Implementation and Comparison of Inductive Learning Algorithms on Timetabling

Azlinah Hj. Mohamed, and Mohd Haris Shukri Bin Jahabar

Faculty of Information Technology and Quantitative Sciences, Universiti Teknologi MARA, 40000 Shah Alam, Selangor

lina67@streamyx.com

Abstract

Academic timetabling has been known as a complex task due to various rules that needs to be satisfied. In order to build an effective and efficient timetabling system, these rules have to be identified and recognized. This paper focuses on the implementation and comparison of inductive learning algorithms on timetabling for the purpose to see the ability of several inductive learning algorithms towards solving complex problems, the ability of the algorithms to adapt to the new environment by changing rules and to compare the performance between the algorithms. The algorithms used for this study are ID3 algorithm, AQ algorithm and ILA algorithm. The study shows that all three inductive learning algorithms were successful in the implementation and able to adapt to new environment. ILA algorithm is shown to have better performance in the number of rules generated and percentage of accuracy compared to ID3 algorithm and AQ algorithm.

Keyword: Inductive Learning, Academic Timetabling, ID3 algorithm, AQ algorithm and ILA algorithm.

I. Introduction

Altintas [2] stated that inductive learning is a process of acquiring knowledge by drawing inductive inference from a teacher or environment-provided facts. This process involves operations of generalizing, transforming, correcting and refining knowledge representations in order to accommodate given facts and satisfy various additional criteria.

Inductive learning, in simple words can be defined as learning by generalizing rules and knowledge from observations and prior knowledge. Generalization can be seen as a method of creating a new knowledge from existing knowledge, where the new knowledge is simpler yet still represents the prior knowledge. Inductive learning provides the ability to identify similarities and patterns in the prior knowledge or training data, and then extract them as generalized rules. The generalized rules that are extracted and identified will then be used for the reasoning and solving of the problems. Various algorithms have been proposed in finding the optimal implementation of inductive learning. These algorithms have their own advantages and weaknesses in providing the generalization and induction techniques. Therefore the purpose of this study is to implement several inductive learning algorithms in timetabling problem and to analyze and compare the effectiveness of the algorithms in solving the conventional programming limitations.

The timetabling problem has been selected as the domain of this study due to its complexities and dependencies on various rules and constraints. In building an optimal timetable system, a lot of rules, knowledge, information and constraints are needed. Failure to identify these rules and knowledge may cause the system to have an inefficient and ineffective solution. And these knowledge and rules are also constantly changing according to the adaptation of the environment. The ability of inductive learning to learn and generalize rules would be fully used in timetabling so that the system would have its own ability to analyze the data and gather the information needed to function properly. The complexities in timetabling would also prove that the inductive learning could be implemented in the normal system and everyday problems in various fields.

The main objective is to study the possibility of inductive learning algorithms to be implemented in timetabling. The study of the implementation would be based on the ability of the inductive learning to extract and generalize rules and constraints from the domain and not on the providing of solution (which is generating the timetable).

The second objective is to study whether the timetabling system that uses inductive learning algorithms can adapt into new environment or situation, or in other words, can adapt into the changes of rules and constraints. Meanwhile, the final objective is to compare the performance of the algorithms and to see which algorithm performs better in timetabling domain. The three identified inductive learning algorithms are Itemized Dichotomozer 3 (ID3), Algorithm Quasi-Optimal (AQ) and Inductive Learning Algorithm (ILA).

Meanwhile, the training data used would be generated by the information gained in the domain of Faculty of Information Technology and Quantitative Science (FTMSK), Universiti Teknologi MARA (UiTM) Shah Alam.

In the next section, the definition and history of inductive learning will be discussed. The inductive learning algorithms that are used in this study, which is ID3 algorithm, AQ algorithm and ILA algorithm would also be discussed in great detail. There would also be discussion on past projects of providing solution on timetabling and the strengths and weaknesses of the methods.

II. Background

Inductive learning started around 1960’s, when Banerji used predicate logic as a description language (Samut, 1993). The predicate logic would then be the foundation in inductive learning and inductive logic programming. Meanwhile in 1969, Brian Cohen suggested that it would be possible to create effective descriptions by learning domain knowledge, which brought to the creation of CONFUCIUS. CONFICIUS was the first program that learned description in first-order logic and be able to reuse the learned concept (Samut, 1993). While in 1975, Steven Vere had developed formal induction algorithms for expressions in predicate calculus. He stated that if the concept being learned is conjunctive, therefore it is sufficient to find the intersection of all products to produce the generalization [10].

A. *ID3 Algorithm*

According to Gestwicki [6], Itemized Dichotomozer 3 algorithm, or better known as ID3 algorithm was first introduced by J.R Quinlan in the late 1970's. The algorithm 'learned' from relatively small training set of data to organize and process very large data sets [6]. Ballard [3] stated that ID3 algorithm is a greedy algorithm that selects the next attributes based on the information gain associated with the attributes. The information gain is measured by entropy, where Claude Shannon first introduced the idea in 1948 [3].

