A HYBRID OF ANT COLONY OPTIMIZATION ALGORITHM AND SIMULATED ANNEALING FOR CLASSIFICATION RULES

RIZAUDDIN SAIAN

DOCTOR OF PHILOSOPHY
UNIVERSITI UTARA MALAYSIA
2013

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Pengoptimuman koloni semut (ACO) adalah pendekatan metaheuristik yang diilhamkan daripada tingkah laku semulajadi semut dan boleh digunakan untuk menyelesaikan pelbagai masalah pengoptimuman kombinatorik. Masalah penginduksian petua klasifikasi telah diselesaikan dengan algoritma *Ant-miner*, satu varian ACO, yang diketengahkan oleh Parpinelli dalam tahun 2001. Kajian lepas menunjukkan bahawa ACO adalah teknik mesin pembelajaran yang berkesan untuk menjana petua klasifikasi. Walau bagaimanapun, *Ant-miner* kurang pemfokusan terhadap kelas kerana petua untuk kelas diberi selepas petua dibina. Terdapat juga kes di mana *Ant-miner* tidak dapat mencari sebarang penyelesaian optima bagi sesetengah set data. Oleh itu, tesis ini mencadangkan dua algoritma varian hibrid ACO dengan simulasi penyepuhlindapan (SA) untuk menyelesaikan masalah induksi petua pengelasan. Algorithm pertama menggunakan SA untuk mengoptimumkan penemuan peraturan oleh setiap semut. Set data tanda aras dari pelbagai bidang telah digunakan untuk menguji algoritma yang dicadangkan. Keputusan eksperimen yang diperolehi daripada algoritma yang dicadangkan ini adalah setanding dengan keputusan *Ant-miner* dan beberapa algorithma induksi petua terkenal yang lain dari segi ketepatan petua, dan menunjukkan keputusan lebih baik dari segi saiz petua. Algoritma kedua pula menggunakan SA untuk mengoptimumkan pemilihan istilah semasa pembinaan petua. Algoritma ini juga menetapkan kelas sebelum pembinaan setiap petua. Penetapan awal kelas membolehkan penggunaan fungsi heuristik dan fungsi kecergasan yang lebih mudah. Keputusan eksperimen algoritma kedua adalah lebih baik berbanding dengan algoritma lain yang diuji, dari segi ketepatan ramalan. Kejayaan dalam menghibridkan algoritma ACO dan SA telah membawa kepada peningkatan keupayaan pembelajaran ACO untuk pengelasan. Oleh itu, model klasifikasi dengan kebolehan ramalan yang lebih tinggi untuk pelbagai bidang boleh dijana.

Kata Kunci: Pengoptimuman koloni semut, Simulasi penyepuhlindapan, *Ant-miner*, Penginduksian petua

Abstract

Ant colony optimization (ACO) is a metaheuristic approach inspired from the behaviour of natural ants and can be used to solve a variety of combinatorial optimization problems. Classification rule induction is one of the problems solved by the Ant-miner algorithm, a variant of ACO, which was initiated by Parpinelli in 2001. Previous studies have shown that ACO is a promising machine learning technique to generate classification rules. However, the Ant-miner is less class focused since the rule's class is assigned after the rule was constructed. There is also the case where the Ant-miner cannot find any optimal solution for some data sets. Thus, this thesis proposed two variants of hybrid ACO with simulated annealing (SA) algorithm for solving problem of classification rule induction. In the first proposed algorithm, SA is used to optimize the rule's discovery activity by an ant. Benchmark data sets from various fields were used to test the proposed algorithms. Experimental results obtained from this proposed algorithm are comparable to the results of the Ant-miner and other well-known rule induction algorithms in terms of rule accuracy, but are better in terms of rule simplicity. The second proposed algorithm uses SA to optimize the terms selection while constructing a rule. The algorithm fixes the class before rule's construction. Since the algorithm fixed the class before each rule's construction, a much simpler heuristic and fitness function is proposed. Experimental results obtained from the proposed algorithm are much higher than other compared algorithms, in terms of predictive accuracy. The successful work on hybridization of ACO and SA algorithms has led to the improved learning ability of ACO for classification. Thus, a higher predictive power classification model for various fields could be generated.

Keywords: Ant colony optimization, Simulated annealing, Ant-miner, Rule induction

Acknowledgement

First and foremost, the author would like to express his gratitude to Allah S.W.T., who has permitted him to complete this thesis.

The author gratefully acknowledges his supervisor, Prof. Dr. Ku Ruhana Ku Mahamud, who has patiently supervised his work; for her continuous encouragement, patience, guidance and promptness in expecting this academic work to anchor the voyage.

The author also wishes to express his gratitude to Ministry of Higher Education and Universiti Teknologi MARA for the study leave granted.

To his beloved wife, Dr. Zeti Zuryani Mohd Zakuan and his three princesses; Rini Barizah, Rini Bazilah and Rini Basyirah; the author appreciates their understanding and for being there with him while he is sailing through the arduous journey.

To his family and friends; the author values their words of encouragement.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

ACO	Ant colony optimization
AD	Air defense
ANN	Artificial neural network
ASA	Adaptive simulated annealing
C2	Command and control
DFR	Distribution feeder reconfiguration
DGs	Distributed generators
GA	Genetic algorithm
IIR	Infinite-impulse-response
IR	Information retrieval
ML	Maximum likelihood
MMAS	Max-Min ant system
MSER DFE	Minimum symbol-error-rate decision feedback equalizer
ODP	DMOZ Open Directory Project
PSO	Particle swarm optimization
SA	Simulated annealing
SAM	Surface to air missile
STWTSDS	Single machine total weighted tardiness with sequence-dependent setups
TAP	Target assignment problem
TS	Tabu search
TSP	Travelling salesman problem
Web->KB	CMU World Wide Knowledge Base

CHAPTER ONE
INTRODUCTION

The tremendous growth in computing power and storage capacity, the availability of increased access to data from Web navigation and intranets, the explosive growth in data collection, the storing of the data in data warehouses, and the competitive pressure to increase market share in globalized economy stimulated the development of data mining. Data mining acts as a tool to extract or yield important information from raw data.

Classification is a data mining task of finding the common properties among different objects and classifying the objects into classes. Figure 1.1 depicts the general framework of classification task. The classification model contains a set of classification rules. The classification model categorizes new unseen example data, by predicting a class label for the example. One way of presenting the classification model is by representing the information as a set of IF-THEN rules (classification rules).

Figure 1.1: Classification Task General Framework

Rule induction technique (Han, Kamber, & Pei, 2011; Witten, Frank, & Mark A., 2011; Tan, Steinbach, & Kumar, 2006) generates the set of classification rules, for the classification model. Each rule in the classification model consists of conditions and class. The condition part includes one or more terms. Each term is a triple <attribute, operator, value>, where value is one of the values belonging to the domain of attribute. One of the emerging approaches for rule induction is through the use of ant colony optimization (ACO) (Dorigo & Stützle, 2004), specifically the Ant-miner algorithm (Parpinelli, Lopes, & Freitas, 2002a, 2002b).

In ACO, each ant incrementally constructs a candidate solution, associated with a path, to a given optimization problem. As each ant follows a path, it deposits pheromone onto that path. Therefore, the amounts of pheromone level for components that make up that path are increased. An ant will choose the next component for the current partial solution based on the density of the pheromone and heuristic function associated with the components to choose.

In Ant-miner algorithm, each ant will create an artificial path, which represents a candidate of classification rule. The original experiment conducted by Parpinelli et al. (2002a, 2002b) fixed one ant for each ant colony. Ant-miner creates each rule by adding one term each time. When there are no more possible terms left, terms addition will stop. A fitness function will determine the quality of a rule. Next, a pruning procedure will remove the irrelevant terms from the rule. Rule pruning avoids the problem of over-fitting to the training data set, and improves the simplicity of the rule. The rule pruning procedure will iteratively remove one term at

a time from the rule while the rule's quality increases. Hence, rule pruning will improve the rule's quality.

The better the quality of a rule constructed by the ant, the higher the pheromone level to the selected terms for that rule. Therefore, the best term will have higher pheromone level. Since ants tend to choose term with higher probability, the chances for the consequent ants to choose the best term increase. The next ant colony repeats the same processes as the previous ant colony, but with new level of pheromones based on the discovered rule by the previous ant colony.

In order to discover more rules, Ant-miner removes all the examples covered by the current discovered rule. Ant-miner repeated these processes until the total number of examples in the training set is less than a threshold value, which is predefined beforehand. Ant-miner uses rule-based ordering scheme. Rule based ordering scheme orders the individual rules by some quality measure. Ant-miner uses the order of rule's discovery to order the rules. After each rule discovery, Ant-miner appends the new rule to the end of the list of discovered rules. Finally, when Ant-miner has discovered all the rules, Ant-miner appends a default rule to the end of the discovered rules list. The default rule is a rule that contains only class without the condition part. The class for the default rule is a class with the majority counts of the left uncovered examples.

Ant-miner used a rule discovery method based on sequential covering algorithm. The sequential covering algorithm is a common technique used to extract rules directly from data. Even though rules are grown in a greedy fashion based on an evaluation

measure, the used of pheromones will potentially increase the quality of the discovered rule. In order to classify new unseen example, Ant-miner compares the set of discovered rules, in the order of the rule's discovery, with the new unseen example. If a rule covers the new unseen example, Ant-miner sets the rule's class as the predicted class for the new unseen example. Else, if there is no rule covers the new example, a default rule is applied. Hence, the predicted class of the new unseen example used the class of the default rule.

The first version of Ant-miner can only cope with categorical (discrete) attributes. Hence, the operator for the term in conditions for the discovered rule will always be a "=". If there is any continuous attributes exist, before Ant-miner can discover rules, the continuous attributes is discretized (Dougherty, Kohavi, & Sahami, 1995) in the pre-processing step.

Ant-miner (Parpinelli et al., 2002a, 2002b) has been compared to two rule induction algorithms in several UCI public domain data sets (Asuncion & Newman, 2007), which are CN2 and C4.5 (Quinlan, 1993) algorithms. The results, showed that the predictive accuracy of Ant-miner is not only at least at par with both compared algorithm, but also produced much more comprehensive rules.

1.1 Problem Statement

Rule induction extracted a set of IF-THEN rules from training data using a sequential covering algorithm (Tan et al., 2006). A sequential covering algorithm learns each rule sequentially, one at a time; by choosing terms that cover a class or by choosing interrelated terms and later on assign a most suitable class for the set of

terms using a greedy search. In greedy search, while constructing a rule, term that appears to be the best choice at the moment is added to the partial rule heuristically. The problem with greedy search is that the heuristic measure does not ensure whether the term chosen is the best choice. The usage of ACO, which consists of an element of memory called the pheromone, can lessen the chance of this happening. However, rule induction algorithm that is based on ACO also has some drawbacks that may reduce the prediction ability of the classification model constructed. The problems were discussed in the following paragraphs.

The first Ant-miner algorithm introduced by Parpinelli et al. (2002a, 2002b) assigned the class after each ant constructs the rule antecedent. In other words, the class is unknown during rule's construction. Since the class is unknown during the rule construction, two problems arose. First, the constructed rule might contain antecedent that are not highly related, since the algorithm calculates the heuristic based on the relationship between terms, but disregarding the class. Second, the pheromone level for each term is based on the differences amount from all classes, rather than to a specific class. These two factors make the original Ant-miner less focused relevance. Therefore, many researchers (Galea & Shen, 2006; Martens, De Backer, Haesen, Baesens, & Holvoet, 2006; Smaldon & Freitas, 2006) proposed that the class should be fixed initially.

There is also the case where the Ant-miner sometimes cannot find any optimal solution for some data. For example, Parpinelli et al. (2002a, 2002b) reported that only four out of seven data sets tested were better than C4.5 algorithm. This shows that for some data sets, the ants may face the local optimization problem, because

Ant-miner does not use local search. Simulated annealing algorithm (SA) is a well-known algorithm, statistically guaranteed, to find an optimal solution. Therefore, this thesis suggested the used of SA algorithm to improve the current implementation of ACO in rule induction.

1.2 Research Objectives

The main objective of this research is to develop enhanced Ant-miner algorithms for rule induction. The specific objectives of the research are:

i. to determine the best attribute selection methods for dimensionality reduction,

ii. to formulate a technique in optimizing the rules constructed by each ant, and

iii. to construct a new method in optimizing terms selection.

1.3 Significance of the Research

Data mining classification task is a form of data analysis that extracts classification models to categorize future data trends. A classification model helps human to understand data better, especially a large data. For example, a classification model can help a finance bank manager to categorize a loan application into safe or risky.

A rule induction algorithm must generate an efficient classification model with a set of simple, comprehensible, and accurate rules. Hence, the work done in this thesis is important in the sense that this thesis develops classification rules induction

algorithm that combines the robust ACO with the efficiency of SA to improve the efficiency of the resultant classification model.

1.4 Scope, Assumptions and Limitations of the Research

This thesis focused on using ACO as the learn-one-rule function to construct classification rules. The enhanced proposed algorithm used simulated annealing (SA) to reduce the problem of local optimization. This research however, not in all means to optimize the SA itself. In addition, this research developed rule induction algorithm that can only classify examples with single class.

All experiments in Chapter Five and Six were carried out using seventeen data sets from Asuncion & Newman (2007) from various fields. The tested data sets include Balance Scale, Breast Cancer (Ljubljana), Breast Cancer (Wisconsin), Credit-a, Credit-g, Diabetes, Heart (Cleveland), Heart (Statlog), Hepatitis, Ionosphere, Iris, Lymphography, Mushroom, Segment, Sonar, Tic-Tac-Toe and Vehicle. The proposed methods were compared with the original Ant-miner (Parpinelli et al., 2002a, 2002b) and PART (Frank & Witten, 1998) in terms of classification accuracy and simplicity of the constructed rules. A ten cross folds validation method (Kohavi, 1995a) measure the classification accuracy and simplicity of the constructed rules. However, this thesis does not measure and optimize the efficiency of the proposed algorithm in terms of time taken for training and prediction.

Like the original Ant-miner, proposed by Parpinelli et al (2002a, 2002b), both proposed methods cannot directly handle data sets with non-discrete attributes. This thesis uses the words “new” and “proposed” interchangeably, to represent the new

algorithms. A discretization method (Dougherty et al., 1995) discretized the non-discrete attributes, and was discussed in Chapter Three.

1.5 Structure of the Thesis

This thesis has seven chapters, including the introductory chapter, which covers the background information that inspires the enhancement of ACO classification modelling.

Chapter Two covers the literature review of the related studies on ACO. The second section of this chapter focuses on various studies on the usage of ACO. The third section discussed the usage of ACO in rule induction. Finally, a review on SA, its applications and the hybrid of ACO and SA were given.

Chapter Three presents the methodology used in conducting this research, divided into two main sections. The first section is on how to pre-process the data sets, while the second section is on the validation process of the experiments.

Chapter Four discusses the experiments and results on determining the best attribute selection for high dimension data. Ant-miner algorithm (Parpinelli et al., 2002a, 2002b) constructed the classification model for this chapter. The large data set used was based on Web->KB project (http://www.cs.cmu.edu/~webkb/) from four (4) universities namely Cornell (867 pages), Texas (827 pages), Washington (1205 pages) and Wisconsin (1263 pages) as well as 4,120 miscellaneous pages collected from other universities. This chapter also tested the performance of Ant-miner on several data sets from UCI database after applying the reduction method.

Chapter Five presents a new hybrid algorithm of ACO and SA. This chapter introduces a local search algorithm using SA, to optimize the rules constructed by each ant. The proposed algorithm in this chapter follows specific-to-general rule-growing strategy to grow classification rule. The algorithm ordered the discovered rules in the order of their discovery. This chapter compares the results with the original Ant-miner and other several well-known rule induction algorithms, in terms of classification accuracy and simplicity, using the original and also attributes reduced data sets.

Chapter Six presents the second proposed hybrid algorithm. Differ from the algorithm proposed in Chapter Five; this algorithm instead uses SA to optimize terms selections while each ant grows one rule. The proposed algorithm in this chapter fixes the class while growing classification rules. The algorithm ordered the discovered rules in descending order according to the quality its quality. This chapter compares the results with the original Ant-miner and other several well-known rule induction algorithms, in terms of classification accuracy and simplicity, using the original and also attributes reduced data sets, as in Chapter Five.

Finally, Chapter Seven gives the concluding remarks on the two proposed hybrid algorithms. The concluding remarks include the description of features, capabilities and the weaknesses of the two proposed methods. The chapter presents some recommendations as guidelines for further research for using ACO for rule induction.

CHAPTER TWO
LITERATURE REVIEW

This chapter reviews the literature related to the research field considered in the thesis. Section 2.1 introduces data mining and classification problem, while Section 2.2 presented several well-known and established rule induction algorithms. Section 2.3 presents ant colony optimization (ACO), while Section 2.4 discussed the applications of ACO. Section 2.5 discussed on the variations of the implementation of ACO for rule induction. Section 2.6 discussed on simulated annealing (SA) algorithm and its applications. Section 2.7 discussed on the implementation of hybrid ACO for rule induction. Finally, Section 2.8 underlay the summary for this chapter.

2.1 Data Mining and Classification

Data mining is a multidisciplinary field whose core is at the intersection of machine learning, statistics and databases that refers to extracting or `mining' knowledge from large amounts of data (Han et al., 2011). Cabena et al. (1998) defined data mining as an interdisciplinary field bringing together techniques from machine learning, pattern recognition, statistics, databases, and visualization to address the issue of information extraction from large data bases. According to Larose (2005), data mining development has been stimulated by various factors: the explosive growth in data collection, the storing of the data in data warehouses, the competitive pressure to increase market share in globalized economy, the tremendous growth in computing power and storage capacity, and the availability of increased access to data from Web navigation and intranets.

There are several tasks in data mining such as classification, clustering, association rules and attribute selection (Han et al., 2011; Larose, 2005). The tasks should be chosen based on the problem that arise, and will be solved by using an appropriate algorithm.

A classification task uses a classification algorithm to a set of training examples to construct a prediction model. The prediction model can predict a class for a given unknown example. Rule induction is a popular classification technique to construct a prediction model from a set of data. The prediction model constructed by rule induction is comprehensible to users and is presented in a high level language.

2.2 Classification Using Rule Induction

Rule induction technique (Han et al., 2011; Witten et al., 2011; Tan et al., 2006) generates a set of classification rules, for the prediction model. The prediction model consists of a set of IF-THEN rules. The set of IF-THEN rules is called the classification rules.

A classification rule is represented in the form of "IF <*condition*> THEN <*class*>". A *condition* consists of a set of attribute terms "<*attribute*, *operator*, *value*>". The *operator* is an equal sign "=" and the *class* is the predicted value for example that satisfies the *condition*. Each attribute consists of more than one value, but can only assigned one value in each rule.

For example, a weather data set might contain a few attributes: outlook, temperature, humidity and a class. This data set will predict whether we should go out and play or

not based on the weather. Figure 2.1 depicts an example of a classification rules for weather data set.

```
If outlook = overcast then yes
If humidity = normal  and windy = false then yes
If temperature = mild and humidity = normal then yes
If outlook = rainy  and windy = false then yes
If outlook = sunny  and humidity = high then no
If outlook = rainy  and windy = true then no
```

Figure 2.1: An Example of a Classification Rules.

A well-known and established rule induction algorithms are OneR (Holte, 1993), ridor (Gaines & Compton, 1995), PART (Frank & Witten, 1998), JRip (Cohen, 1995), decision table (Kohavi, 1995b), DTNB (M. Hall & Frank, 2008) and conjunctive rule (Witten et al., 2011).

OneR algorithm (Holte, 1993), also known as one rule algorithm is a simple algorithm that builds one rule for each of the attributes in the training examples. This algorithm selects the rule with the smallest error rate as its 'one rule' and creates a rule by calculating the most frequent class for each attribute (the class that appears most often for that attribute value). In other words, a rule is a set of attribute values bound to their majority class, and has the lowest error rate. If there are more than one rules that have the same error rate, the rule is chosen at random.

Ridor algorithm (Gaines & Compton, 1995) is the implementation of a RIpple-DOwn Rule learner (Compton & Jansen, 1990). This algorithm generates a default rule first and then the exceptions for the default rule with the least (weighted) error

rate. Then it generates the “best” exceptions for each exception and iterates until pure. Thus, it performs a tree-like expansion of exceptions. The exceptions are a set of rules that predict classes other than the default.

PART (Frank & Witten, 1998) is a separate-and-conquer rule learner. This algorithm use a set of ordered of rules to predict the class of a new unseen example. A new example is compared to each rule in the list one by one until a rule is found satisfies the new example. If there is no rule that satisfy the new example, a default rule is used. PART builds a partial C4.5 (Quinlan, 1993) decision tree in each iteration. The best tree leaf is taken and converted into a rule in order of discovery. The algorithm is a combination of C4.5 and RIPPER rule learning.

JRip algorithm (Cohen, 1995) implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER), as an optimized version of IREP. Ripper builds a rule set by repeatedly adding rules to an empty rule set until all positive examples are covered. Rules are formed by greedily adding conditions to the antecedent of a rule (starting with empty antecedent) until no negative examples are covered. After a rule set is constructed, the algorithm optimizes the rule to reduce its size and improve the accuracy.

Decision table algorithm (Kohavi, 1995b) builds and using a simple decision table majority classifier to build the classification model. This algorithm summarizes the examples in the data set using a ‘decision table’ which contains the same number of attributes as the original data set. In order to predict the class for a new example, this algorithm finds the line in the decision table that matches the non-class values of the

new example. Decision table employs the wrapper method to find a good subset of attributes for inclusion in the table. The algorithm reduces the likelihood of over-fitting and creates a smaller and condensed decision table, by eliminating attributes that contribute little or nothing to a model of the data set.

DTNB (M. Hall & Frank, 2008) is a decision table/naive Bayes hybrid classifier. At each point in the search, the algorithm evaluates the merit of dividing the attributes into two disjoint subsets: one for the decision table, the other for naive Bayes. A forward selection search is used, where at each step, naive Bayes model selected attributes and the remainder by the decision table and all attributes are modelled by the decision table initially. At each step, the algorithm also considers dropping an attribute entirely from the model.

Finally, conjunctive rule (Witten et al., 2011) is a rule induction algorithm which uses a single conjunctive rule learner that can predict for numeric and nominal class labels. The consequent in this algorithm using the distribution of the available classes in the data set. When predicting new example, if the test example is not satisfied by any rules, the test example is predicted using the default class distributions. The default class distribution is class defined from the examples that are not covered by any rules during training phases. During training phases, an antecedent is selected by computing information gain of each antecedent. The information of one antecedent is the weighted average of the entropies of both the examples covered and not covered by the rule. Each constructed partial rule, which contains antecedent only, is pruned afterwards using Reduced Error Prunning (REP)

procedure by calculating a weighted average of the accuracy rates on the pruning data, apart from a simple pre-pruning based on a number of antecedents.

One of the emerging approaches for rule induction is through the use of ant colony optimization (ACO) (Dorigo & Stützle, 2004), specifically the Ant-miner algorithm (Parpinelli et al., 2002a, 2002b).

2.3 Ant Colony Optimization Metaheuristic

ACO Metaheuristic is a branch of an artificial intelligence technique called swarm intelligence, introduced in the early 1990s (Dorigo, Maniezzo, & Colorni, 1991, 1996). It is a probabilistic technique for solving combinatorial optimizations problem, which was inspired by the behaviour of cooperating ants in finding path from its nest to the food source (Dorigo & Stützle, 2004). Hence, the purpose of ACO metaheuristic is to find the best solution using a set of artificial ants that communicates indirectly using an item called the pheromone.

Ants, which are blind or almost blind creatures, are capable to find the shortest path between the nest and the food source, and they have the capability to adapt to environment change. Ants tend to follow a trail, used by the majority ants, although they initially move at random. According to the double bridge experiments (Deneubourg, Aron, Goss, & Pasteels, 1990; Goss, Aron, Deneubourg, & Pasteels, 1989), performed on the Argentine ant species *I*, communication between ants is established through the use of chemical substances called pheromone, which was deposited as each ant move. The amount of pheromone on one of the trails used by the majority ants will increase as time passed by and will decrease on the less used

trails. As a consequent, since ants tend to follow the pheromones, all or most of the ants will finally converge to the best trail, with the high density of pheromones, which happened to be the shortest trail from the nest to the food source.

Figure 2.2(a) is a schematic diagram of the double bridge experiment set up with two branches of equal length. The double bridge experiment released ants from the nest and let them freely moved to the food source. Initially, each ant chooses any branch at random since it has no preference on any branch. While walking, ants will deposit pheromones. After some time, pheromones density on one of the branches will be greater than the other one, and will stimulate ants to use it. Eventually, all ants will use only one of the branches. The chosen branch is the one with the highest density of pheromones. This experiment proved that ants would follow each other based on a communication instrument, called the pheromone.

On the other hand, in Figure 2.2(b), the double bridge is set up with one of the branch is twice the length of the other branch. The experiment found that, finally, most of the ants used the shorter branch, since the pheromones accumulate faster in the shorter branch. However, there are a small number of ants still using the longer branch. This is the effect of “path exploration”, where the density of the pheromone will not bias some of the ants.

The experiment by Goss et al. (1989) also found that even though the pheromones evaporate, it evaporates too slowly to allow the ant colony to forget the current branch that they are using. For example, the ant colony will still favour the longer branch instead of the short branch, even though the shorter branch was added long

after the ant colony has been using the longer branch,. Hence, it shows that pheromone plays an important role in determining which branch the ant colony will choose. The path exploration cannot defeat the power of pheromone, at least in shorter time.

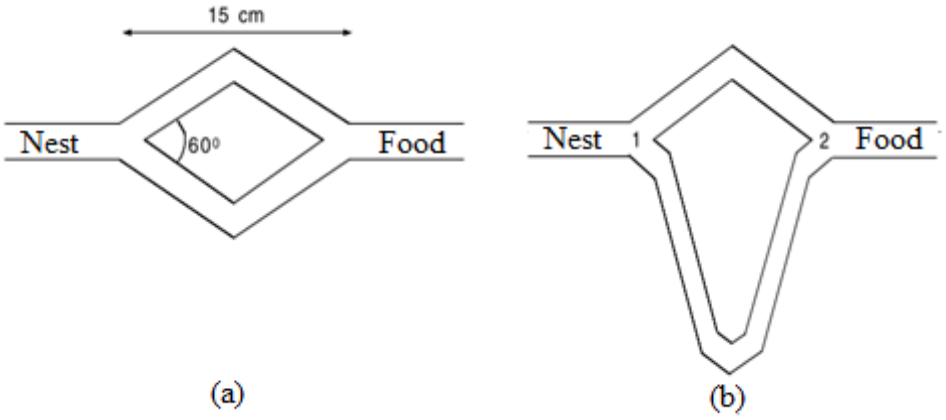

Figure 2.2: Experimental Setup for the Double Bridge Experiment.
(a) Branches have equal length. (b) Branches have different length.
Adapted from Dorigo & Stützle (2004)

2.4 Applications of Ant Colony Optimization

There are various problems of using ACO for optimization. Some of the examples are routing, assignment, scheduling, subset and machine learning. The following subsection will discuss generally, the implementation of ACO on routing, assignment, scheduling, subset, and machine learning.

Routing is a process of sending objects to appropriate destinations via a specific route. The routing problem that has been solved by ACO includes travelling

salesman (Dorigo & Gambardella, 1997; Dorigo et al., 1996), vehicle routing (Reimann, Doerner, & Hartl, 2002) and sequential ordering (Gambardella & Dorigo, 1997, 2000).

The Travelling Salesman Problem (TSP) is a problem to find the shortest possible trip, for travelling from a city to its destination. In this problem, each ant, which is located to a random start city, initially, will develop the solution by visiting all cities in its tour.

The vehicle routing problem is an extension to the TSP problem. The objective of this problem is to determine a fleet of vehicles, each vehicle served customers, and in what order each vehicle should visit the customers assigned to.

The sequential ordering problem is a problem to find a minimum weight Hamiltonian path on a directed graph with weights on the arcs and the nodes, subject to precedence constraints between nodes. The solutions built by ants iteratively are by adding, step-by-step, new unvisited nodes to become the partial solution. The ant added new nodes by using pheromone trails, heuristic, and constraint information.

Besides TSP, vehicle routing, sequential ordering and telecommunication network routing problems have also been using ACO. In telecommunication network, routing is a process of moving packets of data from source node to the destination node. Hence, telecommunication network routing is a problem of finding minimum cost paths among all pairs of nodes in the network. In this type of problem, the algorithm assigned each ant a source and destination node. Each ant will move to the

destination node by moving from one node to another. At each node, the ant will choose to move to the next node using a probabilistic decision rule, which is a function of the ant's memory, of local pheromones, and heuristic information.

Examples of ACO applications that have been applied to the problem of network routing are connection-oriented network routing (Bonabeau et al., 1998), connectionless network routing (Caro & Dorigo, 1997) and optical network routing (Varela & Sinclair, 1999).

The second problem type is assignment. The assignment problem is one of the fundamental combinatorial optimization problems in the branch of optimization or operations research in mathematics. It consists of finding a maximum weight matching in a weighted bipartite graph.

For example, in quadratic assignment problem of *n* order, the objective is to find the best allocation of *n* activities to *n* locations. Generally, ACO will solve this kind of problem by choosing the components to add to the partial solution by using the pheromone trails and heuristic information.

Some examples of the assignment problem include graph colouring (Bui & Nguyen, 2006), quadratic assignment (Maniezo, Colorni, & Dorigo, 1994; Maniezzo & Colorni, 1999), generalized assignment (Lourenço & Serra, 1998), frequency assignment (Maniezzo & Carbonaro, 2000), and university course timetabling (Socha, Knowles, & Sampels, 2002).

Scheduling is a process of assigning priorities to processes in a priority queue. An example of the application of ACO in scheduling problem is job shop (Teich, Fischer, Vogel, & Fischer, 2001). The job shop scheduling problem is part of production planning. The ACO decides which operation is scheduled next, where each ant will generate a feasible solution under the use of a transition rule.

Other examples of the used of ACO in scheduling problem include open shop (Blum & Sampels, 2002), flow shop (Stützle, 1998), total tardiness (Bauer, Bullnheimer, Hartl, & Strauss, 2000), group shop (Blum, 2002a), project scheduling (Merkle, Middendorf, & Schmeck, 2002) and total weighted tardiness (Besten, Stützle, & Dorigo, 2000).

Subset is a problem in complexity theory and cryptography. Some examples of the used of ACO in subset problem includes multiple knapsack (Leguizamón & Michalewicz, 1999), max independent set, redundancy allocation (Liang & Smith, 1999), set covering (Silva & Ramalho, 2001), weight constraint graph tree partition, edge-weighted k-cardinality tree (Blum, 2002b) and maximum clique (Fenet & Solnon, 2003).

For example, in the edge-weighted *k*-cardinality tree problem, ants will construct a solution by building a path on construction graph, and does a number of construction steps. In each step, ACO added edge to the partial *k*-cardinal tree, based on the transition probabilities. The pheromone plays an important role in the generation of the transition probabilities.

Finally, machine learning, a method for creating computer programs by the analysis of data sets, is an area of artificial intelligence concerned with the development of techniques, which allow computers to "learn". The major focus of machine learning is to extract information from data automatically, by computational or statistical methods. For example, an inductive machine learning methods extract rules and patterns out of massive data sets.

Since the fields of statistics also study the analysis of data, machine learning must intersect with statistic field. Nevertheless, machine learning is pertained to the algorithmic complexity of computational implementations, as compared to statistics.

