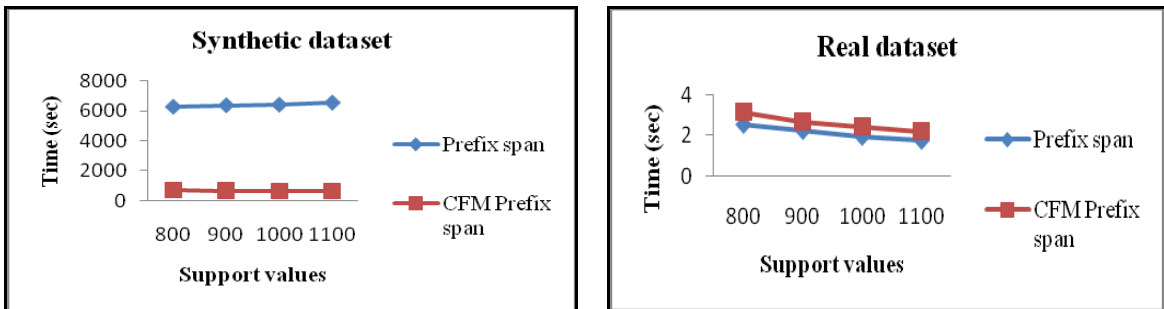


Figure 5.6 Runtime performances by PrefixSpan and CFM-PrefixSpan for real and synthetic databases with various minimum support threshold values.

5.4.5.2. Effect of scalability

The scalability of the algorithms is analyzed in both synthetic and real datasets using number of sequences, computation time and memory usage. For various numbers of records, the number of sequences are obtained, computation time and memory usage are calculated. The values that are obtained through the experiments are plotted as graphs that are shown in Figure 5.7 to Figure 5.9. When comparing the sequential patterns generated, the proposed algorithm mines the more significant patterns in synthetic and real datasets. Furthermore, the computation time required to mine the sequential pattern is comparable with the PrefixSpan algorithm. Comparing the computation time for synthetic datasets, the proposed algorithm needs very less time compared with the naïve algorithm. Similarly, the performance of the proposed algorithm is also effective in comparing the memory usage.

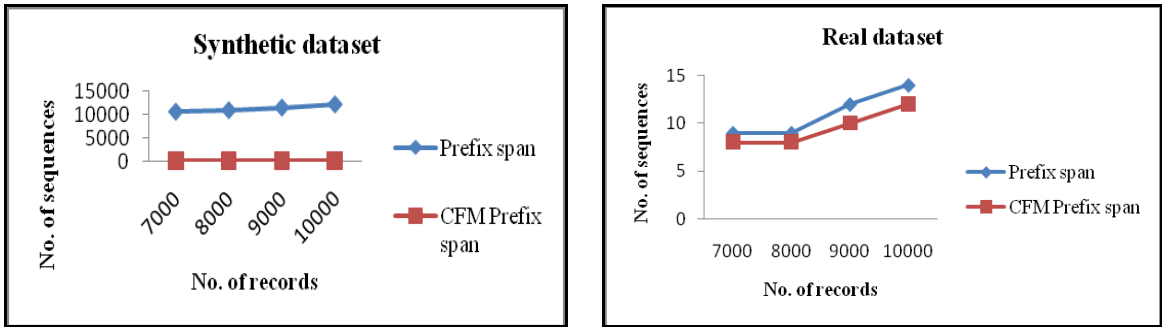


Figure 5.7 Number of sequences generated by PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases.

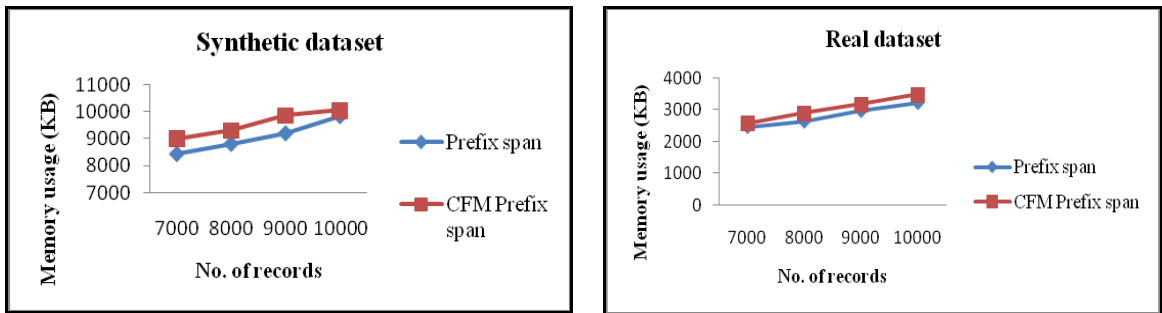


Figure 5.8 Memory usage by PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases.

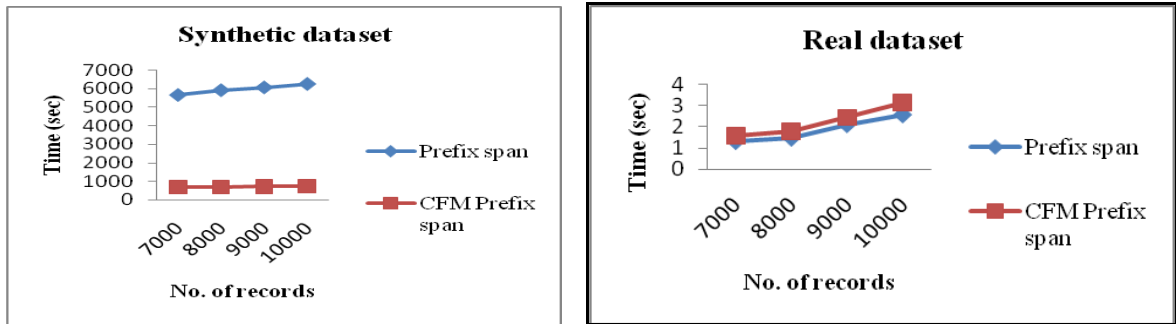


Figure 5.9 Computation time comparisons for PrefixSpan and CFM-PrefixSpan with different number of records of real and synthetic databases.

5.5. SUMMARY

Two efficient algorithms for mining constraint-based sequential patterns from the customer transaction database are presented in this Chapter. These algorithms employed a pattern-growth methodology that finds sequential patterns by utilizing a divide-and-conquer strategy. We have used constraints in addition to frequency for mining interesting and valuable sequential patterns. In our algorithm, the sequence database has been recursively projected into a set of smaller projected databases based on the compact frequent patterns. Besides, CF-sequential patterns have been discovered in each projected database by exploring only locally compact frequent items and then, the constraint based sequential patterns are discovered. The discovered sequential patterns signify valuable information on customer purchasing behavior and customer value analysis ensuring that all patterns have reasonable time spans with good profit. The experimental results have showed that the effectiveness of sequential pattern mining algorithms can be improved significantly by incorporating constraints into the mining process.

These algorithms forms the basis for the design of our proposed algorithm for incremental mining of sequential patterns, Progressive CFM-Miner discussed in Chapter 6. However, the applications of these two algorithms are conversed in detail in Chapter 8.

Chapter 6

Proposed Algorithm for Incremental Mining of Sequential Patterns⁶

6.1. INTRODUCTION

The proposed algorithm for incremental mining of sequential patterns, Progressive CFM (Compactness, Frequency, Monetary)-Miner is developed to extract useful patterns from progressive databases. This work is preceded by development of two algorithms, CFML-PrefixSpan and CFM-PrefixSpan (discussed in Chapter 5). These are constraint-based sequential pattern mining algorithms that work well with static databases. With dynamic changes these algorithms need to re-mine entire database to get new updated constraint based patterns. CFM-PrefixSpan is a special case of CFML-PrefixSpan algorithm without length constraint. As discussed, in Chapter 3, selection of compactness, frequency and monetary constraints is specific to customer value analysis for CRM. However, CFML-PrefixSpan studies customer purchasing behavior for any application. The selection of appropriate constraints makes the difference in their application.

This research work started with study of customer purchasing behavior to present CFML-PrefixSpan algorithm. Later, we continued our study for CRM domain, and hence use CFM-PrefixSpan algorithm as a vital step for our proposed incremental mining algorithm, Progressive CFM-Miner. However, all these three algorithms make use of proposed COPRE framework to incorporate constraints.

6.2. PROGRESSIVE CFM-MINER: AN ALGORITHM FOR INCREMENTAL MINING OF SEQUENTIAL PATTERNS

As databases develop, the problem of maintaining sequential patterns over an extensively long period of time turn into essential, since a large number of new records may be added to a database. To reflect the current state of the database where previous sequential

⁶ The major findings of this chapter have been published in International Journal of Computational Intelligence System, 6(2), pp. 209-222, January 2013.

patterns become insignificant due to the addition of fresh sequential patterns, it is must to have competent approaches and algorithms that can maintain and manage the updated knowledge. Incremental mining algorithms efficiently calculate the new set of frequent item sets by fundamentally reusing beforehand mined information and attempting to merge this information with the fresh data. In fact, several application domains incrementally update the contents of databases. For instance, appending of newly bought items for existing customers for their later buying and/or inclusion of new shopping successions for new customers causes the shopping transaction database to grow on a daily basis (Wang *et al.*, 2007). This helps to reduce the computational and I/O expenses (Chen and Cook, 2007).

Here, an efficient algorithm to handle the maintenance problem of CFM--PrefixSpan algorithm (Compact, Frequent, Monetary-constraints based sequential patterns) is presented. In order to efficiently capture the dynamic nature of data addition and deletion into the mining problem, initially, CFM-tree is constructed using the CFM patterns obtained from static database. Then, the database gets updated from the distributed sources that have data which may be static, inserted, or deleted. Whenever the database is updated from the multiple sources, CFM tree is also updated by including the updated sequence. Then, the updated CFM-tree is used to mine the progressive CFM-patterns using the proposed tree pattern mining algorithm.

6.3. PRELIMINARIES

Let $S = \langle (p_1, t_1, M_1), (p_2, t_2, M_2), \dots, (p_n, t_n, M_n) \rangle$ be a data sequence of database D , where p_j is an item, m_j is a purchasing money and t_j signifies the time at which p_j occurs, $1 \leq j \leq n$ and $t_{j-1} \leq t_j$ for $2 \leq j \leq n$. P denotes a set of items in the database D . $I = \{ i_1, i_2, \dots, i_m \}$ be a set of literals, called items. An item set 'X' is a set of items hence, $X \subseteq I$. A sequence $S = (s_1, s_2, \dots, s_n)$ is an ordered set of item sets. Consider two sequences $S_1 = (a_1, a_2, \dots, a_k)$ and $S_2 = (b_1, b_2, \dots, b_l)$. We say that S_1 contains S_2 , or equivalently, S_2 is a subsequence of S_1 if there exist integers $j_1, j_2, \dots, j_l$, such that $1 < j_1 < j_2 < \dots < j_l < k$ and $b_1 \subseteq a_{j_1}, b_2 \subseteq a_{j_2}, \dots, b_l \subseteq a_{j_l}$, represented as $S_2 \subseteq S_1$. A sequence 'S' is said to be constrained if a sequence 'S' contain a specified constraint, C.

For the incremental update problem, we consider that the constraint sequential pattern mining can be executed on database D to find the constraint sequences. But, the database D is updated by inserting or deleting set of sequences dB . Let us denote the updated database U such that $U = D \cup dB$. Here, the incremental update problem is to find all constraint frequent sequences in the database U for each next time intervals without scanning the whole database U .

The algorithms considers user-specified compact threshold ' T_c ', monetary threshold ' T_m ', length threshold l_s and a user defined minimum support threshold ' $\min_sup$ ' to consider the compactness, monetary, length and frequency constraints in database ' D '.

6.3.1. Terms Used in Algorithm

- i. **CFM-tree:** For a sequence database D , we can construct a CFM-tree after mining the CFM patterns from it. Here, every node ' n ' in the CFM-tree contains items and its relevant information, represented as, $n = [(p \{t_1, t_n\}), (M, F)]$, where, ' p ' is the item, t_1 is the starting time interval, t_n is the ending time interval, M is monetary and F is frequency. Here, the depth of the CFM-tree, ' d ' is equivalent to the larger length of the CFM-sequential patterns.
- ii. **Empty node:** A node in the CFM-tree is called as empty node only if (a) t_1 and t_n is filled with zero, (b) ' p ' should contain the item information and (c) M and F have the zero value. This node is necessary for building the CFM-tree after mining the sequences from the static database because the CFM-miner does not satisfy the downward closure property. So, some of the sequential patterns are frequent but, their subsets may not be frequent. These types of subsets are stored in the empty nodes, but their supersets are stored in the precious CFM-node that is frequent.
- iii. **Precious CFM-node:** A node in the CFM-tree is called as precious CFM-node only if (a) t_1 and t_n contains the information of time occurrence, (b) ' p ' should contain the item and, (c) M and F have the valuable information about its monetary and frequency.
- iv. **Updated CFM tree:** After inserting some nodes in CFM-tree on behalf of updated database, then it is called as, updated CFM-tree, in which some nodal information

may be updated or some new nodes may be included.

- v. **Incompact node:** A node in the updated CFM-tree is known as, incompact node only if, t_n value is less than T , where, T is the user specified threshold.
- vi. **Non-zero infrequent node:** A node in an updated CFM Tree is said to be a non-zero infrequent node if the following conditions are satisfied: (a) the frequent value, F should be less than the 'min_sup', (b) should not be updated and (c) the monetary constraint should be satisfied.
- vii. **Progressive compact sequence:** Let $S = \{(p_1, t_1, M_1), (p_2, t_2, M_2), \dots, (p_n, t_n, M_n)\}$ be a data sequence of database D and a sequence $S_U = \{(q_m, t_{n+1}, M_m)\}$ be an updated sequence, where p_j is an item, m_j is a purchasing money and t_j signifies the time at which p_j occurs, $1 \leq j \leq n$ and $t_{j-1} \leq t_j$ for $2 \leq j \leq n$. 'P' denotes a set of items in the database D . A sequence $S_s = \{(q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m)\}$ is said to be a progressive compact sequence only if, (a) item set S_s is a subsequence of $S || S_U$, (b) S_s should have item, q_m , and (c) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_c$.
- viii. **Progressive compact monetary sequence:** Let $S = \{(p_1, t_1, M_1), (p_2, t_2, M_2), \dots, (p_n, t_n, M_n)\}$ be a data sequence of database D and a sequence $S_U = \{(q_m, t_{n+1}, M_m)\}$ be an updated sequence, where p_j is an item, m_j is a purchasing money and t_j signifies the time at which p_j occurs, $1 \leq j \leq n$ and $t_{j-1} \leq t_j$ for $2 \leq j \leq n$. 'P' denotes a set of items in the database D . A sequence $S_s = \{(q_1, t_1, M_1), (q_2, t_2, M_2), \dots, (q_m, t_m, M_m)\}$ is said to be a progressive compact monetary sequence only if, (a) item set S_s is a subsequence of $S || S_U$, (b) S_s should have item q_m , (c) the compactness constraint is satisfied, i.e. $t_m - t_1 \leq T_c$ and (d) the monetary constraint is satisfied, i.e. $\{(M_1 + M_2 + \dots + M_m) / m\} \geq T_m$.

6.3.2. Types of Update Operations in Incremental Mining

There are two types of updates that can be made in sequence database. These are insertion of new sequences in database, referred as INSERT in this chapter and appending new item/itemsets in existing sequences referred as APPEND. INSERT is easier to handle as compared APPEND (Cheng et al., 2004). A frequent sequence in 'U' as a result of INSERT operation is either due to this sequence being frequent in 'DB' or in

'db' or in both. So, mining algorithm can be extended to handle INSERT. But, in case of APPEND; appended items may produce new local frequent sequences in 'db'. Even, local infrequent sequences may also contribute their occurrence count in original database 'DB' to generate result as frequent sequences. For example, if there are 1000 sequences in 'DB' and 25 in 'db' with min_sup as 5%. Let, there be sequence, 's' which has 49 occurrences in 'DB' so it is infrequent with ' min_sup ' as 4.9%. It is also treated as infrequent with 1 occurrence in 'db' as ' min_sup ' would be 4%. It however, becomes frequent in 'U' with 50 occurrences, with ' min_sup ' as 5%. As the item/itemsets are appended in existing sequences so 'U' have 1000 sequences, even after APPEND.

6.4. Stepwise execution of the algorithm

Generally, the change on a sequential database can be categorized as (a) deleting records, (b) inserting new records and (c) appending new items on the existing records. By handling these issues, the proposed algorithm was designed with the aid of five major steps.

- i. Mining of CFM sequential patterns from the static database
- ii. Building up the CFM-tree from the CFM patterns
- iii. Handling the update operation
- iv. Handling the node deletion operation in the updated CFM-tree
- v. Mining of progressive CFM patterns from the progressive database

6.4.1. Step 1: Mining of CFM Sequential Patterns from the Static Database

In this, CFM patterns from the static database are efficiently mined using the CFM algorithm proposed in previous chapter. It discovers 1-length compact frequent patterns (1-CF) by considering compactness threshold (T_c) and support threshold (min_sup). Then, 1-length sequential patterns (1-CFM) are filtered from mined 1-CF patterns by inputting the monetary constraint (T_m). Subsequently, the projected database is built corresponding to the mined 1-CF patterns and 2- CF patterns that are mined from the projected database. Again, we found the 2-CFM sequential patterns from it and the process was applied recursively until all length CFM sequential patterns were mined.

Example: The sample database is given in Table 6.1 in which the timestamps T1 to T5 are static set of data, whereas the timestamps T6 to T7 are the updated set of data.

Seq. Id	T1	T2	T3	T4	T5	T6	T7
01	a	abc	ac	d	cf	h	
02	ad	c	bc	ae			
03	f	ab	c	df	cb		
04		g	af	c	b		
05						cb	ab
06							

Table 6.1 Sequence Database ‘SD1’

Item	Monetary value
a	2
b	10
c	20
d	20
e	5
f	15
g	25
h	2

Table 6.2 Monetary Value Table ‘MVT2’

The corresponding monetary values of all the items are given in Table 6.2 and the mined CFM-sequential patterns using our previous algorithm for the input thresholds, ($\min_sup \geq 2$, $T_c \leq 3$, $T_m \geq 10$) are shown in Table 6.3.

CFM-Sequential Patterns	
a	<ac>,<acb>,<acc>
b	,<bc>,<bcc>,<bcd>,<bcdc>
c	<c>,<ca>,<cc>
(ab)	<(ab)>,<(ab)c>,<(ab)cc>,<(ab)cd>,<(ab)cdc>
(bc)	<(bc)>,<(bc)a>
d	<d>
f	<f>,<fc>

Table 6.3 Mined CFM-Sequential Patterns

6.4.2. Step 2: Building up the CFM-tree from the CFM Patterns

After the mining process of the CFM-sequential patterns, we have built the CFM-tree from the mined CFM-sequential patterns. The process of building up the CFM-tree is explained as follows. The monetary value and the frequency value of each of the patterns should be maintained properly. The CFM tree that contains all the sequential patterns are building up, by which mines the progressive CFM-patterns without candidate

6.4.3. Step 3: Handling the Update Operation

After building up the CFM-tree from static database, we have to build the tree structure of the updated sequences. After inserting some of the transactions, if items order of the list deviates from current frequency and monetary to a specified degree, the CFM-tree is dynamically restructured by current frequency and monetary and the list updates the pattern order with the current list. The sequential patterns obtained from the updated sequences are incremented based on timestamps, monetary value and the frequency of each patterns. While updating the tree structure, CFM-tree constantly maintains initial sort order of sequential patterns with their information. Thus, it adds new frequent items at the end of a list and it constructs to maintain the frequency of each item and in tree structure as new nodes. The information about frequency and monetary value should be updated in a timely manner. The timestamp of sequence in the child node should be updated as new one. This is reasonable because for every element between old timestamp and new one, they are already appended to this node as a candidate sequential pattern with old timestamp. Thus, sequential patterns between old timestamp and new one can be found. Additionally, for elements after the new timestamp, appending them to the node having sequence with new timestamp is the only way to find up-to-date sequential patterns beginning at the new timestamp.

Example: By considering the updating nodes of the CFM-tree, the newly inserted items are arrived in a periodic manner. Here, in timestamp T6, the items '<h>' and '<cb>' are the new set of items. We have to update these into the existing CFM-tree dynamically. While updating '<h>', the progressive compact sequence obtained are <ch>, <(bc)h>, <c(bc)h>, <(bc)(ae)h>, <cbh>, <cch>, <ccah>, <cceh>. These patterns are updated sequentially into the CFM-pattern tree along with the information about the frequency and the monetary value of the updated nodes. Similarly, the other updated sequence, '<cb>' is also updated in the CFM-pattern tree and form the updated CFM-tree. The updated CFM-tree with timestamp T6 is given in Figure 6.2.

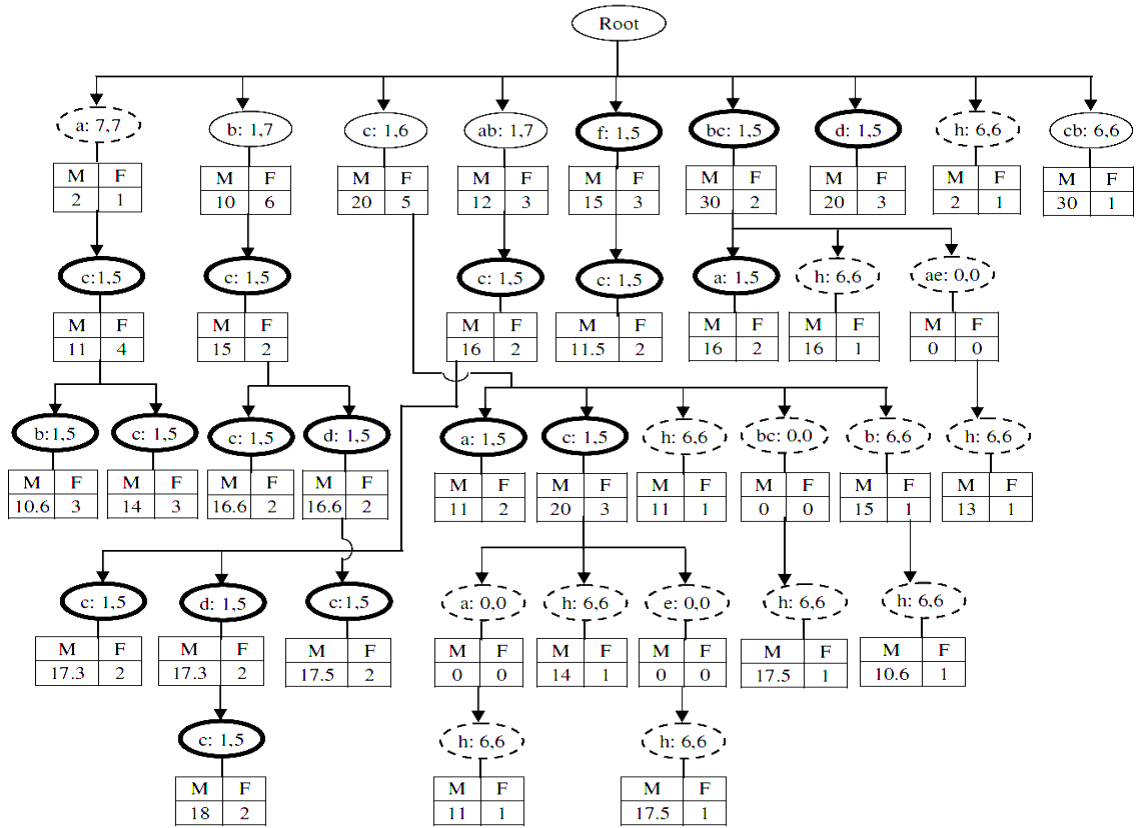


Figure 6.2 Updated CFM Tree with newly inserted items

In the CFM-tree, the newly updated node to the root node is marked as in dotted line, whereas the update process is done in the existing nodes is indicated as a thin line and the dark line represents the nodes in which there is no update is carried out. For mining the progressive CFM patterns, we have used the user specified thresholds, ($\min_sup \geq 1$, $T_c \leq 4$, $T_m \geq 10$).