ID3 algorithm prefers that the generated tree is shorter and the attributes with lower entropies are put near the top of the tree [3]. These techniques satisfy the idea of Occam's Razor. Occam's Razor stated that, "one should not increase, beyond what is necessary, the number of entities required to explain anything", which means that one should not make more assumptions than minimum needed [7]. Hild [8] described the basic technique on the implementation of ID3 algorithm and it is as shown in figure 1 below.

1. For each uncategorized attribute, its entropy would be calculated with respect to the categorized attribute, or conclusion.
2. The attribute with lowest entropy would be selected.
3. The data would be divided into sets according to the attribute's value. For example, if the attribute 'Size' was chosen, and the values for 'Size' were 'big', 'medium' and 'small, therefore three sets would be created, divided by these values.
4. A tree with branches that represent the sets would be constructed. For the above example, three branches would be created where first branch would be 'big', second branch would be 'medium' and third branch would be 'small'.
5. Step 1 would be repeated for each branch, but the already selected attribute would be removed and the data used was only the data that exists in the sets.
6. The process stopped when there were no more attribute to be considered or the data in the set had the same conclusion, for example, all data had the 'Result' = yes.

Fig. 1. ID3 algorithm

ID3 algorithm had been used and implemented in many fields. One of the earliest implementation of ID3 algorithm is on a chess game. Ivan Bratko, the artificial intelligence researcher was the one implemented this chess game [6]. According to Gestwicki [6], Bratko supplied the ID3 program with several pages of textbook recommendations for playing the chess endgame of white king and rook versus black king and knight. He made the rules around the idea of 'knight's side lost in at most *n* moves'. The result shows that ID3 algorithm is efficient in both time and space considerations, where the feature vector of the games and the decision tree size is small, compared to the training instances.

In a study by Gestwicki [6], one experiment had been conducted to predict the greyhound race. The experiment was to compare between the net profit gained by the ID3 algorithm and by three greyhound-racing experts. In this experiment, the system had been trained with 200 training races and 1600 dogs. The result shows that there are 26 races that the ID3 did not make any bet. This showed that the system was restricted from making any illogical choices, which is unlike human that were to gamble without logic in order to gain more winning.

B. *AQ Algorithm*

Ryszard Michalski was the creator of Algorithm Quasi-Optimal, or better known as AQ or A^q algorithm [9]. It was originally created in 1969 [9]. Mihai [9] stated that AQ algorithm was designed to solve general covering problems of high complexity. In other words, the algorithm was designed to generate generalization or induction from very complex problems.

Clark [5] stated that AQ algorithm output a set of ‘if…then…’ rules. This is different compared to ID3 algorithm, which output decision tree to represent the rules. AQ algorithm used ‘separate and conquer’ approach, where the data would be separated and general rules would be created from the separation. According to Mihai [9], the central concept of the algorithm is the ‘STAR’ defined as a set of general descriptions of a particular event (a ‘seed’) that satisfies given constraints. In finding the rules, the AQ algorithm used the ‘beam search’ method to explore the data. The ‘beam search’ method is actually a breadth first search technique with a heuristic function. Therefore, the progression would not go into ‘blind’ method, but instead, on every node, there would be a consideration whether or not to progress further [5]. Sammut [10] described the AQ algorithm as in Figure 2 below.

1. The data set would be divided into two parts, according to the conclusion. These parts are known as positive data set and negative data set.
2. One data would be selected randomly from the positive data set. This data would then be extended against the negative data set by using the STAR method as described above.
3. All the positive data that satisfy the STAR would be removed and one of the remaining positive data would be selected. The STAR method would be applied again.
4. The process stopped when there are no more data in the positive data set.

Fig. 2. AQ algorithm

C. *ILA Algorithm*

Inductive Leaning Algorithm, or in short, known as ILA algorithm was invented by Mehmed R. Tolun and Saleh M. Abu-Soud [11]. The reason behind this creation was to suggest a better inductive learning algorithm compared to current algorithms at that time such as ID3 algorithm, OC1 algorithm and AQ algorithm [11]. ILA algorithm’s main idea was the iteration of data splitting and rules detection. According to Tolun and Soud [11], the algorithm works in an iterative fashion, whereby each iteration searches for a rule that covers a large number of examples in a single class [11].

ILA algorithm requires that the data be constructed in the form of a number of attributes and a class attribute, which holds the decision value. The data should be represented in the table format, where the rows contain the example data and the columns represent the attributes. The ILA algorithm is as shown in figure 3 below.

1. The data would be divided into sub-tables, according to class attribute value.
2. ILA algorithm would process each sub-table.
3. When all the sub-table had been processed and analyzed than the process stopped.

Fig. 3. ILA algorithm

Several experiments had been done to compare ILA algorithm with ID3 algorithm and AQ algorithm [11]. One of the experiments was to classify objects. The result of the experiment is shown in table 1 below.

Algorithm	Rule No	Rule
ID3	1	If color=green and shape=pillar then yes
ILA		If color=green then yes
ID3	2	If shape=brick then yes
ILA		If size=medium then yes
ID3	3	If shape=sphere then yes
ILA		If shape=sphere then yes
ID3	4	If shape=wedge then no
ILA		If shape=wedge then no
ID3	5	If color=red and shape=pillar then no
ILA		If size=large and color=red then no

Table 1. Result of Classifying Objects' Experiment

In this example, even though both algorithms gathered the same amount of rules, but it was shown that the rules generated by ILA algorithm for rules no 1 was simpler.