Some examples of machine learning that is based on the ACO include classification rules for data mining (Parpinelli, Lopes, & Freitas, 2001; Parpinelli et al., 2002a, 2002b; Parpinelli, Lopes, & Freitas, 2002c, 2005; Freitas, Parpinelli, & Lopes, 2008; Oliverio, Sá, & Parpinelli, 2009), Bayesian networks and fuzzy systems (Cordón, Casillas, & Herrera, 2000).

This thesis focused on the development and enhancement of the rule induction technique for extracting classification rules from data using ACO. Parpinelli initially introduced this kind of problem in the early 2000s, called the Ant-miner (Parpinelli et al., 2002a, 2002b).

The next section discussed on the variations of implementations of ACO for rule induction.

2.5 Ant Colony Optimization for Rule Induction

Parpinelli et al. (2002a, 2002b) proposed Ant-miner algorithm in 2002. The Sequential Covering algorithm is the based for Ant-miner algorithm, that extract rules directly from data. According to Tan et al. (2006), Sequential Covering algorithm discovers rules in greedy fashion based on a certain evaluation measure. In other words, this algorithm select terms using some heuristic. Ant-miner uses a heuristic measure as evaluation measure to fill in the antecedent part of the rule, by selecting the best term to be included into the partial rule. The heuristic measure is the normalization of entropy measures between terms. The algorithm selects one best rule from a set of discovered rules, based on a quality measure using some fitness function.

Ant-miner differs from other Sequential Covering algorithms implementation because this algorithm also depends on a value called the pheromone, which contributes to the behaviour of exploration of the algorithm. Hence, Ant-miner uses a probability that is proportional to the product of heuristic value and pheromone level for that term, to add terms to a rule. Dorigo & Stützle (2004), called this transition rule, random proportional transition rule.

Ant-miner updated the pheromone level after each ant colony has selected the best rule from a set of rules constructed by many ants in a colony. For the next ant colony, terms in the rule antecedent that have been selected by the previous ant colony will have a higher level of pheromone and will probably are more favoured than other terms.

Ant-miner selects the best rule after each ant colony has created a set of rules, based on rule's quality. This algorithm measures rule's quality using a fitness function that depends on the product of sensitivity and specificity, which were adapted from the field of information retrieval (IR).

Since the class for rules constructed Ant-miner will only be determined after the creation of each rule, the selected terms are not focused relevance. In other words, this algorithm will select a term that may be high in relevancy with the current set of previously selected terms, but not for the later assigned class. Therefore, like other ACO implementations for other fields, Ant-miner might face the problem of stagnation, where optimized rule cannot be found, and thus make the program to run forever. Consequently, many researchers had proposed variations of modification for the original Ant-miner by Parpinelli et al. (2002a, 2002b). The following subsections discussed variations of modification for the original Ant-miner.

2.5.1 Train by Fixing Classes

The original Ant-miner (Parpinelli et al., 2002a, 2002b) determines the class after each rule's construction, and thus may affect the rule's quality. Several researchers (Galea & Shen, 2006; Martens et al., 2006; Smaldon & Freitas, 2006) have proposed that the class should be predefined earlier. If the algorithm uses a predefined class, the algorithm must use a new heuristic function and pheromone update methods.

2.5.2 New Heuristic Functions

The original Ant-miner uses the entropy measure between terms to calculate the heuristic value for each term. Most researches (Galea & Shen, 2006; B. Liu, Abbass, & McKay, 2004; Martens et al., 2006; Smaldon & Freitas, 2006; Z. Wang & Feng, 2005) who predefined the class before each rule's construction, used a simpler heuristic function instead. The new heuristic function uses the frequency of the terms related to the predefined class.

This simpler heuristic function is more computationally efficient since the calculation will only base on the frequency, unlike the original heuristic function, which depends on the calculation of entropy. The heuristic function uses frequency of examples. The researchers who use this simpler heuristic function claimed that even though the heuristic is simple, the use of pheromones would help to optimize the constructed rules.

2.5.3 New Pheromone Updating Procedure

The Ant-miner evaporates pheromone level for all terms implicitly by normalizing them. This happened after the algorithm had increased the pheromone level on all selected terms. However, this procedure sometimes faced problem in updating pheromone level with low quality rules, since the updating procedure depends on the rule's quality. If the quality is very small, the new pheromone level will almost be the same as the previous one.

Hence, Liu et al. (2004), Wang & Feng (2005) and Smaldon & Freitas (2006) have proposed a different equations for updating pheromone level, that could cope with

rule's quality that is close to zero. Furthermore, Liu et al. (2004), Wang & Feng (2005) and Martens et al. (2006) also proposed that the updating procedure should use an explicit evaporation rate, which is predefined. More interestingly, Wang & Feng (2005) had made the parameter self-adaptive.

2.5.4 Pseudorandom Proportional Transition Rule

Instead of random proportional transition rule, some researchers (Liu et al., 2004; Wang & Feng, 2005) have used pseudorandom proportional transition rule. Pseudorandom proportional transition rule defines a new probability. This probability is used to control the effect of exploration and exploitation, based on a predefined constant parameter value, q_0. If the probability number generated is less than q_0, the term to be added will be determined using the same function as random proportional transition rule which depends on the maximum value from the product of heuristic value and pheromone level from all terms. Otherwise, a random proportional rule will probabilistically select terms, based on the product of heuristic value and pheromone level only.

The disadvantage of this transition is that the researcher needs to optimize empirically the predefined constant parameter value, q_0.

2.5.5 Remove Pruning Procedure

Pruning procedure is a computational expensive procedure in rule induction, as well as the Ant-miner algorithm. Hence, in 2006, Martens et al. proposed that pruning

procedure removal is possible. Martens et al. (2006) claimed that the removal of rule pruning procedure improve the speed of the proposed algorithm.

2.6 Simulated Annealing Algorithm

SA is an algorithm developed by Kirkpatrick et al. (1983). The process of heating and gradually cooling process to hardened metal, and combinatorial optimization inspired the concept of SA. SA is an algorithm that has the ability to reduce the problem of local optimum. A local optimum problem is a problem where a solution is optimal within a neighbouring set of solutions. However, considering all set of solutions, this local optimal solution might not be the optimal solution.

SA is a local search strategy that tries to avoid local optimal problem. SA starts at a very high temperature and it gradually cool down to a lower bound temperature. During the cooling process, SA accepts not only good solution, but also worst solution, with some probability. As the temperature reduced, the probability of accepting worst solution also reduced. SA ability to accept the worst solution with some probability helps to reduce the problem of local optimal.

The main problem with SA is its slow speed for convergence (Nikolaev & Jacobson, 2010; Henderson, Jacobson, & Johnson, 2003). However, many researchers have suggested improvement on SA in order to increase its convergence speed. Approaches to improve the SA's speed include combining SA with evolutionary algorithm, fixing the temperature cooling schedule, implementing two-staged SA and using an adaptive temperature control scheme.

The first approach is to combine SA with evolutionary algorithms. Delport (1998) combines SA with evolutionary algorithm by adjusting the cooling schedule using fast recognition of the thermal equilibrium in terms of selection intensity. Another approach that combines SA with evolutionary algorithm is by linking SA with genetic algorithm (GA) using generalized hill-climbing algorithm (Sullivan & Jacobson, 2000, 2001).

The second approach to improve SA's speed is by fixing the temperature cooling schedule. Fielding (2000) suggested a fix temperature cooling schedules. The researcher tested his suggestion on traveling salesman problem, the quadratic assignment problem and the graph partitioning problem. Experiments by Fielding shows that a fixed temperature cooling schedule can improve the speed of SA. In addition, Orosz & Jacobson (2002a, 2002b) have suggested a finite-time performance measures for SA with fixed temperature cooling schedules. The measures used randomly generated instances of the traveling salesman problem.

The third approach to speed up SA is to implement a two-staged SA algorithm, which are high temperature and low temperature. At the high temperature stage, a fast heuristic is used. At the low temperature, the algorithm works as traditional SA. Varanelli & Cohoon (1999) have proposed a method for determining the initial temperature for two-staged SA.

The forth approach is to use an adaptive temperature control scheme. An adaptive temperature control scheme changes temperature based on the number of consecutive improving moves. For example, Azizi & Zolfaghari (2004) have tested

SA with an adaptive temperature control scheme for job shop scheduling problem. According to Azizi & Zolfaghari (2004), SA speeds have improved.

2.6.1 Applications of Simulated Annealing

Many researchers have used SA to solve many discretes and continuous variable problems. Chen et al. (2007) implemented five conversions of SA from sequential-to-parallel forms on high-performance computers and applied them to a set of standard function optimization problems in order to test their performances. The performance analyses of the best algorithm among the five implemented algorithms shows that a hybrid version of a genetic algorithm combined with SA has proven to be most efficient.

Steinhöfel et al. (2002) uses logarithmic cooling schedules of SA to solve flow shop scheduling problem. The implementation uses an objective to minimize the overall completion time (called the makespan). Steinhöfel et al. (2002) prove that a lower bound for the number of steps that are sufficient to approach an optimal solution with a certain probability.

Chen & Luk (1999) proposed an adaptive SA (ASA) to solve difficult non-linear optimization problems of signal processing. Chen & Luk have applied the proposed algorithm to three signal processing applications: maximum likelihood (ML) joint channel and data estimation, infinite-impulse-response (IIR) filter design, and evaluation of minimum symbol-error-rate decision feedback equalizer (MSER DFE). ASA is found to perform better than genetic algorithm.

Abramson et al. (1999) have used SA for solving the school timetabling problems. Six different SA cooling schedules were compared for solving the school timetabling: the basic geometric cooling schedule, a scheme that uses multiple cooling rates, geometric reheating, enhanced geometric reheating, non-monotonic cooling, and reheating as a function of cost. Experiments performed by Abramson et al. (1999) show that the scheme, which uses the phase transition temperature in combination with the best solution quality found to date, produced the best results.

Emden-Weinert & Proksch (1999) report about a study of SA for the airline crew pairing problem based on a run-cutting formulation. It is found that that run time can be saved and solution quality can be improved by using a problem specific initial solution, by relaxing constraints "as far as possible", by combining SA with a problem specific local improvement heuristic and by multiple independent runs.

Koulamas et al. (1994) have found that SA provides a better performance, in terms of number of iterations and neighbourhood functions, for operation research problems. Thus, increase the probability of converging to the optimal solution.

Johnson et al. (1989, 1991) have used SA to solve discrete problems, the graph partitioning problems (Johnson et al., 1989), and graph colouring and number partitioning problems (Johnson et al., 1991). Johnson et al. (1991) have found that SA performs better for the long run lengths to solve graph colouring problems.

2.6.2 Hybrid ACO and SA Algorithm Variants

Researchers have hybrid SA with ACO for many types of problems. Mostly, ACO used SA as a booster. In other words, SA tries to improve the solutions found by ACO implicitly.

Anghinolfi & Paolucci (2008) proposed a hybrid ACO and SA to solve single machine total weighted tardiness with sequence-dependent setups (STWTSDS) problem. SA works as intensification to validate solutions by all ants after each ant colony runs, and select one best solution for the current ant colony run. Experiments show that the effectiveness of ACO increased with the usage of SA as an intensification mechanism. Thus, combining swarm intelligence algorithm such as ACO with SA is a viable strategy to produce in a simple way high quality metaheuristics.

Wang et al. (2011) solved the target assignment problem (TAP) applied to the air defense (AD) command and control (C2) system of surface to air missile (SAM) tactical unit using a hybrid ACO and SA algorithm. The simulation examples show that the model and algorithms can meet the solving requirement of TAP in AD combat. Wang et al. (2011) found that by using ACO can achieves solution quickly, but the solution is sometimes local minimal value, SA needs more time to get the optimal solution. However, the hybrid optimization strategy based on ACO and SA algorithm can meet the need of solving the TAP.

Olamaei et al. (2010) proposed a hybrid of ACO and SA algorithm for solving for distribution feeder reconfiguration (DFR) considering Distributed Generators (DGs).

DGs is used to minimize the real power loss, the deviation of nodes voltage and the number of switching operations. Experiments on a real distribution feeder show that the proposed hybrid ACO and SA algorithm performs better as compared to ACO, particle swarm optimization (PSO) and genetic algorithm (GA).

In 2011, Olamaei et al. enhanced the proposed hybrid of ACO and SA algorithm (Olamaei, Niknam, Arefi, & Mazinan, 2011) to minimize the cost of distributed network operation including the cost of the active power generated by DGs. This algorithm used a cost based compensation method to encourage DGs in active and reactive power generation due to private ownership of DGs. The experiments were done on a real distribution feeder. Olamaei et al. (2011) claimed that this algorithm minimizes the number of the switches. The small number of switches reduced the required space and computation time. The proposed algorithm also reaches a much better optimal solution in comparison with others and has the small standard deviation for different trials.

Ghanbari et al. (2010) predicts short term electricity load using a hybrid ACO and SA algorithm. The hybrid approach consists of two general stages. First, the algorithm fed time series inputs into ACO to perform a global search that find a globally optimum solution. Second, the algorithm used the solution by ACO as initial solution for SA. SA starts local search around the ACO's global optimum and performs the tuning process on the initial solution. Ghanbari et al. (2010) conducted experiments on Iranian monthly load prediction problem and compared the results with ACO, SA and artificial neural network (ANN). Results show that the proposed

algorithm performs better than the compared algorithms regarding to prediction accuracy.

T'Kindt et al., (2000) solved a 2-machine flowshop bicriteria scheduling problem using a hybrid ACO and SA algorithm. The proposed algorithm uses a variant of ACO called the Max-Min Ant System (MMAS) proposed by Stützle (1998). Each ant builds a permutation schedule that has an optimal value of the makespan at each iteration. The proposed algorithm uses a wheel selection method to choose the next job. SA works as intensification to the solution by ACO. Furthermore, the proposed algorithm used a 2-opt local search algorithm to improve the solution furthermore. T'Kindt et al., (2000) have found that the proposed algorithm is able to reduce the resolution time of the hardest problem.

Researchers have also proposed a hybrid of ACO and SA algorithm the problem of cluster analysis (Niknam, Bahmani, & Nayeripour, 2008; Niknam, Olamaei, & Amiri, 2008). The proposed algorithm uses SA to find the best local position for each ant colony runs. The experimental results show that the proposed algorithm performs at least comparable to SA, ACO and k-means in terms of function evaluations and standard deviations for partitional clustering problem.

2.7 Hybrid ACO for Rule Induction

Most recent trends to improve the earlier implementation of ACO for rule induction (Ant-miner) are by hybrid ACO with other algorithm. The other algorithm will try to solve the problematic, less efficient or lack of features parts of ACO in inducting rules, such as the lack of ability to directly cope with continuous attributes.

Particle swarm optimisation (PSO) based classification algorithms perform well in data sets that contain continuous attributes since it has the ability to convert nominal attributes into numerical values. This feature is lacked from the originally proposed ACO based classification algorithm by Parpinelli et al. (2002a). Hence, Holden & Freitas (2008) have proposed a hybrid particle swarm optimisation with ACO (PSO/ACO2) algorithm for the discovery of classification rules that can directly cope with both continuous attributes. Experiments show that PSO/ACO2 is very competitive in terms of accuracy as compared to PART (Frank & Witten, 1998) and produces simpler rule sets.

However, PSO does not guarantee for local optimal solution. To solve this problem, Nalini & Balasubramnaie (2010) proposed the use of Tabu search (TS) in PSO to improve the search capability and integrate the pheromone concept of ACO. Besides, the proposed algorithm also runs two tasks simultaneously in parallel machines. Experiments have shown that the proposed algorithm is competitive against existing classification algorithms in terms of predictive accuracy and rule's simplicity. As an addition, Nalini & Balasubramnaie (2010) also claimed that since this algorithm runs in parallel, the execution time is reduced.

Shahzad & Baig (2011) proposed a hybrid classification algorithm called ACO-AC. This algorithm integrates classification with association rule mining technique to discover high quality rules. ACO is used to mine only the appropriate subset of class association rules, and not all possible rules. Shahzad & Baig (2011) compared ACO-AC against eight other classification algorithms on twenty six data sets and found that this algorithm achieves higher accuracy rates.

Sequential covering strategy used in the first implementation of ACO for classification, Ant-miner (Parpinelli et al., 2002a) has a drawback of not coping with the problem of rule interaction where the outcome of a rule affects the rules that can be discovered subsequently since the search space is modified due to the removal of examples covered by previous rules. Hence, Otero et al. (2012) proposed a new sequential covering strategy for ACO classification algorithms to lessen the problem of rule interaction. The proposed algorithm implicitly encoded the rules as pheromone values and guides the search using the quality of a candidate list of rules. Experiments using eighteen publicly available data sets show that the proposed algorithm gained a significantly higher predictive accuracy as compared to other rule induction classification algorithms.

Last but not least, Mangat (2012) proposed a hybrid of genetic algorithm (GA), PSO and ACO algorithm to mine rules in the medical domain that can handle all types of attributes. According to Mangat (2012), GA is used to feature selection, and provides inputs for ACO, PSO and/or Shuffled frog-leaping algorithm (SFLA). The major advantage of the usage GA in the discovery of prediction rule is that GA performs a global search based on a greedy approach. Mangat (2012) however, has not test the algorithm, but suggested that a data set like the heart disease data set available at the UCI machine learning repository may be used as input to check the effectiveness of the proposed algorithm in the medical domain.

2.8 Summary

Many researchers have applied ACO in many fields of optimization research, as well as machine learning for data mining. Most researchers in this field claimed that the performance of this kind of algorithm is at least at par when compared to the existing algorithms for classification such as C4.5, CN2 and PART.

The common problem with ACO is the stagnation (Dorigo & Stützle, 2004). Ant-miner (Parpinelli et al., 2002a, 2002b) also faced the same problem when the construction of high quality rule is not possible. Even though many researchers have made improvements to the Ant-miner, a perfect algorithm for data from all domains of problems is not possible. Some improvements may increase the predictive power, but not the simplicity of the constructed rule, and vice versa.

SA is an algorithm that is able to reduce the problem of local optimal. The usage of SA as local search algorithm in ACO could help to improve the ability of ACO to converge to an optimal solution. In other words, most of the researchers used SA to intensify the solutions discovered by each ant colony. For various problems, these hybrid solutions have shown better (or at least comparable) to other standard algorithms for each specific problem. The main idea for the hybrid ACO and SA algorithm is to use SA as a mechanism to improve solutions discovered by ACO since SA is an algorithm that has the ability to reduce the problem of local optimum. Hence, this thesis proposed two variants of hybrid ACO and SA algorithms to generate classification model using rule induction technique.

CHAPTER THREE
RESEARCH METHODOLOGY

This chapter presents the approach to develop techniques to classify Web documents and data sets using ACO algorithm. The development of the new Ant-miner used the same experimental research methodology as Parpinelli et al. (2002a, 2002b). Figure 3.1 depicts the phases of the research, which include the data sets development, the formulation of the algorithm for rule classification and the evaluation of the constructed rules.

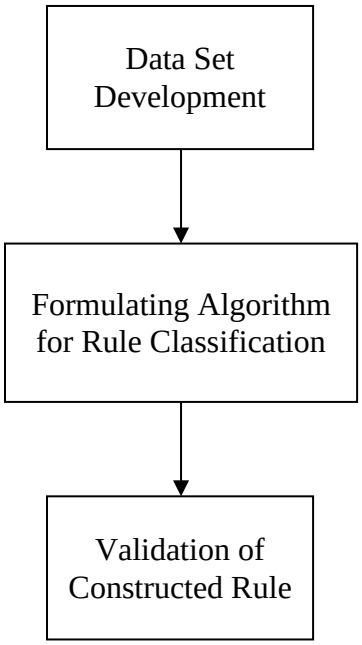

Figure 3.1: Research Phases

The following sections explain the phases of the research.

3.1 Data Set Development

The data set used in Chapter Four has been developed based on CMU World Wide Knowledge Base (Web->KB) project (http://www.cs.cmu.edu/~webkb/). Experiments in Chapter Four use this data set to determine the best attributes reduction methods, for Ant-miner (Parpinelli et al., 2002a, 2002b). Web->KB consist of manually pre-classified 8282 Web pages of computer science departments from four (4) universities namely Cornell (867 pages), Texas (827 pages), Washington (1205 pages) and Wisconsin (1263 pages) as well as 4120 miscellaneous pages collected from other universities, into seven (7) categories. The classes for these Web pages include student (1641 pages), faculty (1124 pages), staff (137 pages), department (182 pages), course (930 pages), project (504 pages) and other (3764 pages). The "other" category consists of Web pages that do not apply to the other six categories, such as publications list, vitae and research interests pages. A folder structure is used to organize each Web pages file by category, one folder for each category. Each of these seven (7) folders contains five (5) subfolders, one for each of the four (4) universities and one for the miscellaneous pages.

In order to build the data set, a set of words from the Web pages for each category was collected. The extracted words will only contain the twenty-six (26) English letters. Next, remove all the punctuations and spaces. As a result, a set of words that occurred in a Web page for a particular category presented in a simple word-based representation, called the bag-of-words. The bag-of-words presents a Web page (document) as a collection of words. Each word occurs in a document for at least once. Paragraph structure, punctuation and the meaning of the word were not taken

into consideration. For each word, information about the count of its occurrences was included. An example of the final data set was presented in Appendix C.

Chapter Five and Six experiments used a collection of seventeen (17) data sets from UCI repository (Asuncion & Newman, 2007). The data sets involved were Balance Scale, Breast Cancer (Ljubljana), Breast Cancer (Wisconsin), Credit-a, Credit-g, Diabetes, Heart (Cleveland), Heart (Statlog), Hepatitis, Ionosphere, Iris, Lymphography, Mushroom, Segment, Sonar, Tic-Tac-Toe and Vehicle. Table 3.1 summarized the main characteristic for these data sets.

Balance Scale is a data set that models psychological experimental results with 625 examples. Each example is classified as having the balance scale tip to the right (288 examples), tip to the left (288 examples), or at balanced (49 examples). There are four numeric attributes includes the left weight, the left distance, the right weight and the right distance.

The University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia by M. Zwitter and M. Soklic created breast cancer (Ljubljana) data set. This 286 examples data set includes 201 examples of one class (no-recurrence-events) and 85 examples of another class (recurrence-events). The examples are described by 9 categorical attributes.

Clinical Sciences Center, University of Wisconsin created the Breast cancer (Wisconsin) data set. The data set consists of 699 examples separated into two

classes: benign (458 examples) and malignant (241 examples). All the nine attributes in this data set are non-discrete.

Table 3.1: Data Sets Used in the Experiments

Data Sets	**Number of Examples**	**Number of Categorical Attributes**	**Number of Continuous Attributes**	**Number of Classes**
Balance Scale	625	0	4	3
Breast Cancer (Ljubljana)	282	9	0	2
Breast Cancer (Wisconsin)	683	0	9	2
Credit-a	690	9	6	2
Credit-g	1000	13	7	2
Diabetes	768	0	8	2
Heart (Cleveland)	303	8	5	5
Heart (Statlog)	270	0	13	2
Hepatitis	155	13	6	2
Ionosphere	351	0	34	2
Iris	150	0	4	3
Lymphography	148	15	3	4
Mushroom	8124	22	0	2
Segment	2310	0	19	7
Sonar	208	0	60	2
Tic-Tac-Toe	958	9	0	2
Vehicle	846	0	18	4

Credit-a data set is about credit card applications. There are 690 examples with 15 attributes, divided into two classes: positive (307 examples) and negative (383 examples). Nine of the attributes are discrete, while the rest are non-discrete.

Professor Dr. Hans Hofmann developed credit-g data set about German credit. This data set consists of 1000 examples divided into two classes: good (700 examples) and bad (300 examples). Thirteen out of twenty attributes in this data set are discrete.

Diabetes data set is a data set collected from Pima Indian heritage females' patients with the age range of at least 21 years old. It is used to investigate whether the patient shows signs of diabetes according to World Health Organization criteria. It consists of 768 numeric examples and divided into two classes: tested positive for diabetes (268 examples) and not tested positive for diabetes (500 examples).

Cleveland Clinic Foundation collected heart (Cleveland) data set. This data set consists of 303 examples divided into five classes: '<50' (164 examples), '>50_1' (55 examples), '>50_2' (36 examples), '>50_3' (35 examples) and '>50_4' (13 examples). There are thirteen attributes with eight discrete attributes.

Heart (Statlog) data set is a heart disease data set that contains 270 examples. All the thirteen attributes are non-discrete, and the examples are divided into two classes: absent (150 examples) and present (120 examples).

Hepatitis data set is a data set developed by Gail Gong from Carnegie-Mellon University. This data set consists of 155 examples divided into 2 classes: DIE (32 examples) and LIVE (123 examples). 6 attributes out of 19 attributes are continuous.

Ionosphere data set is a radar data, collected by a system in Goose Bay, Labrador. The data set consists of 351 examples, divided into 2 classes: good (225 examples) and bad (126 examples). All the 34 attributes in this data set are of continuous type.

Iris data set contains 3 classes of 50 instances each. Each class refers to a type of iris plant. All the four attributes for this data set are continuous. R.A. Fisher developed this data set in July 1988.

M. Zwitter and M. Soklic developed Lymphography data set, which was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. This data set consists of 148 examples divided into four classes: normal find (2 examples), metastases (81 examples), malign lymph (61 examples), and fibrosis (4 examples). 3 out of 18 attributes in this data set are continuous.

Mushroom data set includes descriptions of hypothetical samples corresponding to the species of gilled mushrooms in the agaricus and lepiota family. The data set consists of 8124 examples divided into 2 classes: edible (4208 examples) and poisonous (3916 examples). All the 22 attributes in this data set are continuous.

Vision Group, University of Massachusetts, developed segment data set with 2310 examples. All the 19 attributes are continuous. The examples in this data set were divided equally into seven classes: brickface, sky, foliage, cement, window, path, and grass.

The Sonar data set has 208 examples, used to discriminate between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock, developed by Gorman and Sejnowski. All the 60 attributes for this data set are continuous. This data set is divided into 2 classes: Rock (97 examples) and Mines (111 examples).

David W. Aha developed tic-tac-toe data set that encodes the complete set of possible board configurations at the end of tic-tac-toe games. There are 958 examples divided into 2 classes: negative (332 examples) and positive (626 examples). From the nine attributes for this data set, none are of continuous type.

The last data set used in this thesis is the vehicle data set, developed by Turing Institute, Glasgow, Scotland. Vehicle data set consists of 846 examples. This data set classifies a given silhouette as one of four types of vehicle, using a set of features extracted from the silhouette. All the 18 attributes are continuous and the examples are divided into 4 classes: opel (212 examples), saab (217 examples), bus (218 examples), and van (199 examples).

Many rule induction algorithms, including Ant-miner (Parpinelli et al., 2002a, 2002b) do not support continuous attributes. Discretization is a method to convert continuous attributes into categorical attributes. Typically, the data for the continuous attribute is partitioned into K ranges with equal length, or K % of the total data with equal frequencies (Clarke & Barton, 2000). For example, consider a data set that contains attribute that represent an age of persons. Attribute age is discretized into categorical values by transform it into ranges, such as "less than 15 years", "16 to 30 years", and so on.

This thesis used a discretization method called the Fayyad & Irani's MDL method (Fayyad & Irani, 1993) to discretize all the data set with non-discrete attributes. This discretization method uses information gain to recursively defines the best ranges. Dougherty et al. (1995) has given a detailed discussion on discretization methods for

continuous attributes. This thesis used WEKA software (M. A. Hall et al., 2009) to implement the discretization method on non-discrete attributes.

3.2 Algorithm Formulation

This study developed two hybrid algorithms based on the original Ant-miner proposed by Parpinelli et al. (2002a, 2002b) and SA. The main components of the Ant-miner are rule transaction, pheromone updates, heuristic function and rule pruning. The first proposed algorithm, presented in Chapter Five, used SA to optimize the rule discovered by an ant. The second proposed algorithm presented in Chapter Six, fixed the class before an ant discovers a rule. Besides, the optimization of the terms selection process while an ant constructing a rule, used the SA. The second proposed algorithm uses a simpler heuristic and fitness functions as compared to the first algorithm, which used the same heuristic and fitness functions as the Ant-miner (Parpinelli et al., 2002a, 2002b).

3.3 Rule Validation

This study used stratified *ten*-fold cross-validation method, depicted by Figure 3.2, to evaluate the performance of the classifiers. Cross-validation (Kohavi, 1995a) is a method for estimating how well a classifier performs on new data. Generally, this technique divides data set into almost equal size of ten (10) sets (or folds). Appendix B listed a Bash script to automate the process of creating the train/test sets. To evaluate the performance of a classifier, take the *i*-th set as a test set, and the rest of the sets as training sets. The classifier trained the training set to yield ordered

classification rules, $R = \{r_1, \ldots, r_{default}\}$. The default rule, $r_{default}$ is a rule that contains only the consequent part.

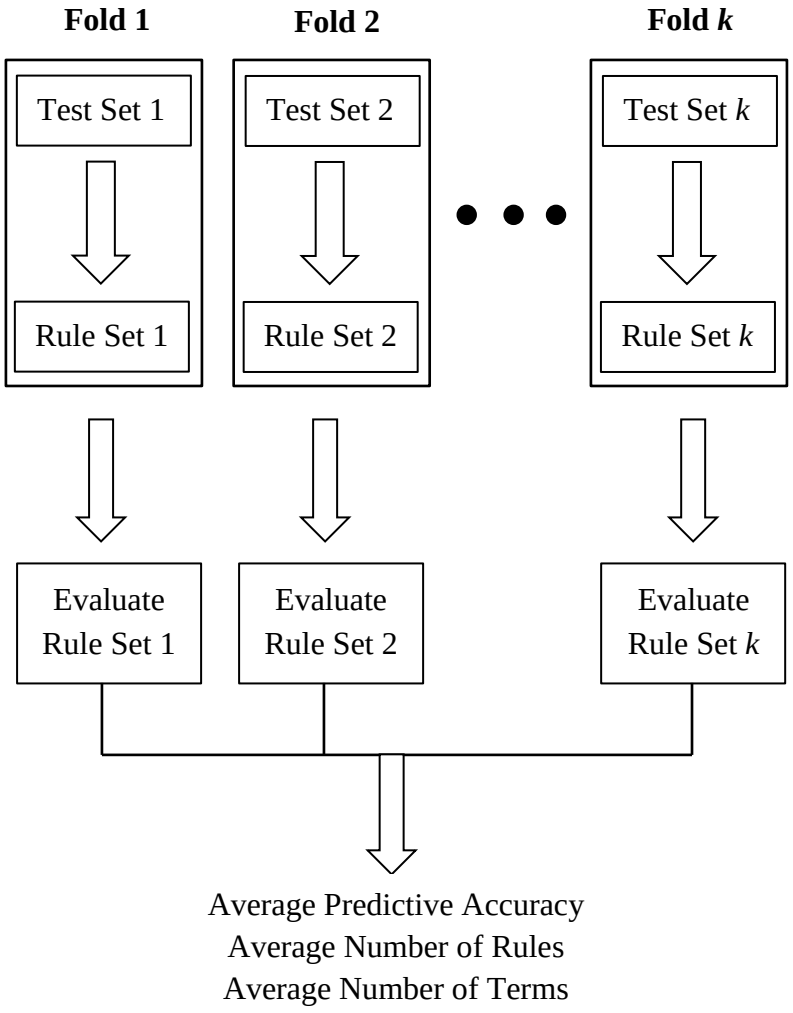

Figure 3.2: k-fold Cross Validation Procedure

Later, the classification rules were evaluated against the test set for predictive accuracy, acc_i. This process repeats *k* times by taking the next set as a test set, and the rest set as the training set. Each run uses different set as a test set. Finally, the

procedure calculates the average predictive accuracy, by dividing the total of all predictive accuracies by the total number of sets, as

$$\text{average predictive accuracy}, acc_{avg} = \frac{1}{10}\sum_{i=1}^{k} acc_i \tag{3.1}$$

where acc_i is the predictive accuracy for training the training set *i*.