6.4.4. Step 4: Handling the Node Deletion Operation in the Updated CFM-tree

On mining progressive CFM sequential patterns, the newly arrived patterns may not be identified as frequent one if static database is a larger one. It is noted that users are usually more interested in recent data than old ones. So, deletion of an item from CFM-tree is carried out utilizing time information stored in every node. Thus, incompact nodes and the non-zero infrequent nodes should be deleted from the final updated CFM-tree.

Example: To delete obsolete sequences, timestamp stored with each node is considered. We have deleted the incompact nodes, which don't satisfy the user specified threshold, where there is no update process are carried out. As well, we have removed

the non-zero infrequent nodes in which the frequent value is less than the threshold. The CFM-tree with no incompact nodes is shown in Figure 6.3.

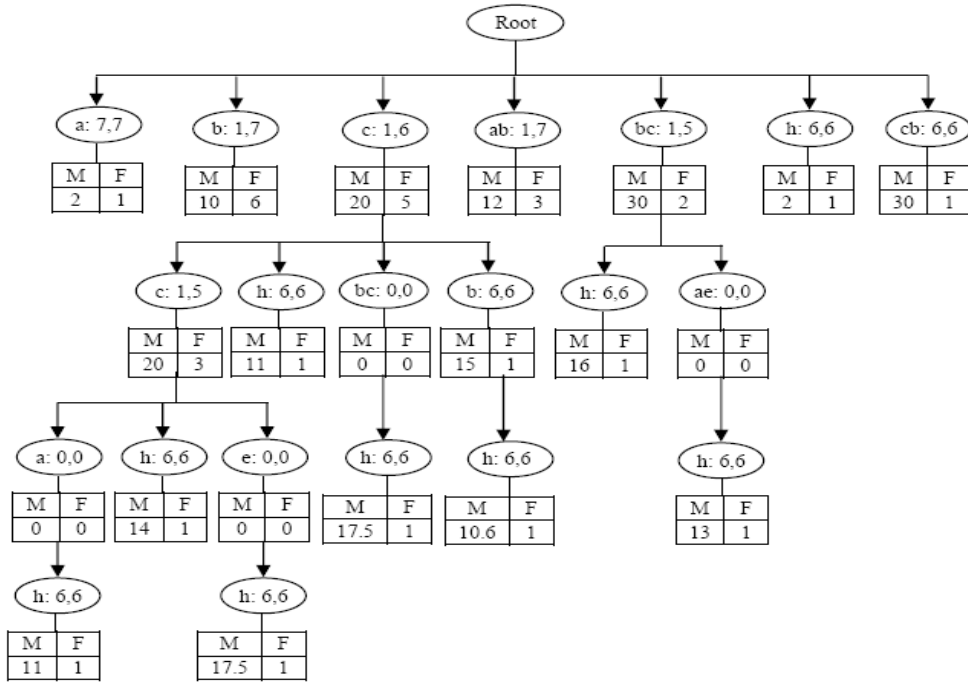


Figure 6.3 Updated CFM -tree with no Incompact Nodes

6.4.5. Step 5: Mining of CFM Patterns from the Progressive Database

After the construction of updated CFM-tree, the progressive CFM patterns are mined from it based on the user specified thresholds. Here, tree pattern mining is done that uses top-down process to mine CFM-patterns. The mining process is started from the top nodes of the CFM-tree and their corresponding paths are extracted from it. Then, by combining the nodes of each level, the progressive CFM patterns are obtained.

Example: From final updated CFM-Tree shown in Figure 6.4, one of the top nodes $\langle bc \rangle$ and its corresponding paths are extracted. From the paths, each level of nodes are combined so that progressive CFM patterns, $\{\langle bc \rangle, \langle bc \rangle h, \langle bc \rangle (ae) h\}$ are obtained. Figure 6.5 (a) shows extracted path for node $\langle bc \rangle$ and each level with previous level are combined so that CFM patterns can be achieved, shown in Figure 6.5 (b), 6.5 (c) and 6.5 (d). The mined sequential CFM-patterns for all top nodes are given in the Table 6.4 below.

CFM-patterns	
	
<c>	<c>, <cc>, <ch>, <cb>, <c(bc)h>, <cbh>, <ccah>, <cch>, <cceh>
<(ab)>	<(ab)>
<(bc)>	<(bc)>, <(bc)h> <(bc)(ae)h>
<(cb)>	<(cb)>

Table 6.4 Final progressive CFM-patterns

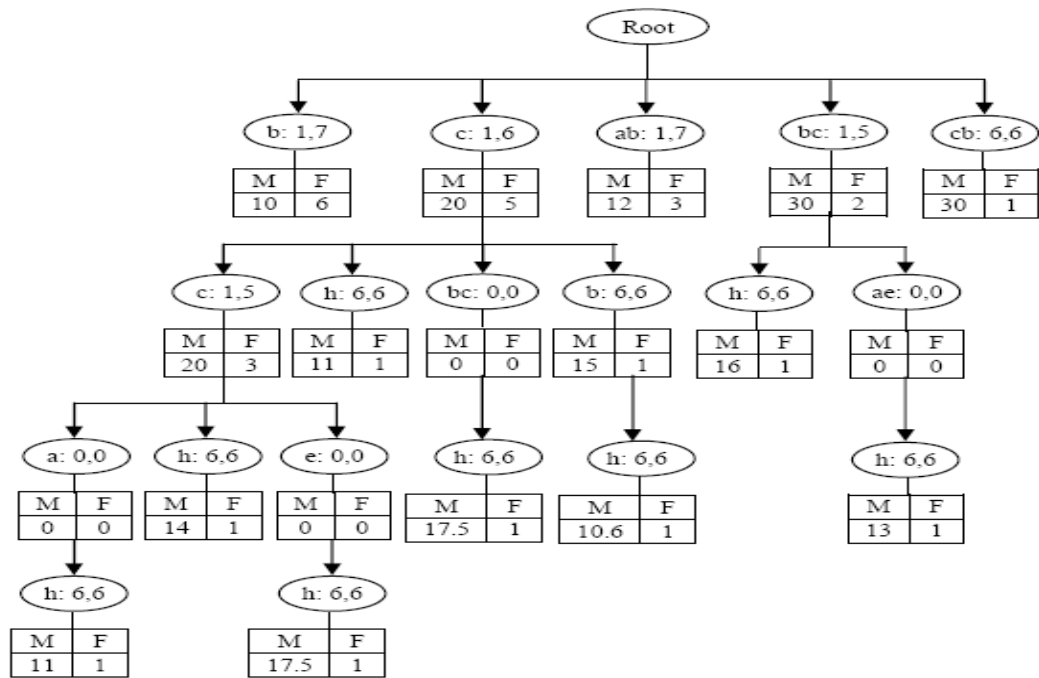


Figure 6.4 Final updated CFM-tree for progressive database

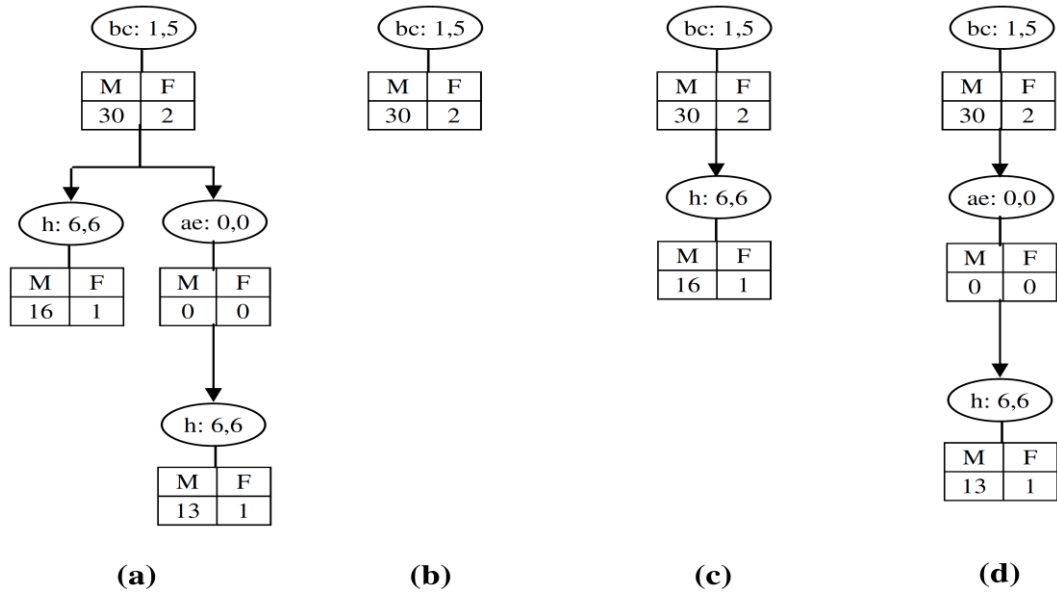


Figure 6.5 Mining of progressive CFM Patterns from the Updated CFM-tree

The pseudo code for the proposed procedure for mining the progressive CFM-patterns is given as follows.

6.5. PSEUDO CODE

Input: CFM-tree, min_sup, T_m

Output: A complete set of Progressive CFM patterns

Assumptions:

- $m \rightarrow$ Number of nodes (next to the root node) in the constructed CFM-tree
- $\text{min_sup} \rightarrow$ Minimum support threshold
- $S_pat \rightarrow$ Sequential pattern
- $PCFM_pat \rightarrow$ Progressive CFM-patterns
- $k \rightarrow$ Number of distinct paths
- $D \rightarrow$ Depth of the path
- $pi \rightarrow$ Item information in the node

begin

for each node 'm' in CFM tree

for ($j = 1; j < k; j++$)

$d[j] = \text{distinctpath.CFM_tree}$

do_miner ($d[j]$)

```

        if ( support(S_pat[l]) ≥ min_sup & Tm)
            PCFM_pat << S_pat
        endif
    endfor
endfor
end

subroutine: do_miner (d[j])
begin
    p.l = top node.d[j]
    S_pat << p.l
    for ( i = 1 ; i < D ; i++)
        p.(i+1) = p.i || p.(i+1)
        S_pat << p.(i+1)
    endfor
end

```

6.6. EXPERIMENT SETUP AND DATASET DESCRIPTION

The experiment has been carried out on a 2.9 GHz, dual core PC machine with 1 GB main memory running a 32-bit version of Windows XP for Progressive CFM-Miner and IncSpan algorithms. The proposed incremental mining algorithm has been designed so that it can execute in a distributed environment, which means the updating of data records can be done from multiple sources. So, the algorithm is executed in thread environment, in which the updating of data records is done in various threads.

Datasets: The performances of the algorithms have been evaluated using the synthetic datasets as well as real life datasets (Appendix). **Synthetic dataset:** A set of synthetic data sequence is generated by a data generator which is designed for testing sequential pattern mining algorithms. It has been implemented using the concept of IBM data generator. Each data sequence contains a sequence of item sets and different time values are assigned to the items in different item sets. However, same time value is assigned to items which are in the same item sets. A sequential database that contains 1000 records with 4 transactions is used for the experiment. **Real life datasets:** The UCI machine learning repository (<http://archive.ics.uci.edu/ml/datasets>) is used. This data describes the page visits of users who visited msnbc.com on September 28, 1999. Visits

are recorded at the level of URL category in time order ("frontpage", "news", "tech", "local", "opinion", "on-air", "misc", "weather", "health", "living", "business", "sports", "summary", "bbs" (bulletin board service), "travel", "msn-news", and "msn-sports"). The real dataset with 1000 records of 10 transactions is used for the experiments.

6.7. RESULT ANALYSIS

The experimental results of the proposed algorithm for mining of progressive CFM-sequential patterns are described in this section. The experimental results are compared with well-known incremental IncSpan algorithm. The progressive CFM-Miner and IncSpan algorithms were implemented using Java language (jdk 1.6).

The Table 6.5 shows the patterns mined by the proposed algorithm and the Incspan algorithm. The Incspan algorithm produced the rules based on frequency of the items. The proposed algorithm mines the most desired and useful sequences (as discussed in Section 3.3. and Section 3.4) compared with the previous algorithm. This is due to the incorporation of the constraints in the Progressive CFM-Miner algorithm. From the table, most of the sequences containing the 1-length patterns such as and <c> are mined from the database using the proposed algorithm. But, the IncSpan algorithm mined the sequences of having the patterns of <a>, <d>, <g> and <e> as well. When analyzing these patterns, it can be identified that <h> and <a> have less monetary value. On the other hand, <d>, <g> and <e> are not recently frequent items.

Progressive CFM-patterns		Incspan Algorithm	
		<a>	(ab), ab, ac, (ac), ad, af, (ab)c, (ab)d, (ab)f, aba, aca, acc, adc,
<c>	<c>, <cc>, <ch>, <cb>, <c(bc)h>, <cbh>, <cceh>, <cch>, <cceh>		(ba), ba, (bc), bc, bd, bf, (ba)c, (ba)d, (ba)f, (bc)a, bcd, bcc, bcf, bdc.
<(ab)>	<(ab)>	<c>	ca, cb, cd, cf, cdc.
<(bc)>	<(bc)>, <(bc)h> <(bc)(ae)h>	<d>	db, dc.
<(cb)>	<(cb)>	<f>	fc, fb, fcb.

Table 6.5 Comparison of algorithms in mining of meaningful sequences

In next section, the performance analysis of the proposed algorithm is done to justify its efficiency which is due incremental mining of sequential patterns from progressive database.

6.8. PERFORMANCE ANALYSIS

The performance of the proposed CFM-Miner algorithm for sequential pattern mining from progressive database is evaluated by three standard evaluation measures. They are (a) number of sequential patterns, that is, the significant number of sequential patterns generated based upon the given minimum support threshold, (b) execution time, that is, the time taken to execute the computer program and the (c) memory usage, that is, the memory utilized by the current jobs present in the particular system. The proposed algorithm is compared with the well known incremental algorithm, IncSpan using both the synthetic and the real life datasets.

6.8.1 Effect of support values

The experiment results are plotted using graphs from Figure 6.6 to 6.8. Here, the input sequences have been varied in certain time intervals. The generated number of sequences shows better results in our proposed approach as given in Figure 6.6 for both synthetic and real datasets. However, the execution time of the CFM-Miner gets slightly slipped down in some cases than the IncSpan algorithm for synthetic datasets. It maintain the same trend for real datasets by giving better results than the IncSpan algorithm as obvious from graph of Figure 6.7. The effective usage of the memory is given by the proposed algorithm for both types of datasets as shown in Figure 6.8.

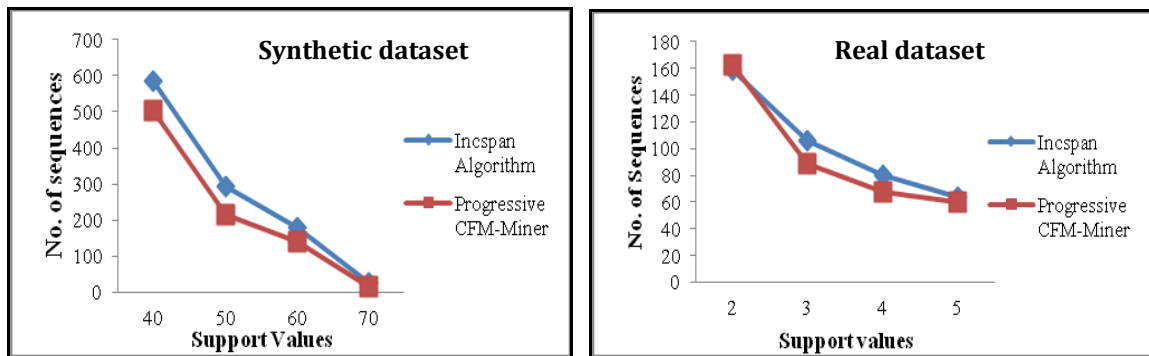


Figure 6.6 Number of sequences generated by Progressive CFM-Miner and IncSpan with different support values for real and synthetic databases.

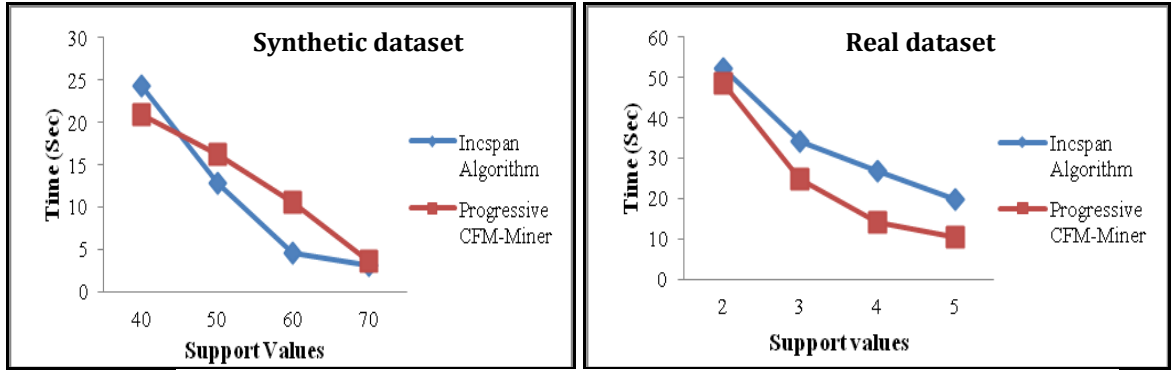


Figure 6.7 Computation time for Progressive CFM-Miner and IncSpan with different support values for real and synthetic databases.

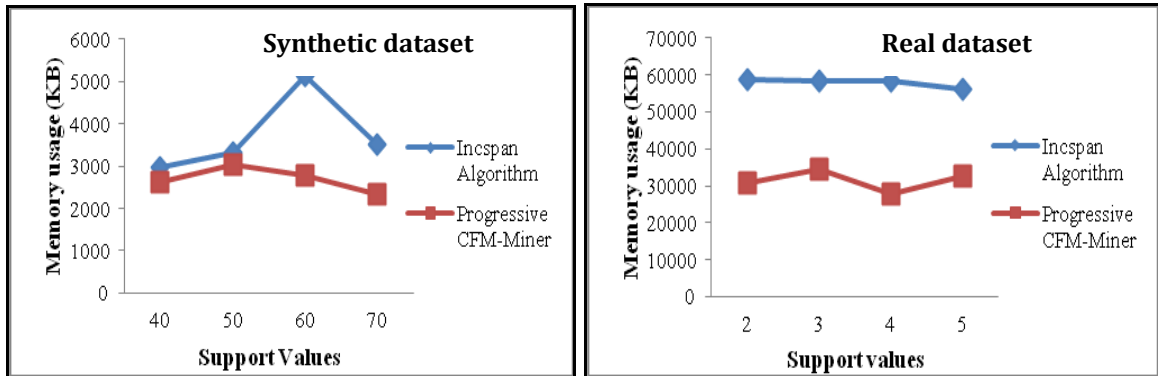


Figure 6.8 Memory usage of Progressive CFM-Miner and IncSpan with different support value for real and synthetic databases.

6.9. SUMMARY

An efficient, Progressive CFM-miner algorithm is proposed in this Chapter to handle the maintenance problem of CFM-sequential patterns. An updated CFM-tree is built using the CFM- sequential patterns obtained from the static database to control the dynamic nature of data updating process and deletion process into the sequential pattern mining problem. Subsequently, the database gets updated from the distributed database that may be static, inserted, or deleted. Whenever the database is updated from the multiple sources, CFM tree is also updated by including the updated sequence. The updated CFM-tree is used to mine the progressive CFM-patterns using the proposed tree pattern mining algorithm. Eventually, the experimentation is carried out using the synthetic and real life datasets that are given to the progressive CFM-miner using thread environment. The experimental results and analysis provides better results in terms of the evaluation measures over the well –known IncSpan algorithm.

The application of this algorithm is discussed in Chapter 8.

Chapter 7

Incorporating Fuzziness in Incremental Mining of Sequential Patterns⁷

7.1. INTRODUCTION

Sequential pattern mining is new but an interdisciplinary field utilizing statistics, machine learning, and other methods. In recent years, fuzzy logic has also been applied to augment pattern mining. The application of fuzzy logics makes mining results more understandable and interpretable, apart from being useful and informative. Fuzzy rules are useful to summarize large databases as they mine rules with timing information, called fuzzy sequential patterns.

The main challenge for the end users who work with large databases is to retrieve concise and understandable summaries. The use of fuzzy logic can help to extract and maintain rules and summaries from huge databases. This is achieved by first deriving linguistic summaries for the database and extending it further to fuzzy summaries. The user interaction is also useful to get interesting information along with different methods. The use of lattice structure for the summarization of fuzzy sequential patterns is proposed. The different item sets of a sequence can be hierarchically grouped together satisfying properties of mathematical lattice structure and hence can be used for knowledge extraction.

Business and real-time databases mostly have numerical data that is time-stamped. In recent times, the fuzzy set theory utilization has reduced bleak cuts over the period, and hence provides more relevant rules. An efficient algorithm, PLM (ProgresLattice Miner) for incremental mining of fuzzy sequential pattern mining from progressive database is proposed. The procedure is not directly carried out from

⁷ The major findings in this Chapter have been (i) published in International Journal of Hybrid Technology, 6(3), May 2013 and (ii) submitted for publication in International Journal of Uncertainty, Fuzziness and Knowledge System.

progressive database, since it takes more time due to scanning required to find the support. Instead, a ProgresLattice structure is built that utilize standard lattice structure to reduce the scanning time required to mine the patterns effectively. The mining procedure gives better computation time as only a single scan is required to build the lattice structure. Particularly, some approaches were proposed to extract fuzzy sequential patterns within the historically stamped quantitative data (Chen *et al.*, 2001; Hu *et al.*, 2004; Fiot *et al.*, 2006b; Huang *et al.*, 2010). However, to the best of our knowledge, there is no study made for incremental mining of fuzzy sequential patterns from progressive databases.