Tolun and Soud did another experiment on three different training sets named Ballons, Balance and Tic-tac-toe [11]. The training sets were obtained from the University of California Irvine Repository of Machine Learning Databases and Domain Theories. The result of the experiment is shown in the table 2 below.

Training Set	Algorithm	No of Rules
Balloons	ID3	3
	AQ	3
	ILA	3
Balance	ID3	401
	AQ	312
	ILA	303
Tic-tac-toe	ID3	218
	AQ	86
	ILA	32

Table 2. Result of Experiments on Three Training Sets

It was shown from the experiment that ILA algorithm could produce less amount of rules compared to ID3 algorithm and AQ algorithm. Therefore Tolun and Soud [11] state that ILA algorithm was capable to detect and classify more unseen rules compared to the other two.

D. *Several Timetabling Methodologies*

Caldeira [4] had approach the timetabling problem in his study by using genetic search. He used the genetic algorithm approach with the chromosome built up from classes and teachers pair. Caldera implemented a cost function, a function that acts as a constraint to avoid a probability of same class to be repeated in a day. It also helped in avoiding of more than seven lessons a day. The cost function acted as one of the prime value in the fitness function evaluation.
In breeding the new chromosome, there are four approaches that had been used by Caldeira [4], which is:

i) Reproduction – the exact copy of the parent chromosome.
ii) Crossover – the technique of exchanging genes between a pair of parent chromosomes.
iii) Mutation – the random generated changes of gene values.
iv) Repair *function – the process of repairing chromosomes so that the gene values were valid values.*

It can be seen from Caldera's research that bigger population would lead to wider search and better results, higher number of best parents would help in narrowing the search and smaller mutation probability would avoid the possibility of bad mutation. Caldera's research showed that timetabling can be achieved by using genetic algorithm, but it still lacked the ability to adapt to any changes in the environment, which in this case, the constraints. A fixed implementation of constraints and rules had been done to the system, and therefore the system did not have the ability of learning.
Abramson [1] had introduced the idea of parallel genetic algorithm in his study, to solve the problem of time efficiency during the process of scheduling. He suggested that with a capability of commercial shared memory multiprocessor, the genetic algorithm approach could be done in parallel and thus allowing for a faster execution time. Abramson [1] approached the problem not from the parent chromosome side, but instead from the children chromosome side. For each

child chromosome's space in new population, random parents were selected for the generation of new chromosome. This process was then spawned so that the parallel child chromosomes' generation could be established.

The study by Abramson [1] was successful in showing that parallel genetic algorithm did has a time efficiency performance. The study would help in reducing processing time for a highly constrained condition of timetabling.

All of these studies showed a similarity, which was the implementation of fixed rules and constraints. This would lead to low reusability value of the systems if the environment has the tendency of constantly changes the rules and constraints. Therefore, it is hoped that inductive learning would provide a better solution for this problem.

III. Scope of the Research

A. *Training Data*

The source of data used was manually generated, which was based on the domain of FTMSK and tried to cover as many real world situations as possible. The data is generated based on the actual timetables that had been used and applied in FTMSK. The data was designed to be as close as possible with the real data. For example, the rooms and subjects are based on the real rooms and subjects.

There are also many additions to the sample data, which is primarily designed to complicate the system in identifying and generalizing the rules. For example, several additional factors were added into the data, such as classes that were far in geographical distance but put in near time sequence were given negative feedback value.

The data used for the training contains several attributes that define the data and one attribute that define the value of the data, whether it is true or not. The attributes for the data are:

i) Subject Code – contains the subject code
ii) Program Code – contains the program code
iii) Part – contains the semester's information of the students taking the subject
iv) Day – contains the day of the class
v) Start – contains the start time for the class
vi) End – contains the end time for the class
vii)Room – contains the room number where the class resides
ix) Lecturer – contains the name of the lecturer that teaches the class

In defining the value of the data, only two values would be used that is 'yes' and 'no'. The 'yes' value means that data is acceptable and logical, while the 'no' value means the data is not acceptable and illogical. These values would be the base for the inductive learning algorithms to derive a hypothesis.

The data used for the training of the system had been designed in two formats. The first format is a single data type and the second format is pair data type. Single data type is use to keep the exact information of the data. For example: the Subject attribute in the single data type would keep the value of subject code such as ITC100, ITC200 and ITC 150, please refer to appendix A. This data type will be used to analyze the rules that did not require relations with other data.

The pair data type was constructed to analyze two data and comparing the attributes between the data. This data type is very important since the algorithms can analyze the effect of having the same or different values on the result value (the value that defines the data). In other words the relation of two data can be analyzed. For example, maybe the value ‘Same’ on attribute ‘Day’, which means that two slot of the same classes resides on the same day, would have the value ‘no’. This means that these classes would not be allowed to be on the same day, refer to appendix B.

The value of ‘Same’, ‘Near’, ‘Far’ ‘Diff’ would be given to the data according to the comparison result. The values ‘Same’ mean that the data has the same value for the same attribute. For example, if both values of attribute ‘Subject Code’ on both data would be ITC150, therefore a new pair data would have the ‘Same’ value on the subject code. The value ‘Near’ and ‘Far’, means near and far respectively, would only be used on the ‘Start’, ‘End’ and ‘Room’ attributes. These values represent how far apart the value between the compared data. The value ‘Diff’, which means different, would be given to the result of comparison that is different.