The predictive accuracy for each test set *i* is the average of the number of correctly classified examples in the test set, by the classification rules. In other words, average predictive accuracy for each test set, is the count of the numbers of correctly classified rules, $n(c_{e(i)} = c_r)$, $i = 1, \ldots, m$, where m is the total number of examples in test set i, $c_{e(i)}$ is the consequent for example *i*, and c_r is the consequent for the rule with antecedent that covered the antecedent of example *i*. In order to measure the predictive accuracy, first, selects the first example, $e(1)$ in the test set. Second, select a rule one by one, in order, until reach a rule that covers the antecedent of $e(1)$, say r_j. If no rule cover $e(1)$, use the default rule, $r_{default}$. Third, compare the consequent of *j*-th rule's consequent, c_{r_j}, with the consequent of $e(1)$, $c_{e(1)}$. If the consequent of r_i and $e(1)$ is equal, increase the count of correctly classified examples, $n(c_{e(1)} = c_r)$, by one. The algorithm calculate the predictive accuracy, after all the examples in the test set have been tested against the classification rules, by calculating the average number of correctly classified rules, $n(c_{e(1)} = c_{r_j})$, as

$$acc_i = \frac{1}{m}\sum_{i=1}^{m} n\left(c_{e(i)} = c_{r_j}\right) \tag{3.2}$$

where acc_i is the predictive accuracy for test set i, and m is the total number of examples in test set i.

Ant-miner (Parpinelli et al., 2002a, 2002b) is the first ACO based algorithm for rule induction, and was credited by the founder of ACO, in his book (Dorigo & Stützle, 2004). Hence, this thesis compares the performance of the proposed algorithms with the Ant-miner (Parpinelli et al., 2002a, 2002b). As a control, this thesis also compares the performance of the proposed algorithms with PART (Frank & Witten, 1998) and some other rule induction algorithms discussed in Section 2.2. PART is known as an industrial standard classification algorithm (Witten et al., 2011), which is an improvement of C4.5 (Quinlan, 1993) included in WEKA software (M. A. Hall et al., 2009). PART builds a partial C4.5 tree in each iteration and makes the best leaf into the rule. In other words, PART extracts the path with the highest coverage to form the first rule. The instances covered by this rule are removed from the training examples and the process is then repeated to generate the second rule, and so on. Hence, a set of C4.5 trees are grown on increasingly smaller subsets of the training examples.

3.4 Summary

In formulating the hybrid Ant-miner, it is important to determine the best attribute selection methods for dimensionality reduction, a technique in optimizing the rules constructed by each ant, and a method for optimizing terms selection. These steps

are important so that the newly formulated algorithms can produce classification model that could help human to understand data better, especially for large data.

CHAPTER FOUR
ATTRIBUTE SELECTION METHODS FOR DIMENSIONALITY REDUCTION

According to a Web Server Survey by Netcraft (http://www.netcraft.com), there are about 677 million Web sites, with rate of growth of 5.1% in April 2012 as compared to March 2012, and thus contributing to the enormous size of information on the Web. Consequently, information searching from the Web is not an easy task. Web directories are Web sites that list other Web sites according to category and subcategory.

Categorizing Web documents into Web directories could facilitate information retrieval. Before committing a search, the user will select a specific category, such as “Arts”, “Business”, “Computers”, or “Sports”, resulting in more accurate and related information to the search engine results. DMOZ Open Directory Project (ODP) (http://www.dmoz.org) is an example of Web directory. A huge number of human editors mainly construct the ODP. However, as the Web is constantly changing and expanding, this manual approach will sooner became less effective. Hence, due to the enormous size of the Web, it would be good to have an automatic classier that will categorized Web pages, to help the development of a Web directory.

The development of Web directories can benefit from the help of an intelligent computer system. Classification is a data mining task of assigning objects to one of several predefined classes. It is an important task in many information management and retrieval tasks on the Web. Examples include helping Web spider to focus crawl,

improve the quality of Web search, and assisting in the development of Web directories.

Web text categorization problem, defines attributes as the words that occur in the Web pages (documents). The number of attributes could be tens or even hundreds of thousands, even for small size Web pages. According to the study conducted by Yang & Pederson, removing up to 98% of the attributes improved the text categorization performance (Yang & Pedersen, 1997).

This chapter presented a study on the performance of various attribute selection methods for reducing the dimension of data set. Next, Ant-miner classifies to see the performance in terms of predictive accuracy and the number of rules generated. This chapter compares the results of classification model to C4.5, as done by Parpinelli (Parpinelli et al., 2002a, 2002b).

4.1 Attribute selection method

Attribute selection is done by searching the space of attributes for a set of attributes that would best predict the class. Several search methods such as Best-First, Exhaustive Search, Genetic Search (Goldberg, 1989), Greedy Stepwise, Race Search (Moore & Lee, 1994) and Random Search (H. Liu & Setiono, 1996) are constantly used by researchers for attribute selection activities. The function for each search method is as listed in Table 4.1.

Table 4.1: Search Methods for Attribute Selection

Search Methods	Function
Best-First	Searches the space of attribute subsets by greedy hill climbing augmented with a backtracking facility.
Exhaustive Search	Performs an exhaustive search through the space of attribute subsets starting from the empty set of attributes.
Genetic Search	Search using simple genetic algorithm.
Greedy Stepwise	Searches the space of attribute subsets by greedy hill climbing augmented without a backtracking facility.
Race Search	Using race search methodology.
Random Search	Search randomly.

Table 4.2: Attribute Evaluation Methods for Attribute Selection

Evaluation Methods	Function
Correlation-based	Evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.
Classifier-based	Use classifier to evaluate attribute set.
Consistency subset	Measure consistency in class values for a chosen subset of attributes.

Evaluation method such as Correlation-based attribute subset selection (Hall, 1999), Classifier-based attribute subset selection and Consistency subset (H. Liu & Setiono, 1996) were used by many researchers to select the attributes in the process of evaluating the attributes subset found by search method. Table 4.2 lists all the functions for each evaluation method.

In this chapter, the performance of three attributes selection evaluation methods against Ant-miner and C4.5 has been undertaken. This chapter compares the performance of the attributes selection methods according to the predictive accuracy and the number of discovered rules, discussed in Section 4.2. Section 4.3 gives the summary.

4.2 Best Attribute Selection Method

This chapter uses the pre-classified Web pages of four Universities data sets (Craven et al., 1998), as mentioned in Section 1.1. The data set contains 8282 manually classified Web pages. The classes are student, faculty, staff, department, course and project. For this project, only two classes were chosen which are student and course, leaving only 2571 Web pages. Figure 4.1 depicts the whole process of the rules generation.

From those documents, a set of binary attributes (words) was extracted, leaving the html tags, numbers, stop words and punctuations (pre-process). Stop words are words that give very little contribution or none to the meaning of the text. Examples of stop words are “the”, “and”, “he”. For each attribute, a value of one (1) will be given if the attribute occurs in the document and zero (0) otherwise. The list of stop words used for this experiment is listed in Appendix A.

The words “car” and “cars” come from the same word root “car”. Instead of keeping both words “car” and “cars”, the number of words will be reduced using stemming algorithm if only the root word is used. Therefore, each word extracted was stemmed using Porter’s stemming algorithm (Porter, 1980).

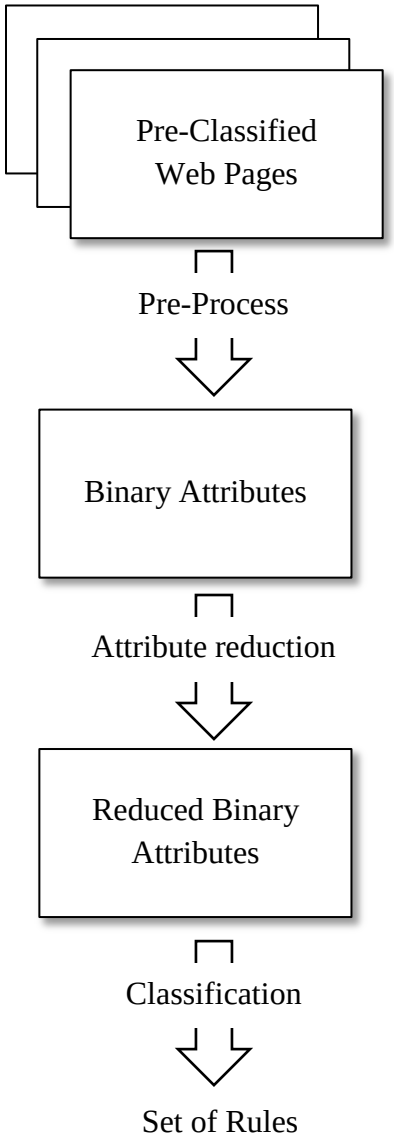

Figure 4.1: The Process of Generating Rules

According to Hull (1996), there is no difference between the stemmers in terms of average performance. However, Porter's stemming algorithm (Porter, 1980) is the most common algorithm for stemming English. Attributes that occur less than a hundred (100) in the whole set of documents were also removed, and for each category, only twenty (20) attributes were chosen.

This chapter compares three different attribute evaluation methods with seven search methods, to reduce the number of attributes. The evaluation methods are Correlation-based attribute subset selection (M. A. Hall, 1999), Classifier-based attribute subset selection and Consistency subset (H. Liu & Setiono, 1996). The search methods used include Best-First, Exhaustive Search, Genetic Search (Goldberg, 1989), Greedy Stepwise, Race Search (Moore & Lee, 1994) and Random Search (H. Liu & Setiono, 1996). The attribute selection was done using WEKA software (M. A. Hall et al., 2009) which generates a number of new sets of data. Table 4.3 shows the number of attribute selected for each attribute selection run.

For each set generated by the attribute selection, classification of documents was performed. The experiment compared these results with C4.5 (Quinlan, 1993). A *10*-fold cross validation procedure (Kohavi, 1995a) measures the accuracy of the discovered rules. The experiment consists of 10 folds (iterations), where each fold uses a different set of data as a test set.

The parameters for the Ant-miner are the same as were used by Parpinelli (Parpinelli et al., 2002a, 2002b), which are as follows:

- Number of ants: 3000
- Minimum number of examples per rule: 10
- Maximum number of uncovered examples: 10
- Number of identical for convergence: 10

Table 4.3: The Numbers of Attributes Generated by Various Attribute Selection Methods

Evaluation Methods	Search Method	Number of Attributes
Classifier-based	Best-First	8
	Exhaustive Search	8
	Genetic Search	8
	Greedy Stepwise	6
	Race Search	6
	Random Search	9
Correlation-based	Best-First	16
	Exhaustive Search	16
	Genetic Search	16
	Greedy Stepwise	16
	Random Search	17
Consistency subset	Best-First	20
	Genetic Search	19
	Greedy Stepwise	20

To evaluate the rules generated by each fold, at the end of each fold run, statistics such as predictive accuracy of the rule set and the number of rules in the rule set are calculated. Table 4.4 reported the average predictive accuracy in the test set over the ten (10) iterations of the cross validation procedure for each new data set created by

the attribute selection in terms of average predictive accuracy, while Table 4.5 list out the average number of rules generated by each classifier for different attribute selection methods. The number after the "±" sign is the standard deviation.

Table 4.4: Comparison Between C4.5 and Ant-miner for Average Predictive Accuracy

Evaluation Methods	**Search Method**	**Average Predictive Accuracy**	
		C4.5 (%)	**Ant-miner (%)**
Classifier-based	Best-First	92.29 ± 1.63	88.77 ± 1.66
	Exhaustive Search	92.29 ± 1.63	88.77 ± 1.66
	Genetic Search	92.29 ± 1.63	88.77 ± 1.66
	Greedy Stepwise	92.38 ± 1.63	89.94 ± 0.86
	Race Search	92.38 ± 1.63	89.94 ± 0.86
	Random Search	92.31 ± 1.62	89.63 ± 0.87
Correlation-based	Best-First	93.74 ± 1.58	91.30 ± 1.45
	Exhaustive Search	93.74 ± 1.58	91.30 ± 1.45
	Genetic Search	93.74 ± 1.58	91.30 ± 1.45
	Greedy Stepwise	93.74 ± 1.58	91.30 ± 1.45
	Random Search	93.92 ± 1.49	93.36 ± 0.67
Consistency subset	Best-First	94.03 ± 1.42	88.96 ± 1.38
	Genetic Search	93.89 ± 1.41	88.80 ± 1.67
	Greedy Stepwise	94.03 ± 1.42	91.02 ± 0.99

Table 4.4 and Table 4.5 show that Correlation-based evaluation with Random Search is the best attribute selection method for Ant-miner, with the highest predictive accuracy and lowest number of rules. As for the C4.5 case, it seems like C4.5 performs slightly better (with Consistency subset evaluation method) than Ant-miner (with Correlation-based attribute subset selection) in terms of predictive accuracy. However, the number of rules generated is more than double the number of rules

generated by Ant-miner. As for the number of attributes concerned, it shows that the lesser the number of attributes slightly decrease the predictive accuracy, but increases the knowledge comprehension (the number of rules). However, if the attribute selection reduces too many attributes, like in the Classifier-based evaluation method case, the predictive accuracy will slightly reduce.

Table 4.5: Comparison Between C4.5 and Ant-miner for Average Number of Rules

Evaluation Methods	**Search Method**	**Average Number of Rules**	
		C4.5	**Ant-miner**
Classifier-based	Best-First	8.27 ± 0.94	8.70 ± 0.21
	Exhaustive Search	8.27 ± 0.94	8.70 ± 0.21
	Genetic Search	8.27 ± 0.94	8.70 ± 0.21
	Greedy Stepwise	7.00 ± 0.00	7.70 ± 0.15
	Race Search	7.00 ± 0.00	7.70 ± 0.15
	Random Search	7.4 ± 0.85	9.10 ± 0.46
Correlation-based	Best-First	18.93 ± 2.61	7.40 ± 0.27
	Exhaustive Search	18.93 ± 2.61	7.40 ± 0.27
	Genetic Search	18.93 ± 2.61	7.40 ± 0.27
	Greedy Stepwise	18.93 ± 2.61	7.40 ± 0.27
	Random Search	19.83 ± 2.45	7.00 ± 0.15
Consistency subset	Best-First	20.20 ± 1.63	7.30 ± 0.33
	Genetic Search	20.94 ± 1.88	8.00 ± 0.26
	Greedy Stepwise	20.20 ± 1.63	7.80 ± 0.33

Eventually, Correlation-based evaluation with Random Search attribute selection, which contributes the best predictive accuracy as well as the number of rules for Ant-miner, does not provide the best number of attributes removed. The number of attributes selected for this attribute selection method is 100%, which is higher than

the one generated by Classifier-based attribute subset selection. Even though predictive accuracy depends on the attributes selected, the choice of classifiers also plays a critical role. Classifier will perform differently for different domain of data sets.

4.3 Performance of Ant-miner on Reduced Attributes Data Sets

Experiments conducted in the previous section have shown that the best attribute selection method to be used in Ant-miner is the combination of Correlation-based evaluation with Random Search as the search method. This section studies the performance of Ant-miner when the number of attributes of the data sets were reduced using the best combination of evaluation method and search method on several UCI data sets (Asuncion & Newman, 2007). Table 4.6 listed the number of attributes before and after reduction using Correlation-based evaluation with Random Search as the search method.

Table 4.7 shows the average predictive accuracy of Ant-miner (Parpinelli et al., 2002a, 2002b) tested on sixteen UCI data sets. It is found that the average predictive accuracy was improved on about 68 % of all tested data sets. For the simplicity of the discovered rules, according to Table 4.8 and Table 4.9, the average number of rules was reduced on 50 % of the sixteen data sets while the average number of terms was reduced on 68 % of the sixteen data sets.

The experiment performs in this section shows that not all attributes occur in data sets are important and can be removed. By removing unnecessary attributes, the performance of a classifier in terms of the prediction power increased. As an

addition, the removal of unnecessary attributes will also increase the simplicity of the discovered rules. The simplicity of the discovered rules is important to the human where complex rules are harder for human to understand.

Table 4.6: The Number of Attributes Before and After Reduction

Data Sets	**Before Reduction**	**After Reduction**
Balance Scale	4	3
Breast Cancer (Ljubjana)	9	6
Credit-a	15	7
Credit-g	20	4
Diabetes	8	5
Heart (Cleveland)	13	7
Heart (Statlog)	13	7
Hepatitis	19	8
Ionosphere	34	13
Iris	4	2
Lymphography	18	11
Mushroom	22	4
Segment	19	6
Sonar	60	19
Tic-tac-toe	9	5
Vehicle	18	9

Table 4.7: Comparison of The Average Predictive Accuracy for Models Constructed by Ant-miner on Original and Reduced UCI Data Sets

Data Sets	**Before Reduction**	**After Reduction**
Balance Scale	71.27 ± 1.54	75.17 ± 1.84
Breast Cancer (Ljubjana)	73.74 ± 2.69	76.5 ± 2.76
Credit-a	84.93 ± 1.21	86.09 ± 1.11
Credit-g	70.5 ± 2.01	73.7 ± 1.51
Diabetes	73.55 ± 1.85	74.71 ± 2

Heart (Cleveland)	76.78 ± 2.09	80.64 ± 2.47
Heart (Statlog)	77.04 ± 2.64	80 ± 2.6
Hepatitis	83.07 ± 2.63	76.31 ± 3.59
Ionosphere	89.25 ± 1.65	86.25 ± 2.04
Iris	95.33 ± 1.42	95.33 ± 1.42
Lymphography	76.92 ± 3.94	70.92 ± 2.82
Mushroom	97.45 ± 0.62	98.52 ± 0.17
Segment	83.94 ± 1	86.23 ± 1.27
Sonar	78.27 ± 3.36	78.88 ± 2.35
Tic-tac-toe	71.17 ± 1.74	70.79 ± 2.05
Vehicle	58.29 ± 1.39	58.98 ± 1.64

Table 4.8: Comparison of The Average Number of Rules for Models Constructed by Ant-miner on Original and Reduced UCI Data Sets

Data Sets	**Before Reduction**	**After Reduction**
Balance Scale	7.8 ± 0.2	6 ± 0
Breast Cancer (Ljubjana)	6.1 ± 0.18	6.3 ± 0.15
Credit-a	7.4 ± 0.22	8.1 ± 0.31
Credit-g	9.1 ± 0.18	9 ± 0.21
Diabetes	9.3 ± 0.21	9.3 ± 0.15
Heart (Cleveland)	6.4 ± 0.27	6.1 ± 0.18
Heart (Statlog)	5.9 ± 0.18	6.1 ± 0.18
Hepatitis	4.8 ± 0.2	5.1 ± 0.18
Ionosphere	6.2 ± 0.2	6.1 ± 0.23
Iris	4.2 ± 0.2	4 ± 0
Lymphography	6.3 ± 0.26	6 ± 0.26
Mushroom	7.7 ± 0.33	10 ± 0
Segment	17.7 ± 0.47	20.4 ± 0.96
Sonar	6 ± 0.15	6 ± 0.15
Tic-tac-toe	7.9 ± 0.55	9 ± 0.63
Vehicle	12 ± 0.33	10.9 ± 0.41

Table 4.9: Comparison of The Average Number of Terms for Models Constructed by Ant-miner on Original and Reduced UCI Data Sets

Data Sets	**Before Reduction**	**After Reduction**
Balance Scale	10.6 ± 0.4	7 ± 0
Breast Cancer (Ljubjana)	7.9 ± 0.38	8.3 ± 0.42
Credit-a	10.6 ± 0.45	9.8 ± 0.29
Credit-g	15.6 ± 0.4	9.1 ± 0.38
Diabetes	10.3 ± 0.54	9.1 ± 0.43
Heart (Cleveland)	10.1 ± 0.66	8.3 ± 0.3
Heart (Statlog)	8.8 ± 0.53	8.7 ± 0.37
Hepatitis	8.5 ± 0.48	7.9 ± 0.6
Ionosphere	9.1 ± 0.78	8.9 ± 0.41
Iris	3.2 ± 0.2	3 ± 0
Lymphography	10 ± 0.68	8.2 ± 0.68
Mushroom	8.1 ± 0.46	9 ± 0
Segment	23.7 ± 0.91	25.8 ± 1.72
Sonar	11.7 ± 0.37	12.6 ± 0.56
Tic-tac-toe	9.5 ± 1.41	9.8 ± 1.15
Vehicle	24.5 ± 1.17	19.2 ± 1.26

4.4 Summary

The best attribute selection method to be used in Ant-miner is the combination of Correlation-based evaluation with Random Search as the search method. However, this attribute selection method will not give the best performance in attributes reduction. Using classifier-based attribute subset selection will reduce more attributes, but sacrifice the performance of the classifier. This experiments in this chapter shows that Ant-miner performed better than C4.5 for Web texts categorization.

Since this thesis only focuses on two classes of Web data set, it is suggested to test the performance of attribute selection on higher dimension of Web data sets, with more classes. On the other hand, a study on reducing the size of attributes dimension could also be done in relation to the linguistic relationship to generalize words, as a manual preliminary step before performing the attribute selection method.

Tests on several reduced data sets from various fields from UCI data sets using Correlation-based evaluation with Random Search as the search method were also conducted in this thesis on Ant-miner (Parpinelli et al., 2002a, 2002b). It is found that using a reduced attributes data sets improve the performance of Ant-miner in terms of prediction power as well as the simplicity of the discovered rules.

CHAPTER FIVE
SIMULATED ANNEALING AS LOCAL SEARCH IN ANT COLONY OPTIMIZATION FOR RULE INDUCTION

This chapter proposed a hybrid of ACO and SA algorithm for rule induction. The hybrid algorithm is part of the sequential covering algorithm, which is the commonly used algorithm to extract classification rules directly from data. SA acted as local search algorithm while each ant discover rule. The usage of SA as a local search algorithm will minimize the problem of low quality discovered rule by an ant in a colony, where the rule discovered by an ant is not the best quality rule. In the experiments, seventeen (17) data sets which consist of discrete and continuous data from UCI repository (Asuncion & Newman, 2007) were used to evaluate the performance of the proposed algorithm. This chapter compares the experiments' results to the Ant-miner and some other rule induction algorithms discussed in Section 2.2. In addition, this chapter also tested the proposed algorithm with the reduced Web data set from Chapter Four, in order to test the credibility of the proposed algorithm on Web pages classification.

5.1 Simulated Annealing as Local Search

The proposed hybrid algorithm follows Sequential Covering algorithm, in order to discover rules from the data set. Figure 5.1 depicts the flowchart for the sequential covering algorithm. Generally, the algorithm starts with an empty discovered rules list. This algorithm will extract rule, in $IF < conditions > THEN < class >$ form, one at a time from the training data set using Learn-One-Rule function. The Learn-One-Rule function contains a set of ant colonies, which will discover a set of rules. The

algorithm selects one best rule from this set of rules, and then remove examples from the training data set that were covered (disregard the class) by this best rule. The best rule refers to the highest quality rule among all the ants from each ant colony. The algorithm appends this rule to the end of the list of discovered rule list, according to the discovery order, called the ordered discovered rules list.

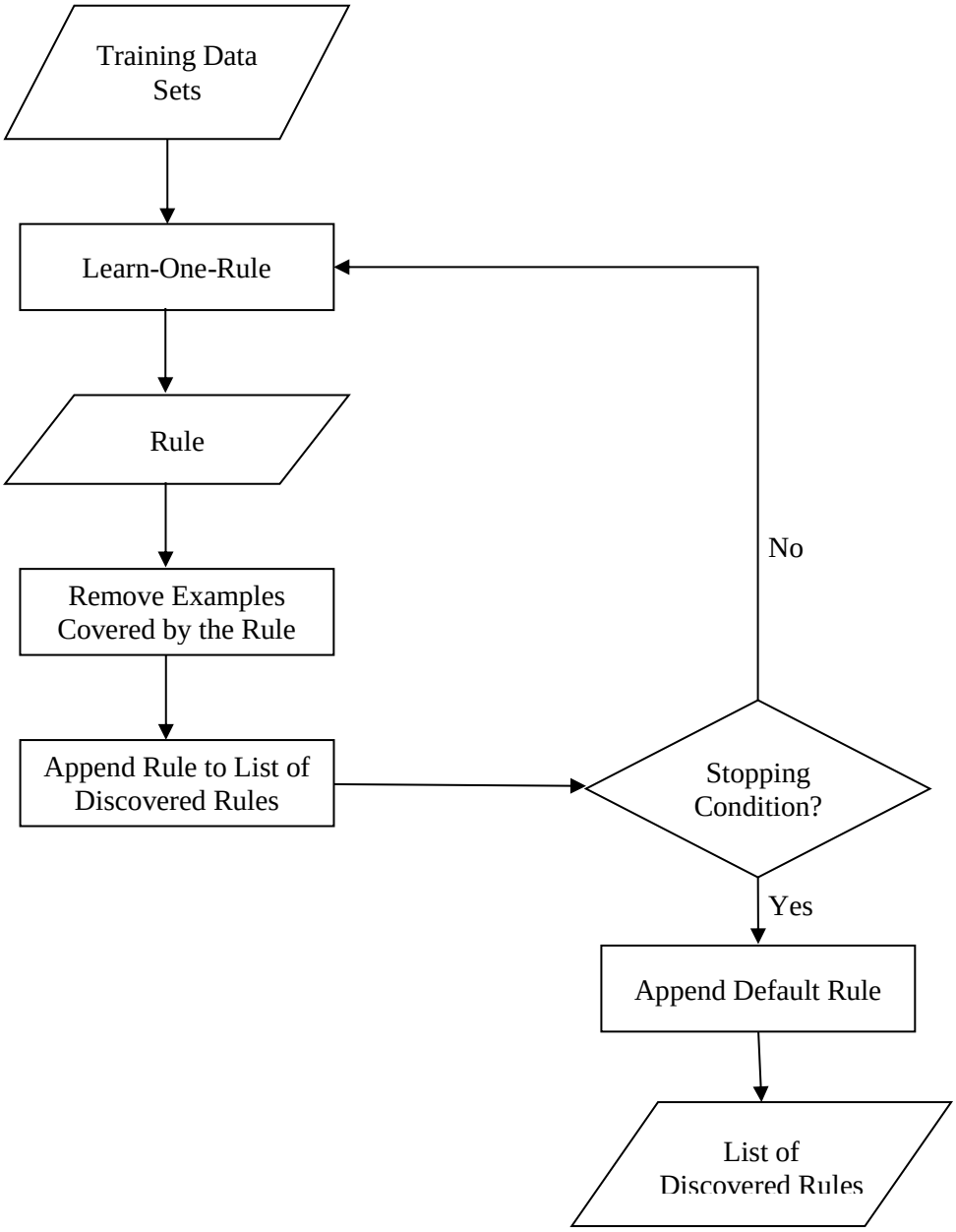

Figure 5.1: Sequential Covering Algorithm

After each ant colony has discovered the rules, the algorithm updates the pheromone level. Initially, the pheromone level for all terms were initialized using

$$\tau_{ij} = \frac{1}{\sum_{i=1}^{a} b_i} \tag{5.1}$$

where

- a is the total number of attributes, and
- b_i is the total number of values for attribute i.

After each rule discovery, by each ant colony, the algorithm increases the pheromone level for terms used in the discovered rule. The algorithm updates pheromone level at time t using

$$\tau_{ij}(t) = (1+Q)\,\tau_{ij}(t-1) \tag{5.2}$$

where

- τ_{ij} is the pheromone level for term with attribute i and value j, and
- Q is the quality of the current discovered rule, defined by Equation 5.4.

The algorithm evaporates pheromone level for all terms by dividing the pheromone level for each term with the total of pheromone level. This normalization procedure is done after the increased of pheromone level for all used terms.

The rules discovery processes repeat until the total number of data set is less than or equal to a predefined threshold value (the maximum number of uncovered data). Finally, the algorithm adds a default rule to the set of discovered rules. A default rule is a rule without the antecedent part, but contains a class with majority examples from the set of uncovered examples.

In order to improve the rule's quality discovered by each ant, this algorithm uses SA as a local search in Learn-One-Rule function. The algorithm applied the local search algorithm iteratively to each ant while each ant is discovering rule, in each ant colony using the Learn-One-Rule function. Figure 5.2 depicts the implementation of SA as a local search algorithm in ACO for rule induction.

Each ant in each colony uses the SA to discover one rule. SA depends on the temperature variable. The temperature starts very high with a predefined value, and gradually gets lower, by a factor of a predefined threshold value. Iteration in the SA will create one rule using a terms selection method, which depends on a probability. Iterations will stop when the temperature has reduced to a predefined lower limit temperature. Figure 5.3 depicts these processes.

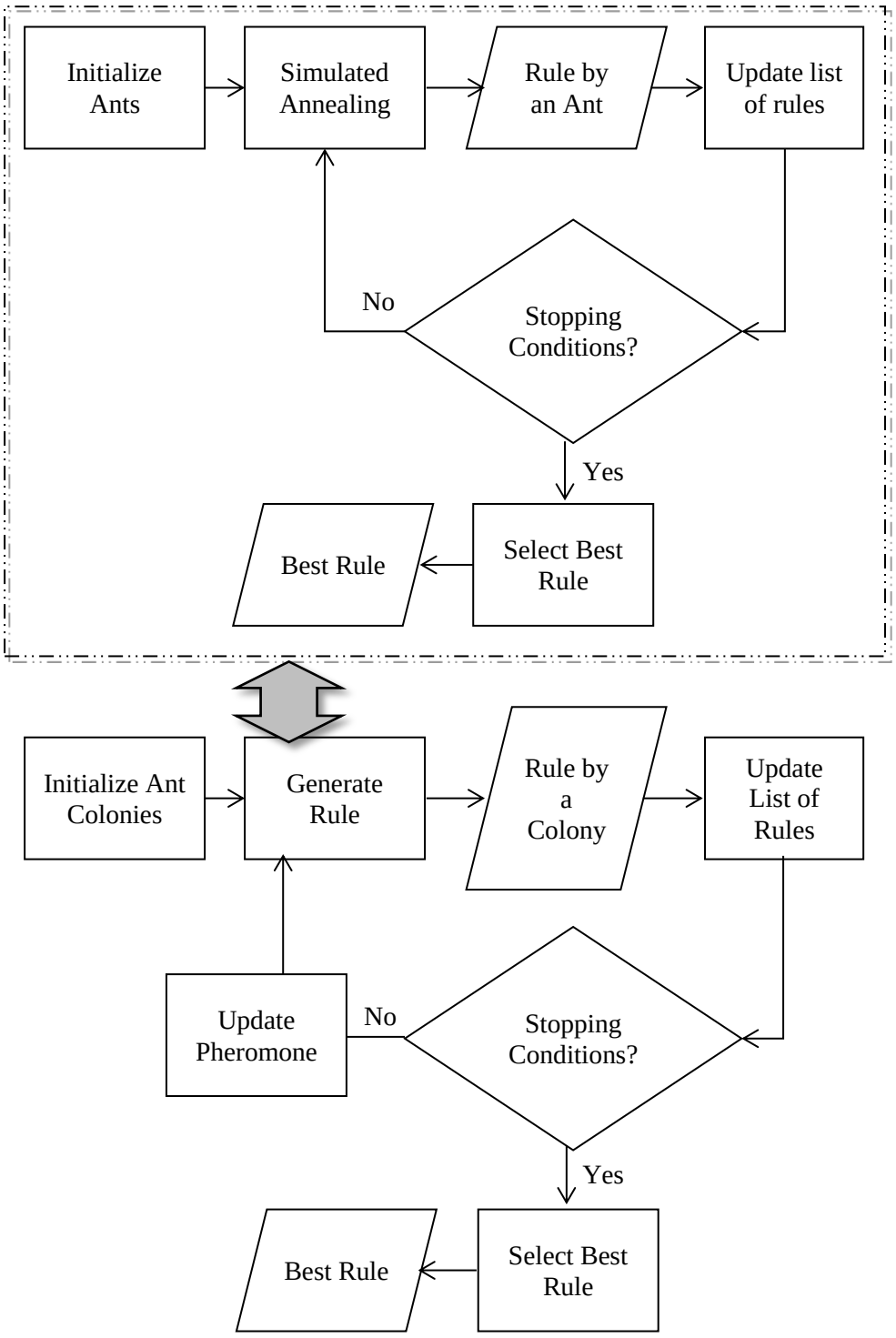

Figure 5.2: SA as Local Search in ACO Flow Chart

The algorithm evaluates the rule based on its quality, and selects the new rule as the best rule for iteration, if the new rule's quality is better than the previous quality. However, rule with lower quality also has a chance to be selected using the probability

$$p = \exp\left(\frac{-q_1 - q_2}{T}\right) \tag{5.3}$$

where

- q_1 and q_2 are the quality for the previous and current rules respectively, and
- T is the temperature for the current iteration.

