MINING ASSOCIATION RULES USING POPULATION BASED STOCHASTIC SEARCH ALGORITHMS

A THESIS

submitted to Pondicherry University in partial
fulfilment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE AND ENGINEERING

by

K.INDIRA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PONDICHERRY ENGINEERING COLLEGE
PUDUCHERRY – 605 014
INDIA

SEPTEMBER 2013

PONDICHERRY UNIVERSITY

PUDUCHERRY – 605014

CERTIFICATE

This is to certify that this thesis titled **"MINING ASSOCIATION RULES USING POPULATION BASED STOCHASTIC SEARCH ALGORITHMS"** submitted by **Mrs. K. INDIRA,** Department of Computer Science & Engineering, Pondicherry Engineering College, Puducherry, India, for the award of the degree of **Doctor of Philosophy** in **Computer Science and Engineering** is a record of bonafide research work carried out by her under my guidance and supervision.

This work is original and has not been submitted, in part or full to this or any other University / Institution for the award of any other degree.

Place: Puducherry
Date:

Dr. S. KANMANI
(SUPERVISOR)
Professor,
Department of Information Technology,
Pondicherry Engineering College,
Puducherry – 605 014,
India.

DECLARATION

I hereby declare that this thesis titled "**MINING ASSOCIATION RULES USING POPULATION BASED STOCHASTIC SEARCH ALGORITHMS** " submitted to the Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry, India, for the award of the degree of Doctor of Philosophy in Computer Science and Engineering, is a record of bonafide research work carried out by me under the guidance and supervision of Prof. S. Kanmani and that this has not formed the basis for the award of any other degree by any University / Institution, before.

Place: Puducherry **K. INDIRA**

Date:

ABSTRACT

Association Rule Mining (ARM) is a structured mechanism for unearthing hidden facts in large data sets and drawing inferences on how a subset of items influence the presence of another subset. These rules have many applications in areas ranging from e-commerce to sports to census analysis to medical diagnosis. The discovery of association rules is an extremely computationally expensive task and it is therefore imperative to have fast scalable algorithms for mining these rules.

The standard ARM methods such as: Apriori, Frequent Pattern (FP) growth tree, scans the whole dataset for each attribute match, increasing the input / output overhead of the system. The rules generated aim a single objective of accuracy alone in spite of generating vast number of rules. Pruning and summarization are needed to filter the significant rules. The efficiency of ARM can be enhanced by reducing the number of passes over the database, making the process multiobjective and sustaining the search space efficiently.

Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) provide solutions for the above requirements while mining association rules. In this research work, mining association rules using GA and PSO have been attempted. GA and PSO are evolutionary computing (EC) methods with effective population-based stochastic search algorithms that include heuristics and an element of non-determinism in traversing the search space. Both the above algorithms move from one point to another in the search space in a non-deterministic manner, guided by heuristics and using population generated from the dataset in consideration, rather generating them randomly. GA and PSO, when applied for association rule mining generate high quality association rules with good rule quality metrics.

Changes were introduced in the methodologies of GA and PSO for mining association rules. Tuning of control parameters in GA was performed and the same has been adapted for association rule mining using PSO. Effective methodologies such as GA with Elitism, dynamic neighborhood selection in PSO, Chaotic PSO, non-data dependent and data-dependent models of adaptation in GA and PSO have been proposed and implemented.

A hybrid approach combining the unique features of GA and PSO has also been proposed for effective mining of association rules. Memetic PSO with shuffle frog leaping algorithm (SFLA) for local search is proposed, to generate quality association rules, with enhanced accuracy. All these algorithms have been validated using data sets from the repository of UCI (University of California, Irvine). Experimental results and analysis confirm the promising and consistent behavior of the algorithms and methods that have been carried out in this research work.

ACKNOWLEDGEMENTS

First and foremost, I express my deepest sense of gratitude to my Supervisor **Dr. S. Kanmani**, Professor, Department of Information Technology, Pondicherry Engineering College, Puducherry, for her great supervision, support and continuous encouragement throughout the period of this research work. Her guidance and suggestions made me more confident to overcome many difficulties during my research.

I am extremely grateful to my Doctoral Committee members **Dr.V.Sitaramaiah**, Professor, Department of Mathematics, Pondicherry Engineering College, Puducherry, and **Dr. K. Porsezian**, Professor, Department. of Physics, Pondicherry University, Puducherry, for their invaluable suggestions and motivation, which was of great help in improving the quality of this research.

My sincere thanks to **Dr. D. Govindarajulu**, Principal, Pondicherry Engineering College, Puducherry, and **Dr. V. Prithiviraj,** former Principal, Pondicherry Engineering College, Puducherry, for permitting me to undergo this research, and extending the facilities available in this institute.

I wish to express my sincere thanks to **Dr. D. Loganathan**, Head, Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry, and **Dr. N. Sreenath**, former Head and Professor, Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry, for providing a peaceful and conducive working environment, during my period of study in this institute.

I am very grateful to **Dr. T. Sundararajan**, Professor, Department of Civil Engineering, Pondicherry Engineering College, Puducherry, for his kind help and invaluable suggestions in improving the quality of this thesis.

I would like to express my heartfelt thanks to my friend **Ms. P.Shunmugapriya,** for the moral support and encouragement given by her during my low time.

I could not forget the invaluable help and support extended by my co-researchers **Ms. P. Maragathavalli, Ms. E. Kodhai, Ms. R. Gowri, Ms. P. Salini, Mr. P. Radhakrishnan, Ms. J. Jayabharathi, Ms. M. Nandhini and Ms. N. Kayarvizhi.**

I would thank the **Technical** and **Non-technical** staff of Department of Information Technology, and Department of Computer Science and Engineering, Pondicherry Engineering College, for their support and cooperation.

I express my deepest sense of gratitude to my dear parents **Ms. V. Vathsala** and **Mr. R. Kuppusamy.** Nothing could ever eclipse the true love, the unconditional dedication, the consistent support and sacrifices, they have shown me throughout my life. There are not enough words to thank them, but there are abundant feelings to cherish forever.

I express my heartfelt gratitude to my beloved husband **Mr. V. Sivakumar** who is always there to provide me his support and special thanks for always backing me up and encouraging me to struggle on.

I am extremely grateful to my dearest daughter **S.Harini**, for her love and moral support. Heartfelt thanks to her, for all the sacrifices she made, during the entire period of this research.

Specially, I want to appreciate the support from my other family members for their blessings, invaluable support and encouragement. The appreciation cannot be expressed in words. Without their support and help, all the things I have are impossible.

At last, I thank the God Almighty, for all the blessings he has showered on me.

K.INDIRA

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.

CHAPTER NO.	TITLE	PAGE NO.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACO	-	Ant Colony Optimization
AGA	-	Adaptive Genetic Algorithm
AIS	-	Agrawal, Imielinski and Swami Algorithm
APSO	-	Adaptive Particle Swarm Optimization
APSO+SFLA	-	Adaptive Particle Swarm Optimization with Shuffled Frog Leaping Algorithm
AR	-	Association Rules
ARM	-	Association Rule Mining
BPSO	-	Binary Particle Swarm Optimization
CARMA	-	Continuous Association Rule Mining Algorithm
CPSO	-	Chaotic Particle Swarm Optimization
DIC	-	Dynamic Itemset Counting
EA	-	Evolutionary Algorithms
EC	-	Evolutionary Computing
EMA	-	Evolutionary Memetic Algorithm
EMOGA	-	Elitist Multi-Objective Genetic Algorithm
FP	-	Frequent Pattern
GA	-	Genetic Algorithm
GCPSO	-	Guaranteed Convergence Particle Swarm Optimization
GPSO	-	GA/PSO Hybrid
ITL Mine	-	Item Trans Link Miner
KDD	-	Knowledge Discovery in Databases
LS	-	Local Search
MA	-	Memetic Algorithm
MPSO	-	Memetic Particle Swarm Optimization
PSO	-	Neighborhood Selection in Particle Swarm Optimization

PA	-	Predictive Accuracy
PSO	-	Particle Swarm Optimization
PSO + SFLA	-	Particle Swarm Optimization with Shuffled Frog Leaping Algorithm
Q-QPSO	-	Quantum behaved Particle Swarm Optimization
RARM	-	Rapid Association Rule Mining
RWDE	-	Random Walk with Direction Exploitation
SAPSO1	-	Self Adaptive Particle Swarm Optimization 1
SAPSO2	-	Self Adaptive Particle Swarm Optimization 2
SACPSO	-	Self Adaptive Chaotic Particle Swarm Optimization
SFLA	-	Shuffled Frog Leaping Algorithm
SI	-	Swarm Intelligence
UCI	-	University of California, Irvine
VIPER	-	Vertical Itemset Partitioning for Efficient Rule
WPSO	-	Weighted Particle Swarm Optimization

CHAPTER 1

INTRODUCTION

1.1 GENERAL

The amount of data kept in computers is growing at a phenomenal rate. It is estimated that the amount of data available in the world is doubling, every 20 months (Frawley et al. 1992). The above data includes industrial, medical, financial and other databases, which constitute an invaluable resource / repository of useful knowledge. Hence there is also a consistent demand from users of these data for more sophisticated and useful information. Simple structured languages (like Simple Query Language - SQL) are not adequate to support the above ever increasing demand. This has led to the challenge of finding new techniques to extract useful patterns from such a huge database. Against this backdrop data mining has emerged as a new research area to meet this challenge and has also received a lot of research focus (Witten and Frank 2005).

Association rule mining (ARM) is one of the dominating data mining technologies. ARM is a process of finding associations or relations between data items or attributes in large datasets. The mining of association rules is a computationally expensive task. Further, the databases are typically very large. It is therefore imperative to have fast and scalable techniques for mining them. In this thesis, efficient techniques for discovering quantitative and qualitative association rules from large databases, efficiently and effectively, are presented.

1.2 DATA MINING

Association rule (AR) discovery is part of a larger field of study called data mining a field that consists of techniques to automatically find interesting patterns and trends in large collections of data. The explosive growth in stored data has generated an urgent need for new techniques and automated tools that can intelligently assist in transforming the vast amounts of data, into useful information and knowledge.

Knowledge Discovery in Databases (KDD) is the automated extraction of novel, understandable and potentially useful patterns which are implicitly stored in large databases, data warehouses and other massive information repositories. KDD is a multi-disciplinary field, drawing attention from areas including database technology, artificial intelligence, machine learning, neural networks, statistics, pattern recognition, information retrieval, high-performance computing and data visualization.

Data mining is an essential step in the process of knowledge discovery in databases, in which intelligent methods are applied in order to extract patterns. Many types of “interesting patterns" have been identified in research literature and ARs constitute one such type. Data mining tasks to find these various patterns include:

- Characterization: Data characterization is a summarization of the general characteristics or features of a user-specified target class of data.
- Discrimination: Data discrimination is a comparison of the general features of a user-specified target class data objects with the general features of objects from one or a set of (user-specified) contrasting classes.

- Association Analysis: Association analysis is the discovery of ARs showing attribute-value conditions that occur frequently together in a given set of data. Association analysis is widely used for market basket or transaction data analysis and forms the subject matter of this thesis.

- Classification and Regression: Classification is the process of finding a set of models that describe and distinguish data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. While classification predicts a categorical value, regression is applied, if the field being predicted comes from a real-valued domain. Common applications of classification include credit card fraud detection, insurance risk analysis, bank loan approval, etc.

- Cluster Analysis: Objects in a database are clustered or grouped based on the principle of maximizing intraclass similarity and minimizing interclass similarity. Applications of clustering include demographic or market segmentation for identifying common traits of group of people, discovering new types of stars in datasets of stellar objects, and so on.

- Outlier Analysis: Outliers are data objects that do not comply with the general behavior or model of the data. Most data mining methods discard outliers as noise or exceptions. However, in some applications such as fraud detection, the analysis and mining of outliers is crucial.

- Evolution Analysis: Data evolution analysis describes and models regularities or trends for objects, whose behavior changes over time. Although this analysis may include any of the above functionalities on time-related data, distinct features of such an analysis include time-series data analysis, sequence or periodicity pattern matching, and similarity-based data analysis.

Association analysis is one of the dominating data mining technologies. The ARs mined is useful in classification (Liu et al. 1998) and clustering (Han et al. 1997). ARM is the process of finding associations or relations between data items or attributes in large databases, bringing important information to the surface.

1.3 ASSOCIATION RULE MINING

ARM is an undirected or unsupervised data mining technique, which works on massive data, and it produces clear and understandable results. ARM is aimed at finding regularities in data. ARM research was originally proposed almost a decade ago (Agrawal et al. 1993), and since then has attracted enormous attention in both academia and industry. ARM identifies associations (patterns or relations) among database attributes and their values. It is a pattern-discovery technique which does not serve to solve neither classification problems (it does not classify samples into some target classes) nor prediction problems (it does not predict the development of the attribute values).

ARM generally searches for any associations among any attributes present in the database. An example for association rule (AR) can be the following (Wijsen and Meersman, 1998): 'if a customer buys a toothbrush, then he also probably buys toothpaste (in the same transaction)', the rule can be written as: {toothbrush} $\rightarrow$ {toothpaste}.

AR is commonly understood as an implication $x \rightarrow y$ in a transaction database $D = \{t_1, t_2, \ldots, t_m\}$(Agrawal et al. 1993). Each transaction $t_i \in D$

contains a subset of items I = {$i_{1,}\ldots, i_n$}; x and y are disjoint itemsets, it holds x; y ⊆ I and x∩y=Φ. The left hand side of this implication is called antecedent and the right hand side is referred as consequent. The transaction database D can also be viewed as a boolean dataset, where the boolean values of attributes in records express occurrence of items in transactions.

There are two basic measures for ARs: support and confidence. Support of an association rule is defined as the percentage/fraction of records that contain $x \cup y$ to the total number of records in the database. Support is calculated using the following equation

$$support(x) = \frac{No.of\ transactions\ containing\ x}{Total\ No.of transactions} \qquad (1.1)$$

Confidence of an AR is defined as the percentage/fraction of the number of transactions that contain x∪y to the total number of records that contain x, where if the percentage exceeds the threshold of confidence an interesting association rule x→y can be generated. Confidence is a measure of strength of the association rule.

$$confidence\ (x\ \rightarrow y) = \frac{support(xy)}{support(x)} \qquad (1.2)$$

To identify the ARs, the user has to preset a threshold that segregates frequently observed patterns from infrequent patterns. This threshold is called minimum support threshold. A set of items that appear together above this minimum support threshold, are searched. Rules connecting two sets of frequently observed items, appearing together above a minimum support threshold are found and, in many cases, a second measure of interestingness such as confidence, is used to further filter the rules found for interesting association rules. The confidence measure is called minimum confidence.

This classic approach discovers the association rules among the frequently observed patterns in a set of transaction records.

1.4 POPULATION BASED STOCHASTIC SEARCH METHODS

Stochastic search is a class of search methods which includes heuristics and an element of nondeterminism in traversing the search space. Stochastic search algorithm moves from one point to another in the search space in a nondeterministic manner, guided by heuristics. The next move is partly determined by the outcome of the previous move. Population based stochastic search algorithms employ a population of individuals to solve the problem on hand.

Population based search algorithms were proposed for solving unconstrained optimization problems. These search methods belong to the category of metaheuristic optimization algorithms in general, and naturally inspired computation methodologies, in specific. Grouped under the term evolutionary computation or evolutionary algorithms (Back 1996), are the domains: Genetic Algorithms (GA) (Holland 1975), Evolution Strategies (Rechenberg 1973; Schwefel 1977), Evolutionary Programming (Fogel, Owens and Walsh 1966), Genetic Programming (Koza 1992), Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995) and Ant Colony Optimization (ACO) (Dorigo et al. 1991).

Stochastic optimization has been the popular choice for solving complex and intricate problems, which are otherwise difficult to solve by traditional methods. Among stochastic search algorithms, population based methods namely GA and PSO follow the multiobjective requirements of, reduced number of scan on the database while mining ARs (Jacinto et al. 2002; Deepa Shenoy et al. 2003). GA and PSO move from a set of points (population) to another set of points in a single iteration with likely improvement using a combination of deterministic and probabilistic rules. Both the above methods generate ARs with better predictive accuracy and reduced complexity.

GA can be viewed as a general-purpose search method, an optimization method, or a learning mechanism, based loosely on the Darwinian principles of biological evolution, reproduction and "the survival of the fittest" (Goldberg 1989). GA maintains a set of candidate solutions called population and repeatedly modifies them. At each step, GA selects individual from the current population to be the parent and uses them to produce children for the next generation. Over successive generations, the population evolve towards an optimal solution and remains in the genome composition of the population over traits with weaker undesirable characteristics. GA is well suited and has been extensively applied to, solve complex design optimization problems as it can handle both discrete and continuous variables, nonlinear objective and constrain functions, without requiring gradient information (Varsek at al. 1993; Abdel-Magid and Abido 2003; Abido 2005)

PSO is inspired by the ability of flocks of birds, schools of fish, and herds of animals, to adapt to their environment, find rich sources of food, and avoid predators by implementing an information-sharing approach. PSO technique was invented in the mid 1990s while attempting to simulate the choreographed, graceful motion of swarms of birds as part of a sociocognitive study investigating the notion of collective intelligence in biological populations (Kennedy and Eberhart, 1995). In PSO, a set of randomly generated solutions propagate in the design space towards the optimal solution over a number of iterations based on large amount of information about the design space that is assimilated and shared by all members of the swarm (Kennedy and Eberhart 2001).

1.5 PROBLEM IDENTIFICATION

The standard ARM methods such as: Apriori (Agrawal et al. 1993; Agrawal and Srikant 1994), Frequent pattern growth tree (Piatetsky-Shapiro 1991; Han et al. 2000) scans the whole dataset for each attribute match, thus increasing the input/output overhead of the system. The rules generated aim a single objective of accuracy alone, whereas, the number of rules generated is vast. Pruning and summarization are needed to obtain the significant rules.

The application areas of AR mining vary from market analysis to business intelligence, which has now been extended to epidemiology, clinical medicine, fluid dynamics, astrophysics, and crime prevention. Hence, the accuracy of the ARs mined and the relationship between attributes has become an important issue. There is a clear need for developing automatic methods for extracting knowledge from data that not only have a high predictive accuracy (PA) but also, are comprehensible by users (Fayyad et al. 1996; Freitas 1997; Freitas 1999). Evolutionary algorithms (EAs) have inspired many research efforts for optimization as well as rule generation (Fonseca and Fleming1995; Robinson et al. 2002).

GA and PSO provide robust and efficient approach in exploring large search space and have proven to be successful in solving difficult problems. The success or failure of any population based algorithm depends on its ability to establish proper trade-off between exploration and exploitation. A poor balance between exploration and exploitation may result in a weak optimization method, which may suffer from premature convergence, trapping in a local optima and stagnation.

Setting the right values for the parameters involved and modifying the basic operations of the methods maintain the search space effectively, thereby achieving a balance between exploration and exploitation. Therefore, improving the efficiency of ARM algorithms using GA and PSO by tuning the parameters and modifying the methodologies appropriately, have been set as the focus of the present study.

1.6 OBJECTIVES OF THE RESEARCH

Motivated by the open issues in ARM using GA and PSO, the objectives of our research work are formulated as:

- ➢ To develop efficient methodology for mining ARs using population based search methods namely
    - Genetic Algorithm with parameter tuning and modifications in methodology
    - Particle Swarm Optimization with parameter tuning and modifications in swarm movements
- ➢ To formulate an effective methodology for mining ARs
    - By Combining the unique features of GA and PSO
    - With effective local search namely Shuffle Frog Leaping Algorithm (SFLA) in PSO
- ➢ To validate the generated ARs quantitatively through predictive accuracy and qualitatively using rule measures.

1.7 ORGANISATION OF THE THESIS

This thesis is organized in seven chapters. Chapter 1 gives a preface to the proposed research work, motivation that lead to the research, problem statement and a brief description of the basic concepts underlying this research work.

Chapter 2 depicts the review of state of the art research related to ARM. The classical methods and population based search algorithms for mining rules are reviewed. The preliminary works that served as the main motive for most of present work have been narrated in this chapter.

Chapter 3 gives an overview of the important concepts like stochastic search, GA and PSO. The details of the datasets from the repository of University of California, Irvine (UCI) which have been used in this study for experimental purpose, are discussed. Further, the various evaluation parameters that have been used to measure the performance of ARM are also included, in this chapter.

Chapter 4 describes the methodology of mining ARs using GA. The objective function applied for ARM is discussed. The tuning of control parameters of GA for association rue mining is done to understand the role of these parameters. The concept of Elitism is introduced in GA for mining ARs. The proposal of Adaptive GA, where the mutation probability is made self adaptive for ARM, is also presented and discussed.

Chapter 5 elaborates the methodology of mining ARs using PSO. ARM through PSO available in the literature is taken and the effect of inertia weight on the performance has been analyzed in this chapter. The proposal of Chaotic PSO for ARM and the dynamic selection of the neighborhood for the local best particle to replace the particles personal best are also explained in this chapter. ARM by adaptive parameter setting for PSO has been proposed for ARM. The acceleration coefficients and the inertia weight are adapted dynamically. Two approaches, namely, data independent and data dependent approaches for ARM using PSO are presented.

Chapter 6 elaborates the design and development of hybrid algorithms for ARM. GA and PSO are hybridized effectively to optimize the ARM process. Weak local search, the major drawback of PSO is overcome by proposing a memetic algorithm, combining PSO and SFLA for mining ARs.

Chapter 7 contains the conclusion of this research work and discusses the enhancements made to design GA and PSO for ARM. The scope for future work has also been highlighted.

CHAPTER 2

LITERATURE REVIEW

2.1 GENERAL

In this chapter, current and past research on generating ARs is reviewed. The overview of the related topics are presented under three themes; (a) review of general approaches for mining ARs, (b) review on mining ARs based on population based stochastic search algorithms and (c) review on hybrid methodologies using GA and PSO.

2.2 TRADITIONAL METHODS FOR MINING ASSOCIATION RULES

In this section the traditional ARM algorithms, have been reviewed. Most of the approaches depend on finding frequent patterns and then search for correlations between them to generate ARs.

2.2.1 Agrawal, Imielinski and Swami (AIS) Algorithm

The AIS was the very first algorithm proposed for ARM. It was developed by Agrawal, Imielinski and Swami (Agrawal et al. 1993) and hence referred to as AIS algorithm. This is a “multi-pass" algorithm in which candidate itemsets are generated while scanning the database, by extending known-frequent itemsets with items from each transaction. An estimate of the supports of these candidates is used to guide whether these candidates need to be extended further to produce more candidates. The efficiency of the AIS algorithm was improved by the addition of an estimation method to filter candidate itemsets that had no chance of being above the threshold (Zhao et al. 2003)

2.2.2 Apriori and Apriori-based Algorithms

The AIS algorithm was followed by the Apriori algorithm (Zhao and Bhowmick, 2003) that was shown to perform better than AIS by an order of magnitude. The most important aspect of Apriori is to completely incorporate the subset frequency-based pruning optimization. It utilizes a data structure called hash tree to store the counters of candidate itemsets. The main drawback in this algorithm is that it performs n passes over the database, where, n is the length of the longest frequent itemset. Mao (2001) proposed a methodology where candidate itemsets are generated by joining the frequent itemsets level-wise and the candidates are pruned according to the Apriori properties to reduce the search space.

The original Apriori algorithm has to deal with the drawback of multiple scan and thus the candidate generation process was complex. Several modifications have been proposed to the Apriori algorithm to reduce the multiple scans and some of the modified approaches namely (1) Apriori-TID, (ii)Apriori-Brave, (iii) Apriori-DF and (iv) F-Apriori are presented below.

Apriori-TID extends the original Apriori algortihm by constructing a counting base set during first pass through the dataset. This counting base set is used later for the determination of the frequent itemsets (Ceglar and Roddick 2006). A hybrid structure combining Apriori and Apriori-TID was proposed, eliminating the need for construction a new structure. This was done using a hash tree like structure containing pointers, instead of holding counters (Hipp et al. 2000).

Bodon (2003) proposed Apriori-Brave, featuring delayed accrual and organization switching based on memory usage. This also featured an implementation of dataset pruning to remove invalid items from the dataset. Perfect hash pruning was proposed by Ozel and Guvenir (2001), including optimization through the use of perfect hashing, when it comes to creating the hash table. This is considered to effectively eliminate a collision which means to avoid recounting the occurrence or support of an itemset.

Another Apriori-based approach proposed is Apriori-DF (Pijls and Bioch 1999), in which a novel extension technique was used to calculate the valid subtree from the already formed subtree. This algorithm builds the initial itemsets through the use of traditional Apriori method.

FApriori (Patel et al. 2013) method combines confidence and support as constraints with Apriori algorithm and reduces (i) storage required to store candidate and (ii) the execution time by reducing CPU time. CPU time is saved by reducing candidate sets size and time required to calculate the support of each candidate. The concept of checkpoint based on support value has been introduced to reduce the execution time and overall storage space required to store candidate generated.

2.2.3 FP-Tree and FP-Growth / FP-Tree Based Algorithms

One of the efforts of the various works to overcome the Apriori drawback is the design of tree structures for use in ARM. Frequent Pattern Tree (FP-Tree) was first introduced by Han and Pei (2000a). This approach requires only two passes through the database to generate the frequent itemsets and does so, without the need to generate candidate itemsets.

While the FP-Tree was shown to be effective and efficient, the size of the tree usually increases exponentially as the number of unique items increase. To address this shortcoming FP-Growth tree was developed (Grahne and Zhu 2003) and it uses an extra array based structure, to decrease the number of tree traversals required during analysis. Another FP-Growth based approach was proposed by Wang et al. (2002), which is a top down variation to the base FP-Growth approach. This approach is said to alleviate the need or demand to generate conditional pattern bases and physical projections of the tree.

H-Mine proposed by Pei, Han and Lakshmanan (2001) was designed to extend the pattern growth concepts in FP-Growth even though it uses an array-based hyperstructure. The population of the hyperstructure occurs in a manner similar to FP-Growth, where, the first valid itemset and the second create the H-Struct hyperstructure from that itemset. Item Trans Link Miner (ITL-Mine) (Gopalan and Suchayo 2002) optimizes H-Mine by maintaining a static set of links in the hyperstructure. This needs one scan of the database reducing Input / Output demands.

A hybrid pattern growth algorithm known as Opportunistic Projection (Liu et al. 2002), works by constructing a prefix tree that contains the associated counts, through the use of breath first tracking and database reduction techniques, until the data structure can be held in memory.

2.2.4 Other Association Rule Mining Methods

Rapid association rule mining (RARM) algorithm is an approach that uses a tree structure to represent the database and does not utilize candidate generation process. It was first proposed by Das, Ng and Woon, which was projected as being faster than the exiting algorithms (Das et al. 2001). The partitioning strategy was introduced by Savasere et al. (1995), wherein the database is logically divided into a number of disjoint partitions. The partition algorithm requires at most two passes and is based on the observation that an item-set can globally be frequent over the entire database, if it is locally frequent in at least one partition.

Sampling algorithm (Toivonen 1996) first mines a random sample of the database to obtain itemsets that are frequent within the sample. These itemsets could be considered as a representative of the actual frequent itemsets in applications, where approximate mining results are sufficient. In order to obtain accurate mining results, this algorithm requires one or two scans over the entire database. A variation of partition was proposed by Lin and Dunham (1998), which made use of the cumulative count of each candidate to achieve an illusion of a “large partition". At any instant, it stores only the candidates that are frequent over their respective large partitions.

In Dynamic Itemset Counting (DIC) method for mining ARs (Brin et al. 1997), generates candidates are generated and remove after every M transaction, where M is a parameter to the algorithm. Although it is a multi-pass algorithm, it was shown to complete within two passes typically. Continuous Association Rule Mining Algorithm (CARMA) proposed by Hidber (1999) is a 2-pass algorithm that has the feature of dynamically generating and removing candidates after each tuple of the database is processed. Though a novel approach, the CARMA algorithm suffers from the drawbacks of tuple-by-tuple approaches.

While the above algorithms were primarily horizontal (tuple) based approaches, the MaxClique (Zaki et al., 1997) algorithm is designed to efficiently mine databases that are available in a vertical layout. Unlike earlier vertical mining algorithms, which were subject to various restrictions on the underlying database size, shape, contents or the mining process, the Vertical Itemset Partitioning for Efficient Rule (VIPER) (Shenoy et al., 2000) algorithm does not have any such restrictions. It includes many optimizations to enable efficient processing and was shown to outperform earlier vertical mining algorithms. It also scales well with the database size.

2.3 STOCHASTIC SEARCH METHODS FOR MINING ASSOCIATION RULES

This section focus on reviewing the literature for mining ARs using the stochastic search methods namely; GA and PSO.

2.3.1 Genetic Algorithm for ARM

The traditional methods of mining ARs face two major drawbacks namely, it needs more than one pass / scan of the database to generate frequent itemsets. Processing is to be done to generate rules from these frequent itemsets. To overcome these drawbacks, evolutionary computation is the solution. Evolutionary computation methodologies provide robust and efficient approach in exploring large databases and are more suitable for multiobjective optimization problems.

GA has received wide attention over the past two decades on various areas of optimization. This subsection focuses on performance analysis of the existing work on mining ARs based on GA.

A GA based data mining system suitable for both supervised and certain types of unsupervised knowledge extraction from large and possibly noisy databases was proposed by Cattral, Oppacher and Deugo (1999). This differs from a standard GA in several crucial aspects, including the following: (i) its ‘chromosomes’ are variable-length symbolic structures, (ii) besides typed crossover and mutation operators, it uses macromutations as generalization and specialization operators to efficiently explore the space of rules, and (iii) it evolves a default hierarchy of rules.

Zhu Yu et al. (2009) proposed a method of mining multi-dimensional ARs based on artificial immune algorithm, and the algorithm has faster speed of mining multi-dimensional ARs. Wu Zhao-hui et al. (2005) presented a new ARM algorithm based on improved simulated annealing genetic algorithm, wherein, adaptive crossover probability and mutation probability, and restrained premature convergence were effectively used.

Salleb-Aouissi et al. (2007) developed a system QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules. This system is based on a GA that dynamically discovers "good" intervals in ARs by optimizing both the support and the confidence. Dehuri et al. (2006) presented a fast and scalable multi-objective ARM technique using GA from a large database. Confidence factor, comprehensibility, and interestingness were thought of as multi objectives of the ARM problem and were treated as the basic input to the GA.

The Elitist Multi-Objective Genetic Algorithm (EMOGA) has been proposed for mining classification rules from large databases (Dehuri et al. 2008). It uses the concept of elitism to retain the high quality rules. A hybrid crossover operator combining the best attribute of single point and uniform crossover is used. Extracting ARs from data with both discrete and continuous attributes, is an important problem in Knowledge Discovery in Databases (KDD). A new model of immune GA was formulated for solving this problem (Yang 2010). This algorithm uses three-segment chromosomes, integrating the discretization, attributes reduction and mining ARs. The immune mechanism is introduced into GA to avoid premature phenomenon and improve its efficiency.

To solve the two problems of low efficiency and the real demanded rules, a quick response data mining method was proposed (Wenxiang et al. 2008). This method avoids mining rules through huge candidate itemsets. Further it mines maximal frequent itemsets and produces rules, after users select their interested maximal frequent itemsets. It also scans the database for obtaining real support and confidence of these rules. By adopting the global searching character of GA, it can be used to mine the maximal frequent itemsets in set time or set number and show them to users.

The minimum threshold for support and confidence were usually set for generating ARs. Setting of the minimum values for these thresholds is the primary key for the success of the algorithms. In GA, this threshold values need not be set.

Alatas and Akin (2006) proposed GA methodology that performs a database-independent approach, not relying upon the minimum support and the minimum confidence thresholds for each database. Instead of randomly generated initial population, uniform population that forces the initial population not to be far away from the solutions and distributes it in the feasible region uniformly, is used. An adaptive mutation probability, a new operator called uniform operator that ensures the genetic diversity and an efficient adjusted fitness function are used for mining all interesting ARs from the last population in only single run of GA.

A GA based strategy for identifying ARs without specifying actual minimum support was proposed by Xiaowei et al. (2009). In this approach, an elaborate encoding method was developed, and the relative confidence was used as the fitness function. Furthermore, this strategy was expanded to cover quantitative AR discovery. For efficiency, a generalized FP-tree is designed to implement this algorithm.

Hamid et al. (2011) proposed a method based on GA without taking the minimum support and confidence into account. In order to improve the algorithm efficiency, the FP-tree algorithm was employed. Yan proposed a method based on GA without considering minimum support (Yan et al., 2008). This method uses an extension of elaborate encoding while relative confidence is the fitness function. A public search is performed based on GA and instead of using minimum support, a system automation procedure was used.

The control parameters of GA play a vital role in the performance of the algorithms and are dependent on the problem for which the algorithm is applied. A Self-Adaptive Migration Model GA was proposed by Srinivasa et al. (2007), where the parameters population size, the number of points of crossover and mutation rate for each population were adaptively fixed. Further, the migration of individuals between populations is decided dynamically. Thus the adaptation of parameters is made problem-dependent.

The issue with GA is that it easily leads to premature convergence and into the plight of local optimum, also consumes a large amount of time to search. For resolving these issues, Guo and Zhou (2009) proposed an algorithm by introducing an adaptive mutation rate and improving the methods of individual choice, for mining ARs. Min et al. (2011) proposed a method of mining multidimensional AR based on the Adaptive Genetic Algorithm (AGA) with crossover matrix and mutation matrix. In this ARM system, selection, mutation, and crossover are all parameter-free in the evolution process.

Mining large datasets to obtain classification models with prediction accuracy (PA) can be a very difficult task, as the size of the dataset can make data mining algorithms inefficient. To meet this requirement an efficient distributed GA for classification rule extraction in data mining, which promotes a new method of data distribution in computer networks, was proposed (Miguel Rodriguez et al. 2011). This was done by spatial partitioning of the population into several semi-isolated nodes, each evolving in parallel, and possibly exploring different regions of the search space.

2.3.2 Particle Swarm Optimization for ARM

Traditional methods of ARM generate more number of rules. Pruning is to be done to filter and obtain the required rules. Getting limited relevant rules is of more concern. Therefore, for generating optimum number of ARs with minimum complexity and at reduced time, PSO technique was used.

Esmaeli et al. (2008) proposed a method for rule mining with PSO without any modifications in the basic PSO steps proposed by Eberhart and Kennedy. A rough particle swarm optimization algorithm, based on the notion of rough patterns using rough values defined with upper and lower intervals that represent a range or set of values, was proposed by Alatas and Akin (2008a). They also proposed another numeric ARM method called chaos rough particle swarm algorithm (Alatas and Akin 2008b)

Kuo et al. (2011) proposed PSO technique to obtain quality ARs. Since defining support and confidence measure is a challenging issue in pattern mining, the above authors have introduced a novel approach for suggesting suitable threshold values for generating quality rules. Initially the data was transformed into binary values and using suitable fitness function in the PSO, the rules were generated. A New Quantum behaved Q-QPSO (Quantum behaved Particle Swarm Optimization) algorithm starts like the usual PSO (Mourad Ykhlef, 2011). At the end of the iteration, the quadratic interpolation recombination operator was invoked to generate a new swarm particle.

Nandhini et al. (2012) proposed a technique to reduce the quantity of the rules, without compromising the usefulness factor thereby improving the computational efficiency of rule mining. The above framework reduces the number of rules by combining mining and post-mining techniques. PSO was used in the mining process to compute an optimal support and confidence parameters, and a collection of strong rules was then obtained using these computed parameters.

Gupta (2012) used weighted particle swarm optimization for ARM for finding the suitable threshold values for minimum support and minimum confidence. These parameters were used for extracting valuable information. Asadi et al. (2012) used PSO for finding the threshold values for the Apriori algorithm. Xios (2012) proposed ARM framework based on self-adaptation swarm, applying PSO on mass stock data. It cannot only dig hidden rules behind deal data but also verify the efficiency of the algorithm and dig ARs totally.

Maragatham and Lakshmi (2012) proposed an approach where the main processes involved were calculation of the support and confidence from the input data, rule generation, initialization, updation of the velocity, position of the rules and evaluation of fitness function.

Most approaches to ARM assume that the items within the dataset have a uniform distribution. Therefore, weighted ARM was introduced to provide a notion of importance to individual items. These approaches require users to assign weights for each item. This is infeasible when we have millions of items in a dataset. To meet the above aspiration a novel method called, weighted ARM using PSO was proposed (Pears and Koh 2012), which uses PSO to assign meaningful item weights for ARM.

A Binary Particle Swarm Optimization (BPSO) based AR miner generates the ARs from the transactional database by formulating a combinatorial global optimization problem (Sarath and Ravi 2013), without specifying the minimum support and minimum confidence, unlike the Apriori algorithm. This algorithm generates the best *M* rules from a given database, where *M* is a given number and the quality of the rules was measured by a fitness function, defined as the product of support and confidence.

2.4 HYBRID APPROACHES OF GA AND PSO

Hybridization is a growing area of intelligent systems research, which aims in combining the desirable properties of different approaches to mitigate their individual weaknesses. The scope of this research is applying GA and PSO; population based stochastic search methods for ARM. In literature the hybrid of PSO and GA for ARM has seen to be less investigated and reported.

The reviews considered are hybrids of GA and PSO for optimization of other application areas. Both approaches being population based, such hybrids are readily formulated. Angeline (1998a) applied a tournament selection process that replaced each poorly performing particle's velocity and position with those of better performing particles. This movement in space, improved the performance in three out of four test functions, but moved away from the social metaphor of PSO. Brits et al. (2001) used Guaranteed Convergence Particle Swarm Optimization (GCPSO) (van den Bergh and Engelbrecht, 2002) in their niching PSO algorithm. Borrowing techniques from GAs, the NichePSO initially sets up sub-swarm leaders by training the main swarm using Kennedy's (1997) cognition only model.

Gaussian mutation was combined with velocity and position update rules by Higashi and Iba (2003) and was tested on unimodal and multimodal functions. The hybrid achieved better results than those of GA and PSO alone. Juang (2004) also incorporated mutation alongside crossover and elitism. Inspiration for the Cooperative PSO (van den Bergh and Engelbrecht, 2004) was provided by the more specialized Cooperative Coevolutionary Genetic Algorithm developed by Potter and de Jong (1994), which aims to minimize the exponential increase of difficulty in optimizing problems with higher dimensions through targeting each dimension as a single dimensional problem.

A new hybrid GA to solve the job shop scheduling problem was introduced (Tang et al. 2010) where PSO algorithm was used to get the initial population and evolutionary genetic operations solve the lack of the major evolution direction problem of GA. To improve the performance of PSO an enhanced evolutionary algorithm based on the characteristics of PSO, multi-parent crossover algorithm and differential evolution was proposed by Dhazi et al. (2011).

A hybrid algorithm, PSO with Dynamic Inertia Weight and Genetic Algorithm for classification rule mining was proposed by Uma et al. (2012). The dynamic inertia weight helps the algorithm to find global optima and to overcome the problem of convergence at local optima, and GA performs a global search over the entire search space with faster convergence speed.

2.5 MEMETIC PARTICLE SWARM OPTIMIZATION

There are several works in literature which have applied evolutionary algorithms for mining ARs, but the idea of using Memetic Algorithm (MA) for obtaining high quality ARs is seen to be much less investigated and reported. Memetic algorithm applies local search techniques to newly created offspring to quickly find individuals with high fitness, and discovers near optimal regions of the search space. There are several advantages for incorporating local search into evolutionary algorithms. One of them is to improve their performance by increasing the convergence rate and maintaining solutions diversity.

Memetic PSO (MPSO) is a hybrid algorithm that combines PSO with local search techniques. MPSO consists of two main components: a global one that is responsible for the global search of the search space, and a local one, which performs more refined search around potential solutions of the problem on hand.

MPSO algorithm called MeSwarm was proposed for tackling the overshooting problem in the motion behavior of PSO. While the overshooting problem occurs, particles may be led to wrong or opposite directions against the direction to the global optimum. As a result, MeSwarm integrates the standard PSO with the Solis and Wets local search strategy to avoid the overshooting problem and it is based on the probability of success to efficiently generate a new candidate solution around the current particle (Liu et al. 2005).