7.2 MOTIVATION FOR THE USE OF FUZZY SEQUENTIAL PATTERNS

This has been noticeably observed that the real-world databases consist of numerical and time-stamped data. We can get more relevant rules from these databases by making use of fuzzy set theory to minimize the sharp cuts between intervals. The sequential pattern mining is based on binary valued transaction data. This should however be extended to fuzzy sequential pattern mining for quantitative valued data. Fuzzy sequential mining generate simple and practical patterns which are close to human reasoning. The term fuzzy was introduced by Zadeh (1965) and further studied to state that there could be additional zone apart from only true and false. The concept of fuzzy provides flexibility to model imprecisely defined conditions. It allows for approximate reasoning that is useful for expert systems with powerful reasoning capabilities. In fact, the logic behind any thought process is hardly two valued but based on imprecise and unclear truths and rules of inference. A fuzzy set theory is a logical extension of a crisp set. Crisp sets have only two values, 0 or 1, that means an object may have no membership or complete membership. On the contrary, fuzzy set theory allows for any value between 0 and 1 and an object may have partial membership. This is done by introducing gradual memberships for the quantitative data using membership functions. Therefore, each quantitative item has to be partitioned into several fuzzy sets to mine fuzzy sequential patterns (Chen *et al.*, 2001). This redefines the concept of attribute and itemset as compared to classical sequential patterns. The association of one item (attribute) and corresponding fuzzy set formulates the fuzzy item.

Example: Table 7.1 has 4 sequences of customer transactions showing purchases of hardware, software and accessories items from a computer shop.

Sequence Identifier	Sequences
S100	< PC (Windows Media Player, Video Card, Sound Card)(Barcode reader, Printer) Quick Heal (Loudspeaker, CD-ROM)>
S200	<(PC, Windows 7) Adobe Reader (Logical Disk Manager, Oracle) (Digital Camera, Microphones)>
S300	<(PC, OS/2.1x) (Adobe Flash Player, Sound Card) (Windows Media Player, Video Card) JExpress,Printer >
S400	< PC, Network Card (Windows Media Player, Video Card) McAfee Internet Security, Acronis Disk Director, LHMelt >

Table 7.1 Sequence Database ‘SD3’

The software purchased by the customer can be modeled with [number, low] as a fuzzy item where low is a fuzzy set defined by a membership function (explained in detail in Section 7.3) on the software universe of possible values of the item ‘number’. The list of fuzzy items is called a fuzzy itemset denoted as a pair of sets (set of items, set of fuzzy sets associated to each item) (Hu *et al.*, 2003). To elaborate, ([number, low][quantity, small]) is a fuzzy itemset having two fuzzy item ‘number’ and ‘quantity’ for quantitative items ‘software’ and ‘hardware’ respectively. One fuzzy itemset contains only one fuzzy item related to one single attribute. A s-f-sequence $S = \langle s_1 \cdot \cdot \cdot s_g \rangle$ is a sequence constituted by ‘s’ fuzzy itemsets $s = (X, A)$ grouping together ‘f’ fuzzy items [x, a]. The sequence $\langle ([number, low][quantity, small]) ([accessories, many]) \rangle$ groups together 3 fuzzy items into 2 itemsets. It is a fuzzy 2-3-sequence.

7.3. FUZZY SEQUENTIAL PATTERNS SUMMARIZATION

Mining fuzzy rules is one of the best ways to summarize large databases while keeping information as clear and understandable as possible for the end-user. The common approach to express such knowledge consists in deriving linguistic summaries, which can further be extended to fuzzy summaries. Such summarization often requires a user-interaction for quality and validity in order to select interesting and useful knowledge from the huge datasets. There are few methods based on functional dependencies or

association rule mining that perform summarization using automatic generation. However these methods are useful for quantifying and reasoning (Hu *et al.*, 2004). For applications where it is vital to mine rules that express information about the order fuzzy sequential patterns summarization is helpful.

Initially, the fuzzy partitions are created for each of the numerical attributes in the given crisp dataset. Then, using these fuzzy partitions, the fuzzy version of the dataset are created by converting crisp numerical attributes and its associated numerical values into fuzzy attributes and its associated values/ membership degrees. Fuzzy sequential pattern mining is a significant approach, which deals with temporally annotated numerical data (Chen and Hu, 2002). It allows mining of frequent sequences embedded in the records. However, such fuzzy sequential patterns, in their current form, do not allow extracting temporal tendencies that are typical of sequential data (Fiot *et al.*, 2006a). We can elaborate this by considering the example Table 7.1, and deriving the fuzzy sets for it. Table 7.1 has been parsed to see the frequency of purchases made for the hardware, software and accessories items from a computer retail shop. The results derived from this parsing are being shown in Table 7.2. The hardware item/s that cannot be placed on the slots of the motherboard has been considered as accessories.

Sequence Identifier	Hardware	Software	Accessories
S100	4	2	3
S200	1	4	4
S300	3	4	1
S400	3	4	0

Table 7.2 Parsing results of Sequence Database ‘SD3’

The reduction of item sets in the mining process is possible due to the role of fuzzy sets that transform quantitative values into linguistic terms. An object or item may either belong to a particular set or not, in case of classical set theory. However, the fuzzy set theory makes it possible that the object can even belong to a set to a certain degree. This is achieved by using the linguistic knowledge for the property that defines the set.

Membership degrees indicate resemblance, ambiguity and inclination of an object with the corresponding set. The quantitative attributes are first divided into fuzzy sets using linguistic terms like small, moderate, medium, large etc.

We further use the membership functions to get the membership degrees for each attribute and fuzzy set. There are different types of membership functions that can be used like triangular, trapezoidal, bell shaped, gaussian curves, polynomial curves, sigmoid function. By making use of membership function and linguistic knowledge, the fuzzy sets for each attribute are formulated, as given in Figure 7.1 (a), (b) and (c) for our example of Table 7.2.

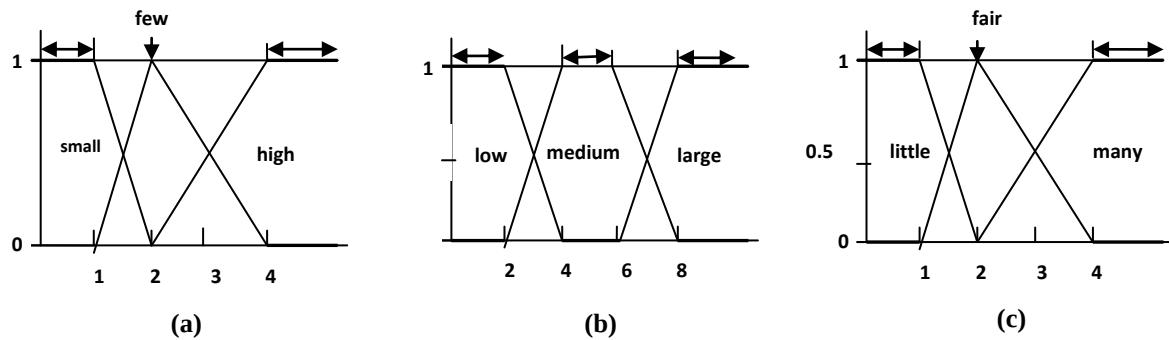


Figure 7.1 Fuzzy sets for hardware, software and accessories itemsets

As shown above, a membership function is a curve to represent each point of the input space with a membership value, that is, degree of membership between 0 and 1. As an example, consider fuzzy sets small, few and high for universe of discourse hardware with attribute as 'quantity'. With a crisp set, hardware quantity of '0', '1' is considered as 'small', '2' as 'few' and any quantity '4' and 'above' is considered as 'high'. However, it is not clear that the hardware quantity of 3 purchased by customer is to be treated as which type of set. For a given membership function the fuzzy set is shown in Figure 7.1 (a). The curve gives the transition from few to high quantity of hardware with the membership degree. Figure 7.1 (b) and 7.1(c) shows the fuzzy sets for 'number' attribute of software and accessories respectively.

The fuzzy membership function is applied according to the measurements of spatial relations in order to examine and compare the resemblance between mining result and real spatial configuration. Subsequently, fuzzy comprehensive judgment of mining results is conducted based on correlative fuzzy theory. Usually the membership functions

are selected by the user with his experience or sometimes even randomly. Therefore, the membership function chosen by two users could be different depending upon their experiences, perspectives and more. They can be also designed using machine learning methods like artificial neural networks, genetic algorithms. The following factors are considered to determine fuzzy membership functions,

- i. The membership function defines the fuzzy set.
- ii. A measure of the degree of similarity of an object to its corresponding fuzzy set is given by the membership function.
- iii. Membership functions are of different types and forms.

Fuzzy sets can be used to (i) illustrate the relationship between data (ii) represent different types of associations (iii) facilitate to give mining rules in linguistic terms and (iv) prevent abrupt boundaries when dividing the attribute domains. It is obvious that an element can belong to more than one fuzzy set at a time. So, a fuzzy set 'A' in universe 'U' may be represented as a set of ordered pairs. Each pair consists of a generic element 'E' having a linguistic variable 'x' and its grade of membership 'm'. For given sequential database 'D', {S100. high, 1} shows that sequence identifier 'S100' has '1' degree membership for fuzzy set 'high', and {S300.few, 0.5} represents 'S300' has a membership degree of '0.5' for fuzzy set 'few' for hardware. The membership degrees of all attributes for customer sequences are summarized Table 7.3.

Sequence Identifier	Hardware= quantity			Software= number			Accessories		
	small	few	high	low	medium	large	small	few	high
S100			1	1				0.5	0.5
S200	1				1				1
S300		0.5	0.5		1		1		
S400		0.5	0.5		1		1		

Table 7.3 Membership degree of customer sequences for fuzzy sets

7.4. ROLE OF LATTICE STRUCTURE

Lattice is an important concept of discrete mathematics. The lattice is characterized by specific properties which makes it quite different from other data structures like graph, tree (Kumar *et al.*, 1995). We can make use of this structure to represent and maintain the

fuzzy sequential patterns. The typical definition of the lattice as given by mathematical theory is:

Let $(P, \sqsubseteq)$ be a partially order set and $A \subseteq P$ a subset of P . An element $p \in P$ is called an upper bound for A if $a \sqsubseteq p$ for all $a \in A$. It is called a lower bound for A if $p \sqsubseteq a$ for all $a \in A$. If the set of all upper bounds of A has a smallest element, then this element is called the join or supremum or least upper bound of A . Similarly the largest lower bound of A (if it exists) is called the meet or infimum or greatest lower bound of A . For example, if we consider the set N of natural numbers with order relation $|$, “is a divisor of”. Then the supremum of two elements $a, b \in N$ equals $\text{lcm}(a, b)$. The greatest common divisor $\text{gcd}(a, b)$ is the infimum of $\{a, b\}$. A partially ordered set or poset $(P, \sqsubseteq)$ is called a lattice, if for all $x, y \in P$ the subset $\{x, y\}$ of P has a supremum and an infimum. The supremum of x and y is denoted by $x \sqcup y$ and the infimum as $x \sqcap y$. We can say, $(R, \leq)$ is a lattice, if $x, y \in R$, then $\sup\{x, y\} = \max\{x, y\}$ and $\inf\{x, y\} = \min\{x, y\}$. Similarly, if S is a set and $P = P(S)$ the poset of all subsets of S with relation $\subseteq$ then P is a lattice with $\sqcup = \cup$ and $\sqcap = \cap$. As stated before, if $(P, \sqsubseteq)$ is a lattice, then for all $x, y, z \in P$ has certain properties (Huaiguo *et al.*, 2008) as;

- i. $x \sqcup x = x$ and $x \sqcap x = x$; (Reflexive)
- ii. $x \sqcup y = y \sqcup x$ and $x \sqcap y = y \sqcap x$; (Symmetric)
- iii. $x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z$ and $x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$; (Transitive)

A lattice in which every subset has a supremum and infimum, is called a complete lattice. We can represent this visually by means of directed acyclic graph. This graph has nodes which represents elements of the poset and there is directed arc from node y to node x if and only if $y \sqsubseteq x$. This type of graph is called as Hasse diagram that is used to represent a poset (Zhang *et al.*, 2008b). Usually the direction of arcs in the graph are avoided by showing node x above node y if $y \sqsubset x$. So, the lattice can be used to represent the order relation or the hierarchical order of elements.

This data structure could be useful for the fuzzy summarization of the sequential pattern once the fuzzy set has been formulated using the linguistic knowledge and each data element is associated with the membership degree with these fuzzy sets using the membership function. To explain this let us take the example as discussed in Section 7.3. Considering a domain of discourse in which each element of a set of sequence of

transaction, $SID = \{s_1, s_2, \dots, s_n\}$ have one or more attributes $ATT = \{a_1, a_2, \dots, a_m\}$. We have triple $F = \langle SID, ATT, M \rangle$, where M is a membership degree attained by a particular sequence identifier for a specific attribute. This membership degree is derived by using some membership function. For example in Table 7.3, we can define a database $D = \langle P, \sqsubseteq \rangle$ related to triple F as consisting of a set P , of sequences of transactions which are partially ordered by the relation $\sqsubseteq$. The database is restricted as the attributes (ATT) in F form the maximal elements and the sequences of transactions (SID) in F are the minimal elements. We can represent this database as in Figure 7.2.

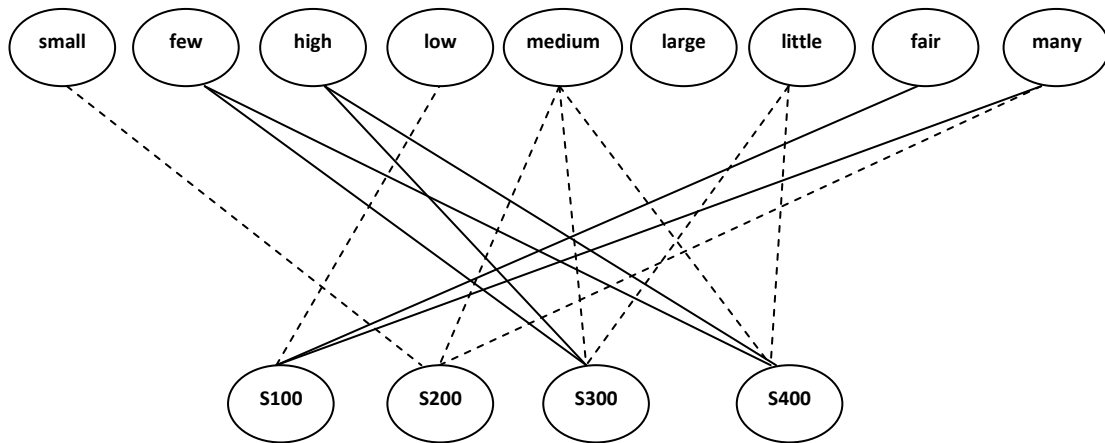


Figure 7.2 Graph summarizing the sequences with the fuzzy sets

However the above graph has a key limitation, that is, when the ascending paths of two different nodes are followed, it is not necessary that we will reach to one single common node at their top. For instance, by ascending path from 'S300' and 'S400' we reach to node 'few' as well as 'high'. It is very unclear and ambiguous that whether the sequence 'S300' has feature of attribute 'few' or 'high'. In fact due to fuzziness it has features of both the attributes that is 'few' and 'high' to certain membership degree. This is not reflected by the use of the graph.

Making use of lattice structure, P , with a property that every two elements of P have a least upper bound, called join or supremum and greatest lower bound known as meet or infimum we can take care of this problem. This typical feature that there would be always one single supremum for any two nodes in the lattice is termed as a closure property of lattice. This often may require adding some extra nodes to achieve the consistency of the system. The lattice theory ensures that these extra nodes, added as 'artificial' supremum of two closed sets will consist on the union of the maximal

sequences that are contained by their immediate predecessors. We can have following lattice structure represented using the Hasse diagram in Figure 7.3 for the Table 7.3 with added nodes;

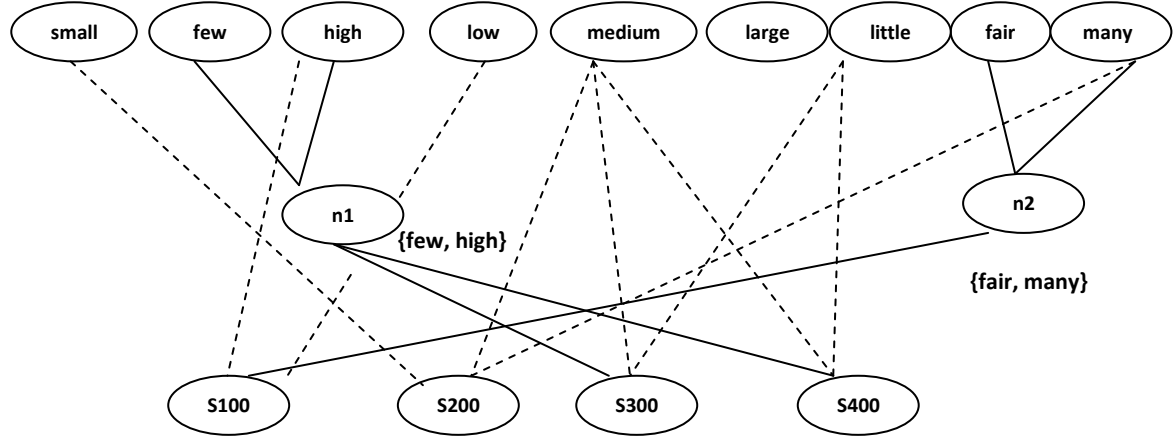


Figure 7.3 Lattice structure summarizing the sequences with fuzzy sets

The fuzziness in the sequences for attributes is reflected by using the lattice structure. It has good properties involving the completeness of pairs and independence of the order of input variable or the attributes. No doubt, lattice structure is a robust tool that can be utilized for data analysis and knowledge discovery.

7.5. PROGRESLATTICE MINER (PLM): PROPOSED ALGORITHM FOR INCREMENTAL MINING OF FUZZY SEQUENTIAL PATTERNS

This section proposes an algorithm, ProgresLattice Miner (PLM) which integrates the concepts of fuzzy sets and incremental mining to find interesting sequential patterns from dynamic progressive transaction database. The limited number of existing algorithms for sequential pattern mining from progressive database cannot cope for quantitative data. This is the first study (to the best of our knowledge) to find the fuzzy sequential patterns from the progressive database. The algorithm use lattice structure to store data along with frequency and support count. The following sections discuss the algorithm in detail.

7.5.1. Preliminaries

The formal definitions of some of the most relevant terms are presented below;

- i. **Fuzzy item:** It is the association of one item and one fuzzy set. It is denoted by $[x, a]$, where x is the item (also called attribute) and a is the associated fuzzy set. For example, $[\text{length}, \text{short}]$ is a fuzzy item where short is a fuzzy set that is given by a membership function on the quantity universe of the possible values of the item length.
- ii. **Fuzzy itemset:** It is a set of fuzzy items. It can be denoted as a pair of sets (set of items, set of fuzzy sets associated to each item) or as a list of fuzzy items. We will note, $\{X, A\} = ([x_1, a_1], \dots, [x_p, a_p])$, where X is a set of items, A is a set of corresponding fuzzy sets and $[x_i, a_i]$ are fuzzy items.
- iii. **Lattice:** A partially ordered set (poset) is a set L with a reflexive, symmetric, transitive relation. A poset L is a complete lattice if every subset $S \subseteq L$ has a least upper bound $\vee S$ (join) and a greatest lower bound $\wedge S$ (meet). A poset L is a lattice if every two elements of L have a join and a meet. These meets and joins of binary sets will be written in infix notation, $\vee(x, y) = x \vee y$ and $\wedge(x, y) = x \wedge y$.
- iv. **Lattice support L_s :** Let I be the set of items and $v(i)$ be the value of attribute i in record. Each attribute i is divided into fuzzy sets. Then one record in a fuzzy sequence database consists of the membership degrees of each attribute to each fuzzy set, e.g. $r(x, a) = f(x)$ represents the membership degree of item/attribute x to the fuzzy set a in record. The support of a fuzzy sequence S is then computed by the formula: $L_s = F / N$, where F is the fuzzy membership value and N is the number of records. The maximum operator is used to calculate lattice support for an item with more than one membership degree in a record.
- v. **ProgresLattice:** It is a data structure used in the proposed algorithm to mine the fuzzy sequential patterns from progressive database. We make use of the traditional lattice structure which, hold much good properties and utilized it for mining frequent patterns, including the completeness of pairs in the lattice, the independence of the order of input variables or attributes, and the convenience of combining with domain knowledge.
- vi. **PLM (ProgresLattice Miner):** It is a mining technique for fuzzy sequential pattern mining from ProgresLattice. It has the ability to be applied to various fields.

- vii. **Fuzzy sequential pattern:** It is a type of knowledge representation with fuzzy occurrence order information of items. It can also be defined as the fuzzy rules that describe the evolution of data over time.
- viii. **Positive Fuzzy sequential pattern:** A fuzzy sequential pattern is called as a positive fuzzy sequential pattern when it expresses only the occurrences of the fuzzy itemsets.
- ix. **Negative Fuzzy sequential pattern:** A fuzzy sequential pattern is called as a negative fuzzy sequential pattern when it also expresses the absences of fuzzy itemsets.

7.5.2. Phases of the Algorithm

The proposed PLM (Progreslattice Miner) is mainly achieved in the following three phases as represented with the model given in Figure 7.4.

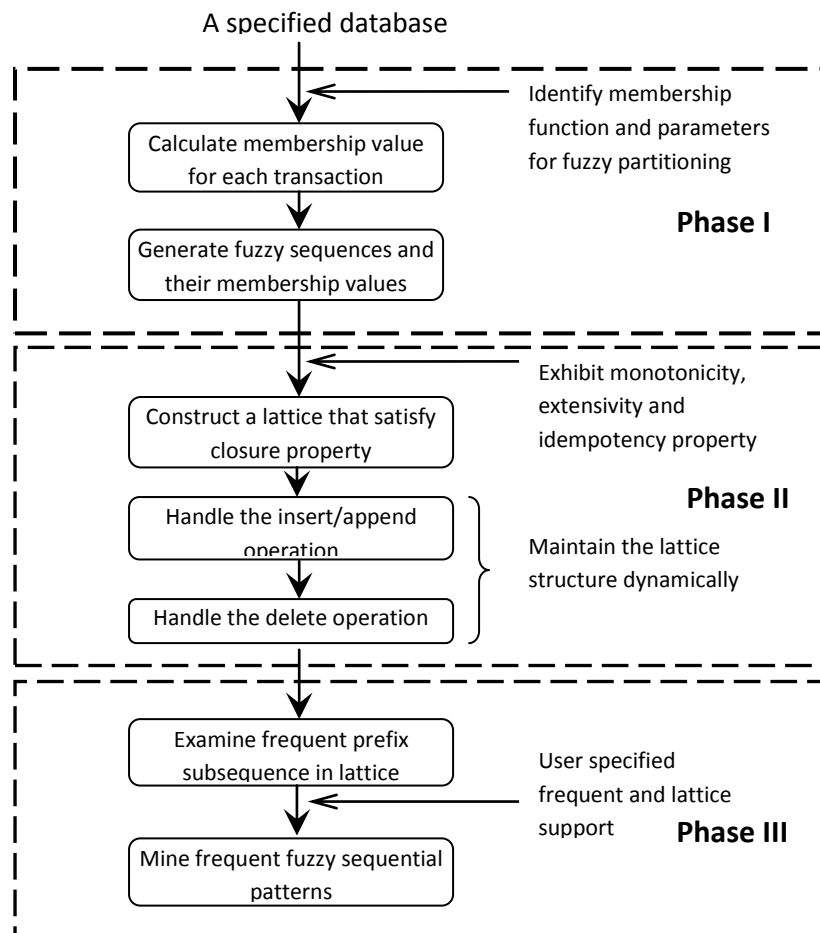


Figure 7.4 Model for ProgresLattice Miner Algorithm

This fuzzy sequential pattern mining algorithms is achieved using three main phases:

- i. Finding fuzzy membership value for each transaction of progressive database.
- ii. Building a ProgresLattice structure with the aid of standard lattice structure.
- iii. Mining of fuzzy sequential patterns from ProgresLattice structure.