These two data types could not be analyzed together, because of the difference in their value type. Therefore the system would analyze the data type one after another. Sample of these data is as shown in appendix A and appendix B, respectively.

Once the system is accurate, three sets of training data, each consists of 50, 100 and 150 data were feed into the system. The training sets were fed triple times in order to make sure that the results generated was consistent.

The test data set, which is 100 data with no result value, was feed into the system. This is to analyze the accuracy of the rules generated, and to see whether the rules covered all the conditions of the test data. The result of the test data were check manually and the percentage is calculated by the equation below:

% accuracy = (test data correct / total test data) x 100

This process was repeated with another three sets of training data, with the same amount of data. The second data sets were feed into the system to see its ability in learning new rules and constraints.

B. *Selection of Inductive Learning Algorithms*

The types of inductive learning algorithms that were selected to be implemented and studied are crucial, since they will affect the outcomes and the findings of this study. Therefore, algorithms that were selected must specify certain criteria, which are described below:

i) The algorithm should concentrate on implementing the inductive learning algorithm, and there should be no additional features, which can affect the result.

ii) All the algorithms selected would not have the same methods or techniques between each other in the induction process.

iii) The algorithms would not deal with more complicated inductive learning techniques such as noise elimination and bias, due to the data given is considered accurate and exact.

The three algorithms selected are ID3 algorithm, AQ algorithm and ILA algorithm. ID3 algorithm basically used a decision tree method in its induction method, while AQ algorithm used the STAR and beam search approach, and ILA algorithm used classification and attribute combination techniques as elaborated in Section 2 above.

AQ algorithm had to be executed twice since it only deals with one result value (either ‘yes’ or ‘no’) in one execution (as shown in figure 2 above). Therefore in order to make sure that it deals with all of the training data, the algorithm would be executed once for each value.

All algorithms were rewritten in pseudo-code and were redesigned so that it could be implemented in the object oriented programming environment. This was due to the recursive requirement from the algorithms and the data format, which consists of attribute-value pair. Initially, the algorithms that had been successfully programmed would be tested with several small test training data to see its accurateness. The test training data was also calculated manually, and the results were compared with the system’s result. Any differences or problems were analyzed to make sure that the rules generated by the algorithms were accurate.

IV. Findings

The findings and analysis for this study were divided into two main parts; the first part was to see whether inductive learning could be implemented into the timetabling problem, and the second part was to compare the performance of the algorithms, which would be better in solving the problem of timetabling.

The analysis and findings of the implementation of inductive learning on timetabling problem would be based on whether the algorithms used could generalize the rules and constraints in the training data sets. The generalized rules and constraints found would also be studied on the accuracy when tested on the test data. In other words, if the accuracy of the algorithms were too low when tested on the test data, then the algorithms would not be suitable for timetabling problem. This is because low accuracy means that the algorithms would give inaccurate and false result when implemented and therefore the implementation would be considered not successful.

The ability of the inductive learning algorithms in adaptability towards new environment would also be analyzed. This is to see whether the algorithms were able to adapt to new environment and learn new rules and constraints. This is important since the success in learning new rules and constraints would allow the system to have better reusability value when there were changes in the environment.

Finally, the comparison of the three algorithms mentioned before which is ID3 algorithm, AQ algorithm and ILA algorithm, the measurement would be done on the amount of rules and constraints detected by the algorithms. Fewer rules would be considered better, and the algorithm would be considered more generalized, as long as the rules covered all of the training data. The accuracy value would also be compared between the algorithms to see which algorithm have better accuracy given the same training data sets.

A. *Implementation of the Algorithms on Timetabling*

All the algorithms were given three different training data sets, each consisting of 50, 100 and 150 data. After the rules and constraints had been generalized and generated, it would then be compared to the test data. The result for the number of rules generated and the accuracy is shown in table 3 below.

Data Set	Algorithm	Rules Generated	Accuracy
50	ID3	22	92%
	AQ	16	95%
	ILA	13	98%
100	ID3	34	98%
	AQ	26	99%
	ILA	20	100%
150	ID3	45	100%
	AQ	34	100%
	ILA	26	100%

Table 3. Result of the Implementation

From the table, it is shown that all three algorithms were successful in generalizing rules from the training data sets, even though the numbers of rules were different (which will be discussed in Section 4.1.1). This means that given the data representing the situation in the timetabling domain, the algorithms would be able to identify and generalize the rules and constraints.

Analysis on the Rules Generated. The result of rules generated by the experiment above is shown in figure 4 below.