The temperature starts very high. Therefore, p will almost be one, and all rules have almost the same probability to be chosen. As the temperature decreases, the difference between the previous and current quality become more significant. Hence, rule with slightly lower quality will be preferred over much lower ones.

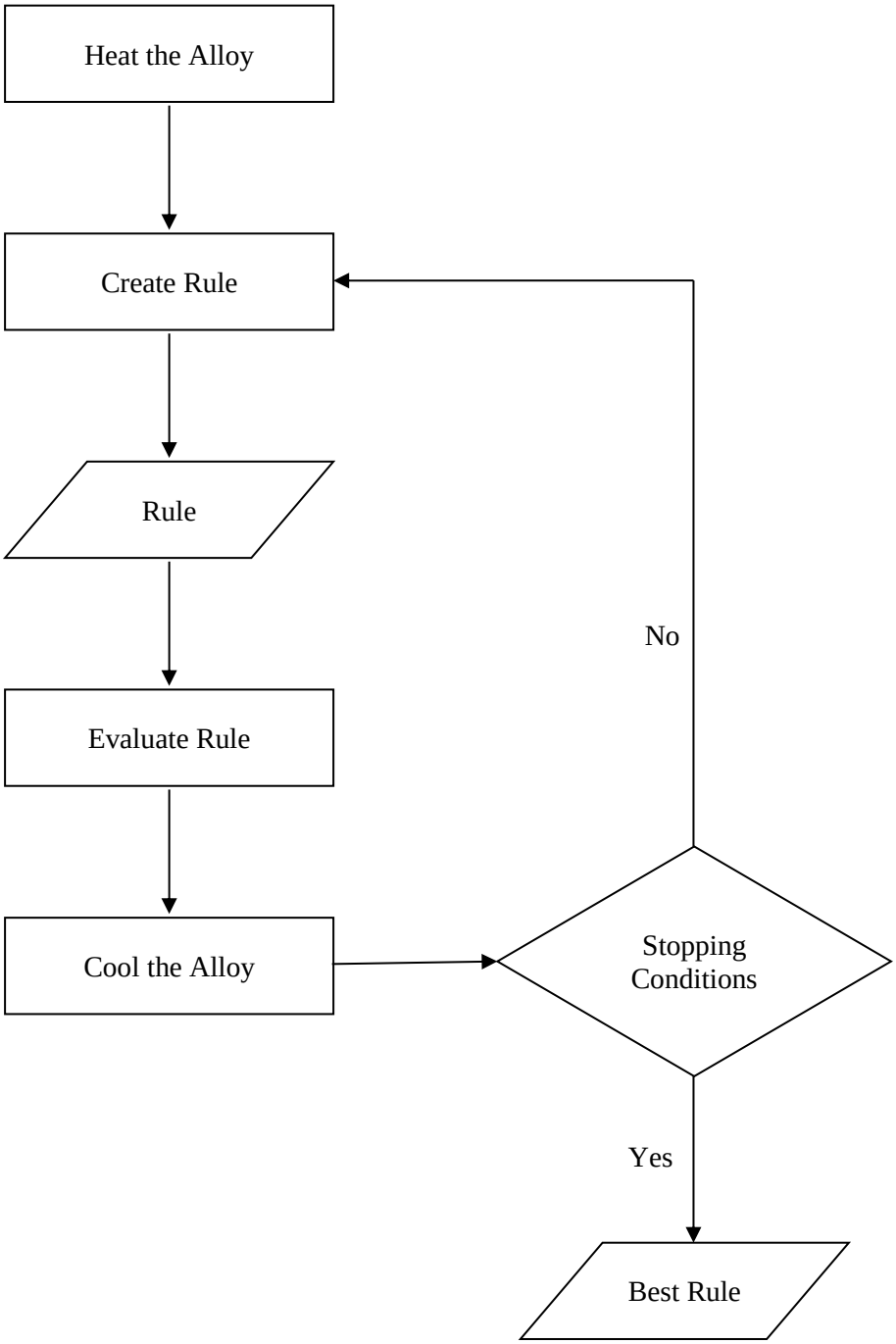

Figure 5.3: SA Flow Chart to Construct Best Rule for an Ant

After each ant discovers a rule, the algorithm performs pruning procedure. The pruning procedure iteratively removes one term at a time while rule's quality is improved. The rule quality is calculated using fitness function

$$Q = \frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP} \tag{5.4}$$

where

- TP is the number of examples covered by the rule and having the same class from the class predicted by the rule,
- FP is the number of examples covered by the rule and having a different class from the class predicted by the rule,
- FN is the number of examples that are not covered by the rule, but have the same class with the class predicted by the rule, and
- TN is the number of examples that are not covered by the rule, and have a different class with the class predicted by the rule.

In the terms selection procedure (depicted by Figure 5.4), terms will be added to the partial rule until there is no more possible terms left. A new selected term's attribute is an unused previously attribute. In other words, a rule cannot have an attribute with more than one value. It is a one to one mapping of an attribute and its value. After terms addition process has finished, the algorithm adds a consequent to the partial

rule. The consequent is the class with the majority examples, from all examples covered by the rule. The term selection method uses the following probability:

$$P_{ij} = \frac{\tau_{ij} \times \eta_{ij}}{\sum_{i=1}^{a} \sum_{j=1}^{f_i} \tau_{ij} \times \eta_{ij}} \tag{5.5}$$

where

- η_{ij} is the heuristic value for term with attribute i and value j,
- τ_{ij} is the pheromone level for term with attribute i and value j, and
- f_i is the number of values for attribute i.

The probability depends on the heuristic value, $\eta_{ij}(t)$, and current pheromone level, $\tau_{ij}(t)$ for each term, $term_{ij}$. By time, after each ant colony has selected one best rule, the pheromone level for the higher quality terms will increase, and hence increases the chances of selecting those terms. The heuristic value for terms remains the same.

The heuristic function uses the entropy for preferring a term to the others, as

$$\eta_{ij} = \frac{\log_2 k - i\text{nfo}_{ij}}{\sum_{i=1}^{a} \sum_{j=1}^{b_i} \log_2 k - \text{info}_{ij}} \tag{5.6}$$

where

- info_{ij} is the entropy value for term with attribute i and value j,
- k is the total number of classes,
- T_{ij} is the total number of examples for attribute i with value f_i,
- a is the total number of attributes, and
- b_i is the total number of values for attribute i.

The entropy is calculated using Equation 5.7.

$$\text{info}_{ij} = -\sum_{c=1}^{k} P(c \mid A_i = V_{ij}) \bullet \log_2 P(c \mid A_i = V_{ij}) \tag{5.7}$$

where

- c is the class value,
- k is the total number of class values, and
- $P(c \mid A_i = V_{ij})$ is the probability of observing class c conditional on observing attribute i having value V_{ij}, $A_i = V_{ij}$.

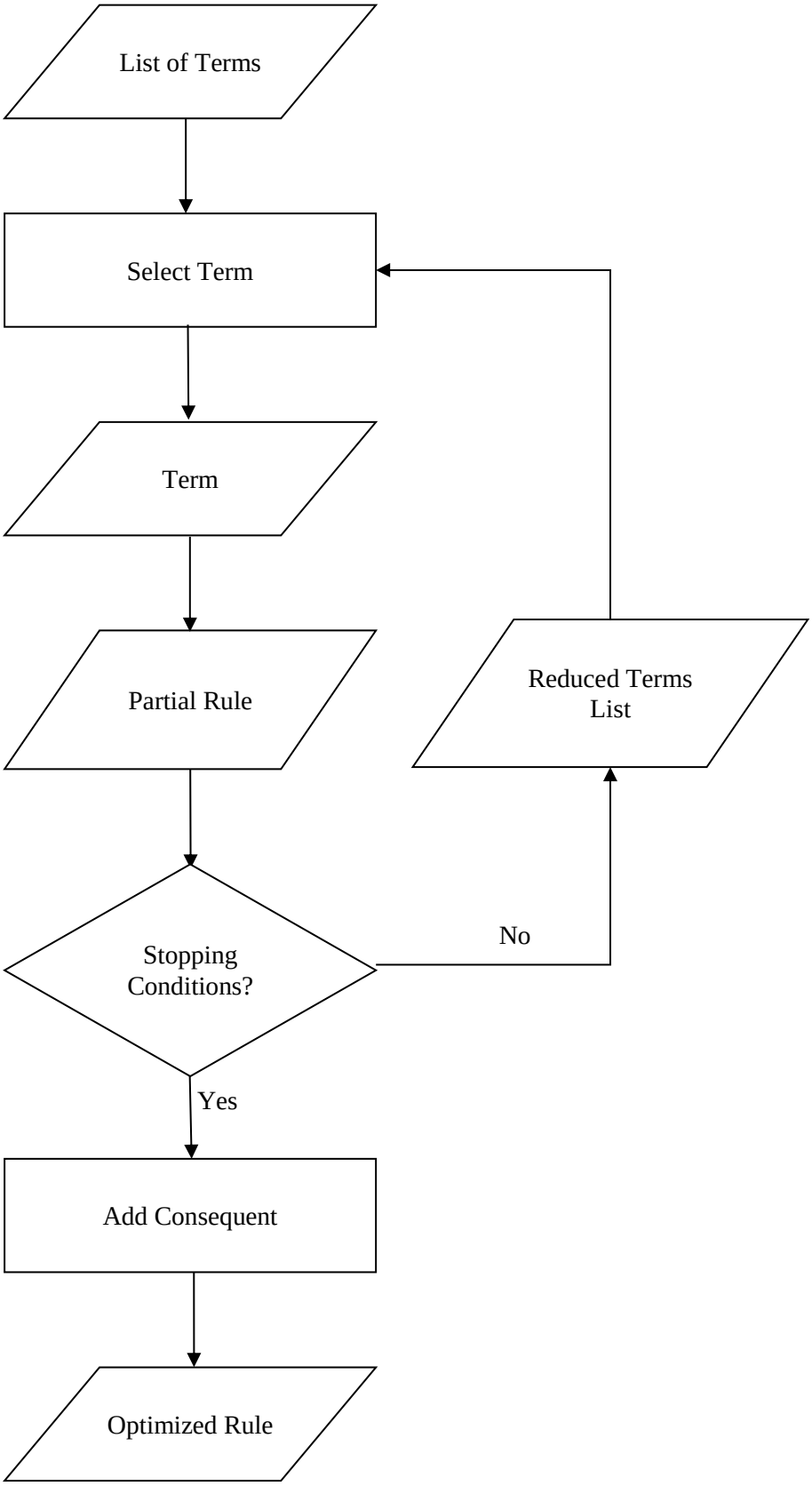

Figure 5.4: Terms Selection Procedure Flow Chart

5.2 Experiment and Results

The chapter tested the proposed algorithm on 17 data sets from UCI repository (Asuncion & Newman, 2007), as well as the reduced Web data set from Chapter Four, using a *ten*-fold cross-validation technique (Kohavi, 1995a). Each data set is randomly shuffled and split into ten approximately equally sized subsets using WEKA software (M. A. Hall et al., 2009). First, the experiment takes one subset as the testing set, while the rest subsets become the training set. After the proposed algorithm trained the training set, the proposed algorithm evaluated the discovered rules against the test subset. The experiment repeats this processes ten times, producing ten individual sets of performance statistics, such as predictive accuracy, number of rules and number of terms in the rules. Finally, the experiment averages these performance statistics and calculates the standard deviations for each performance statistics.

The proposed hybrid algorithm, cannot cope with non-discrete attributes, but only nominal attributes. Hence, the experiment discretizes non-discrete attributes in the first place, using the discretization method proposed by Fayyad & Irani (1993), as discussed in Chapter Three. All the experiments use 100 ant colonies with 5 ants each.

5.2.1 Classification of 17 Data Sets from UCI Repository

The performance of the proposed algorithm is evaluated by comparing it with Ant-miner (Parpinelli et al., 2002a, 2002b). Table 5.1-5.3 summarized the performance statistics for Ant-miner (Parpinelli et al., 2002a, 2002b), and the proposed algorithm.

Figure 5.5–5.7 visualized the performance statistics laid out by Table 5.1–5.3, respectively.

Table 5.1 and Figure 5.5 compare the predictive accuracy between Ant-miner (Parpinelli et al., 2002a, 2002b) and the proposed algorithm. Compared to the Ant-miner, the proposed algorithm achieves almost similar predictive accuracy. It is found that the proposed algorithm only performed better than the Ant-miner on only 35% (6 out of 17) of the data sets. For the data sets where the Ant-miner wins, the differences are very small. However, when it comes to simplicity, the proposed algorithm performed a significant different where the proposed algorithm construct fewer number of rules for 82% (15 out of 17) of the data sets, as shown in Table 5.2 and Figure 5.6. Moreover, the proposed algorithm produced rule with fewer number of terms on all data sets, compared to the Ant-miner, as shown in Table 5.3 and Figure 5.7.

Table 5.1: Average Predictive Accuracy (%) of Ant-miner and Proposed Algorithm 1

Data Sets	**Ant-miner**	**Proposed Algorithm 1**
Balance Scale	71.27 ± 1.54	63.68 ± 6.19
Breast Cancer (Ljubljana)	73.74 ± 2.69	74.11 ± 3.95
Breast Cancer (Wisconsin)	94.99 ± 0.49	93.99 ± 3.71
Credit-a	84.93 ± 1.21	84.64 ± 4.31
Credit-g	70.50 ± 2.01	72.90 ± 4.13
Diabetes	73.55 ± 1.85	74.62 ± 4.70
Heart (Cleveland)	76.78 ± 2.09	74.83 ± 7.95
Heart (Statlog)	77.04 ± 2.64	75.56 ± 7.07
Hepatitis	83.07 ± 2.63	82.00 ± 4.62
Ionosphere	89.25 ± 1.65	85.48 ± 5.62
Iris	95.33 ± 1.42	96.00 ± 4.42
Lymphography	76.92 ± 3.94	78.95 ± 7.41
Mushroom	97.45 ± 0.62	91.14 ± 5.93
Segment	83.94 ± 1.00	86.15 ± 2.44
Sonar	78.27 ± 3.36	73.14 ± 8.96
Tic-tac-toe	71.17 ± 1.74	71.09 ± 3.86
Vehicle	58.29 ± 1.39	57.57 ± 6.04

Table 5.2: Average Number of Rules of Ant-miner and Proposed Algorithm 1

Data Sets	**Ant-miner**	**Proposed Algorithm 1**
Balance Scale	7.80 ± 0.20	5.00 ± 0.00
Breast Cancer (Ljubljana)	6.10 ± 0.18	5.20 ± 0.60
Breast Cancer (Wisconsin)	7.40 ± 0.22	7.40 ± 0.49
Credit-a	7.40 ± 0.22	5.80 ± 1.08
Credit-g	9.10 ± 0.18	8.10 ± 0.83
Diabetes	9.30 ± 0.21	9.10 ± 0.30
Heart (Cleveland)	6.40 ± 0.27	6.20 ± 0.98
Heart (Statlog)	5.90 ± 0.18	5.30 ± 0.46
Hepatitis	4.80 ± 0.20	4.50 ± 0.50
Ionosphere	6.20 ± 0.20	6.40 ± 1.11
Iris	4.20 ± 0.20	4.00 ± 0.00
Lymphography	6.30 ± 0.26	5.40 ± 0.49
Mushroom	7.70 ± 0.33	3.70 ± 1.10
Segment	17.70 ± 0.47	19.50 ± 1.96
Sonar	6.00 ± 0.15	5.20 ± 0.75
Tic-tac-toe	7.90 ± 0.55	6.20 ± 2.44
Vehicle	12.00 ± 0.33	11.90 ± 0.83

Table 5.3: Average Number of Terms of Ant-miner and Proposed Algorithm 1

Data Sets	**Ant-miner**	**Proposed Algorithm I**
Balance Scale	10.60 ± 0.40	4.00 ± 0.00
Breast Cancer (Ljubljana)	7.90 ± 0.38	4.20 ± 0.60
Breast Cancer (Wisconsin)	7.70 ± 0.26	6.40 ± 0.49
Credit-a	10.60 ± 0.45	4.80 ± 1.08
Credit-g	15.60 ± 0.40	7.10 ± 0.83
Diabetes	10.30 ± 0.54	8.10 ± 0.30
Heart (Cleveland)	10.10 ± 0.66	5.20 ± 0.98
Heart (Statlog)	8.80 ± 0.53	4.30 ± 0.46
Hepatitis	8.50 ± 0.48	3.50 ± 0.50
Ionosphere	9.10 ± 0.78	5.40 ± 1.11
Iris	3.20 ± 0.20	3.00 ± 0.00
Lymphography	10.00 ± 0.68	4.40 ± 0.49
Mushroom	8.10 ± 0.46	2.70 ± 1.10
Segment	23.70 ± 0.91	18.50 ± 1.96
Sonar	11.70 ± 0.37	4.20 ± 0.75
Tic-tac-toe	9.50 ± 1.41	5.20 ± 2.44
Vehicle	24.50 ± 1.17	10.90 ± 0.83

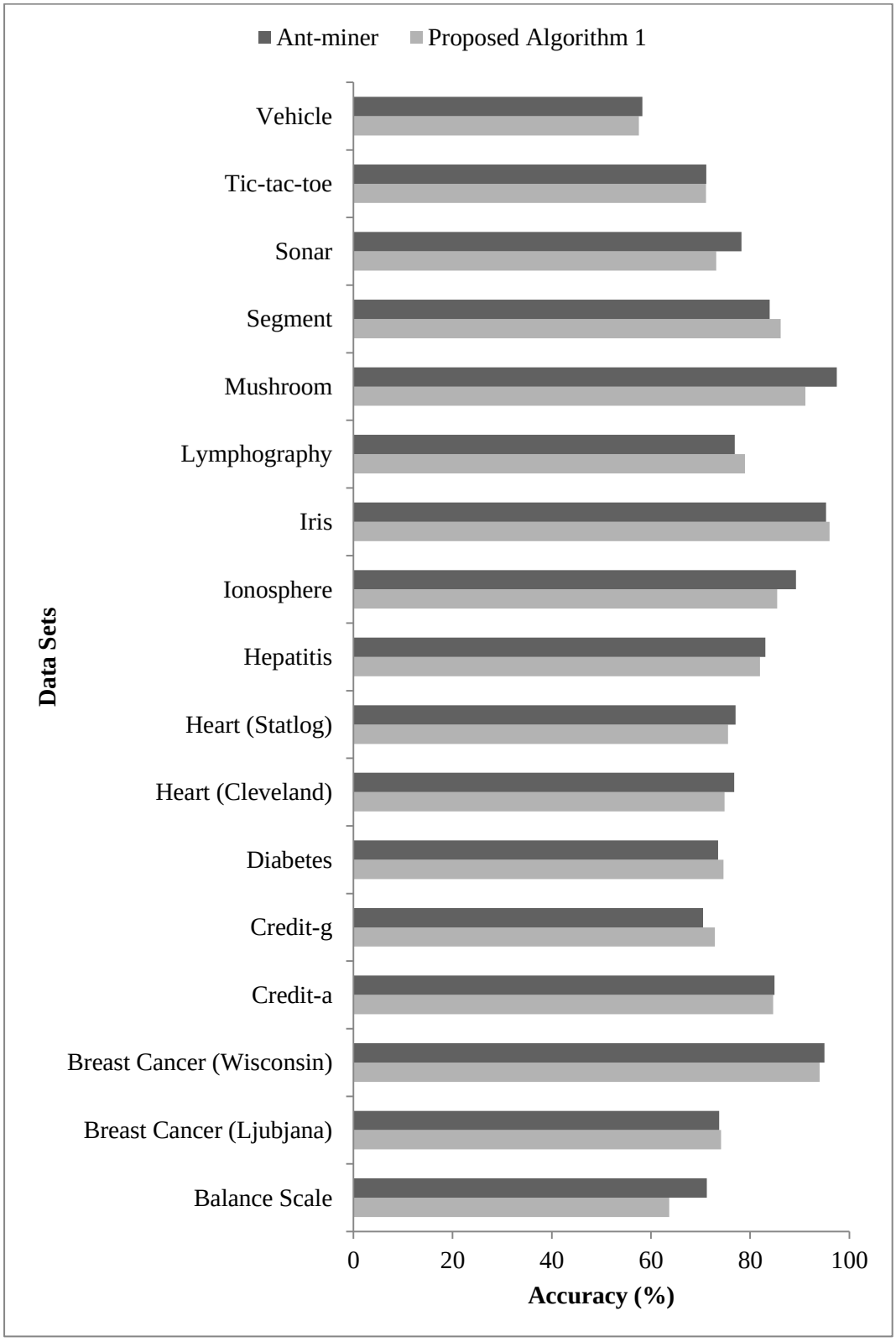

Figure 5.5: Comparison of Average Predictive Accuracy Between Ant-miner and Proposed Algorithm 1

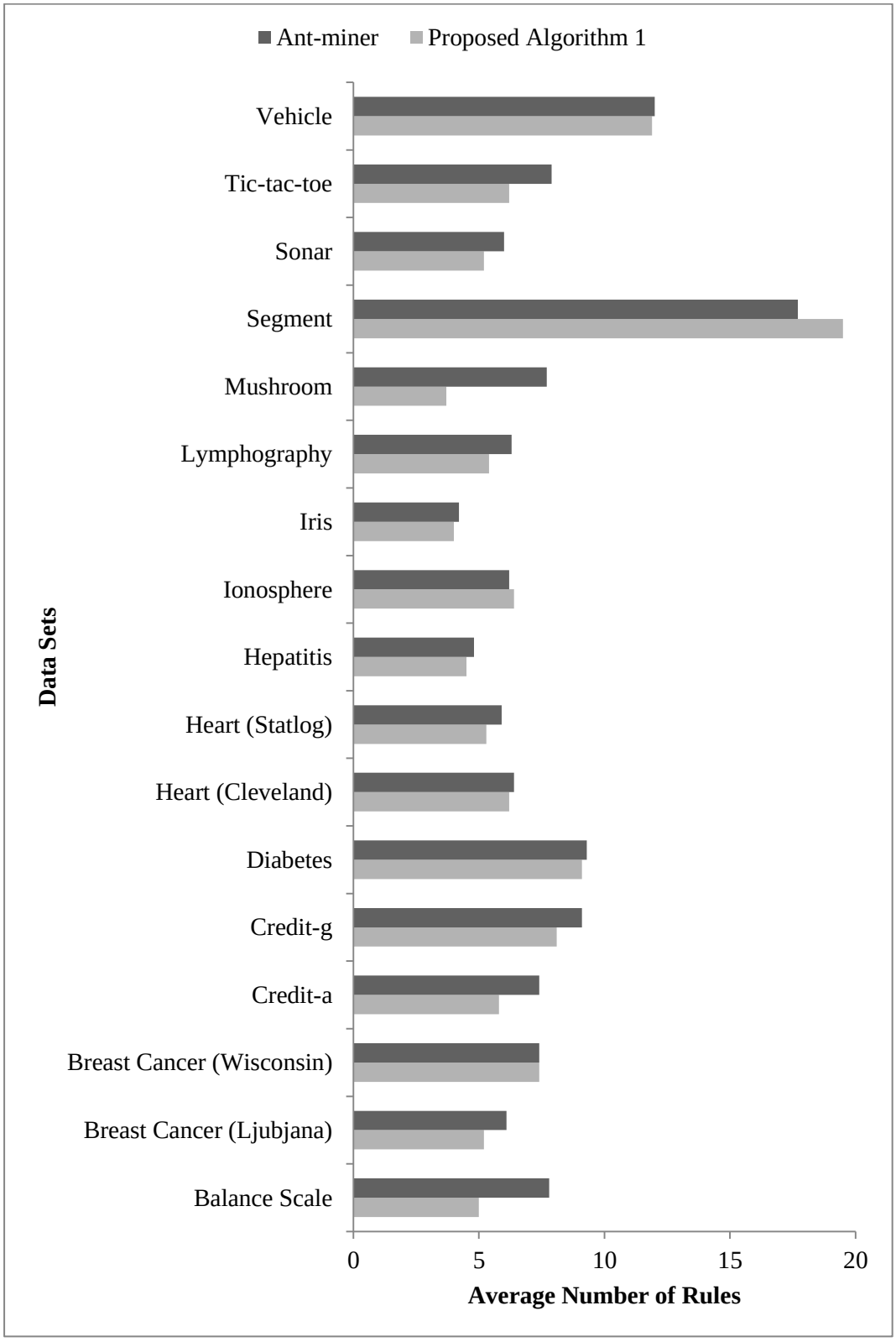

Figure 5.6: Comparison of Average Number of Rules Between Ant-miner and Proposed Algorithm 1

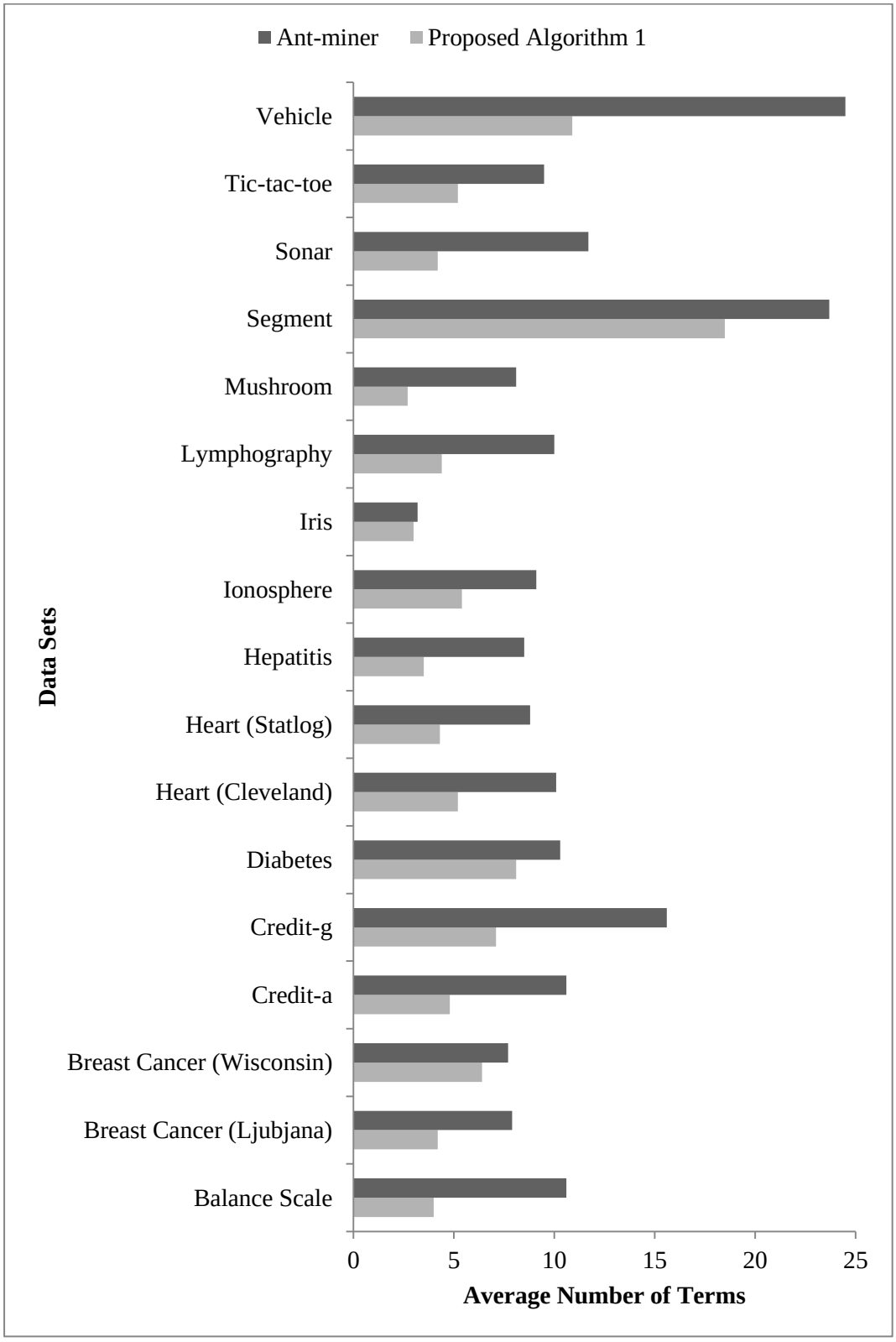

Figure 5.7: Comparison of Average Number of Terms Between Ant-miner and Proposed Algorithm 1

Experiments in Chapter 4 have shown that the removal of the unnecessary attributes will increased the predictive power of the classification model constructed by Ant-miner (Parpinelli et al., 2002a, 2002b) as well as improving the simplicity of the classification rules. This section tested the proposed algorithm on the seventeen UCI reduced data sets using the methods from Chapter 4: the combination of Correlation-based evaluation with Random Search as the search method.

According to the average accuracy laid out in Table 5.4 and depicted by Figure 5.8, the proposed algorithm wins on 6 out of 17 data sets (35%), the same as without attributes reduction method applied. However, the mean for all average accuracy is higher (79.5%) compared to when using the proposed algorithm to construct the classification model without reducing the attributes (78.58%).

Moreover, the simplicity (the less number of rules and the less number of terms per rule, the better the simplicity of the rules) of the constructed rules also improved when reducing the number of attributes as shown in Table 5.5 and Figure 5.9, where the proposed algorithm wins on 15 out of 17 (88.2%) data sets with a mean of 6.73 as compared to 14 out of 17 (82.4%) without reducing the attributes for the average number of rules with a mean of 6.99.

The same goes for the number of terms per rule. The mean for the average number of terms per rule, when the method of attributes reduction is applied on the data sets initially, was reduced by 4.3. The results were shown in Table 5.6 and Figure 5.10.

Experiments in this section have shown that even though the number of rules was reduced (simpler rules) and the number of terms for each rule was reduced, the predictive power (accuracy) of the constructed classification model still improved, if we reduced the unnecessary attributes using the proposed method in Section 4 from the data sets, before using the proposed algorithm in this section to constructed the classification model.