7.5.3. Stepwise execution

The detailed steps of algorithm for the proposed ProgresLattice Mimer for mining fuzzy sequential patterns are;

Algorithm: ProgresLattice Miner algorithm (PLM)

Input: a) A specified database, DB.

b) User specified minimum fuzzy support and threshold value.

Output: Phase I: Fuzzy membership values of the transaction.

Phase II: Progreslattice data structure.

Phase III: Fuzzy sequential patterns.

Method:

Phase I: Finding fuzzy membership values of transactions on progressive database.

Step 1. Identify the membership function to transform transactions of progressive database.

Step 2. Define the fuzzy partition set for each of the numerical transaction value.

Step 3. Identify and formulate the parameters for fuzzy partition set using linguistic terms.

Step 4. Calculate membership value for each transaction using the membership function identified in Step 1.

Step 5. Associate membership values of each transaction with fuzzy partitions using linguistic terms.

Step 6. Goto Phase II with the computed fuzzy sequences and their membership values.

Phase II: Buiding a Progreslattice structure with the help of standard lattice structure.

Step 7. Construct a lattice such that it satisfies

- a) the closure property, that is, any two nodes must have single supremum. So, by following the ascending paths of two different nodes we will reach one single common node on the top. It operates as follows: given DB , the closure of a set of

sequences S , i.e. $\Delta(S)$, includes all the maximal sequences that are present in all the transactions $db \in DB$ where sequences in S are contained.

- b) add some extra nodes to ensure the fulfilment of Step 7a) and the consistency of the system. These extra nodes added as artificial supremum of two closed sets will consist on the union of the maximal sequences contained by their immediate predecessors. This avoids to scan the input sequences in DB for completing the lattice.
- c) the property of monotonicity, that is, preserve the order which is vital for fuzzy sequential pattern.
- d) of being extensive, that is, it covers all transactions and is completely thorough.
- e) of being idempotent with the addition and deletion of nodes.
- f) to store the frequency and lattice support values by each node. The maximum operator is used to calculate the lattice support when a node has more than one membership degree.

Step 8. The lattice will exhibit the tendencies of the data from top to bottom: nodes located in the top part of the lattice correspond to concepts/patterns with a support, thus the semantic split performed between the concepts of the same level is more significant. On the other hand, the nodes located in the bottom of the lattice represent the unifying concepts of the different tendencies coming from the top.

Step 9. Handle the updates from the existing sequence (append) in the lattice structure:

- a) Consider the frequency and the lattice support to insert the new node in the lattice structure at an appropriate level, maintaining the length of the subsequent subsequence.
- b) While creating the subsequent node of this sequence, increment the frequency with the existing same set of sequence nodes and update their lattice support.
- c) Update the frequency of fuzzy sequence in the Progreslattice, if this appended sequence already exists. However, there is no change in the lattice support, as number of records remain unchanged.

Step 10. Handle the updates from the new sequence (insert) in the lattice structure:

- a) Add the sequence directly with the root node.
- b) The corresponding frequency and lattice support values are stored.

- c) Update the frequency of the fuzzy sequence in the ProgresLattice, if it already exists.
- d) Update the lattice support values of all the nodes of the ProgresLattice, as the number of records increase with each new sequence.

Step 11. Handle the delete operation in the lattice structure:

- a) The required nodes of the sequence is deleted if a set of items are to be removed. The frequency of the subsequent nodes are decreased from top to bottom in the ProgresLattice for the node that is deleted.
- b) If a complete record of an item is to be deleted, then the complete sequence for that node is removed and the frequency is updated for the corresponding other sequences. However, the lattice support for all the nodes has to be updated in this case, as there is decrease in the number of records.

Phase III: Mining of fuzzy sequential patterns from the ProgresLattice structure.

Step 12. Examine the frequent prefix subsequence with corresponding postfix subsequence in the constructed Progreslattice as in Phase II.

Step 13. Traverse each level of depth of the Progreslattice and generate all possible sequences that include nodes in that specific level.

Step 14. Consider the minimum support threshold and lattice support to determine the frequent fuzzy sequential patterns among all the mined sequential patterns at each level.

7.5.4. An Illustrative example

In this section, an example is given to illustrate the proposed fuzzy sequential pattern mining algorithm. The dataset in Table 7.4 is a simple example to show how the proposed PLM algorithm is used to generate interesting fuzzy sequential patterns for customer purchasing behaviour according to historical data. The quantitative progressive transaction data in Table 7.5 consists of transactions of seven customers and five items, denoted as 'a' to 'e'. The timestamps value from T2 to T6 and first four customer transactions are considered as part of static database. The timestamps from T8 to T11 are dynamically appended in the database. On the other hand, last three customer transactions are inserted in the database on the fly. We assume the fuzzy membership

function is the same for all the items. It has been taken as the triangular membership function for this example.

Customer id	Purchase sequences						
	T2	T4	T6	T8	T9	T10	T11
01		b	cd		a		
02	c	b		d		e	a
03		a	e		b	c	
04	a	d			da		
05	c	b		d		ac	
06		a			d	c	
07	b		c				d

Table 7.4 Customer transactions from Progressive database

Phase I. The formula $f(x)$ given in Figure 7.5 calculates the membership values for the transaction database of Table 7.4. In this example, the fuzzification is done in the time of arrival instead of frequency of certain quantitative sequences, the frequency of sequences with minimum relevancy, weighted frequency. The purchase sequences given in Table 7.5 is given on the timestamps of monthly basis. These sequences ought to be converted into fuzzy partition sets with their computed membership values with respect to every timestamps. The three defined parameters utilized to compute the membership values are *Low (L)*, *Medium(M)* and *High (H)*. The predicted values of each item are computed by utilizing the function $f(x)$ where a , b and c represent the x coordinates of the three vertices of $f(x)$ in a fuzzy set A (a : lower boundary and c : upper boundary where membership degree is zero, b : the centre where membership degree is 1). The standard triangular function is given in Figure 7.6.

$$f(x) = \begin{cases} 0 & \text{if } x \leq a \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{if } b \leq x \leq c \\ 0 & \text{if } x \geq c \end{cases}$$

Figure 7.5 Formula $f(x)$ to compute triangular membership function

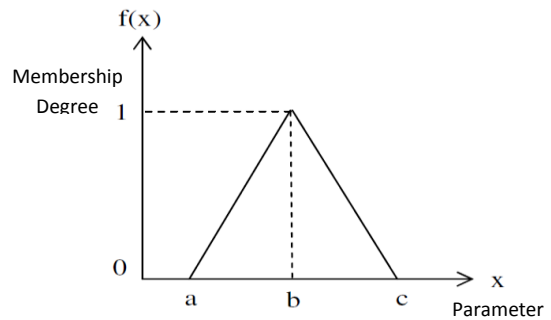


Figure 7.6 Triangular Membership Function

The triangular membership functions with defined parameters and their values for our example is depicted in Figure 7.7. The *Low* parameter has $a=0$, $b=3.5$, $c=5$ values to reflect the timestamps of transactions, while *Medium* has $a=3.5$, $b=5$, $c=8.5$ and *High* has $a=5$, $b=8.5$, $c=12$ values.

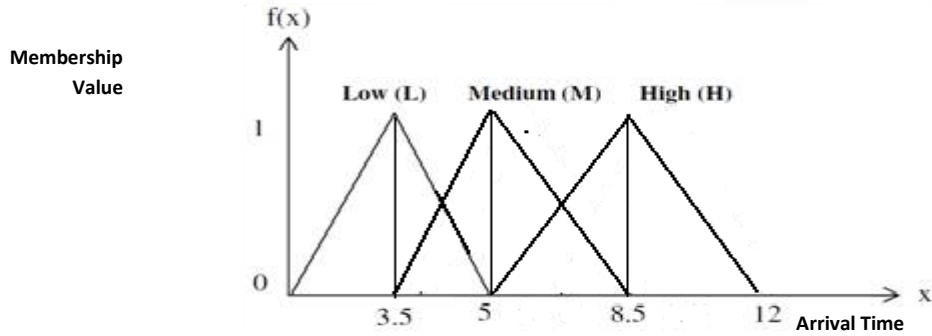


Figure 7.7 Triangular membership functions with defined parameter values

Table 7.5 describes the fuzzy sequences transformed from progressive database with their computed membership values. Here, the membership values of the item b.L with T4, where $x=4$, $a=0$, $b=3.5$, $c=5$ is 0.7, and 0.3 for b.M. Similarly, we have found out other membership values for the three parameters of all customer transactions and generate corresponding fuzzy sequences.

Customer id	Fuzzy sets: (L: low; M: medium; H: high)						
	T2	T4	T6	T8	T9	T10	T11
01		(0.7/b.L+ 0.3/b.M)	(0.7/cd.M + 0.3/cd.H)	0.9/a.H			
02	(0.6/ c.L)	(0.7/b.L+ 0.3/b.M)		(0.1/d. M +0.9/d. H)		0.6/e.H	0.3/a.H
03		(0.7/a.L+ 0.3/a.M)	(0.7/e.M +0.3/e.H)		0.9/b.H	0.6/c.H	
04	(0.6/a.L)	(0.7/d.L+ 0.3/d.M)		0.9/da.H			
05	(0.6/c.L)	(0.7/b.L+ 0.3/b.M)		(0.1/d. M +0.9/d. H)		0.6/ac.H	
06		(0.7/a.L+ 0.3/a.M)			0.9/d.H	0.6/c.H	
07	(0.6/b.L)		(0.7/c.M +0.3/c.H)				0.3/d.H

Table 7.5 Fuzzy sequences transformed from progressive database

Phase II. After the transformation of the fuzzy sequences, we have built the novel ProgresLattice structure by utilizing the standard lattice structure in this phase.

In the Table 7.5, initially there are four customer sequences with the fuzzy membership values, without considering the updates dynamically done in the progressive database. While creating the ProgresLattice, add each customer's sequences to the root node with the frequency. If it was the sequence of n-length, then subsequently we build the lattice structure by creating the n-1 length nodes of it. For example, consider the 2-length sequence (b, cd: M), we can add this to the root node. Then, we create the subsequent nodes of this particular sequence with 2- length and the 1- length sequences. The attained subsequent branch nodes are (c, d: M), (b, d: M), (b, c: M) and (d: M), (c: M), (b: M). Similarly, we build the lattice structure with the other corresponding sequences. The lattice support is also updated along with the frequency in every node, keeping the tendencies of the data from top to bottom. The nodes located in the bottom of the lattice represent the unifying concepts of the different tendencies coming from the top.

Yet, to guarantee that the lattice is a closure system, we need to check any two nodes must have one single supremum, that is, following the ascending paths of two different nodes we will reach one single common node being their top. To fulfill this condition, we need to add some extra nodes. These artificial supremum of two closed sets, will consist on the union of the maximal sequences contained by their immediate predecessors, so that there is no need to scan the input sequences in *DB* for completing the lattice. The addition of (c, d: M) is an artificial supremum to satisfy the closure property, though (b, d: M) and (b, c: M) nodes reflect corresponding fuzzy sequences of the database. The lattice structure is given in Figure 7.8.

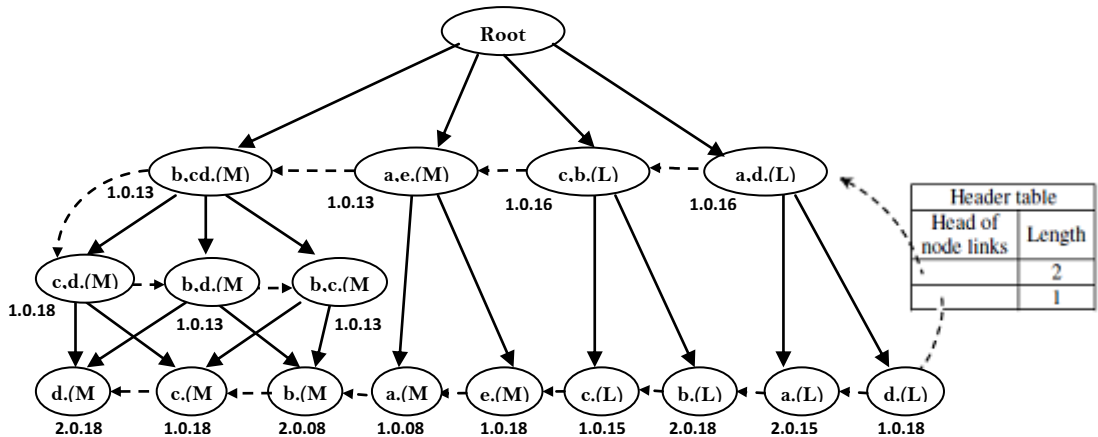


Figure 7.8 Lattice structure for the sequences of T2 to T6 of Table 7.5

7.5.4.1 Updating an existing sequence (append) of the ProgresLattice structure

If the same customer updates the data, then it is the data updates in the same record. This problem complicates the incremental mining since one cannot ignore the infrequent sequences in 'db', but there are an exponential number of infrequent sequences even in a small 'db' and checking them against the set of infrequent sequences in 'DB' is very costly.

The Figure 7.9 is derived from the Table 7.5, in which the fuzzy sequences are converted into the lattice structure with their updated data as well. Here, for timestamp 'T8', there is newly updated data from the same customer 02, that is, fuzzy item 'd'.

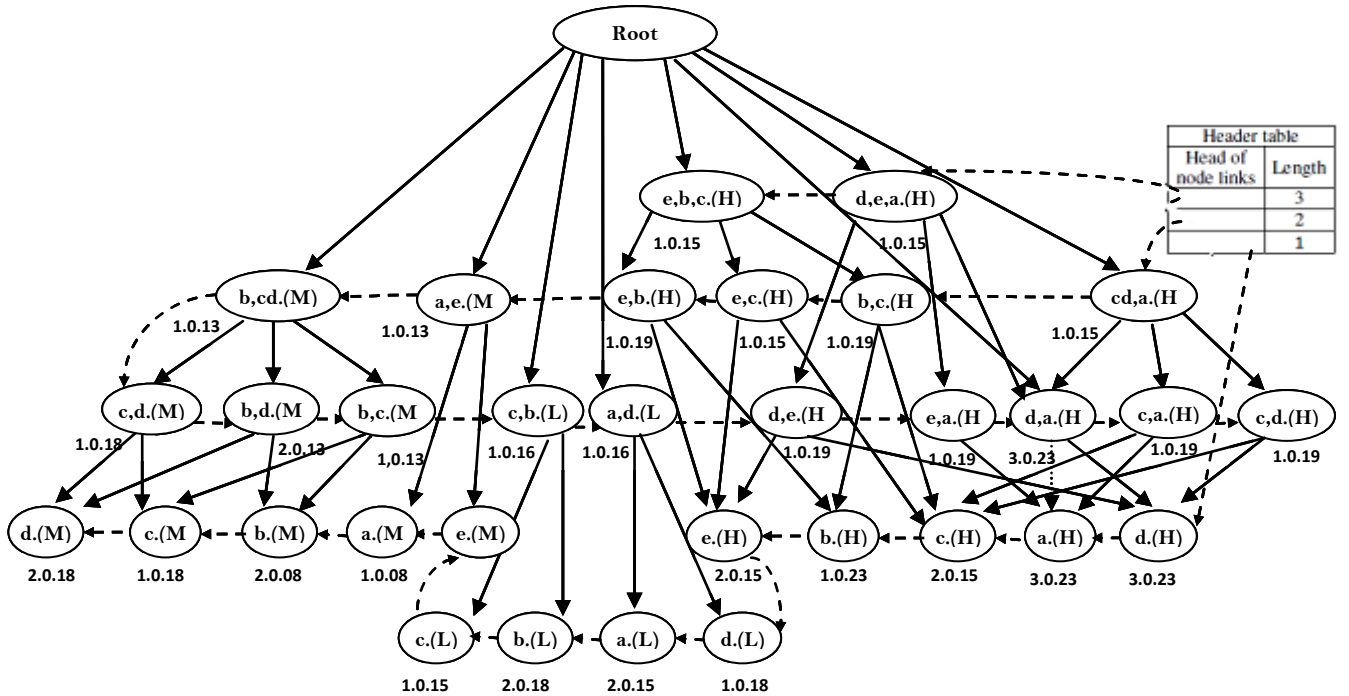


Figure 7.9 Complete lattice structure with updating in the existing customer

As shown in the Figure 7.9, the sequences, (b, d: M), is already present in the lattice so the frequency of this node is increased by one. However, a new node for fuzzy sequence (d: H) is added to construct the ProgresLattice structure. Considering the updating done from T8 to T11, the four customers, with id 01 to 04 give new fuzzy sequences, (d, e, a : H) and (e, b, c : H). The corresponding new nodes are added to the root node in the ProgresLattice of 3-length sequences. Here, while creating the subsequent node of this sequence, the frequencies get incremented with the existing same set of sequence nodes like (b, d: M), (d, a: H), (a: H). Here we need to compare it with all existing nodes.

However the lattice support for new added nodes needs to be updated. Many nodes like $(d, e : H)$, $(c, a : H)$, $(e, a : H)$ are added to satisfy the closure property. Similarly, all the fuzzy sequences get updated in the ProgresLattice tree based on the frequency and the lattice support for all customers.

7.5.4.2. Updating new sequence (insert) in the ProgresLattice structure

It is much easier to handle this case, that is, INSERT, which means the updates happening from the new sequences, added in the database. An important property of INSERT is that a frequent sequence in $U = DB \cup db$ must be frequent in either DB or db (or both). If a sequence is infrequent in both DB and db, it cannot be frequent in U. Thus, only those patterns that are frequent in db but infrequent in DB need to be searched in DB to find their occurrence count.

Suppose, there is an update from the new sequence like customer 05, the sequences are directly added into the root node with the corresponding frequency along with the lattice support. Here, in the Figure 9.7, the sequence $(d, ac : H)$ with the branch nodes are the newly added one. However, the frequency of $(c, b : L)$ and $(b, d : M)$ is incremented as these nodes exist in the lattice structure. The lattice support for all fuzzy sequences is updated as the number of records increased from 4 to 5 with addition of fifth customer. We can do the similar operation for other two new customers 06 and 07, though we have avoided the changes in Figure 7.10 to maintain the simplicity of the figure.

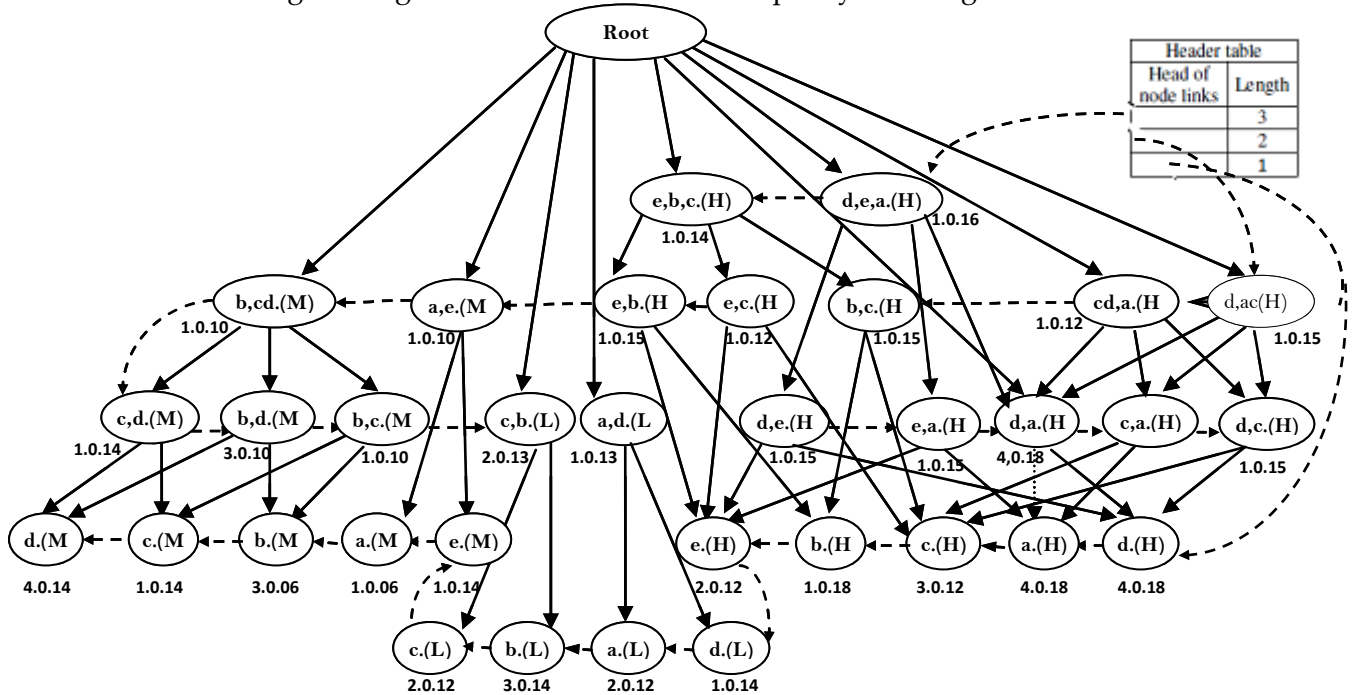


Figure 7.10 Complete lattice structure with updating of new customer sequences

7.5.4.3. Updating due to the deletion operation in the ProgresLattice structure

There are two types of possibilities with the deletion operation, either a complete sequence can be removed or a set of items be removed from the progressive database with time. In case, if a customer sequence is deleted for our example, corresponding sequence from root node till leaf node is removed from the ProgresLattice and the frequency of existing nodes in other sequences is decremented. However, if a set of items are deleted from the database, then we need to carefully select the corresponding node in the ProgresLattice for deletion and further decrease the frequency of subsequent node or remove them completely (if frequency is 1 for them). The other case is little tricky.

For our fuzzy database, if with time, we need to remove the timestamp T2 for all customers, then we need to delete (c : L) and (c, b ; L) nodes from the ProgresLattice structure and decrease the frequency of (a : L) by one. The complete lattice for the progressive fuzzy database after deletion is given in Figure 7.11.