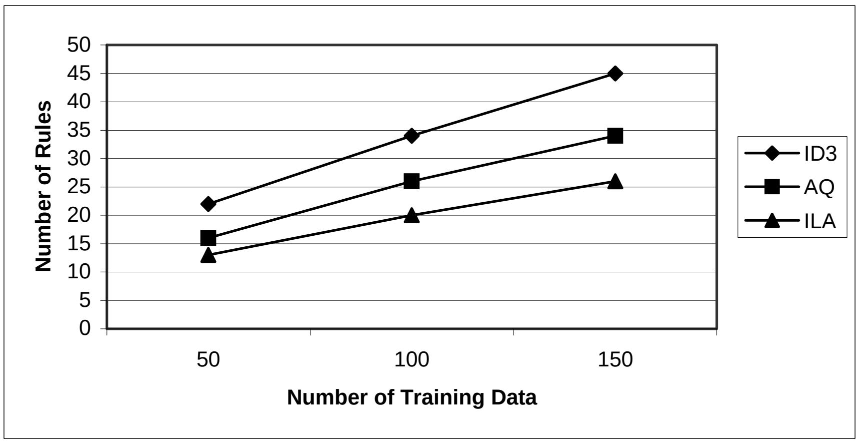

Fig. 4. Rules Generated From the Training

It is shown that the amount of rules generated by the algorithms was directly related with the amount of data given to the algorithms. In other words, the bigger the amount of training data, then the bigger the amount of rules generated. For example, ID3 generated 22 rules for the 50 training data, 34 rules for the 100 training data and 45 rules for the 150 training data.

Analysis on the Accuracy of Result. The accuracy of the rules when compared to the test data was different among three sets. This is shown in figure 5 below.

From the chart, it is shown that the accuracy of the algorithms is directly related to the number of training data trained to the system, where the more amount of training data given to system, the more accurate the system would be. And at a certain point (as in the graph, when training data is 150), all the algorithms would have the same 100% accuracy.

This result shows that the inductive learning algorithm could be implemented successfully on the timetabling problem. It could be seen that all the algorithms gave more than 90% accuracy on the timetable domain, which is quite good. Since the accuracy of the system depends on how much training data were feed into the system, therefore, in order to have a more accurate system, a vast amount of training data should be fed into the system.

This is a slight disadvantage compared to the conventional programming style, where a fixed number of rules had already been implemented into the system. The amount of rules given to the system by conventional programming would be a lot smaller in amount compared to the amount of training data needed in feeding the inductive learning system.

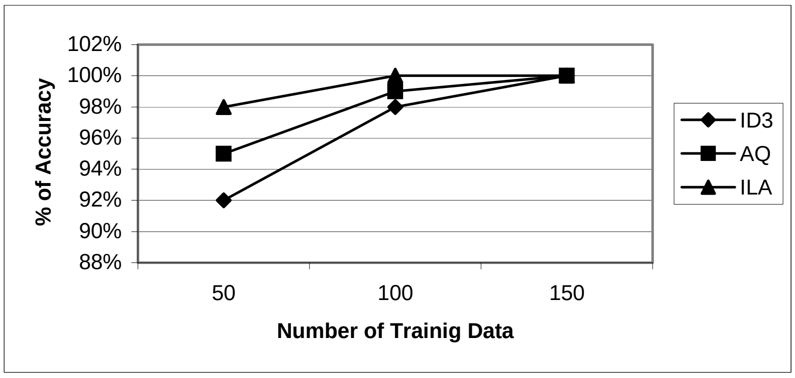

Fig. 5. Accuracy Result on Test Data

Analysis on the Learning Ability and Adaptation. However, the advantage of implementing inductive learning algorithms on timetabling is shown in the next experiment. This experiment was done to show whether the inductive learning algorithms were capable in learning new rules. In the next experiment, new training data sets were given to the system.

The format of the data is the same with the previous data sets, which consists of 50, 100 and 150 training data respectively. The result for the second experiment is as shown in table 4 below.

In the second experiment, the new training sets caused the algorithms to generate new set of rules. These new rules and constraints were based on the new training data. This shows the advantage of the system, where it can adapt to new environment.

This result shows that the inductive learning algorithms not only successfully implemented into the system, but its ability to learn new rules and constraints allows the system to adapt to the new situation and environment without having to reprogram the system.

However the slight disadvantage that can be seen was, once the algorithms learn the new rules, the old rules would be removed. This caused the system to only adapt to one type of environment at a time, and the rules generated were not cumulative. Therefore, in order for the system to acquire both the new rules and the old rules, the new training data must be combined with the old training data. In order words, the algorithms should be taught again for the old rules together with the new rules.

Data Set	Algorithm	Rules Generated	Accuracy
50	ID3	26	95%
	AQ	19	95%
	ILA	15	97%
100	ID3	40	98%
	AQ	29	100%
	ILA	24	100%
150	ID3	51	100%
	AQ	37	100%
	ILA	32	100%

Table 4. Result of the Second Implementation

Comparison on Rules Generated Between 2 Experiments. Another thing that can be noticed from these two experiments was that the results for both experiments were not the same. For example, the first experiment shows that AQ algorithm was 100% accurate only at 150 training data set, but for the second experiment, it was 100% accurate for both 100 and 150 training data sets. To analyze this situation, the results of number of rules from both experiments are shown in the 3 graphs that is figure 6, 7 and 8 below. From these results, it can be seen that even though the amount of rules generated from both experiments were different, the relation between number of training data and number of rules is still the same. It means that the number of training data would not affect the exact amount of rules generated; the only effect was that more training data would result too more generalized rules.