Table 5.4: Average Predictive Accuracy (%) of Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 1**
Balance Scale	75.17 ± 1.84	65.29 ± 5.32
Breast Cancer (Ljubljana)	76.50 ± 2.76	74.46 ± 3.94
Breast Cancer (Wisconsin)	94.71 ± 0.83	94.57 ± 3.60
Credit-a	86.09 ± 1.11	86.09 ± 4.99
Credit-g	73.70 ± 1.51	71.80 ± 4.75
Diabetes	74.71 ± 2.00	75.53 ± 4.47
Heart (Cleveland)	80.64 ± 2.47	75.17 ± 8.48
Heart (Statlog)	80.00 ± 2.60	76.30 ± 7.44
Hepatitis	76.31 ± 3.59	81.96 ± 7.41
Ionosphere	86.25 ± 2.04	88.04 ± 3.77
Iris	95.33 ± 1.42	94.67 ± 4.00
Lymphography	70.92 ± 2.82	80.95 ± 8.27
Mushroom	98.52 ± 0.17	98.52 ± 0.46
Segment	86.23 ± 1.27	85.80 ± 2.95
Sonar	78.88 ± 2.35	73.07 ± 7.45
Tic-tac-toe	70.79 ± 2.05	73.07 ± 1.87
Vehicle	58.98 ± 1.64	55.80 ± 3.07

Table 5.5: Average Number of Rules of Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 1**
Balance Scale	6.00 ± 0.00	4.00 ± 0.00
Breast Cancer (Ljubljana)	6.30 ± 0.15	4.90 ± 0.30
Breast Cancer (Wisconsin)	7.40 ± 0.22	7.50 ± 0.50
Credit-a	8.10 ± 0.31	6.90 ± 1.04
Credit-g	9.00 ± 0.21	8.90 ± 0.94
Diabetes	9.30 ± 0.15	8.10 ± 0.30
Heart (Cleveland)	6.10 ± 0.18	5.80 ± 0.60
Heart (Statlog)	6.10 ± 0.18	5.20 ± 0.40
Hepatitis	5.10 ± 0.18	4.10 ± 0.70
Ionosphere	6.10 ± 0.23	6.70 ± 0.78
Iris	4.00 ± 0.00	3.40 ± 0.49
Lymphography	6.00 ± 0.26	5.10 ± 0.30
Mushroom	10.00 ± 0.00	5.00 ± 0.00
Segment	20.40 ± 0.96	19.40 ± 2.29
Sonar	6.00 ± 0.15	5.10 ± 0.70
Tic-tac-toe	9.00 ± 0.63	5.20 ± 2.18
Vehicle	10.90 ± 0.41	9.10 ± 1.45

Table 5.6: Average Number of Terms of Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 1**
Balance Scale	7.00 ± 0.00	3.00 ± 0.00
Breast Cancer (Ljubljana)	8.30 ± 0.42	3.90 ± 0.30
Breast Cancer (Wisconsin)	8.20 ± 0.36	6.50 ± 0.50
Credit-a	9.80 ± 0.29	5.90 ± 1.04
Credit-g	9.10 ± 0.38	7.90 ± 0.94
Diabetes	9.10 ± 0.43	7.10 ± 0.30
Heart (Cleveland)	8.30 ± 0.30	4.80 ± 0.60
Heart (Statlog)	8.70 ± 0.37	4.20 ± 0.40
Hepatitis	7.90 ± 0.60	3.10 ± 0.70
Ionosphere	8.90 ± 0.41	5.70 ± 0.78
Iris	3.00 ± 0.00	2.40 ± 0.49
Lymphography	8.20 ± 0.68	4.10 ± 0.30
Mushroom	9.00 ± 0.00	4.00 ± 0.00
Segment	25.80 ± 1.72	18.40 ± 2.29
Sonar	12.60 ± 0.56	4.10 ± 0.70
Tic-tac-toe	9.80 ± 1.15	4.20 ± 2.18
Vehicle	19.20 ± 1.26	8.10 ± 1.45

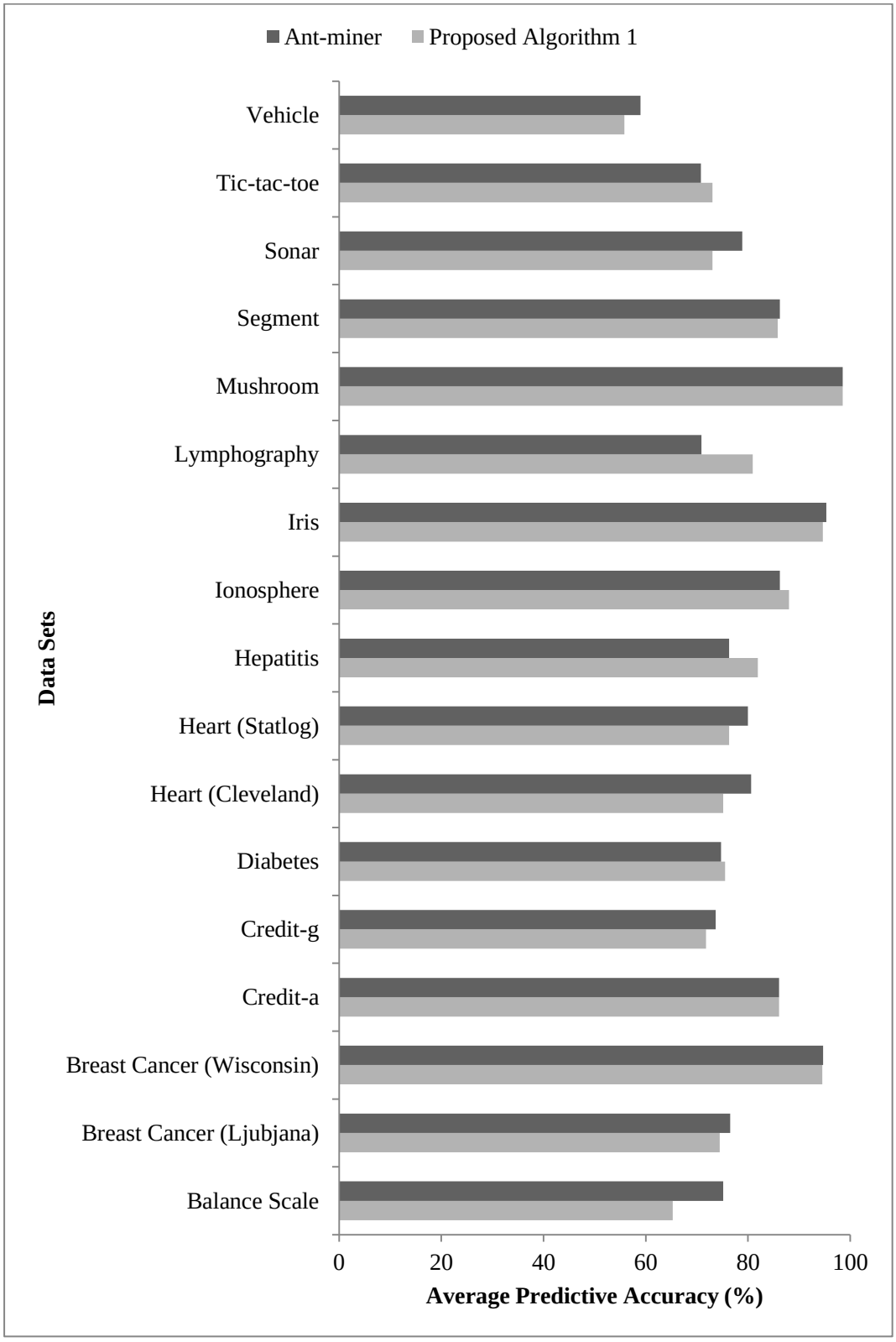

Figure 5.8: Comparison of Average Predictive Accuracy Between Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

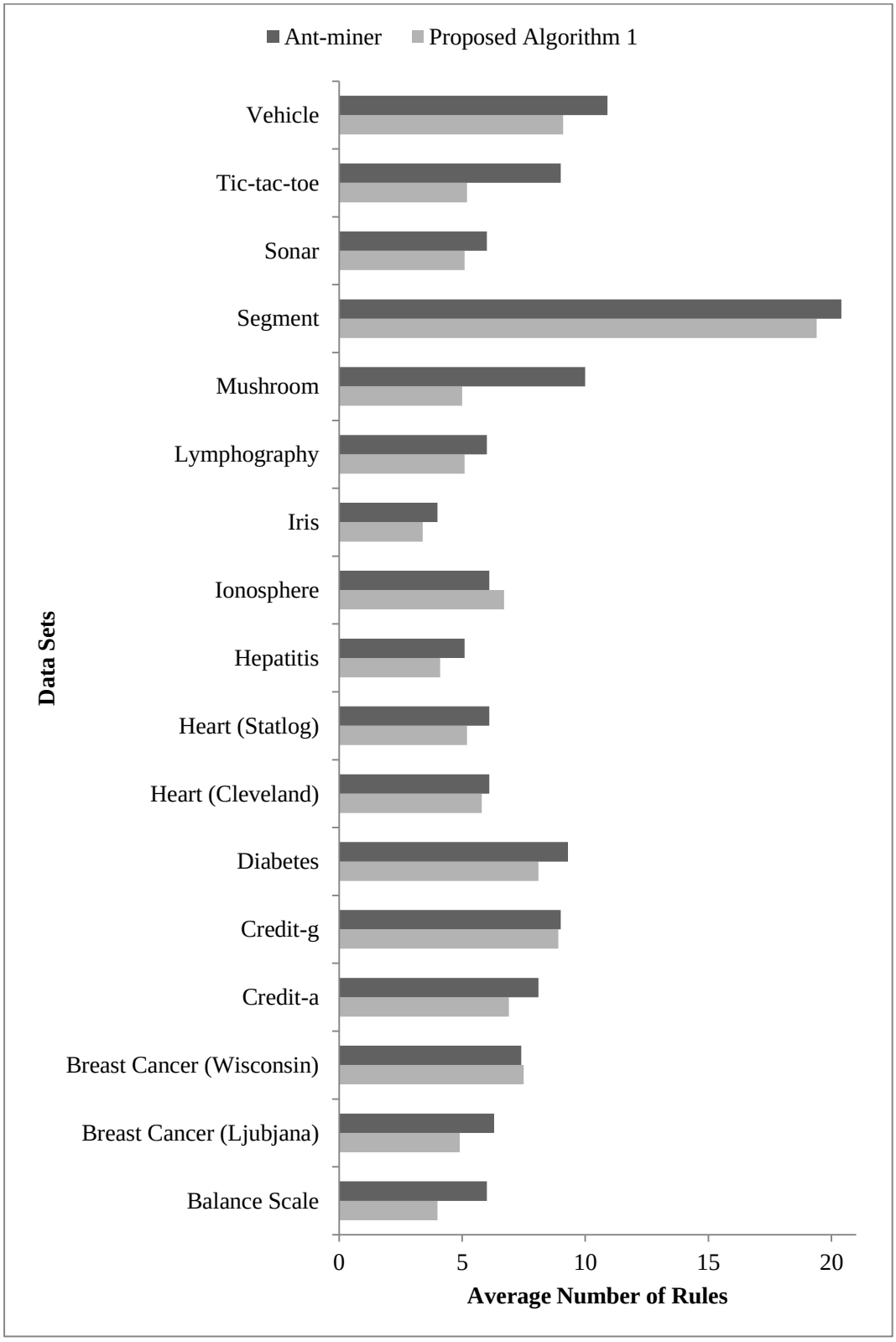

Figure 5.9: Comparison of Average Number of Rules Between Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

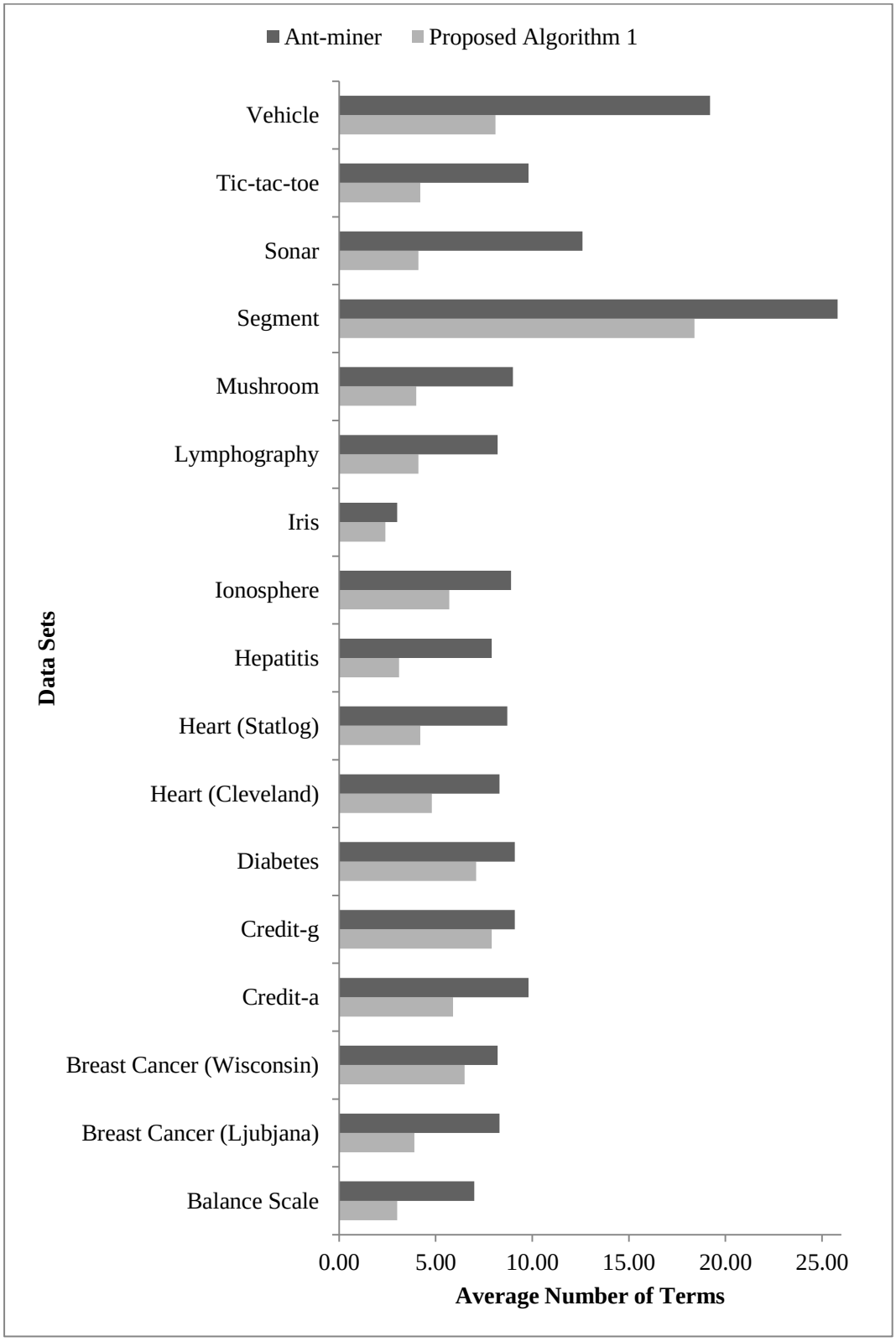

Figure 5.10: Comparison of Average Number of Terms Between Ant-miner and Proposed Algorithm 1 on Reduced Attributes Data Sets

The performance of the proposed algorithm was also tested on several other well-known rule induction algorithm discussed in Section 2.2. The tested algorithms are the conjunctive rule (Witten et al., 2011), decision table (Kohavi, 1995b), DTNB (M. Hall & Frank, 2008), JRip (Cohen, 1995) and PART (Frank & Witten, 1998). As an addition, comparison was also done with a hybrid algorithm for rule induction, the PSO/ACO2 (Holden & Freitas, 2008).

Table 5.7 shows the average predictive accuracy comparison of conjunctive rule (Witten et al., 2011), decision table (Kohavi, 1995b), DTNB (M. Hall & Frank, 2008), JRip (Cohen, 1995) and PART (Frank & Witten, 1998) against the proposed algorithm in this chapter. The mean for the average predictive accuracy for each algorithm were calculated. The conjunctive rule algorithm produced the lowest mean at 69.97%, and JRip produced the highest predictive accuracy with a mean of 83.75%. The hybrid of PSO and ACO algorithm for rule induction, the PSO/ACO2 scores a mean of 79.03%. The proposed algorithm produced average predictive accuracy with a mean that is at par with the rest of other the tested algorithm with a mean of 78.58%. Even though the proposed algorithm predictive power is only at par with other tested rule induction algorithms, the proposed algorithm produced much simpler rules as shown in Table 5.8. The proposed algorithm produced the fewest average number of rules from all data sets with a mean of 6.73. The hybrid PSO/ACO2 produced the highest mean of average number of rules with a value of 17.79 rules.

Table 5.7: Average Predictive Accuracy (%) of Conjuctive Rule, Decision Table, DTNB, JRip, PART, ACO/PSO2 and Proposed Algorithm 1

Data Sets	Conjunctive Rule	Decision Table	DTNB	JRip	PART	ACO/ PSO2	Prop. Algo. 1
Balance Scale	60.93	71.26	70.89	74.87	69.84	82.72	63.68
	± 5.4	± 3.79	± 3.74	± 4.41	± 3.97	± 4.77	± 6.19
Breast Cancer (Ljubjana)	69.03	73.73	69.94	71.45	69.41	72.62	74.11
	± 4.17	± 5.34	± 6.93	± 6.44	± 7.63	± 6.84	± 3.95
Breast Cancer (Wisconsin)	88.03	94.96	97.01	95.48	95.05	93.42	93.99
	± 3.6	± 2.54	± 1.91	± 2.3	± 2.39	± 3.79	± 3.71
Credit-a	85.51	84.67	85.48	86.38	85.25	85.31	84.64
	± 3.96	± 4.21	± 4.09	± 3.75	± 4.00	± 4.14	± 4.31
Credit-g	70	72.75	71.76	71.74	72.24	67.9	72.9
	± 0	± 3.66	± 3.71	± 3.67	± 4.24	± 5.82	± 4.13
Diabetes	73.47	77.02	77.77	77.41	76.84	72.67	74.62
	± 4.91	± 4.7	± 4.45	± 4.78	± 4.19	± 4.98	± 4.70
Heart (Cleveland)	72.98	77.22	80.81	81.65	77.49	77.38	74.83
	± 7.18	± 7.5	± 8.75	± 6.45	± 8.30	± 5.45	± 7.95
Heart (Statlog)	73	83.41	81.56	82.89	83.33	81.11	75.56
	± 8.09	± 7.03	± 6.55	± 6.67	± 7.44	± 6.16	± 7.07
Hepatitis	79.83	81.22	81.1	81.45	81.59	-	82
	± 4.95	± 8.07	± 8.31	± 9.16	± 8.62	-	± 4.62
Ionosphere	80.77	89.04	91.68	91.68	90.03	88.06	85.48
	± 6.12	± 4.58	± 4.54	± 4.84	± 4.43	± 4.91	± 5.62
Iris	66.67	93.8	94.13	94.6	94.87	94.67	96
	± 0	± 5.04	± 5.38	± 5.25	± 5.09	± 5.26	± 4.42
Lymphography	71.77	73.47	77.25	77.4	78.24	83.05	78.95
	± 11.86	± 10.5	± 10.22	± 11.08	± 10.54	± 6.67	± 7.41
Mushroom	88.68	100	99.91	100	100	99.9	91.14
	± 1.11	± 0	± 0.11	± 0	± 0.00	± 0.11	± 5.93
Segment	28.57	90.45	94.57	92.84	94.82	96.67	86.15
	± 0	± 2.06	± 1.5	± 2.28	± 1.53	± 1.17	± 2.44
Sonar	70.33	74.77	79.49	79	81.26	75.05	73.14
	± 9.94	± 11.14	± 8.92	± 8.34	± 8.51	± 9.11	± 8.96
Tic-tac-toe	69.45	73.83	69.94	97.55	93.85	100	71.09
	± 4.31	± 4.13	± 4.31	± 1.66	± 3.08	± 0	± 3.86
Vehicle	40.4	67.34	66.3	67.32	70.45	73.05	57.57
	± 2.09	± 4.36	± 3.63	± 3.94	± 4.24	± 4.45	± 6.04

Table 5.8: Average Number of Rules of JRip, PART, PSO/ACO2 and Proposed Algorithm 1

Data Sets	JRip	PART	PSO/ACO2	Proposed Algorithm 1
Balance Scale	4.67 ± 0.47	4.92 ± 0.27	26.6 ± 1.07	4 ± 0
Breast Cancer (Ljubjana)	2.87 ± 0.82	16.64 ± 3.63	12.4 ± 2.27	4.9 ± 0.3
Breast Cancer (Wisconsin)	6.79 ± 1	9.77 ± 1.53	9.9 ± 1.6	7.5 ± 0.5
Credit-a	4.89 ± 1.47	16.49 ± 4.27	22.7 ± 2	6.9 ± 1.04
Credit-g	5.08 ± 1.52	14.16 ± 3.25	54.3 ± 1.89	8.9 ± 0.94
Diabetes	4.71 ± 0.56	13.15 ± 2.32	33.4 ± 1.43	8.1 ± 0.3
Heart (Cleveland)	4.22 ± 0.66	10.43 ± 2.33	12.6 ± 0.84	5.8 ± 0.6
Heart (Statlog)	4.88 ± 0.82	9.36 ± 2.04	9.7 ± 1.34	5.2 ± 0.4
Hepatitis	2.37 ± 0.51	6.59 ± 3.71	-	4.1 ± 0.7
Ionosphere	8.12 ± 1.65	9.84 ± 1.56	3.6 ± 0.97	6.7 ± 0.78
Iris	3.1 ± 0.3	3 ± 0	3 ± 0	3.4 ± 0.49
Lymphography	6.73 ± 0.97	9.65 ± 2.25	14.7 ± 2	5.1 ± 0.3
Mushroom	8.84 ± 0.6	13.19 ± 1.34	8.7 ± 0.48	5 ± 0
Segment	46.55 ± 8.44	55 ± 4.67	21.9 ± 0.99	19.4 ± 2.29
Sonar	4.62 ± 0.83	12.85 ± 1.89	4.4 ± 1.58	5.1 ± 0.7
Tic-tac-toe	10.33 ± 1.42	33.09 ± 3.37	9 ± 0	5.2 ± 2.18
Vehicle	16.04 ± 2.53	38.83 ± 3.38	37.8 ± 1.2	9.1 ± 1.45

5.2.2 Classification of Web Data Set

This subsection experiments the proposed algorithm with the data set build in Chapter Four. The number of attributes was reduced using Correlation-based evaluation method with Random search as the search method. The results was compared to PART (Witten et al., 2011) and Ant-miner (Parpinelli et al., 2002a, 2002b).

Table 5.9 and Figure 5.11 show that the average predictive accuracy for proposed algorithm is slightly less accurate than PART and Ant-miner. However, the proposed algorithm outperformed PART in terms of simplicity of discovered rules. The

number of rules and number of terms are both far less than PART, even though it is only at par with Ant-miner.

Table 5.9: Performance Comparison for Reduced Web Data

Algorithm	Average Predictive Accuracy (%)	Average Number of Rules	Average Number of Terms
PART	94.01 ± 4.30	28.00 ± 3.4	90.00 ± 3.5
Ant-miner	93.36 ± 0.67	7.00 ± 0.15	10.1 ± .45
Proposed Algorithm 1	79.15 ± 5.17	8.80 ± 0.98	7.80 ± 0.98

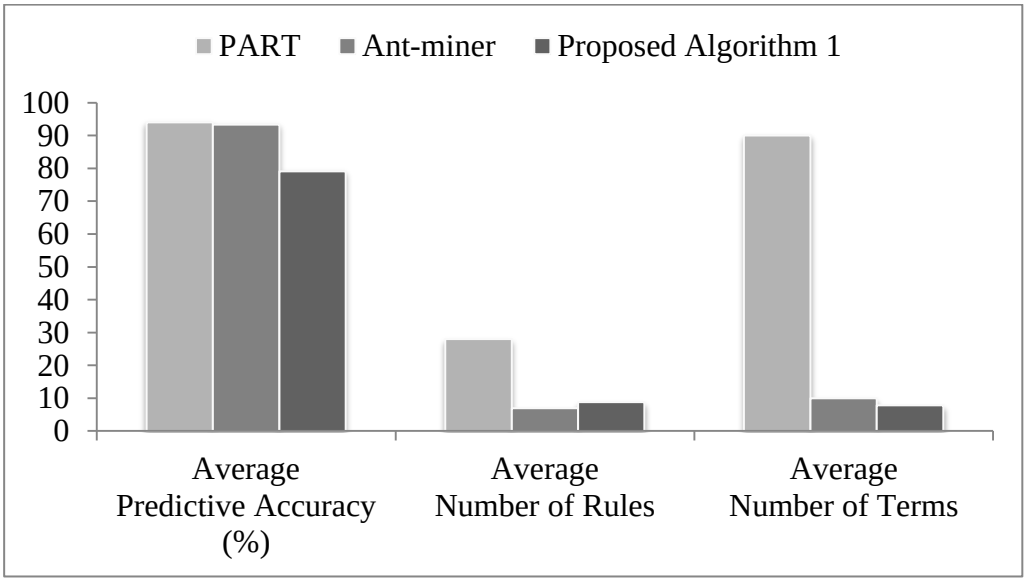

Figure 5.11: Performance Comparison for Reduced Web Data

5.3 Summary

This chapter had proposed a hybrid algorithm based on ACO and SA to discover classification rules from data. The SA acts as a local search algorithm in order for an

ant to discover a good quality rule. This chapter compares the performance of the proposed algorithm to the original Ant-miner and PART concerning the predictive accuracy, and found to be competitive with Ant-miner and PART. Moreover, the experiments in this chapter found that the rules discovered by the proposed algorithm were simpler that the compared algorithms. Therefore, SA is able to help ACO produced simpler rules, even though the .predictive accuracy is only at par with Ant-miner and PART.

Chapter Six discusses the usage of SA to select terms to be included in a rule. Besides, the proposed algorithm in Chapter Six will fix the class before selecting terms in order to improve the rule's predictive accuracy.

CHAPTER SIX
SIMULATED ANNEALING FOR BEST TERMS SELECTION

Chapter Five proposed a hybrid algorithm to optimize rule discovered by each ant in the ACO. The proposed hybrid algorithm used the SA to optimize the rule. The usage of SA improved the simplicity of the discovered rules, by producing fewer rules, and rule with lesser number of terms. Even though the accuracy of the discovered rules are competitive to the original Ant-miner (Parpinelli et al., 2002a, 2002b), the accuracy is only at par. The main problem of the proposed algorithm, as well as the original Ant-miner, is that the class is not pre-defined. The class is determined at the end of the rule construction process. Since the ant does not know the class at the beginning of the rule's construction process, the ant will heuristically find the best related term to the set of current terms in the partial rule. Thus, the algorithm may not discover any rule for the class with very few examples or in the data set with unequally distributed examples for each class.

This chapter proposed a new hybrid algorithm that fixed the class before the rule's construction process. Each term inclusion to the partial rule, will reflect the pre-defined rule's class. The proposed algorithm used a sequential covering based algorithm to extract classification rules for known class. SA is used to optimize the terms selection process, in each rule's construction process, which differ from the proposed algorithm in Chapter Five. The proposed algorithm in Chapter Five uses the SA to optimized rule discovered by each ant. The pre-defining of class in constructing a rule enabled the use of simpler heuristic function for terms selection, as well as the fitness function to measure the quality of a rule.

The following section discussed the construction of the proposed algorithm, followed by experiments setup and the discussion on the results from the experiments.

6.1 Simulated Annealing for Term Selection

The proposed algorithm uses sequential covering algorithm as the basis algorithm. The algorithm extracts the rules by training the examples in the training data set, E, one class at a time. The list of k classes are ordered based on their prevalence, C_0. Class prevalence is the fraction of examples in the training set that belongs to a particular class. The k-th class in the classes list is not used to extract rules, but is set as default rule, $r_{default}$. A default rule is a rule that contains no antecedent part, but only the consequent part, placed at the bottom of the list of discovered rules. If there are only two classes available, the algorithm performs the training for class with the highest prevalence only, and set the second class as the default rule.

Figure 6.1 summarizes sequential covering algorithm used in the proposed algorithm, and Figure 6.2 depicted the flow chart for the algorithm. The algorithm starts with an empty list of discovered rules, R. For each class, c_i in C_0, except for the last class c_k, an ACO algorithm is used to extract the best rule, that covers the current set of training examples. After ACO has found one best rule for the current set of training examples, r, the examples covered by the rule are removed from the training set. Then, the constructed rule, r is added to the list of discovered rules, R. This procedure repeats until the number of examples in the training set is less than a pre-defined threshold value, *max_uncovered*. In order to extract rules for the next i-

th class, c_i, the original set of training examples is restored. The algorithm repeats the same procedure for the next class.

Let E be the training examples.
Let $Terms$ be the set of attributes-values pairs, $term_{ij}$.
Let C_0 be an ordered set of classes $\{c_1, c_2, \ldots, c_k\}$.
Initialize empty set of discovered rules $R = \{\}$.
for each class $c_i \in C_0 \setminus \{c_k\}$:
    while the number of examples > *max_uncovered*:
        $r \leftarrow ACO(E, Terms, c_i)$.
        Remove examples from E covered by r.
        $R \rightarrow R \vee r$.
Restore the list of raining examples.
$R_{sorted} \leftarrow \text{sorted}(R)$.
$R \rightarrow R \vee r_{default}$.

Figure 6.1: Sequential Covering Algorithm with Pre-Defined Class

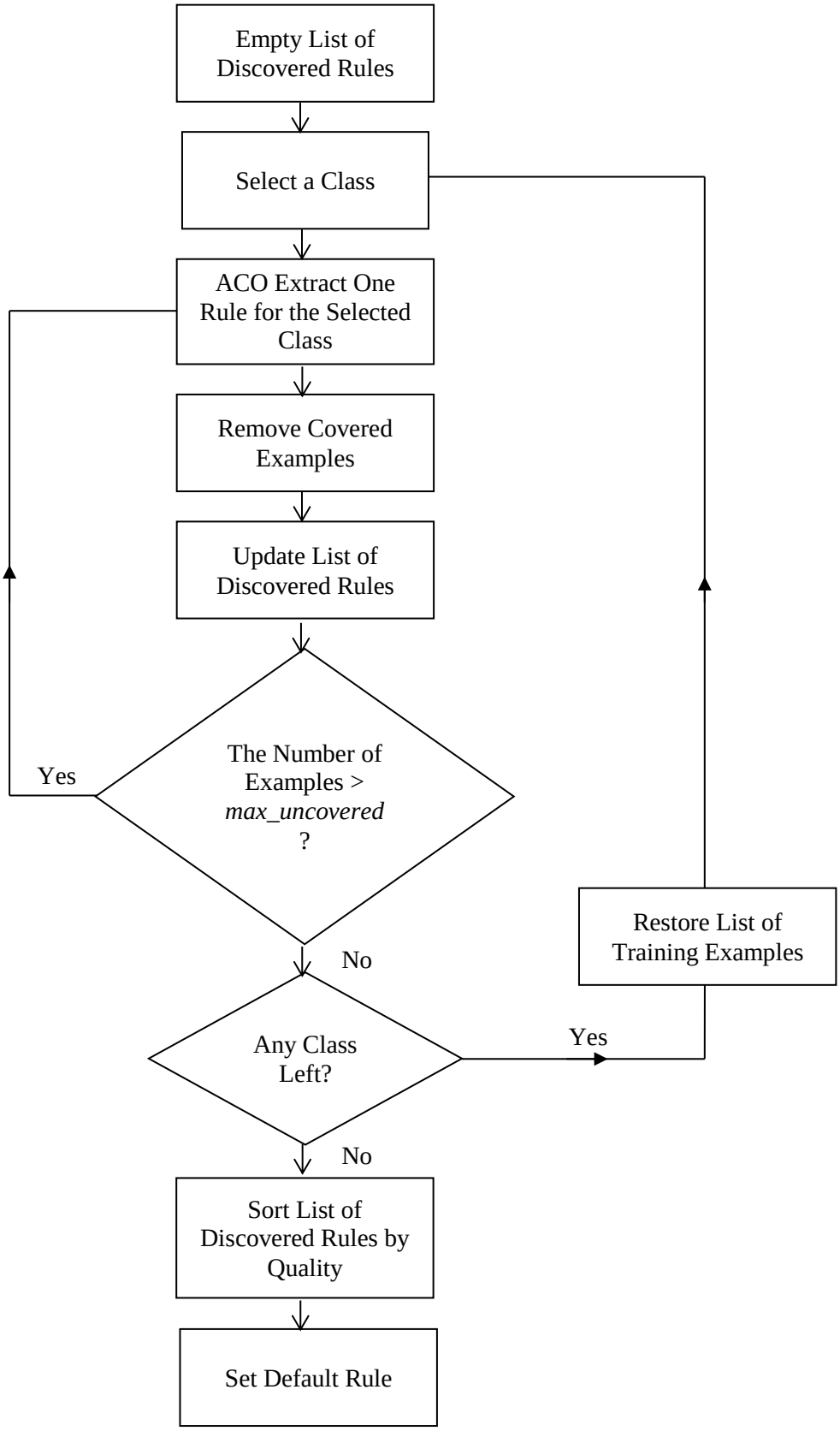

Figure 6.2: Sequential Covering with Pre-Defined Class Flow Chart

The algorithm sorted the list of discovered rules for all $k-1$ classes in descending order of the quality. By disregarding the rule's class, the algorithm place the highest quality rule at the top of the list, followed by the second highest quality rule, and so on. In order to predict a new unseen example, the algorithm search for a rule in order, in the list of discovered rules, which cover the example. The class for the rule that covers the new example is set as the predicted class for the new example. If there is no rule found to cover the new example, the default rule is used. In other words, the algorithm predicted the new example to have the same class with the class that has the minority number of examples.