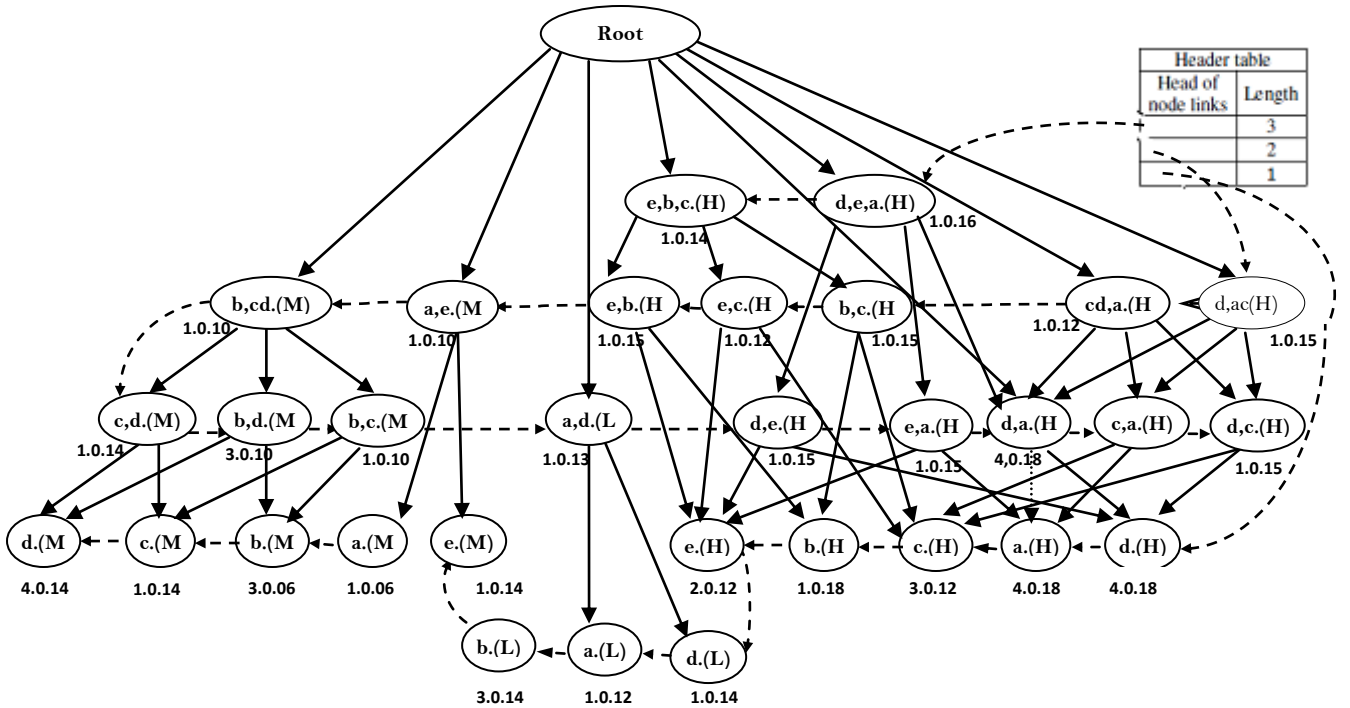


Figure 7.11 Complete ProgresLattice structure with deletion operation in database

Phase III. Its general idea to examine only the frequent prefix subsequences and project only their corresponding postfix subsequences into projected databases because any frequent subsequence can always be found by growing a frequent prefix. The

ProgresLattice tree structure provides an efficient structure for mining, although the combinatorial problem of mining sequential patterns still has to be solved.

To discover all sequential patterns, the ProgresLattice takes a look at each level of depth of the lattice structure and generate all possible sequences that include nodes in that specific level. After having mined the sequential patterns for every level of considering the minimum support threshold and fuzzy lattice support, they are stored in the complete set of patterns. Here, the resultant frequent sequences which meet the minimum threshold support value of 2 and fuzzy lattice support value of 0.1 are listed as below. The sequence of 2- length pattern (b, d : M) and (c, b : L) satisfies both the threshold and the sub-nodes along with other nodes are considered here as the frequent fuzzy sequential patterns given in Table 7.6.

2- length sequences	(c, b : L), (b, d : M), (d, a : H)
1-length sequences	(c : L), (b : L), (a : L), (d : M), (e : H), (c : H), (a : H), (d : H)

Table 7.6 Mined frequent fuzzy sequential patterns from incremental database

However, the above patterns are true for incremental databases. When we consider the old and obsolete data of timestamp T2 are removed from the database, (c, b : L), (c : L) and (a : L), we get more relevant and interesting patterns from the progressive database as given in Table 7.7.

2- length sequences	(b, d : M), (d, a : H)
1-length sequences	(b : L), (d : M), (e : H), (c : H), (a : H), (d : H)

Table 7.7 Mined frequent fuzzy sequential patterns from progressive database

7.5.5. Experimental Setup and Dataset Description

The experiment has been carried out on a 2.9 GHz, dual core PC machine with 1 GB main memory running a 32-bit version of Windows XP for ProgresLattice Miner and Nancy P.'s algorithms. The proposed incremental mining algorithm has been designed so that they can execute in a distributed environment, which means the updating of data records can be done from the multiple sources. So, we run the algorithm in thread environment, in which the updating of data records is done in various threads.

Datasets: The performances of the algorithms have been evaluated using the synthetic datasets as well as real life datasets. **Synthetic dataset:** A set of synthetic data sequence is generated by a data generator similar in spirit to the IBM data generator designed for testing sequential pattern mining algorithms. Each data sequence contains a sequence of item sets. However, different time values are assigned to the items in different item sets but the same time values to those in the same item sets. A dataset of 1000 records with 4 transactions are compared. **Real life datasets:** The UCI machine learning repository (<http://archive.ics.uci.edu/ml/datasets>) is used for the implementation of all algorithms. This data describes the page visits of users who visited msnbc.com on September 28, 1999. Visits are recorded at the level of URL category ("frontpage", "news", "tech", "local", "opinion", "on-air", "misc", "weather", "health", "living", "business", "sports", "summary", "bbs" (bulletin board service), "travel", "msn-news", and "msn-sports") and are recorded in time order. We have utilized this real dataset with 1000 records of 10 transactions.

7.5.6. Performance Analysis

The experiment results of the proposed algorithm for mining of fuzzy sequential patterns from progressive database using lattice structure are described in this section. It is compared with the negative fuzzy sequential pattern mining approach proposed by Lin N.P *et al.* (2007c). The experiment results show the advantageous feature of single scan due to lattice structure in terms of computational time. The performance of the Progreslattice Miner (PLM) is evaluated by means of three standard evaluation metrics, a) generated number of fuzzy sequential patterns b) computational time and c) memory usage.

7.5.6.1. Effect of support values

Here, the performance analysis of our proposed fuzzy sequential pattern mining approach is depicted with the aid of the synthetic dataset and the real dataset by showing the results with the effects of diverse support values. The result analysis is plotted as a graph by computing the generated number of sequences, computational time and the memory usage with different minimum support threshold.

The number of fuzzy sequences generated for various support thresholds for both synthetic and real dataset is shown in Figure 7.12. By analyzing the graphs, we conclude

that the performance study of our proposed approach shows better performance with the existing tested Lin N. *et al.*'s approach. In this, the number of frequent fuzzy sequences grows up exponentially for synthetic database, when the support threshold is low. The proposed algorithm performs fairly well in all the support values with minimum number of fuzzy sequences of frequent sequences for real dataset. The running time for the mining of sequences are shown in Figure 7.13 for two databases. However, when there are a large number of frequent sequences, the run time performance of both the approaches is on higher end. The computational time decrease with increasing support values for synthetic database and remain nearly constant for real database. The PLM algorithm proves well in the memory usage than the existing algorithm for synthetic database but consume a huge amount of memory for real database as given in Figure 7.14.

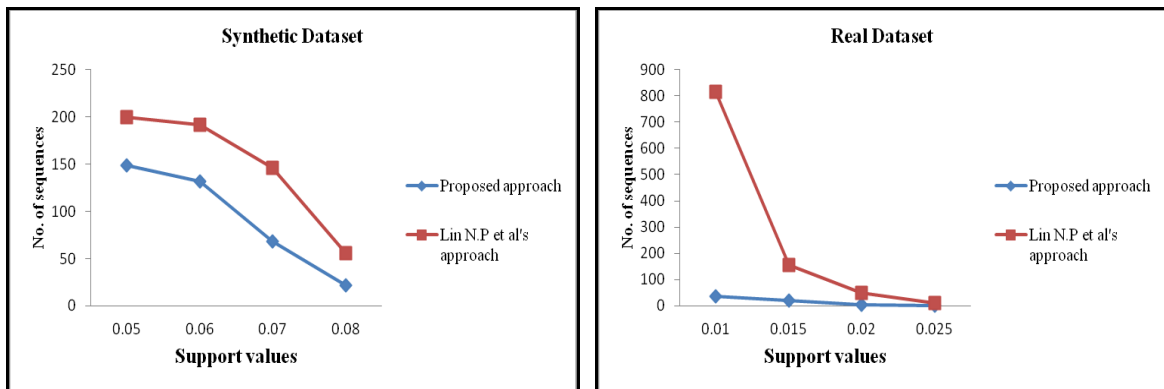


Figure 7.12 Number of sequences generated by ProgresLattice-Miner and Lin N.P. et al's with different support values for real and synthetic databases.

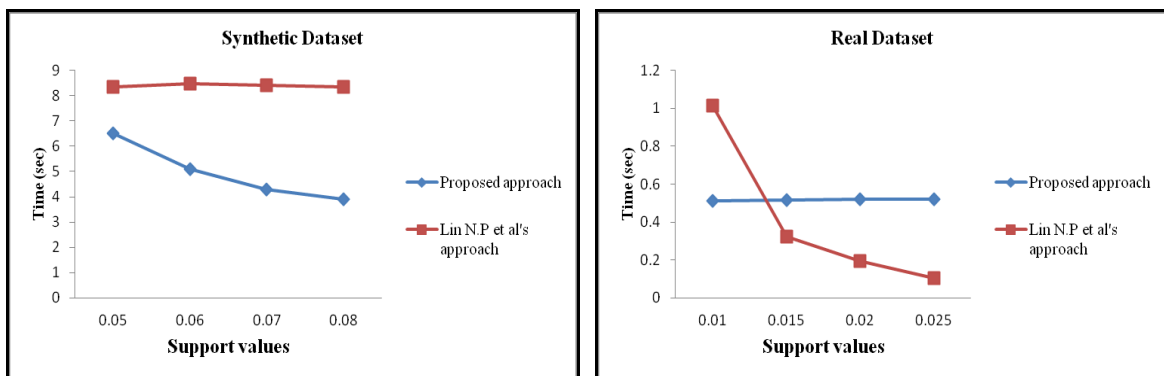


Figure 7.13 Computation time for ProgresLattice-Miner and Lin N.P. et al's with different support values for real and synthetic databases.

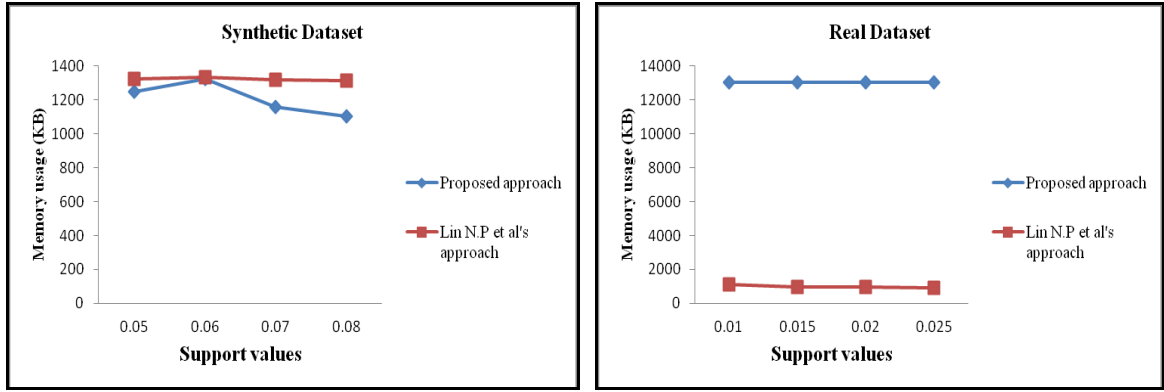


Figure 7.14 Memory usage of ProgresLattice-Miner and Lin N.P. et al's with different support values for real and synthetic databases.

7.5.6.2 Effect of scalability

Here, the performance analysis of our proposed fuzzy sequential pattern mining approach is depicted by computing the generated number of sequences, computational time and the memory usage with different number of records in order to prove the scalability of the proposed one. The number of fuzzy sequences generated for different number of records is shown in Figure 7.15 for synthetic and real datasets. By analyzing the graphs, the number of fuzzy sequences generated is far less than the Lin N. *et al.*'s approach in both cases. Similarly, the runtime and the memory usage for the mining of sequences are shown in Figure 7.16 and 7.17 respectively. The ProgresLattice Miner algorithm performs very well in terms of computational time with varying number of records for both synthetic and real datasets given in Figure 7.16. However, the memory usage of the algorithm is high as compared to Lin P. *et al.*'s approach with the various numbers of records in case of real datasets. This is though comparable in case of synthetic database as given in Figure 7.17.

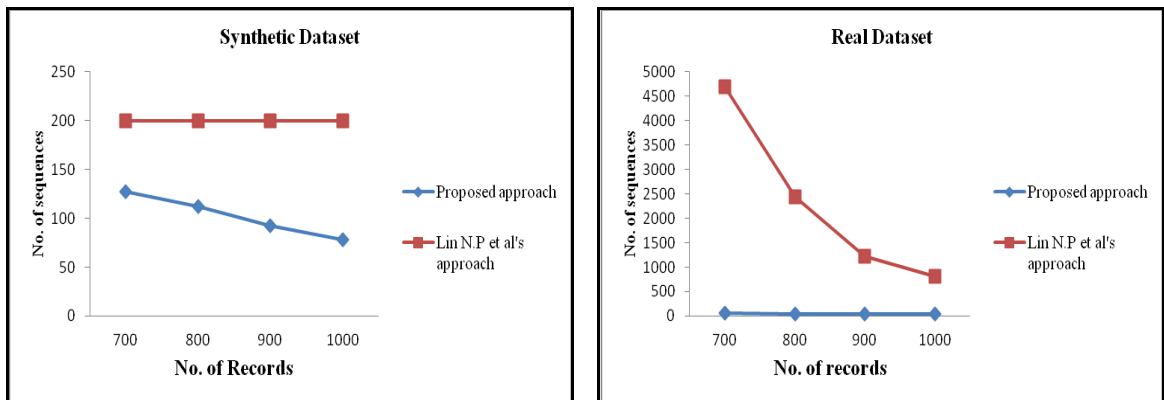


Figure 7.15 Number of sequences generated for ProgresLattice Miner and Lin N. P. et al's with different number of records of real and synthetic databases.

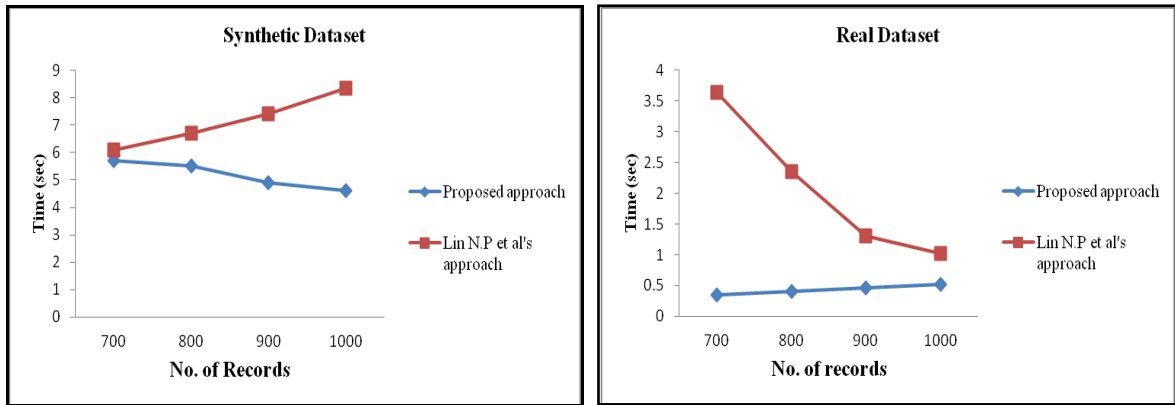


Figure 7.16 Computation time for ProgresLattice Miner and Lin N P et al's with different number of records of real and synthetic databases.

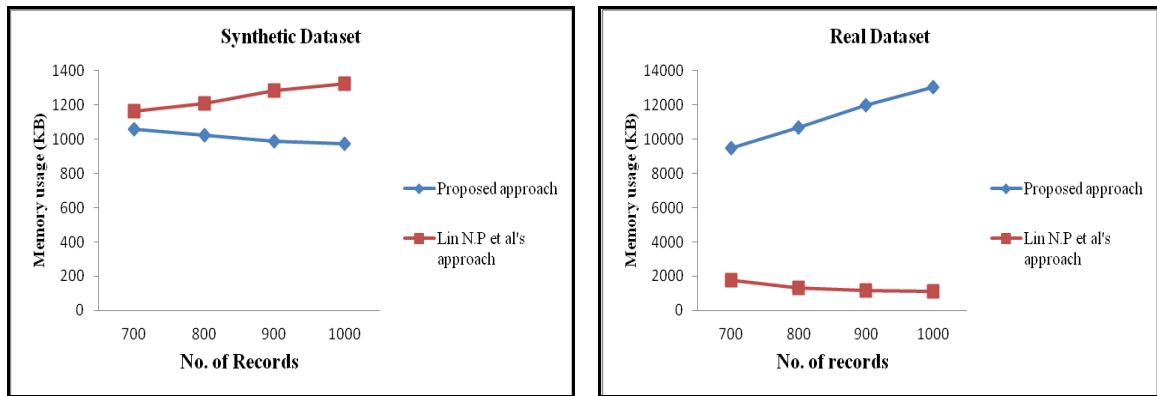


Figure 7.17 Memory usage for ProgresLattice Miner and Lin N P et al's with different number of records of real and synthetic databases.

Experiments have highlighted that this study could be applied to different kinds of data and build many standpoints. The experiments are carried out using different synthetic and real life datasets to prove the efficiency of the proposed algorithm. The obtained results showed that the proposed algorithm is better than Lin *et al.*'s approach in terms of generated number of fuzzy sequences and computational time with varying number of fuzzy support values and records of the datasets. However, the improved memory utilization for the algorithm can be taken as a future work.

7.6. SUMMARY

The non-numeric information provides truly useful results from user perspective. In recent times, there is need for methods that help the user to process, retrieve, exploit and

clarify the available knowledge in a simple way. Here, the fuzzy set theory comes to an aid. There can be a compromise in terms of correctness, completeness and efficiency of extracted information using this technique, but they provide simple and understandable results. Fuzzy sequential mining is used to get knowledge regarding order of the mining results. The sequential mined results need to be represented and maintained with timestamp. This can be done efficiently using the lattice-theoretic framework. The closure and certain properties of lattice model helps in making it a good option to represent the fuzzy data. The lattice can be represented using the undirected graph to get the fuzzy sequential patterns and consequently useful knowledge for business applications and otherwise. The use of lattice representation avoids scanning of the database for mining patterns. The entire database is scanned once to build the lattice structure which is an intermediate data structure in the memory. This reduces the computational time where the complexity of lattice representation is not an issue. This is a temporary data structure generated by the algorithm for its own further use for sequential pattern mining. The only disadvantage it offers here is the more memory usage; however overcoming it is suggested as future work.

In this chapter, an efficient algorithm for incremental mining of fuzzy sequential patterns from progressive database is proposed. This allows the extraction of frequent fuzzy sequences based on minimum support threshold as well as the fuzzy lattice support. This algorithm can be used for incorporating fuzziness in the incremental mining of sequential patterns; however incorporation of constraints to this can be taken as future work.

Chapter 8

Conclusion and Future Research⁸

8.1. CONCLUDING OBSERVATIONS

The proposed algorithms discussed in Chapter 5, 6 and 7, have been designed to study customer purchasing behavior and perform customer value analysis for CRM in dynamic and distributed environment. CRM has been realized as a core for the growth of any enterprise. This requires customer satisfaction and fulfillment of customer requirement, achieved by analyzing consumer behaviors (Rigby *et al.*, 2004). However the traditional data mining techniques have no relevant mechanism to provide guidance for business understanding, model selection and dynamic changes made in the databases.

This chapter discusses how and when the proposed algorithms can be used for CRM to give useful information regarding customers in dynamic environment. A novel integrative model, **Constraint Guided Progressive Sequential Mining Waterfall (CGP-SMW)** is proposed for this reason. The key performance factors that include management of marketing, sales, knowledge, technology, among others are required for the successful implementation of CRM. Moreover, maintenance of progressive database would be time consuming and tedious, if database is modified with every addition, deletion and updating. The key performance factors are recommended to decide for the frequency of maintaining the progressive databases. The sequential pattern mining performed on progressive databases in conjunction with CRM performance indicators can give efficient and effective results. These patterns can further be used to segment the customers and help enterprise to know which customers they should focus on to achieve growth.

Further, the chapter discusses in general, the application of incremental mining algorithms for sequential patterns from progressive databases. The limitations of this research study are also discussed in Section 8.4. It concludes with the recommendation for future research scope in this field.

⁸ The major findings of this Chapter have been communicated to Journal of Database Management.

8.2. APPLICATION OF PROPOSED ALGORITHMS

In the competitive market, it has become very essential to maintain the regular customers along with the effort to acquire new ones. This requires the necessity of developing long-term and pleasant relationships with regular customers by understanding and satisfying their requirements. That is the main objective of any customer relationship management application.

This study attempts to integrate the constraints identified from the customer value analysis along with the sequential pattern mining on progressive database to establish a method of mining customer behavior rules in the dynamic environment. In this study, we have proposed the Constraint Guided Progressive Sequential Mining Waterfall (CGP-SMW) model that can assist managers in developing better marketing strategies and planning for successful implementation of CRM. This model shows how and when the proposed algorithms can be used to suffice the purpose of CRM.

To make optimum use of this model, first the factors that can contribute to the success of CRM are identified.