Petalas et al. (2007) introduced MPSO algorithm which consisted of two main components: a global one that is responsible for the global search of the search space, and a local one, which performs more refined search around potential solutions of the problem on hand. They employed a stochastic iterative Local Search (LS) technique in their MA, called Random Walk with Direction Exploitation (RWDE), where, a sequence of approximations of the optimizer were generated by assuming a random vector as a search velocity.

Motivated by the compensatory property of EA and PSO, where, the latter can enhance solutions generated from the evolutionary operations by exploiting their individual memory and social knowledge of the swarm, Chiam et al. (2009), proposed the implementation of PSO as a local optimizer for fine tuning in evolutionary search. Hongfeng et al. (2010) investigated a PSO based memetic algorithm that hybridizes PSO with a local search technique for dynamic environments. Within the framework of the proposed algorithm, a local version of PSO with a ring-shape topology structure was used as the global search operator, and a fuzzy cognition local search method was proposed, as the local search technique.

A hybrid multiobjective EA combining two heuristic optimization techniques was proposed by Mousa et al. (2012). This approach integrates the merits of both GA and PSO. Hongfeng et al. (2010) proposed a new memetic PSO algorithm that combines PSO and a LS method for optimization problems with many optima. In the proposed MPSO algorithm, a special PSO neighborhood structure, where the particles are located on a ring-shaped topology and adaptively form different species based on their indices in the population, was proposed for locating multiple optima, and an adaptive LS operator, which employs two different LS methods in an adaptive cooperation fashion, was used for enhancing the exploitation capacity of the proposed algorithm.

The literature so far reviewed are summarized in Table 2.1 for analysis on the shortcomings of existing methods and provide a roadmap to the direction to be taken to achieve the objective of the work to be carried out.

Table 2.1 Analysis on Existing Literature

S.No	Category	Method	Features Attempted	Applied for ARM	Multi-objective
1	Traditional Methods	AIS	Multiple Scan of database to identify the patterns	Yes	No
		Apriori Based	Attempts in reducing the number of scans and reduced to n scan where n is number of attributes in rule	Yes	No
		FP Tree based	Tree structure used to store patterns for reducing scans	Yes	No
		Other methods	Attempts slight modification as introducing hash, id counts to make ARM simple	Yes	No
2	Stochastic search methods	GA	Changes made in GA parameter ,parameter tuning and Elitism	Yes	Yes
		PSO	Changes like chaos, Quantum behavior, weighted PSO and changes in velocity updation	Yes	No
3	Hybrid	GA + PSO	GA with modifications in operators and during displacement GA applied within PSO	Hardly yes	No
4	Memetic	PSO + LS	Local searches like RWDE, Wets LS, Fuzzy Cognitive LS and adaptive LS were tried	No	No

2.6 CONCLUDING REMARKS

The state-of-the-art methods related to the proposed research work have been briefly described and discussed in this chapter. From the following inferences are drawn:

(i) The traditional ARM methods are complex methods requiring too many scans of the database and generate more number of rules.

(ii) GA and PSO perform better for data mining applications.

(iii) Hybrid and memetic methods of GA and PSO assure global optimization avoiding premature convergence.

CHAPTER 3

CONCEPTS AND PROBLEM SETTINGS

3.1 GENERAL

In this chapter an overview of the fundamental concepts namely stochastic search methods, evolutionary computing methodologies, GA and PSO that are related to this research work are presented. The datasets and the measures used for evaluating the proposed methods in this research, are also discussed.

3.2 STOCHASTIC SEARCH METHODS

An optimization problem consists of a pair (X; H), where X is a set, called the state space, $H: X \rightarrow R$ is a real-valued function, called the objective function or energy function. A global maximum solution to the optimization problem is an element (state) $x \in X$ for which $H(x) \geq H(y)$, for every $y \in X$.

In order to search through X for a global solution, one must first define a neighborhood function N(x). A neighborhood function maps each $x \in X$ to a subset of X. Thus we have $N(x) \subset X$. Now given such a neighborhood function, a path of length n in the state space is a sequence of elements $L = x_0, x_1, \ldots, x_n$ for which $x_i \in N(x_{i-1})$, for all $i = 1,2,\ldots,n$.

Usually search through the state space consists of traversing one or more paths in the state space until a solution is found, or until some other stopping condition has been met. This implies that there has to be some method for computing the next element in the path. If the method follows a deterministic rule, then, that the search is deterministic. Similarly, if the method is stochastic in nature, then that the search is stochastic.

Many engineering problems encountered in real-world applications require solving high-dimensional optimization tasks in a reasonable computational time. Often such problems cannot be solved in a deterministic manner, and hence there is a need for alternative, stochastic approximation approaches. Stochastic search methods are optimization methods that generate and use random variables. The study of stochastic optimization problems dates back to the 1950's and the work of Dantzig (1955) attempts to model uncertainty in the data by assuming that (part of) the input is specified in terms of a probability distribution, instead of deterministic data given in advance. Stochastic search plays a significant role in the analysis, design, and operation of modern systems.

Methods for stochastic optimization provide a means of coping with inherent system noise and coping with models or systems that are highly nonlinear, high dimensional, or otherwise inappropriate for classical deterministic methods of optimization. Stochastic search optimization methods include:

- Simulated Annealing
- Reactive Search Optimization
- Random Search
- Stochastic Tunneling
- Stochastic Hill Climbing
- Swarm Intelligence

- Evolutionary Algorithms
    - Genetic Algorithm
    - Evolution strategies

Many modern data mining packages include methods such as simulated annealing, GAs and PSO as tools for extracting patterns in data.

3.3 OVERVIEW OF EVOLUTIONARY COMPUTING

The collection of techniques for optimization purpose in use today is derived from three research paradigms that arose independently in the late 1950s and early 1960s. GAs (and a subsequent variant called genetic programming) exploit the inheritance of genetic traits, emphasizing the importance of recombination and the adaptive properties of populations. Evolutionary strategies focus on the inheritance of behavior traits rather than genetic traits, and emphasize the key role that mutation plays in evolutionary search. Evolutionary programming also emphasizes the inheritance of behavior traits, at the level of species as an alternative to the level of individuals. The distinction between these paradigms have blurred in recent years, and the more inclusive term, evolutionary computation, is often used to emphasize that the various algorithms share many features.

The power of Evolutionary Computation (EC) algorithms is derived from a very simple heuristic assumption: the best solutions to a problem will be found in regions of the search space containing relatively high proportions of good solutions and these key regions can be identified by strategic and robust sampling of the space. In practice, these algorithms have proven strongly capable of finding improvements in complex, nonlinear search spaces. They have been applied to a wide variety of search and optimization problems.

The idea behind the EC is quite straightforward – take a population of points from some search space, assign them numbers that reflect their probability to survive the selection process, and perform a randomized selection. The selected points undergo a randomized variation, yielding a new population of points, and the process is iterated many times. The layout of the algorithm for evolutionary computation is given below.

Step 1. Create an initial random population.

Step 2. Evaluate the fitness values of the population.

Step 3. Perform the following steps on the current generation:

- Select an individual in the population based on a selection scheme
- Adapt the selected individuals
- Evaluate the fitness value of the adapted individuals
- Select adapted individuals for the next generation according to a selection scheme

Step 4. If the termination criterion is fulfilled, then output the final population. Otherwise, set the current generation as nest generation and go to step 3.

This algorithm is more complex than it seems at first glance. There are five important decisions that factor in designing the algorithm (Ashlock 2006).

- ➢ The data structure to be used
- ➢ Fitness function to be designed
- ➢ Adapting of individuals for evolution process

- Selecting parents from the population and inserting children in population
- Termination condition to end the algorithm

Representation: The structure of a solution varies from problem to problem and the solution of a problem can be represented in a number of ways. The representation of individuals in search methods differs. The representation scheme depends both on the application problem and the chosen search method. For any method choosing the right representation for problems makes it simpler to solve (DeJong 1997).

Initial Population: Evolutionary computing methods are stochastic population based search algorithms. Each methodology maintains a population of candidate solutions. The first step in applying an EC to solve a problem is to generate an initial population. The standard way of generating an initial population is to assign a random value from the allowed domain. The initial population should ensure uniform representation of the entire search space. The size of the initial population has consequences in terms of computational complexity and exploration abilities (Engelbrecht 2002).

Fitness Function: In the Darwinian model of evolution, individuals with best characteristics have the best chance to survive. To determine the ability of an individual to survive, a mathematical function called fitness function is used to quantify the goodness of the represented solution. Fitness function has an important role in evolutionary computing because the evolutionary operators make use of the fitness values of individual for evolution.

Given a particular individual, the fitness function returns a single numerical "fitness," or "figure of merit," which is supposed to be proportional to the "utility" or "ability" of the individual which the chromosome represents. For many problems, particularly function optimization, the fitness function should simply measure the value of the function.

Selection: Selection is one of the main operators used in evolutionary computation. The primary objective of the selection operator is to emphasize better solutions in a population. This operator does not create a new population, instead, it selects relatively a good solution from a population for the adaption into future generations (DeJong 1997). The selection is usually accomplished using the fitness values of solutions. The main idea is that a solution having a better fitness must have a higher probability of selection.

Adapting Selected Individuals: This is a major activity of EC. The selected members of the population undergo transformations to form new solutions. The operators for the transformation vary from one method of EC to other. This step differentiates the various methods in EC. The Evolutionary algorithms use genetic operators as crossover and mutation for transformation purposes. The Swarm intelligent algorithms such as PSO and ACO follow different patterns for transforming the solutions from one population to the next.

Stopping Criteria: The evolution of population is done in evolutionary computing until a stopping criterion is achieved. Common stopping conditions are:

- On reaching a fixed number of generations or
- On reaching the allocated budget (computation time/money) or
- On reaching the highest ranking solution's fitness or has reached a plateau such that successive iterations no longer produce better results or
- By manual inspection or
- Combinations of the above.

GA and PSO could start with initial population defined from datasets, instead of assuming them randomly. This makes GA and PSO more suitable for ARM where the aim is to find interesting correlations between attributes from the datasets defined.

3.3.1 Genetic Algorithm

The grand idea behind GAs is the survival of the fittest, essentially evolution. In real life, a species of animal or plant, etc. will begin to die-off over time if a more dominant species is competing with it for the same resources. The weakest specimens of the species start to die off first, while the stronger ones survive and continue to breed. Over time, the strong mate with the strong, produce even stronger or fitter, specimens, and the species adapts to the challenges by evolving a population better able to compete. GA try to mimic this action of nature. They take solutions to problems and, by controlling the death and breeding of the specimens, influence the population to become collectively and individually fitter (Man et al. 1999). The block diagram of GA is shown in Figure 3.1.

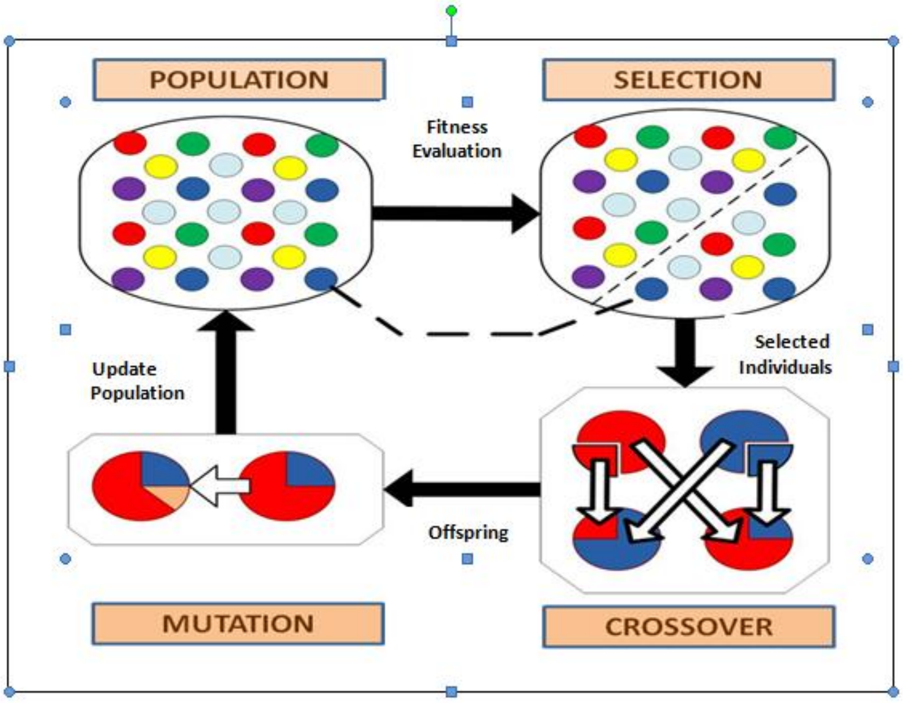


Figure 3.1 Block Diagram of Genetic Algorithm

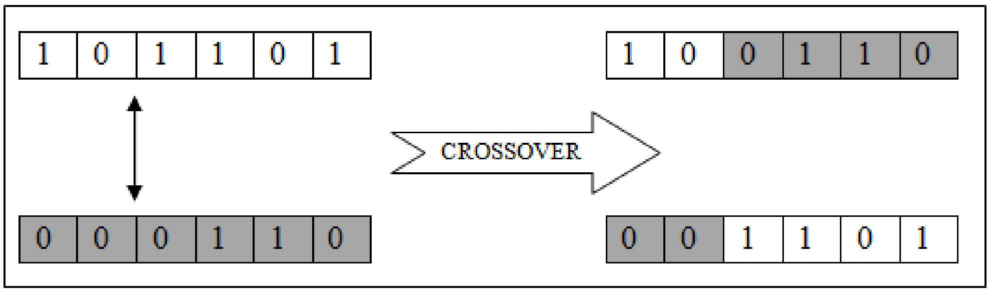
Every GA has three basic elements. There must first be a group of solutions to a problem, called the population. Secondly, there must be a way of measuring solutions against each other and determining which solution is best, or most fit. This is called a fitness test or fitness function. And lastly, there must be a way of combining together different solutions from any one population to produce new solutions, called breeding or mating. By continuously breeding the fit specimens of each population with each other the hope is to produce even more fit specimens (Kim et al. 1999). The pseudocode of the simple GA is given in Figure 3.2.

```
Step 1.  Initialization : Generate initial population at random or with prior
         knowledge
Step 2.  Fitness Evaluation : Evaluate the fitness for all individuals in the
         current population
Step 3.  Selection : Select a set of promising candidates from the current
         population
Step 4.  Crossover: Apply crossover to the mating pool for generating a set
         of offspring
Step 5.  Mutation : Apply mutation to the offspring set for obtaining its
         perturbed set
Step 6.  Replacement : Replace the current population with the set of
         offspring
Step 7.  Termination : If the termination criteria are not met, go to Step 2
```

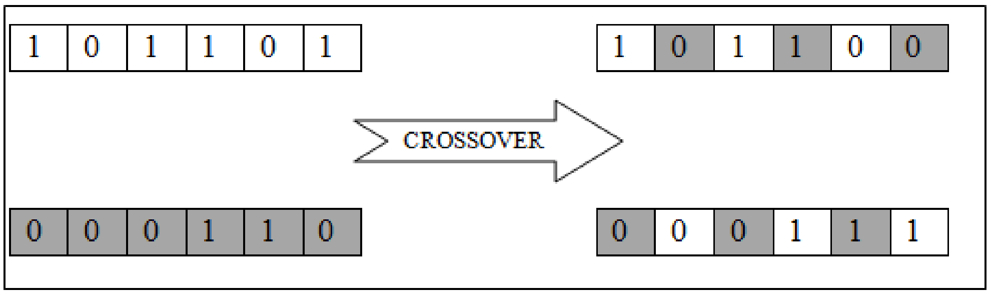
Figure 3.2 Algorithm for Simple GA

Selection chooses the individuals with higher fitness as parents for the next generation. In other words, selection operator is intended to improve average quality of the population by giving superior individuals a better chance to get copied into the next generation. There is a *selection pressure* that characterizes the selection schemes. It is defined as the ratio of the probability of selection of the best individual in the population to that of an average individual (Back, 1994; M¨uhlenbein and Voosen 1993). There are two basic types of selection scheme in common usage: *proportionate* and *ordinal* selection. Proportionate selection picks out individuals based on their fitness values relative to the fitness of the other individuals in the population. Examples of such a selection type include roulette-wheel selection (Goldberg, 1989; Holland, 1975), stochastic remainder selection (Booker, 1982), and stochastic universal selection (Baker 1985). Ordinal selection selects individuals not based upon their fitness, but upon their rank within the population. The individuals are ranked according to their fitness values. Tournament selection (Brinde 1981), (μ, λ) selection (Schwefel 1977), linear ranking selection (Baker 1985), and truncation selection (M¨uhlenbein and Voosen 1993), are included in the ordinal selection type.

Crossover exchanges and combines partial solutions from two or more parental individuals according to a crossover probability, p_c, in order to create offspring. That is, the crossover operator exploits the current solution with a view to find better ones. Two popular crossover operators, among many variants, are: *one-point* and *uniform* crossover. One point crossover (Goldberg 1989; Holland 1975) randomly chooses a crossover point (i.e., crossing site) in the two individuals and then exchanges all the genes behind the crossover point (Figure. 3.3a). Uniform crossover (Syswerda 1989) exchanges each gene with probability 0.5 (Figure. 3.3b), hence achieving the maximum allele-wise mixing rate.



a) One point crossover



b) Uniform Crossover

Figure 3.3 Crossover Operators

3.3.2 Particle Swarm Optimization

PSO is one of the evolutionary computation methods, based on swarm intelligence. PSO was first designed to simulate birds seeking food. Birds would find food through social cooperation with other birds within a neighborhood. Assume that there is a group of birds searching food in an area and if there are small pieces of food near the food centre, then bigger the food becomes. No bird knows where the food center is. So the best strategy to find the food is to follow the bird which has found the biggest pieces. PSO just simulates this scenario and uses it to solve optimization problems.

In PSO, each single solution is a "bird" in the search space. We call it a "particle". All particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. This value is called pBest. Another best value that is tracked by a particle is the best value obtained so far by any particle in the problem space. This best value is the global best and is called gBest. After finding the two best values, each particle updates its corresponding velocity and position. The pseudocode for the PSO is given in Figure 3.4.

```
Step 1.   Initialize particles with random positions and velocities.
Step 2.   Set particles’ pBest to their current positions.
Step 3.   Calculate particles’ fitness and set gBest.
Step 4.   For T generations do
              Update particles’ velocities.
              Update particles’ positions.
              Recalculate particles’ fitness.
              Update particles’ pBest and gBest.
          end For
```

Figure 3.4 Algorithm for Simple PSO

Each particle p, at some iteration t, has a position x(t), and a displacement velocity v(t). The particles best (pBest) and global best (gBest) positions are stored in the associated memory. The velocity and position are updated using equations 3.1 and 3.2 respectively.

$$v_i^{new} = v_i^{old} + c_1 rand()(pBest - x_i) + c_2 rand()(gBest - x_i) \qquad (3.1)$$

$$x_i^{new} = x_i^{old} + v_i^{new} \qquad (3.2)$$

where,

v_i^{old}	is particle velocity of the ith particle before updation
v_i^{new}	is particle velocity of the ith particle after updation
x_i	is the ith, or current particle
i	is the particle's number
d	is the dimension of search space
rand ( )	is a random number in (0, 1)
c_1	is the individual factor
c_2	is the societal factor
pBest	is the particle best
gBest	is the global best

The particle velocity on each dimension is clamped to a maximum velocity V_{max}. If the sum of accelerations cause the velocity in that dimension to exceed V_{max}, which is a parameter specified by the user, then the velocity in that dimension is limited to V_{max}. This method is called V_{max} method (Song and Gu, 2004). The diagrammatic representation of the position update of particle in PSO following the velocity and position update is shown in Figure 3.5.

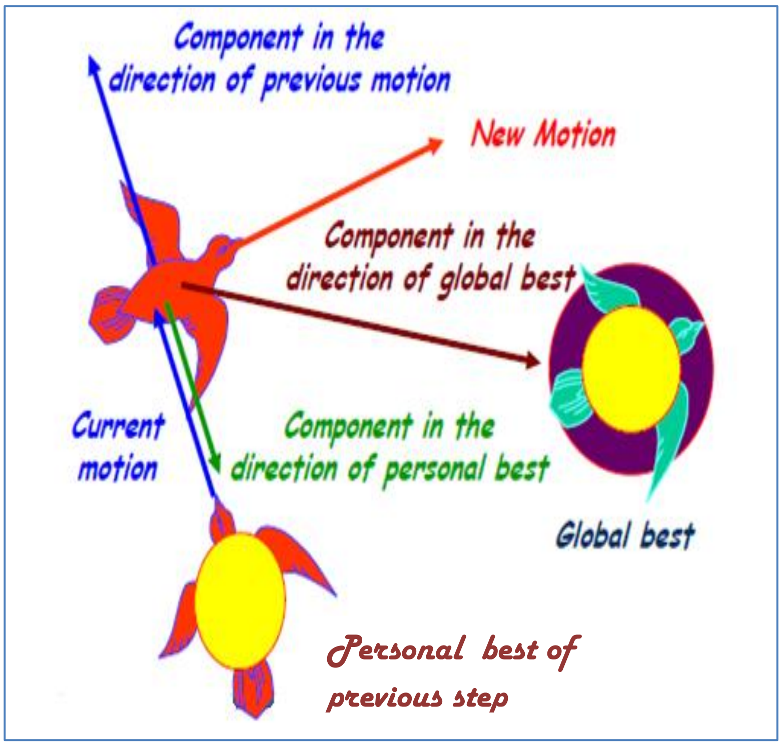

Component in the direction of previous motion
New Motion
Component in the direction of global best
Current motion
Component in the direction of personal best
Global best
Personal best of previous step

Figure 3.5 Displacement of a PSO Particle

With the position and velocity updates, the particles shift their location with successive generations. The positions in which the particle lies can be categorized into one of the four states namely:

- Exploration
- Exploitation
- Convergence
- Jumping Out

Exploration is defined as locating the global optima effectively. Exploitation is performing the local search in the global optima. The movement of all particles towards the global optima is convergence and Jumping out is the process of moving away from the global optima.

The implement PSO effectively balance between exploration and exploitation have to be maintained properly, to avoid jumping out and converge correctly.

3.4 DATASETS USED IN THIS STUDY

All our experiments were conducted using benchmarked data sets from different fields. Table 3.1 gives the description of the datasets that have been used. These datasets are sourced from the University of California, Irvine (UCI) database (Blake and Merz, 1998). The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The datasets with varying number of instances and features have been selected to validate the performance of proposed methodologies. The total number of instances in each of the dataset, the number of features along with the number of classes is presented in the dataset. The details of all UCI datasets that have been used in this study are presented in Annexure I

Table 3.1 Description of the Datasets Used

Name of the Dataset	No. of Attributes	No. of Instances	Attribute characteristics
Lenses	4	24	Categorical
Car Evaluation	6	1728	Categorical, Integer
Haberman’s Survival	4	310	Integer
Post-operative Patient Care	9	87	Categorical, Integer
Zoo	18	101	Categorical, Binary, Integer
Iris	4	150	Categorical, Integer
Nursery	8	12960	Categorical, Integer
Tic Tac Toe	9	958	Categorical
Wisconsin Breast Cancer	10	699	Categorical, Integer
Adult	14	48842	Categorical, Integer

3.5 PERFORMANCE EVALUATION

ARs generated are usually large in numbers and have redundant information. Much work on ARM has been focused on efficiency, effectiveness and redundancy. Focus is also needed on the quality of rules mined. In this work ARM using GA and PSO is treated as a multiobjective problem, where rules are evaluated quantitatively and qualitatively based on the following five measures.

- Predictive Accuracy (PA)
- Laplace
- Conviction
- Leverage
- Lift

i. **Predictive Accuracy** (also named Confidence) measures the effectiveness of the rules mined. The mined rules must have high PA.

$$Predictive\ accuracy = \frac{support(x \cup y)}{support(x)} \qquad (3.3)$$

Where, x is the antecedent and y the consequent.

ii. **Laplace** is a confidence estimator that takes support into account, becoming more pessimistic as the support of x decreases (Good, 1965; Deepa et. al, 2003). It ranges between [0, 1] and is defined as

$$Laplace(x \rightarrow y) = \frac{support(x \cup y) + 1}{support(x) + 2} \qquad (3.4)$$

iii. **Conviction** is sensitive to rule direction and attempts to measure the degree of implication of a rule (Brin et al. 1997b). It ranges between [0.5, ∞]. Values far from 1 indicate interesting rules.

$$Conviction(x \rightarrow y) = \frac{1 - support(y)}{1 - confidence(x \rightarrow y)} \tag{3.5}$$

iv. **Leverage** also known as Piatetski-Shapiro measure (Piatetsky-Shapiro 1991). It measures how much more counting is obtained by the rule from the co-occurrence of the antecedent and consequent from the union. It ranges between [−0.25, 0.25] and is defined as:

$$Leverage(x \rightarrow y) = support(x \cup y) - support(x) \times support(y) \tag{3.6}$$

v. **Lift** (also called interest by Brin et al. 1997a) measures how far from independence are antecedent and the consequent. It ranges within [0, +∞]. A lift value of 1 indicates that the items are co-occurring in the database as expected under independence. Values greater than one indicate that the items are associated.

$$Lift(x \rightarrow y) = \frac{Confidence(x \rightarrow y)}{support(y)} \tag{3.7}$$

3.6 SUMMARY

This chapter provides the necessary background information for the proposed research. Introduction to stochastic search methods and Evolutionary Computing are first explained. This is followed by the discussion on the two population-based search methods GA and PSO that have been used in this research. Finally the datasets used and evaluation measures are presented.

CHAPTER 4

ASSOCIATION RULE MINING USING GENETIC ALGORITHM

4.1 GENERAL

This chapter discusses the need and motivation for using GA to mine ARs and focuses on the modifications that has been suggested and implemented in GA for mining ARs. ARM using GA and the effective setting of parameters in GA for ARM have been explained in detail. The concept of Elitism is introduced in GA for ARM purpose. The qualitative and quantitative measures of the rules mined with the datasets taken up for the study of Elitism in GA is presented and the adaptive parameter setting of the mutation probability and its effect on the performance of ARM is also analyzed. The results obtained are compared with the results from simple GA and the salient inferences are drawn.

4.2 NEED AND MOTIVATION

Many algorithms for generating ARs were developed over time. Some of the well known algorithms are Apriori, Eclat and FP-Growth tree. The existing algorithms traverse the database many times and hence the I/O overhead and computational complexity becomes very high and cannot meet the requirements of large-scale database mining. At present, GA-based data mining methods have yielded some progress, and based on GA classification system has also yielded good results.

The original motivation for the GA approach was a biological analogy. In the selective breeding of plants or animals, offspring are sought to have certain desirable characteristics—characteristics that are determined at the genetic level by the way the parents' chromosomes combine. In the case of GAs, a *population* of strings is used, and these strings are often referred in the GA literature as *chromosomes.* The recombination of strings is carried out using simple analogies of genetic *crossover* and *mutation*, and the search is guided by the results of evaluating the objective function for each string in the population. Based on this evaluation, strings that have higher *fitness* (i.e., representing better solutions) can be identified, and these are given more opportunity to breed.

4.3 MINING AR USING GA

This section deals with the ARM problem as a multi-objective problem using multi-objective evolutionary algorithms with emphasis on GA. The main motivation for using GA is that they perform a global search and cope better with attribute interaction than the greedy rule induction algorithms often used in ARM tasks.

4.3.1 Experimental Setup

The pseudo code for GA as given in Figure 3.2 is used for mining ARM. It starts with a population of individuals randomly generated according to some probability distribution, usually uniform, and updates this population in steps called generations. In each generation, multiple individuals are randomly selected from the current population based on some application of fitness, bred using crossover, and modified through mutation to form a new population. The number of generations is maintained as 50.

Genetic Operators

GA maintains a population of n chromosomes (solutions) with associated fitness values. Parents are selected to mate, on the basis of their fitness, producing offspring via a reproductive plan (mutation and crossover). Consequently highly fit solutions are given more opportunities to reproduce (selected for next generation), so that offspring inherit characteristics from each parent. As parents mate and produce offspring, room must be made for the new arrivals, since the population is kept at a static size (population size). In this way it is hoped that over successive generations better solutions will thrive while the least fit solutions die out. The representation scheme, population size, crossover rate, mutation rate, and fitness function and selection operator are the GA operators, and are discussed below.

Encoding chromosomes: The Michigan approach of binary encoding (Ghosh and Nath 2004) is adopted for representing rules. Each attribute is represented in binary form, thereby creating array representation, with each entry in an array representing a rule (Avendano and Martin 2010; Ghou and Zou 2010).

Initialization: Based on the population size fixed, the initial population is selected randomly.

Fitness function: The fitness of an individual in a GA is the value of an objective function for its phenotype

This work adopts minimum support and minimum confidence for filtering rules. Then, correlative degree is confirmed in rules which satisfy minimum support degree and minimum confidence degree. After support degree and confidence degree are synthetically taken into account, fitness function is defined as follows.

$$Fitness(x) = R_s \frac{support(x)}{support_{min}} + R_C \frac{confidence(x)}{confidence_{min}} \tag{4.1}$$

In the above formula, $R_s + R_c = 1$ ($R_s \geq 0$, $R_c \geq 0$); $\text{support}_{\text{min}}$, and $\text{confidence}_{\text{min}}$ are respective values of minimum support and minimum confidence. By all appearances if the $\text{support}_{\text{min}}$ and $\text{confidence}_{\text{min}}$ are set to higher values, then, the value of fitness function is also found to be high.

Selection operator: During each successive generation, a proportion of the existing population is selected to breed a new generation. Figure 4.1 depicts the Roulette wheel selection method used for ARM using GA in this work.

Reproduction: The reproduction mechanism involves rule selection and the application of the crossover operators. The rule selection method used by this algorithm follows the "universal suffrage" approach proposed in (Giordana et al. 1997). With this approach each AR is represented by a single individual. The actual reproduction takes place by performing crossover and mutation operations on the new individuals.

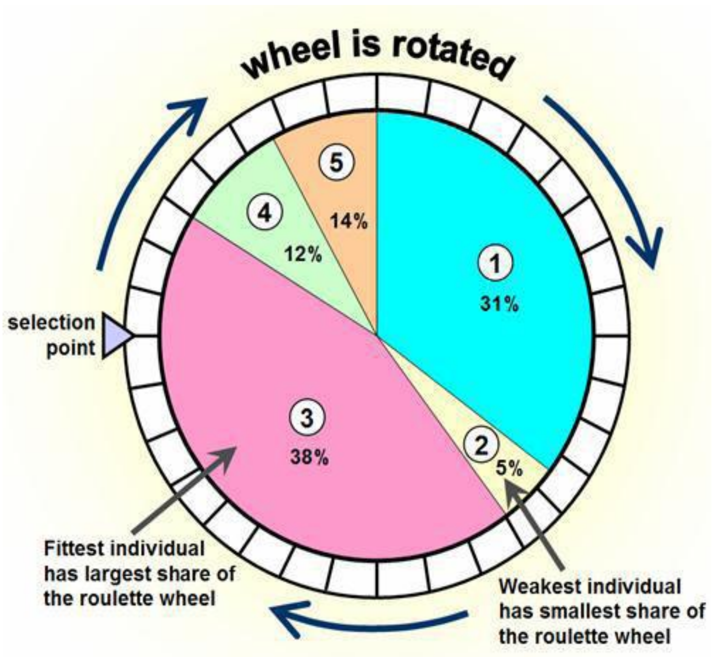

wheel is rotated
5
14%
4
12%
1
31%
selection point
3
38%
2
5%
Fittest individual has largest share of the roulette wheel
Weakest individual has smallest share of the roulette wheel

Figure 4.1 Roulette Wheel Selection Methodology

Crossover Operator: Crossover selects genes from parent chromosomes and creates a new offspring. In this work, single point crossover with crossover at attribute level is performed. So the optimal value of the crossover rate is independent on the number of attributes in the datasets.

Mutation Operator: Mutation changes randomly the new offspring. For binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation provides a small amount of random search, and ensures that no point in the search has a zero probability of being examined. The mutation rate when set to zero retains the original values of the offspring.

4.3.2 Results and Discussion

Five datasets from UCI Machine Learning Repository (Blake and Merz 1998) have been considered for experimentation. The experimental setup is made as given in Table 4.1.

Table 4.1 GA Parameters for ARM

Parameter Name	Value
Population Size	21
Crossover rate	0.5
Mutation rate	0.0
Selection Method	Roulette Wheel Selection
Minimum Support	0.2
Minimum Confidence	0.8

The crossover rate is set as 0.5. This generates two offspring which exchanges 50% of bits from each parent. The mutation rate is made as 0.0 to retain the original values of the offspring generated by crossover operation, to study the is significance of crossover operation.

Quantitative measures for analyzing the performance of the ARs are PA and the number of rules generated. The PA of the ARs mined with the above discussed GA method given in Table 4.2. The PA for the mined rules is optimal for all datasets considered for the analysis. The PA of the ARs mined with GA is enhanced upto 2 % when compared to the accuracy by methods given in literature.

The number of rule generated by GA for the PA, 0.05 percentage less than the maximum PA achieved for each dataset is given in Table 4.3.

Table 4.2 Predictive Accuracy for ARM with GA

Dataset	PA attained (%)	PA by existing methods (%)
Lenses	85	-
Car Evaluation	81	-
Haberman's Survival	87	-
Post operative Patient	74	-
Zoo	81	-
Nursery	90	89 (Dehuri and Mall, 2006)
Adult	88	86 (Dehuri and Mall, 2006)
Wisconsin Breast Cancer	97	96.14 (Chen and Hsu, 2006)

Table 4.3 Number of Rules Generated by GA

Dataset	No. of Rules	No. of Rules mined by existing methods
Lenses	10	-
Car Evaluation	178	11 (Avendano and Gutierrez, 2010)
Haberman's Survival	46	-
Post operative Patient	22	-
Zoo	15	Around 325 (Ramesh and Iyakutti 2011)
Nursery	22	4 (Hamid et al. 2011)
Adult	117	Around 350 (Ramesh and Iyakutti 2011)
Wisconsin Breast Cancer	83	-

The number of rules generated by any methodology should be optimum. If the number of rules generated is too large, then pruning or filtering of the rules should be done. If too few rules are generated, then the chance of missing interesting patterns may also occur. Many existing methods in literature identify the number of rules based on support values and hence number of rules generated is more. In the proposed method association rule is generated based on PA which takes into account the support value and confidence value we claim that the number of rules generated is optimum as shown in Table 4.3.

The qualitative measures indicate the importance or interestingness of the ARs mined. The Conviction, Laplace, Leverage and Lift measures for analyzing the interestingness of the rules generated with GA are given in Table 4.4

Table 4.4 Qualitative Measures of AR mined by GA

Dataset	Conviction	Laplace	Leverage	Lift
Lenses	Infinity	0.52	0.002	1
Car Evaluation	Infinity	0.625	0	1
Haberman's Survival	Infinity	0.52	0.0625	2
Postoperative Patient	Infinity	0.5	0.016	6.314
Zoo	Infinity	0.644	0.046	1.06
Nursery	Infinity	0.601	0.028	1
Adult	Infinity	0.533	0.052	1
Wisconsin Breast Cancer	Infinity	0.544	0.032	1.05

The values of the defined measures for all the five datasets are within the specified range as denoted in section 3.5. Thus the rules generated by the GA methodology are of significance.

The qualitative and quantitative measures of the ARs mined by GA presented in this section indicate that the chosen GA methodology is more opt for the application chosen.

4.4 EFFECTIVE PARAMETER SETTING OF GA FOR ARM

There have been several attempts for mining ARs using GA. Robert Cattral et al. (1999) described the evolution of hierarchy of rule using GA with chromosomes of varying length and macro mutations. The initial population was seeded instead of random selection. Manish Saggar et al. (2004) proposed an algorithm with binary encoding and the fitness function was generated based on confusion matrix. The individuals were represented using the Michigan's Approach. Roulette Wheel selection was implemented by first normalizing the values of all candidates.

Shi and Lei (2008) proposed GA where the fitness function is based on PA, comprehensibility and interestingness factor. The selection method was based on elitist recombination. In Ma and Li (2009) the encoding of data was done with gene string structure, where the complexity concepts were mapped to form linear symbols. The fitness function was considered as a measure of the overall performance of the process and not the individual rule measure, when the bit strings were interpreted as a complex process.

Adaptive exchange probability (P_c) and mutation probability (P_m) was introduced in this work. Guo and Zhou (2009) applied the method of adaptive mutation rate to avoid excessive variation causing non-convergence, or a local optimal solution. A sort of individual- based selection method is applied to the evolution in GA, in order to prevent the high-fitness individuals converging early by the rapid growth of the number of individual.

As the parameters of the GA and the fitness function are found to be the major area of interest in the above studies, the effects of the genetic parameters and the controlling variables of fitness function on different datasets are explored.

4.4.1 Methodology

The parameters of GA are the key components enabling the system to achieve good enough solution for possible terminating conditions. The parameters of GA and their role in solving problems are given in Table 4.5.

Table 4.5 Role of GA Parameters in Problem Solving

Parameter Name	Parameter Role
Population Size	Fixes the number of chromosomes and indirectly the crossover
Selection	Selection of the chromosomes for crossover
Mutation rate (p_m)	Maintains the genetic diversity of the population from one generation to other
Crossover rate (p_c)	Sharing Information between chromosomes
Minimum support Minimum confidence	Set by the user for fitness calculation

ARM using GA was performed following the steps given in the Figure 4.2. Effective ARM using GA by tuning the parameters manually is experimented and presented in this section. The population size, crossover rate, mutation rate are tuned manually to find the optimum values for better performance of ARM. The same fitness function described in equation 4.1 is adopted here. The user-defined values of minimum support and minimum confidence were also varied to analyze their role in mining ARs.

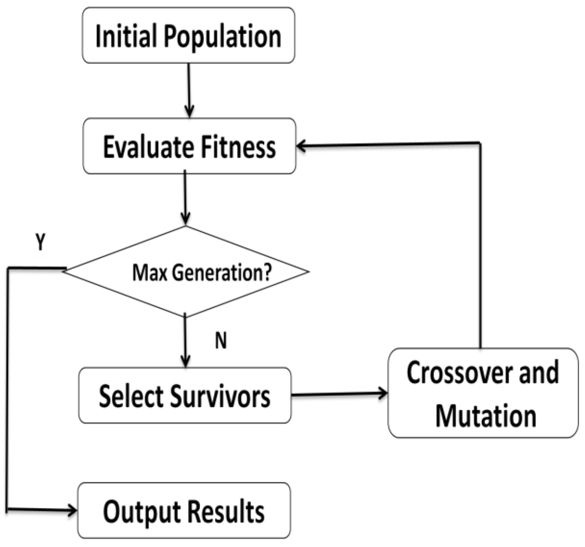


Figure 4.2 Flowchart of Simple GA

4.4.2 Experimental Results and Discussion

The objective of this study is to compare the accuracy achieved in datasets by varying the GA Parameters. The encoding of chromosome is binary encoding with fixed length. As the crossover is performed on attribute level, the mutation rate is set to zero so as to retain the original attribute values. The selection method used is tournament selection. The default parameters set for the experiment is listed in Table 4.6.

Table 4.6 Default GA Parameters

Parameter Name	Value
Population Size	Instances * 1.5
Crossover Probability	0.5
Mutation Probability	0.0
Selection Method	Tournament Selection
Minimum Support	0.2
Minimum Confidence	0.8

The parameters listed are tuned with varying values for each datasets and the results are recorded while mining ARs with GA. The population size is varied based on the number of instances in the datasets. For achieving optimal results the population size is set to three different values as given below:

- Equal to the number of instances in the dataset;
- 1.25 times the number of instances in the dataset and
- 1.5 times the number of instances in the dataset

The range of the population size is set so as to balance the number of rules generated without redundancy. The plotted results are shown in Figure 4.3.