The proposed algorithm uses ACO, depicted by Figure 6.3, as the Learn-One-Rule function in the sequential covering algorithm to extract a rule from the current set of training examples. Figure 6.4 visualize the flow chart for ACO. Each ant colony in ACO contains only one ant each, and discovers one rule each. However, the algorithm may stop earlier if the set of rules have converged. Therefore, the total number of rules is less than or equal to the number of ant colonies.

The convergence of the rules occurs when p number of consequent rules has the same value of quality. The value p is pre-defined earlier before the algorithm starts. The algorithm updates the pheromone level for all terms after each ant colony has discovered a rule. The chosen terms will have a higher level of pheromone compared to the other terms. Since ants tend to choose terms with higher pheromone level, the next ant colony will discover a rule that is biased by the rule discovered by the

previous ant colony (foraging behaviour of ants). Finally, the rules discovered by the consequents ant colony will converge to the same rule.

Initialize ant colonies.
Initialize empty set of rules $R_{colonies} = \{\}$.
for all ant colonies, while rules are not converged:
    Extract one rule, r.
    Update pheromone level for all terms.
    $R_{colonies} \rightarrow R_{colonies} \vee r$
Select best rule from $R_{colonies}$.

Figure 6.3: ACO Algorithm to Extract One Rule

The algorithm initialized all pheromones level equally using Equation 6.1, as

$$\tau_{ij} = \frac{1}{\sum_{i=1}^{a} b_i} \tag{6.1}$$

where

- a is the total number of attributes, and
- b_i is the total number of values for attribute i.

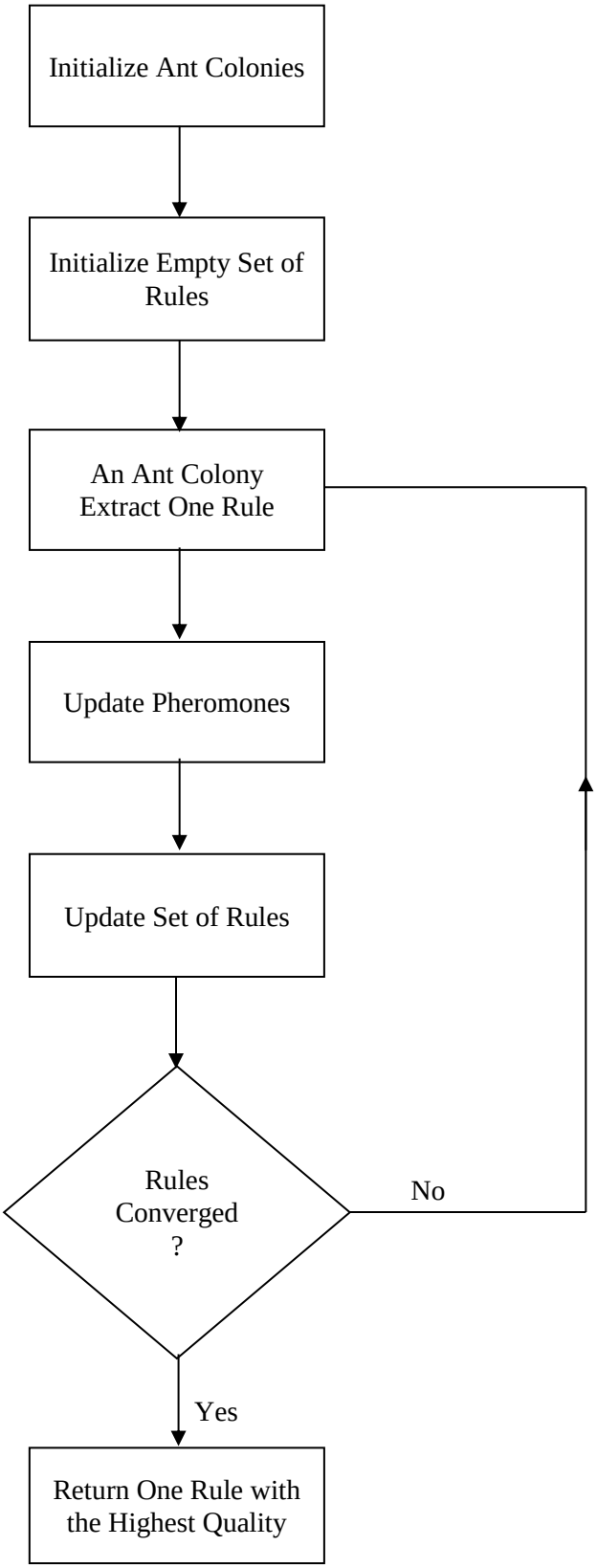

Figure 6.4: ACO Algorithm to Extract One Rule Flow Chart

The pheromones level for terms used in the discovered rule, by each ant colony are colony are increased. The pheromone level update at time t for term with attribute i and value j, $term_{ij}$ is carried out using

$$\tau_{ij}(t) = \tau_{ij}(t-1) + \tau_{ij}(t-1) * Q \tag{6.2}$$

where

- τ_{ij} is the updated pheromone level for $term_{ij}$,
- $\tau_{ij}(t-1)$ is the current pheromone level for $term_{ij}$, and

Q is quality of the newly constructed rule, defined by Equation 6.4.

Subsequently, to mimic the evaporation of pheromone by the real ants, the pheromone level for all terms are normalize by dividing the pheromone level for each term by the sum of all pheromone level, as

$$\tau_{ij}(t) = \frac{\tau_{ij}(t-1)}{\sum_{i=1}^{m}\sum_{j=1}^{n}\tau_{ij}(t-1)} \tag{6.3}$$

where

- τ_{ij} is the updated pheromone level for $term_{ij}$,
- $\tau_{ij}(t-1)$ is the current pheromone level for $term_{ij}$,

- m is the total number of attributes, and

- n is the total number of values for each attribute.

After all ant colonies have discovered the rules, or when the rules have converged, where p number of consequent rules have the same value of quality, the algorithm will select one best rule from the set of all rules. The best rule is a rule that has the highest quality among all the rules discovered by ACO.

The algorithm calculate the rule's quality using a fitness function, defined by Equation 6.4,

$$Q = \frac{1 + TP}{1 + k + TP + FP} \tag{6.4}$$

where

- TP is the number of examples covered by the rule, and having the same class from the class predicted by the rule,

- FP is the number of examples covered by the rule, and having a different class from the class predicted by the rule, and

- k is the total number of classes.

Since the ant knows the class when constructing the rule, the proposed algorithm used a simpler fitness function as compared to the fitness function used in the

original Ant-miner proposed by Parpinelli. Therefore, the relationship between terms is not important, since the selected terms for inclusion are already from the same group. The fitness function is the proportion of the number of correctly covered examples by the rule to the number of covered examples of the rule. However, if there are no examples covered by the rule, the equation is undefined, since both the numerator and denominator are both zero. Hence, Equation 6.4 adds one to the numerator, and $1+k$ to the denominator. As a consequent, if there are no examples covered by the rule, the rule's quality is still be given a value.

Each ant constructs a rule using terms selection procedure. Figure 6.5 summarizes the terms selection procedure, while Figure 6.6 depicts the flow chart for the whole processes. The terms selection procedure starts by initializing a list of terms, and an empty partial rule, but with a known class, c_i. Terms are added to the partial rule one by to the partial rule each time, from a set of terms, *Terms*. SA finds the best term, t, to be included into the partial rule, r, each time. The terms selection procedure stops when there are no more terms with unused attributes left; or the new term addition does not improve the quality of the rule. There is one caveat when constructing the rule. A rule cannot contain two or more terms with the same attribute. Therefore, the procedure stops when there are no more attributes left.

Rule pruning helps to avoid the over-fitting to the training data, since the algorithm selects terms greedily based on a heuristic measure. Moreover, the pruning procedure may improve the simplicity of the rule, by reducing the number of terms in a rule. The algorithm prunes each constructed rule if there are more than one term

available in the rule's antecedent, and remove terms one at a time. After each term removal, the new rule's quality is calculated. If the rule's quality has improved, the pruning procedure removes the term permanently. This process continues until there is only one term left, or no terms removal would improve the rule's quality. This is different from the pruning procedure used by the original Ant-miner by Parpinelli (2002a, 2002b) and the one used in Chapter Five. The pruning procedure used in this proposed algorithm does not change the class for the rule. In other words, the removal of a term is to improve the rule's quality subject to a known class.

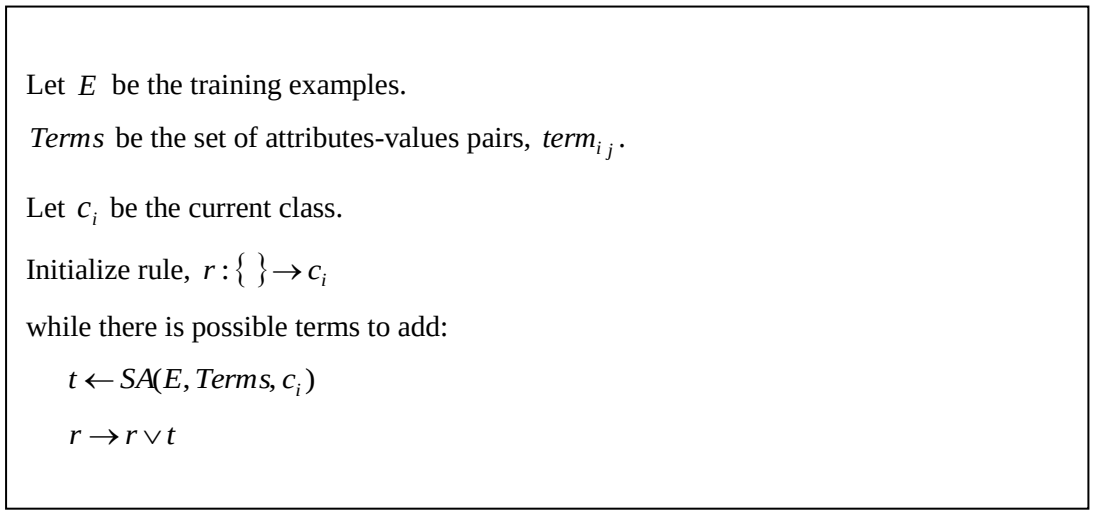

Figure 6.5: Terms Selection Procedure

This proposed terms selection procedure produced a rule with an optimize group of terms for a selected class, optimized by SA. In other words, SA optimizes each term inclusion into the partial rule. Figure 6.5 depicted the flow chart for the SA procedure to select a term. The SA starts by first initializing a very high value temperature. For a set of available terms, a known class, and an empty partial rule, the algorithm select an optimize term, and add it to the partial rule.

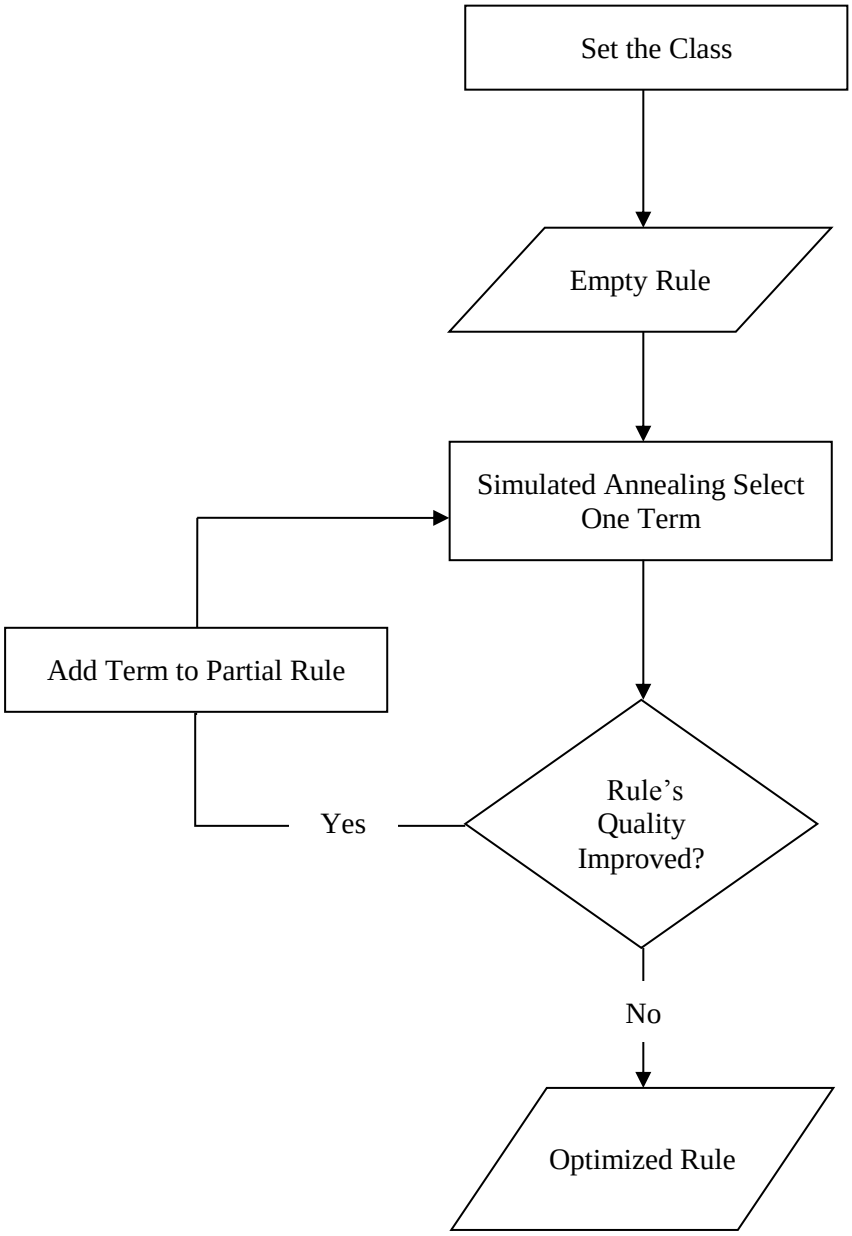

Figure 6.6: Terms Selection Procedure Flow Chart

Figure 6.7 depicts the flow chart for selecting an optimize term with SA. The simulated algorithm optimized the selected term, by selecting a term to be tested, and check whether the new selected term increase the quality of the rule. Equation 6.4 defines the fitness function to calculate the rule's quality. If the new term addition makes the rule's quality improved, the term is marked as the best term so far.

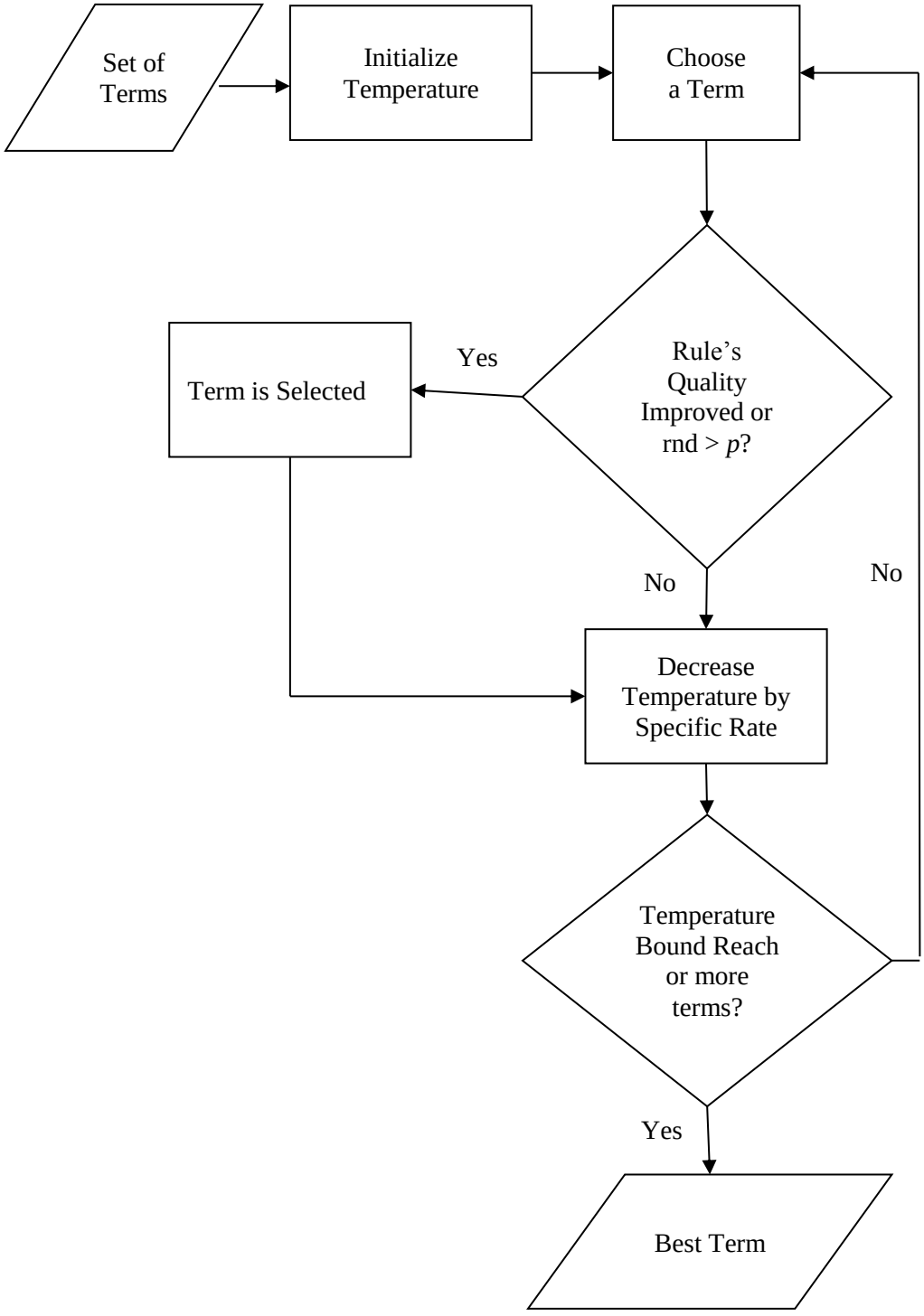

Figure 6.7: SA Algorithm to Select One Term Flow Chart

SA reduced the temperature based on a pre-defined cooling factor, after selecting and testing a term. The cooling process continues until the algorithm has tested all terms, or the temperature has reduced to a value less than a pre-defined lower bound temperature value.

The algorithm uses a roulette wheel procedure to select a term to test each time. The procedure is as follows. First, generate a random number. For each term in the terms' list, cumulate all the probability calculated using Equation 6.5. The cumulating process stops if the current total probability for the current term in the cumulating process exceeds the random number. The algorithm uses this current term for terms selection testing for inclusion into the partial rule.

The probabilities for all terms are calculated using a random proportional transitional rule, as used in the Ant System for the Travelling Salesman problem (Dorigo et al., 1996), defined by

$$P_{ij} = \frac{\tau_{ij} \times \eta_{ij}}{\sum_{i=1}^{a} \sum_{j=1}^{b_i} \tau_{ij} \times \eta_{ij}} \tag{6.5}$$

where

- η_{ij} is the heuristic value for term with attribute i and value j,
- τ_{ij} is the pheromone level for term with attribute i and value j, defined by Equation 6.3, and

- b_i is the total number of value for attribute i.

The probability depends on the heuristic value and current pheromone level for each term. Over time, the pheromone level for the higher quality terms will increase, and hence increase the chances of selecting those terms. The heuristic value for terms remains the same, but has a different value for the same terms with different class.

This algorithm proposed a simpler heuristic function to calculate the heuristic for each term. Equation 6.6 defines the heuristic function used by the proposed algorithm in this chapter.

$$\eta_{ij} = \frac{freq_{ij}^{w}}{\sum_{i=1}^{a}\sum_{j=1}^{b_i} freq_{ij}^{w}} \tag{6.6}$$

where

- $freq_{ij}^{w}$ is the number of examples for term with attribute i and value j for a given class w, and

- b_i is the total number of value for attribute i.

The SA also gives chance to a term with lower quality rule to be accepted. Besides selecting a term that improves the rule's quality, if a generated random number, *rndNum* exceeded the probability defined by Equation 6.7, the term is also accepted.

$$p = e^{\frac{-q_1 - q_2}{T}} \tag{6.7}$$

where

- q_1 and q_2 are the quality for the previous and current partial rules respectively, and
- T is the current temperature value.

The temperature starts very high. Therefore, p is almost one, and all rules have almost the same probability to be chosen. As the temperature decreases slowly, the difference between the previous and current quality becomes significant. Hence, a term with a slightly lower quality rule will be preferred over much lower ones.

6.2 Experiment and Results

This study conducts experiments on seventeen (17) data sets from a UCI repository (Asuncion & Newman, 2007), as well as the reduced Web data set from Chapter Four. The experiments are undertaken using a *ten*-fold cross-validation technique (Kohavi, 1995a). Each data set is randomly shuffled and split into ten approximately equally sized subsets. Each experiment uses one subset as the test set, and trains the rest of the subsets. The experiment repeats the processes ten times, producing ten individual sets of performance statistics, the predictive accuracy. These performance statistics are averaged and the standard deviations for each of the performance statistics are calculated.

The proposed algorithm, like the original Ant-miner (Parpinelli et al., 2002a, 2002b), cannot cope with non-discrete attributes, only nominal attributes. Hence, the experiments discretized non-discrete attributes in the first place; using a discretization method used in this study based on the method proposed by Fayyad & Irani (1993). The experiments used five hundred ant colonies with one ant per colony.

6.2.1 Classification of 17 Data Sets from UCI Repository

Table 6.1 and Figure 6.8 shows the predictive accuracy of the proposed algorithm as compared to the original Ant-miner (Parpinelli et al., 2002a, 2002b) as well as the PART (M. A. Hall et al., 2009; Witten et al., 2011). The proposed algorithm discovers rules with a better accuracy rate than Ant-miner on 82.4% (14 out of 17) of the 17 data sets, with a huge improvement on three data sets: Vehicle: Tic-tac-toe and segment data sets, with an average percentage difference of 15.4 %. Moreover, the average percentage difference for the three data sets (Balance Scale, Ljubljana and Iris) where the original Ant-miner (Parpinelli et al., 2002a, 2002b) wins, is very small for an average of 1.2 %.

In Chapter Five, the proposed algorithm produced rules that are good in terms of simplicity as compared to Ant-miner, but, with predictive accuracy that is only at par with other rule induction algorithms. However, the proposed algorithm in this chapter has performed better in terms of the predictive accuracy compared to Ant-miner, but, produced much larger rule set than Ant-miner as shown in Table 6.2 and

Figure 6.9. The proposed algorithm also produced much longer rules on all data sets compared to Ant-miner as shown in Table 6.3 and Figure 6.10.

Table 6.1: Average Predictive Accuracy of Ant-miner and Proposed Algorithm 2

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	71.27 ± 1.54	71.04 ± 3.91
Breast Cancer (Ljubljana)	73.74 ± 2.69	72.39 ± 9.09
Breast Cancer (Wisconsin)	94.99 ± 0.49	96.14 ± 2.93
Credit-a	84.93 ± 1.21	85.80 ± 2.58
Credit-g	70.50 ± 2.01	75.50 ± 3.29
Diabetes	73.55 ± 1.85	76.70 ± 4.11
Heart (Cleveland)	76.78 ± 2.09	81.78 ± 7.29
Heart (Statlog)	77.04 ± 2.64	81.11 ± 9.14
Hepatitis	83.07 ± 2.63	83.25 ± 5.79
Ionosphere	89.25 ± 1.65	90.89 ± 3.98
Iris	95.33 ± 1.42	93.33 ± 8.43
Lymphography	76.92 ± 3.94	78.29 ± 6.88
Mushroom	97.45 ± 0.62	99.01 ± 2.55
Segment	83.94 ± 1.00	92.42 ± 1.60
Sonar	78.27 ± 3.36	80.36 ± 7.40
Tic-tac-toe	71.17 ± 1.74	97.18 ± 1.88
Vehicle	58.29 ± 1.39	69.98 ± 4.04

Table 6.2: Average Number of Rules of Ant-miner and Proposed Algorithm 2

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	7.80 ± 0.20	19.20 ± 1.72
Breast Cancer (Ljubljana)	6.10 ± 0.18	16.40 ± 1.02
Breast Cancer (Wisconsin)	7.40 ± 0.22	11.90 ± 0.83
Credit-a	7.40 ± 0.22	20.40 ± 2.33
Credit-g	9.10 ± 0.18	48.00 ± 3.97
Diabetes	9.30 ± 0.21	29.30 ± 1.10
Heart (Cleveland)	6.40 ± 0.27	12.80 ± 0.87
Heart (Statlog)	5.90 ± 0.18	12.50 ± 1.12
Hepatitis	4.80 ± 0.20	7.80 ± 0.98
Ionosphere	6.20 ± 0.20	12.60 ± 1.36
Iris	4.20 ± 0.20	4.70 ± 0.46
Lymphography	6.30 ± 0.26	7.90 ± 0.83
Mushroom	7.70 ± 0.33	24.90 ± 1.76
Segment	17.70 ± 0.47	57.60 ± 2.42
Sonar	6.00 ± 0.15	11.90 ± 1.04
Tic-tac-toe	7.90 ± 0.55	33.30 ± 3.32
Vehicle	12.00 ± 0.33	41.30 ± 1.35

Table 6.3: Average Number of Terms of Ant-miner and Proposed Algorithm 2

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	10.60 ± 0.40	42.90 ± 5.15
Breast Cancer (Ljubljana)	7.90 ± 0.38	33.20 ± 3.74
Breast Cancer (Wisconsin)	7.70 ± 0.26	18.90 ± 2.02
Credit-a	10.60 ± 0.45	53.50 ± 8.88
Credit-g	15.60 ± 0.40	127.90 ± 14.82
Diabetes	10.30 ± 0.54	65.70 ± 3.90
Heart (Cleveland)	10.10 ± 0.66	29.10 ± 3.53
Heart (Statlog)	8.80 ± 0.53	27.60 ± 3.98
Hepatitis	8.50 ± 0.48	15.70 ± 3.47
Ionosphere	9.10 ± 0.78	24.50 ± 3.75
Iris	3.20 ± 0.20	4.80 ± 1.08
Lymphography	10.00 ± 0.68	16.50 ± 2.97
Mushroom	8.10 ± 0.46	37.00 ± 2.90
Segment	23.70 ± 0.91	121.60 ± 5.97
Sonar	11.70 ± 0.37	25.70 ± 3.61
Tic-tac-toe	9.50 ± 1.41	96.50 ± 11.14
Vehicle	24.50 ± 1.17	116.80 ± 7.14

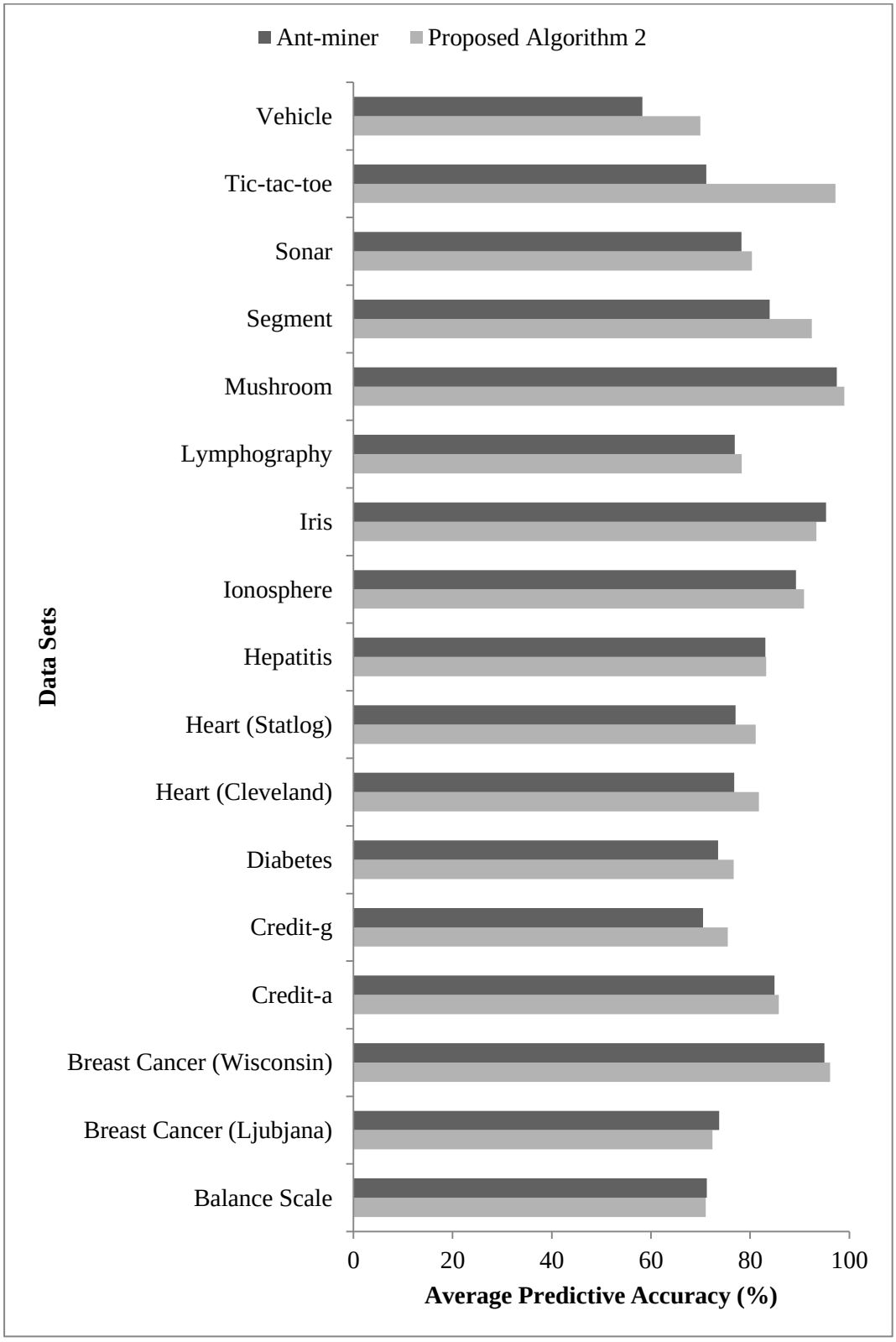

Figure 6.8: Comparison of Average Predictive Accuracy between Ant-miner and Proposed Algorithm 2