8.2.1 CRM Key Performance Indicators

For implementing best CRM, different factors contribute. These include management of different aspects, like, technology, sales, marketing, customer service, supply chain, inventory control, knowledge and retail. The technology can help to reach, market, and publicize the products to the customers. It could be used to make convenient, easy better services to them like making use of online payment, maintaining customer call centers for feedbacks and complaints etc. The sales management is often the backbone of CRM as is responsible for enhancing the sales of the products. It could make use of efficient planning, strategy formulation to achieve the same. The other important factor is marketing management which ensures the appropriate suggestions for new products. It takes care of strategy based on customer, company and competitor analysis. Every effort goes in vain if we are not able to maintain the customer once acquired. This requires good customer support and service. It includes answering the customer's query, satisfying the customers for their complaints and assessing their valuable suggestions for future improvements and advancements. The very first level of customer satisfaction is achieved

by delivering the required product on time. This could only be maintained by appropriate inventory control and supply chain management. Appropriate inventory control ensures timely delivery of new and fresh product without the need of much storage requirement. However, the supply chain management requires the well and proper coordination of all departments of organization since the storage of raw materials, production and delivery to customers that is from point-of-origin to point of consumption. Knowledge management is other crucial factor of CRM, as it helps to make forecast of products based on previous data. It also enables to attract the customers for the products and ideas of the enterprise. The employees of the organization can improve their future by making benefit for the same. Lastly, retail management contributes for the CRM by delivering the manufactured product reach to the customers through short and convenient channels.

Due to the importance and relevance the above factors contribute to the success of the CRM, hence they are termed as the Key Performance Indicators (KPIs) for the CRM in this study.

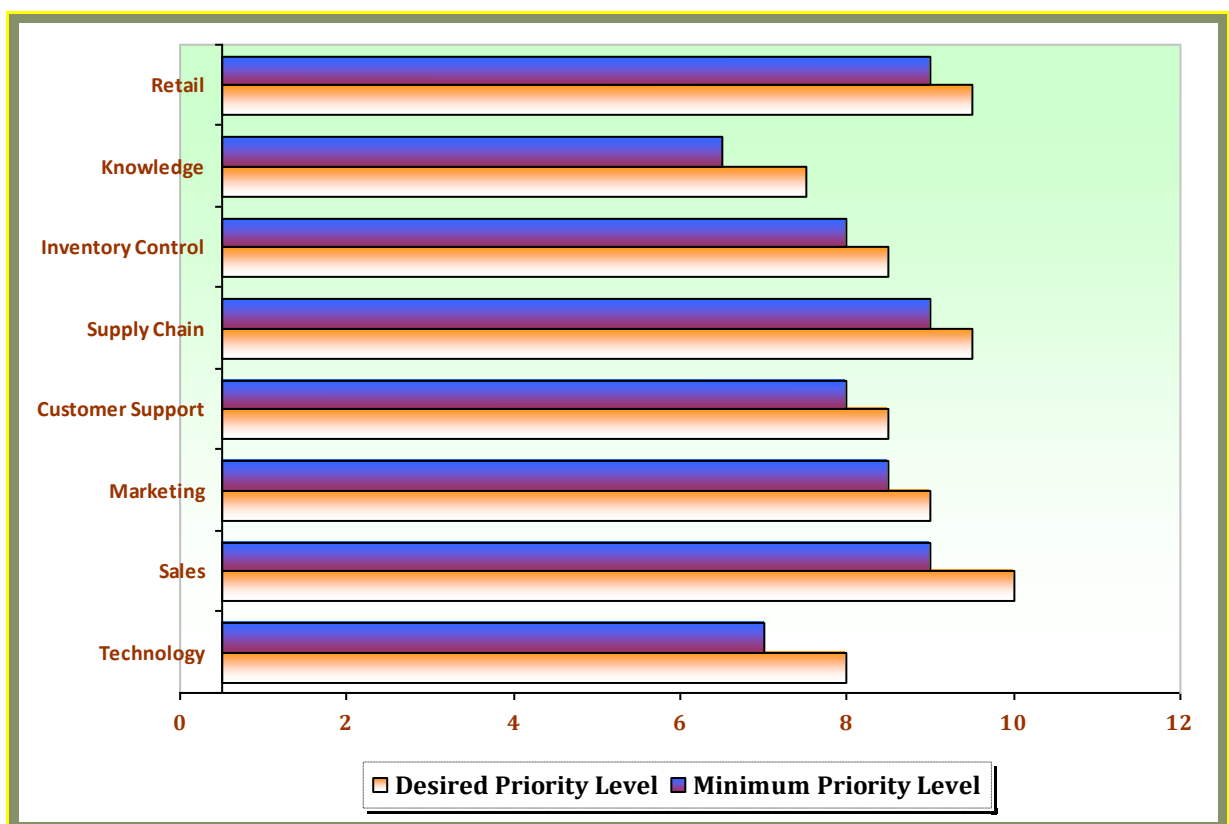


Figure 8.1 CRM Key Performance Indicators along with the Desired and Minimum Priority level for an Enterprise

In Figure 8.1, the identified eight key performance indicators are shown with their desired and minimum priority on a scale of 10 for CRM implementation by any enterprise. The desired priority level specifies that the CRM successful execution requires a specific KPI to achieve this priority in normal scenarios. However, under all circumstances the KPI should at least maintain the minimum priority level for the CRM. This priority level is assigned in the figure, as per the feedback and suggestion from ERP department of an enterprise.

These factors should be used as the basis to decide sequential pattern mining on progressive database and create rules based on proposed algorithms. This would help in better implementation of CRM. CRM through constraint based sequential mining can help to efficiently attain fresh customers, augment value of present customers and maintain valuable customers. But this has to be performed on progressive database with the frequency and access mechanism being governed by the CRM key performance indicators. This is well demonstrated with the proposed model in next section.

8.2.2. Constrained Guided Progressive Sequential Mining Waterfall (CGP-SMW) Model

The discussions made in previous chapters can make us to conclude that the incorporation of constraints in the mining process could give useful and relevant business- centric patterns. There has to be basis for the selection of constraints and we would recommend that for CRM application the customer value analysis being the core should be used. The identified constraints should be further used to generate sequential patterns by using them within the mining process which will improve the value of the results obtained. The discovered patterns could be further used to segment the customers based on scores or priority assigned to each resultant sequential pattern given by mining process. These customers are the ones on which any enterprise should focus and concentrate their limited resources to achieve the goals of the CRM. However, looking at the dynamic and versatile nature of the CRM due to customer purchasing behavior, the discovered patterns lose their significance with time. So it becomes necessary to maintain the progressive databases, instead of static or incremental as discussed in previous

chapters. This lead to an obvious query how to maintain the progressive database, that is are we required to change and modify the database with every addition, modification or deletion. If we are to follow this, then the progressive database management would become tedious, expensive and time consuming process itself. Though we would be able to generate the most relevant and useful patterns giving valuable customers for CRM. To take this account, we would recommend that the frequency of maintaining the progressive database could be decided based on the CRM's key performance indicators. We have identified these factors which are must for the successful implementation of the CRM. Based on the requirement of these KPIs the progressive database could be maintained.

To get the valuable customers for any enterprise for the objectives of CRM that are relevant in scalable and dynamic environment as well, we can follow the steps as given below:

1. Perform the customer value analysis.
2. Identify the constraints relevant for the business based on step 1.
3. Conduct the sequential mining by incorporating the constraints in step 2.
4. Segment the customers based on patterns generated in step 3.
5. Identify KPIs for CRM and use them to decide the frequency of maintaining progressive database.
6. Perform sequential mining on progressive databases to get the valuable customers with the changing environment.

As prominent, these steps need to be performed in a sequence that is each step need to be executed after the successful completion of the previous one. At the same time, the results of earlier step are being used by the next step for its implementation. It is typical behavior of the waterfall model. We can therefore, consider our model as a Constraint Guided Progressive – Sequential Mining Waterfall (CGP-SMW) model, as given in Figure 8.2. *The constraint –based algorithms proposed in Chapter 5 can be used in Step 3 of this model to give useful patterns. However, the algorithms proposed in Chapter 6 and 7 can be used in Step 6 of this model to give constraint based sequential patterns and fuzzy sequential patterns respectively from progressive databases.*

This model is iterative in nature to take care of the changes with time. The parameters of customer value analysis could differ in dynamic environment. For example, we have suggested that to acquire new customers, maintain regular customers and preserve valuable customers are important parameters for customer value analysis. This could be modified, added or appended with other parameters by any enterprise in competitive market. It would require to further analyze and this lead to identification of different constraints altogether.

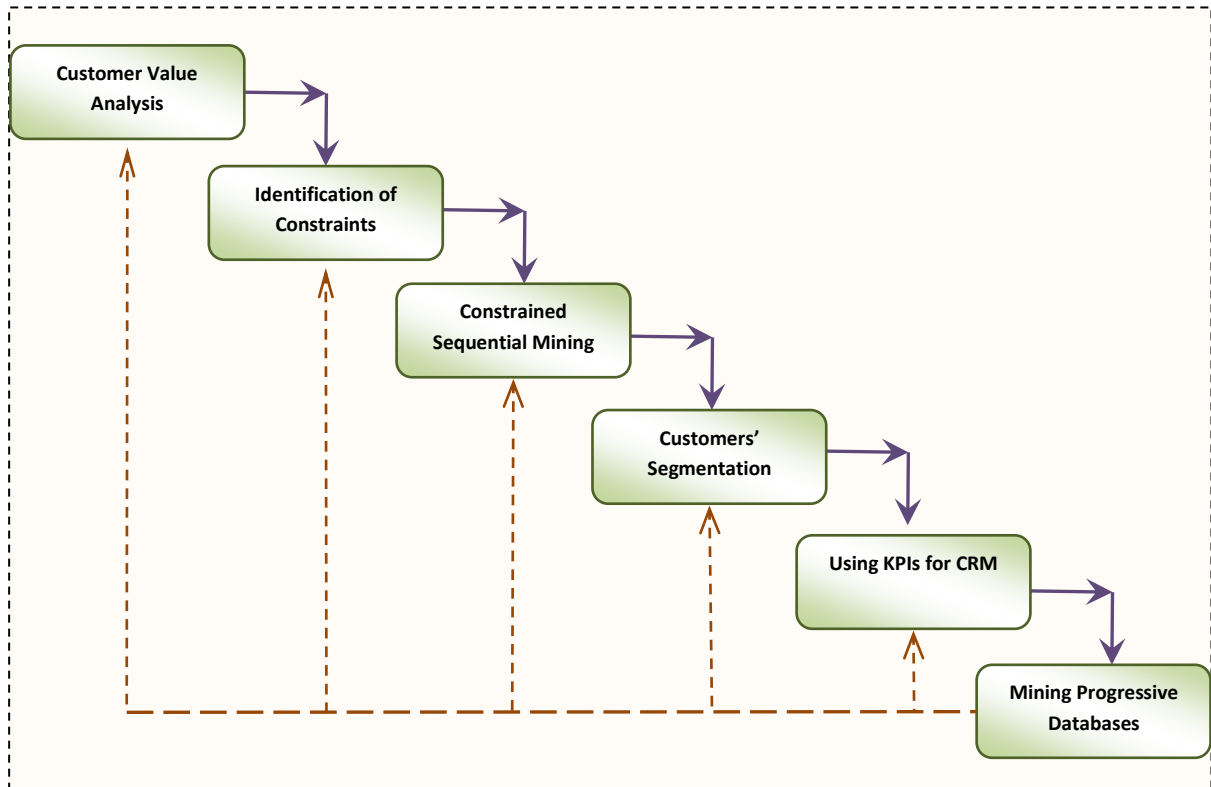


Figure 8.2 Constraint Guided Progressive- Sequential Mining Waterfall (CGP-SMW) Model

As the sequential pattern mining is based on constraints, so it is to be performed again with every alteration in Step 1 of the model, leading to change in Step 2. In fact it is not necessary that we need to always transform from Step 1 to take care of scalability in modern business scenario. For example, if required to change the CRM's KPI to modify the frequency of maintenance of progressive databases, then the iteration is limited to step 5 and 6.

We can conclude that the CGP-SMW model is hence iterative waterfall model as shown in Figure 8.2. The detailed and elaborative version of this model is given in Figure 8.3.

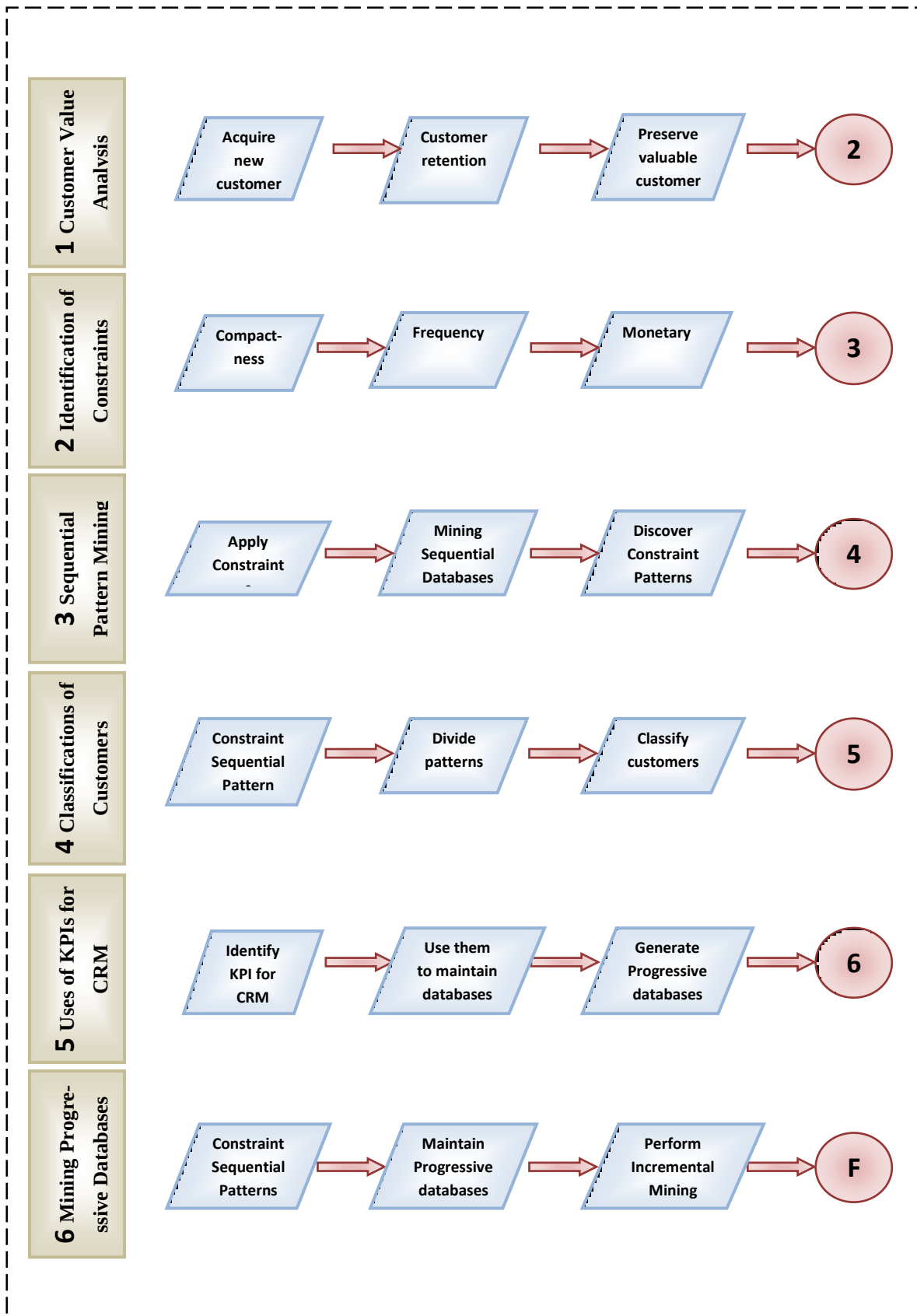


Figure 8.3 Proposed CGP-SMW Model in Detail.

8.3. APPLICATIONS OF INCREMENTAL MINING OF SEQUENTIAL PATTERN ALGORITHMS

As mentioned in Chapter 1, the mining of sequential patterns finds vast applications. With the discussion made so far, it is needless to state that, in practice with dynamic and distributed databases, the Incremental Mining algorithms from progressive databases have more vital role to play. We demonstrate this, with some real-time applications as given below:

- i. ***Text mining:*** There are tools like CP- Miner, Clospan, Mapo, Muvi, PR- Miner that tokenize the source text in certain ways. The tokens are then mapped to the sequence database. This database could be further mined to extract frequent sequential patterns using PSPM algorithms. Useful information, like programming rules, API usage, and copy-pasted code segments can be extracted. For example, CP – Miner is a tool that detects copy-pasted text segments from the given text effectively. It first tokenizes every statement of the source text. Then it maps these tokens into a sequence database. When this database is mined for closed sequential patterns say with minimum support threshold of 3, it provides details of all copy pasted text segments that were repeated twice.
- ii. ***Stock Market analysis:*** Usually as per the local and global market scenarios, as well as economic condition of a state, stock market follows a particular trend in sales and purchase of stocks. These are the sequential patterns which could be mapped to our sequence database, which is dynamic in nature. This database, mined by using PSPM algorithms, could provide useful information like, which type of stocks suffers from major losses in particular situation, or in making decisions that purchase of which stock could book profit, based on past history.
- iii. ***Medical/ Biological based application:*** Nowadays, there is an increase in the amount of biological data sequences, either DNA or protein sequences. These sequences once mapped to the sequence database could be mined based on their structure, function and role in chromosome. For example, one of the common

problems that occur in analysis of biological sequences is to find the similarity in sequences of related proteins or genes. Such similarity (frequent sequential pattern) is due to the residues that are conserved during evolution of mankind. PSPM algorithm applied in this area can give desired results.

- iv. **Retail Industry:** The study of customer purchasing behavior is an example of modeling the behavior of some entity along with time. The next purchase of the customer can be well predicted at any instant of time by using a customer transaction database having all past transaction details of the customer. This knowledge could help in decision making for sales and production of customer goods. This may even help in demand forecasting, inventory management, supply chain management and so on.
- v. **World Wide Web (WWW) application:** There is a huge volume of daily log data collected on the server sites. This includes the sequences of page accesses by thousands of users on the web. This data (server logs) can be studied and analyzed to discover frequent user access patterns on WWW. We can use this knowledge to improvise the system design of the web application, like maintaining better hyperlinked structure between correlated pages. Going further, this could even help in making better marketing decisions by strategically placing the advertisement on the web.

8.4 LIMITATION OF PRESENT RESEARCH

This research work has focused on identification of constraints from the perspective of customer purchasing behavior and customer value analysis for CRM. However, other constraints can be identified for other domains and applications. The incorporation of constraints with the incremental mining algorithm to discover sequential patterns has been done in this work. The constraints are not incorporated with the incremental mining of fuzzy sequential patterns.

The algorithms have been compared with the naïve and well known algorithms using two datasets – synthetic dataset based on IBM dataset generator and real-time dataset taken from UCI data repository. More extensive study can be done using various datasets.

8.5 RECOMMENDED FUTURE WORK

Although some aspects of sequential pattern mining are relatively well understood, many more aspects are in need of research. This study needs to be more practical, domain and business specific instead of driven by academic considerations only. Complex object-oriented, CAD-CAM, textual, and multimedia databases (Sikora *et al.*, 2007; Liu *et al.*, 2013) need to be analyzed, with more domain knowledge. Much larger databases will require more efficient algorithms and much more incremental methods for dynamic environment. Multiple approaches for mining are required to be integrated for difficult problems like control and management of network. It is also desirable that the results are presented in simple and understandable manner to users. These directions can further be categorized as:

- i. **Domain knowledge:** This reduces the search space as well as gives more business specific and interesting patterns. However a specialized algorithm that will outperform any general technique may not be useful outside the domain. A desirable compromise is to develop a framework for augmenting the general method with the specific domain knowledge.
- ii. **Efficient algorithms:** The mining algorithms are required to make use of latest advances in hardware and software like parallel computers, large memories, more restricted rule types, heuristic techniques. This will not only improve the existing methods but also give fast and more efficient new techniques.
- iii. **Interactive systems:** This approach combines the use of human decision and machine for search. The mined knowledge is required to be provided to user in understandable manner by the interactive approach. Use of uncertainty, fuzziness and neural networks in the algorithms make them more understandable and useful from human perspective.
- iv. **Distributed Data Mining:** As data mining requires huge amount of resources in storage space and computation time, so either to make system scalable or when data is inherently distributed into several databases for centralized processing, efficient mechanisms need to be further developed and improvised.

- v. **Data Stream Mining:** Data is not always stored in databases, from where we can retrieve it as and when required. Data may also arrive in stream(s) and if not processed or immediately stored (which is also not feasible sometimes due to speed) then it can be lost forever like network traffic, phone conversations, sensor data etc. More efficient algorithms for mining data from streams are required.
- vi. **Integration** is required to be done at different levels to mine huge databases. The combination of efficient algorithms with concepts of incremental mining, domain-specific knowledge and interactivity can help to resolve issues of dynamic and application-specific real-time databases. The approach could further be made scalable for the distributed systems.

REFERENCES

1. Aggarwal, C. C., Procopiuc, C., and Yu, P. S., 2002. Finding Localized Associations in Market Basket Data. *IEEE Transactions on Knowledge and Data Engineering*, 14(1), pp: 51–62.
2. Aggarwal, Gargi, and Bhatia, M. P. S., 2012. A Novel Faster Approximate Fuzzy Clustering Approach with Highly Accurate Results. *International Conference on Contemporary Computing*, 2012, India, August 6-8, 2012. *Proceedings in Communications in Computer and Information Science*, 309, Springer 2012, ISBN 978-3-642-32128-3, pp: 213-224.
3. Agrawal, R., and Srikant, R., 1995. Mining Sequential Patterns. In *Proceedings of the 11th International Conference on Data Engineering*, Taipei, Taiwan, pp: 3-14.
4. Agrawal, R., Imielinski, T., and Swami, A., 1993. Database Mining: A Performance Perspective. *IEEE Transaction Knowledge and Data Engineering*, 5(6), pp: 914 - 925.
5. Ahmed, D.M., Sundaram, D., and Piramuthu, S., 2010. Knowledge-based Scenario Management – Process and Support. *Decision Support Systems*, 49(4), pp: 507-520.
6. Albert-Lorincz, H., and Boulicaut, J.F., 2003. Mining Frequent Sequential Patterns under Regular Expressions: A Highly Adaptive Strategy for Pushing Constraints. In *Proceedings of the Third SIAM International Conference on Data Mining SDM 2003*, San Francisco, USA.
7. Antunes, Claudia, and Oliveira, Arlindo L., 2003. Generalization of Pattern-growth Methods for Sequential Pattern Mining with Gap Constraints. *Machine Learning and Data Mining in Pattern Recognition, Lecture Notes in Computer Science*, 2734, pp: 239-251.
8. Antunes, Claudia, and Oliveira, Arlindo L., 2004a. Sequential Pattern Mining Algorithms: Trade-Offs Between Speed and Memory. *Second International Workshop on Mining Graphs, Trees and Sequences*, Pisa, Italy.
9. Antunes, Claudia, and Oliveira, Arlindo L., 2004b. Sequential Pattern Mining with Approximated Constraints. In *proceedings of the International Conference on Applied Computing*, pp: 131-138.