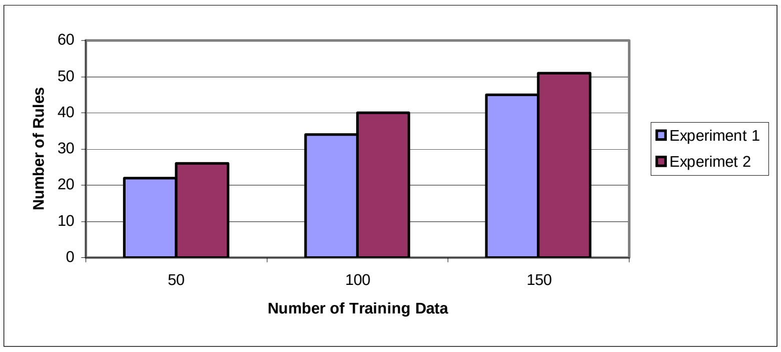

Fig. 6. Comparison of ID3 Rules Results between 2 Experiments

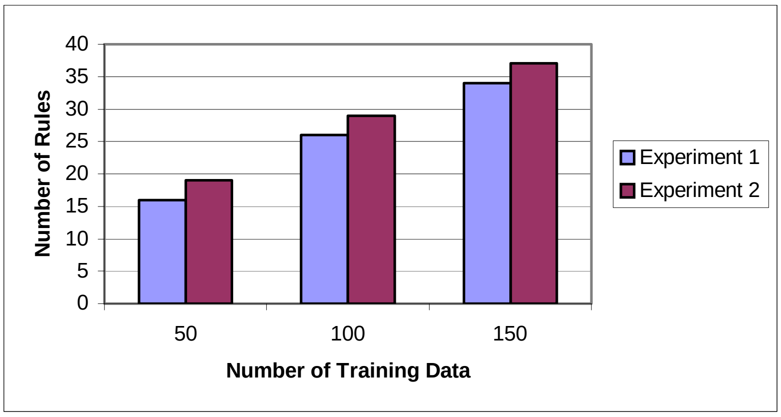

Fig. 7. Comparison of AQ Rules Results between 2 Experiments

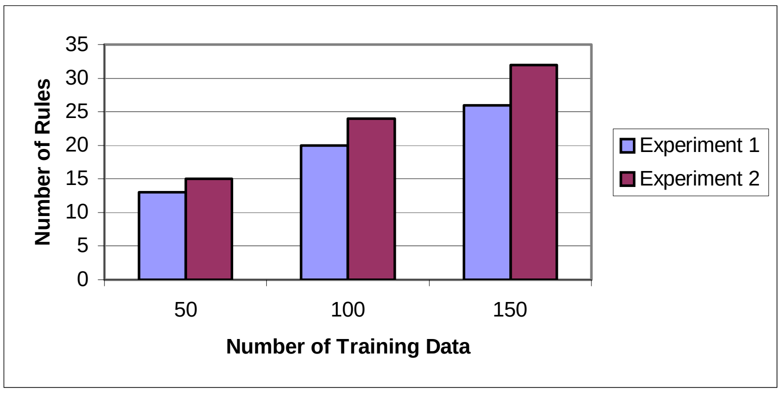

Fig. 8. Comparison of ILA Rules Results between 2 Experiments

Comparison on Accuracy between 2 Experiments. The comparison of accuracy results for each algorithm on both experiments is shown in the 3 graphs that is figure 9, 10 and 11 below. The accuracy results on both experiments also did not show the same value. Still, the direct relation between the increase of accuracy and the increase of training data can be seen. Therefore, the number of training data would not affect the exact percentage of accuracy, but the increased amount of training data would result in the increased percentage of accuracy.