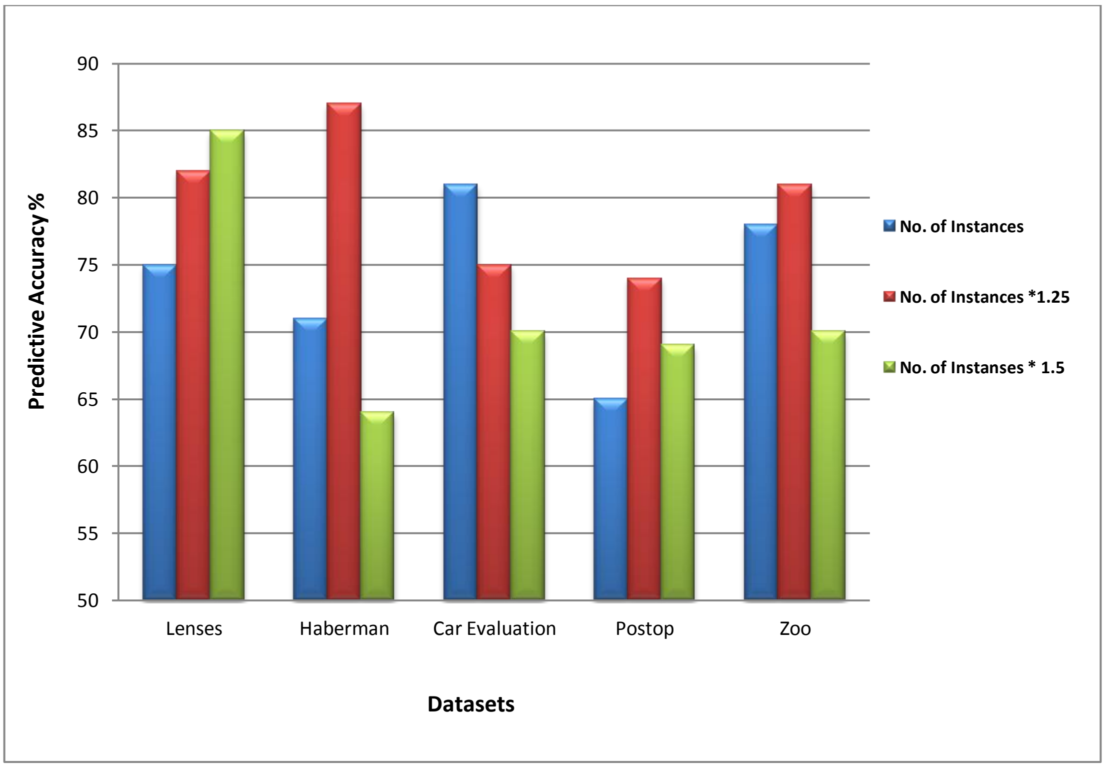


Figure 4.3 Population Size vs Accuracy for ARM with GA

The optimal value of the population size to be set depends on the dataset size. For the lenses dataset where the number of instances is small i.e. 24 only, the population size is set 1.5 times the number of instances, generated ARs with higher PA. For the datasets with moderate number of instances (Post operative care: 90, Zoo: 101 and Haberman's Survival: 306) the PA is found to be optimum when the population size is set to 1.25 times the number of instances in the dataset. For the car evaluation dataset with large number of instances (1728), when the population size is set to the actual number of instances, maximum PA is achieved.

The crossover operator in the GA helps in sharing information between chromosomes thereby maintaining the exploitation capability of the chromosomes. The crossover is performed at attribute level for ARM. As the minimum number of attributes in the dataset taken up for the study is 4 (Lenses and Haberman's survival), the crossover could be done only for 3 attributes at the maximum. Hence, the crossover probabilities are altered to three different values to analyze the role of this operator for ARM using GA.

The PA achieved by the rules mined with GA along with the generation number at which the PA is optimum is presented in the Table 4.7.

Table 4.7 Comparison based on Variation in Crossover Probability for ARM using GA

	Pc =0 .25		**Pc = 0.5**		**Pc = 0.75**	
Dataset	**Accuracy %**	**Generation Number**	**Accuracy %**	**Generation Number**	**Accuracy %**	**Generation Number**
Lenses	95	8	95	16	95	13
Haberman's Survival	69	77	71	83	70	80
Car Evaluation	80	80	81	83	81	85
Post Operative Patient	74	57	74	63	73	68
Zoo	81	90	80	88	81	85

The PA achieved for each dataset is almost same regardless of the crossover probability set. However, only the generation number at which the accuracy is achieved differs from dataset to dataset. The crossover rate as defined earlier shares information between chromosomes, thereby finding the global region effectively and hence controls the convergence rate of ARM.

The mutation rate helps in maintaining the diversity of the population. In this study, the mutation rate was set to zero thereby relying on the diversity of population generated by reproduction, using crossover alone. The selection operator enables the selection of the chromosomes for reproduction. The selection operator is not varied in this study and tournament selection is used for selection of chromosomes.

The fitness function employed for ARM has four parameters: R_S, R_C, $support_{min}$ and $confidence_{min}$. The R_S and R_C values are set less than 1 and hence their impact on fitness value is less when compared to the $support_{min}$ and $confidence_{min}$ values set by the user. These are the minimum values for the support and confidence expected from the ARs mined. To study the impact of these two values on the PA of the rules mined, different combination of $support_{min}$ and $confidence_{min}$ are analyzed and the values are recorded for four significant combinations alone in the chart.

The PA of the ARs achieved for the five datasets are plotted in Figure 4.4.

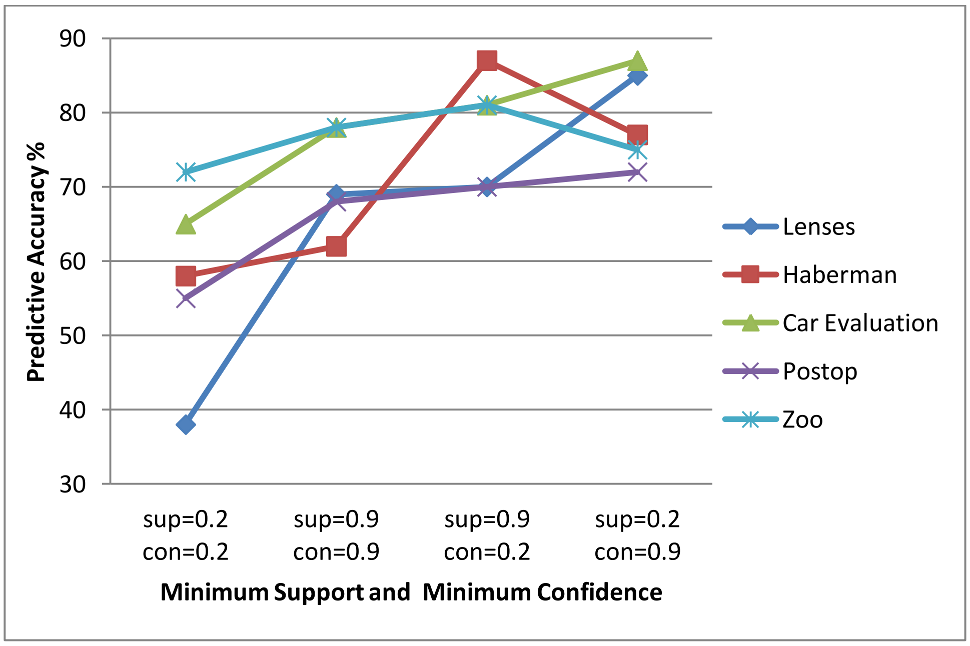


Figure 4.4 Minimum Support and Minimum Confidence vs Accuracy for ARM with GA

From the figure, it is clear that the variation in minimum support and minimum confidence brings greater changes in accuracy. When the values of minimum support and minimum confidence are set to the minimum, the accuracy is found to be low regardless of the size of the dataset. The same is noted when both the values are set to the maximum. Optimum accuracy is achieved when a tradeoff value between the minimum confidence and the minimum support is set.

4.5 GA WITH ELITISM FOR ARM

When GA is applied for mining ARs the PA achieved is optimum. The reproduction operators at times generate offspring well deviated from the actual population thereby misleading the global optima. Available literature (Rudolph 1999, Eckart et al. 2000) show that elitism can speed up the performance of GA significantly, and help to prevent the loss of good solutions, once they are found.

In general terms, elitism consists of archiving the “best” solutions generated during the search (e.g., Pareto optimal solutions). A secondary population, named *archive*, is used to store these high-quality solutions. The strategy used in updating the archive (elite population) relies on size, convergence, and diversity criteria. Elitism can be implemented in two ways:

- Passive Elitism Strategy
- Active Elitism Strategy

In passive elitism strategy, the archive is considered as a separate secondary population that has no impact on the search process (Figure 4.5). This will only guarantee that an algorithm has a monotonically non-degrading performance in terms of the approximated Pareto front.

In active elitism strategy the archived solutions are used to generate new solutions (Figure 4.5). Active elitism allows achieving faster and robust convergence toward the Pareto front for a better approximation of the Pareto front (Obayashi et. al, 1998; Zitzler and Thiele, 1999; Meunier et al. 2000).

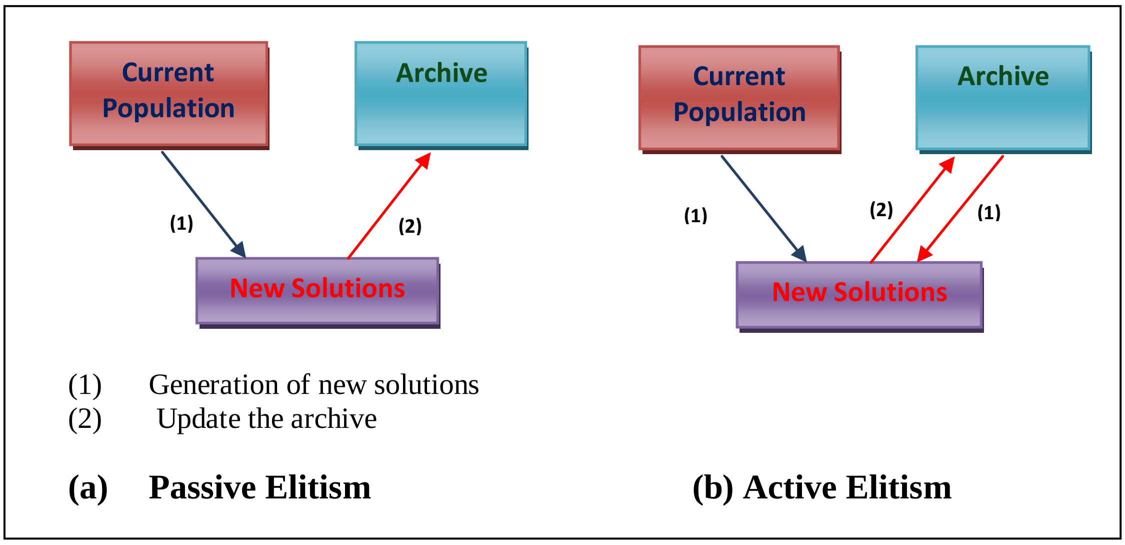


Figure 4.5 Elitism in Multiobjective Optimization

4.5.1 Elitism in GA

In ARM using GA there are chances of the selection of individuals with minimum fitness value (non-opt parent for reproduction) over chromosomes with better fitness values (opt parent for reproduction). To prevaricate elitism has been used in the search process of multiobjective optimization (active elitism).

When elitism is being carried out, care should be taken to prevent being trapped by a premature convergence, if a high-elitist pressure is applied to the generation of new solutions. In ARM using Elitist GA, the elitism was set as 10% of the population size. The active elitism allows the elite (archive) population for further reproduction. In other words, the elite population is fed to the selection process so as to perform the crossover and mutation operation, if selected. Thus, being trapped into local optima by premature convergence is avoided. The block diagram of the GA with Elitism is given in Figure 4.6.

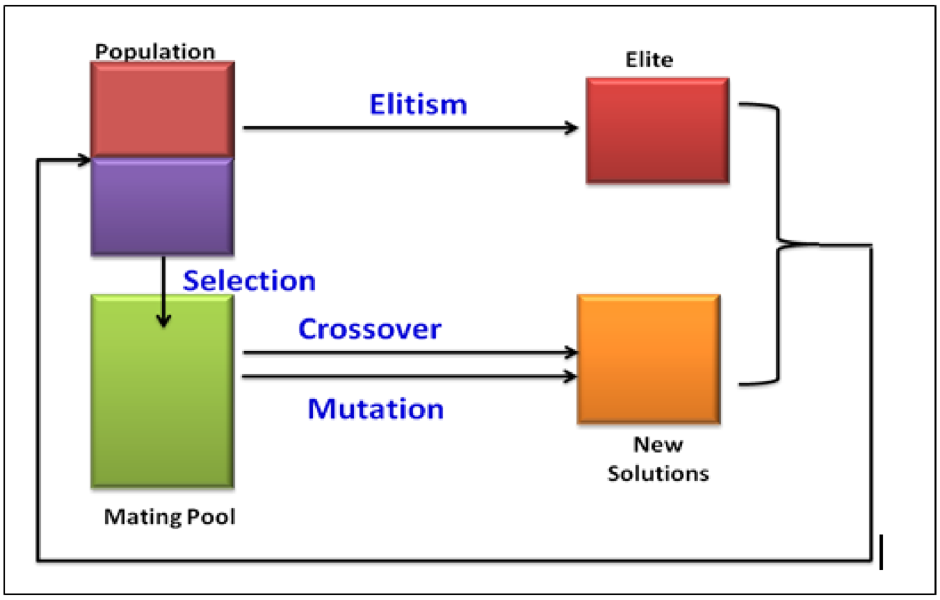


Figure 4.6 Block Diagram of Elitist GA for Association Rule Mining

An external population is maintained at every generation storing all non-dominated solutions discovered so far beginning from the initial population. This external population participates in all genetic operations. At each generation, a combined population with the external and the current population are first constructed. All nondominated solutions in the combined population are assigned a fitness based on the number of solutions they dominate. This assignment of fitness makes sure that the search is directed toward the nondominated solutions.

It was demonstrated in section 4.4 that the user-defined values of the $\text{support}_{\text{min}}$ and $\text{confidence}_{\text{min}}$ for the fitness function (Eqn. (4.1)) adopted affect the performance of the system. The actual trade-off between these two values is not easy to set by trial and error method (manual tuning).

For ARM with Elitism based GA a new fitness function to overcome the setback was defined. The fitness function for mining ARs with GA is given in Equation (4.2).

$$\text{Fitness}(x) = confidence(x) \times (\log(support(x)) \times length(x) + 1 \quad (4.2)$$

Where, *confidence(x)* is the confidence value of the rule, *support(x)* is the support for the antecedent and *length(x)* is the length of the rule.

The flowchart of the Elitist GA algorithm for ARM is given in the Figure 4.7.

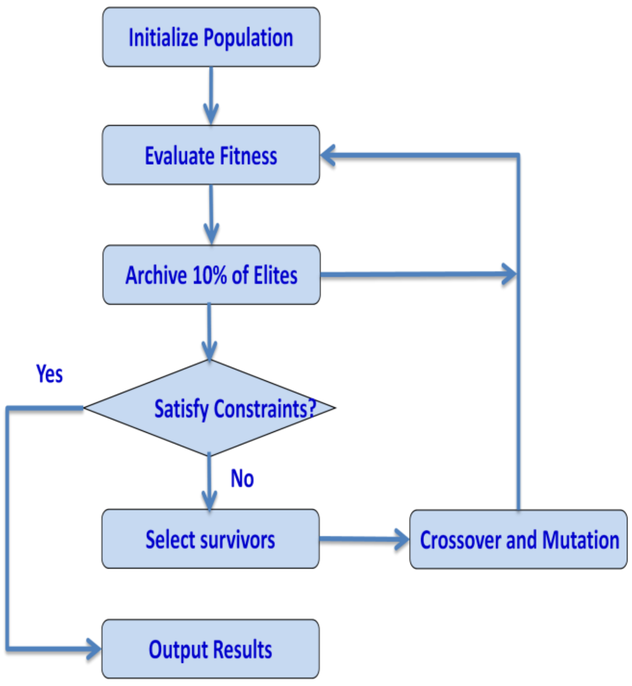


Figure 4.7 Flowchart of ARM with Elitist GA

4.5.2 Experimental Results and Discussion

ARM was performed using GA with Elitism on the five datasets. The parameter values set for the experiment are given in Table 4.8.

Table 4.8 Default Parameters for ARM with Elitist GA

Parameter Name	Value
Elitism	10% of population size
Crossover Probability	0.5
Mutation Probability	0.2
Selection Method	Roulette Wheel Selection
Population Size	Lenses : 20 Haberman's Survival : 300 Car Evaluation : 700 Postoperative Patient : 75 Zoo : 90

ARs were mined from the specified datasets and the PA achieved over 50 generations is presented in the Table 4.9.

Table 4.9 Predictive Accuracy for AR mined with Elitist GA

No. of Iterations	Lenses	Car Evaluation	Haberman's Survival	Postoperative Patient	Zoo
4	90	82.4	70	70	70.4
6	87.5	81.6	75	70	70.4
8	91.6	82.8	91.6	73.5	75
10	90	7.5	75	70.8	75
15	87.5	80	83.3	73	75
20	91.6	77.5	97	68	79
25	87.5	85	92.5	70	82
30	83.3	83.75	83.3	79	78.6
50	90	75	75	75	86

The PA achieved by mining AR through GA with Elitism is optimal and the convergence rate varies from dataset to dataset, depending on the size of the dataset and the correlation between attributes.

The maximum PA achieved by Elitist GA is plotted against the accuracy of the rules mined with simple GA defined in section 4.3, is shown in Figure 4.8.

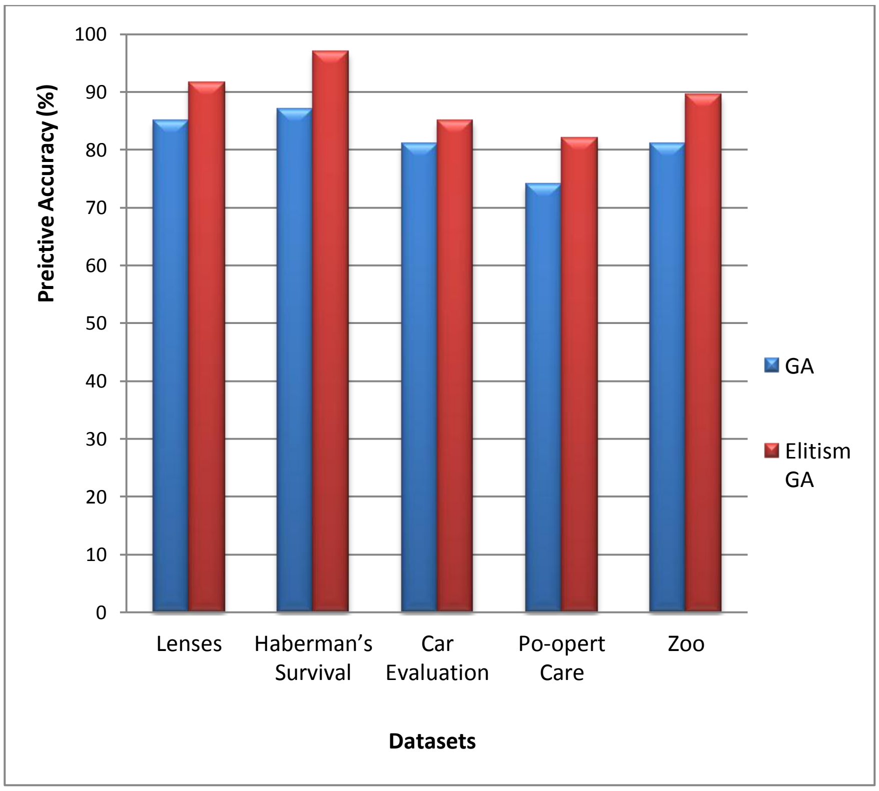


Figure 4.8 Predictive Accuracy for Mining AR based on GA with Elitism

The elitist based GA generates ARs with better PA when compared to simple GA for all the five datasets. The enhancement in the accuracy of the mined rules is significant.

The elites that could be archived by elitism for the lenses dataset is compared by varying selection methods without elitism concept and is plotted as shown in Figure 4.9. Here random selection and roulette wheel selection without elitism is compared with the elites of roulette wheel selection with Elitism.

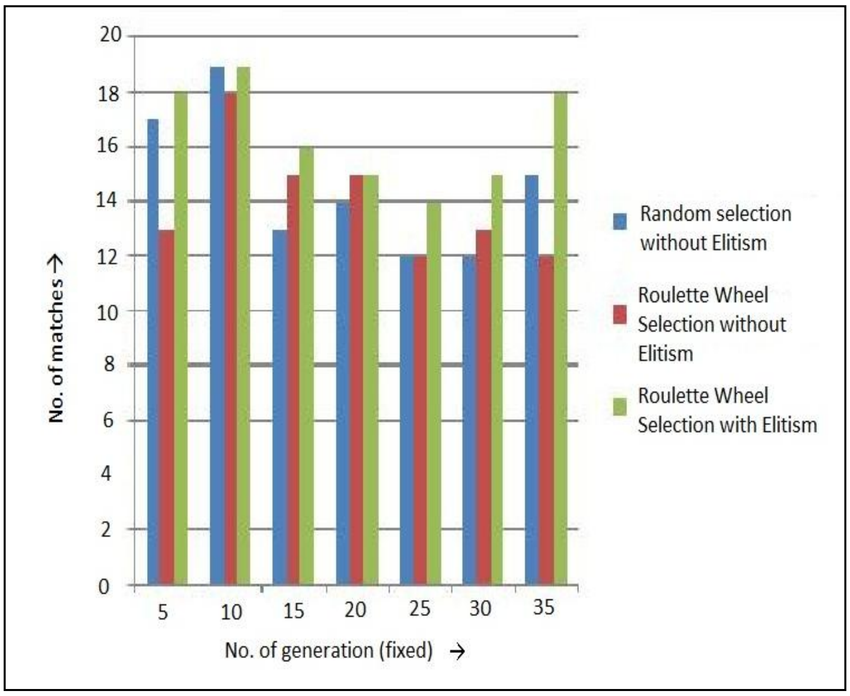


Figure 4.9 Comparison of Elites for ARM with and without Elitism

It can be inferred from the above figure that the roulette wheel selection method with elitism generated more elites when compared to the selection methods without elitism as the iteration progresses.

The qualitative measures of the ARM with elitist based methodology are given in the Table 4.10. The Laplace measure is a confidence estimator and the value away from 1 indicates that the rules generated are of value. The conviction measure being infinity for all datasets indicates that the rules generated are interesting. The range of the values of Leverage measure ranges between [-0.25, 0.25]. The measure of ARs mined with Elitist GA are all less than 0.01 indicates that the rules generated are compact (antecedent and consequent are correlated). Similarly the Lift measure being close to one signifies that the antecedent and the consequent in rules generated with Elitist GA are associated.

Table 4.10 Qualitative Measures of ARM with Elitist GA

Measures	**Lenses**	**Haberman's Survival**	**Car Evaluation**	**Postoperative Patient**	**Zoo**
Laplace	0.505582	0.538462	0.5	0.5024	0.5
Lift	3.263158	1.846154	2.08	5.05	1.279
Conviction	Infinity	Infinity	Infinity	Infinity	Infinity
Leverage	0.015661	0.076389	0.006	0.0079	0.007

The other quantitative measures like the number of rules generated by the Elitist GA with PA less than 0.05% is plotted against the ARM with simple GA in the Figure 4.10.

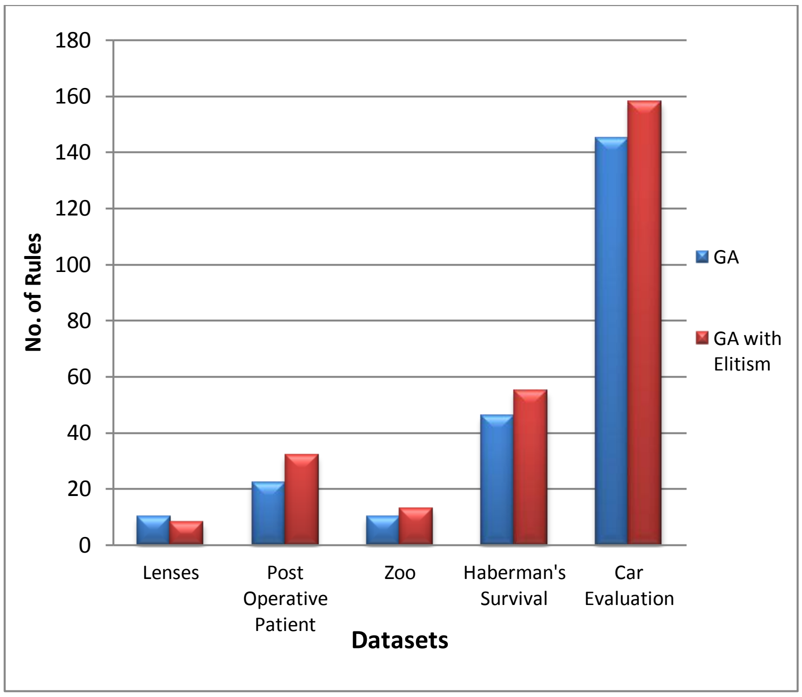


Figure 4.10 Comparison of Number of Rules Generated with Elitist GA and Simple GA

The number of rules generated by GA with elitism is more when compared to simple GA. Preserving the Elite population as archive to the next generations helps in retaining better solutions for future generation. Hence, more number of rules is generated with this method. In summary in can be stated that GA with Elitism performs better than simple GA for ARM.

4.6 ADAPTIVE GA FOR ARM

Rule acquisition is a technique of data mining that is used to deduce inferences from large databases. These inferences cannot be noticed easily without data mining applications. GA is considered as a global search approach for optimization problems. Through the proper evaluation strategy, the best “chromosome” can be found from the genetic combinations. In the self-adaptive GA, the main thought is to let control parameters (crossover rate, mutation rate) adjust adaptively within the proper range, and thus help to achieve an optimum solution.

In the traditional GA, the crossover and mutation rates are fixed values, which are selected based on experience. Generally, it is believed that when the crossover rate is too low, the evolutionary process can easily fall into local optimum resulting in groups of premature convergence due to population size and the lack of diversity. When the crossover rate is too high, the process is optimized to the vicinity of optimal point and the individual is difficult to reach optimal point which can slow the speed of convergence significantly, though groups can ensure the diversity.

4.6.1 Need for Adaptation

GA has found a wide amount of application in data mining, where knowledge is mined from large databases. GAs can be used to build effective classifier systems (De Jong et al. 1993; Holland,1986), mining ARs (Jacinto et al. 2002; Deepa et al. 2003) and other such data mining problems. Their robust search technique has given them a central place in the field of data mining and machine learning.

In GA, the three basic operators namely, selection, crossover and mutation operators, are fixed apriori. The optimum values for these operators depend on the problem to which the GA is applied, and also on the fitness of the current population.

The issue of parameter setting has been relevant since the beginning of the EA research and practice. Eiben et al. (2003 and 2007) emphasize that the parameter values greatly determinate the success of an EA in finding an optimal or near-optimal solution to a given problem. Choosing appropriate parameter values is a demanding task. There are many methods to control the parameter setting during an EA run. Parameters are not independent. But trying all combinations is an impossible task and the found parameter values are appropriate only for the tested problem instances. Nannen et al. (2008) state that major problem of parameter tuning is weak understanding of the effects of EA parameters on the algorithm performance.

Researchers have proposed different adaptive approaches for EA to make the parameters evolve by themselves. Back (1992) embedded the mutation rate into chromosomes in GA to observe the performance of the self-adaptive approach in different functions. Spears (1995) added an extra bit in the schematic of chromosome to investigate the relative probability of operating two-point crossover and uniform crossover in GA. Hop and Tabucanon (2005), presented a new and original approach to solve the lot size problem using an adaptive GA with an automatic self-adjustment of three operator rates, namely, the rate of crossover, mutation and reproduction operations. A hybrid and adaptive fitness function (Tang and Tseng 2012), in which both filter and wrapper approaches were applied for feature selection via GA to generate different subsets for the individual classifiers.

The efficiency of the rules mined by GA mainly depends on the mutation rate and the crossover rate which affects the convergence rate (Eiben et al. 2007). Therefore tuning the value of the mutation rate becomes an important criterion for mining ARs with GA. Higher mutation rate generates chromosomes much deviated from original values resulting in higher exploration time. Lower mutation rate results in crowding of chromosomes towards the global optima thereby limiting the search space. The dataset from which ARs are mined is considered for modifying the mutation rate during evolution process. So mutation rate is made adaptive based on the history of mutation rate and the fitness value. The algorithm for adaptive GA for mining ARs is as given below:

```
Initialize population randomly;
Evaluate fitness of each individual in the population;
While the stopping condition is not met
{
   Perform selection;
   Perform crossover and mutation;
   Evaluate fitness of each individual;
   Change mutation operator.
}
```

The mutation operator is made adaptive as given in equation (4.3)

$$p_m^{(n+1)} = \lambda p_m^0 \sqrt{\frac{(f_{max}^{(n+1)} - f_{mean})^2}{(f_{max}^{n+1} - f_i^n)^2}} \tag{4.3}$$

where, $p_m^{(n+1)}$ is the mutation rate of $(n+1)^{th}$ generation; the first generation mutation rate is p_m^0; f_{mean} is the mean fitness of itemset; $f_{max}^{(n+1)}$ is the highest fitness of the $(n+1)^{th}$ individual stocks; f_i^n is the fitness of the n^{th} individual I and λ is the adjustment factor, which is set within the range [0,1].

The selection of individuals for reproduction (crossover and mutation) is based on fitness value. The fitness function given in Eqn. (4.2) is applied for ARM. Adapting the mutation rate based on the fitness values of the itemset results in generation of offspring for the new population with higher fitness value (Srinivasa et al. 2007). This leads easily towards the global optima thereby maintaining the search space effectively. The main drawback of GA is its lack of memory (ability to retain the previous fitness values). The mutation rate adaption based on the history of mutation rates solves this problem, thus enhancing the performance of GA.

4.6.2 Experimental Results and Discussion

The objective of the adaptive GA and adaptive PSO is to enhance the performance of ARM by making the adaption dependent on dataset used. To validate the performance of the proposed methodology, five datasets from University of California Irvine (UCI) repository have been used for the study. The experiments were developed using Java and run in windows environment. The best of the five executions were recorded. The number of iterations was fixed to 100.

In adaptive GA mutation rate was made self adaptive based on the analysis of control parameters, where the mutation operation was found to influence the accuracy of the system while the crossover operation affects the convergence rate alone. Hence the crossover rate is kept at fixed value. For the mutation rate, in addition to the feedback from earlier mutation rate, the fitness value is also considered during adaptation. This enhances the accuracy of the ARs mined and makes the adaption data-dependent. The initial parameter setting of the GA parameters for ARM is listed in Table 4.11.

Table 4.11 Default GA Parameters for ARM

Parameter Name	Value
Population Size	Varies as per the dataset
Initial Crossover rate	0.9
Initial Mutation rate	0.1
Selection Method	Roulette wheel selection

The PA of the ARs mined by this method is compared with the results of Simple GA as shown in Figure 4.11.

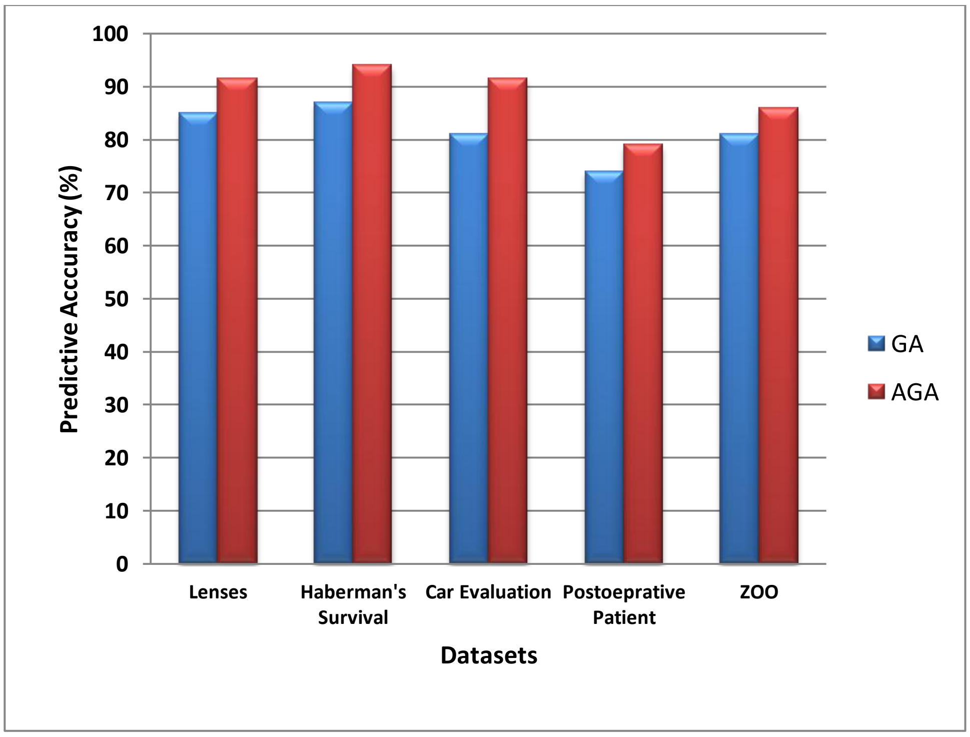


Figure 4.11 Comparison of the Accuracy between GA and AGA when Parameters are Ideal for Traditional GA

A considerable improvement is achieved in PA for all datasets by using AGA for ARM. The PA of the rules mined by AGA has improved between 8% and 12.5% for the datasets in comparison with the simple GA. The adaption of the mutation rate based on its previous value along with the fitness value results in generating the diversified population enhances the performance of AGA. The mutation rate is GA is fixed for all generations and hence lower performance.

To analyze the effects of AGA on accuracy the mutation rate of the adaptive GA at the final iteration was noted. The noted mutation rate was applied for the simple GA and compared with the results of AGA. This is shown in Figure 4.12.

The adaptive GA methodology gives better performance than GA. The performance of GA with the mutation rate obtained at final iteration of AGA is inferior to the simple GA's performance.

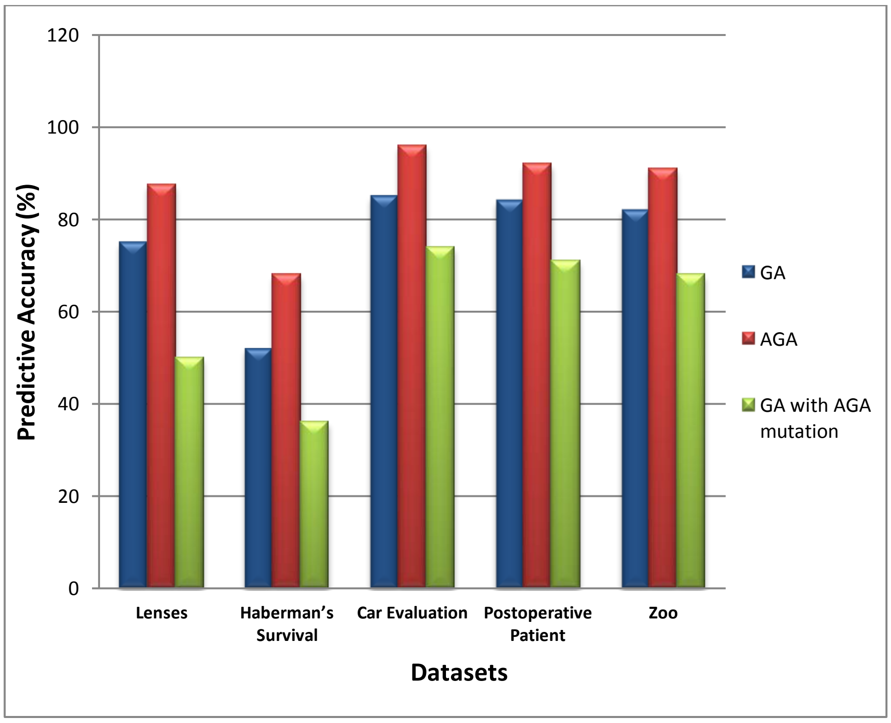


Figure 4.12 Predictive Accuracy Comparison of AGA, GA and GA with AGA Mutation Rate

Accuracy alone is not the objective of mining ARs. The interestingness of the ARs mined is also measured through Lift, Laplace, Conviction and Leverage measures and presented in Table 4.12.

Table 4.12 Qualitative Measures of ARM by AGA

Dataset	Lenses	Haberman's Survival	Car Evaluation	Postoperative Patient	Zoo
Laplace	0.65201	0.64782	0.652488	0.6528	0.6524
Conviction	Infinity	Infinity	Infinity	Infinity	Infinity
Leverage	0.0324	0.0569	0.04356	0.0674	0.0765
Lift	1.89	1.89	1.89	1.89	1.89

The Laplace and Leverage values are away from 1 for all datasets which indicates that the rules generated are of interest. The conviction value, infinity also signifies the importance of the rules generated. Similarly the Lift measure being close to 1 means that the antecedents and the consequents are related, indicating the importance of the rules generated.

The PA of the mines ARs achieved using the above proposed methodologies are given in Table 4.13.

Table 4.13 Comparison of PA Achieved by the GA based Methods

Dataset	GA	GA with Elitism	Adaptive GA
Lenses	85	91.6	91.6
Haberman's Survival	87	94	97
Car Evaluation	81	91.6	85
Postoperative Patient	74	79	82
Zoo	81	86	89.54

While the PA achieved by the GA was found to be optimal, GA with elitism generated ARs with enhanced PA due to carry over of elites through generations. The Adaptive GA methodology for ARM generated ARs with better PA for the datasets except the car evaluation. The increase in PA when compared to GA is 4% for the car evaluation dataset; 6.6% for the lenses dataset; 7.54 % for the zoo dataset; 8 % for the postoperative patient dataset and 10% for the haberman's survival dataset.

The age of the patient, an attributes in haberman's survival datasets has values with broad range. The PA increase of 10% achieved for this dataset signifies the superior performance of the AGA methodology. Similarly for the zoo dataset with 18 attributes the 7.54% increase in PA also supports this significance.

4.7 SUMMARY

The literature shows ARM implemented using GA has always resulted in better PA. However, setting the values of parameters required in GA mechanism is usually a time-consuming process. These parameters are usually set by trial and error. By assigning all the allowed values to the GA parameters and using different data sets, optimal values for these parameters were arrived in this study based on the experiments carried out. It is found that such optimal values are within the allowed range of values. The values of minimum support, minimum confidence decides upon the accuracy of the system while crossover rate affects the convergence rate. The Elitist based GA generated ARs with better PA than simple GA and the number of significant rules generated with Elitist GA is more than the rules generated with Simple GA. The adaptive parameter setting of the mutation rate was carried out based on the mutation rate history and the fitness values of the individuals. The AGA methodology of ARM generated AR with enhanced PA, when compared to simple GA and Elitist GA.

CHAPTER 5

PARTICLE SWARM OPTIMIZATION FOR MINING ASSOCIATION RULES

5.1 GENERAL

This chapter focuses on ARM using PSO methodology. Modifications were made in the PSO velocity update for avoiding premature convergence. In this regard, chaotic behavior and dynamic neighborhood selection were introduced in the basic PSO method. The methodology and experimental results of the new PSO variants are also presented in this chapter. The avoidance of premature convergence is attempted by adjusting the control parameters: inertia weight and acceleration coefficients. Two different adaption mechanisms namely, data-independent and data-dependent mechanisms are also dealt in this chapter for mining ARs.

5.2 PSO FOR ARM

GA when applied for ARM generates ARs with better PA. But, the time taken for mining the rules is more, and the reproduction operator at times produced offspring crowding towards local optima.

To overcome the above demerits, PSO was taken up for mining ARs. PSO shares many similarities with evolutionary computation techniques such as GA. However unlike GA, PSO has no evolution operators such as crossover and mutation and also possess the advantage of storing the previous history of particles. PSO has proved to be competitive with GA in several tasks, mainly in optimization areas.