10. Aseervatham, S., Osmani, A., and Emmanuel, Viennet, 2006. BitSPADE: A Lattice-based Sequential Pattern Mining Algorithm Using Bitmap Representation. Sixth International Conference on Data Mining ICDM06 (2006), Publisher: IEEE, pp: 792-797.
11. Ayres, J., Gehrke, J., Yiu, T., and Flannick, J., 2002. Sequential Pattern Mining Using A Bitmap Representation. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp: 429-435.
12. Baesens, B., Verstraeten, G., Vanden Poel, D., Egmont-Petersen, M., Van Kenhove, P., and Vanthienen, J., 2004. Bayesian Network Classifiers for Identifying the Slope of the Customer Lifecycle of Long-Life Customers. European Journal of Operational Research, 156, pp: 508-523.
13. Barik, Sabita, Mishra, D., Mishra, Shruti, Satapathy, S. K., Rath, A. K., and Acharya, M., 2010. Pattern Discovery using Fuzzy FP-growth Algorithm from Gene Expression Data. International Journal of Advanced Computer Science and Applications (IJACSA), 1(5).
14. Bhatia, M. P. S., and Kumar, Akshi, 2007. Contextual Proximity Based Term-Weighting for Improved Web Information Retrieval. Knowledge Science, Engineering and Management, Second International Conference, Melbourne, Australia, Lecture Notes in Computer Science 4798 Springer 2007, ISBN 978-3-540-76718-3, pp: 267-278.
15. Bhatia, M. P. S., and Kumar, Akshi, 2009. Contextual paradigm for ad hoc retrieval of user-centric web data. IET Software 3(4), pp: 264-275.
16. Bhatia, M. P. S., and Kumar, Akshi, 2010. Paradigm shifts: from pre-web information systems to recent web-based contextual information retrieval. Webology 7(1).
17. Bigus, J., 1996. Data Mining with Neural Networks: Solving Business Problems from Application Development to Decision Support. McGraw-Hill. ISBN: 0-07-005779-6.
18. Bisaria, J., Shrivastava, N., and Pardasani, K.R., 2009a . A Rough Set Model for Sequential Pattern Mining with Constraints. In Proceedings of the (IJCNS) International Journal of Computer and Network Security, 1(2), November 2009.

19. Bisaria, J., Shrivastava, N., and Pardasani, K.R., 2009b. A Rough Sets Partitioning Model for Mining Sequential Patterns with Time Constraint. *International Journal of Computer Science and Information Security*, 2(1), pp: 1-9.
20. Bisaria, J., Srivastava, N., and Pardasani K. R., 2010. A Rough Sets Partitioning Model for Incremental Mining of Sequential Patterns in Large Databases. *Applied Mathematical Sciences*, 4 (38), pp: 1869 – 1881.
21. Bullo, Hadrien, Gupta, S.K., and Mohania, Mukesh K, 2003. A Data-Mining Approach for Optimizing Performance of an Incremental Crawler. *Web Intelligence*, pp: 610-615.
22. Chang, J. Hyuk., 2011. Mining Weighted Sequential Patterns in a Sequence Database with a Time-Interval Weight. *Knowledge Based System*, 24, pp: 1-9.
23. Chang, Lei, Yang, D., Wang, T., and Tang, S., 2007. IMCS: Incremental Mining of Closed Sequential Patterns. *Advances in Data and Web Management, Lecture Notes in Computer Science*, 4505, pp: 50- 61.
24. Changchien, S., Lee, C. F., and Hsu, Y. J., 2004. On-Line Personalized Sales Promotion in Electronic Commerce. *Expert Systems with Applications*, 27, pp: 35–52.
25. Chelluri, Kiran, and Kumar, Vijay, 2001. Data Classification and Management in Very Large Data Warehouses (Invited paper). 3rd International Workshop on Advanced Issues of E-commerce and Web-based Information Systems, San Jose, California, June 21-22, 2001, pp: 52--57.
26. Chen, Enhong, Cao, Huanhuan, Li, Qing, and Qian, Tieyun, 2008. Efficient Strategies for Tough Aggregate Constraint-Based Sequential Pattern Mining. *Information Sciences*, 178(6), pp: 1498-1518.
27. Chen, G., Wu, X., and Zhu, X., 2005. Sequential Pattern Mining in Multiple Streams. In *Proceedings of the 5th IEEE International Conference on Data Mining*, Houston, USA, pp: 27- 30.
28. Chen, Jinlin, and Cook, T., 2007. Mining Contiguous Sequential Patterns from Web Logs. In *Proceedings of the 16th international conference on World Wide Web*, Banff, Alberta, Canada, pp: 1177 - 1178.
29. Chen, R. S., Tzeng, G. H., Chen, C. C., and Hu, Y. C., 2001. Discovery of Fuzzy Sequential Patterns for Fuzzy Partitions in Quantitative Attributes. In *Proceedings*

- of ACS/IEEE International Conference on Computational System Application, pp: 144–150.
30. Chen, Ruey-Shun, and Hu, Yi-Chung, 2002. A Novel Method for Discovering Fuzzy Sequential Patterns Using the Simple Fuzzy Partition Method. *Journal of the American Society for Information Science and Technology*, 54(7), pp: 660–670.
 31. Chen, Y. L., Hao, K. M., Yi, W. S., and Tang, Kwei, 2009. Discovering Recency, Frequency, and Monetary (RFM) Sequential Patterns from Customers' Purchasing Data. *Electronic Commerce Research and Applications*, 8 (5), pp: 241–251.
 32. Chen, Y.L., and Hu, Y.H., 2006. Constraint-based Sequential Pattern Mining: The Consideration of Recency and Compactness. *Decision Support Systems*, 42(2), pp: 1023-1215.
 33. Chen, Yue, Guo, J., Wang, Yaqin, Xiong, Y., and Zhu, Yangyong, 2007. Incremental Mining of Sequential Patterns Using Prefix Tree. *Advances in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science*, 4426, pp: 433-440.
 34. Cheng, C. H., and Chen, Y. S., 2009. Classifying the Segmentation of Customer Value via RFM Model and Rough Set Theory. *Expert Systems with Applications*, 36(3), pp: 4176–4184.
 35. Cheng, Hong, Yan, X., Han, Jiawei, 2004. Incspan: Incremental Mining of Sequential Patterns in Large Database. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pp: 527 - 532.
 36. Cheng, Yi, 2011. The incremental method for fast computing the rough fuzzy approximations. *Data and Knowledge Engineering, Elsevier*, 70, pp: 84-100.
 37. Cheung, D., Han, J., Ng, V., and Wong, C., 1996. Maintenance of Discovered Association Rules in Large Databases: An Incremental Update Technique. In *Proceedings of the 12th International Conference on Data Engineering (ICDE'96)*.
 38. Chiang, D. A., Wang, Y. F., Lee, S. L., and Lin, C. J., 2003. Goal-Oriented Sequential Pattern for Network Banking Churn Analysis. *Expert Systems with Applications*, 25, pp: 293–302.
 39. Chiang, W. Y., 2010. To Mine Association Rules of Customer Values via a Data Mining Procedure with Improved Model: An Empirical Case Study. *Expert Systems with Applications*.

40. Collins, K., 2001. Analytical CRM: Driving Profitable Customer Relationships, Strategic Planning, SPA-12-7120.
41. Cui, W., and Zhong H., 2009. Discovering interesting sequential pattern in large sequence database. In IEEE Second Asia-Pacific Conference on Computational Intelligence and Industrial Applications, pp: 270 – 273.
42. Dennis, C., Marsland, D., and Cockett, T., 2001. Data mining for shopping centres-customer knowledge-management framework. Journal of Knowledge Management, 5, pp: 368–374.
43. Dubois, D., Hüllermeier, E., and Prade, H., 2003. A note on quality measures for fuzzy association rules. In Proceedings of 10th IFSA World Congress on Fuzzy Sets System, pp: 346–353.
44. Dubois, D., Hullermeier, E., and Prade, H., 2006. A systematic approach to the assessment of fuzzy association rules. Data Mining and Knowledge Discovery, 13, pp: 167-192.
45. Dunham, M., Al-Mogren, A., Seydim, A., and Kumar, Vijay, 2001. Data in Your Space (Invited Paper). 3rd International Workshop on Advanced Issues of E-commerce and Web-based Information Systems, San Jose, California, June 21-22, 2001, pp: 66–73.
46. Eirinaki, M., and Vazirgiannis, M., 2003. Web Mining for Web Personalization. ACM Transactions on Internet Technology (TOIT), 3(1), pp: 1-27.
47. Fayyad, U., Shapiro, G., and Smyth, P., 1996. From Data Mining to Knowledge Discovery: An Overview. Advances in Knowledge Discovery and Data Mining, AAAI Press/The MIT Press.
48. Feng, Yuzhang, Li, Yuan-Fang, Tan, Colin K-Yan, Wadhwa, Bimlesh, and Wang, Hai H., 2007. Belief-augmented OWL (BOWL) Engineering the Semantic Web with Beliefs. 12th International Conference on Engineering of Complex Computer Systems (ICECCS 2007), 10-14 July 2007, Auckland, New Zealand, Proceedings in IEEE Computer Society 2007, ISBN 978-0-7695-2895-3, pp: 165-174.
49. Fiot, C., Laurent, A., and Teisseire, M., 2006a. Web access log mining with soft sequential patterns. In Proceedings of 7th International FLINS Conference on Applied Artificial Intelligence, pp: 519–524.
50. Fiot, C., Laurent, A., and Teisseire, M., 2007. Extended Time Constraints for Sequence Mining. In Proceedings of the 14th International Symposium on

- Temporal Representation and Reasoning (TIME'07), June 28-June 30, Alicante, Spain, pp: 105-116.
51. Fiot, Celine, Laurent, A., Teesside, M., and Laurent, B., 2006b. Why Fuzzy Sequential Patterns can Help Data Summarization: an Application to the INPI Trademark Database. In Proceedings of the IEEE International Conference on Fuzzy Systems, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada.
 52. Frawley, W., Shapiro, G., and Matheus, C., 1992. Knowledge Discovery in Databases: An Overview. *AI Magazine*, 13(3), pp: 213-228.
 53. Garofalakis, M.N, Rastogi, R., and Shim, K., 1999. SPIRIT: Sequential pattern mining with regular expression constraints. In Proceeding of 1999 International Conference on Very Large Data Bases (VLDB'99), pp: 223–234.
 54. Garofalakis, M.N., Rastogi, R., and Shim, K., 2002. Mining sequential patterns with regular expression constraints. *IEEE Transactions on Knowledge and Data Engineering*, 14(3), pp: 530 – 552.
 55. Guha, S., Rastogi, R., and Shim, K., 1999. Rock: A Robust Clustering Algorithm for Categorical Attributes. In Proceedings of International Conference on Data Engineering (ICDE '99), pp: 512-521.
 56. Gupta, S.K, Bhatnagar, Vasudha, and Wasan, S.K, 2005. Architecture for Knowledge Discovery and Knowledge Management. *Knowledge Information Systems*, 7(3), pp: 310-336.
 57. Gupta, S.K., Bhatnagar Vasudha, and Wasan, SK, 2001. On Mining of Data. *IETE Journal of Research*, Special issue on Data and Knowledge Engineering, 47 (1), pp: 5-18.
 58. Gupta, S.K., Bhatnagar,Vasudha, and Wasan, SK, 2000. User-Centric Mining of Association Rules. Workshop on Data Mining, Deductive Logic, Meta-Learning and ILP, PKDD 2000, Lyons, France. Sept 2000.
 59. Gupta, S.K., Rao, K.Sambasiva, and Bhatnagar, Vasudha, 1999. K-Means Clustering Algorithm for Categorical Attributes. DaWaK99 (Data Warehouse and Knowledge Discovery), Florence, Italy, September 1999. Proceedings published in Lecture Notes in Computer Science, Springer Verlag.
 60. Han, J., and Fu, Y., 1996. Attribute-Oriented Induction in Data Mining. *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press, pp: 399-421.

61. Han, J., and Kamber, M., 2001. Data Mining: Concepts and Techniques. Morgan Kaufman publishers, ISBN: 1-55860489-8.
62. Han, J., Cheng, H., Xin, D., and Yan, X., 2007. Frequent pattern mining; current status and future directions. Data Mining and Knowledge Discovery, Springer, 15, pp: 55-86.
63. Han, J., Pei, J., and Yan, X., 2005. Sequential pattern mining by pattern-growth: principles and extensions. Springer-Verlag Berlin Heidelberg, StudFuzz, 180, pp: 183-220.
64. He, Z., Xu, X., Huang, J. Z., and Deng, S. 2004. Mining class outliers: concepts, algorithms and applications in CRM. Expert Systems with Applications, 27, pp: 681-697.
65. Ho, C. C., Li, H. F., Kuo, F. F., and Lee, S. Y., 2006. Incremental mining of sequential patterns over a stream sliding window. In proceedings of Sixth IEEE International Conference on Data Mining Workshop, Hong Kong, China, pp: 677 – 681.
66. Hong, T. P., Chen, H. Y., Lin, C. W., and Li, S. T., 2008. Incrementally fast updated sequential patterns trees. In Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, Kunming, pp: 3991-3996.
67. Hong, T. P., Lin, K. Y., and Wang, S. L., 2001a. Mining fuzzy sequential patterns from multiple-items transactions. In Proceedings of Joint ninth IFSA World Congress and twentieth NAFIPS International Conference, pp: 1317-1321.
68. Hong, T. P., Lin, K. Y., and Wang, S. L., 2003. Fuzzy data mining for interesting generalized association rules. Fuzzy Sets System, 138, pp: 255-269.
69. Hong, T. P., Wang, C. Y., and Tao, Y. H., 2001b. A new incremental data mining algorithm using pre-large item sets. Intelligent Data Analysis, pp: 111- 129.
70. Hong, T. P., Wang, C. Y., and Tseng, S. S., 2001c. An incremental mining algorithm for maintaining sequential patterns using pre large sequences. Expert Systems with Applications, pp: 7051-7058.
71. Hong, Tzung-Pei, Chen, Chun-Hao, and Tseng, Vincent S., 2010a. A Summary of Genetic-Fuzzy Data Mining Techniques. In proceeding of IEEE International Conference on Granular Computing, pp:28-29.

72. Hong, Tzung-Pei, Lin, Chun-Wei, and Lin, Tsung-Ching, 2010b. Mining Complete Fuzzy Frequent Itemsets by Tree. *Systems Man and Cybernetics (SMC)*, 2010 IEEE International Conference, 10-13 Oct. 2010, Istanbul, pp: 563-567.
73. Hosseini, S. M. S., Maleki, A., and Gholamian, M. R., 2010. Cluster analysis using data mining approach to develop CRM methodology to assess the customer loyalty. *Expert Systems with Applications*, 37, pp: 5259–5264.
74. Hou, Sizuo, and Zhang, X., 2008. Alarms Association Rules Based on Sequential Pattern Mining Algorithm. In *proceedings of the Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, Shandong, 2, pp: 556-560.
75. Hu, Y. C., Chen, R. S., Tzeng, G. H., and Shieh, J. H., 2003. A fuzzy data mining algorithm for finding sequential patterns. *International Journal of Uncertainty Fuzziness Knowledge-Based System*, 11(2), pp: 173–193.
76. Hu, Y. C., Tzeng, G. H., and Chen, C. M., 2004. Deriving two-stage learning sequences from knowledge in fuzzy sequential pattern mining. *Information Science*, 159, pp: 69–86.
77. Hu, Ya-Han., 2007. The Research of Customer Purchase Behavior Using Constraint-based Sequential Pattern Mining Approach. Thesis Report.
78. Hua, Jiang, and Zhenxing, Cui, 2009. Optimization of Data Mining in CRM Based on Rough Set Theory. *International Forum on Information Technology and Applications*, DOI 10.1109/IFITA.2009.111 and in *proceedings of IEEE*, pp: 252-257.
79. Huaiguo, Fu, and Engelbert, M. N., 2008. A Lattice Algorithm for Data Mining.
80. Huang, Jen-Wei, Tseng, C.Y., Ou, Jian-Chih, and Chen, M. S., 2008. A General Model for Sequential Pattern Mining with a Progressive Database. *IEEE Transactions on Knowledge and Data Engineering*, 20(9), pp: 1153 - 1167.
81. Huang, S. C., Chang, E. C., and Wu, H. H., 2009. A case study of applying data mining techniques in an outfitter's customer value analysis. *Expert Systems with Applications*, 36, pp: 5909–5915.
82. Huang, Tony, and Cheng-Kui, 2010. Knowledge gathering of fuzzy multi-time-interval sequential patterns. *Information Sciences*, 180, pp: 3316-3334.
83. Hwang, H., Jung, T., and Suh, E., 2004. An LTV model and customer segmentation based on customer value: A case study on the wireless telecommunication industry. *Expert Systems with Applications*, 26, pp: 181–188.

84. Jiang, T., and Tuzhilin, A. 2006. Segmenting customers from population to individuals: Does 1-to-1 keep your customers forever. *IEEE Transactions on Knowledge and Data Engineering*, pp: 1297–1311.
85. Julisch, K., 2002. *Data Mining for Intrusion Detection -A Critical Review. Application of Data Mining In Computer Security*, Kluwer Academic Publisher, Boston.
86. Kapoor, G., and Piramuthu, S., 2009. Sequential Bias in Online Product Reviews. *Journal of Organizational Computing and Electronic Commerce*, 19(2), pp. 85-95.
87. Kaur, Prabhjot and Gosain, Anjana, 2009. Improving the performance of fuzzy clustering algorithms through outlier identification. *FUZZ-IEEE, IEEE International Conference on Fuzzy Systems*, Jeju Island, Korea, 20-24 August 2009, *Proceedings in IEEE 2009*, ISBN 978-1-4244-3596-8, pp: 373-378.
88. Kaur, Prabhjot, Soni, A. K., and Gosain, Anjana, 2013. A robust kernelized intuitionistic fuzzy C-means clustering algorithm in segmentation of noisy medical images. *Pattern Recognition Letters (PRL)* 34(2), pp:163-175.
89. Khan, Javed I., and Tao, Qingping, 2005. Exploiting Webspace Organization for Accelerating web Prefetching. *Web Intelligence and Agent Systems (WIAS)* 3(2), pp:117-129.
90. Khan, Javed I., and Hardas, Manas, 2007. A Technique for Representing Course Knowledge Using Ontologies and Assessing Test Problems. *Proceedings of the 5th Atlantic Web Intelligence Conference - AWIC 2007*, Fontainebleau, France, June 25 - 27, 2007. *Advances in Soft Computing* 43 Springer 2007, ISBN 978-3-540-72574-9.
91. Khan, Javed I., and Tao, Qingping, 2003. Webspace Surfing Patterns and Their Impact on Web Prefetching, *Proceedings of the IEEE 2003 International Conference on Cyberworlds, CYBERWORLDS2003*, Singapore, December 2003, pp: 478-485.
92. Khan, Javed. I., 1997. Intermediate Annotationless Content-Based Query and Retrieval in Massive Image Archives with Holographic Information Representation. *Journal of Visual Communication and Image Representation*, Special Issue on Indexing, Storage, Retrieval and Browsing of Image and Video, 7(4), pp: 378-393.

93. Kim, S. Y., Jung, T. S., Suh, E. H., and Hwang, H. S., 2006. Customer segmentation and strategy development based on customer lifetime value: A case study. *Expert Systems with Applications*, 31, pp: 101–107.
94. Kim, Y. S., 2006. Toward a successful CRM: Variable selection, sampling, and ensemble. *Decision Support Systems*, 41, pp: 542–553.
95. Kim, Y. S., and Street, W. N., 2004. An intelligent system for customer targeting: A data mining approach. *Decision Support Systems*, 37, pp: 215–228.
96. Kracklauer, A. H., Mills, D. Q., and Seifert, D., 2004. Customer management as the origin of collaborative customer relationship management. *Collaborative Customer Relationship Management*, pp: 3–6.
97. Kumar, Vijay, and Mullins, Judy, 1995. An Integrated Data Structure with Multiple Access Paths for Database Systems and its Performance. *Data and Knowledge Engineering*, 16, pp: 51–72.
98. Kuo, Mi-Hao, Chen, Y.L., 2009. Discovering RFM sequential patterns from customers' purchasing data. *Journal of Electronic Commerce Research and Applications*, 8(5).
99. Kuo, R.J., Chao, C.M., and Liu, C.Y. 2009. Integration of K-means algorithm and AprioriSome algorithm for fuzzy sequential pattern mining, *Applied Soft Computing*, 9, pp: 85–93.
100. Kuok, C. M., Fu, W.C., and Wong, M. H., 1998. Mining fuzzy association rules in databases. *SIGMOD Record*, 27(1), pp: 41–46.
101. Langley, P., and Simon, H. A., 1995. Applications of machine learning and rule induction. *Communications of the ACM*, 38, pp: 54–64.
102. Lariviere, B., and Vanden Poel, D., 2005. Investigating the post-complaint period by means of survival analysis. *Expert Systems with Applications*, 29, pp: 667–677.
103. Lau, H., Wong, C., Hui, I., and Pun, K. 2003. Design and implementation of an integrated knowledge system. *Knowledge-Based Systems*, 16, pp: 69–76.
104. Lee, Jong Bum, Piao, M., Shin, Jin-ho, Kim, H. S., Ryu, Keun Ho, 2010. ITFP: Incremental TFP for mining frequent patterns from large data sets. In *proceedings of the 2nd International Conference on Computer Engineering and Technology (ICCET)*, Chengdu, pp: V2- 181 - V2-185.

105. Leleu, M., Rigotti, C., Boulicaut, J.F., and Euvarard, G., 2003. Constraint based mining of sequential patterns over datasets with consecutive repetitions. In *Proceedings of the Seventh European conference on principles and practice of knowledge discovery in databases (PKDD 03)*, pp: 303–314.
106. Li, J., Yang B. and Song, W., 2009. A new algorithm for mining weighted closed sequential pattern. In *proceedings of IEEE Second International Symposium on Knowledge Acquisition and Modeling*, pp: 338 – 341.
107. Liao, S. H., and Chen, Y. J., 2004. Mining customer knowledge for electronic catalog marketing. *Expert Systems with Applications*, 27, pp: 521–532.
108. Liao, W. C., Yang, D. L., Wu, J., and Hung, M. C., 2009. Fast and effective generation of candidate-sequences for sequential pattern mining. In *proceedings of IEEE Fifth International Joint Conference on INC, IMS and IDC*, pp: 2006 – 2009.
109. Liao, Zhung-Xun and Peng, Wen-Chih, 2007. Exploring Lattice Structures in Mining Multi-Domain Sequential Patterns. In *proceedings of the InfoScale '07: 2nd international conference on Scalable information systems*, pp: 1-4.
110. Lin, C. W., Hong, T. P., and Lu, W. H., 2009a. An efficient FUSP-tree update algorithm for deleted data in customer sequences. In *proceedings of Fourth International Conference on Innovative Computing, Information and Control*, pp: 1491 – 1449.
111. Lin, C. W., Hong, T. P., Lu, W. H., and Lin, W. Y., 2008b. Incremental FUSP-tree maintenance algorithm. In *proceedings of Eighth International Conference on Intelligent Systems Design and Applications*, pp: 445 – 449.
112. Lin, M. Y., and Lee, S. Y., 2004a. Incremental update on sequential patterns in large databases by implicit merging and efficient counting. *Information Systems*, 29(5), pp: 385 – 404.
113. Lin, M. Y., and Lee, S. Y., 2004b. Interactive sequence discovery by incremental mining. *An International Journal of Information Sciences-Informatics and Computer Science*, 165(3-4), pp: 187 - 205.
114. Lin, M. Y., Hsueh, S. C., and Chan, C. C., 2009b. Incremental discovery of sequential patterns using a backward mining approach. In *Proceedings of IEEE International Conference on Computational Science and Engineering, Vancouver, BC*, pp: 64-70.