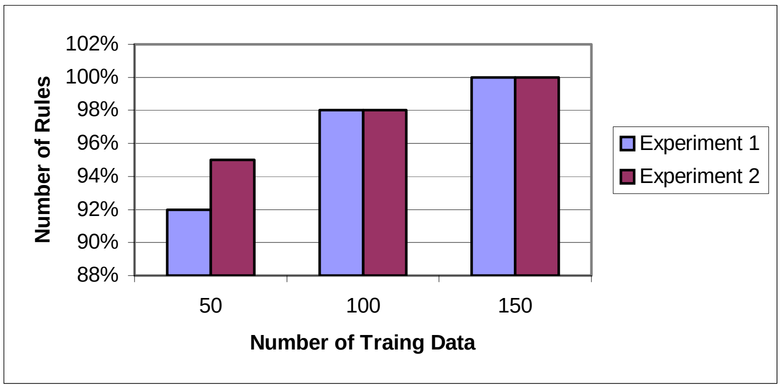

Fig. 9. Comparison of ID3's Accuracy Result on 2 Experiments

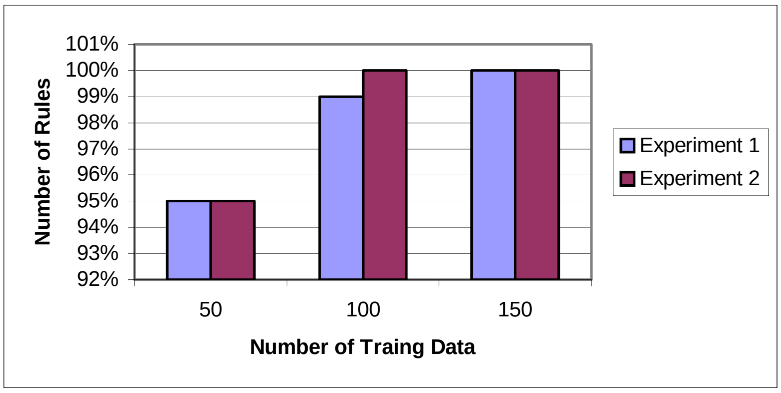

Fig. 10. Comparison of AQ's Accuracy Result on 2 Experiments

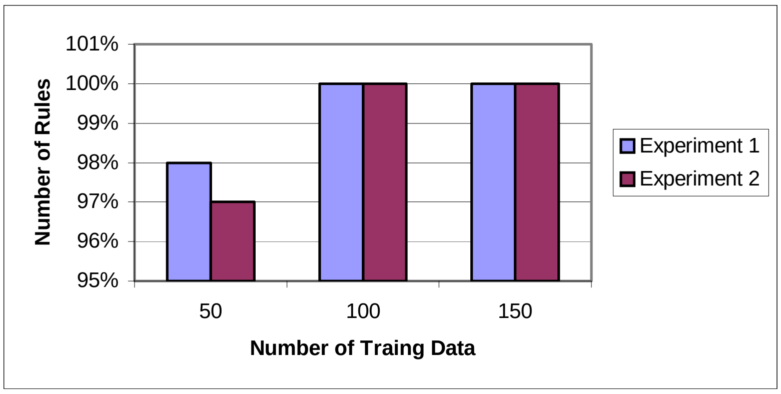

Fig. 11. Comparison of ILA's Accuracy Result on 2 Experiments

B. *Comparison Between the Inductive Learning Algorithms*

Two comparisons methods were done for this study. The first comparison would be on the amount of rules generated by the algorithms. The second comparisons would be on the accuracy of the algorithms

Comparison between Algorithms on the Amount of Rules Generated. The amount of rules generated from the first experiment is shown in figure 12 below.

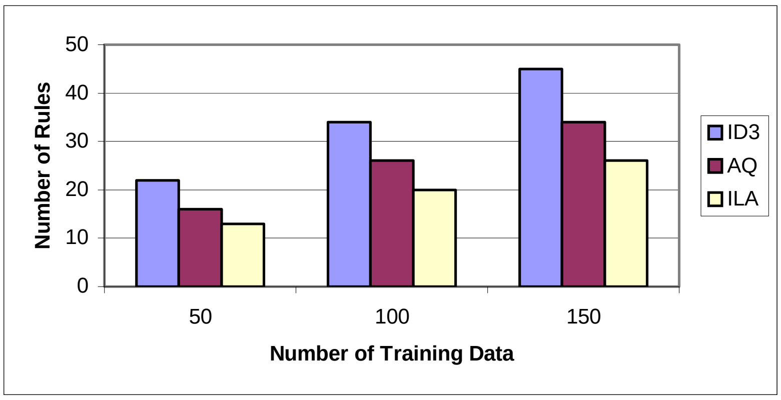

Fig. 12. Comparison on Rules Generated

From the graph, it is shown that for all three training data sets, ILA algorithm had less number of rules compared to ID3 algorithm and AQ algorithm. This shows that ILA is better in its induction and rules generalization compared to the other two. However, the differences between the numbers of rules were smaller at the smaller amount of training data. This means that the performance of ILA would only be very noticeable on the system with large training data set. Therefore, for the system that requires small amount of data, all the three algorithms can be applied.

Comparison between Algorithms on the Percentage of Accuracy of the Algorithm. The second comparison is done on the accuracy of the algorithms towards the test data. The accuracy generated from the first experiment is shown on figure 13 below.
From the graph, it is clearly shown that when the amount of training data is small, ILA algorithm had a better accuracy compared to the two other algorithms. However, when the system was feed with large amount of training data, it could be clearly seen that there were no difference in terms of accuracy.
From this result, it can be said that ILA has a better accuracy in the environment with small training data, but all three algorithms have the same result of accuracy in the environment with higher training data.

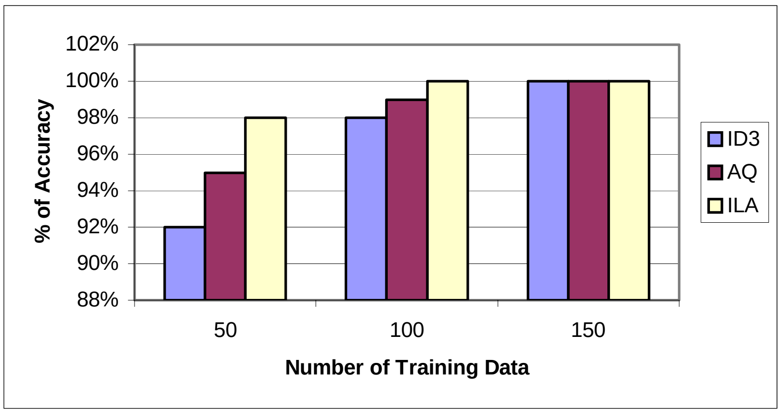

Fig. 13. Comparison on accuracy of Algorithm

V. Conclusion

From the results in section 4, there are several conclusions could be made. The first conclusion is that inductive learning algorithms can be successfully implemented into timetabling. The inductive learning algorithms had successfully recognized and generalized the rules contains in the training data given. The accuracies for the algorithms were also very high, which means the system produced a reliable result.
This result also showed that inductive learning can be successfully implemented in a complex problem domain, and therefore it is very useful to be implemented in the real world problems.
The second conclusion is that the algorithms had the ability to learn new rules and therefore had the ability to adapt to changes.
Finally it can be concluded that between the three algorithms, the ILA algorithm performed better in performance of rules generated and accuracy. ILA algorithm produced less rules yet was more