The initial ideas on particle swarms of Kennedy (a social psychologist) and Eberhart (an electrical engineer) were essentially aimed at producing computational intelligence by exploiting simple analogues of social interaction, rather than purely individual cognitive abilities. The first simulation (Kennedy and Eberhart 1995) was influenced by Heppner and Grenander's work (Heppner and Grenander 1990) and involved analogues of bird flocks searching for corn. Soon this soon was developed into a powerful optimization method— PSO (Kennedy and Eberhart 1995; Eberhart and Kennedy 1995; Eberhart et al. 1996). The concept of ARM using PSO is discussed in the following section.

5.2.1 Methodology

PSO is initialized with a group of random particles (solutions) and then searches for optimum value by updating particles in successive generations. In each iteration, all the particles are updated by following two "best" values: pBest and gBest. The movement of the particle in PSO towards the global optima (target) is shown in figure 5.1 below.

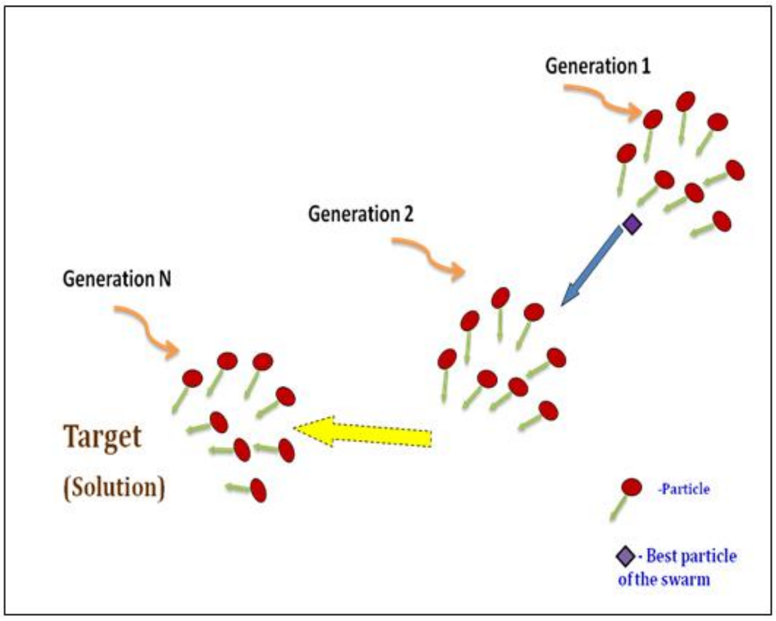


Figure 5.1 Particles Movement towards the Target

PSO is a new evolutionary algorithm, which simulates the coordinated motion in flocks of birds. Sousa, Silva, and Neves (2004), proposed the use of PSO for data mining. PSO can achieve the rule discovery process. The rule representation in PSO uses the Michigan approach. PSO needs fewer particles than GA to obtain the same results.

The flowchart of the PSO algorithm for mining ARs is shown in Figure 5.2. Each record in the dataset with its attributes is represented as a particle in PSO. During iteration process, the velocity and position of all particles are updated based on velocity and position update equations (3.1) and (3.2), respectively.

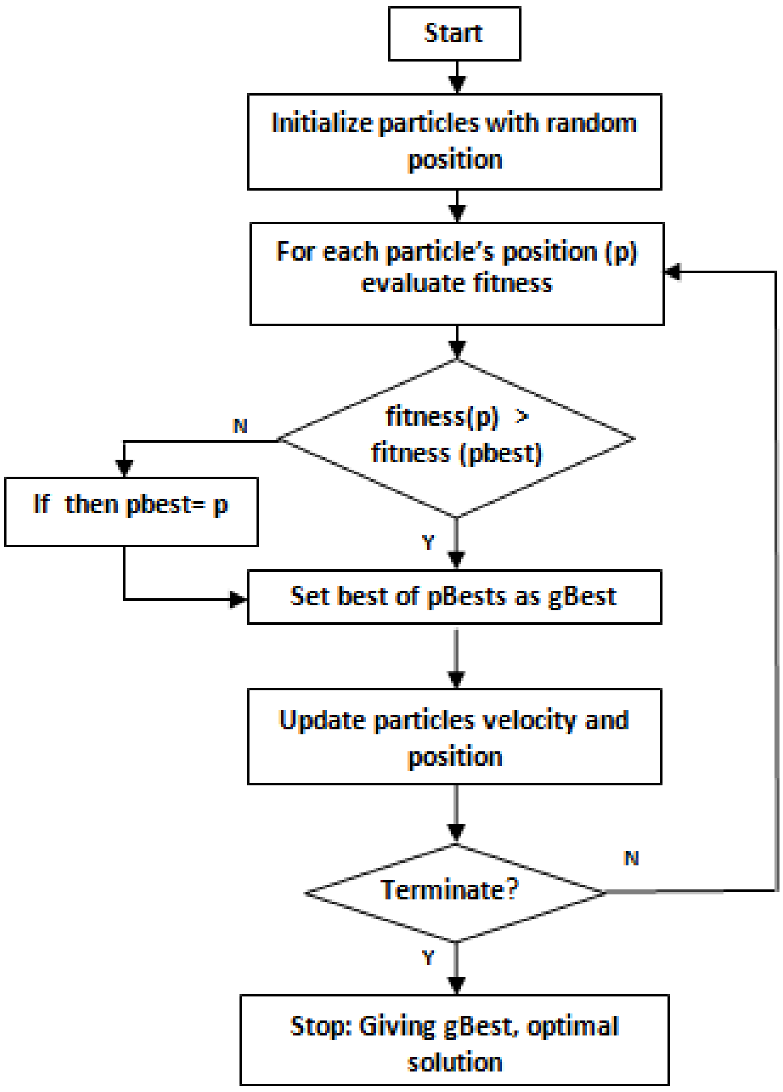

Start
Initialize particles with random position
For each particle's position (p) evaluate fitness
fitness(p) > fitness (pbest)
N
If then pbest= p
Y
Set best of pBests as gBest
Update particles velocity and position
Terminate?
N
Y
Stop: Giving gBest, optimal solution

Figure 5.2 Flowchart of PSO for ARM

Binary encoding is applied for representing each particle. The particles in this population are called initial particles. Initially the velocity and position of all particles are randomly set, within predefined range.

To reduce the possibility of particles flying out of the problem space, Eberhart et al. (1996) put forward a clamping scheme that limited the speed of each particle to a range [$-v_{max}$, v_{max}] with v_{max} usually being somewhere between 0.1 and 1.0 times the maximum position of the particle. In this study v_{max} is set as the maximum value of the particles (x_i).

Whilst experimenting with the standard algorithm, Shi and Eberhart (1998), noted that without the velocity memory v_i^{old}, the first part of Eqn. (3.1), the swarm would simply contract to the global best solution found within the initial swarm boundary (providing a local search). Conversely, with the velocity memory, the swarm will behave in the opposite sense, expanding to provide a global search. In order to achieve the balance between exploration and exploitation, a modified PSO incorporating an inertia weight (ω), was introduced thus:

$$v_i^{new} = \omega v_i^{old} + c_1 rand()(pBest - x_i) + c_2 rand()(gBest - x_i) \quad (5.1)$$

The initial experimentation (Carlisle and Dozier 2000; Trelea 2003) suggested that a value between 0.8 and 1.2 provided good results, although in later work (Eberhart and Shi 2000) they indicate that the value is typically set to 0.9 (reducing the stepwise movement of each particle, allowing greater initial exploration) reducing linearly to 0.4 (speeding convergence to the global optimum), during an optimization run.

To study the impact of inertia weight on PSO for ARs mining the inertia weight is varied and the accuracy achieved by ARM was analyzed.

5.2.2 Experimental Results and Discussion

The objective of this study is to analyze the performance of PSO for mining ARs. PSO models without and with inertia weights were applied for ARM. ARs are mined from the five datasets described in section 3.5. The parameters of the PSO model without inertia weight for ARM is given in Table 5.1.

Table 5.1 Initial Parameter Setting of PSO for ARM

Parameter Name	**Value**
Population Size	Lenses : 20 Haberman's Survival : 300 Car Evaluation : 1000 Postoperative Care : 50 Zoo : 100
No. of Generations	100
c_1, c_2	2
v_{max}	Maximum (x_i)

Each methodology was run 10 times on the dataset chosen. The PA achieved for the five datasets is presented in Table 5.2. The same is compared with the simple GA methodology described in Chapter 4.

Table 5.2 Predictive Accuracy Comparison of PSO and GA for ARM

Dataset Name	GA	PSO
Lenses	85	92.8
Haberman's Survival	87	94.4
Car evaluation	81	91.6
Post Operative Patient	74	83.33
Zoo	81	95.45

The PA obtained through PSO methodology is better than the performance of simple GA for all the five datasets. The simplicity of PSO with minimum function evaluations generates better ARs when compared to GA.

The execution time (CPU) is the time taken for mining ARs by PSO methodology until the completion of specified number of generations. The execution time results obtained for GA and PSO are presented in Figure 5.3.

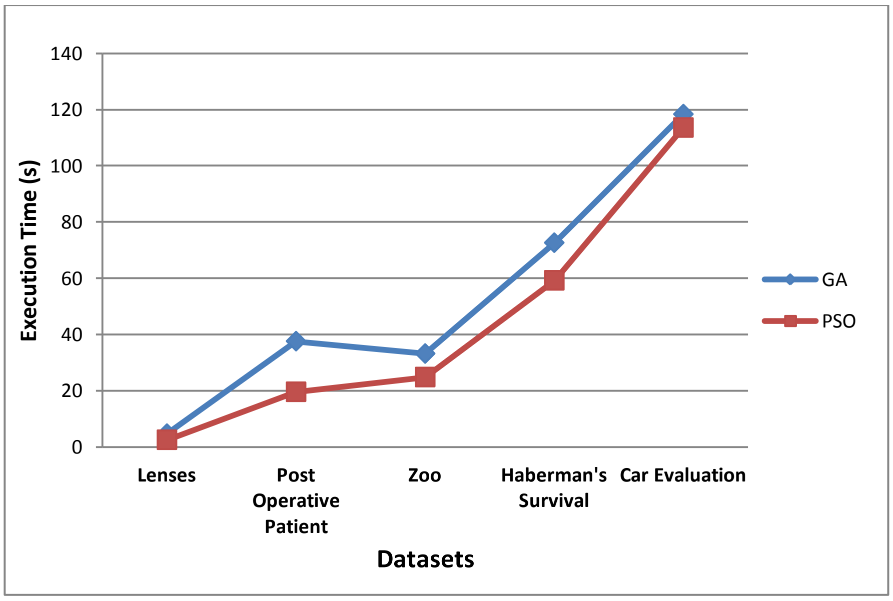


Figure 5.3 Execution Time comparison of PSO and GA for ARM

In terms of computational effectiveness PSO is found to be marginally faster than GA. PSO shares many similar properties with GAs. Both methods begin with a group of randomly initialized population, evaluate their population based on fitness function. However, the main difference between PSO and GA is that PSO does not have the genetic operators as crossover and mutation. In PSO, only the best particle passes information to others and hence, the computational capability of PSO is marginally better than GA.

As the efficiency of PSO in terms of PA and execution time have been found to be better than GA, the qualitative measures defined in chapter 3 are determined for the five datasets. The results are given in Table 5.3.

Table 5.3 Qualitative Measures of ARM with PSO

	Lenses	**Haberman's Survival**	**Car Evaluation**	**Postoperative Patient**	**Zoo**
Laplace	0.52	0.5	0.5	0.5	0.5
Lift	2	2.08	1.05	1.279	6.314
Conviction	Infinity	Infinity	Infinity	Infinity	Infinity
Leverage	0.0625	0.006	0.0015	0.007	0.016

The Laplace measure, away from the value 1, signifies that the antecedent values are dependent on the consequent and hence the rules generated are of importance. The conviction measure which is infinity for all datasets show that the rules generated are interesting. The Leverage measure being far away from 1 again insists on the interestingness of the rules generated. The measure of Lift for all the datasets is greater than1 signifies that the dependency of the consequent on the antecedent. Thus the rules generated by PSO are of importance based on the Lift measure.

In the velocity update function (Eqn 3.1) the velocity of the particle in previous generation is added as such, to the current velocity. This makes an impact on the previous particle's movement on the new velocity. Thus, the exploration ability of the particle gets restricted. To reduce the impact of earlier velocity on current velocity, the inertia weight parameter is added as given in Equation (6.1). The inertia weight impart only portion of earlier velocity into new one. The inertia weight is varied and the PA achieved by the generated rules from the five datasets is given in Figure 5.4 below.

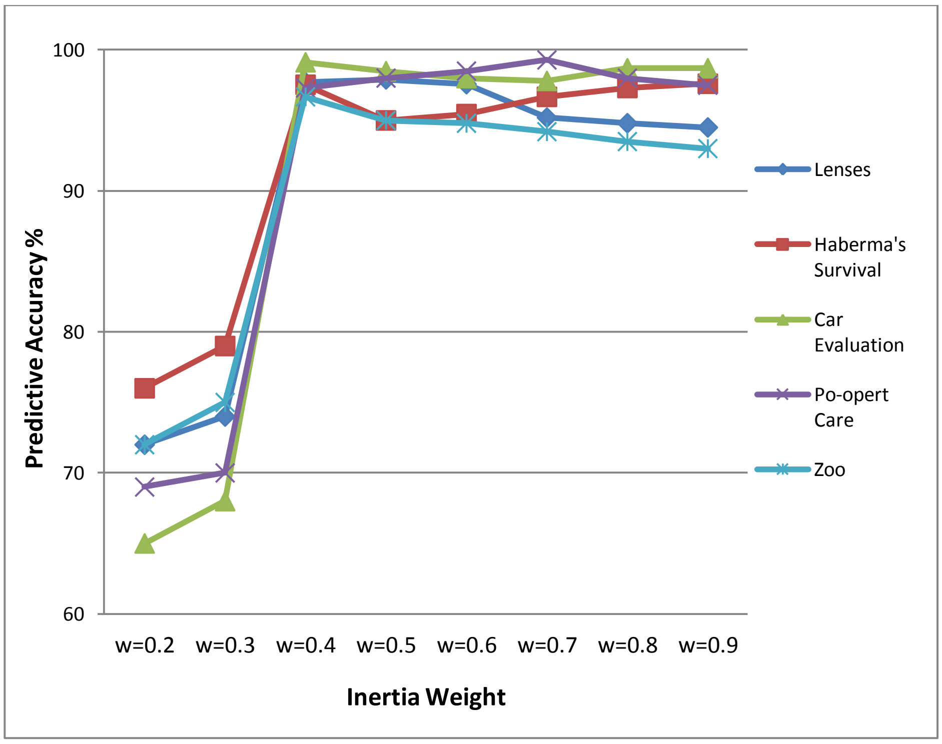


Figure 5.4 Impact of Inertia Weight on Predictive Accuracy for ARM

From the above figure it is inferred that for the inertia weight (ω) in the range [0.4, 0.9], the PA achieved is optimum. For inertia weight in range [0.5, 0.7] the PA achieved is maximum depending on datasets used. When inertia weight is between 0.2 and 0.3 the PA achieved is inconsistent. The PA achieved by the weighted PSO for ARM is plotted in comparison with PSO without inertia weight, as shown in Figure 5.5.

The PA of weighted PSO (WPSO) has enhanced, for all the five datasets in comparison with PSO. The accuracy enhancement achieved is varying from 1.27% to 14% for the datasets chosen. An increase in PA of 14% is achieved for the postoperative care dataset. Marginal enhance of 1.27% of PA is noted for the zoo dataset. A better balance between exploration and exploitation is obtained by the addition inertia weight and hence, better performance by the WPSO.

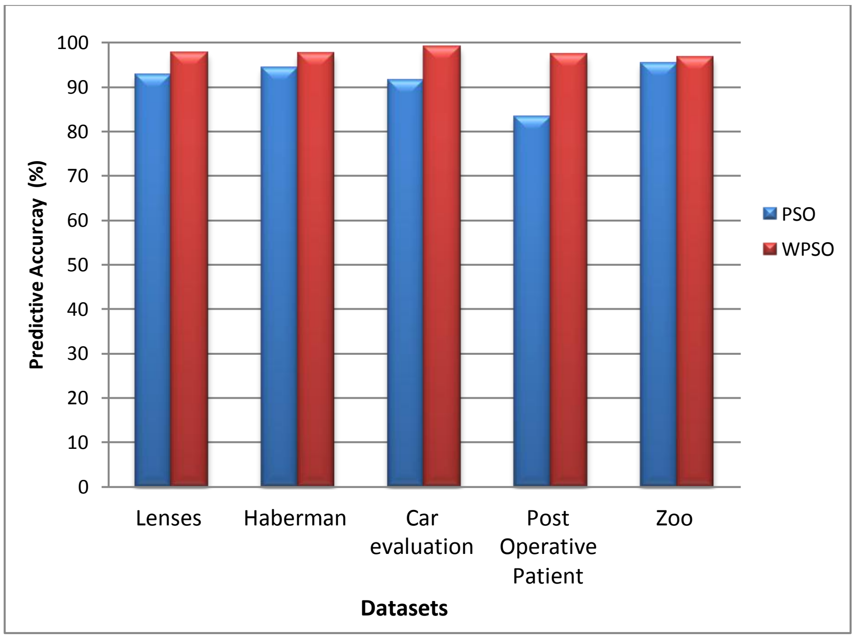


Figure 5.5 Comparison of PSO with and without Inertia Weight for ARM

5.3 CHAOTIC PSO FOR ARM

The canonical PSO has a tendency to get struck at local optima and thereby leading to premature convergence when applied for solving practical problems. To improve the global searching capability and escape from the local optima, chaos is introduced in PSO (Alatas et al. 2007). Chaos is a deterministic dynamic system, very sensitive and dependent on its initial conditions and parameters. In this section chaotic behavior introduced in PSO is presented to maintain the diversity of population and to enhance the performance of ARM process.

5.3.1 Chaotism in PSO

To improve the global searching capability by escaping from the local solutions, sequences generated from different Zaslavskii Map (Zhang and Huang 2004) substitute random numbers for different parameters of PSO, where it is necessary to make a random-based choice. Coupling such behavior with PSO and complex dynamics, can improve the quality of results in some optimization problems and also that chaos may be a desired process. It has been empirically shown that introducing chaotic maps with ergodicity, irregularity and the stochastic property in PSO, the global convergence has been improved (Alatas et al., 2007).

The common method of generating chaotic behavior is based on Zaslavskii map (Zhang and Huang, 2004). This representation of map involves many variables. Setting right values for all the variables involved, increases the complexity of the system. And erroneous values might bring down the accuracy of the system involved. Logistic map and tent map also have the most frequently used chaotic behavior. The drawback of these maps is that the range of values generated by both maps after some iteration becomes fixed to a particular range. To overcome this defect the tent map undisturbed by the logistic map (Kong et al., 2010) is introduced as the chaotic behavior. A new chaotic map model is proposed with the following equation, based on the tent and logistic maps.

$$\begin{aligned} u_{k+1} &= 4 * u_k(1-u_k) \qquad 0 \le u_k \le 1 \\ v_{k+1} &= \begin{cases} 1/1.001 * (2 * v_k + 0.001 * u_{k+1}) & 0 \le v_k \le 0.5 \\ 1/1.001 * (2 * (1-v_k) + 0.001 * u_{k+1}) & 0.5 < v_k \le 1 \end{cases} \end{aligned} \tag{5.2}$$

The initial value of u_0 and v_0 are set to 0.1. The slight tuning of initial values of u_0 and v_0 creates wide range of values with good distribution. Therefore, the chaotic operator *chaotic_operator(k)* = v_k is designed to generate different chaotic operators by tuning u_0 and v_0. The value u_0 is set to two different values for generating the chaotic operators 1 and 2.

The velocity updation equation based on chaotic PSO is given in Equation 5.3.

$$v_i^{new} = v_i^{old} + c_1 * Chaotic\_operator_1(lbest - x_i) + c_2 * Chaotic\_operator_2\ (gbest - x_i) \quad (5.3)$$

In the above equation, the random number generator is replaced with *chaotic_operator₁ and chaotic_operator₂*. Chaotic behavior is replaced by random number to create a new population with diversity. When diversity exists, the exploitation capability of PSO increases. Thus, the global search is performed effectively and the performance also increases.

5.3.2 Experimental Results and Discussion

The chaotic PSO (CPSO) was tested on the bench mark datasets discussed earlier in chapter 3, for performance analysis. The number of generations was set to 100. The acceleration coefficients c_1 and c_2 were both set to 2. The initial velocity was set as 0 for all the particles and the value for V_{max} was set as 1. Each methodology was run 10 times on the chosen dataset.

The PA achieved by CPSO for ARM from the five datasets is plotted in Figure 5.6. The PA achieved by PSO is compared with the obtained results.

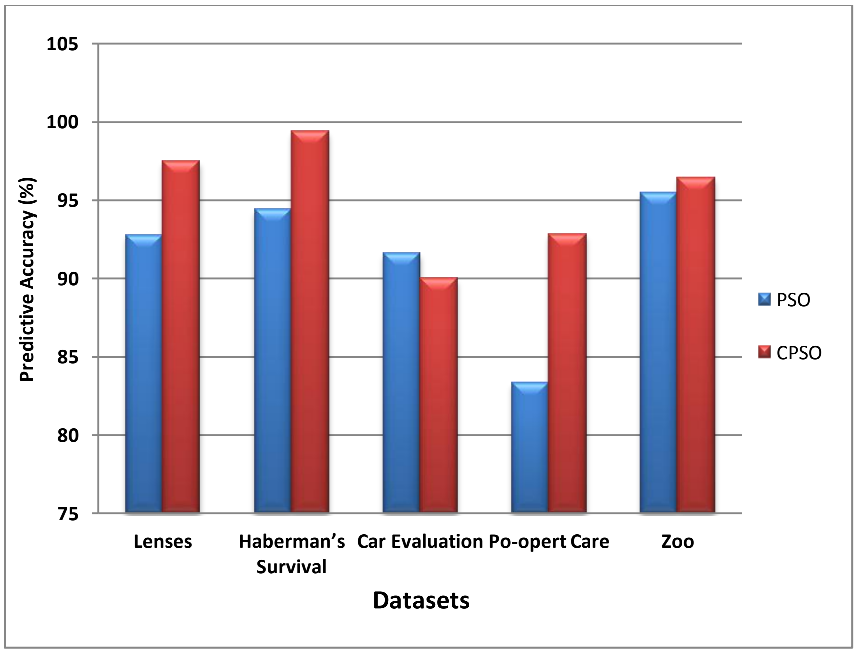


Figure 5.6 Comparison of Predictive Accuracy of CPSO and PSO for ARM

Chaotic PSO generated ARs with enhanced PA, in comparison to PSO for all the five datasets. The convergence rate (generation at which maximum PA achieved) of the CPSO for mining ARs from the five datasets chosen for analysis is presented in the Figure 5.7.

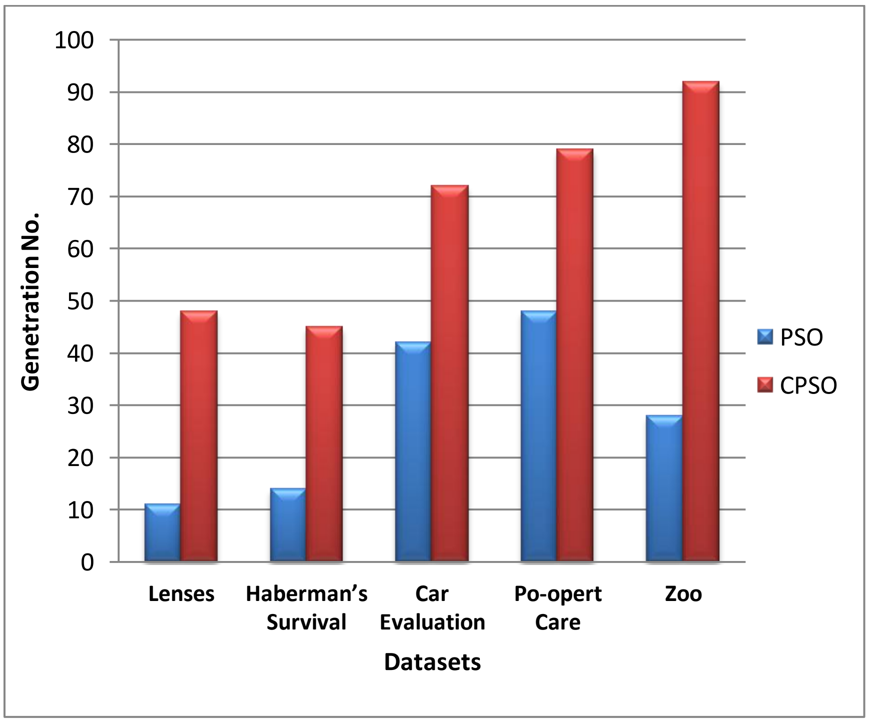


Figure 5.7 Convergence Rate of CPSO for ARM

It could be noted form the Figure 5.7 that the PSO methodology generated ARs with maximum accuracy at the earlier stages of iteration (10-40) for the five datasets. To avoid premature convergence chaotic behavior was introduced in PSO. This chaotic behavior resulted in diversification of particles, thereby evading the premature convergence. Thus the convergence has shifted to later iterations in CPSO (30-70).

The chaotic operators could be changed by altering the initial values in chaotic operator function. The balance between exploration and exploitation is maintained by CPSO methodology.

5.4 DYNAMIC NEIGHBORHOOD SELECTION IN PSO FOR ARM

The problem of not reaching the optimal solution and deviation from optimal solution space are addressed via gBest and pBest values respectively in PSO. The global best propagates information at the fastest, in the population dealing with exploration, while, the local best using a ring structure speeds up the system balancing the exploration.

In published literature efforts have been directd to enhance the performance of PSO. In Gregarious PSO (Pasupuleti and Battiti, 2006) the social knowledge of the particle is used for discovery in the search space. If particles are trapped in the local optimum, a stochastic velocity vector thereby self sets the parameters. In Dynamic neighborhood PSO (Lu and Chen, 2008) instead of using the current gBest, another parameter Neighborhood Best (Nbest) is utilized. This term is the best particle among the current particle's neighbors in a specified neighborhood.

Fixing up the best position for particles after velocity updation by using Euclidean distance helps in generating the best particles (Kuo et al. 2011). The problem of getting struck at local optimum and hence premature convergence is overcome by self adaptive PSO (Lu et al. 2010), where, the diversity of population is maintained. This copes up with the deception of multiple local optima and reduces computational complexity. An adaptive chaotic PSO (Cehng at al. 2011) enhances the global searching capability and local searching capability by introducing chaotic operators based on Logistic map and tent map. In addition an adaptive search strategy which optimizes continuous parameters is employed.

To avoid premature convergence and enhance the accuracy the Neighborhood selection in PSO (NPSO) was proposed replacing the particle best concept by local best.

5.4.1 Dynamic Neighborhood Selection in PSO

The balance between exploration and exploitation in PSO is the main issue when applied for solving complex problems. The velocity update equation plays a major role in enhancing the performance of the PSO. To maintain the diversity of the particles and enhance the performance of the PSO, the concept of adapting the local best particle among the neighbors is proposed for mining ARs. The proposed work is to review the PSO for mining ARs with dynamic neighborhood selection.

From the study carried out on the five datasets, PSO proves as effective as GA in mining ARs. In terms of computational efficiency, PSO is marginally faster than GA. The pBest and gBest values tend to pass the information between populations more effectively than the reproduction operators in GA. The neighborhood best called local best (lBest) selection is as follows;

- Calculate the distance of the current particle from other particles by equation (5.4).

$$\Delta x_i = |(x_i - x_{gbest})| \quad (5.4)$$

- Find the nearest m particles as the neighbor of the current particle based on distance calculated
- Choose the local optimum lBest among the neighborhood in terms of fitness values

The flowchart of the proposed algorithm is given in Figure 5.8. The number of neighborhood particles m is set to 2. Velocity and position updation of particles are based on equation (3.1) and (3.2). The velocity updation is restricted to maximum velocity V_{max}, set to the maximum of x_i. The termination condition is set as fixed number of generations.

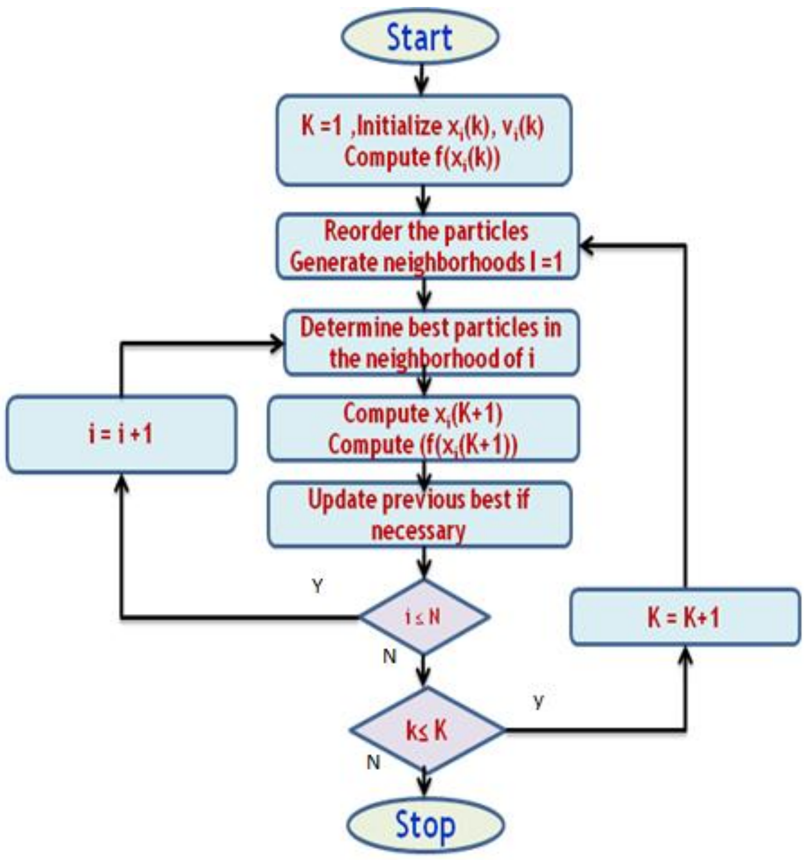

Start
K =1 ,Initialize x_i(k), v_i(k)
Compute f(x_i(k))
Reorder the particles
Generate neighborhoods I =1
Determine best particles in the neighborhood of i
i = i +1
Compute x_i(K+1)
Compute (f(x_i(K+1))
Update previous best if necessary
Y
i ≤ N
N
K = K+1
y
k≤ K
N
Stop

Figure 5.8 Flowchart for Association Rule Mining with Dynamic Neighborhood Selection in PSO

The velocity update function of NPSO is as defined in equation (5.5) and the position update function is adopted using Eqn. (3.2).

$$v_{id}^{new} = v_{id}^{old} + c_1 rand()(lBest - x_{id}) + c_2 rand()(gBest - x_{id}) \quad (5.5)$$

The interestingness of discovered ARs is an important and active area within data mining research. The measure of interestingness varies from application to application and from expert to expert. Each interestingness measure produces different results, and experts have different opinions of what constitutes a good rule. The interestingness measure for a rule is taken from relative confidence and is as follows:

$$interestingness(k) = \frac{\text{support}(x \cup y) - \text{support}(x)\text{support}(y)}{\text{support}(x)(1 - \text{support}(y))} \quad (5.6)$$

where, k is the rule; x the antecedent part of the rule and y the consequent part of the rule k.

5.4.2 Experimental Results and Discussion

The effectiveness of the proposed dynamic neighborhood selection in PSO methodology for ARM was tested on the five datasets selected for analysis. The parameter setting of the proposed NPSO methodology for ARM is listed in Table 5.4.

Table 5.4 Default Parameter Setting of NPSO for ARM

Parameter Name	Value
Population Size	Lenses : 20 Haberman's Survival : 300 Car Evaluation : 1000 Postoperative Care : 50 Zoo : 100
Initial Velocity	0
c_1	2
c_2	2
V_{max}	1
No of Iterations	100
No. of Runs	10

The maximum accuracy achieved from repeated runs is recorded as the PA for each case. The interestingness is calculated from the corresponding run. The PA achieved was compared with PSO for the same datasets and the highest PA achieved for multiple runs is plotted in Figure 5.9.

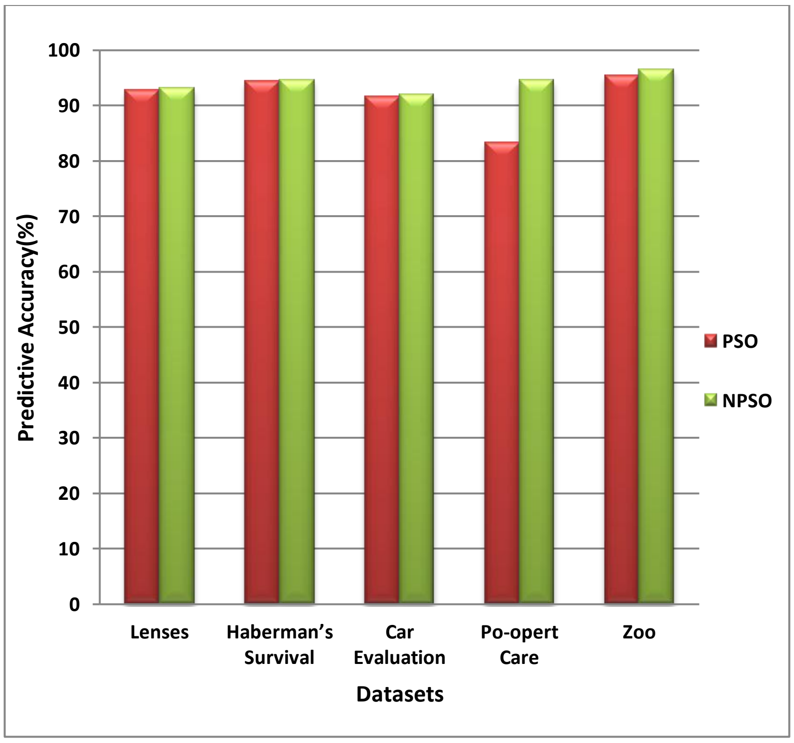


Figure 5.9 Predictive Accuracy Comparison for Dynamic Neighborhood selection in PSO

The PA achieved for all the five datasets chosen was better compared to PSO methodology. A maximum enhancement of 11.18% is achieved for the postoperative care dataset.

The interestingness or relative confidence measures of the mined rule are shown in Table 5.5. The ARs mined with dynamic neighborhood selection PSO is with good interestingness measure, indicating the importance of the mined rules.

Table 5.5 Measure of Interestingness for Dynamic Neighborhood Selection PSO

Dataset	Lens	Car Evaluation	Haberman's Survival	Postoperative Patient	Zoo
Interestingness Value	0.82	0.73	0.8	0.78	0.76

Experts using evolutionary algorithms observe that the time complexity of PSO is less, when compared to GA. Premature convergence may also result in reduced execution time. The scope of this work is to avoid premature convergence. The concept of local best based on neighborhood particles, instead of the individual best particles, focuses on this issue.

The convergence rate of NPSO for mining ARs is given in Table 5.6.

Table 5.6 Comparison of Convergence Rate of NPSO with PSO for ARM

Dataset	PSO (Generation No.)	NPSO (Generation No.)
Lens	10	60
Car Evaluation	20	60
Haberman’s Survival	10	60
Postoperative Patient	50	60
Zoo	20	60

The PSO methodology produces maximum results at the earlier iteration stages itself (10-50), whereas, the neighborhood selection in PSO extends the convergence rate to iterations more than 50, thus, avoiding premature convergence.

The selection of local best particles based on neighbors (lBest), replacing the particles own best (pBest), enhances the accuracy of the rules mined. The concept of local best (lBest) based on neighborhood selection in fitness space instead of other measures helps in maintaining the diversity of local points optimally, thus striking a balance between premature convergence and diversification of particles in problem space.

5.5 MOTIVATION FOR ADAPTIVE TECHNIQUES

While solving problems with PSO, its properties affect the performance. The properties of PSO depend on the parameter setting and hence users need to find the opt value for the parameters to optimize the performance. The interactions between the parameters have a complex behavior and so each parameter value will have a different effect depending on the value set for others. Without prior knowledge of the problem, parameter setting is difficult and time consuming. The two major ways of parameter setting are through parameter tuning and parameter control (Eiben and Smith, 2003). Parameter tuning is the commonly practiced approach that amounts to finding appropriate values for the parameters, before running the algorithm. Parameter control steadily modifies the control parameter values during the run. This could be achieved either through one of: deterministic, adaptive or self-adaptive techniques (Eiben et al. 2007).

Deterministic parameter control is effected through a deterministic rule that modifies the strategy parameter without any feedback. This method becomes unreliable for most problems because the parameter adjustments must rely on the status of the problem at current time. In self-adaptive approach, the parameters to be controlled are encoded into the candidate solution which may result in deadlock. Good solution depends on finding the good setting of parameters and obtaining the good setting of parameters depends on finding the good solution. Moreover, extra bits are required to store these strategy parameters, so the dimensionality of the search space increases. Thus, the corresponding search space becomes larger and hence the complexity of the problem increases. In this context, adaptive method is the better solution.

The parameters of PSO and their role in optimization process are presented in Table 5.7.

Table 5.7 Parameters of PSO and their Roles

Parameter	Parameter Role
Inertia weight (ω)	Controls the impact of the velocity history into the new velocity
Acceleration Coefficient c_1	Maintains the diversity of swarm
Acceleration Coefficient c_2	Convergence towards the global optima

The research attempts on dynamically adapting the parameter of PSO to enhance the efficiency of ARM based on: data independent and data dependent techniques as discussed in the following sections.

5.6 DATA INDEPENDENT ADAPTATION IN PSO

The velocity update equation plays a major role in enhancing the performance of the PSO. However, similar to other evolutionary computation algorithms, the PSO is also a population-based iterative algorithm. Hence, it can easily get trapped in the local optima when solving complex multimodal problems (Liang et al., 2006). These weaknesses have restricted wider applications of the PSO (Engelbrecht 2006).

To balance the global search and local search, inertia weight (ω) was introduced. It can be a positive constant or even a positive linear or nonlinear function of time (Shi and Eberhart 2002). Inertia weight plays a key role in the process of providing balance between exploration and exploitation process. The inertia weight determines the contribution rate of a particle's previous velocity to its velocity at the current time step. Eberhart and Shi (2001), proposed a random inertia weight strategy and experimentally found that this strategy increases the convergence of PSO in early iterations of the algorithm. In global-local best inertia weight (Arumugam and Rao 2006), the inertia weight is based on the function of local best and global best of the particles, in each generation. It neither takes a constant value nor a linearly decreasing time-varying value. Using the merits of chaotic optimization, Chaotic Inertia Weight has been proposed by Feng et al. (2010).

Gao et al. (2008), proposed a new PSO algorithm which combined the logarithm decreasing inertia weight with chaos mutation operator. Adaptive parameter control strategies can be developed based on the identified evolutionary state and by making use of existing research results on inertia weight (Shi and Eberhart 1999; Shi and Eberhart 2001). Some use a self-adaptive method by encoding the parameters into the particles and optimizing them together with the position during run time (Tripathi et al. 2007; Yamaguchi and Yasuda 2006).

To maintain effective balance between exploration and exploitation, the inertia weight is adapted during evolution independent of the dataset involved. The inertia weight is made adaptive by two different techniques.

5.6.1 Methodology

The original PSO has pretty good convergence ability, but also suffers the demerit of premature convergence, due to the loss of diversity. Improving the exploration ability of PSO has been an active research topic in recent years. Thus, the proposed algorithm introduces the concept of self adaptation as the primary key to tune the two basic rules velocity and position. By improving the inertia weight formulae in PSO the diversity of population could be achieved. The basic PSO, as resented by Eberhart and Kennedy (1995), has no inertia weight. In 1998, first time Shi and Eberhart presented the concept of constant inertia weight.