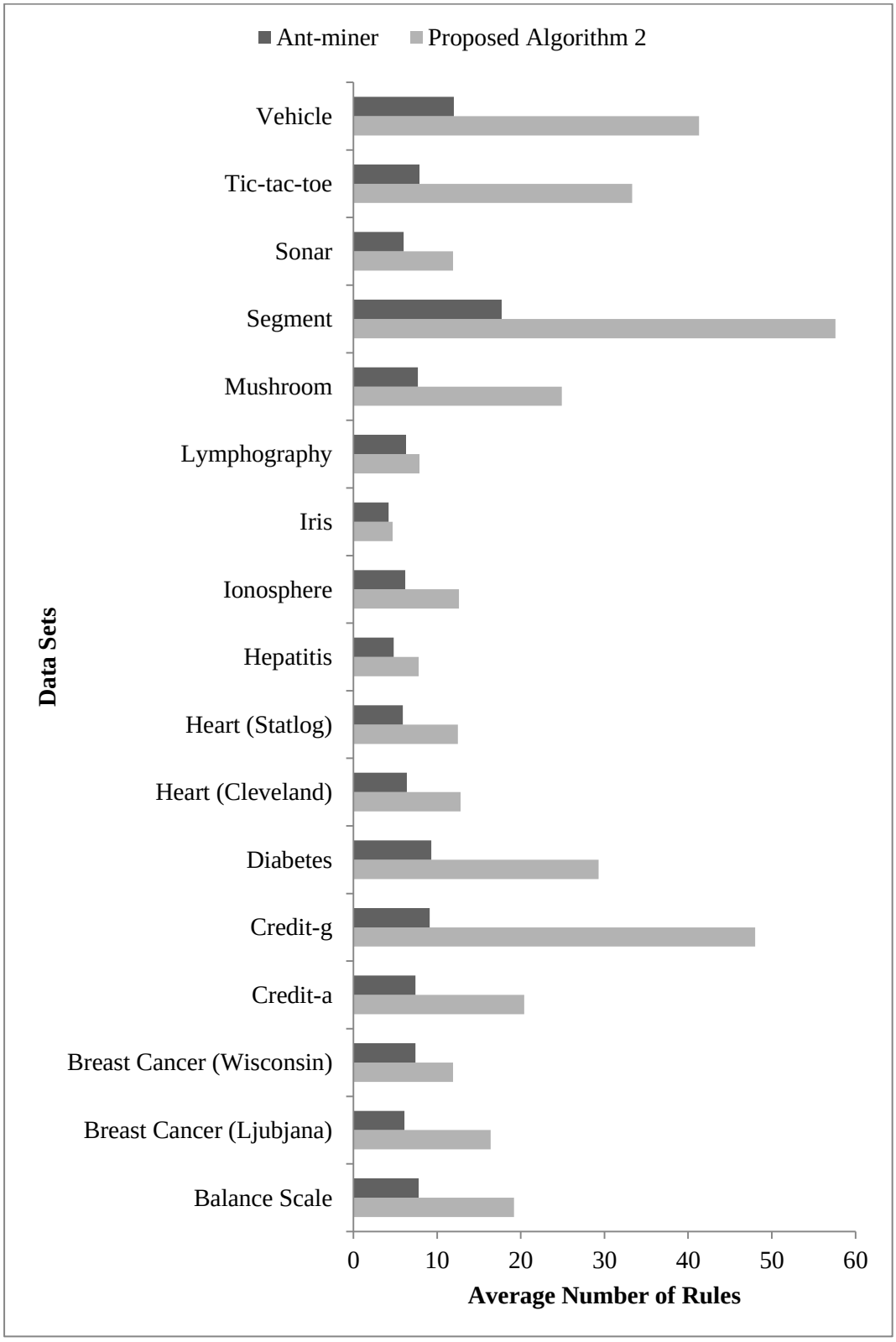

Figure 6.9: Comparison of Average Number of Rules between Ant-miner and Proposed Algorithm 2

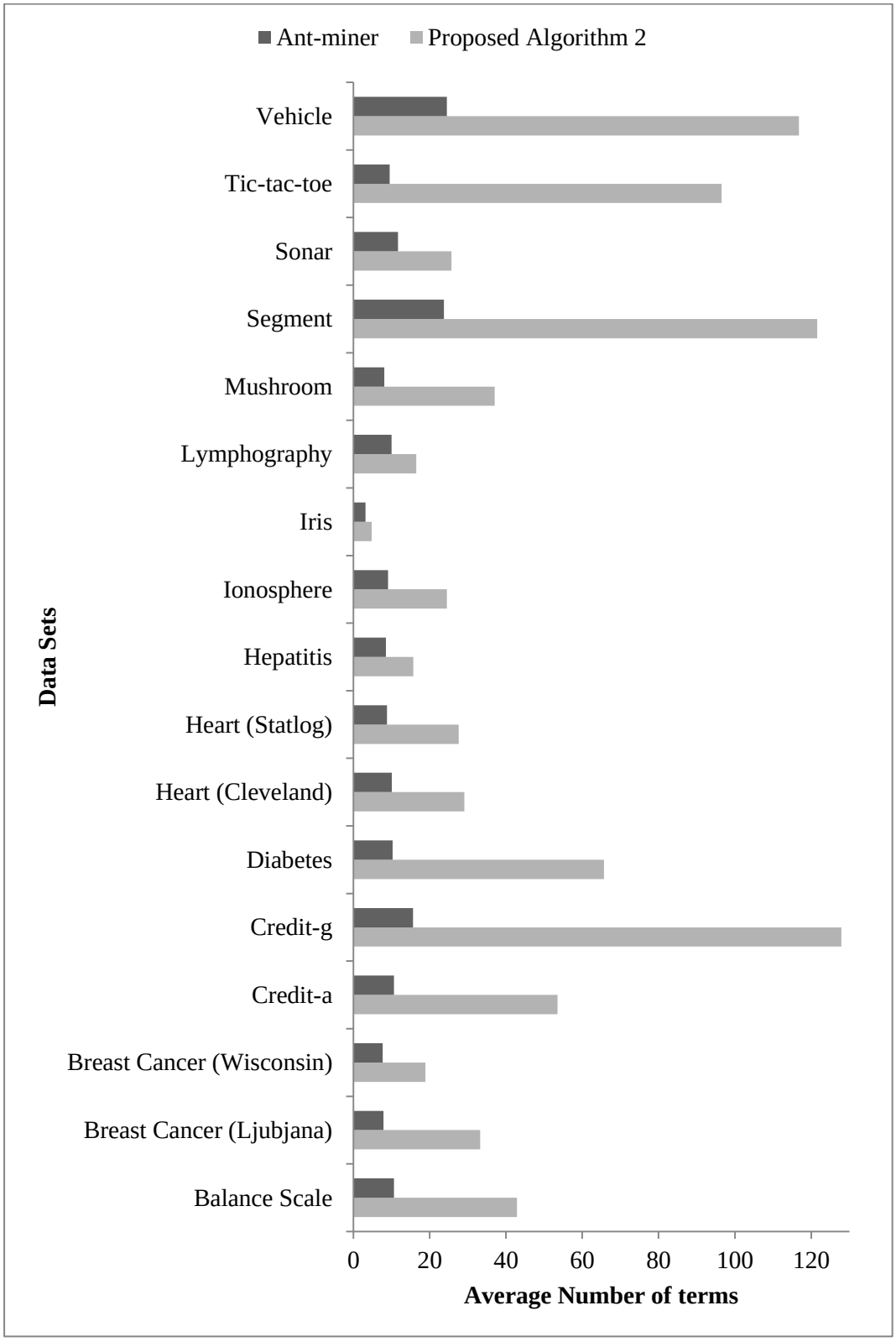

Figure 6.10: Comparison of Average Number of Terms between Ant-miner and Proposed Algorithm 2

In order to see the effect of reducing attributes as proposed in Chapter 4 where the removal of the unnecessary attributes increased the predictive power of the classification model constructed by Ant-miner (Parpinelli et al., 2002a, 2002b) as well as improving the simplicity of the classification rules, this section also tested the proposed algorithm on the seventeen UCI reduced data sets using the methods from Chapter 4: the combination of Correlation-based evaluation with Random Search as the search method. The results were compared against the Ant-miner.

According to the average accuracy laid out in Table 6.4 and depicted by Figure 6.11, the proposed algorithm wins on all 17 data sets (100%).. However, the mean for all average accuracy is slightly lower (82.5%) compared to when using the proposed algorithm to construct the classification model without reducing the attributes (83.8%) by 1.3%.

However, the simplicity (the less number of rules and the less number of terms per rule, the better the simplicity of the rules) of the constructed rules improved when reducing the number of attributes as shown in Table 6.5 and Figure 5.12, where the mean of the average number of rules was reduced from 21.91 to 19.28.

The same goes for the average number of terms per rule. The mean for the average number of terms per rule, when the method of attributes reduction is applied on the data sets initially, was reduced by 10.54. The results were shown in Table 6.6 and Figure 6.13.

Hence, when applying the attribute reduction on the data sets before using the proposed algorithm in this section, even though the predictive power is slightly reduced, the simplicity of the constructed rules increased.

Table 6.4: Average Predictive Accuracy of Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	75.17 ± 1.84	75.22 ± 3.58
Breast Cancer (Ljubljana)	76.5 ± 2.76	72.80 ± 7.85
Breast Cancer (Wisconsin)	94.71 ± 0.83	95.14 ± 2.94
Credit-a	86.09 ± 1.11	85.94 ± 3.43
Credit-g	73.7 ± 1.51	73.80 ± 6.11
Diabetes	74.71 ± 2	77.74 ± 3.61
Heart (Cleveland)	80.64 ± 2.47	82.44 ± 7.81
Heart (Statlog)	80 ± 2.6	81.11 ± 7.11
Hepatitis	76.31 ± 3.59	82.58 ± 7.44
Ionosphere	86.25 ± 2.04	89.47 ± 4.02
Iris	95.33 ± 1.42	92.67 ± 5.54
Lymphography	70.92 ± 2.82	75.62 ± 6.32
Mushroom	98.52 ± 0.17	99.00 ± 0.33
Segment	86.23 ± 1.27	92.86 ± 1.26
Sonar	78.88 ± 2.35	81.79 ± 9.41
Tic-tac-toe	70.79 ± 2.05	78.39 ± 3.30
Vehicle	58.98 ± 1.64	65.84 ± 3.14

Table 6.5: Average Number of Rules of Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	6.00 ± 0.00	16.30 ± 0.78
Breast Cancer (Ljubljana)	6.30 ± 0.15	14.00 ± 1.55
Breast Cancer (Wisconsin)	7.40 ± 0.22	11.80 ± 0.75
Credit-a	8.10 ± 0.31	18.00 ± 2.32
Credit-g	9.00 ± 0.21	29.50 ± 2.20
Diabetes	9.30 ± 0.15	27.20 ± 1.66
Heart (Cleveland)	6.10 ± 0.18	12.00 ± 0.77
Heart (Statlog)	6.10 ± 0.18	11.80 ± 0.75
Hepatitis	5.10 ± 0.18	7.50 ± 0.67
Ionosphere	6.10 ± 0.23	12.50 ± 0.92
Iris	4.00 ± 0.00	4.00 ± 0.00
Lymphography	6.00 ± 0.26	7.60 ± 0.80
Mushroom	10.00 ± 0.00	16.60 ± 0.66
Segment	20.40 ± 0.96	56.00 ± 1.61
Sonar	6.00 ± 0.15	12.40 ± 0.80
Tic-tac-toe	9.00 ± 0.63	34.50 ± 2.50
Vehicle	10.90 ± 0.41	36.10 ± 2.43

Table 6.6: Average Number of Terms of Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

Data Sets	**Ant-miner**	**Proposed Algorithm 2**
Balance Scale	7.00 ± 0.00	28.00 ± 2.19
Breast Cancer (Ljubljana)	8.30 ± 0.42	24.60 ± 4.00
Breast Cancer (Wisconsin)	8.20 ± 0.36	19.80 ± 1.40
Credit-a	9.80 ± 0.29	41.00 ± 6.48
Credit-g	9.10 ± 0.38	56.50 ± 6.70
Diabetes	9.10 ± 0.43	62.10 ± 5.63
Heart (Cleveland)	8.30 ± 0.30	22.80 ± 2.60
Heart (Statlog)	8.70 ± 0.37	25.10 ± 1.87
Hepatitis	7.90 ± 0.60	12.40 ± 1.85
Ionosphere	8.90 ± 0.41	22.80 ± 2.82
Iris	3.00 ± 0.00	3.00 ± 0.00
Lymphography	8.20 ± 0.68	12.90 ± 2.30
Mushroom	9.00 ± 0.00	24.70 ± 1.35
Segment	25.80 ± 1.72	110.70 ± 3.44
Sonar	12.60 ± 0.56	33.80 ± 4.45
Tic-tac-toe	9.80 ± 1.15	88.70 ± 9.03
Vehicle	19.20 ± 1.26	89.80 ± 6.24

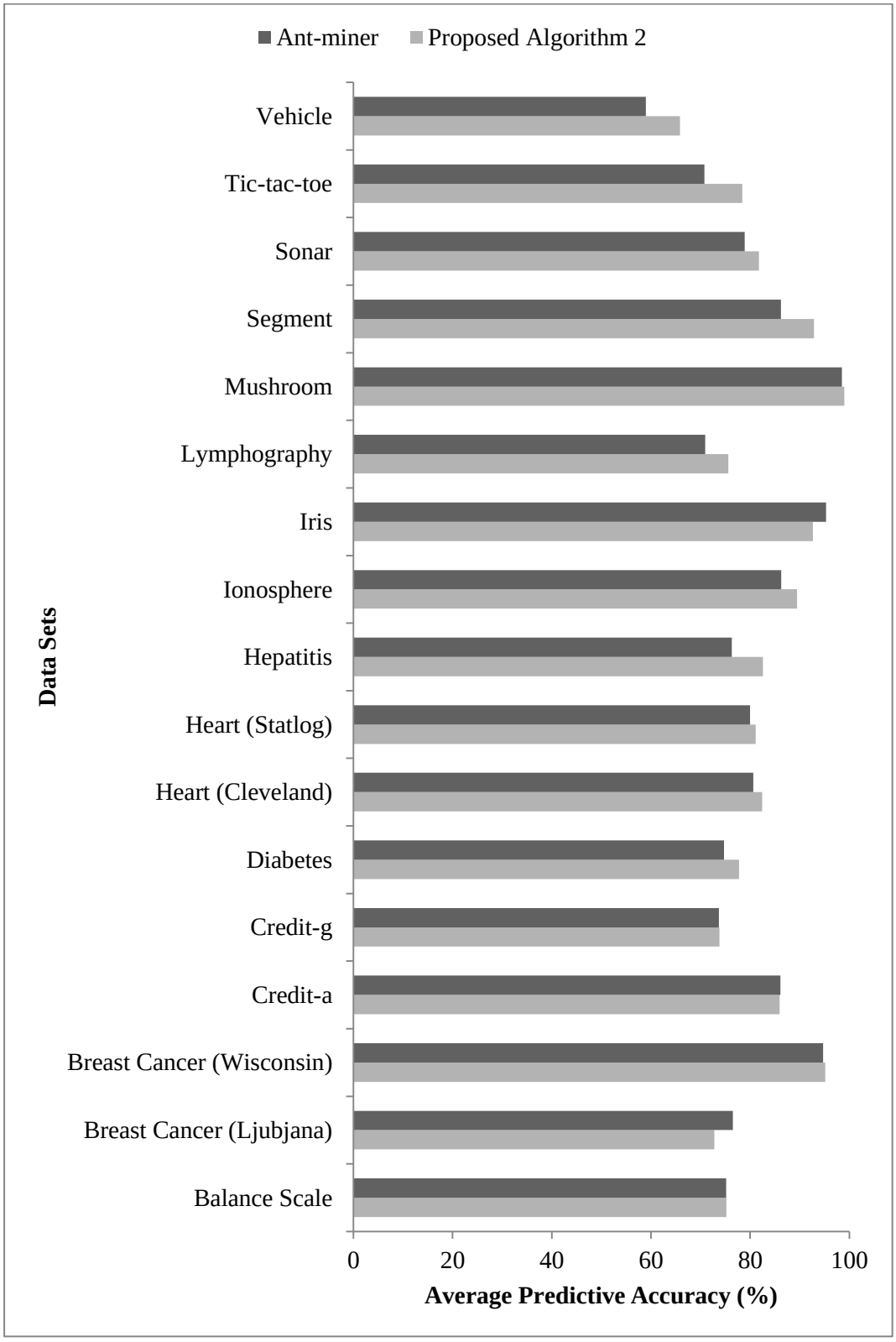

Figure 6.11: Comparison of Average Predictive Accuracy between Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

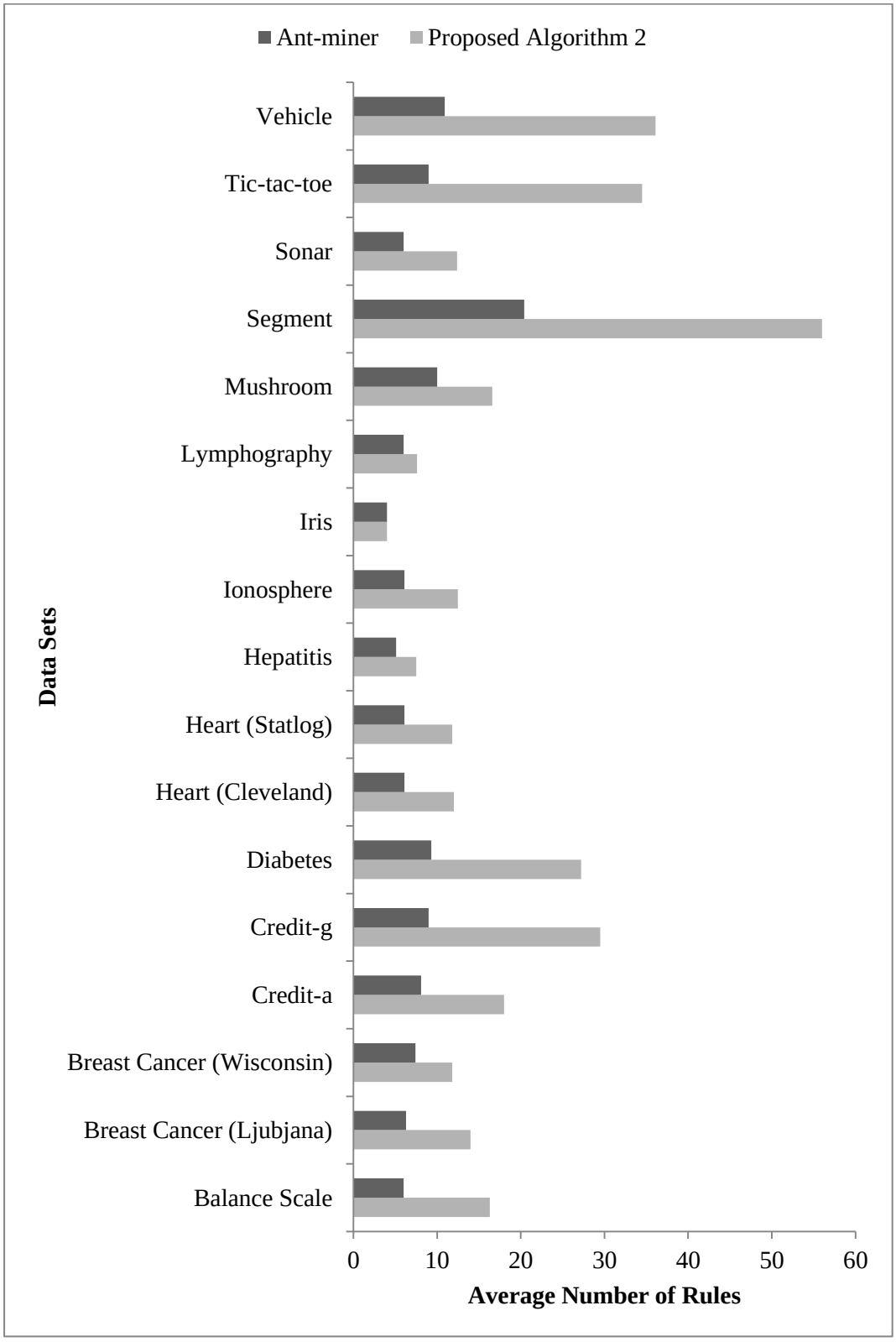

Figure 6.12: Comparison of Average Number of Rules between Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

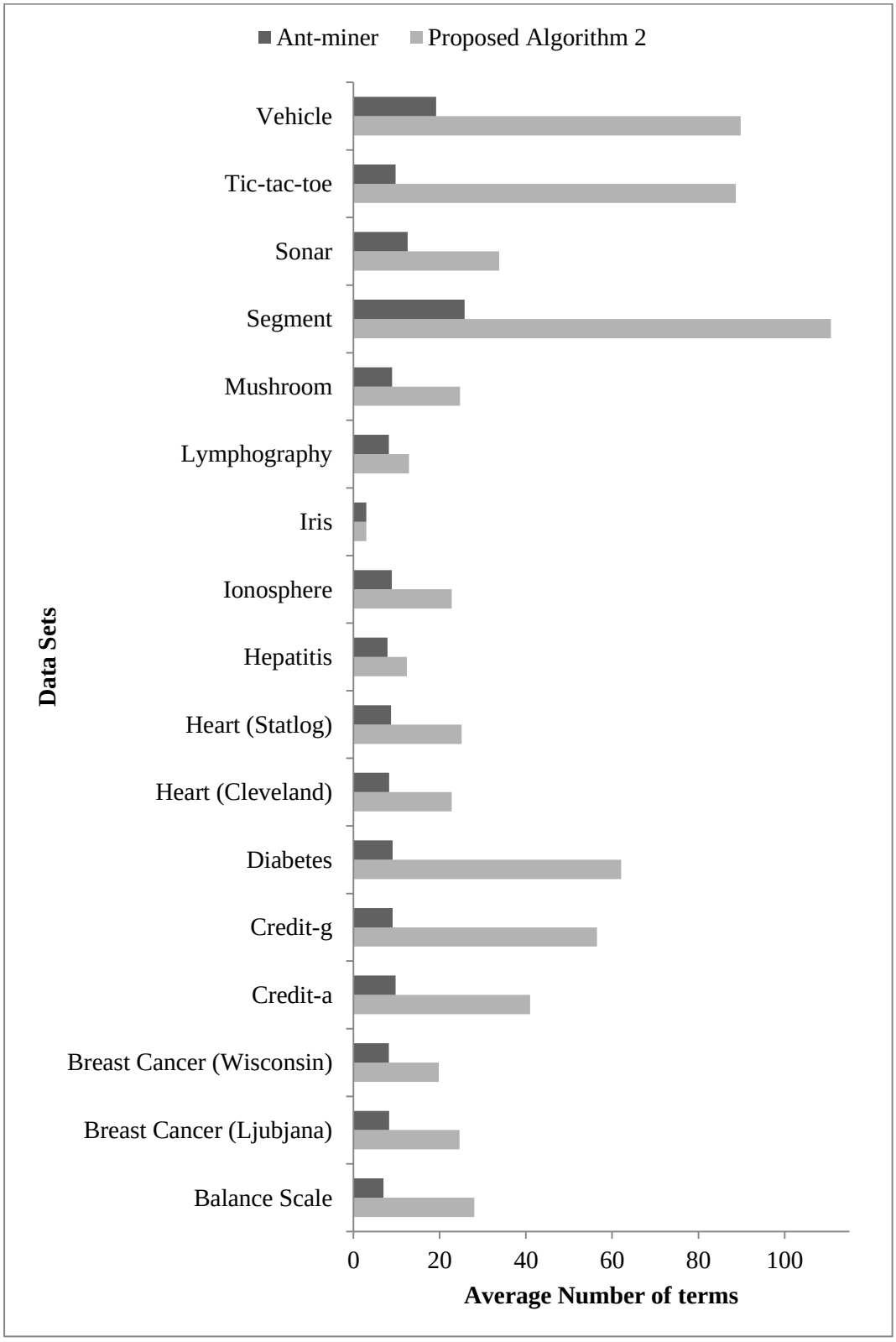

Figure 6.13: Comparison of Average Number of Terms between Ant-miner and Proposed Algorithm 2 on Reduced Attributes Data Sets

The performance of the algorithm proposed in this chapter was also compared against several other well-known rule induction algorithm discussed in Section 2.2 that includes the conjunctive rule (Witten et al., 2011), decision table (Kohavi, 1995b), DTNB (M. Hall & Frank, 2008), JRip (Cohen, 1995) and PART (Frank & Witten, 1998).

Table 6.7 shown the average predictive accuracy comparison of conjunctive rule (Witten et al., 2011), decision table (Kohavi, 1995b), DTNB (M. Hall & Frank, 2008), JRip (Cohen, 1995) and PART (Frank & Witten, 1998) against the proposed algorithm in this chapter. The mean for the average predictive accuracy for each algorithm were calculated. It is found that the proposed algorithm in this chapter produced the highest mean of average predictive accuracy at 83.83% while the conjunctive rule algorithm produced the lowest mean at 69.97%, the hybrid of PSO and ACO algorithm for rule induction, the PSO/ACO2 scores a mean of 79.03%.

Table 6.8 depicted the average number of rules comparison between the proposed algorithm and other rule induction algorithms. The table shows that the proposed algorithm produced the highest number of rules as compared to other algorithms with a mean of 19.28 rules. However, the mean is only slightly higher to PART (16.29) and PSO/ACO2 (17.79). However, JRip produced the lowest number of rules with a mean of 8.52.

As a conclusion, the proposed algorithm performs not just much better as compared to the original Ant-miner (Parpinelli et al., 2002a, 2002b), but also, it is competitive with other tested rules induction algorithms. It seems that the high improvement on

predictive accuracy, sacrificed the simplicity of the rules. Hence, the results in this chapter suggested that the algorithm needs a mechanism to control the terms inclusion, without scarifying the predictive accuracy. Perhaps, the algorithm also needs a better pruning strategy in order to improve the simplicity of the rules, for training data while fixing class.

Table 6.7: Average Predictive Accuracy of Conjuctive Rule, Decision Table, DTNB, JRip, PART and Proposed Algorithm 2

Data Sets	Conjunctive Rule	Decision Table	DTNB	JRip	PART	ACO/ PSO2	Prop. Algo. 2
Balance Scale	60.93	71.26	70.89	74.87	69.84	82.72	71.04
	± 5.4	± 3.79	± 3.74	± 4.41	± 3.97	± 4.77	± 3.91
Breast Cancer (Ljubjana)	69.03	73.73	69.94	71.45	69.41	72.62	72.39
	± 4.17	± 5.34	± 6.93	± 6.44	± 7.63	± 6.84	± 9.09
Breast Cancer (Wisconsin)	88.03	94.96	97.01	95.48	95.05	93.42	96.14
	± 3.6	± 2.54	± 1.91	± 2.3	± 2.39	± 3.79	± 2.93
Credit-a	85.51	84.67	85.48	86.38	85.25	85.31	85.80
	± 3.96	± 4.21	± 4.09	± 3.75	± 4.00	± 4.14	± 2.58
Credit-g	70	72.75	71.76	71.74	72.24	67.9	75.50
	± 0	± 3.66	± 3.71	± 3.67	± 4.24	± 5.82	± 3.29
Diabetes	73.47	77.02	77.77	77.41	76.84	72.67	76.70
	± 4.91	± 4.7	± 4.45	± 4.78	± 4.19	± 4.98	± 4.11
Heart (Cleveland)	72.98	77.22	80.81	81.65	77.49	77.38	81.78
	± 7.18	± 7.5	± 8.75	± 6.45	± 8.30	± 5.45	± 7.29
Heart (Statlog)	73	83.41	81.56	82.89	83.33	81.11	81.11
	± 8.09	± 7.03	± 6.55	± 6.67	± 7.44	± 6.16	± 9.14
Hepatitis	79.83	81.22	81.1	81.45	81.59	-	83.25
	± 4.95	± 8.07	± 8.31	± 9.16	± 8.62	-	± 5.79
Ionosphere	80.77	89.04	91.68	91.68	90.03	88.06	90.89
	± 6.12	± 4.58	± 4.54	± 4.84	± 4.43	± 4.91	± 3.98
Iris	66.67	93.8	94.13	94.6	94.87	94.67	93.33
	± 0	± 5.04	± 5.38	± 5.25	± 5.09	± 5.26	± 8.43
Lymphography	71.77	73.47	77.25	77.4	78.24	83.05	78.29
	± 11.86	± 10.5	± 10.22	± 11.08	± 10.54	± 6.67	± 6.88
Mushroom	88.68	100	99.91	100	100	99.9	99.01
	± 1.11	± 0	± 0.11	± 0	± 0.00	± 0.11	± 2.55
Segment	28.57	90.45	94.57	92.84	94.82	96.67	92.42

	± 0	± 2.06	± 1.5	± 2.28	± 1.53	± 1.17	± 1.60
Sonar	70.33	74.77	79.49	79	81.26	75.05	80.36
	± 9.94	± 11.14	± 8.92	± 8.34	± 8.51	± 9.11	± 7.40
Tic-tac-toe	69.45	73.83	69.94	97.55	93.85	100	97.18
	± 4.31	± 4.13	± 4.31	± 1.66	± 3.08	± 0	± 1.88
Vehicle	40.4	67.34	66.3	67.32	70.45	73.05	69.98
	± 2.09	± 4.36	± 3.63	± 3.94	± 4.24	± 4.45	± 4.04

Table 6.8: Average Number of Rules of JRip, PART, PSO/ACO2 and Proposed Algorithm 2

Data Sets	JRip	PART	PSO/ACO2	Proposed Algorithm 2
Balance Scale	4.67 ± 0.47	4.92 ± 0.27	26.6 ± 1.07	16.3 ± 0.78
Breast Cancer (Ljubjana)	2.87 ± 0.82	16.64 ± 3.63	12.4 ± 2.27	14 ± 1.55
Breast Cancer (Wisconsin)	6.79 ± 1	9.77 ± 1.53	9.9 ± 1.6	11.8 ± 0.75
Credit-a	4.89 ± 1.47	16.49 ± 4.27	22.7 ± 2	18 ± 2.32
Credit-g	5.08 ± 1.52	14.16 ± 3.25	54.3 ± 1.89	29.5 ± 2.2
Diabetes	4.71 ± 0.56	13.15 ± 2.32	33.4 ± 1.43	27.2 ± 1.66
Heart (Cleveland)	4.22 ± 0.66	10.43 ± 2.33	12.6 ± 0.84	12 ± 0.77
Heart (Statlog)	4.88 ± 0.82	9.36 ± 2.04	9.7 ± 1.34	11.8 ± 0.75
Hepatitis	2.37 ± 0.51	6.59 ± 3.71	-	7.5 ± 0.67
Ionosphere	8.12 ± 1.65	9.84 ± 1.56	3.6 ± 0.97	12.5 ± 0.92
Iris	3.1 ± 0.3	3 ± 0	3 ± 0	4 ± 0
Lymphography	6.73 ± 0.97	9.65 ± 2.25	14.7 ± 2	7.6 ± 0.8
Mushroom	8.84 ± 0.6	13.19 ± 1.34	8.7 ± 0.48	16.6 ± 0.66
Segment	46.55 ± 8.44	55 ± 4.67	21.9 ± 0.99	56 ± 1.61
Sonar	4.62 ± 0.83	12.85 ± 1.89	4.4 ± 1.58	12.4 ± 0.8
Tic-tac-toe	10.33 ± 1.42	33.09 ± 3.37	9 ± 0	34.5 ± 2.5
Vehicle	16.04 ± 2.53	38.83 ± 3.38	37.8 ± 1.2	36.1 ± 2.43

6.2.2 Classification of Web Data Set

This subsection experiments the proposed algorithm with the data set built in Chapter Four. The number of attributes was reduced using Correlation-based evaluation method with Random search as the search method. The results was

compared to PART (Witten et al., 2011) and Ant-miner (Parpinelli et al., 2002a, 2002b).