accurate than the other two algorithms. This showed that the ILA algorithm is better in induction and rules generalization compared to ID3 algorithm and AQ algorithm.

A. *Future Recommendations*

Even though the inductive learning algorithms were successfully implemented in the timetabling domain, still the implementation was on a basic level. Therefore, a few enhancements can be made.

Research should be done to see how the inductive learning algorithms would react when given a bias. Bias is a method that allows the algorithms to have preference towards certain situations. This is useful since in the real world, there are many solutions for a single problem. Therefore by implementing bias, the algorithms would have a preference on certain solutions.

Research should also be done to study the effect of noise on the inductive learning algorithms. Noise is the data that would have no meaning, such as data that have no value or data that contradicts with other data. Since real world data usually have noise; therefore the inductive learning algorithms should be analyzed on how they deal with noise.

References

[1] Abramson, D. 1991. A Parallel Genetic Algorithm for Solving The School Timetable Problem (on line), http://citeseer.nj.nec.com/8438.html

[2] Altintas, N. I. 1995. Incremental Conceptual Clustering Without Order Dependancy: 9. (on line) http://citeseer.nj.nec.com/altintas95incremental.html

[3] Ballard, L. 2002. Topics in Machine Learning: Decision Tree Learning (on line) http://www.cs.brandeis.edu/~cs113/classprojects/~lballard/cs113/proj1.html

[4] Caldeira, J.P. 1997. School Timetabling Using Constraint Logic Programming (on line) http://citeseer.nj.nec.com/goltz99university.html.

[5] Clark, P. 1990. Machine Learning: Techniques and Recent Development (on line) http://citeseer.nj.nec.com/clark90machine.html

[6] Gestwicki, P. 1997. ID3: History, Implementation, and Applications (on line) http://citeseer.nj.nec.com/398697.html

[7] Heylighen, F. 1997. Occam's Razor (on line) http://pespmc1.vub.ac.be/:/OCCAMRAZ.html

[8] Hild, H. 1989. Variations on ID3 for Text to Speech Conversation (on line) ftp.cs.orst.edu/pub/tgd/papers/hild-ms-thesis.ps.gz

[9] Mihai, I. F. 2001. Exploring Places Rated and Body Fat Data Using AQ and Crystal Vision (on line) http://www.galaxy.gmu.edu/stats/syllabi/inft979/MihaiPaper.pdf

[10] Sammut, C. 1993. The Origins of Inductive Logic Programming: A Prehistoric Tale (on line) http://citeseer.nj.nec.com/sammut93origins.html

[11] Tolun, M.R. and Abu-Soud, S.M. An Inductive Learning Algorithm for Production Rule Discovery (on line) http://www.ceng.metu.edu.tr/~tolun/courses/ila.ps

Appendix A: Sample of Data for Single Data Type

ID	Subject Code	Program Code	Part	Day	Start	End	Room	Lecturer	Value
Data 1	ITC 100	CS 110	2	Mon	9:00 AM	11:00 AM	TH1	Pn Aminah	Yes
Data 2	ITC 150	CS 110	3	Tues	9:00 AM	11:00 AM	TH1	Pn Aini	Yes
Data 3	ITC 130	CS 225	4	Mon	9:30 AM	11:30 AM	TH3	En Ali	No

Appendix B: Sample of Data for Pair Data Type

ID	Subject Code	Program Code	Part	Day	Start	End	Room	Lecturer	Value
Data 1	Same	Same	Same	Diff	Same	Same	Diff	Same	Yes
Data 2	Diff	Diff	Same	Same	Same	Same	Same	Diff	No
Data 3	Diff	Same	Same	Same	Near	Near	Far	Diff	Yes

Azlinah Hj Mohamed (MSc Artificial Intelligence, University of Bristol UK, PhD., Universiti Kebangsaan Malaysia) is currently working as a lecturer in Universiti Teknologi MARA. Prior to this she was a Tutor in University of Bristol and a Research Fellow in Universiti Kebangsaan Malaysia. Dr. Azlinah's current areas of interest are Hybrid Techniques and Web-based decision support systems using intelligent agents in electronic government applications. She has presented her research in a number of conferences internationally and locally.

Photo

Mohd Haris Shukri Bin Jahabar (BSc Intelligence Systems, Universiti Teknologi MARA) is currently working for a Software Engineering organization. Prior to this he was a researcher in Universiti Teknologi MARA. He has been involved in building intelligent systems for about 10 years. His current area of interest is Intelligent Agents in networking and communication.