By looking at equation (3.1) more closely, it can be seen that the maximum velocity allowed actually serves as a constraint that controls the maximum global exploration ability the PSO can have. By setting a too small maximum for the velocity allowed, the maximum global exploration ability is limited, and hence, PSO will always favor a local search, no matter what the inertia weight is. PSO can have a large range of exploration ability, by setting optimum value for inertia weight. Since the maximum velocity allowed affects global exploration ability indirectly, and the inertia weight affects it directly, it is better to control global exploration ability through inertia weight only. A way to achieve this is by allowing inertia weight to control exploration ability. Thus, the inertia weight is made to change automatically (self adaptive). Three self adaptive inertia weights methods namely: Self Adaptive PSO1 (SAPSO1), Self Adaptive PSO2 (SAPSO2) and Self Adaptive chaotic PSO (SACPSO) are introduced for mining ARs in this work.

In order to linearly decrease the inertia weight as iteration progresses the inertia weight is made adaptive through the equation (5.7) in SAPSO1.

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min})\frac{g}{G} \tag{5.7}$$

where, ω_{max} and ω_{min} are the maximum and minimum inertia weights, g is the generation index and G is the predefined maximum number of generation.

In SAPSO2 the inertia weight adaptation is made to depend upon the values from previous generation so as to linearly decrease its value with increasing iterations, as given by equation (5.8).

$$\omega(t+1) = \omega(t) - \frac{(\omega_{max} - \omega_{min})}{G} \tag{5.8}$$

where, $\omega(t+1)$ is the inertia weight for the current generation, $\omega(t)$ inertia weight for the previous generation, ω_{max} and ω_{min} are the maximum and minimum inertia weights and G is the predefined maximum number of generation.

The value of ω_{max} and ω_{min} are set as 0.9 and 0.4 respectively. The values are arrived from section 5.2 (Figure 5.4) where the PA of ARM arrived is optimum, when values are within the range [0.4, 0.9].

The velocity update equation specified in Eqn. (6.1) is applied for mining ARs. The position update is made as given in equation 3.2. For both SAPSO1 and SAPSO2 methods, the inertia weight is adjusted during evolution based on equation (5.7) and (5.8).

In the proposed SACPSO methodology, the inertia weight adaption is made in chaotic PSO for ARM. The velocity update equation of chaotic PSO (Eqn. 5.3) is altered by adding the inertia weight as given in Equation (5.9).

$$v_i^{new} = \omega v_i^{old} + c_1 * Chaotic\_operator_1(lbest - x_i) + c_2 * Chaotic\_operator_2\ (gbest - x_i) \qquad (5.9)$$

Here, the inertia weight adaptation is performed based on the equation (5.7), for mining ARs.

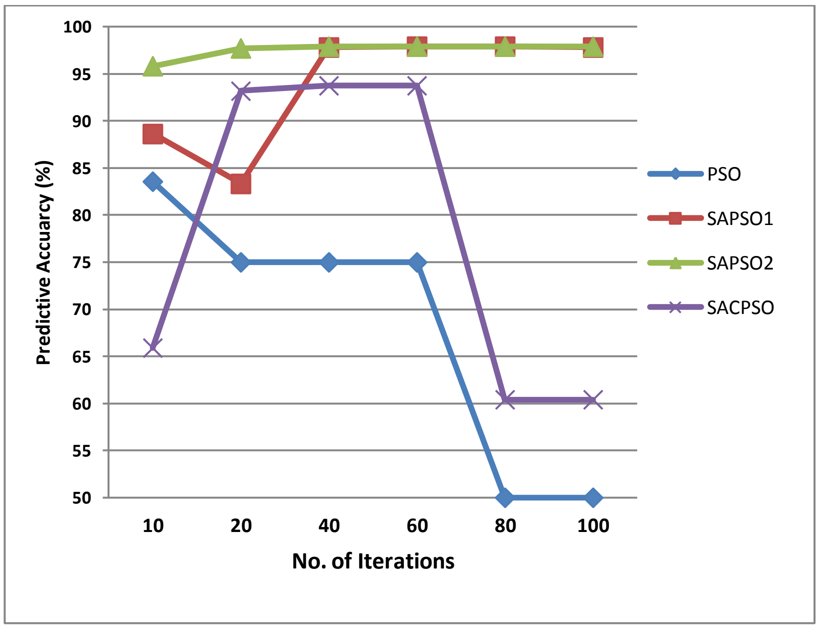
5.6.2 Experimental Results and Discussion

ARM is performed on the five datasets using the three adaptive methodologies discussed in the previous section. The initial parameter setting for the experiment is listed in Table 5.8.

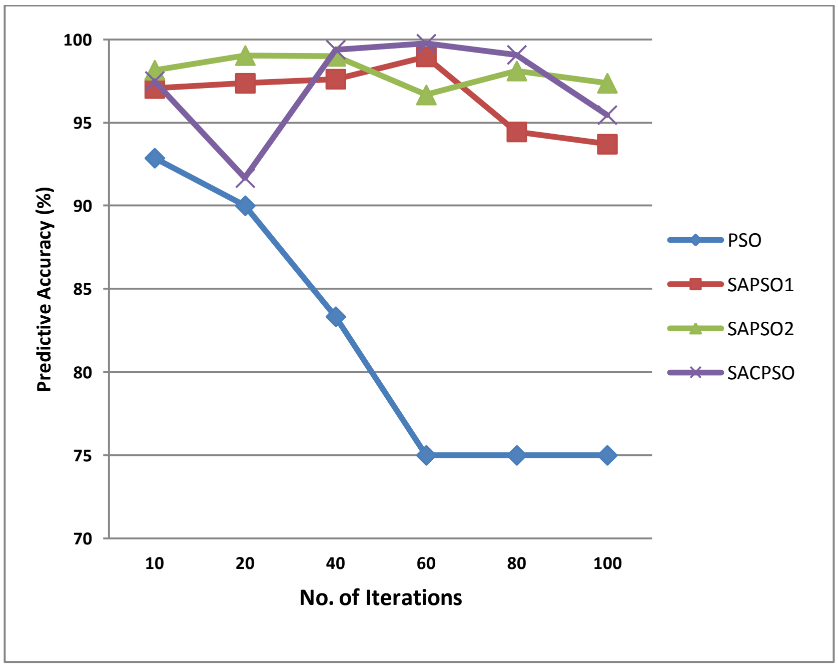
Table 5.8 Default Parameter Setting of Adaptive Methodologies for ARM

Parameter Name	Value
Population Size	Lenses : 20 Haberman's Survival : 300 Car Evaluation : 1000 Postoperative Care : 50 Zoo : 100
Initial Velocity	0
c_1, c_2	2
V_{max}	1
ω_{max}	0.4
ω_{min}	0.9

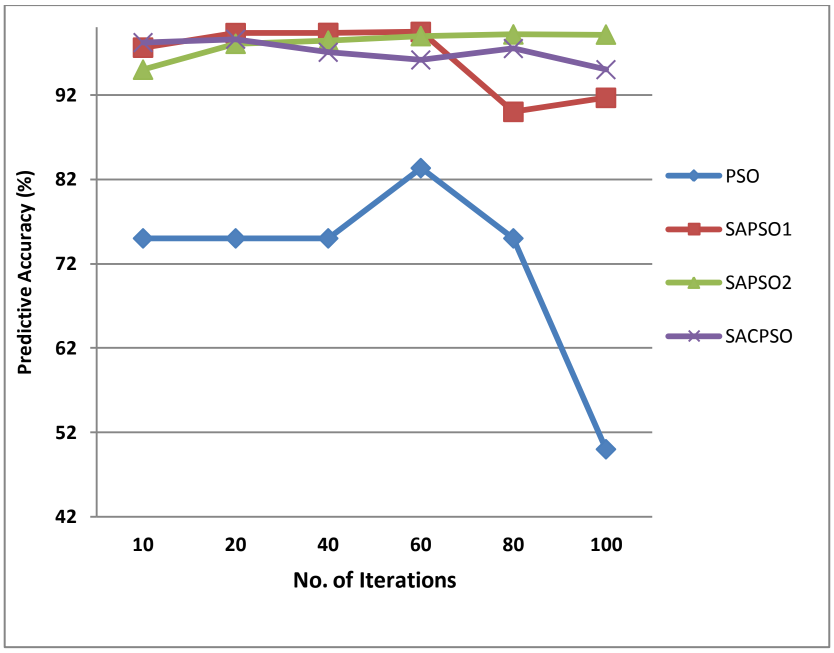
ARs are mined with the three methodologies; SAPSO1, SAPSO2 and SACPSO for the five datasets. The PA achieved by the three methods over generations is plotted individually for the five datasets as given in Figure 5.10 (a-e).



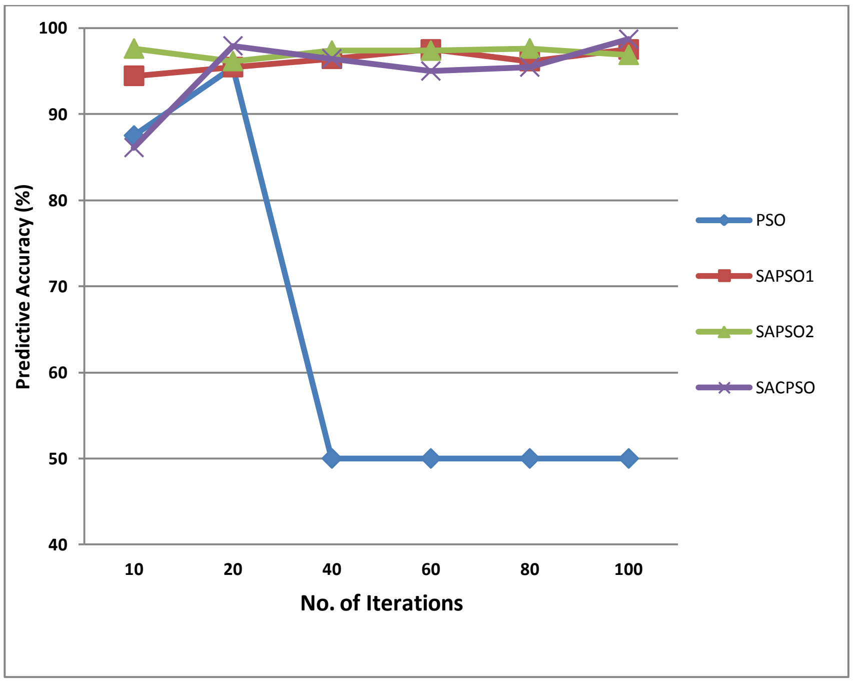
(a) Lenses Dataset



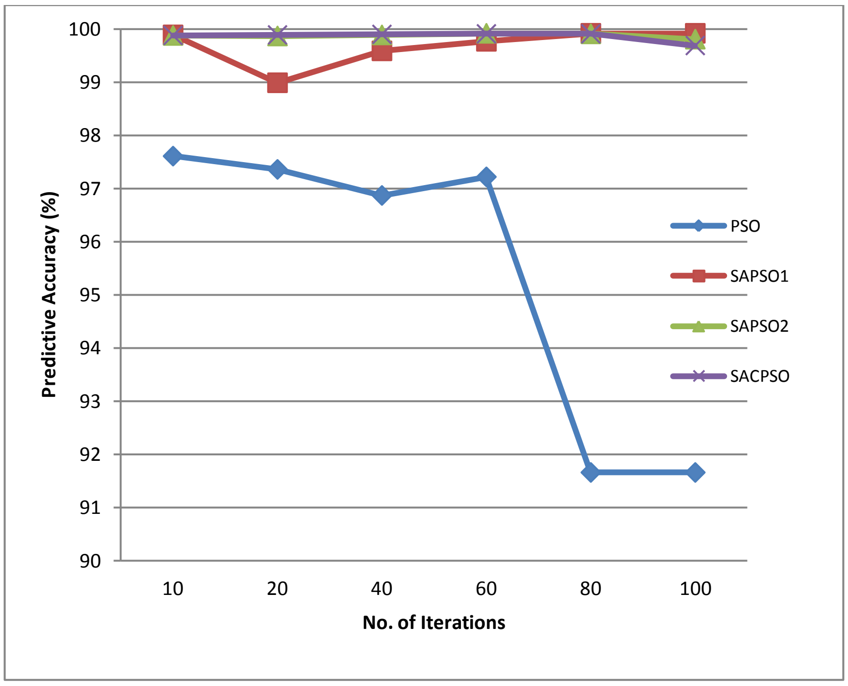
(b) Haberman's Survival Dataset



(c) Car Evaluation Dataset



(d) Postoperative Patient Dataset



(e) Post Operative Patient Dataset

Figure 5.10 Comparison of Predictive Accuracy of three Adaptive Methodologies with PSO for ARM

From Figure 5.10 a to e it is observed that the three adaptation techniques: SAPSO1, SAPSO2 and SACPSO generated ARs with better PA than PSO methodology. SAPSO1 methodology performs marginally better when compared to other two methodologies except for the Haberman's Survival dataset, where SACPSO's PA is better. The performance of SAPSO2 is consistent for all the five datasets throughout the evolution process.

Introduction of adaptation in inertia weight enhances the accuracy of the system considerably. The inertia weight controls the impact of previous flying experience, which is utilized to keep the balance between exploration and exploitation. The adaptation of the control parameters result in enhancement of the accuracy thereby increasing the ability of PSO for mining ARs.

5.7 DATA DEPENDENT ADAPTATION IN PSO

The literature on adaptive strategies in GA and PSO focuses on the mechanisms of adaptation over generation either decreasing or increasing the parameter values, independent of the data involved to enhance the performance. As ARM is mainly based on relationship or correlation between items in a dataset, data dependent strategy of adaptation could generate better associated rules both qualitatively and quantitatively. The performance of the adaptive strategy for mining ARs could be enhanced, if the parameter adjustments are done depending on: i) the data involved and ii) fitness values used for evolution through generations. The proposed ARM based on data dependent adaptation of control parameters in PSO, is discussed in the following section.

5.7.1 Methodology

The roles of the parameters are significant in the performance of the methodologies. PSO is mainly conducted by three key parameters important for the speed, convergence and efficiency of the algorithm (Yao et al. 1999): the inertia weight (ω) and two positive acceleration coefficients (c_1 and c_2). Inertia weight controls the impact of the velocity history into the new velocity. Acceleration parameters are typically two positive constants, called the cognitive parameter c_1 and social parameter c_2.

Inertia weight plays a key role in the process of providing balance between exploration and exploitation process. The linearly decreasing inertia weight over time enhances the efficiency and performance of PSO (Xin et al., 2009). Suitable fine-tuning of cognitive and social parameters c_1 and c_2 may result in faster convergence of the algorithm and alleviate the risk of settling in one of the local minima. To adapt the parameters of PSO in the proposed adaptive PSO methodology for ARM, parameter tuning was performed based on the evolutionary state to which the data is fits. The acceleration coefficients are adapted based on the dataset from which ARs are mined. The inertia weight is adjusted based on the fitness value of the individuals. The pseudo code for the Adaptive PSO is given below:

/* Ns: size of the swarm, C: maximum number of iterations, O_f : the final output*/

```
i.   t = 0, randomly initialize $S_0$,
        •  Initialize $x_i$, $\forall_i$, i ∈ {1, . . .,Ns}      /* $x_i$ : the i^th particle */
        •  Initialize $v_i$, $\forall_i$, i ∈ {1, . . .,Ns}      /* $v_i$ : the velocity of the i^th
                                                   particle*/
        •  Initialize $\omega_1$, $c_1(1)$, $c_2(1)$             /* $\omega$ : Inertia weight, $c_1$, $c_2$ :
                                                   Acceleration Coefficients */
        •  $Pb_i \leftarrow x_i$, $\forall_i$, i ∈ {1, . . .,Ns}         /* $Pb_i$ : the personal best of the i^th
                                                   particle */
        •  $Gb \leftarrow x_i$                               /* Gb : the global best particle */
ii.  for t = 1 to C,                               /* C : total no of iterations */
        •  for i = 1 to $N_s$
           $F(x_i)$ = confidence($x_i$) × log (support ($x_i$) x (length(x) + 1)
                                                  /* $F(x_i)$ : Fitness of $x_i$ */
           If ( $F(x_i) < F(Pb_i)$)
              Pbi ← $x_i$                          /* Update particle best */
           Gb ← min($Pb_1$, $Pb_2$,… , $Pb_{Ns}$)      /* Update global best */
           adjust parameters($\omega(t)$, $c_1(t)$, $c_2(t)$)  /* Adaptive Adjustment */
           $v_i(t) = \omega(t)v_i(t-1) + c_1(t)r1(Pb_i - x_i) + c_2(t)\, r2(Gb - x_i)$
                                                 /* Velocity Updation
                                                r1, r2 : random values*/
           $x_i(t) = x_i(t-1) + v_i(t)$                 /* Position Updation */

        •  A(t) ← non dominated(S(t) ∪ A(t))  /* Updating the Archive*/
iii. $O_f \leftarrow$ A(t) and stop                        /* $O_f$ : Output*/
```

Adjust parameters ((t), c_1(t), c_2(t)**)** in the above pseudo code is achieved through adaptive mechanism newly proposed. The proposed approach based on estimation of evolutionary state distributes the evolution of data into four states, namely, Convergence, Exploration, Exploitation and Jumping out.

Estimation of Evolutionary State (EES)

During the PSO process, the population distribution characteristics vary not only with the generation number, but also with the evolutionary state. For example, at an early stage, the particles may be scattered in various areas, and hence, the population distribution is dispersive. As the evolutionary process goes on, the particles would cluster together and converge to a local or globally optimal area. Hence, the population distribution information would differ from the early stage. The estimation of evolutionary state detects the population distribution information and locates the region in which the particle lies.

Based on the search behaviors and the population distribution characteristics of the PSO, estimation of evolution state is done as follows

a. The distance between particles is calculated using the Euclidean distance measure for each particle i using the Equation (5.10).

$$d_i = \frac{1}{N-1}\sum_{j=1, j\neq i}^{N}\sqrt{(x_j - x_i)^2} \tag{5.10}$$

where, N is the population size, x_i and x_j are the ith and jth particles in the population, respectively.

b. Calculate the evolutionary state estimator (e), defined as

$$e = \frac{d_g - d_{\min}}{d_{max} - d_{\min}} \tag{5.11}$$

where, d_g is the distance measure of the gBest particle, d_{max}, d_{min} are the maximum and minimum distance measures respectively, from step a, above.

c. Record the evolutionary (e) factor for 100 generations for each dataset individually.

d. Classify the estimator (e), to the state it belongs: Exploration, Exploitation, Convergence, Jumping out, for the datasets based on the evolutionary states.

The evolutionary state estimator e over hundred generations is plotted for the zoo dataset as shown in Figure 5.11. The evolution states are estimated based on the e values. The state transition is different for the five datasets and is nondeterministic and fuzzy. Based on the evolution state estimator values, classification is done for determining the intervals at which the class transition occurs.

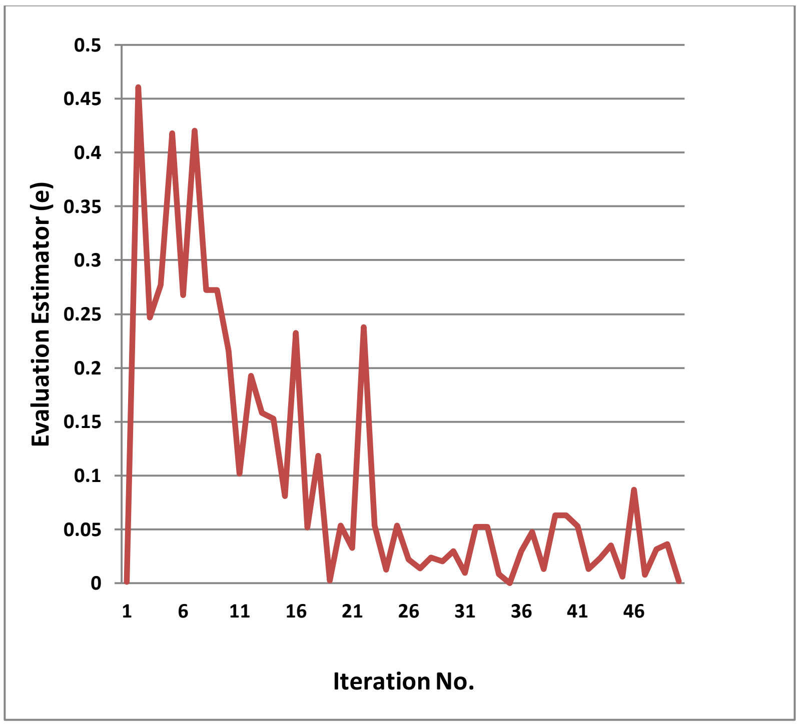


Figure 5.11 Evolutionary State Information Robustly Revealed by *e* at Run Time for Zoo Dataset

The intervals arrived for the five datasets based on EES is given in Table 5.9.

Table 5.9 Classification into States by Evolutionary Factor

Datasets / **EES State**	**Lenses**	**Car Evaluation**	**Haberman's Survival**	**Postoperative Patient**	**Zoo**
Convergence	0.0 – 0.3	0.0- 0.15	0.0 - 0.4	0.0 - 0.5	0.0- 0.15
Exploitation	0.1- 0.4	0.15- 0.25	0.3-0.7	0.2- 0.6	0.1- 0.35
Exploration	0.2- 0.7	0.1- 0.3	0.6- 0.9	0.4-0.8	0.2- 0.4
Jumping out	0.6-1	0.3-1	0.8-1	0.7-1	0.3-1

The change of state reflected as per the PSO sequence is Convergence ⇒ Exploitation ⇒ Exploration ⇒ Jumping Out ⇒ Convergence.

The formulation of numerical implementation of the classification for the Haberman's Survival dataset is as follows:

Case (a) – Exploration: A medium to large value of e represents exploration, whose membership function is defined as:

$$\mu_{exploration}(e) = \begin{cases} 0, & e < 0.6 \\ 5 \times e - 3, & 0.6 < e \leq 0.7 \\ 1, & 0.7 < e \leq 0.8 \\ -10 \times e + 9 & 0.8 < e \leq 0.9 \\ 0, & 0.9 < e \leq 1 \end{cases} \tag{5.12a}$$

Case (b)—Exploitation: A shrunk value of e represents exploitation whose membership function is defined as:

$$\mu_{exploitation}(e) = \begin{cases} 0, & 0 < e \leq 0.3 \\ 10 \times e - 3, & 0.3 < e \leq 0.4 \\ 1, & 0.4 < e \leq 0.6 \\ -5 \times e + 4 & 0.6 < e \leq 0.7 \\ 0, & 0.7 < e \leq 1 \end{cases} \tag{5.12b}$$

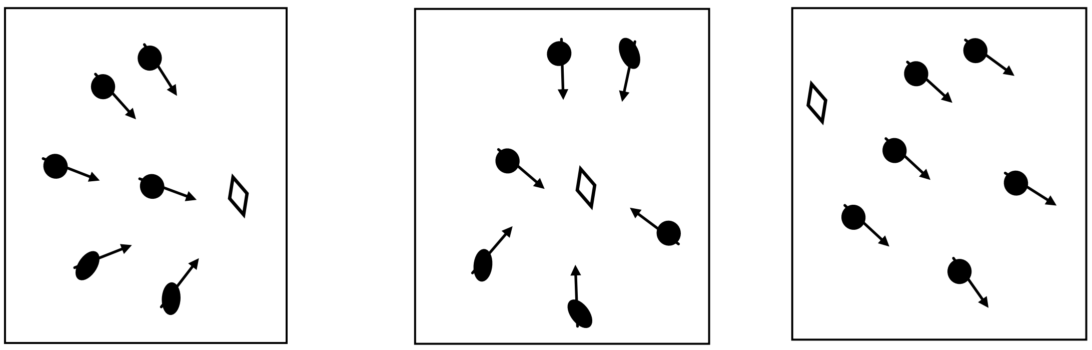
Case (c)—Convergence: A minimal value of *e* represents convergence, whose membership function is defined as:

$$\mu_{convergence}(e) = \begin{cases} 1, & 0 < e \leq 0.3 \\ -5 \times e + 2.5 & 0.3 < e \leq 0.4 \\ 0, & 0.4 < e \leq 1 \end{cases} \tag{5.12c}$$

Case (d)—Jumping Out: When PSO is jumping out of a local optimum, the global best particle is distinctively away from the swarming cluster. Hence, the largest value of *f* reflects *S*4, whose membership function is, thus, defined as:

$$\mu_{jumping\ out}(e) = \begin{cases} 0, & 0 < e \leq 0.8 \\ -5 \times e + 5 & 0.8 < e \leq 0.9 \\ 1, & 0.9 < e \leq 1 \end{cases} \tag{5.12d}$$

Therefore, at a transitional period, two memberships will be activated, and *e* can be classified to either state. The singleton method of defuzzification is adopted. The distribution of all particles in PSO, based on Euclidean distance calculated is illustrated in Figure 5.12.



$d_g \approx d_{pi}$, Exploration $d_g \ll d_{pi}$, Exploitation, Convergence $d_g \gg d_{pi}$, Jumping Out

● – Particle , ◇ - Best particle of the swarm

Figure 5.12 Population Distribution in PSO based on Evolutionary factor e

Based on the estimation into states using evolutionary estimation factor e and fitness values, the adaptation of acceleration coefficients and inertia weight is performed.

Adaptive Control of Acceleration Coefficients

The acceleration coefficients are made adaptive through the classification of evolutionary states. Parameter c_1 represents the “self-cognition” that pulls the particle to its own historical best position, helping in exploration of local niches and maintaining the diversity of the swarm. Parameter c_2 represents the “social influence” that pushes the swarm to converge to the current globally best region, helping with fast convergence. Both c_1 and c_2 are initially set to 2 (Song and Gu 2004; Mendes et al. 2004), based on the published literature. The acceleration coefficients are adaptively altered during evolution, according to the evolutionary state, with strategies developed. The strategy to adopt for the four states is given in Table 5.10. The values for δ and Δ are empirically arrived for the datasets taken up for study.

Table 5.10 Control Strategies for c_1 and c_2

State/Acceleration Coefficient	**c_1**	**c_2**
Exploration	Increase by δ	Decrease by δ
Exploitation	Increase by Δ	Decrease by Δ
Convergence	Increase by Δ	Increase by Δ
Jumping out	Decrease by δ	Increase by δ

The increase or decrease in the values of c_1 and c_2 arrived in Table 5.10 is discussed below.

Exploration: During exploration, particles should be allowed to explore as many optimal regions as possible. This avoids crowding over single optima, probably the local optima, and explores the target thoroughly. Increase in the value of c_1 and decrease in c_2 facilitate this process.

Exploitation: In this state, based on the historical best positions of each particle, they group towards those points. The local information of the particle aids this process. A slight increase in c_1 advances the search around particle best (pBest) positions. At the same time the slight decrease in c_2 avoids the deception of local optima, as the final global position has yet to be explored.

Convergence: In this state, the swarm identifies the global optima. All the other particles, in the swarm should lead towards the global optima region. The slight increase in the value of c_2 helps this process. To fasten up the process of convergence, a slight increase in the value of c_1 is adopted.

Jumping Out: The global best (gBest) particle move away from the local optima towards global optima, taking it away from the crowding cluster. Once any particle in the swarm reaches this region, then all particles are to follow the same pattern rapidly. A large c_2 along with a relatively small c_1 value helps to obtain this goal.

Bounds for Acceleration Coefficient

Adjustments on the acceleration coefficients should be minimum so as to maintain the balance between values c_1 and c_2. Hence, the maximum increase or decrease between two generations is in the range [0.02, 0.1]. The value of Δ is set as 0.02 based on trials and the value for δ is set as 0.06 similarly. c_1 and c_2 values are clamped in the interval [1.5, 2.5] in order to maintain the diversity of the swarm and fast convergence (Carlisle and Dozier 2001). The sum of the acceleration coefficients is limited to 4.0 (Zhan et al. 2007), when the sum exceeds this limit then both c_1 and c_2 are normalized based on equation (5.13).

$$c_i = \frac{c_i}{c_1 + c_2} 4.0, \qquad i = 1,2. \tag{5.13}$$

The adaptation behavior of the acceleration coefficients c_1 and c_2 through change of states, in estimation of evolution state for Zoo dataset is shown in Figure 5.13.

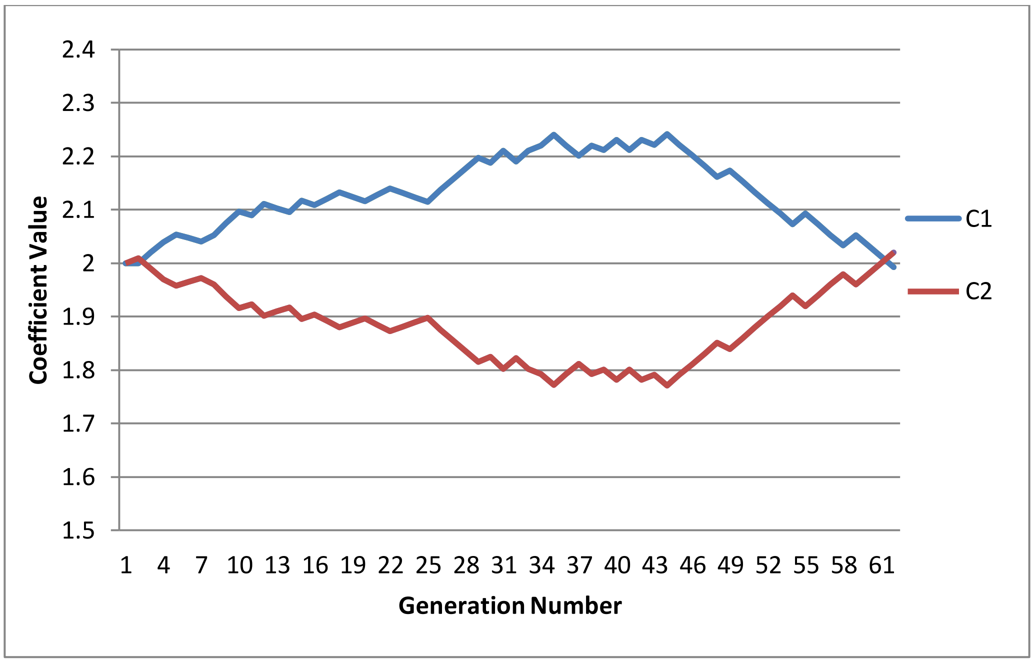


Figure 5.13 Adaptation of Acceleration Coefficients through EES for Zoo Dataset

Inertia Weight Adaptation

The inertia weight controls the impact of previous flying experience, which is utilized to keep the balance between exploration and exploitation. The particle adjusts its trajectory according to its best experience and enjoys the information of its neighbors. In addition, the inertia weight is also an important convergence factor; smaller the inertia weight, faster the convergence of PSO. A linear decrease in inertia weight gradually may swerve the particle from their global optima (Chatterjee and Siarry 2006). Hence, a nonlinear adaptation of inertia weight as proposed in the given Equation (5.14), is considered as the solution to the above issue. The global best particle is derived based on the fitness value of the particles in the swarm. The proposed methodology for adopting the inertia weight is based on the fitness values exhibited by the particles.

$$\omega = \begin{cases} \omega_{max}, & f \geq f_{avg} \\ \omega_{max} - \frac{(f_{max} - f)(\omega_{max} - \omega_{min})}{f_{max} - f_{min}}, & f_{avg}/2 < f \leq f_{avg} \\ \omega_{min} + i\,\frac{(\omega_{max} - \omega_{min})}{G}, & f < f_{avg}/2 \end{cases} \tag{5.14}$$

where, f, f_{min} and f_{avg} are the fitness values, which are decided by the current particles in the swarm. The minimum fitness value of all particles in swarm is defined as f_{min}, while the mean value is selected as f_{avg}. The adaptation of inertia weight for zoo dataset over generations is given in Figure 5.14.

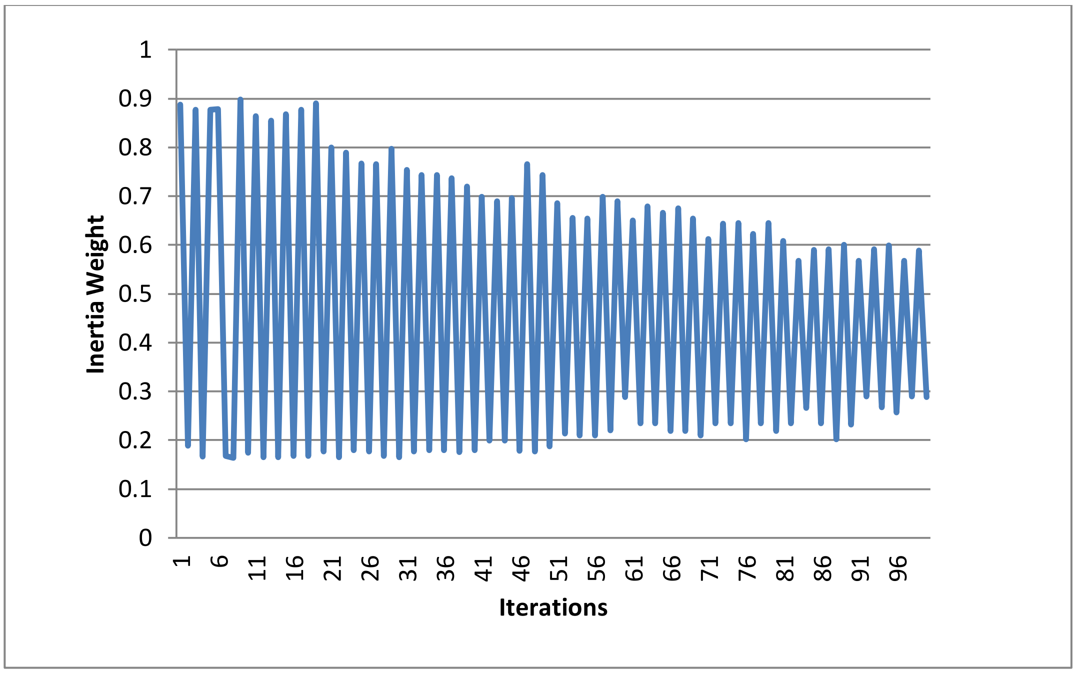


Figure 5.14 Inertia Weight Adaptation for Zoo dataset

From figure it can be concluded that in general, the change in inertia weight decreases linearly. In detail, it is adapted based on the fitness information dynamically. The value of inertia weight swings around 0.461 frequently for all generations. With the above value of inertia weight, the convergence ability in the search space, to avoid the risk of being trapped into local minima, and improve the diversity of PSO.

5.7.2 Experimental Results and Discussion

The objective of the adaptive PSO is to enhance the performance of ARM by making the adaption dependent on dataset used. The proposed methodology was validated using the bench mark datasets chosen for validating the experiment.

Five datasets were used for analyzing the performance of APSO and comparing with that of simple PSO. The parameter setting for the APSO methodology is given in the Table 5.11.

Table 5.11 Parameter Setting for APSO

Parameter Name	Value
Population Size	Lenses : 20 Haberman's Survival : 300 Car Evaluation : 1000 Postoperative Care : 50 Zoo : 100
No. of Generations	100
p_m^0	0.3
p_c	0.3
λ	0.25
c_1 , c_2	2
ω_{max}	0.9
ω_{min}	0.4

The scope of this study on ARM using adaptive PSO is to

- Mine ARs with adaptive PSO with multiobjective, namely, accuracy and significance of the rules generated;
- Compare the performance with simple PSO (non adaptive methodologies);
- Analyze the performance over generations and
- Convergence analysis of APSO in terms of execution time

The APSO methodology was applied on the five datasets to mine ARs and the PA was recorded. The results are compared with that of simple GA discussed in chapter 4 and PSO discussed earlier in this chapter.

The APSO methodology for mining ARs adapt the acceleration coefficients based on the estimation of evolutionary state. Based on the fitness values the inertia weight is adapted. Then, the velocity modification is based on the state in which the particle lies. This acts as a balance between exploration and exploitation, thus escaping from premature convergence. The PA of the ARs mined is improved through the evolution process.

The PA for the adaptive PSO methodology over 100 generations, is shown in Figure 5.15. The adaptive PSO's performance on Car Evaluation dataset and Zoo data set is consistent. The performance on Haberman's survival dataset and Post operative patient dataset, is maximum at the end of evolution. This avoids premature convergence at initial stages. For the Lenses dataset where the dataset size is small, global search space is maintained effectively making the convergence possible, even at early iterations.

The Laplace, Conviction, Leverage and Lift measures of the five datasets for the adaptive PSO is recorded as given in Table 5.12.

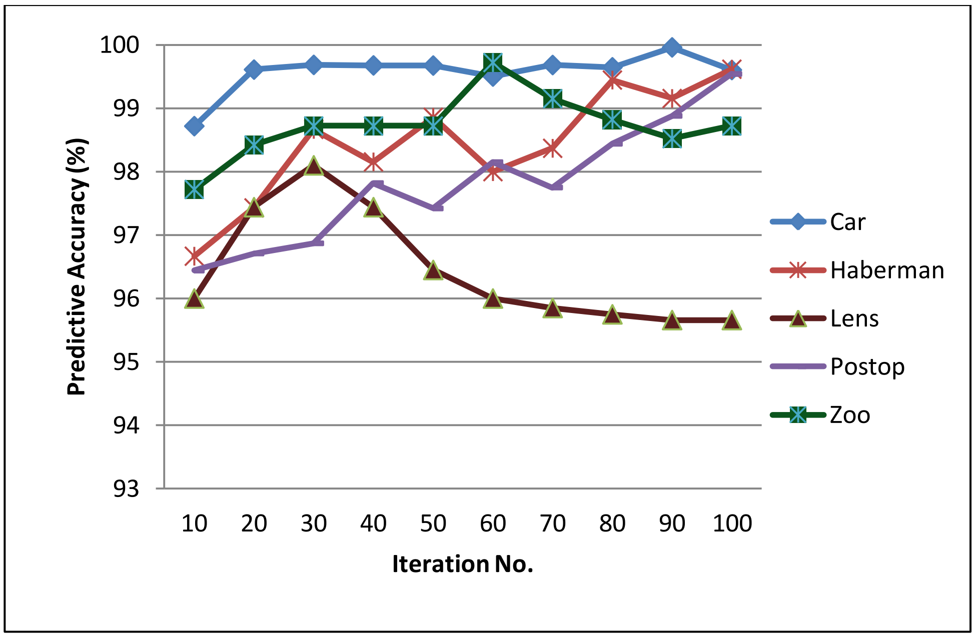


Figure 5.15 Predictive Accuracy of Adaptive PSO over Hundred Generations

Table 5.12 Multiobjective Measures of APSO

Measures	**Lenses**	**Haberman's Survival**	**Car Evaluation**	**Postoperative Patient**	**Zoo**
Laplace	0.52941	0.501608	0.502488	0.5028	0.5024
Conviction	Infinity	Infinity	Infinity	Infinity	Infinity
Leverage	0.026	0.003039	0.002394	0.0301	0.0249
Lift	1.548	2.0458	1.9875	2.2478	2.4558

When the Laplace measure is away from 1, it indicates that the antecedent values are dependent on the consequent values, and hence the rules generated are of significant. The conviction measure which is infinity for all datasets show that the rules generated are interesting. The Leverage measure being far away from 1 again insists on the interestingness of the rules generated. The measure of Lift for all the datasets, greater than1 signifies the dependency of the consequent on the antecedent. The rules generated by Adaptive PSO are of importance based on the Lift measure.

The PA achieved from the adaptive PSO methodology is compared with that of simple PSO, as shown in Figure 5.16. The Adaptive PSO performs better than the simple PSO for all the five datasets, when applied for ARM.