Table 6.9 and Figure 6.14 show that the average predictive accuracy for proposed algorithm is at par with PART and Ant-miner. However, consistent with the results in Section 6.2.1, the proposed algorithm discovered less comprehend rule as compared to Ant-miner, but at par with PART. Even though Ant-miner produced much simpler rules than the proposed algorithm, the proposed algorithm is more accurate than Ant-miner. It shows in order to improve the accuracy; the simplicity of the rules must be sacrifice.

Table 6.9: Performance Comparison for Reduced Web Data

Algorithm	**Average Predictive Accuracy (%)**	**Average Number of Rules**	**Average Number of Terms**
PART	94.01 ± 4.30	28.00 ± 3.4	90.00 ± 3.5
Ant-miner	93.36 ± 0.67	7.00 ± 0.15	10.1 ± .45
ProposedAlgorithm 2	93.89 ± 1.38	26.30 ± 2.05	79.4 ± 8.91

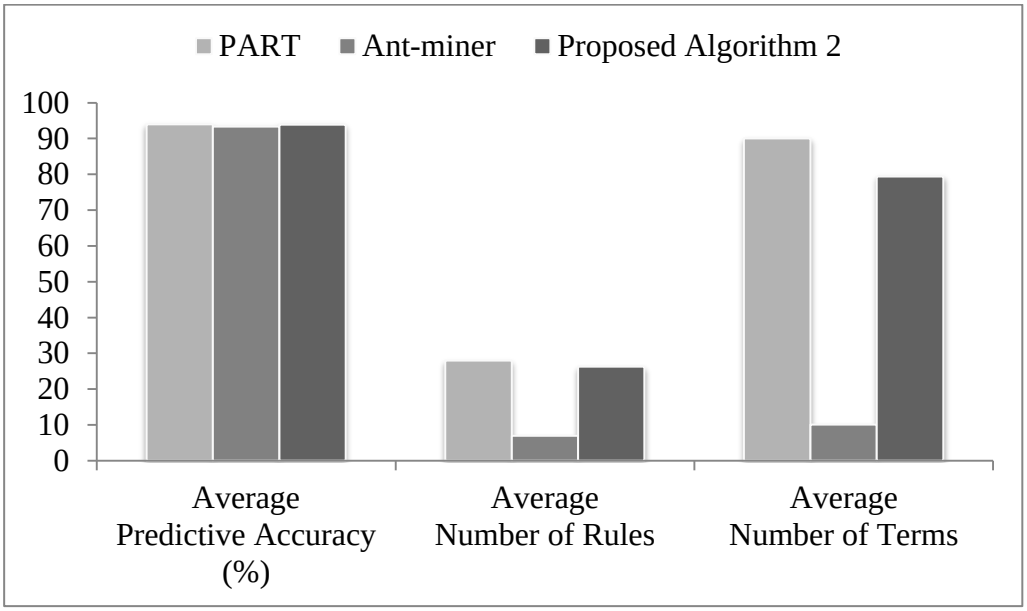

Figure 6.14: Performance Comparison for Reduced Web Data

6.3 Summary

This chapter proposed an algorithm based on a sequential covering algorithm, which uses ACO as the Learn-One-Rule function to extract rules from data sets. In order to improve the rule's quality, the proposed algorithm uses SA for terms selection while building the rule. The proposed algorithm used a simpler heuristic that does not take the relationship between terms into account, while selecting terms for growing a rule. By fixing the class, the proposed algorithm is able to use a much simpler heuristic function, since the algorithm fixed the class before selecting terms for inclusion in a rule. Moreover, this chapter also proposed a simpler fitness function to evaluate the rule's quality.

The performance of the proposed algorithm was compared to the original Ant-miner (Parpinelli et al., 2002a, 2002b) as well as the PART concerning the predictive accuracy, and found to be competitive with PART, but much better than the original

Ant-miner (Parpinelli et al., 2002a, 2002b). Therefore, SA is able to reduce the local optimum problem, where the ant found a more related term for inclusion into the partial rule.

In future, this thesis suggested an enhancement of the proposed hybrid algorithm to cope with non-discrete attributes on the fly. Furthermore, it would be interesting to evaluate the performance of the proposed hybrid algorithm for selecting terms and fitness function for rule evaluation. Since there is no perfect classification algorithm for all types of data sets; it is also practical to evaluate the performance of the proposed algorithm on other types of data sets, such as sparse and huge data sets.

CHAPTER SEVEN
CONCLUSION AND FUTURE WORK

The main goal of this thesis is to design a new ACO based approach for classification rule induction. To achieve this goal, first, this thesis used SA as a local search to improve the rule's quality by optimizing each rule constructed by an ant. Second, this thesis used SA as terms selection algorithm during the construction a rule. The next section summaries the conclusion drawn from the experiments and results discussed in Chapter Five and Six, while the following section suggests possible directions for future work.

7.1 Research Contribution

There are three main contributions of the research. The first contribution is finding the best attribute selection method for Web texts categorization. The combination of correlation-based evaluation with random search as the search method is the best approach for attribute selection. However, this attribute selection method will not give the best performance in attributes reduction. Using classifier-based attribute subset selection will reduce more attributes, but sacrifice the performance of the classifier. Experiments in Chapter Four found that Ant-miner performed better than C4.5 for Web texts categorization.

The first proposed hybrid algorithm based on ACO and SA to discover classification rules from data is the second main contribution. The proposed algorithm discovered simpler rules, without scarifying the predictive accuracy. Hence, the use of SA was able to produce simpler rules.

The second proposed hybrid algorithm based on ACO, and SA to selecting terms for inclusion in a rule is the third main contribution. The proposed algorithm discovered higher predictive accuracy rules compared to both Ant-miner and PART.

As a conclusion, SA is able to improve the discovered rules by ACO. Experiments in this thesis have shown that SA was able to reduce the local extrema problem of ACO. In other words, SA works successfully to intensify the solutions (set of rules in) discovered by ACO.

7.2 Future Work

In future, it is suggested to test the performance of attribute selection on higher dimension of Web data sets, with more categories, since this study has only focus on two categories. Higher dimension of data sets may cause higher dimension of attributes. On the other hand, a study on reducing the size of attributes dimension could also be done in related to the linguistic relationship to generalize words, as a manual preliminary step before performing the attribute selection method.

The future work also can enhance the proposed hybrid ACO and SA algorithms to cope with non-discrete attributes. It is also interesting to evaluate the performance of the proposed hybrid algorithm using different heuristic function for selecting terms and fitness function for rule evaluation.

Fixing the class before constructing the rule improved the predictive accuracy significantly. However, the improvement sacrifices the rules' simplicity. Hence, it is also suggested that a new mechanism is needed to control the size of the rules set and

the number of terms for each rule. Perhaps a new pruning procedure is needed in order to cope with this kind of rules induction algorithm.

Experiments in this thesis have shown that SA was able to improve the solutions discovered by ACO. However, SA is known for its slow speed for convergence (Nikolaev & Jacobson, 2010; Henderson et al., 2003), and hence may affected the algorithm performance in terms of speed. Therefore, it is suggested that in future, the SA part of the proposed algorithms should be enhanced to improve its speed.

REFERENCES

Abramson, D., Krishnamoorthy, M., & Dang, H. (1999). Simulated Annealing Cooling Schedules For The School Timetabling Problem. *Asia-Pacific Journal of Operational Research*, *16*, 1–22.

Anghinolfi, D., & Paolucci, M. (2008). Simulated Annealing as an Intensification Component in Hybrid Population-based Metaheuristics. In M. T. Cher (Ed.), *Simulated Annealing*. InTech.

Asuncion, A., & Newman, D. J. (2007). *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences. Retrieved from http://www.ics.uci.edu/~mlearn/MLRepository.html

Azizi, N., & Zolfaghari, S. (2004). Adaptive temperature control for simulated annealing: a comparative study. *Computers & Operations Research*, *31*(14), 2439–2451. doi:10.1016/S0305-0548(03)00197-7

Bauer, A., Bullnheimer, B., Hartl, R. F., & Strauss, C. (2000). Minimizing total tardiness on a single machine using ant colony optimization. *Central European Journal of Operations Research.*, *8*(2), 125–141.

Besten, M. den, Stützle, T., & Dorigo, M. (2000). Ant Colony Optimization for the Total Weighted Tardiness Problem. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VI, Sixth International Conference on Parallel Problem Solving from Nature* (Vol. 1917, pp. 611–620). Berlin, Germany: Springer Verlag.

Blum, C. (2002a). ACO applied to Group Shop Scheduling: A case study on Intensification and Diversification. In M. Dorigo, G. D. Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002 – From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms* (Vol. 2463, pp. 14–27). Berlin, Germany: Springer-Verlag.

Blum, C. (2002b). *Ant Colony Optimization for the Edge-Weighted k-Cardinality Tree Problem* (No. TR/IRIDIA/2002-05). Belgium: IRIDIA, Université Libre de Bruxelles, Belgium.

Blum, C., & Sampels, M. (2002). Ant colony optimization for FOP shop scheduling: a case study on different pheromone representations. In *CEC '02: Proceedings of the Evolutionary Computation on 2002. CEC '02. Proceedings of the 2002 Congress* (pp. 1558–1563). Washington, DC, USA: IEEE Computer Society.

Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., & Theraulaz, G. (1998). Routing in Telecommunications Networks with "Smart" Ant-Like Agents. In S. Albayrak & F. Garijo (Eds.), *Second International Workshop on Intelligent Agents for Telecommunications Applications "98 (IATA"98)* (Vol. 1437, pp. 60–72). Berlin, Germany: Springer-Verlag.

Bui, T. N., & Nguyen, T. H. (2006). An agent-based algorithm for generalized graph colorings. In *GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation* (pp. 19–26). New York, NY, USA: ACM Press. doi:http://doi.acm.org/10.1145/1143997.1144001

Cabena, P., Hadjinian, P., Stadler, R., Verhees, J., & Zanasi, A. (1998). *Discovering data mining: from concept to implementation*. Prentice-Hall, Inc.

Caro, G. D., & Dorigo, M. (1997). *AntNet: a mobile agents approach to adaptive routing* (No. IRIDIA/97-12). Belgium: Université Libre de Bruxelles.

Chen, D.-J., Lee, C.-Y., Park, C.-H., & Mendes, P. (2007). Parallelizing simulated annealing algorithms based on high-performance computer. *J. of Global Optimization*, *39*(2), 261–289. doi:10.1007/s10898-007-9138-0

Chen, S., & Luk, B. L. (1999). Adaptive simulated annealing for optimization in signal processing applications. *Signal Processing*, *79*(1), 117–128. doi:10.1016/S0165-1684(99)00084-5

Clarke, E. J., & Barton, B. A. (2000). Entropy and MDL discretization of continuous variables for Bayesian belief networks. *International Journal of Intelligent Systems*, *15*(1), 61–92. doi:http://dx.doi.org/10.1002/(SICI)1098-111X(200001)15:1<61::AID-INT4>3.0.CO;2-O

Cohen, W. W. (1995). Fast Effective Rule Induction. In *In Proceedings of the Twelfth International Conference on Machine Learning* (pp. 115–123). Morgan Kaufmann.

Compton, P., & Jansen, R. (1990). Knowledge in context: a strategy for expert system maintenance. In *Proceedings of the Second Australian Joint Conference on Artificial Intelligence* (pp. 292–306). New York, NY, USA: Springer-Verlag New York, Inc.

Cordón, O., Casillas, J., & Herrera, F. (2000). Learning Fuzzy Rules Using Ant Colony Optimization. In *Proceedings of ANTS '2000 – From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 13–21). Brussels, Belgium.

Craven, M., Dipasquo, D., Freitag, D., Mccallum, A. K., Mitchell, T. M., Nigam, K., & Slattery, S. (1998). Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence* (pp. 509–516). Madison, US: AAAI Press, Menlo Park, US.

Delport, V. (1998). Parallel Simulated Annealing and Evolutionary Selection for Combinatorial Optimisation. *Electronics Letters*, *34*(8), 758 –759. doi:10.1049/el:19980546

Deneubourg, J. L., Aron, S., Goss, S., & Pasteels, J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, *3*(2), 159–168.

Dorigo, M., & Gambardella, L. M. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991). Positive feedback as a search strategy.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *26*(1), 29–41.

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization*. the MIT Press.

Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *ICML-95*.

Emden-Weinert, T., & Proksch, M. (1999). Best Practice Simulated Annealing for the Airline Crew Scheduling Problem. *Journal of Heuristics*, *5*(4), 419–436. doi:10.1023/A:1009632422509

Fayyad, U., & Irani, K. (1993). Multi-interval discretization of continuos attributes as preprocessing for classification learning. In *Proceedings of the 13th International Join Conference on Artificial Intelligence, Morgan Kaufmann Publishers* (pp. 1022–1027).

Fenet, S., & Solnon, C. (2003). Searching for Maximum Cliques with Ant Colony Optimization. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. Cardalda, D. W. Corne, … E. Marchiori (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003* (Vol. 2611, pp. 236–245). Berlin, Germany: Springer-Verlag.

Fielding, M. (2000). Simulated Annealing With An Optimal Fixed Temperature. *SIAM Journal on Optimization*, *11*(2), 289–307. doi:10.1137/S1052623499363955

Frank, E., & Witten, I. H. (1998). Generating Accurate Rule Sets Without Global Optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 144–151). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Freitas, A. A., Parpinelli, R. S., & Lopes, H. S. (2008). Ant colony algorithms for data classification. In M. Khosrow-Pour (Ed.), *Encyclopedia of Information Science and Technology* (2nd ed., pp. 154–159). Information Science Reference.

Gaines, B. R., & Compton, P. (1995). Induction of Ripple-Down Rules Applied to Modeling Large Databases. *J. Intell. Inf. Syst.*, *5*(3), 211–228.

Galea, M., & Shen, Q. (2006). Simultaneous ant colony optimization algorithms for learning linguistic fuzzy rules. *Swarm intelligence in data mining*, 75–99.

Gambardella, L. M., & Dorigo, M. (1997). *HAS-SOP: Hybrid Ant System for the Sequential Ordering Problem* (Technical Report No. IDSIA-11-97). Lugano, Switzerland: IDSIA.

Gambardella, L. M., & Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing*, *12*(3), 237–255.

Ghanbari, A., Hadavandi, E., & Abbasian-Naghneh, S. (2010). An intelligent ACO-SA approach for short term electricity load prediction. In *Proceedings of the Advanced Intelligent Computing Theories and Applications, and 6th International Conference on Intelligent Computing* (pp. 623–633). Berlin, Heidelberg: Springer-Verlag.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Goss, S., Aron, S., Deneubourg, J. L., & Pasteels, J. M. (1989). Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, *76*(12), 579–581.

Hall, M. A. (1999). *Correlation-based feature selection for machine learning*. (Doctoral dissertation, University of Waikato, 1999).

Hall, M. A., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, *11*(1), 10–18.

Hall, M., & Frank, E. (2008). Combining Naive Bayes and Decision Tables. In *Proceedings of the 21st Florida Artificial Intelligence Society Conference (FLAIRS)* (pp. 318–319). AAAI Press.

Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques*. Elsevier Science.

Henderson, D., Jacobson, S., & Johnson, A. (2003). The Theory and Practice of Simulated Annealing. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics* (Vol. 57, pp. 287–319). Springer New York.

Holden, N., & Freitas, A. A. (2008). A hybrid PSO/ACO algorithm for discovering classification rules in data mining. *J. Artif. Evol. App., 2008,* 2:1–2:11. doi:10.1155/2008/316145

Holte, R. C. (1993). Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning, 11*, 63–91.

Hull, D. A. (1996). Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for Information Science*, *47*(1), 70–84.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., & Schevon, C. (1989). Optimization by Simulated Annealing: an Experimental Evaluation. Part I, Graph Partitioning. *Operations research*, *37*(6), 865–892. doi:10.1287/opre.37.6.865

Johnson, D. S., Aragon, C. R., McGeoch, L. A., & Schevon, C. (1991). Optimization by Simulated Annealing: an Experimental Evaluation; Part II, Graph Coloring and Number Partitioning. *Operations research*, 378–406.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *science*, *220*(4598), 671.

Kohavi, R. (1995a). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence* (Vol. 14, pp. 1137–1145).

Kohavi, R. (1995b). The Power of Decision Tables. In *Proceedings of the 8th European Conference on Machine Learning* (pp. 174–189). London, UK, UK: Springer-Verlag.

Koulamas, C., Antony, S., & Jaen, R. (1994). A survey of simulated annealing applications to operations research problems. *Omega*, *22*(1), 41–56. doi:10.1016/0305-0483(94)90006-X

Larose, D. T. (2005). *Discovering Knowledge in Data: An Introduction to Data Mining*. John Wiley and Sons, Inc.

Leguizamón, G., & Michalewicz, Z. (1999). A new version of ant system for subset problems. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC '99.* (pp. 1459–1464). Piscataway, NJ: IEEE Press.

Liang, Y.-C., & Smith, A. E. (1999). An ant system approach to redundancy allocation. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC '99.* (pp. 1478–1484). Piscataway, NJ: IEEE Press.

Liu, B., Abbass, H. A., & McKay, B. (2004). Classification Rule Discovery with Ant Colony Optimization. *The IEEE Computational Intelligence Bulletin*, *3*(1), 31–35.

Liu, H., & Setiono, R. (1996). A probabilistic approach to feature selection-a filter solution. In *Proceedings of the International Conference on Machine Learning* (pp. 319–327).

Lourenço, H. R., & Serra, D. (1998). *Adaptive Approach Heuristics for The Generalized Assignment Problem* (Economics Working Papers No. 288). Plaça de la Mercè, Barcelona: Department of Economics and Business, Universitat Pompeu Fabra.

Mangat, V. (2012). A Novel Hybrid Framework using Evolutionary Computing and Swarm Intelligence for Rule Mining in the medical domain. *IJCA Proceedings on*

International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT 2012), iRAFIT(6), 7–13.

Maniezo, V., Colorni, A., & Dorigo, M. (1994). *The Ant System Applied to The Quadratic Assignment Problem* (No. IRIDIA/94-28). Belgium: Université Libre de Bruxelles.

Maniezzo, V., & Carbonaro, A. (2000). An ANTS heuristic for the frequency assignment problem. *Future Generation Computer Systems*, *16*(8), 927–935.

Maniezzo, V., & Colorni, A. (1999). The Ant System Applied to the Quadratic Assignment Problem. *IEEE Transactions on Knowledge and Data Engineering*, *11*(5), 769–778.

Martens, D., De Backer, M., Haesen, R., Baesens, B., & Holvoet, T. (2006). Ants constructing rule-based classifiers. *Swarm Intelligence in Data Mining*, 21–43.

Merkle, D., Middendorf, M., & Schmeck, H. (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, *6*(4), 333–346.

Moore, A. W., & Lee, M. S. (1994). Efficient algorithms for minimizing cross validation error. In *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 190–198).

Nalini, C., & Balasubramnaie, P. (2010). Performance Analysis of Hybrid Swarm Intelligence Rule Induction Algorithm. *INFOCOMP Journal of Computer Science*, *9*(1), 53–60.

Niknam, T., Bahmani, B. F., & Nayeripour, M. (2008). An Efficient Hybrid Evolutionary Algorithm for Cluster Analysis. *World Applied Sciences Journal*, *4*(2), 300–307.

Niknam, T., Olamaei, J., & Amiri, B. (2008). A Hybrid Evolutionary Algorithm Based on ACO and SA for Cluster Analysis. *Journal of Applied Sciences*, *8*(15), 2695–2702. doi:10.3923/jas.2008.2695.2702

Nikolaev, A. G., & Jacobson, S. H. (2010). Simulated Annealing. In M. Gendreau & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* (Vol. 146, pp. 1–39). Springer US.

Olamaei, J., Arefi, A., Mazinan, A. H., & Niknam, T. (2010). A hybrid evolutionary algorithm based on ACO and SA for distribution feeder reconfiguration. In *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference On* (Vol. 4, pp. 265–269). doi:10.1109/ICCAE.2010.5451699

Olamaei, J., Niknam, T., Arefi, A., & Mazinan, A. H. (2011). A novel hybrid evolutionary algorithm based on ACO and SA for distribution feeder reconfiguration with regard to DGs. In *GCC Conference and Exhibition (GCC), 2011 IEEE* (pp. 259–262). doi:10.1109/IEEEGCC.2011.5752495

Oliverio, V., Sá, C. C. de, & Parpinelli, R. S. (2009). Building a navigational environment for autonomous agents with reinforcement learning. In *IADIS AC (2)* (pp. 353–355).

Orosz, J. E., & Jacobson, S. H. (2002a). Analysis of Static Simulated Annealing Algorithms. *J. Optim. Theory Appl.*, *115*(1), 165–182. doi:10.1023/A:1019633214895

Orosz, J. E., & Jacobson, S. H. (2002b). Finite-Time Performance Analysis of Static Simulated Annealing Algorithms. *Comput. Optim. Appl.*, *21*(1), 21–53. doi:10.1023/A:1013544329096

Otero, F. E. B., Freitas, A. A., & Johnson, C. G. (2012). A New Sequential Covering Strategy for Inducing Classification Rules with Ant Colony Algorithms. *Evolutionary Computation, IEEE Transactions on*, *PP*(99), 1–21. doi:10.1109/TEVC.2012.2185846

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2001). An Ant Colony Based System for Data Mining: Applications to Medical Data. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, … E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (pp. 791–797). San Francisco, California, USA: Morgan Kaufmann.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002a). An Ant Colony Algorithm for Classification Rule Discovery. In H. A. Abbas, R. A. Sarker, & C. S. Newton (Eds.), *Data Mining: A Heuristic Approach* (pp. 191–208). London: Idea Group Publishing.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002b). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation, special issue on Ant Colony Algorithms*, *6*(4), 321–332.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002c). Mining Comprehensible Rules from Data with an Ant Colony Algorithm. In G. Bittencourt & G. L. Ramalho (Eds.), *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence (SBIA-2002)* (pp. 259–269). Springer-Verlag.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2005). Classification-Rule Discovery with an Ant Colony Algorithm. In M. Khosrow-Pour (Ed.), *Encyclopedia of Information Science and Technology* (pp. 420–424). Hershey: Idea Group.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann.

Reimann, M., Doerner, K., & Hartl, R. F. (2002). Insertion Based Ants for Vehicle Routing Problems with Backhauls and Time Windows. In *ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms* (pp. 135–148). London, UK: Springer-Verlag.

Shahzad, W., & Baig, A. R. (2011). Hybrid Associative Classification Algorithm Using Ant Colony Optimization. *International Journal of Innovative Computing, Information and Control*, *7*(12), 6518–6826.

Silva, R. M. de A., & Ramalho, G. L. (2001). Ant system for the set covering problem. In *IEEE International Conference on Systems, Man, and Cybernetics, 2001* (pp. 3129–3133). Tucson, Arizona: IEEE Press.

Smaldon, J., & Freitas, A. A. (2006). A new version of the ant-miner algorithm discovering unordered rule sets. In *GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation* (pp. 43–50). New York, NY, USA: ACM Press. doi:http://doi.acm.org/10.1145/1143997.1144004

Socha, K., Knowles, J., & Sampels, M. (2002). A MAX-MIN Ant System for the University Timetabling Problem. In M. Dorigo, G. D. Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002 – Third International Workshop on Ant Algorithms* (Vol. 2463, pp. 1–13). Berlin, Germany: Springer-Verlag.

Steinhöfel, K., Albrecht, A., & Wong, C. K. (2002). The convergence of stochastic algorithms solving flow shop scheduling. *Theoretical Computer Science*, *285*(1), 101–117. doi:10.1016/S0304-3975(01)00293-6

Stützle, T. (1998). An ant approach to the flow shop problem. In *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing EUFIT'98* (Vol. 3, pp. 1560–1564). Aachen, Germany: Verlag Mainz, Wissenschaftsverlag.

Sullivan, K. A., & Jacobson, S. H. (2000). Ordinal Hill Climbing Algorithms for Discrete ManufacturingProcess Design Optimization Problems. *Discrete Event Dynamic Systems*, *10*(4), 307–324. doi:10.1023/A:1008302003857

Sullivan, K. A., & Jacobson, S. H. (2001). A convergence analysis of generalized hill climbing algorithms. *Automatic Control, IEEE Transactions on*, *46*(8), 1288 –1293. doi:10.1109/9.940936

T'Kindt, V., Monmarche, N., Laugt, D., Tercinet, F., & Portalis, A. J. (2000). Combining Ants Colony Optimization and Simulated Annealing to solve a 2-machine flowshop bicriteria scheduling problem. In *13th Eropean Chapter on Combinatorial Optimization (ECCO XIII)* (pp. 129–130).

Tan, P.-N., Steinbach, M., & Kumar, V. (2006). *Introduction to data mining*. Pearson Addison Wesley Boston.

Teich, T., Fischer, M., Vogel, A., & Fischer, J. (2001). A new Ant Colony Algorithm for the Job Shop Scheduling Problem. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, … E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (p. 803). San Francisco, California, USA: Morgan Kaufmann.

Varanelli, J. M., & Cohoon, J. P. (1999). A fast method for generalized starting temperature determination in homogeneous two-stage simulated annealing systems. *Computers & Operations Research*, *26*(5), 481–503. doi:10.1016/S0305-0548(98)00062-8

Varela, G. N., & Sinclair, M. C. (1999). Ant Colony Optimisation for Virtual-Wavelength-Path Routing and Wavelength Allocation. In Peter J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the Congress on Evolutionary Computation* (Vol. 3, pp. 1809–1816). Mayflower Hotel, Washington D.C., USA: IEEE Press.

Wang, J., Gao, X., & Zhu, Y. (2011). Solving algorithm for TA optimization model based on ACO-SA. *Systems Engineering and Electronics, Journal of*, *22*(4), 628 –639. doi:10.3969/j.issn.1004-4132.2011.04.012

Wang, Z., & Feng, B. (2005). Classification Rule Mining with an Improved Ant Colony Algorithm. In G. Webb & X. Yu (Eds.), *AI 2004: Advances in Artificial Intelligence* (Vol. 3339, pp. 177–203). Springer Berlin / Heidelberg.

Witten, I. H., Frank, E., & Mark A., H. (2011). *Data Mining: Practical Machine Learning Tools and Techniques* (3rd ed.). Morgan Kaufmann Pub.

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997)* (pp. 412–420). Nashville, Tennessee, USA: Morgan Kaufmann Publishers.

Appendix A
List of Stop Words

This list of stop words was compiled by The Information Retrieval Group, University of Glasgow, led by Professor Keith van Rijsbergen. (http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words).

a about above across after afterwards again against all almost alone along already also although always am among amongst amoungst amount an and another any anyhow anyone anything anyway anywhere are around as at back be became because become becomes becoming been before beforehand behind being below beside besides between beyond bill both bottom but by call can cannot cant co computer con could couldnt cry de describe detail do done down due during each eg eight either eleven else elsewhere empty enough etc even ever every everyone everything everywhere except few fifteen fify fill find fire first five for former formerly forty found four from front full further get give go had has hasnt have he hence her here hereafter hereby herein hereupon hers herself him himself his how however hundred i ie if in inc indeed interest into is it its itself keep last latter latterly least less ltd made many may me meanwhile might mill mine more moreover most mostly move much must my myself name namely neither never nevertheless next nine no nobody none noone nor not nothing now nowhere of off often on once one only onto or other others otherwise our ours ourselves out over own part per perhaps please put rather re same see seem seemed seeming seems serious several she should show side since sincere six sixty so some somehow someone something sometime sometimes

somewhere still such system take ten than that the their them themselves then thence there thereafter thereby therefore therein thereupon these they thick thin third this those though three through throughout thru thus to together too top toward towards twelve twenty two un under until up upon us very via was we well were what whatever when whence whenever where whereafter whereas whereby wherein whereupon wherever whether which while whither who whoever whole whom whose why will with within without would yet you your yours yourself yourselves

Appendix B

Bash Script for Creating Train/Test Sets

This bash script will create the train/test sets for a 10-fold stratified cross-validation at the same location as the original dataset.

```
#!/bin/bash
## expects the weka.jar as first parameter and the datasets to work on as  second parameter.
#
# FracPete, 2007-04-10
# http://weka.wikispaces.com/Generating+cross-validation+folds+%28Filter+approach%29

DATASET=$1
FOLDS=10
FILTER=weka.filters.supervised.instance.StratifiedRemoveFolds
SEED=1

mkdir `echo $DATASET | sed s/"\.arff"//g`

for ((i = 1; i <= $FOLDS; i++))
do
  echo "Generating pair $i/$FOLDS..."

  OUTFILE=`echo $DATASET | sed s/"\.arff"//g`

  # train set
  java $FILTER -V -N $FOLDS -F $i -S $SEED -i $DATASET -o "$OUTFILE/train$i.arff"
      -c last
  # test set
  java $FILTER    -N $FOLDS -F $i -S $SEED -i $DATASET -o "$OUTFILE/test$i.arff"  -c
      last
done
```

Appendix C

Web Classification Sample Data Set

```
@relation 'student_course'
@attribute assignments {0,1}
@attribute instructor {0,1}
@attribute syllabus {0,1}
@attribute class {0,1}
@attribute hours {0,1}
@attribute homework {0,1}
@attribute lecture {0,1}
@attribute final {0,1}
@attribute exam {0,1}
@attribute interests {0,1}
@attribute notes {0,1}
@attribute university {0,1}
@attribute grading {0,1}
@attribute introduction {0,1}
@attribute research {0,1}
@attribute midterm {0,1}
@attribute assignment {0,1}
@attribute handouts {0,1}
@attribute description {0,1}
@attribute fall {0,1}
@attribute 'Copy of class' {student,course}

@data
0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,course
0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,student
0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,student
0,0,0,1,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,course
0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,student
0,0,0,0,0,0,0,0,0,1,0,1,0,0,1,0,0,0,0,0,student
0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,student
0,0,0,0,0,0,0,0,0,1,0,1,0,0,1,0,0,0,0,0,student
0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,student
1,0,1,1,0,0,1,0,0,0,1,0,1,1,0,1,1,1,1,1,course
0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,course
0,0,0,0,0,0,0,0,0,1,0,1,0,0,1,0,0,0,0,0,student
0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,1,1,course
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,student
1,1,0,1,1,1,1,1,1,0,0,1,1,1,0,1,1,0,0,0,course
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,student
0,1,0,1,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,1,course
0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,student
1,1,1,0,1,1,1,0,0,0,1,0,0,1,0,0,0,0,0,1,course
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,student
```