115. Lin, M., and Lee, S., 1998. Incremental update on sequential patterns in large databases. In proceedings of the Tools for Artificial Conference (TAI'98), pp: 24 – 31.
116. Lin, Ming-Yen, and Lee, Suh-Yin, 2005. Efficient mining of sequential patterns with time constraints by delimited pattern growth. *Knowledge and Information Systems*, 7 (4), pp: 499 - 514.
117. Lin, Ming-Yen, Hsueh, Sue-Chen, and Chang, C.W., 2006. Mining closed sequential patterns with time constraints. In *Proceedings of Workshop on software engineering, databases, and knowledge discovery and International Computer Symposium*.
118. Lin, Ming-Yen, Hsueh, Sue-Chen, and Chang, C.-Wen, 2008a. Fast discovery of sequential patterns in large databases using effective time-indexing. *Information Sciences*, 178, pp: 4228-4245.
119. Lin, Nancy P., Hao, Wei-Hua, and Chen, Hung-Jen, 2007b. Fast Accumulation Lattice Algorithm for Mining Sequential Patterns. In *Proceedings of the 6th Conference on WSEAS/ International Conference on Applied Computer Science*, Wisconsin, USA, 6.
120. Lin, Nancy P., Hao, Wei-Hua, Chen, Hung-Jen, Chang, Chung-I, and Chueh, Hao-En, 2007c. An Algorithm for Mining Strong Negative Fuzzy Sequential Patterns. *International Journal of Computers*, 1(3).
121. Lin, Nancy P., Hao, Wei-Hua, Chen, Hung-Jen, Hao, E., and Chang, C. C., 2007a. Discover sequential patterns in incremental database. *International Journal of Computers*, pp: 196- 201.
122. Ling, Xu, Li, Song, and Li, Jie, 2010. CRM Customer Value Based On Data Mining. *Third International Conference on Knowledge Discovery and Data Mining*, DOI 10.1109/WKDD.2010.28 and in *Proceedings of IEEE Computer Society*, pp: 32-37.
123. Liu, D.R., and Shih, Y.Y., 2005a. Hybrid Approaches to Product Recommendation Based on Customer Lifetime Value and Purchase Preferences. *The Journal of Systems and Software*, 77(2), pp: 181–191.
124. Liu, D.R., and Shih, Y.Y., 2005b. Integrating AHP and Data Mining for Product Recommendation Based on Customer Lifetime Value. *Information and Management*, 42(3), pp: 387-400.

125. Liu, D.R., Lai, C.H., and Lee, W.J., 2009. A Hybrid of Sequential Rules and Collaborative Filtering for Product Recommendation. *Information Sciences*, 179(20), pp: 3505-3519.
126. Liu, Jiabin, Yan, Shuting, and Ren, Jiadong, 2010. The design of Storage Structure for sequence in incremental sequential pattern mining. In *Proceedings of Sixth International Conference on Networked Computing and Advanced Information Management (NCM)*, pp: 330 - 334.
127. Liu, Jiabin, Yan, Shuting, and Ren, Jiadong, 2011. The design of Frequent Sequence Tree in incremental mining of sequential patterns. In *Proceedings of IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS)*, pp: 679-682
128. Liu, Shuang, Liu, Yang, André, Étienne, Choppy, Christine, Sun, Jun, Wadhwa, Bimlesh, Dong, Jin Song, 2013. A Formal Semantics for Complete UML State Machines with Communications. *IFM 2013, Integrated Formal Methods, 10th International Conference, Turku, Finland, June 10-14. Lecture Notes in Computer Science 7940 Springer 2013, ISBN 978-3-642-38612-1, 978-3-642-38613-8* pp: 331-346.
129. Lo, David, and Khoo, Siau-Cheng, 2006. SMArTIC: towards building an accurate, robust and scalable specification miner. In *Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering, Portland, Oregon, USA, November 5-11, 2006*.
130. Luming, Yang, and Lao, Benxin, 2007. Customer relationship management. Chongqing University Press.
131. Luo, C., and Chung, S. M., 2005. Efficient mining of maximal sequential patterns using multiple samples. In *Proceedings of SDM*, pp: 64-72.
132. Malhotra, Navneet, and Gosain, Anjana, 2006. Better Quality of Decisions: Approaches to more Effective Analysis. *International Multiconference of Engineers and Computer Scientist, IMECS '06, June 20-22, 2006, Hong Kong, China. Lecture Notes in Engineering and Computer Science Newswood Limited 2006, ISBN 988-98671-3-3*, pp: 317-323.
133. Mangalampall, Ashish, and Pudi, Vikram, 2011. Fuzzy Associative Rule-based Approach for Pattern Mining and Identification and Pattern-based Classification. In *Proceeding of International World Wide Web Conference, March 28-April 1, 2011, Hyderabad, India*, pp: 379 - 383.

134. Mannila, H., Toivonen, H., and Verkamo, A. I., 1995. Discovering frequent episodes in sequences. In Proceedings of 1995 International Conference on Knowledge Discovery and Data Mining (KDD'95), pp: 210–215.
135. Maragatham, G., and Lakshmi, M., 2011. A weighted Particle Swarm Optimization Technique for optimizing association rules. 4th International Conference on Recent trends in Computing, communication and information technologies, Dec 9 -11, 2011 and in Proceedings of Springer (LNCS) - Communications in Computer and Information Science (CCIS) Series Part II, pp: 675-684.
136. Masegla, F., Cathalat, F., and Poncelet, P., 1998. The PSP approach for mining sequential patterns. In Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery in Databases (PKDD'98), Nantes, France, pp: 176–184. Lecture Notes in Artificial Intelligence, Springer Verlag.
137. Masegla, F., Poncelet, P., and Teisseire, M., 2003. Incremental mining of sequential patterns in large databases. *Data & Knowledge Engineering*, 46(1), pp: 97- 121.
138. Masegla, F., Poncelet, P., and Teisseire, M., 2009. Efficient mining of sequential patterns with time constraints: Reducing the combinations. *Expert Systems with Applications* 36(2), pp: 2677–2690.
139. Mhatre, A., Verma, M., and Toshniwal, D., 2009. Extracting sequential patterns from progressive databases: a weighted approach. In Proceedings of IEEE International Conference on Signal Processing Systems, pp: 788- 792.
140. Miglautsch, John R., 2002. Application of RFM Principles: What to Do with 1-1-1 Customers? *Journal of Database Marketing*, 9(4), pp: 319-324.
141. Myra, S., 2000. Web usage mining for Web site evaluation. *Communications of the ACM*, 43(8), pp: 127–134.
142. Ng, R.T., Lakshmanan, L.V.S., Han, J., and Pang, A., 1998. Exploratory mining and pruning optimizations of constrained association rules. In Proceedings of the 1998 ACM SIGMOD, International Conference on Management of Data.
143. Nguyen, S. N., Sun, X., and Orlowska, M. E., 2005. Improvements of IncSpan: Incremental mining of sequential patterns in large database. *Lecture Notes in Computer Science*, 3518, pp: 442-451.

144. Orlando, Salvatore, Raffaele, Perego and Claudio, Silvestri, 2004. A new algorithm for gap constrained sequence mining", In Proceedings of the ACM Symposium on Applied Computing, pp. 540 – 547, Nicosia, Cyprus, 2004.
145. Ouyang, Weimin, Huang, Qinhua, and Luo, Shuanghu, 2008. Mining Positive and Negative Fuzzy Sequential Patterns in Large Transaction Databases Structures. Fifth International Conference on Fuzzy Systems and Knowledge Discovery. In proceedings of IEEE, pp: 18-23.
146. Parmar, Jatin D., and Garg, Sanjay, 2007. Modified Web Access Pattern (mWAP) Approach for Sequential Pattern Mining. Journal of computer Science, 6 (2), pp: 46-54.
147. Parthasarathy , S., Zaki, M., Ogihara, M., and Dwarkadas, S., 1999. Incremental and Interactive Sequence Mining. In proceedings of the 8th International Conference on Information and Knowledge Management (CIKM'99), Kansas City, MO, USA, pp: 251–258.
148. Parvatiyar, A., and Sheth, J. N., 2001. Customer relationship management: Emerging practice, process, and discipline. Journal of Economic and Social Research, 3, pp: 1–34.
149. Pei, Jian, Han, Jiawei, and Wang, Wei, 2002. Mining sequential patterns with constraints in large databases. In Proceedings of the 11th ACM Int. Conf. on Information and Knowledge Management, McLean, Virginia, USA, pp: 18-25.
150. Pei, Jian, Han, Jiawei, and Wang, Wei, 2007. Constraint-based sequential pattern mining: the pattern-growth methods. Journal of Intelligent Information Systems, 28(2), pp: 133-160.
151. Pei, Jian, Han, Jiawei, Behzad, M.-Asl, Wang, Jianyong , Pinto, H., Chen, Qiming , Dayal, Umeshwar and Hsu, Mei-Chun, 2004. Mining Sequential Patterns by Pattern- Growth: The PrefixSpan Approach. IEEE transactions on Knowledge and Data Engineering. 16(11), pp: 1424-1440.
152. Peng, Wen-Chih, and Liao, Z. X., 2009. Mining sequential patterns across multiple sequence databases. Journal of Data & Knowledge Engineering, 68(10).
153. Piramuthu, S., 2004a. Evaluating Feature Selection Methods for Learning in Data Mining Applications. European Journal of Operational Research, 156, pp. 483-494.
154. Piramuthu, S., 2004b. Knowledge-based Framework for Validating Workflow Sequences. Journal of Information & Knowledge Management, 3(3), pp. 1-6.

155. Ren, J. D., and Li, Y., 2008. Mining weighted closed sequential patterns in large databases. *IEEE Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, pp: 640- 644.
156. Ren, J. D., and Zhou, X. L., 2006a. A new incremental updating algorithm for mining sequential patterns. *Journal of Computer Science*, 2(4), pp: 318-321.
157. Ren, J. D., and Zhou, X. L., 2006b. An efficient algorithm for incremental mining of sequential patterns, *Lecture Notes in Computer Science*, Springer, 3930, pp: 179-188.
158. Rigby, D.K., and Ledingham D., 2004. CRM Done Right, *Harvard Business Review*, November, pp: 118-129.
159. Sakurai, Shigeaki, Youichi, Kitahara, and Ryohei, Orihara, 2008. A Sequential pattern mining method based on sequential interestingness. *International Journal of Computational Intelligence*, 4(4), pp: 252-261.
160. She, Rong, Chen, Fei, Wang , K., Ester, Martin, Gardy, J., Fiona, L. and Brinkman, S. L., 2003. Frequent-subsequence based prediction of outer membrane proteins. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp: 436 – 445.
161. Sikora, R., and Piramuthu, S., 2007. Framework for Efficient Feature Selection in Genetic Algorithm based Data Mining. *European Journal of Operational Research*, 180, pp. 723-737.
162. Sinha, A. P., and Zhao, H., 2008. Incorporating domain knowledge into data mining classifiers: An application in indirect lending. *Decision Support Systems*, 46, pp: 287–299.
163. Sobh, Tarek, 2007. *Innovations and Advanced Techniques in Computer and Information Sciences*. Springer, ISBN 978-1-4020-6268-1.
164. Srikant, R., and Agrawal, R., 1996a. Mining quantitative association rules in large relational tables. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp: 1–12.
165. Srikant, R., and Agrawal, R., 1996b. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96)*, Avignon, France, pp: 3–17.

166. Tang, H., Fang, W., and Cao, Y., 2008. A simple method of classification with VCL components. In Proceedings of the 21st international CODATA Conference.
167. Thampi, Sabu M., and Sekaran, K. Chandra, 2008. Mobile Agents for Content-Based WWW Distributed Image Retrieval. CoRR abs/0803.0053.
168. Thampi, Sabu M., and Sekaran, K. Chandra, 2010. Protocols for Bio-Inspired Resource Discovery and Erasure Coded Replication in P2P Networks. CoRR abs/1009.1685.
169. Tsai, C. F., and Chen, M. Y., 2010. Variable selection by association rules for customer churn prediction of multimedia on demand. *Expert Systems with Applications*, 37, pp: 2006–2015.
170. Tsai, C. F., and Lu, Y. H., 2009. Customers churn prediction by hybrid neural networks. *Expert Systems with Applications*, 36, pp: 12547–12553.
171. Vineslas, Lionel, Symphor, J.-E., Mancheron, Alban and Poncelet, Pascal, 2009. SPAMS: A novel Incremental Approach for Sequential Pattern Mining in Data Streams. In Proceedings of EGC, pp: 1-6.
172. Vineslas, Lionel, Symphor, J.-E., Mancheron, Alban, and Poncelet, Pascal, 2010. SPAMS: A novel Incremental Approach for Sequential Pattern Mining in Data Streams. *Advances in Knowledge Discovery and Management, Studies in Computational Intelligence*, 292, pp: 201-216.
173. Wang, J., and Han, J., 2004. Bide: Efficient mining of frequent closed sequences. In Proceedings of 2004 International Conference on Data Engineering (ICDE'04).
174. Wang, Jianyong, Zhang, Y., Zhou, L., Karypis, George , and Aggarwal, C. C., 2007. Discriminating Subsequence Discovery for Sequence Clustering. In Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA.
175. Wang, K., and Tan, J., 1996. Incremental discovery of sequential patterns. In Proceedings of ACM SIGMOD'96 Data Mining Workshop, Montreal, Canada, pp: 95-102.
176. Wu, Di, Wang, Xiaoxue, Zuo, T., Sun, Tieli, and Yang, Fengqin, 2010. A Sequential pattern mining algorithm with time constraints based on vertical format. In Proceedings of IEEE- Information Science and Engineering (ICISE), 2010, 2nd International Conference on 4-5 December 2010.
177. Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan,

- G.J., Ng, Angus, Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand, D.J., and Steinberg, Dan, 2007. Top 10 algorithms in data mining Knowledge and Information System, Springer-Verlag, New York, 14(1), pp: 1-37.
178. Yafi, E., Al-Hegami, A. S., Alam, Afsar, and Biswas, Ranjit, 2012. YAMI: Incremental Mining of Interesting Association Patterns. The International Arab Journal of Information Technology, 9(6).
 179. Yan, X., Han, J., and Afshar, R., 2003. CloSpan: Mining closed sequential patterns in large datasets. In Proceedings of 2003 SIAM International Conference on Data Mining (SDM'03).
 180. Yang, S. Y., Chao, C. M., Chen, P. Z., and Sun, C. H., 2011. Incremental mining of closed sequential patterns in multiple data streams. Journal of Networks, 6 (5), pp: 728 – 735.
 181. Yu, J. X., Ou, Y., Zhang, C., and Zhang, S., 2005. Identifying interesting visitors through Web log classification. IEEE Intelligent Systems, 20, pp: 55–59.
 182. Yuan D., Lee, K., Cheng, H., Krishna, G., Li, Z., Ma, X., Zhou, Y., and Han, J., 2008. CISpan: comprehensive incremental mining algorithms of closed sequential patterns for multi-versional software mining. In proceedings of SIAM International Conference on Data Mining - SDM, pp. 84-95.
 183. Yun, Unil, 2008. An efficient mining of weighted frequent patterns with length decreasing support constraints. Knowledge-Based Systems, 21, pp: 741–752.
 184. Yun, Unil, and Leggett, John J, 2006. WSpan: Weighted Sequential pattern mining in large sequence databases. 3rd International IEEE Conference Intelligent Systems, pp: 512-517.
 185. Zadeh, L.A., 1965, Fuzzy sets. Information Control, 8, pp: 338-353.
 186. Zaki, M. J., 1998. Efficient enumeration of frequent sequences. In Proceedings of the 7th International Conference on Information and Knowledge Management, Washington, USA, pp. 68-75.
 187. Zaki, M. J., 2000. Sequence mining in categorical domains: Incorporating constraints. In Proceedings of the 9th International Conference on Information and Knowledge Management CIKM, pp: 422-429.
 188. Zaki, M. J., 2001. SPADE: An efficient algorithm for mining frequent sequences. Machine Learning, 42(1/2), pp: 31-60.

189. Zhang, M., Kao, B., Cheung, D. W., and Yip, K. Y., 2002. Efficient algorithms for incremental update of frequent sequences. In Proceeding of Sixth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '02), pp: 186-197.
190. Zhang, Z., Zhang, L. , Zhong S., and Guan J., 2008a. A new algorithm for mining sequential patterns. IEEE Fifth International Conference on Fuzzy Systems and Knowledge Discovery, pp: 625- 629.
191. Zhang, Zhennan, Qian, Xu, and Zhao, Yu, 2008b. Galois Lattice for Web Sequential Patterns Mining. In Proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application, PACIIA '08. Wuhan, pp: 102 - 106.
192. Zhao, Qiankun., and Bhowmick, Sourav S., 2003. Sequential Pattern Mining: A Survey. Technical Report, CAIS, Nanyang Technological University, Singapore, 118.
193. Zheng, Q., Xu, K., Ma, S., and Lv, W., 2002. The algorithms of updating sequential patterns. In Proceeding of 5th International Workshop on High Performance Data Mining (HPDM), in conjunction with 2nd SIAM Conference on Data Mining, USA.
194. Zhou, Mingming, and Xu, Yabo, 2010. Injecting Pedagogical Constraints into Sequential Learning Pattern Mining. In Proceedings of 10th IEEE International Conference on Advanced Learning Technologies, pp: 380-381.

Appendix A. Details and Characteristics of Real Dataset

msnbc.com anonymous web data

The real dataset has been taken from the UCI Machine Learning repository (<http://archive.ics.uci.edu/ml/datasets>). This data describes the page visits of users who visited msnbc.com on September 28, 1999. These visits are recorded in time order at the level of URL category. The typical sequential characteristic of the dataset makes it suitable for our use.

Data Characteristics

This data is of news-related portions of msn.com for the entire day of September, 28, 1999. It is retrieved from Internet Information Server (IIS) logs for msnbc.com where each sequence in the dataset corresponds to page views of a user during that twenty-four hour period. Each item in the sequence represents the user's request being made for a page. The user requests are recorded at the level of web page category, instead of at the level of URL. The categories as determined by a site administrator are "frontpage", "news", "tech", "local", "opinion", "on-air", "misc", "weather", "health", "living", "business", "sports", "summary", "bbs" (bulletin board service), "travel", "msn-news", and "msn-sports". However the data does not include any such page requests that is served through caching mechanism and hence were not recorded in the server logs.

Attribute Information:

Each web page category is mapped with an integer (used for its representation) starting with "1". For example, "frontpage" is associated with 1, "business" with 2, "news" with 3 and so on. Each row describes the order in which a single user hits the webpage.

Example: Data Format

```
2 1 4 5
2 1 1 3 8
3 6 5 4 7 10 2 3 3
5 2 11
1 7 6 4 8
6 11 5 2 1
4 9 6 3 1 1
6 12 3 1 4 7 3 2 9
6 7 7 7 6 6 8 8 8 8
6 9 4 4 4 10 3 10 5 10 4 4 4
```

Here, the first user hits "business", followed by "frontpage". The second user hits "business" once, followed by "frontpage" twice, "news" once.

Appendix B. Details and Characteristics of Synthetic Dataset

A set of synthetic data sequence is generated by a data generator similar in spirit to the IBM data generator designed for testing sequential pattern mining algorithms.

Data Characteristics

Each data sequence contains a sequence of item sets. However, different time values are assigned to the items in different item sets but the same time values to those in the same item sets. The data items occur in strictly sequential manner. So, the time value of first data item set in the sequence would always be less than the time value of the next data item set.

Attribute Information:

The data items are represented as integers, separated by comma ','. Items in the same item set are separated from other item set with the help of colon ':'. Two colons (: :) without any data items represent that there is no data for that time value. Each and every data sequence/ record is assigned a number starting from "1".

Example: Data Format

```
1 9,0,2,2,:0,5,:7,:5,:5,:5,4,:3,9,8,2,:5,8,:0,7,2,:
2 3,1,3,:6,4,::2,::0,1,:4,7,:1,0,6,:1,2,1,5,:0,7,
3 9,9,:2,3,:1,6,0,::9,:7,5,7,:2,1,5,8,:2,9,3,0,::
4 :6,:4,1,0,:3,3,2,6,:9,8,:2,:8,8,::7,:1,
5 ::2,:9,8,6,::9,9,0,:5,:7,1,8,:1,0,9,:6,
6 8,:6,2,1,8,:0,1,:3,::7,:6,:3,3,7,7,:
7 0,9,:5,0,::0,2,6,:6,4,1,:9,0,:9,:1,6,0,:2,4,:
8 3,5,7,0,:5,2,::9,2,8,:7,:0,3,3,6,:6,9,7,:6,7,:5,5,:4,0,
9 8,8,4,1,:0,2,3,9,:6,:8,9,9,:0,0,5,:0,6,5,:7,8,8,:0,7,2,4,:9,:
10 3,:7,1,0,8,:3,8,::4,2,:0,:8,7,::0,1,:9,2,
11 6,8,:7,3,6,9,:5,7,4,:4,::5,5,6,:2,8,6,:9,:2,1,8,2,
12 4,1,8,7,:4,1,::0,6,:8,7,2,:8,0,:1,3,6,::0,4,7,6,:7,6,7,
13 0,8,6,:1,1,4,5,:6,8,::2,::8,2,3,::5,6,7,:2,5,9,9,
14 6,:7,2,2,1,:4,3,:5,7,7,0,:9,:9,1,7,:8,5,:2,9,:1,3,7,:3,2,2,
15 2,0,:9,:8,1,::2,9,5,5,:6,2,7,0,:1,9,5,:2,0,7,:2,8,
16 9,8,:7,:2,:6,:2,::0,4,6,8,:3,:2,6,9,5,:4,8,
17 9,4,8,:8,2,5,6,:6,9,:4,3,3,:5,0,:4,::1,9,8,:3,7,1,
18 2,8,:6,5,3,:2,6,:5,8,:1,4,6,8,::7,::1,8,8,4,:
19 1,5,:8,3,4,7,:2,:3,7,1,6,:5,9,6,:0,3,:7,6,1,6,::2,4,4,:1,9,8,
20 ::3,4,:6,7,1,:1,:4,4,2,:5,3,7,::7,2,2,
```

Here, for the first data sequence, 9, 0, 2, 2, are data items of same item set and hence separated from other item set 0, 5, with a colon (:). Hence they have different time value, say T1 for the first data item set and T2 for the other, where $T1 < T2$ as the data item sets occur in sequential manner.