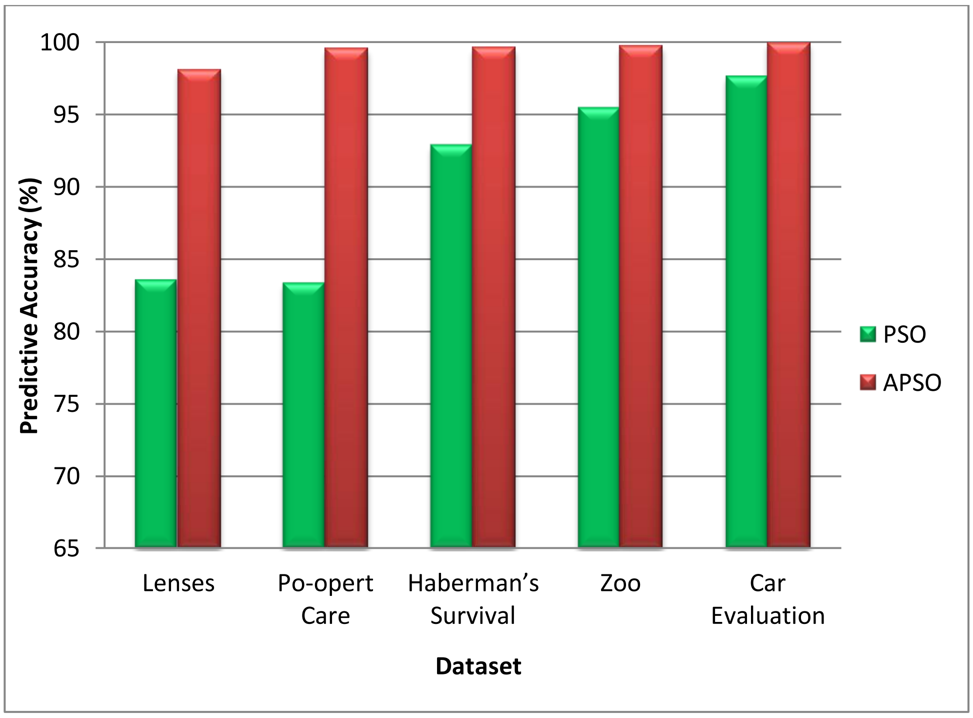


Figure 5.16 Predictive Accuracy Comparison of APSO with PSO

The number of rules generated by APSO with PA less than 0.05 percentage of the maximum PA arrived, is plotted for mining ARs against the simple PSO methodology for the datasets in Figure 5.17. In terms of number of rules generated adaptive PSO performs better for Hagerman's Survival, Zoo and car evaluation datasets. The number of rules generated for lens and postoperative patient dataset is marginally better than that of simple PSO. As the number of instances in both datasets is small, the increase in the number of rules is found to be minimum.

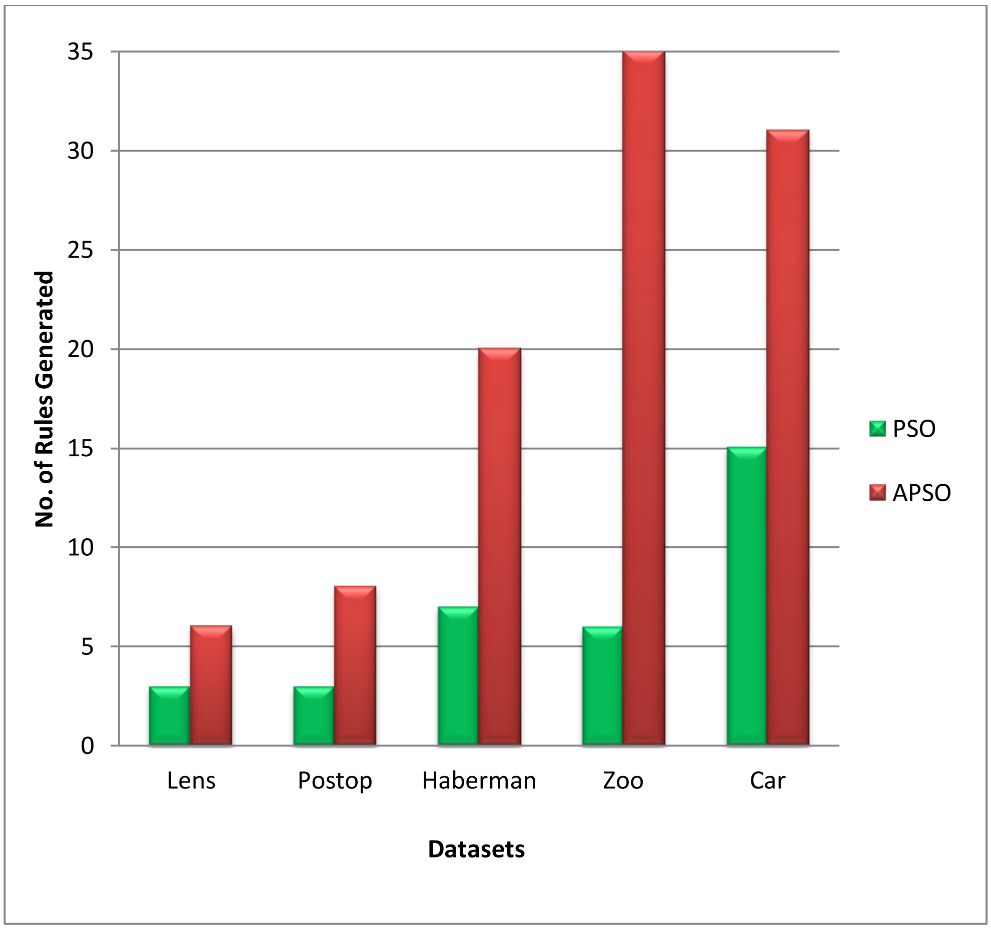


Figure 5.17 Comparison of Number of Rules Generated by APSO and PSO

The execution time of APSO for mining ARs from the datasets in comparison with PSO methodology is given in Figure 5.18. This is the time taken by the system to generate ARs (CPU time). The number of iterations when the PA is at maximum, is taken as point at which the execution time is recorded. The execution time of the APSO for mining ARs is more, when compared to PSO. The adaption mechanism of acceleration coefficients based on evolution factor, and inertia weight adaptation increases the complexity of the algorithm, resulting in higher execution time when compared to PSO.

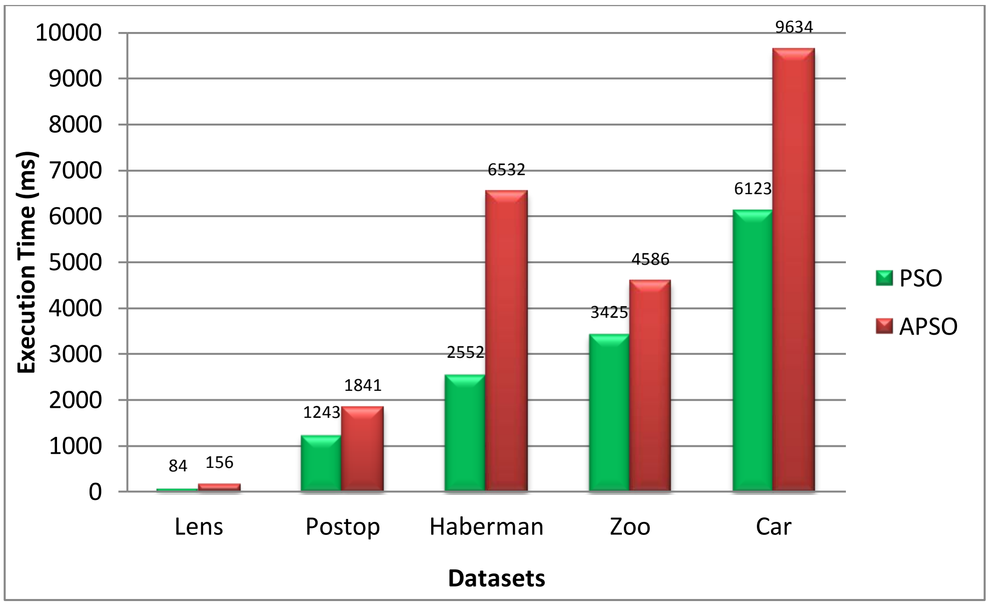


Figure 5.18 Comparison of Execution Time of APSO and PSO

When number of instances in the dataset is less then increase in execution time is found to be marginal. Thus, the difference in the execution time for mining rules for the Lenses, Postoperative Patient and Zoo datasets, are minimum for APSO than PSO. For the Haberman's survival dataset with moderate number of instances, there is a noticeable increase in execution time, whereas, for the car evaluation dataset, the difference in execution time is large. The consistent performance of the APSO in terms of PA throughout evolution and the PA achieved from APSO for mining ARs, balances the difference in the execution time.

The major drawback of the simple PSO is its premature convergence, where, the particle fixes some local optima as the target (global search area), where all particles converge locally. One of the objectives of Adaptive PSO is to avoid the premature convergence, thereby striking a balance between exploration and exploitation. The iteration number at which PA is high is plotted for both APSO and PSO methodologies, for mining ARs for the datasets used in Figure 5.19.

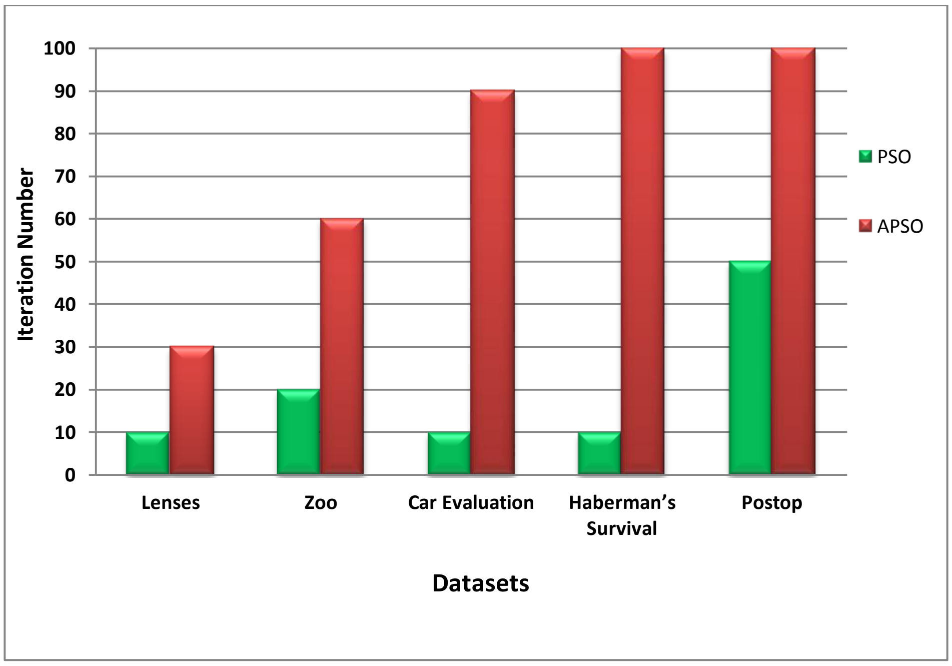


Figure 5.19 Comparison of Convergence of APSO and PSO

The performance of APSO is compared with SAPSO1 and SAPSO2 as given in Figure 5.20 and it is noted that the data dependent methodology of mining ARs performs better than data independent methodologies in terms of PA.

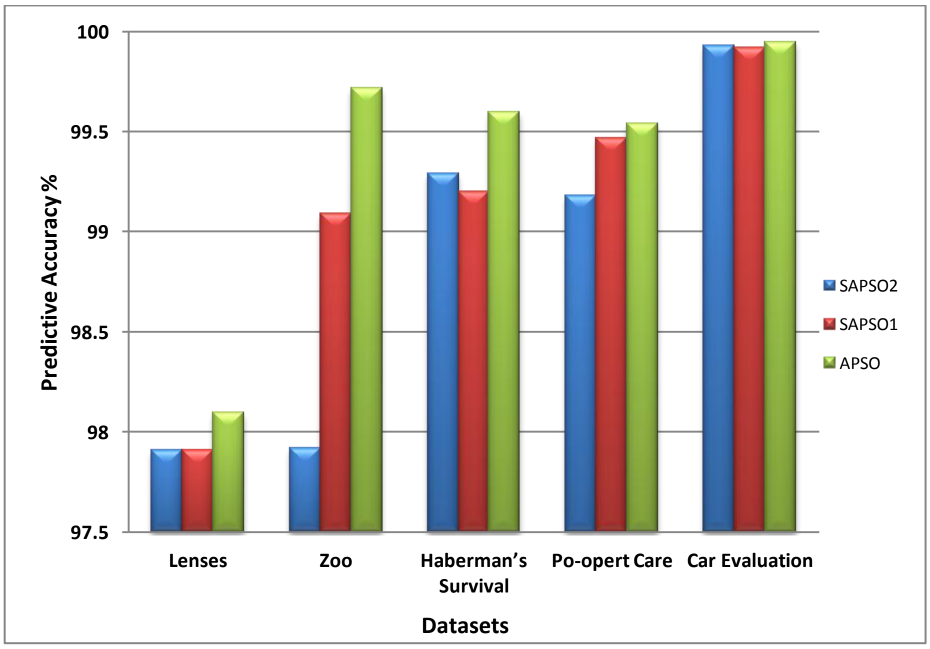


Figure 5.20 Comparison of Convergence of APSO and SAPSO

The analysis of the performance of the five datasets through the adaptive methods of PSO has been compared with simple PSO. The Adaptive PSO generated ARs with better PA and qualitative measures. To test the significance of the proposed APSO, it is compared with Ant Colony Optimization (ACO) (a similar population based search method) for mining rules. Four more datasets from UCI repository for which ARM using ACO has been carried out, have been chosen from published literature. The PA of the mined rules using APSO for these datasets was recorded. The comparative results obtained by AGA, APSO and ACO are given in Table 5.13.

Table 5.13 Predictive Accuracy Comparison APSO with AGA and ACO

Dataset	**Adaptive GA**	**Adaptive PSO**	**Lale et al. 2010**	**Parpinelli et al. 2002**	**Liu et al. 2003**	**Lale et al. 2008**
Lenses	87.5	98.1	-	-	-	-
Haberman's Survival	68	99.6	-	-	-	-
Car Evaluation	96	99.95	-	-	-	-
Postoperative Care	92	99.54	-	-	-	-
Zoo	91	99.72	-	-	-	-
Iris	96	98.86	98	-	-	98.67
Nursery	97.30	99.12	97.24	-	-	97.16
Tic Tac Toe	88.36	98.76	-	75.57	76.58	97.59
Wisconsin Breast Cancer	92.54	98.75	-	96.97	94.32	-

Note : - Data not available in study

The adaptive PSO methodology generates ARs with better PA when compared to other methodologies analyzed. The adaptive PSO methodology when applied for mining ARs generates rules with enhanced PA, increase in the number of rules generated, along with better rule set measures. The increase in the execution time of APSO when compared to simple PSO methodology is also minimum taking into account the complexity of the methodology and the increase in accuracy achieved over PSO.

The balance in setting the global search space between exploration and exploitation is also attained, while mining ARs with APSO, for all the datasets used. This is noted from the shift in iteration number of APSO when compared to simple PSO, where highest PA is noted. Adopting the acceleration coefficients c_1 (cognitive factor) and c_2 (social factor) through EES methodology, makes the adjustments in fixing up the global search space effectively with the help of local optima. The fitness value of the particles is used in adapting inertia weight. This helps in better balance between exploration and exploitation when particles fly through the search space.

The comparison of PA for mining ARs is between the methods proposed based on PSO in given in Table 5.14. From the table it is clear that the APSO methodology performs better in terms of PA among the other methods for ARM. Thus the data-dependent adaptation of control parameters maintains the diversity of the population effectively, thereby resulting in optimal PA.

Table 5.14 Predictive Accuracy Comparison of PSO based Methods

Dataset	PSO	CPSO	NPSO	WPSO	SAPSO2	SAPSO1	SACPSO	APSO
Lenses	83.57	87.5	93.1	97.91	97.91	97.91	97.82	98.1
Car Evaluation	97.61	99.86	97.1	99.93	99.93	99.92	99.91	99.95
Haberman's Survival	92.86	96.15	92	99.48	99.29	99.2	99.74	99.6
Po-opert Care	83.33	92.86	94.5	99.29	99.18	99.47	98.61	99.54
Zoo	95.45	94.44	96.5	96.67	97.92	99.09	98.69	99.72

The APSO methodology of mining association rules resulted in maximum PA among the modifications introduced in PSO. The data dependent adaption of control parameters enhanced the performance of PSO. The other adaptive methods proposed namely SAPSO1, SAPSO2, SACPSO also generates association rules with higher Predictive accuracies the reason again being adaptive the parameters involved throughout evolution. WPSO generated association rules with next level of PA due to introduction of inertia weight. CPSO and NPSOs performance is better than PSO in terms of PA. It could be noted that the parameter tuning methods performs better than modifications introduced in velocity displacements.

5.8 SUMMARY

PSO algorithm with and without inertia weight was applied for mining ARs. Based on the results two new proposals for ARM were made namely; Chaotic PSO and Dynamic Neighborhood PSO. These proposed methodologies generated ARs with better PA and quantitative measures. In data independent adaptation three different methodologies are proposed for mining ARs. Among SAPSO1, SAPSO2 and SACPSO the SAPSO1 performs better in terms of PA of the AR mined. The proposed APSO methodology is dependent on data for adjusting the control parameters, and the generated ARs with better PA and in terms of number of rules generated.

CHAPTER 6

HYBRID AND MEMETIC METHODS FOR MINING ASSOCIATION RULES

6.1 GENERAL

This chapter reviews the necessity of hybridizing GA and PSO methods, salient aspects of GA and PSO hybrid methods as available in published literature, and proposes a new hybrid methodology of GA and PSO for ARM. As the main drawback of PSO being weak local search, it is overcome by proposing Shuffle Frog Leaping Algorithm for the local search to mine ARs. The experimental results of both proposals are also presented, and discussed along with the salient inferences there from.

6.2 NEED FOR HYBRIDIZATION

The PSO was inspired by insect swarms and has been proven as a competitor to the standard GA for function optimization. Since then several researchers have analyzed the performance of the PSO with different settings, e.g., neighborhood settings (Kennedy1999; Suganthan1999). Comparisons between PSOs and the standard GA were done analytically in (Eberhart and Shi1998) and also with regard to performance in (Angeline 1998). It has been pointed out that the PSO performs well in the early iterations, but has problems reaching a near optimal solution in several real-valued function optimization problems (Angeline 1998). Both Eberhart and Angeline conclude that hybrid models of standard GA and PSO could lead to further advances.

6.3 HYBRID OF GA AND PSO FOR ARM

ARM using either GA or PSO methodology often generates rules with high PA. But balancing between exploration and exploitation is often a tailback. GA has good exploration capability, but at times it leads to exploiting the search space. PSO has the capability to converge quickly thus avoiding exploitation but results in premature convergence. Hybrid system combining GA and PSO is a solution for the balancing phenomena.

Recently the hybridization between evolutionary algorithms (EAs) and other metaheuristics has shown very good performance in many kinds of multi-objective optimization problems, and thus has attracted considerable attention from both academic and industrial communities.

An evolutionary circle detection method based on a novel Chaotic Hybrid Algorithm combines the strength of PSO, GA and chaotic dynamics. It involves the standard velocity and position update rules of PSO, with the ideas of selection, crossover and mutation from GA (Chun-Ho at al., 2010). A hybrid multi-objective evolutionary algorithm incorporating the concepts of personal best and global best in PSO and multiple crossover operators to update the population, maintains a nondominated archive of personal best (Tang and Wang, 2012). A hybrid method combining GA and PSO creates individuals in a new generation not only by crossover and mutation operations as found in GA, but also by mechanisms of PSO. Further the above approach solves the problem of local minimum of the PSO, and has higher efficiency of searching global space (Nie Ru and Yue Jianhua 2008).

The hybridization methods of GA and PSO analyzed here are process based, integrating the GA steps into PSO or vice versa. Population based hybridization is attempted in this study, where the population is split based on fitness values and both methods are run individually on the respective subpopulation. The objective of this hybrid methodology is to sum up the advantages of both GA and PSO in overcoming the drawbacks.

6.3.1 GA-PSO Hybrid Algorithm for ARM

Each evolutionary algorithm proposed in the literature has some advantage over the other. PSO provides a faster convergence than GA. A hybrid technique utilizing the effectiveness and uniqueness of these two algorithms can be implemented to achieve high performance (Dong et al., 2012). The proposed hybrid methodology that is the GPSO proposed combines the features of GA and PSO to enhance the performance of ARM. The uniqueness of GA and PSO are

- The optimization process of GA is a steady state process converging at global optima.
- The global search space is maintained by GA avoiding premature convergence.
- PSO is faster a process with the ability to search the solution space quickly
- PSO keeps track of each particles’ best position thereby effectively fixing up the search space

The hybrid methodology employs these features for evolution of individuals over generations. The total population is split into two subpopulations based on fitness values. The fitness function designed is for maximization and the individuals with maximum fitness values are aimed to evolve towards the global optima. So GA is applied on the subpopulation with higher fitness values. PSO with the ability to converge quickly is applied on the lower ranked subpopulation based on fitness values. The GA with optimization at global optima enhances the exploration capability and PSO with easy convergence avoids exploitation. Thus the hybrid methodology effectively balances between exploration and exploitation thereby resulting in ARs with better PA. The upper and lower ranked subpopulations after evolution through GA and PSO are updated at the end of each iteration. This aims at consistency in performance, avoiding the loss of the better individuals over generations.

Applying GA on the upper ranked subpopulation avoids premature convergence. Evolution through PSO on lower ranked subpopulation avoids exploitation of GA of individuals, thereby increasing the speed of evolution process. The proposed hybrid model is shown in Figure 6.1.

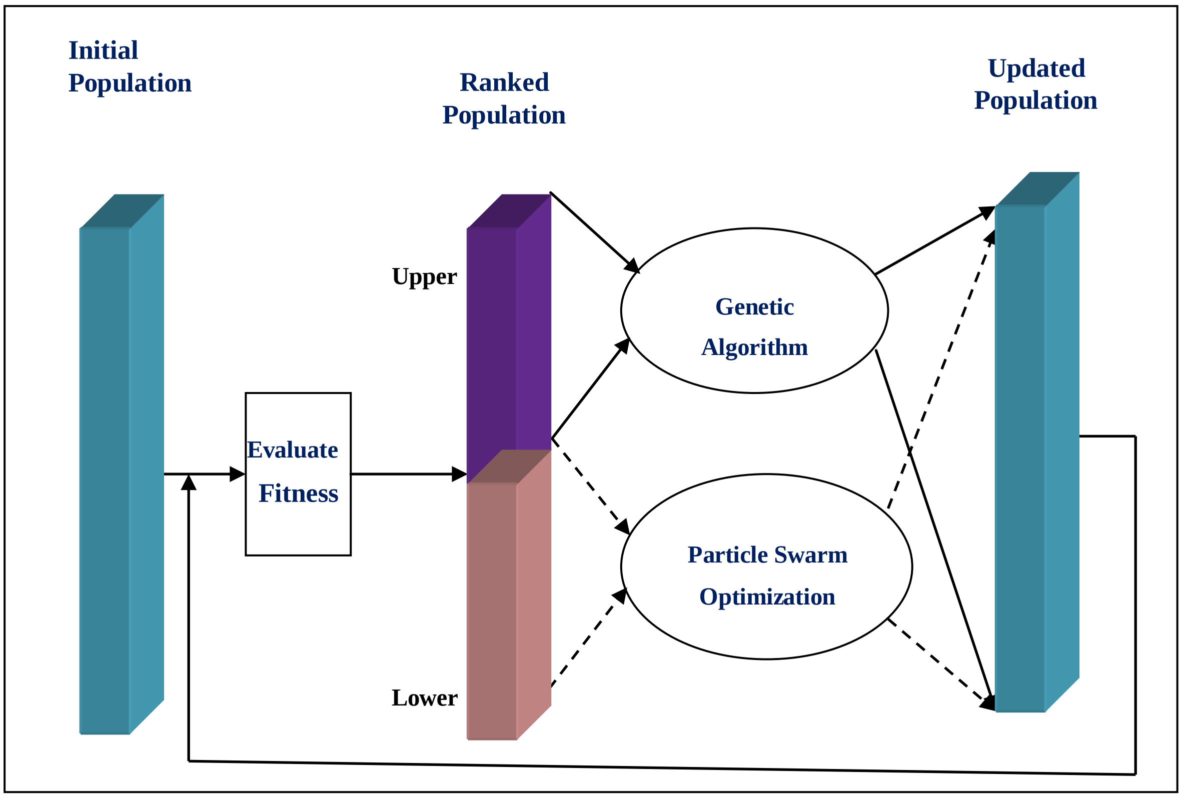


Figure 6.1 Hybrid GA/ PSO (GPSO) Model

As can be seen in Figure 6.1, GA and PSO both work with the same initial population. The hybrid approach takes N individuals that are randomly generated. These individuals may be regarded as chromosomes in the case of GA, or as particles in the case of PSO. The N individuals are sorted by fitness, and the upper half individuals are fed into the GA to create new individuals by crossover and mutation operations. The lower half of the population sorted through fitness values are fed into PSO for evolution. The velocity update and position update are done based on the personal best and global best positions determined on the lower half individuals. The output of GA and PSO are combined for propagating into next generation and the process repeated till an end criterion is met.

By performing evolution based on GA on the upper half of the individuals sorted through fitness value the global optimal solution space is retained. This facilitates the steady progress towards the optimization through exploration. PSO applied on the other half of the individuals where fitness value is low, the convergence is achieved easily avoiding exploitation. Thus, the hybrid GPSO model balances between exploration and exploitation by combining the strengths of GA and PSO. The pseudo-code for GPSO is given below:

```
Initialize all GA variables
Initialize all PSO variables
Repeat
  Calculate fitness value of the population
  Rank the population based on fitness function
  Split the population into two halves: higher ranked range, lower
  ranked range
  On higher ranked range partition perform GA
```

```
    [Selection] Select two parent chromosomes from the population
        according to their   fitness
    [Crossover] With a crossover probability cross over the parents to
        form a new  offspring
    [Mutation] With a mutation probability mutate new offspring at
        each locus
    [Updation] Place the resulting new offspring in a new population
    On low ranked partition perform PSO
        For each particle
                Update velocity
                Update position
                Update pBest and gBest
        End for
    Update new population combining PSO particles and GA
    chromosomes
  Until (stopping condition)
```

The population size is fixed based on the size of the dataset for which ARM is applied. Binary encoding is adopted for representation of data. The fitness function as given in Equation (4.2) is adopted for calculating the fitness values. The experimental setting and results of the GPSO methodology for mining ARs are presented in next section.

6.3.2 Experimental Results and Discussion

To test the performance of the hybrid GPSO for mining ARs, experiments were carried out on the well-known benchmark datasets from UCI repository.

The parameters, which play a major role during the rule discovery in the hybrid GPSO methodology, are listed in Table 6.1. The population size is the size of the individuals taken up for experimentation. The crossover and mutation rates are the GA operator specifications arrived from parameter tuning analysis carried out in section 4.4. c_1 and c_2 are the acceleration coefficients used in velocity updation of PSO as in Equation (3.2), set to default value 2 as seen from literature review.

Table 6.1 GPSO Parameter for ARM

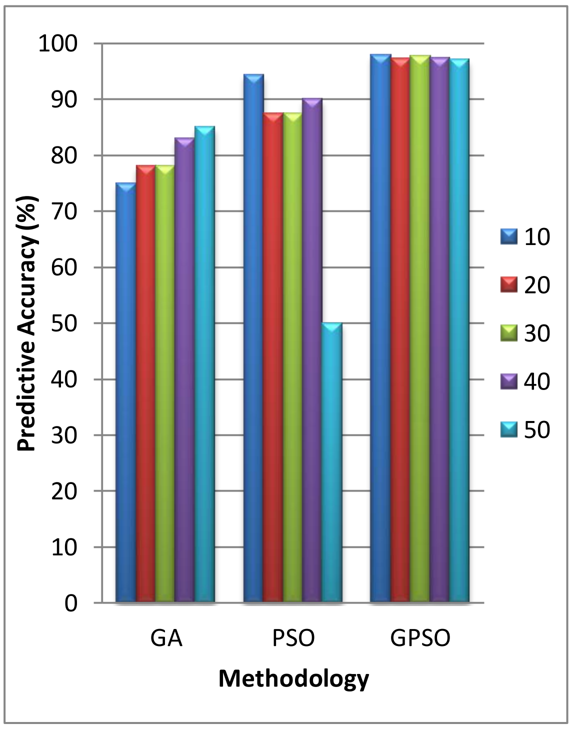
Parameter Name	Value
Population Size	Lenses : 20 Car Evaluation : 700 Haberman's Survival : 300 Post-operative Patient Care : 80 Zoo : 100
Crossover Rate	Lenses : 0.6 Car Evaluation : 0.7 Haberman's Survival : 0.75 Post-operative Patient Care : 0.8 Zoo : 0.8
Mutation Rate	Lenses : 0.5 Car Evaluation : 0.4 Haberman's Survival : 0.25 Post-operative Patient Care : 0.2 Zoo : 0.2
Selection Operation	Roulette wheel selection
c_1	2
c_2	2
No. of Generations	50

Evolutionary algorithms are relatively simple to implement, robust and perform very well on a wide spectrum of problems. This study proposes a hybrid methodology of evolutionary algorithms: GA and PSO for ARM. The scope of this study on mining ARs using GPSO is to:

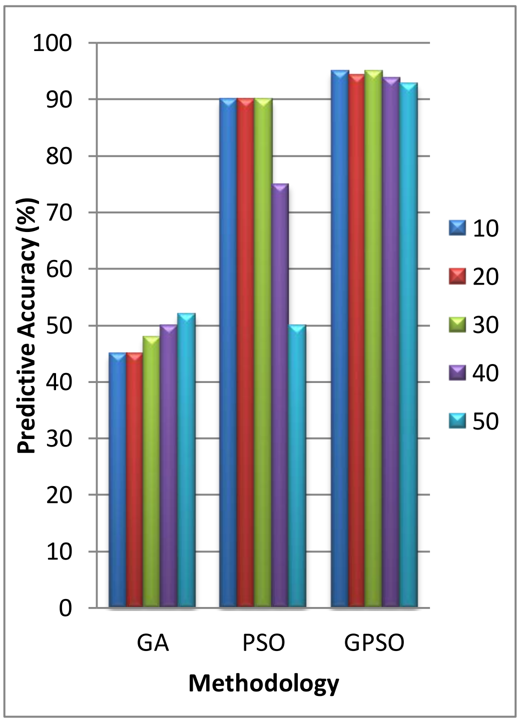
- Study the performance of GA over generations
- Analyze the performance of PSO over generations
- Identify the limitations of GA and PSO while mining ARs in terms of PA and execution speed.
- Compare the performance of GPSO with GA and PSO

Evolution over Generations Analysis

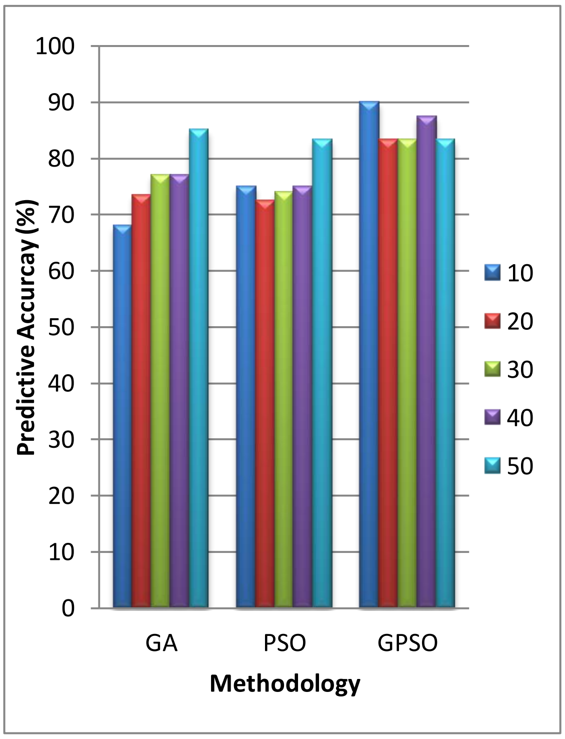
GA is known for maintaining the global optima throughout evolution and steady progress in performance over generations, whereas, PSO converges quickly with chances of converging at local optima. So to analyze the performance of GPSO over generations, the maximum PA of the ARs mined by GPSO is recorded at intervals over evolution for all the five datasets. This data is plotted against the results obtained with GA and PSO as shown in Figure 6.2.



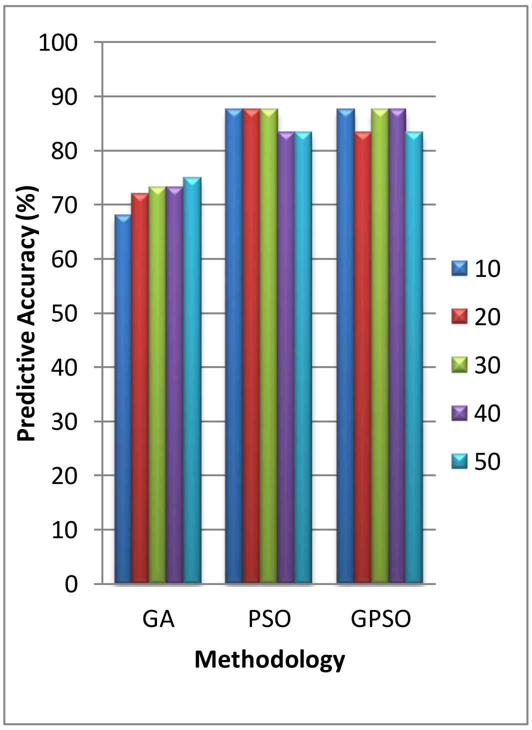
(a) Car Evaluation Dataset



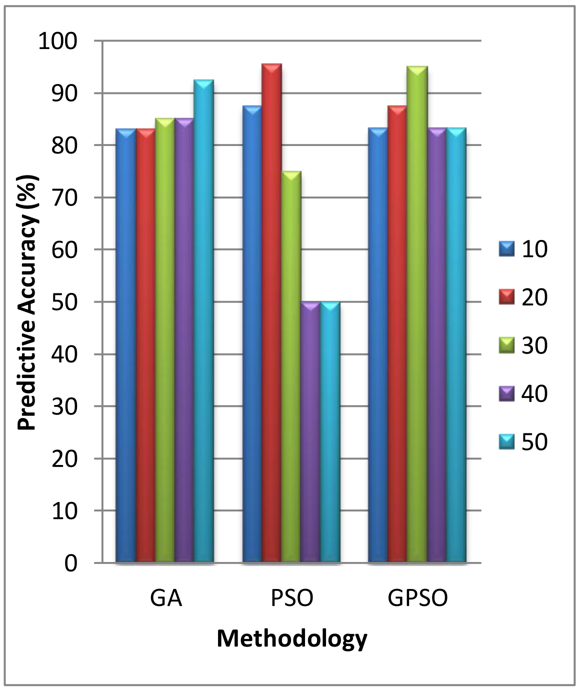
(b) Haberman's Survival Dataset



(c) Lenses Dataset



(d) Postoperative Patient Dataset



(e) Zoo Dataset

Figure 6.2 Predictive accuracy vs. Number of Generations for ARM with GPSO

From Figure 6.2 it is observed that mining ARs using GA results in lesser PA than PSO and GPSO. The increase in PA is achieved steadily over generations. The PSO methodology for mining ARs generates AR with better PA at earlier stages of evolution. In further generations, the particles move away from global optima bringing down the accuracy, thus exploiting the search space. For all the five datasets, GPSO methodology generates ARs with better PA and maintains the same over generations, thereby balancing between exploitation and exploration. Thus the stability in performance is obtained while mining ARs with GPSO.

From the Figure 6.2 a to e it is observed that

- The performance of GA in terms of PA increases over generations indicating its effectiveness in terms of global search capability;
- PSO generates maximum accuracy at initial generations thereby converging quickly;
- The deviation from global optima and convergence at local optima is attained in later generation for PSO. This exploitation of search space results in reduction of PA;
- GPSO produces consistent PA with minimal difference over generations and
- The PA of GPSO is better than GA and PSO.

Predictive Accuracy Analysis

The objective of the study is to enhance the PA of ARs mined by utilizing the uniqueness of both GA and PSO. The PA of the ARs mined by GPSO is plotted for the five datasets as shown in Figure 6.3. The results obtained using simple GA and PSO are also shown in same figure for comparison.

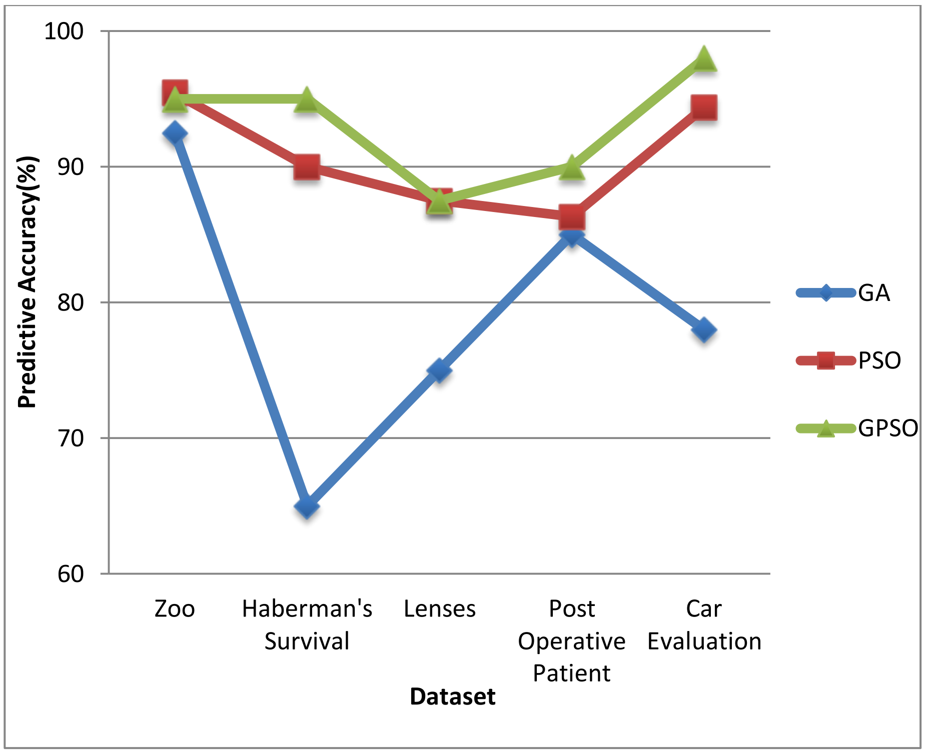


Figure 6.3 Comparison of Predictive Accuracy of GPSO with GA and PSO

The PA of rules mined with PSO is better than GA. The PA obtained by GPSO is enhanced when compared to GA or PSO. The PA for the Lenses and Zoo datasets are equivalent to that of PSO, but better than GA. Increase in PA upto 5% is achieved over PSO by GPSO. An increase of 30% is obtained by GPSO over GA for Haberman's survival dataset. The difference in accuracy achieved by GPSO over GA and PSO is noticeable for datasets of larger sizes, than smaller ones. Applying GA for higher ranked population enhances the optimization through global search and applying PSO for lower ranked population balances between exploration and exploitation, resulting in enhanced performance of GPSO.

Analysis of Execution Time

ARM with GPSO performs both GA and PSO operations on their respective subpopulation. The effect of GPSO on execution time while mining ARs is shown in Figure 6.4. The execution time of GPSO is better than genetic algorithm and takes more time than that of PSO. The evolution of the lower ranked population by PSO makes the convergence quicker, thereby reducing the execution time achieved compared to GA. The GA operations take more time and are complex, when compared to PSO. Thus there is an increase in execution time of GPSO over PSO. The increase in prediction accuracy over PSO compensates the time trade-off.

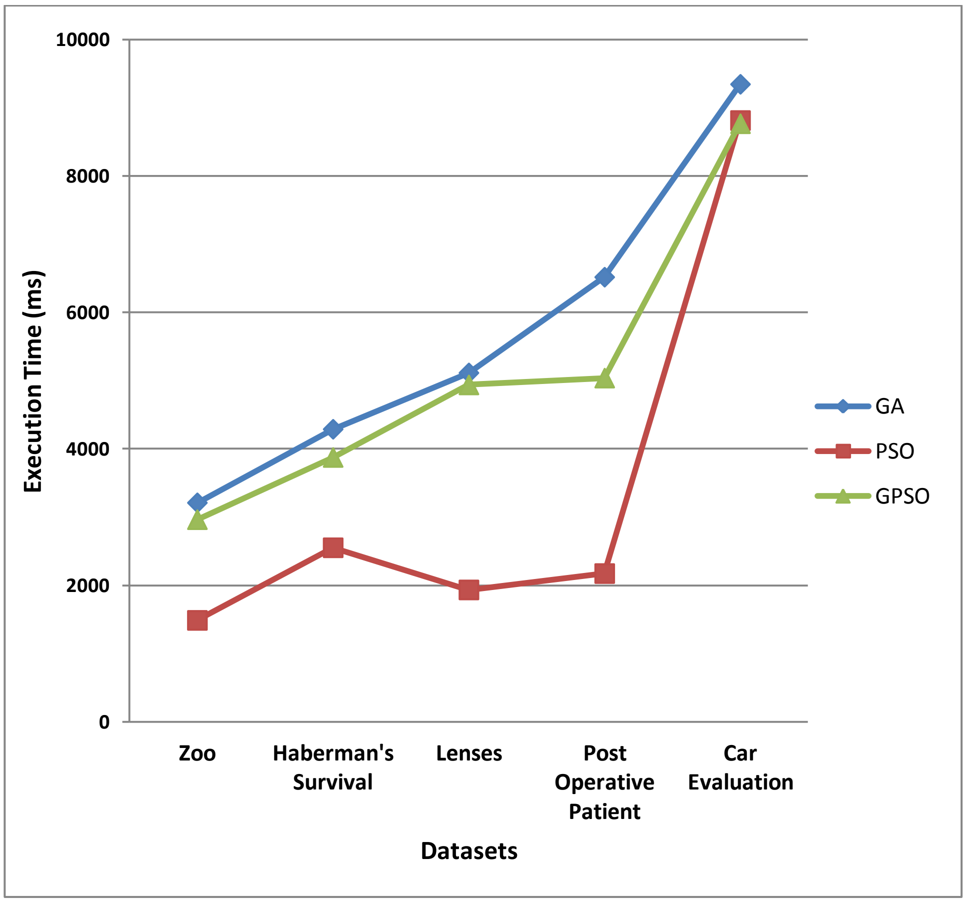


Figure 6.4 Comparison of Execution Times of GPSO with GA and PSO

As standalone both GA and PSO produce results inconsistently for all the five datasets. The drawback of GA is its lack of memory, which limits, its search and convergence ability. The mutation operation also leads to exploitation and hence the inconsistency in accuracy for all the five datasets. PSO tends to converge at local optima resulting in premature convergence, thereby generating inconsistent results. By combining the advantages of both GA and PSO the GPSO method mine ARs with consistent performance. The GPSO method of mining AR outperforms both GA and PSO methodology in terms of prediction accuracy, consistence and execution time.

6.4 PSO WITH LOCAL SEARCH

Some important situations that often occur in PSO is *overshooting*, which is a key issue to premature convergence and essential to the performance of PSO. From the velocity update mechanism of PSO, it is observed that the pB*est* and gB*best* make the particles oscillate. The overshooting problem occurs due to the velocity update mechanism, leading the particles to the wrong or opposite directions against the direction to the global optimum. As a consequence, the pace of convergence of the whole swarm to the global optimum slows down. One possible way to prevent the overshooting problem from happening is to appropriately adjust the algorithmic parameters of PSO. However, it is a difficult task, as the parameter adjustment depends a lot on domain knowledge and the optimization problem.

In consequence of overshooting, the particle will move to the opposite direction against the direction to the global optimum. Two major approaches that can be used to tackle the overshooting problem are described below.

- **Analysis of problems**: By analyzing the structure of problems or identifying the fitness landscape of problems, a lethal movement could be prevented. It poses great challenge on automated problem structure analysis (Bucci and Pollack, 2004)

- **Generate and test strategies**: There are many methods to generate a new solution for testing, such as: heuristics of the specific problem, statistic sampling, and local search techniques. However, the computation costs in generating and testing new solutions may be generally high.

In order to develop a general-purpose algorithm and overcome the overshooting problem, an efficient local search strategy – Shuffle Frog Leaping Algorithm (SFLA), is adopted and combined with the standard PSO for ARM.

6.5 PSO WITH SFLA FOR MINING ASSOCIATION RULES

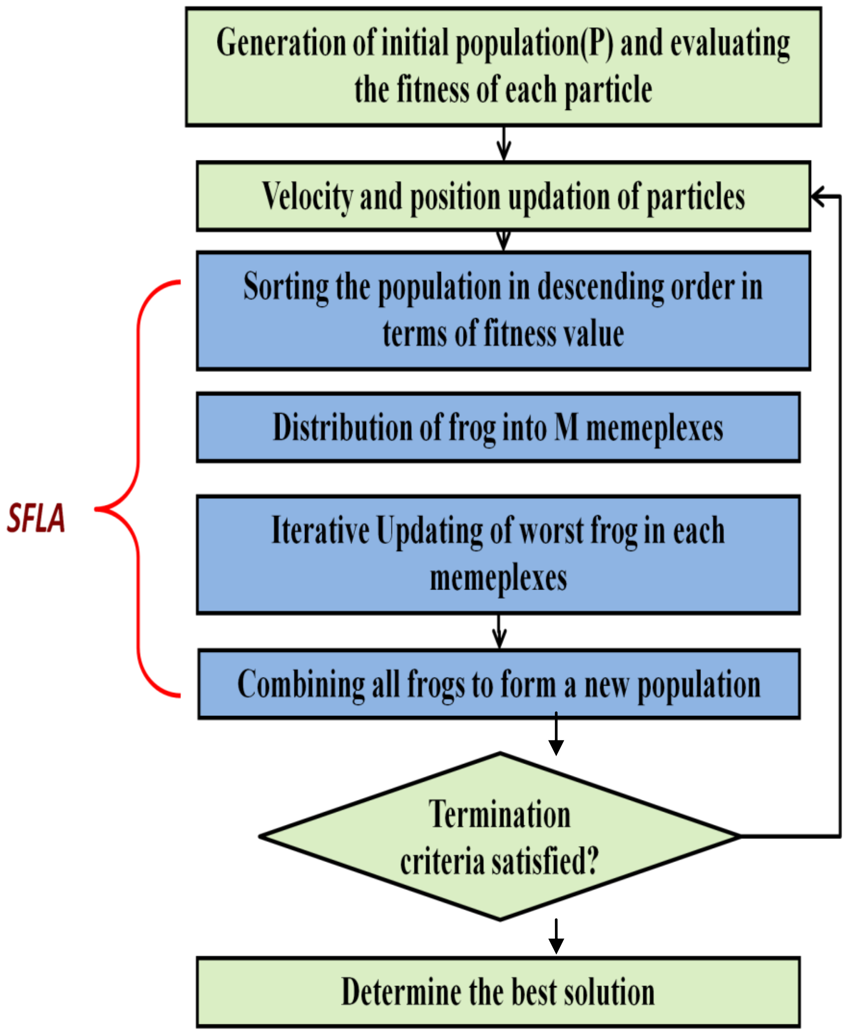
Population-based heuristics inherently improve the implementation of a local search algorithm, since the heuristic approach of a population of solutions results in rather poor local search properties. Incorporating local search algorithm into the population based heuristic is called a Memetic Algorithm.

In published literature many LS schemes have been employed with PSO for optimization. Petalas et al. (2007) employed a stochastic iterative LS technique in their MA, called RWDE, where a sequence of approximations of the optimizer are generated by assuming a random vector as a search velocity. It was noticed by Victoire and Jeyakumar (2004) that early on in the PSO search, particles were almost close to the proximity of the global optimum, then move away from these areas. For this reason the local search method was chosen for implementation. Inspired from literature a memetic PSO with SFLA for local search is proposed for mining ARs.

The SFLA is a memetic metaheuristics that is designed to seek a global optimal solution by performing a heuristic search. It is based on the evolution of memes carried by individuals and a global exchange of information among the population (Eusuff and Lansey 2003). The SFLA involves a population of possible solutions defined by a set of frogs (i.e. solutions) that is partitioned into subsets referred as memeplexes. The different memeplexes are considered as different cultures of frogs, each performing a local search. Within each memeplex, the individual frogs hold ideas that can be influenced by the ideas of other frogs, and evolve through a process of memetic evolution. After a number of memetic evolution steps, ideas are passed among memeplexes in a shuffling process (Liong and Atiquzzaman 2004). The local search and the shuffling processes continue until convergence criteria are satisfied (Eusuff and Lansey 2003). As SFLA and PSO both are designed for optimization problem and SFLA works in similar with PSO population with no new complexity additions, SFLA is chosen for local search in PSO for mining association rules.

6.5.1 Methodology

Initially the particles are distributed in the search space and fitness of the particles is calculated. The velocity and position updation of the particles are carried out using PSO’s velocity and position update equations. Then the particles are sorted in descending order based on their fitness value and memeplexes are formed. Within each memeplex the position of the Frog with worst fitness is updated. Later during the shuffling process, the frogs in different groups communicate with each other to exchange information about the global best. The overall steps involved in Memetic PSO are given in the Figure 6.5.



Generation of initial population(P) and evaluating the fitness of each particle
Velocity and position updation of particles
Sorting the population in descending order in terms of fitness value
Distribution of frog into M memeplexes
Iterative Updating of worst frog in each memeplexes
Combining all frogs to form a new population
Termination criteria satisfied?
Determine the best solution
SFLA

Figure 6.5 Flowchart for PSO with SFLA for ARM

In SFLA, the population consists of a set of frogs (solutions) that is partitioned into subsets and it is named as memeplexes. The different memeplexes are considered as different cultures of frogs, each performing a local search. Within each memeplex, the individual frogs have different ideas, that can be influenced by the ideas of other frogs, and evolve through a process of memetic evolution. After a defined number of memetic evolution steps, ideas are passed among memeplexes in a shuffling process. The local search and the shuffling processes continue until defined convergence criteria are satisfied.

The SFLA starts with an initial population of P frogs created randomly. Then, the frogs are sorted in a descending order of fitness. Then, the entire population is divided into m memeplexes, each containing N frogs. In this process, the first frog goes to the first memeplex, the second frog goes to the Second memeplex, frog M goes to the Mth memeplex, and frog M+1 goes back to the first memeplex, etc. The process is as shown in Figure 6.6. Within each memeplex, the frogs with the best and the worst fitness are identified as X_b and X_w, respectively. Also, the frog with the global best fitness is identified as X_g. Then, a process similar to PSO is applied to improve only the frog with the worst fitness (not all frogs) in each cycle.

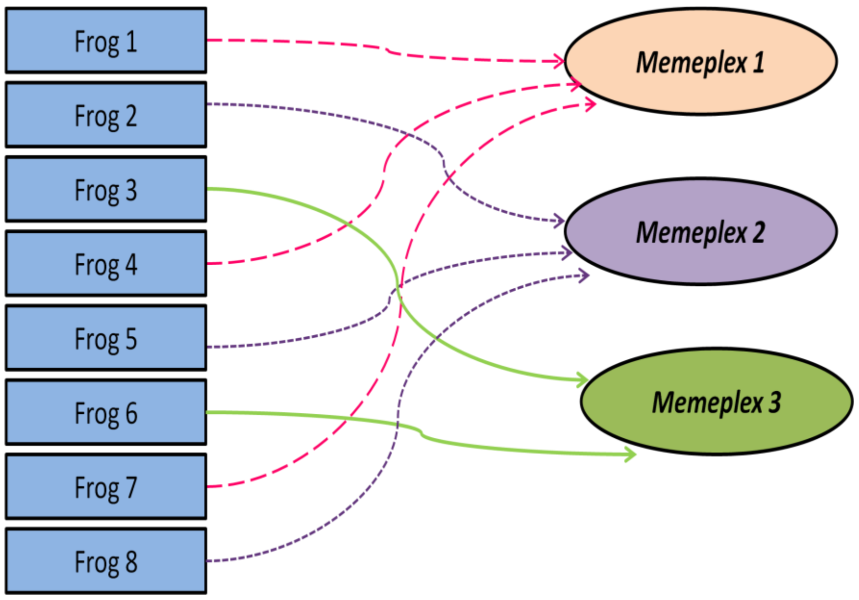


Figure 6.6 Formation of Memeplexes

Accordingly, the position of the frog with the worst fitness is adjusted based on Equations (6.1) and (6.2).

$$\text{Change in frog position}(D_i) = \text{rand}() * X_b - X_w \tag{6.1}$$

$$\text{New Position } (X_w) = \text{Current Position } (X_w) + (D_i) \tag{6.2}$$

where rand ( ) is a random number between 0 and 1; X_b is the position of best frog in the group; X_w is the position of worst frog in the group. If this process produces a better solution, it replaces the worst frog. Otherwise, the calculations in Equations are repeated, but with respect to the global best frog (i.e. X_g replaces X_b).

The fitness function defined in equation (4.2) is used for evaluating the fitness of the individuals. Both PSO and APSO methods are combined with SFLA resulting in two proposals, namely, PSO+SFLA and APSO+SFLA for ARM.

6.5.2 Experimental Results and Discussion

The PSO and APSO methodologies are both combined with SFLA for local search to mine ARs as described in previous section. The five datasets used for all the other methodologies is adopted for generating ARs.

ARs are mined from the datasets using the two proposals and the PA of the generated rules are plotted as shown in Figure 6.7.

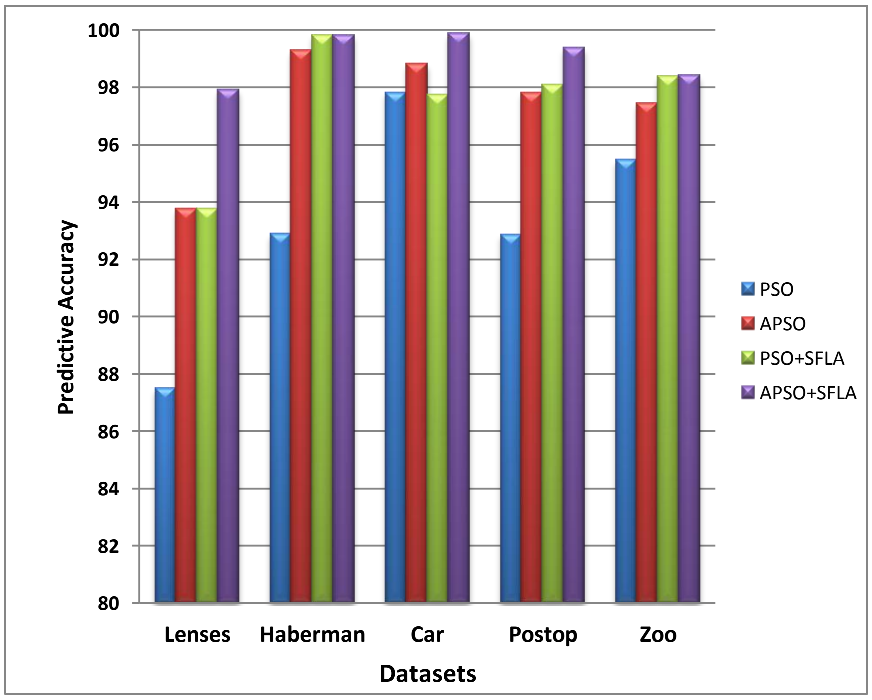
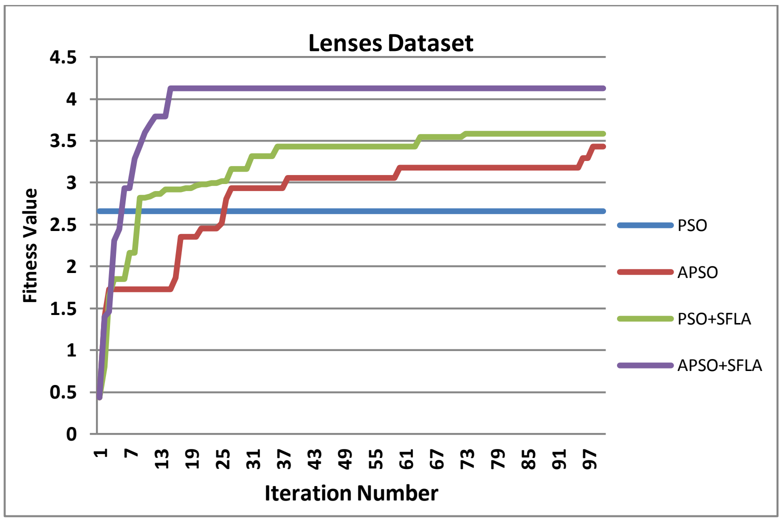


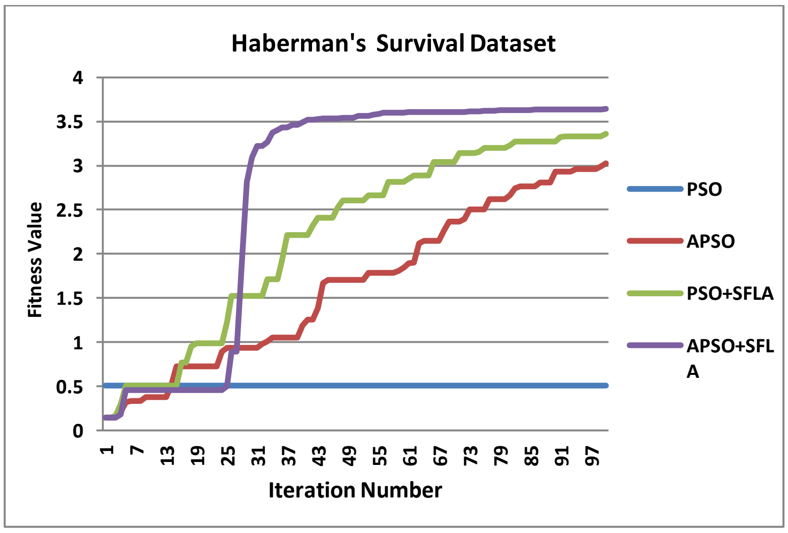
Figure 6.7 Predictive Accuracy Comparison for ARM

PSO+SFLA methodology of mining ARs performs better than simple PSO in terms of PA for all the five datasets taken up for analysis. The APSO+SFLA methodology for mining ARs outperforms the other three methods.

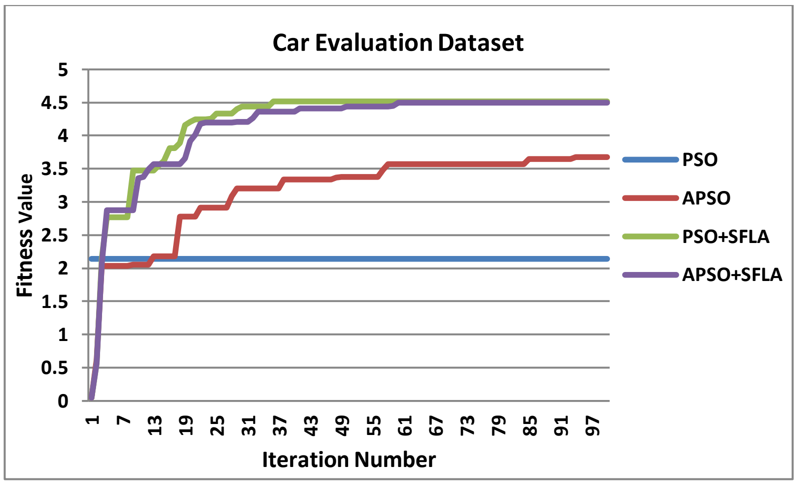
The fitness function objective is maximization. In order to enhance the results the fitness values generated should be optimal. The fitness values for the proposed methodologies are plotted in Figure 8.8 for all the five datasets.



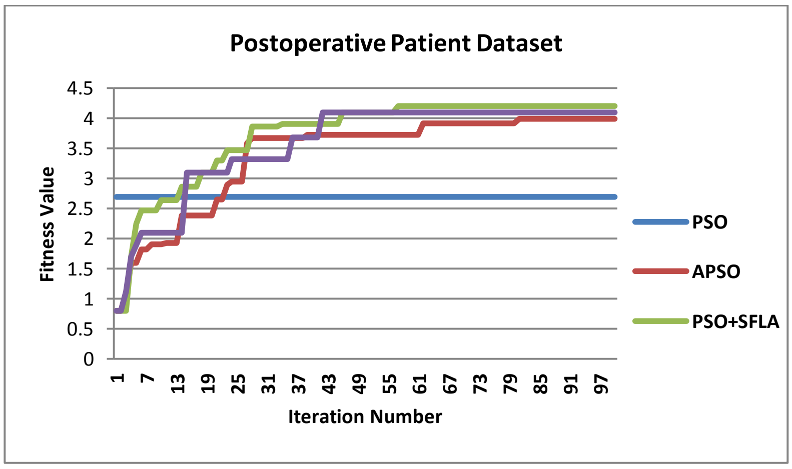
(a) Lenses Dataset



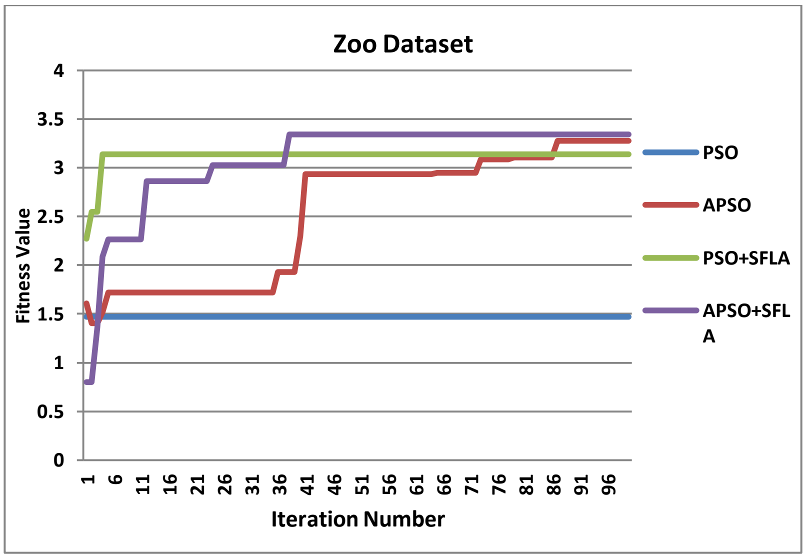
(b) Haberman's Survival Dataset



(c) Car Evaluation Dataset



(d) Postoperative Patient Dataset



(e) Zoo Dataset

Figure 6.8 Fitness Value for PSO with SFLA for ARM

The fitness values of both proposed methodologies; PSO +SFLA and APSO+SFLA are more than the respective individual PSO and APSO values. Thus, both methods perform better by generating ARs with enhanced PA, than PSO and APSO methods.

The performances of the proposed two methods are compared with GA and PSO methods discussed so far in terms of the PA of the ARs mined and the results for the five datasets are shown in Table 6.2.

The APSO+SFLA methodology outperforms the other methods for all the five datasets by generating ARs with better PA. The APSO methodology too generates ARs with optimal accuracy compared to other methods. The data independent adaptation methodologies (SAPSO1, SAPSO2 and SACPSO) rank next in terms of performance for all the five datasets. However the performance of other methods varies among datasets considered in this study.

The number of rules generated by each methodology for the datasets taken up for analysis is given in Table 6.3. The SACPSO1 methodology performs better among the data independent adaptation methodologies, considered for analysis here as SAPSO.

The APSO+SFLA methodology of mining ARs generates more rules than the other methods discussed. The SAPSO (SAPSO1) performs better by generating optimal number of ARs. For Car Evaluation dataset WPSO and SAPSO methods generates same number of rules. The inertia weight attained by Eqn (5.7) of SAPSO1 and the inertia weight set for WPSO for the instance of execution noted might have induced the same number of rules.

Thus the proposed APSO+SFLA methodology performs better when compared to the other methods, in terms of PA and number of rules generated. The SFLA performs effective local search thereby balancing between exploration and exploitation and hence better performance.

Table 6.2 Comparison of Predictive Accuracy for ARM

	GA	AGA	Elitist GA	PSO	WPSO	CPSO	NPSO	SAPSO	APSO	GPS0	PSO+ SFLA	APSO +SFLA
Lenses	85	91.6	91.6	83.57	97.91	87.5	93.1	97.91	98.1	87.5	93.75	98.92
Car Evaluation	87	94	97	97.61	99.93	99.86	97.1	99.92	99.82	95.12	99.82	99.88
Haberman’s Survival	81	91.6	85	92.86	99.48	96.15	92	99.2	99.83	98.05	97.73	99.88
Postopertative Care	74	79	82	83.33	99.29	92.86	94.5	99.47	99.37	90.42	98.07	99.54
Zoo	81	86	89.54	95.45	96.67	94.44	96.5	99.09	99.72	95.35	99.38	99.84

Table 6.3 Comparison of the Number of Rules Generated in ARM

	GA	AGA	Elitist GA	PSO	WPSO	CPSO	NPSO	SAPSO	APSO	GPSO	PSO+ SFLA	APSO+ SFLA
Lenses	10	12	8	3	15	4	13	14	8	3	16	18
Post Operative Patient	22	18	32	13	58	17	57	60	67	24	24	71
Zoo	10	13	13	6	32	14	18	20	38	23	22	41
Haberman's Survival	46	38	55	35	63	24	110	114	12	47	54	121
Car Evaluation	120	126	130	118	135	128	131	135	142	128	130	151

6.6 SUMMARY

A hybrid method combining both genetic algorithm and Particle Swarm Optimization called GPSO has been proposed. This method brings a balance between exploration and exploitation, resulting in higher prediction accuracy of the ARs mined and consistency in performance. Two methodologies using PSO with SFLA for local search (PSO+SFLA and APSO+SFLA) has been proposed. Among them, APSO+SFLA methodology generate ARs with better PA, than all other methodologies discussed so far.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 GENERAL

In this research, various important issues concerned with ARM have been addressed. Investigations carried out in this research were mainly focused towards developing an efficient methodology for ARM model by making use of the two population based stochastic search algorithms, namely GA and PSO. The salient conclusions of this research work and the scope for future work are presented in this chapter.

7.2 SALIENT CONCLUSIONS

ARM was attempted using GA and PSO methodologies. Modification and parameter tuning was done on both methods to enhance the PA of rules mined. A hybrid of GA and PSO was proposed for ARM. PSO with SFLA for local searched was proposed and from the results attained, The salient inferences arrived are as follows:.

(i) Genetic Algorithm when used for mining ARs performs better than other existing traditional methods.

(ii) Particle swarm optimization when applied for mining ARs produce results better than GA, but with minimum execution time. The increase in PA of PSO over GA is: 7.8% for Lenses dataset, 7.4% for Haberman’s survival dataset, 10.6% for the Car evaluation dataset, 9.33% for the Postoperative patient dataset and 14.45% for the Zoo dataset. The reduction in execution of PSO over GA is achieved varying from 2 to 18 seconds for the chosen datasets.

(iii) Modification introduced in GA and PSO as Elitism, Chaotic behavior and Dynamic neighborhood selection strikes a good balance between exploration and exploitation, thus enhancing its performance when compared to its simple version.

(iv) Variations when introduced in both GA and PSO indicate adaptive mechanism performs better than others. i.e. The data dependent adaption method (APSO) ARs with higher PA when compared to methods without local search. The PA accuracy achieved for all the five datasets are varying from 98.1% to 99.82%. The number of rules generated by this methodology is also more for all methods except the memetic methods.

(v) GPSO methodology produces consistent results in comparison with GA and PSO. The PA achieved is almost same throughout the generations for each dataset, by utilizing GA to maintain diversity on high ranked population and applying PSO's easy convergence property, on low ranked population.

(vi) The proposed APSO +SFLA methodology performs better than all other methods considered, in terms of PA and number of rules generated. The PA achieved by PSO+SFLA is maximum among all the discussed methods. The adaptation of control parameters depending on data involved by APSO and the effective LS by SFLA resulted in PA varying from 98.92% to 99.88% for all the five datasets. The number of rules generated by this methodology is also high when compared to other methods (Table 6.3).

(vii) The adaptive methods (dependent and independent) for ARs perform better among the other methods proposed. Increase in performance in terms of PA achieved over its simple version (GA and PSO) is: 6.6% by AGA and 14.8% by APSO for lenses dataset; 10% by AGA and 2.3% by SAPSO for Car evaluation dataset; 10.6% by AGA and 7.03% by APSO for Haberman's survival dataset; 5% by AGA and 16.04% by APSO for Postoperative patient dataset; 5% by AGA and 3.13% by APSO for the Zoo dataset.

(viii) The rule set measures namely; Lift, Laplace, Leverage and Conviction when tested on the ARs generated by the proposed methods are within specified range proposed in literature, signifying the quality of the rules.

7.3 SCOPE FOR FUTURE WORK

The ARM methods proposed have only been experimented with benchmarked datasets from UCI repository. In future, this could be extended to complex and real life problems belonging to unexplored application-domains, and the execution time analysis of these methods can be carried out. Methods to reduce the execution time could be explored.

ANNEXURE 1

UCI DATABASES

The datasets that have been used in this research work have been taken from the UCI machine learning repository. The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The archive was created as an ftp archive in 1987 by David Aha and fellow graduate students at UC Irvine. The details of the datasets used in this study from http://archive.ics.uci.edu/ml/about.html are presented below.

1. LENSES DATASET

Number of Instances	:	24
Number of Attributes	:	4 (all nominal)
Attribute Information	:	3 Classes

1 : the patient should be fitted with hard contact lenses,

2 : the patient should be fitted with soft contact lenses,

3 : the patient should not be fitted with contact lenses.

- age of the patient: (1) young, (2) pre-presbyopic, (3) presbyopic
- spectacle prescription: (1) myope, (2) hypermetrope
- astigmatic: (1) no, (2) yes
- tear production rate: (1) reduced, (2) normal

Number of Missing Attribute Values: 0

2. HABERMAN'S SURVIVAL DATASET

Number of Instances	:	306
Number of Attributes	:	4 (including the class attribute)
Attribute Information	:	

1. Age of patient at time of operation (numerical)
2. Patient's year of operation (year - 1900, numerical)
3. Number of positive axillary nodes detected (numerical)
4. Survival status (class attribute)

1	=	the patient survived 5 years or longer
2	=	the patient died within 5 year

Missing Attribute Values : None

3. CAR EVALUATION DATASET

Number of Instances	:	1728 (instances completely cover the attribute space)
Number of Attributes	:	6
Attribute Values	:	

buying	v-high, high, med, low
maint	v-high, high, med, low
doors	2, 3, 4, 5-more
persons	2, 4, more
lug_boot	small, med, big
safety	low, med, high

Missing Attribute Values : none

4. POST OPERATIVE PATIENT DATASET

Number of Instances	: 90
Number of Attributes	: 9 including the decision (class attribute)
Attribute Information	:

1. L-CORE (patient's internal temperature in C):

   high (> 37), mid (>= 36 and <= 37), low (< 36)

2. L-SURF (patient's surface temperature in C):

   high (> 36.5), mid (>= 36.5 and <= 35), low (< 35)

3. L-O2 (oxygen saturation in %):

   excellent (>= 98), good (>= 90 and < 98),

   fair (>= 80 and < 90), poor (< 80)

4. L-BP (last measurement of blood pressure):

   high (> 130/90), mid (<= 130/90 and >= 90/70), low (< 90/70)

5. SURF-STBL (stability of patient's surface temperature):

   stable, mod-stable, unstable

6. CORE-STBL (stability of patient's core temperature)

   stable, mod-stable, unstable

7. BP-STBL (stability of patient's blood pressure)

   stable, mod-stable, unstable

8. COMFORT (patient's perceived comfort at discharge, measured as an integer between 0 and 20)

9. decision ADM-DECS (discharge decision):

   I (patient sent to Intensive Care Unit),

   S (patient prepared to go home),

   A (patient sent to general hospital floor)

Missing Attribute Values : Attribute 8 has 3 missing values

5. ZOO DATASET

Number of Instances	:	101
Number of Attributes	:	18 (animal name, 15 Boolean attributes, 2 numerics)
Attribute Information	:	(name of attribute and type of value domain)

1. animal name	Unique for each instance
2. hair	Boolean
3. feathers	Boolean
4. eggs	Boolean
5. milk	Boolean
6. airborne	Boolean
7. aquatic	Boolean
8. predator	Boolean
9. toothed	Boolean
10. backbone	Boolean
11. breathes	Boolean
12. venomous	Boolean
13. fins	Boolean
14. legs	Numeric (set of values: {0,2,4,5,6,8})
15. tail	Boolean
16. domestic	Boolean
17. catsize	Boolean
18. type	Numeric (integer values in range [1,7])

Missing Attribute Values : None

6. IRIS DATASET

Number of Instances	:	150 (50 in each of three classes)
Number of Attributes	:	4 numeric, predictive attributes and the class
Attribute Information	:	Sepal length, Sepal width, Petal length and Petal width in cm
Class Types	:	Iris Setosa, Iris Versicolour and Iris Virginica.
Missing Attribute Values	:	None

7. NURSERY DATASET

Number of Instances	:	12960 (instances completely cover the attribute space)
Number of Attributes	:	8
Attribute Values	:	

parents	usual, pretentious, great_pret
has_nurs	proper, less_proper, improper, critical, very_crit
form	complete, completed, incomplete, foster
children	1, 2, 3, more
housing	convenient, less_conv, critical
finance	convenient, inconv
social	non-prob, slightly_prob, problematic
health	recommended, priority, not_recom

Missing Attribute Values : none

8. TIC TAC TOE DATASET

Number of Instances	:	958 (legal tic-tac-toe endgame boards)
Number of Attributes	:	9, each corresponding to one tic-tac-toe square
Attribute Information	:	(x=player x has taken, o=player o has taken, b=blank)

1. top-left-square: {x,o,b}
2. top-middle-square: {x,o,b}
3. top-right-square: {x,o,b}
4. middle-left-square: {x,o,b}
5. middle-middle-square: {x,o,b}
6. middle-right-square: {x,o,b}
7. bottom-left-square: {x,o,b}
8. bottom-middle-square: {x,o,b}
9. bottom-right-square: {x,o,b}
10. Class: {positive,negative}

Missing Attribute Values : None

9. WISCONSIN BREAST CANCER DATASET

Number of Instances	: 699
Number of Attributes	: 10 plus the class attribute
Attribute Information	:

Sample code number, Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses and Class (2 for benign, 4 for malignant).

Missing attribute values : 16

10. ADULT DATASET

Number of Instances	:	48842
Number of Attributes	:	14
Attribute Information	:	Listing of attributes: >50K, <=50K.
age	:	continuous.
workclass	:	Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local- gov, State-gov, Without-pay, Never-worked.
Fnlwgt	:	continuous.
Education	:	Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
education-num	:	continuous.
marital-status	:	Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
Occupation	:	Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
Relationship	:	Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

Race	:	White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
Sex	:	Female, Male.
capital-gain	:	continuous.
capital-loss	:	continuous.
hours-per-week	:	continuous.
native-country	:	United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

REFERENCES

1. Abdel-Magid, Y. L. and Abido, M.A. (2003) “Coordinated Design of a PSS and a SVC-based Controller to Enhance Power System Stability”, *International Journal of Electrical Power and Energy Systems*, Vol. 25, No.9, pp. 695-704.

2. Abido, M.A. (2005) “Analysis and Assessment of STATCOM-based Damping Stabilizer for Power System Stability Enhancement”, *Electrical Power System Research*, Vol. 73, No. 2, pp. 177-185.

3. Agrawal, R. and Srikant,R. (1994) “Fast Algorithms for Mining Association Rules in Large Databases”, *Proceeding of the 20th International Conference on Very Large DataBases”*, Santiago de Chile, Chile, September 12-15, pp. 487-499.

4. Agrawal, R., Imielinsky, T. and Swami, A. (1993) “Mining Association Rules Between Sets of Items in Large Databases*”, ACM SIGMOD Conference on Management of Data*, Washington, D.C, pp. 207-216.

5. Alatas, B. and Akin, E. (2006) “An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules”**,** *Soft Computing*, Vol.10, No.3, pp 230-237.

6. Alatas, G. and Akin, E. (2008a) “Rough Particle Swarm Optimization and its Applications”, *Soft Computing*, Vol. 12, pp. 1205–1218.

7. Alatas, G. and Akin, E. (2008b) “Chaos Rough Particle Swarm Optimization and its Applications”, *Information Sciences*, Vol. 163, pp. 194–203.

8. Alatas, B., Akin, E. and Ozer, A.B. (2009) “Chaos Embedded Particle Swarm Optimization Algorithms”, *Chaos, Solitons & Fractals,* Vol. 40, No.4, pp. 715–734.

9. Ang, J.H., Tan, K.C. and Mamun, A.A, (2010) “An evolutionary memetic algorithm for rule extraction”, *Expert Systems with Applications,* Vol. 37, pp.1302–1315.

10. Angeline, P.J. (1998a) “Using Selection to Improve Particle Swarm Optimization”, *Proceedings of the IEEE Congress on Evolutionary Computation,* Anchorage, Alaska, May 4-9, pp. 84-89.

11. Angeline, P.J. (1998) ”Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences”, *Evolutionary Programming VII, Lecture Notes in Computer Science,* Vol. 1447, pp. 601–610.

12. Arumugam, M.S. and Rao, M.V. (2006) “On the Performance of the Particle Swarm Optimization Algorithm with Various Inertia Weight variants for Computing Optimal Control of a Class of Hybrid Systems”, *Discrete Dynamics in Nature and Society,* Vol. 2006, Article ID 79295,pp.1–17.

13. Asadi, A., Afzali, M., Shojaei, A. and Sulaimani, S. (2012) “New Binary PSO based Method for Finding Best Thresholds in Association Rule Mining”, *Applied Soft Computing,* pp. 260–264.

14. Avendano J. Christian, Gutierrez P Martin, (2010) “Optimization of Association Rules with Genetic Algorithms”, *29th IEEE International Conference of the Chilean Computer Science Society,* pp.193-197.

15. B¨ack, T. (1992) “Self-Adaptation in Genetic Algorithms”, *Proceedings of the 1st European Conference on Artificial Life,* Paris, France, September 1-3., pp. 263–271.

16. B¨ack, T. (1994). “Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms”. *Proceedings of the First IEEE Conference on Evolutionary Computation*, Orlando, Florence, June 27-29, Vol.1, pp. 57–62.

17. B¨ack, T. (1996) “Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic algorithms”, *Oxford University Press,* New York.

18. Baker, J. E. (1985) “Adaptive Selection Methods for Genetic Algorithms”, *Proceedings of International Conference on Genetic Algorithms and Their Applications*, pp.101–111.

19. Blake, C., L. and Merz, C., (1998) UCI Repository of Machine Learning Databases, Irvine, CA: University of California, Department of Information and Computer Science available at http://www.ics.uci.edu/~mlearn.

20. Bodon, F. (2003) “ A Fast Apriori Implementation”, *IEEE ICDM workshop on Frequent Itemset Mining Implementation (FIMI’03),* Melbourne, Florida, USA, December 19.

21. Booker, L. B. (1982) “Intelligent Behavior as an Adaptation to the Task Environment”, A Thesis submitted for the Degree of Doctor of Philosophy, University of Michigan, Ann Arbor, MI.

22. Brin, S., Motwani, R. and Silverstein, C. (1997a) “Beyond Market Baskets: Generalizing Association Rules to Correlations”, *ACM SIGMOD/PODS Joint Conference*, Tucson, AZ, USA, May 11 - 15, pp. 265–276.

23. Brin, S., Motwani, R., Ullman, J.D, and Tsur, S. (1997b) “Dynamic Itemset Counting and Implication Rules for Market Basket Data, *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, Tucson, AZ, USA , May 11 - 15, pp. 255–264.

24. Brindle, A. (1981) "Genetic Algorithms for Function Optimization". A Thesis submitted for the Degree of Doctor of Philosophy, University of Alberta, Edmonton, Canada.

25. Brits R, Engelbrecht AP, and van den Bergh F (2002) "A Niching Particle Swarm Optimizer", *Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning,* Orchid Country Club, Singapore, November 18-22.

26. Bucci, A., Pollack, J.B. and De Jong, E.D. (2004) "Automated Extraction of Problem Structure", *Proceedings of the Genetic and Evolutionary Computation Conference,* Seattle, WA, USA, June 26-30, pp. 501-512.

27. Carlisle, A. and & Dozier, G. (2000) "Adapting Particle Swarm Optimization to Dynamic Environments", Proceedings of International Conference on Artificial Intelligence", Las Vegas, Neveda, June 26-29, pp. 429–434.

28. Carlisle, A. and Dozier, G (2001) "An off-the-shelf PSO", *Proceeding on Particle Swarm Optimization,* Indianapolis, Purdue School of Engineering and Technology, pp 1–6.

29. Cattral, R., Oppacher, F. and Deugo, D. (1999) "Rule Acquisition with a Genetic Algorithm", *Proceedings of the 1999 Congress on Evolutionary Computation,* Washington, DC, USA, July 6-9, Vol.1.

30. Ceglar, A. and Roddick, J.F. (2006) "Association Mining", *ACM Computing Surveys (C SUR),* Vol. 38, No.2.

31. Chatterjee, A. and Siarry, P. (2006) "Nonlinear Inertia Weight Variation for Dynamic Adaptation in Particle Swarm Optimization", *Computers and Operations Research,* Vol. 33, No.3, pp. 859–871.

32. Chiam, S.C., Tan, K.C and Mamun, A.Al. (2009) “A Memetic Model of Evolutionary PSO for Computational Finance Applications”, *Expert Systems with Applications,* Vol. 36, No. 2, pp. 3695–3711.

33. Clark P, Boswell (1991) “Rule Induction with CN2: Some Recent Improvements”, *Proceedings of the European Working Session on Learning*, Porto, Portugal, March 6-8, pp 151-163.

34. Chun-Ho Wu, Na Dong, Wai-Hung Ip, Ching-Yuen Chan, Kei-Leung Yung and Zeng-Qiang Chen. (2010) “Chaotic Hybrid Algorithm and Its Application in Circle Detection, Applications of Evolutionary Computation”, *Lecture Notes in Computer Science,* Vol.6024, pp. 302-311.

35. Dantzig, G.B. (1955) “Linear Programming under Uncertainty”, *Management Science,* Vol.1, pp.197–206.

36. Das, A., Ng, W.-K. and Woon, Y.-K. (2001) “Rapid Association Rule Mining”, *Tenth International Conference on Information and Knowledge Management CIKM’01.* Atlanta, Georgia, November 5-10, pp. 474-481.

37. Dazhi Jiang, Sanyou Zeng, Hui Wang and Zhijian Wu (2011), “Particle Swarm Optimizer with Hybrid Multi-parent Crossover and Discrete Recombination”, *International Journal of Intelligent Information and Database Systems,* Vol. 5, No.6, pp. 597 – 617.

38. Deepa Shenoy,P., Srinivasa,K.G., Venugopal,K.R. and Lalit M. Patnaik, (2003). “Evolutionary Approach for Mining Association Rules on Dynamic Databases”, *Proceeding of Pacific-Asia Conference on Knowledge Discovery and Data Mining,* LNAI, 2637, Springer-Verlag, 325–336.

39. Dehuri, S. and Mall, R. (2006) “Predictive and Comprehensible Rule Discovery using a Multiobjective Genetic Algorithm”, *Knowledge Based Systems*, Vol. 19, No.6, pp. 413-421.

40. Dehuri, S., Jagadev, A.K., Ghosh, A. and Mall, R. (2006) “Multi-objective Genetic Algorithm for Association Rule Mining Using a Homogeneous Dedicated Cluster of Workstations”, *American Journal of Applied Sciences,* Vol.3 , No. 11, pp. 2086-2095.

41. Dehuri, S., Patnaik, S., Ghosh, A. and Mall, R. (2008) “Application of elitist multi-objective genetic algorithm for classification rule generation”, *Applied Soft Computing*, pp.477–487.

42. DeJong, K. A. (1975) “An Analysis of the Behavior of a Class of Genetic Adaptive Systems*”*. A Thesis submitted for the Degree of Doctor of Philosophy, University of Michigan, Ann Arbor.

43. De Jong, K.A., Spears, W.M., and Gordon, D.F. (1993) “Using Genetic Algorithms for Concept Learning”, *Machine Learning,* Vol. 13, No. 2-3, pp.161–188.

44. Dong Na, Chun-Ho Wu, Wai-Hung Ip, Zengqiang Chen, Ching-Yuen Chan and Kai-Leung Yung (2012) “ An OPposition-based Chaotic GA/PSO Hybrid Algorithm and its Application in Crcle Detection”, *Computers & Mathematics with Applications,* Vol. 64, No. 6, PP. 1886-1902.

45. Dorigo, M., Maniezzo, V. and Colorni, A. (1991) “Positive Feedback as a Search Strategy”. *Technical Report 91-016,* Dipartimento di Elettronica, Politecnico di Milano, IT, pp: 91-106.

46. Eberhart, R. C. and Kennedy, J. (1995) “A New Optimizer using Particle Swarm Theory”, *Proceedings of the Sixth International Symposium on Micro Machine and Human Science”*, Nagoya, Japan. Piscataway, October 4-6, pp. 39–43.

47. Eberhart, R. C., Simpson, P. K. and Dobbins, R. W. (1996) “Computational intelligence PC tools*”,* Boston, Academic Press.

48. Eberhart, R.C. and Shi, Y. (1998)”Comparison between Genetic Algorithms and Particle Swarm Optimization”, *Evolutionary Programming VII (1998), Lecture Notes in Computer Science,* Vol. 1447, pp. 611–616.

49. Eberhart, R. C. and Shi, Y. (2000) “Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization”, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)”,* San Diego, CA. Piscataway, July 16-19, pp. 84–88 IEEE.

50. Eberhart, R.C. and Shi, Y. (2001) “Tracking and Optimizing Dynamic Systems with Particle Swarms”, *Proceedings of the Congress on Evolutionary Computation, IEEE,* Seoul, South Korea, May 27-30, Vol.1, pp 94–100.

51. Eckart Zitzler, Kalyanmoy Deb and Lothar Thiele. (2000) “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results”. *Evolutionary Computation,* Vol.8, No. 2, pp.173-195.

52. Eiben, A.E. and Smith, J.E., (2003) “Introduction to Evolutionary Computing”. Springer-Verlag, Berlin Heidelberg New York.

53. Eiben A. E., Michalewicz, Z., Schoenauer, M. and Smith, J. E. (2007) “Parameter Control in Evolutionary Algorithms”, *Lobo F.G. and Lima C.F. and Michalewicz Z. (eds.) , Parameter Setting in Evolutionary Algorithms* , Springer, pp. 19-46.

54. Engelbrecht, A.P. (2006) “Fundamentals of Computational Swarm Intelligence”, John Wiley & Sons.

55. Esmaeli, A., Nasiri, M., Minaei, B. and Mozayani., N. (2008) "A method for association rule mining with PSO based on confidence", *17th Conference of Electronic (ICEE),* University of Sciences and Technology of Oran, Algeria, Oct. 20-22, pp. 120–124.

56. Eusuff, M.M. and Lansey, K.E. (2003) "Optimization of Water Distribution Network Design using the Shuffled Frog Leaping Algorithm", *Journal of Water Resource, Planning Management,* Vol. 129, 210 – 225.

57. Fayyad, U.M, Piatetsky-Shapiro, G. and Smyth, P. (1996), "From Data Mining to Knowledge Discovery: An Overview", *R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining,* AAAI/MIT, pp. 1–34.

58. Feng Lu, Yanfen Ge and Liqun Gao. (2010) "Self Adaptive Particle Swarm Optimization Algorithm for Global Optimization", *Sixth IEEE International Conference on Natural Computation*, TBD, Yantai, China, August 10-12, pp.2692-2696.

59. Feng, Y., Teng, G.F., Wang, A.X. and Yao, Y.M. (2008) "Chaotic Inertia Weight in Particle Swarm Optimization", *Second IEEE International Conference on Innovative Computing, Information and Control,* Dalian, Liaoning, June 18-20, pp 475-501.

60. Fogel, L.J., Owens, A.J. and Walsh, M.J. (1966) "Artificial Intelligence Through simulated Evolution", John Wiley & Sons, New York.

61. Fonseca C.M. and Fleming, P.J. (1995) "An Overview of Evolutionary Algorithms in Multi-objective Optimization", *Evolutionary Computation*, Vol.3, No.1, pp.1–16.

62. Frawley, W. J., Piatetsky-Shapiro, G. and Matheus, C. J. (1992) "Knowledge Discovery In Databases: An Overview". *AI Magazine,* Vol.*13, No.*3, pp. 57-70.

63. Freitas A.A. (1997) "A Genetic Programming Framework for two Data Mining Tasks: Classification and Generalized Rule Induction", Genetic Programming, *Proceedings of 2nd Annual Conference on Genetic Programming*, Morgan Kaufman, pp. 96–101.

64. Freitas A.A. (1999), "On Rule Interestingness Measures", *Knowledge-Based Systems*, Vol.12, No. 5-6, pp.309–315.

65. Gao, Y., An, X., and Liu, J. (2008) "A Particle Swarm Optimization Algorithm with Logarithm Decreasing Inertia Weight and Chaos Mutation", *International IEEE Conference on Computational Intelligence and Security,* Suzhou, China, Dec. 13-17, Vol.1, pp. 61–65.

66. Ghosh, A., and Nath, B. (2004) "Multi−objective Rule Mining using Genetic Algorithms", *Information Sciences*, Vol. 163, No. 1-3, pp.123–133.

67. Giordana, A., Anglano, C., Giordana, A., Bello, G. L. and Saitta, L. (1997) "A Network Genetic Algorithm for Concept Learning", *Proceedings of the Sixth International Conference on Genetic Algorithms,* East Lansing, MI, USA, July 19-23,436–443. Morgan Kaufmann.

68. Goldberg, D.E. (1989) "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley.

69. Good, I.J. (1965) "The Estimation of Probabilities: An Essay on Modern Bayesian Methods", MIT Press, Cambridge.

70. Goplan, R.P. and Sucahya, Y.G. (2002) “ITL-Mine: Mining Frequent Itemsets more Efficiently”, *The 2002 International Conference on Fuzzy Systems and Knowledge Discovery,* Orchid Country Club, Singapore, November 18-22, pp. 167-174

71. Grahne, G. and Zhu, J. (2003) “Efficiently Using Prefix-trees in Mining Frequent Itemsets”, *IEEE Workshop on Frequent Itemset Mining Implementation (FIMI’03),* Melbourne, Florida, USA, December 19.

72. Guo, H. and Zhou, Y. (2009) “An Algorithm for Mining Association Rules Based on Improved Genetic Algorithm and its Application”, *3rd International Conference on Genetic and Evolutionary Computing, WGEC 2009,* Guilin, China, October 14-17, pp. 117–120.

73. Gupta, M. (2012) “Application of weighted particle swarm optimization in association rule mining”, .International Journal of Computer Science and Information, Vol.1, pp.2231–5292.

74. Hamid Reza Qodmanan , Mahdi Nasiri, and Behrouz Minaei-Bidgoli. (2011) “Multi Objective Association Rule Mining with Genetic Algorithm without Specifying Minimum Support and Minimum Confidence”, *Expert Systems with Applications*, Vol. 38, No. 1, pp. 288–298.

75. Han,E., Karypis,G., Kumar,V. and Mobasher. B. (1997) “Clustering Based on Association Rule Hypergraphs”, *Workshop on Research Issues on Data Mining and Knowledge Discovery,* Tucson, Arizona, May 11.

76. Han,J., Pei,J. and Yin,Y. (2000) “Mining Frequent Patterns without Candidate Generation”, *Proceeding of SIGMOD’00,* Dallas, TX, USA, May 15 – 18, pp.1-12.

77. Han, J. and Pei, J. (2000a) “Mining Frequent Patterns by Pattern-Growth: Methodology and Implication”, *ACM SIGKDD Explorations Newsletter*, Vol. 2, No.2, pp.14-20.

78. Heppner, H. and Grenander, U. (1990) “A Stochastic Non-linear Model for Coordinated Bird Flocks”, In S. Krasner (Ed.), *The ubiquity of chaos*, Washington: AAAS, pp. 233–238.

79. Hidber, C. (1999) “Online Association Rule Mining”, *Proceedings of ACM SIGMOD International Conference on Management of Data*, Philadelphia, Pennsylvania, USA, June 1-3, pp. 145-156.

80. Higashi, N. and Iba, H. (2003) “Particle Swarm Optimization with Gaussian Mutation”, *Proceedings of the IEEE Swarm Intelligence Symposium,*, Indianapolis, Indiana, USA, April 24-26, pp. 72–79.

81. Hipp, J., Guntzet, U. and Nakhaeizadeh, G. (2000) “Algorithms for Association Rule Mining – A General Survey and Comparison”, *ACM SIGKDD Explorations Newsletter,*, Vol. 2, No. 1, pp. 58-64.

82. Hongfeng Wang, Ilkyeong Moon, Shenxiang Yang and Dingwei Wang (.2012) “A Memetic Particle Swarm Optimization Algorithm for Multimodal Optimization Problems”, *Information Sciences*, Vol. 197, pp. 38–52.

83. Hongfeng Wang, Shengxiang Yang, W. H. Ip, and Dingwei Wang. (2010) “A Particle Swarm Optimization based Memetic Algorithm for Dynamic Optimization Problems”**,** *Natural Computing*, Vol.9, No. 3, pp 703-725.

84. Hong Guo, Ya Zhou, (2009) “An algorithm for mining association rules based on improved genetic algorithm and its applications”, *Third IEEE international conference on Genetic and evolutionary computing*, pp.117-120.

85. Holland, J. (1975) “Adaptation in Natural and Artificial Systems”, University of Michican Press, Ann Harbor, MI.

86. Holland, J.H. (1986) “Escaping Brittleness: The Possibilities of General-purpose Learning Algorithms applied to Parallel Rule-based Systems”, *Computation & Intelligence*, pp. 275-304.

87. Hop, N.V. and Tabucanon, M.T. (2005) “Adaptive Genetic Algorithm for Lot- sizing Problem with Self-adjustment Operation Rate”, *International Journal of Production Economics*, Vol. 98, No.2, pp. 129–135.

88. Jacinto Mata Vzquez, Jos Luis lvarez Macas and Jos Cristbal Riquelme Santos, (2002) “Discovering Numeric Association Rules Via Evolutionary Algorithm”, *Proceeding of Pacific-Asia Conference on Knowledge Discovery and Data Mining,* Taipei, Taiwan, May 6-8 , 40–51, 2002.

89. Juang C-F (2004) “A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design”, *IEEE Transactions on System, Man and Cybernatics – Part B: Cybernetics,* Vol.34, No.2, pp.997–1006.

90. Kennedy, J. (1997) “The Particle Swarm: Social Adaptation of Knowledge”, *Proceedings of the International Conference on Evolutionary Computation, IEEE,* Piscataway, NJ, April 13-16, pp 303–308.

91. Kennedy, J. (1999) ”Small Worlds and Mega- Minds: Effects of Neighborhood Topology on Particle Swarm Performance”, *Proceedings of the 1999 Congress of Evolutionary Computation,* Washington, DC, USA, July 6-9, Vol. 3, pp.1931–1938.

92. Kennedy, J. and Eberhart, R. (1995) "Particle Swarm Optimization", *Proceedings of the IEEE International Conference on Neural Networks,* Piscataway, Nov. 27- Dec. 1, pp: 1942-1948.

93. Kennedy, J. and Eberhart, R. (2001) "Swarm Intelligence", Academic Press, 1sted. San Diego, CA.

94. Kim, K., Man, F., Tang, K.S. and Kwong, S. (1999) "Genetic Algorithms: : Concepts and Designs, Avec disquette". London, England. Springer-Verlag London Limited.

95. Kong, W.J., Cheng, W.J., Ding, J.L. and Chai, T.Y. (2010) "A Reliable and Efficient Hybrid PSO Algorithm for Parameter Optimization of LS-SVM for Production Index Prediction Model", *Third International Symposium on Computational Intelligence and Design*, Vol.2, pp.140-143.

96. Koza, J.R. (1992) "Genetic Programming", MIT Press, Cambridge, MA.

97. Kuo R.J, Chao C.M and Chiu Y.T (2011) "Application of Particle Swarm Optimization in Association Rule Mining", Applied Soft Computing, Vol. 11, No. 1, pp.323-336.

98. Lale Ozbakir, Adil Baykasoglu and Sinem Kulluk. (2008) "Rule Extraction from Neural Networks Via Ant Colony Algorithm for Data Mining Applications", *Lecture Notes in Computer Science,* Vol. 5313, pp. 177-191.

99. Lale Ozbakir, Adil Baykasoglu, Sinem Kulluk and Huseyin Yapıcı. (2010) "TACO-miner: An Ant Colony based Algorithm for Rule Extraction from Trained Neural Networks", *Expert Systems with Applications,* Vol. 36, No. 10, pp. 12295–12305.

100. Liang, J.J., Qin, A.K., Suganthan,P.N. and Baskar, S. (2006) "Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions", *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 3, pp. 281–295.

101. Lin, J and Dunham, M.H. (1998) "Mining Association Rules: Anti-skew Algorithms", *Proceedings of International Conference on Data Engineering (ICDE),* Orlando, FL, Feb. 23-27, pp. 486-493.

102. Lin, W., Alvarez, S.A. and Ruiz, C. (2002) "Efficient Adaptive – Support Association Rule Mining for Recommender Systems", *Data Mining and Knowledge Discovery*, Vol. 6, No. 1, pp. 83-105.

103. Liong, S.Y., and Atiquzzaman, M.D. (2004) "Optimal Design of Water Distribution Network using Shuffled Complex Evolution", *Journal of the Institution of Engineers*, Vol. 44, pp. 93 – 107.

104. Liu Bo-Flu, Hung-Ming Chen, Jian-Hung Chen, Shiow-Fen Hwang, and Shinn-Ying Ho (2005) "MeSwarm: Memetic Particle Swarm Optimization", *GECCO'05*, Washington, DC, USA, June 25-29, pp. 267-268.

105. Liu, B., Hsu, W. and Ma, Y. (1998) "Integrating Classification and Association Rule Mining", *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD*), New York, USA, August 27-31, pp. 80-86.

106. Liu Bo, Abbas, H.A. and McKay, B. (2003) "Classification Rule Discovery with Ant Colony Optimization", *IEEE/WIC International Conference on Intelligent Agent Technology*, Halifax, Canada, October 13-17, pp. 83-88.

107. Lu, H. and Chen, W. (2008) "Self-adaptive Velocity Particle Swarm Optimization for Solving Constrained Optimization Problems", *Journal of Global Optimization*, Vol.41, No.3, pp. 427-445.

108. Ma, H. and Li, X. (2009) “Application of Data Mining in Preventing Credit Card Fraud”, *International Conference on Management and Service Science,* Wuhan , Sep. 20-22 , pp. 1–6.

109. Mao, R. (2001) “Adaptive-FP: An Efficient Method for Multi-Level Multi Dimensional Frequent Pattern Mining”. Simon Fraser University.

110. Maragatham, G. and Lakshmi, M. (2012) “A weighted particle swarm optimization technique for optimizing association rules”, *Communications in Computer and Information Science*, Vol. 270 CCIS (PART II) , pp. 655-664.

111. Mendes, R., Kennedy, J. and Neves, J. (2004) “The Fully Informed Particle Swarm: Simpler, maybe Better”, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No.3, pp. 204–210.

112. Meunier,H., Talbi,E.G. and Reininger, P. (2000) “A Multiobjective Genetic Algorithm for Radio Network Optimization”. *Proceedings of the 2000 Congress on Evolutionary Computation (CEC’00),* La Jolla, CA. IEEE Press, pp. 317–324.

113. Miguel Rodriguez, Diego M. Escalante and Antonio Peregrin (2011) “Efficient Distributed Genetic Algorithm for Rule extraction”, *Applied Soft Computing,* Vol. 11, pp. 733–743.

114. Min Wang, Qin Zou and Caihui Liu(2011) “ Multi-dimension Association Rule Mining Based on Adaptive Genetic Algorithm”, *IEEE International Conference on Uncertainty Reasoning and Knowledge Engineering,* pp.150-153.

115. Mourad Ykhlef (2011) “A Quantum Swarm Evolutionary Algorithm for Mining association rules in large databases “, *Journal of King Saud University – computer and Information sciences, Vol.* 23, pp. 1-6.

116. Mousa, A.A., El-Shorbagy, M.A. and Abd-El-Wahed, W.F. (2012) "Local Search Based Hybrid Particle Swarm Optimization Algorithm for Multiobjective Optimization", *Swarm and Evolutionary Computation*, Vol. 3, pp. 1–14.

117. M¨uhlenbein, H. and Schlierkamp-Voosen, D. (1993) "Predictive Models for the Breeder Genetic Algorithm: Continuous Parameter Optimization", *Evolutionary Computation,* Vol.1, No. 1, pp. 25–49.

118. Nandhini, M., Janani, M. and Sivanandham, S.N. (2012) "Association rule mining using swarm intelligence and domain ontology", *IEEE International Conference on Recent Trends in Information Technology (ICRTIT),* Coimbatore, India, pp.537– 541.

119. Nannen, V., Smit, S.K., and Eiben, A.E. (2008) "Costs and Benefits of Tuning Parameters of Evolutionary Algorithms", *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature*, Dortmund, Germany, September 13-17, pp. 528–538.

120. Nie Ru and Yue Jianhua (2008) "A GA and Particle Swarm Optimization based Hybrid Algorithm", *IEEE Congress on Evolutionary Computation,* Hong Kong, June 1-6, Vol.1, pp.1047-1050.

121. Obayashi,S., Takahashi,S., and Takeguchi,Y.(1998) "Niching and Elitist Models for Multiobjective Genetic Algorithms", *In Parallel Problem Solving from Nature (PPSN'5),* Amsterdam, Sept. , Springer.

122. Ozel, S.A. and Guvenir, H.A. (2001) "An Algorithm for Mining Association Rules Using Perfect Hashing and Database Pruning", *The 10th Turkish Symposium on Artificial and Neural Networks*, Gazimagusa, T.R.N.C., June 2001, pp.257-264.

123. Patel, M.R. , Rana, D.P. and Mehta, R.G. (2013) "FApriori: A modified Apriori algorithm based on checkpoint", *International Conference on Information Systems and Computer Networks (ISCON)*, Mathura, March 9-10, pp. 50-53.

124. Parpinelli, R. S., Lopes, H. S. and Freitas, A. A. (2002) "Data Mining with an Ant Colony Optimization Algorithm", *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 4, pp. 321–332, Special issue on Ant Colony Algorithms.

125. Pasupuleti, S. and Battiti, R. (2006). "The Gregarious Particle Swarm Optimizer (GPSO*)", Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, Seattle, WA, USA, July 08 – 12, pp. 67-74.

126. Pears, R. and Koh, Y.S. (2012) "Weighted association rule mining using particle swarm optimization", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 7104, pp. 327-338.

127. Pei, J., Han, J. and Lakshmanan, L.V.S.(2001) " Mining Frequent Itemetss with Convertible Constraints", *17th International Conference on Data Engineering (ICDE'01)*, Heidelberg, Germany, April 2-6, pp. 433-442.

128. Petalas, Y.G., Parsopoulos, K.E. and Vrahatis, M.N. (2007) "Memetic particle Swarm Optimization", *Annals of operations research*, Vol. 156, No. 1, pp.99–127.

129. Piatetsky-Shapiro G (1991) "Discovery, Analysis, and Presentation of Strong Rules", *Piatetsky-Shapiro G, Frawley WJ, eds. Knowledge Discovery in Databases*. Cambridge, MA: AAAI/MIT Press.

130. Pijls, W. and Bioch, J.C. (1999) “Mining Frequent Itemsets in Memory-Resident Databases”. *Eleventh Belgium/Netherlands Artificial Intelligence Conference (BNAIC’99)*, Kasteel Vaeshartelt, Maastricht, Netherlands, November 3-4, pp. 1-9.

131. Ping-Hung Tang and Ming-Hseng Tseng . (2012) “Adaptive Directed Mutation for Real-coded Genetic Algorithms”, *Applied Soft Computing*, Vol.13, No.1, pp. 600–614.

132. Potter, M.A. and De Jong, K.A. (1994) “A Cooperative Coevolutionary Approach to Function Optimization”. *Proceedings of the Third Conference on Parallel Problem Solving from Nature*. Jerusalem, Israel, October 9-14, pp 249–257.

133. Ramesh Kumar,M. and Iyakutti, K. (2011) “Application of Genetic algorithms for the prioritization of Association Rules”, IJCA Special Issue on Artificial Intelligence Techniques - Novel Approaches & Practical Applications AIT,

134. Rechenberg, I. (1973) “Evolution strategy*”,* Fromman-Holzboog, Stuttgart, pp.147-159.

135. Robinson, J., Sinton, S. and Rahmat-Samii, Y. (2002) “Particle Swarm, Genetic Algorithm, and their Hybrids: Optimization of a Profiled Corrugated Horn Antenna*”, IEEE Antennas and Propagation Society International Symposium and URSI National Radio Science Meeting*, Vol.1, pp.314 – 317.

136. Rudolph, G. “Evolutionary Search under Partially Ordered Sets”. *Technical Report No. CI-67/99*, Dortmund: Department of Computer Science/LS11, University of Dortmund, Germany, 1999.

137. Salleb-Aouissi A, Vrain C and Nortet C (2007) “QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules”, *IJCAI Conference On Artificial Intelligence In Proc. of the 20th international joint conference on Artificial intelligence*, Hyderabad, India, pp 1035-1040.

138. Saggar, M., Agrawal, A.K. and Lad, A. (2004) “Optimization of Association Rule Mining”, *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, pp. 3725–3729.

139. Sarath, K.N.V.D. and Ravi, V. (2013) “Association rule mining using binary particle swarm optimization”, *Engineering Application of Artificial Intelligence* (2013), http://dx.doi.org/ 10.1016/ j.engappai. 2013.06.003.

140. Savasere, A., Omiecinski, E. and Navathe, S. (1995) “An Efficient Algorithm for Mining Association Rules in Large Databases”, *Proceedings of International Conference on Very Large Databases (VLDB),* Zurich, Switzerland, September 11-15, pp. 432-444.

141. Schwefel, H.P. (1977) “Numerische Optimierung von Computer modellen Mittelsder Evolutions strategie”, Birkhaeuser, Basel. (English edition: Numerical Optimization of Computer Models, John Wiley & Sons, Chichester, 1981.

142. Shenoy, P., Haritsa, J., Sudarshan, S., Bhalotia, G., Bawa, M. and Shah, D (2000) “Turbo-charging Vertical Mining of Large Databases”, *Proceedings of ACM SIGMOD International Conference on Management of Data*, Dallas, Texas, USA, May 16-18, pp. 22-33.

143. Shi, Y. and Eberhart, R.C. (1998a) ”Parameter Selection in Particle Swarm Optimization”, *Evolutionary Programming VII , Lecture Notes in Computer Science*, Vol. 1447, 591–600.

144. Shi,Y., and Eberhart, R.C. (1998b) “A Modified Particle Swarm Optimizer”, *Proceedings of the International Conference on Evolutionary Computation, IEEE*, pp 69–73.

145. Shi, Y., and Eberhart, R.C. (1999) “Empirical study of particle swarm optimization” , *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 1945–1950.

146. Shi, Y., and Eberhart, R.C. (2001) “Fuzzy adaptive particle swarm optimization”, *Proceedings on IEEE Congress on Evolutionary Computation*, vol. 1, pp. 101–106.

147. Shi, X.-J. and Lei, H. (2008) “Genetic Algorithm-Based Approach for Classification Rule Discovery”, *International Conference on Information Management, Innovation Management and Industrial Engineering,*Vol. 1, pp. 175–178.

148. Song, M.P. and Gu, G.C. (2004) “Research on Particle Swarm Optimization: A Review”, *Proceedings of the IEEE International Conference on Machine Learning and Cybernetics*, Shangai, China, August 26-29, pp 2236–2241.

149. Sousa, T., Silva, A. and Neves, A. (2004) “Particle Swarm Based Data Mining Algorithms for Classification Tasks”, *Parallel Computing,* Vol. 30, 267–783.

150. Spears, W.M. (1995) “Adapting crossover in evolutionary algorithms”, J. McDonnel, R. Reynolds, D. Fogel (Eds.), Evolutionary Programming IV: Proc. of the Fourth Annual Conf. on Evolutionary Programming, MIT Press, Cambridge, MA, pp. 367–384.

151. Srinivasa, K.G., Venugopal, K.R., and Patnaik, L.M. (2007) “A Self-adaptive Migration Model Genetic Algorithm for Data Mining Applications”, *Information Sciences,* Vol.177, pp. 4295–4313.

152. Suganthan,P.N., (1999) ”Particle Swarm Optimiser with Neighbourhood Operator”, *Proceedings of the 1999 Congress of Evolutionary Computation*, Vol. 3, pp.1958–1962. IEEE Press.

153. Syswerda, G. (1989) “Uniform crossover in genetic algorithms”, *Proceedings of the 3rd International Conference on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann, pp. 2–9.

154. Ta-Cheng Chen and Tung-Chou Hsu (2006) "GAs based approach for mining breast cancer pattern", *Expert Systems with Applications,* pp. 674–681.

155. Tang, L. and Wang, X. (2012) "A Hybrid Multi-objective Evolutionary Algorithm for Multi-objective Optimization Problems", *IEEE Transactions on Evolutionary Computation,* Vol.9, pp.1-12.

156. Tang, J., Zhang, G., Lin, B. and Zhang, B. (2010) "A Hybrid PSO/GA Algorithm for Job Shop Scheduling Problem", *Advances in Swarm Intelligence, Lecture Notes in Computer Science,* Springer Berlin, Vol. 6145, pp. 566-573.

157. Toivonen, H. (1996) "Sampling Large Databases for Association Rules", *Proceedings of International Conference on Very Large Databases (VLDB),* Mumbai (Bombay), India, September 3-6, pp. 134-145,

158. Trelea, I. C. (2003) "The particle swarm optimization algorithm: convergence analysis and parameter selection", *Information Processing Letters,* Vol.85, No.6, pp. 317–325.

159. Tripathi, P.K., Bandyopadhyay, K.S., and Pal, S.K. (2007) "Adaptive multi-objective particle swarm optimization algorithm", *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 2281–2288.

160. Uma.S.M,, Rajiv Gandhi, K. and Kirubakaran, E. (2012) "A Hybrid PSO with Dynamic Inertia Weight and GA Approach for Discovering Classification Rule in Data Mining", *International Journal of Computer Applications,* Vol.40, No.17, pp. 32-37.

161. Van den Bergh F and Engelbrecht, A.P. (2002) "A New Locally Convergent Particle Swarm Optimizer", *Proceedings of the IEEE Conference Systems, Man and Cybernetics,* Hammamet, Tunisia, Oct. 6-9, Vol.3.

162. Van den Bergh F and Engelbrecht, A.P. (2004) “A Cooperative Approach to Particle Swarm Optimization*”, IEEE Transactions on Evolutionary Computing,* Vol. 3, No.3, pp.225–239.

163. Varsek, T. Urbancic and Filipic, B. (1993) “Genetic Algorithm in Controller Design and Tuning”, *IEEE Transactions on System, Man, and Cybernetics,* Vol. 23, No. 5, pp. 1330-1339

164. Victoire, T.A.A and Jeyakumar, A.E. (2004) “Hybrid PSO-SQP for Economic Dispatch with Vlave Point Effect”, *Electric Power Systems Research,* Vol.71, No.1, pp. 51-59.

165. Wang, K., Tamg, L., Han, J. and Liu, J. (2002) “Top Down FP-Growth for Association Rule Mining”, *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKKD’02),* Taipei, Taiwan, May 6–8, Vol. 2336, pp.334-340, Springer-Verlag.

166. Weijian Cheng, Jinliang Ding, Weijian Kong, Tianyou Chai, and Joe Qin, J (2011) “An Adaptive Chaotic PSO for Parameter Optimization and Feature Extraction of LS-SVM Based Modelling”, *American Control Conference*, San Francisco, CA, June 29 -July 1, pp. 3263 – 3268.

167. Wenxiang Dou, Jinglu Hu, Hirasawa, K. and Gengfeng Wu (2008) “Quick Response Data Mining Model using Genetic Algorithm”, *SICE Annual Conference*, pp.1214 – 1219.

168. Wijsen, J. and Meersman, R. (1998) “On the Complexity of Mining Quantitative Association Rules*”, Data Mining and Knowledge Discovery,* Vol.2, pp.263-281.

169. Witten, I. H. and Frank, E. (2005) “Data mining: Practical Machine Learning Tools and Techniques*”*. San Francisco, CA: Morgan Kaufmann.

170. Wu Zhao-hui, Zhang Gui-juan, and Liu Xi-yu (2005) "Association Rule Mining based on Simulated Annealing Genetic Algorithm", *Computer Applications,* Vol. 25, No . 5, pp.1009-1012.

171. Xiao, G. (2012) "The Application of Particle Swarm Optimization in Stock Prediction and Analysis", *Lecture Notes in Electrical Engineering,* Vol. 107 LNEE, pp. 659-664.

172. Xiaowei Yan, Chengqi Zhang and Shichao Zhang (2009) " Genetic Algorithm-Based Strategy for Identifying Association Rules without Specifying Actual Minimum Support", *Expert Systems with Applications,* Vol. 36, pp. 3066–3076.

173. Xin, J., Chen, G. and Hai, Y. (2009) "A Particle Swarm Optimizer with Multistage Linearly-Decreasing Inertia Weight", *International Joint Conference on Computational Sciences and Optimization, IEEE,* pp. 505–508.

174. Yamaguchi, T. and Yasuda, K. (2006) "Adaptive Particle Swarm Optimization: Self-coordinating Mechanism with Updating Information", *Proceedings of IEEE International Conference on System, Man, Cybernatics,* Taipei, Taiwan, pp. 2303–2308.

175. Yan, X., Zhang, C. and Zhang, Sh. (2008) "Genetic Algorithm-based Strategy for Identifying Association Rules Without Specifying Actual Minimum Support", *Expert Systems with Applications*, Vol.36, No.2, pp. 3066–3076.

176. Yang, G, (2010) "Mining Association Rules from Data with Hybrid Attributes based on Immune Genetic Algorithm", *Proceedings of 7th International Conference on Fuzzy Systems and Knowledge Discovery,* pp. 1446-1449.

177. Yao, X., Liu, Y. and Lin, G. (1999) “Evolutionary Programming Made Faster”. *IEEE Transactions on Evolutionary Computation*, Vol. 3, No.2, pp. 82–102.

178. Zaki, M.J., Parthasarathy, S., Ogihara, M. and Li, W. (1997) “New Algorithms for Fast Discovery of Association Rules”, *Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD),* Newport Beach, California , August 14–17, pp. 283-286.

179. Zhan, Z.H., Xiao, J., Zhang, J. and Chen, W.N. (2007) “Adaptive Control of Acceleration Coefficients for Particle Swarm Optimization based on Clustering Analysis”, *Proceedings of IEEE Congress on Evolutionary Computation,* Singapore, pp 3276–3282.

180. Zhang,Y. and Huang,S. (2004) “A novel multiobjective particle swarm optimization for buoys-arrangement design”, *Proceedings of the Intelligent Agent Technology, IEEE,* pp. 24–30.

181. Zhao, Q. and Bhowmick, S.S. (2003) “Association Rule Mining: A Survey”, Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116 .

182. Zhu Yu, Zhang Hong, and Kong Ling-dong (2009) “ Research and Application of Multi-dimensional Association Rules Mining Based on Artificial Immune System”. *Computer Science*, Vol. 36 No. 8, pp.239-242.

183. Zitzler, E., and Thiele, L. (1998) “An evolutionary algorithm for multi-objective optimization: The Strength Pareto approach”, *Technical Report 43*, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology, Zurich, Switzerland, May 1998.

LIST OF PUBLICATIONS

International Journals

1) **K.Indira,** S.Kanmani, "Performance Analysis of Genetic Algorithm for Mining Association Rules", International Journal of Computer Science Issues, Vol. 9, Issue 2, No 1, 368-376, March 2012

2) **K.Indira,** S.Kanmani, " Rule Acquisition using Genetic Algorithm", Journal of Computing, Volume 4, Issue 5, 128-133, May 2012

3) **K.Indira,** S.Kanmani, "Enhancing Particle Swarm optimization using chaotic operators for Association Rule Mining", Elixir Computer Science & Engineering Journal, 46 ,8563-8566, 2012

4) **K.Indira,** S.Kanmani, "Association Rule Mining by Dynamic Neighborhood Selection in Particle Swarm Optimization", International Journal of Advanced Information Science and Technology , Vol.7, No.7, November 2012

5) **K.Indira**, S.Kanmani, "Association Rule Mining using Self Adaptive Particle Swarm Optimization'', IJCA Special Issue on Computational Intelligence & Information Security CIIS, 27-31, 2012

6) **K.Indira,** S.Kanmani, "Measures for Improving Premature Convergence in Particle Swarm Optimization for Association Rule Mining", communicated to ACTA - International journal of Computers and Applications.

7) **K.Indira,** S.Kanmani, "Mining Association Rules using Hybrid Genetic Algorithm and Particle Swarm Optimization Algorithm (GPSO)", Accepted in International Journal of Data Analysis techniques and strategies, Inderscience Publishers.

8) **K.Indira,** S.Kanmani, “Association Rule Mining through Adaptive Parameter Control in Genetic Algorithm and Particle Swarm Optimization”, Accepted in Computational Statistics, Springer.

9) **K.Indira**, S.Kanmani, “Measures for Improving Premature Convergence in Particle Swarm Optimization for Association Rule Mining”, Communicated to ACTA - International journal of Computers and Applications.

10) **K.Indira**, S.Kanmani, “Review of Particle Swarm Optimization : Basic Concepts, Variants and Applications “, Communicated to IOS Press - Intelligent Data Analysis.

International Conferences

1) **K.Indira**, S.Kanmani, “Framework for Comparison of Association Rule Mining Using Genetic Algorithm”, International Conference On Computers, Communication & Intelligence , 2010.

2) **K.Indira**, S.Kanmani, “Mining Association Rules Using Genetic Algorithm: The role of Estimation Parameters”, International conference on advances in computing and communications, Communication in Computer and Information Science, Springer LNCS, Volume 190, Part 8, 639-648, 2011

3) **K.Indira**, S.Kanmani , Gaurav Sethia.D, Kumaran.S, Prabhakar.J, “Rule Acquisition in Data Mining Using a Self Adaptive Genetic Algorithm”, First International conference on Computer Science and Information Technology, Springer LNCS Volume 204, Part 1, 171-178, 2011.

4) **K.Indira,** S.Kanmani, Prasanth, Harish, Jeeva, “Population Based Search Methods in Mining Association Rules”, Third International Conference on Advances in Communication, Network, and Computing – CNC 2012, LNICST pp. 255–261, 2012.

5) **K.Indira,** S.Kanmani, Aswini, Rangalakshmi, Sumithra, Divyamary, “Mining Association Rules using Adaptive Particle Swarm Optimization”, Third International Conference on advances in IT and Mobile Communication**,** Bangalore April 26-27, 2013. (Accepted).

CURRICULUM VITAE

K.INDIRA, the author of this thesis, is currently doing research in full-time in the Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry, India for the past three years. Previously, she had been working as a Assistant Professor in the Department of Computer Science and Engineering, E.S. College of Engineering and Technology, Affiliated to Anna University, Chennai.

She received her B.E. degree from Madurai Kamaraj University in 1994 and her M.E. degree in 2005 from Department of Computer Science and Engineering, FEAT, Annamalai University, Chidambaram. Her areas of interest include Artificial Intelligence, Knowledge Based Systems and Swarm Intelligence.