Knowledge Discovery in Databases: A Rule-Based Attribute-Oriented Approach

David Wai-lok Cheung[1]*, Ada Wai-Chee Fu[2] and Jiawei Han[3]**

[1] Department of Computer Science, The University of Hong Kong, Hong Kong.
[2] Department of Computer Science, Chinese University of Hong Kong, Hong Kong.
[3] School of Computing Science, Simon Fraser University, Canada.

Abstract. An attribute-oriented induction has been developed in the previous study of knowledge discovery in databases. A concept tree ascension technique is applied in concept generalization. In this paper, we extend the background knowledge representation from an unconditional non-rule-based concept hierarchy to a rule-based concept hierarchy, which enhances greatly its representation power. An efficient rule-based attribute-oriented induction algorithm is developed to facilitate learning with a rule-based concept graph. An information loss problem which is special to rule-based induction is described together with a solution suggested.

Keywords: learning and adaptive systems, knowledge discovery in databases, rule-based concept generalization, attribute-oriented induction.

1 Introduction

Knowledge Discovery is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [2]. Over the past twenty years, huge amounts of data have been collected and managed in relational databases by industrial, commercial or public organizations. This has created a need and a challenge for extracting knowledge from these databases. Knowledge extracted can be used to support deductive system or intelligent query answering system [7, 8]. Researches have been done with different approaches to tackle this problem [2].

In the previous studies [5], a **Basic Attribute-Oriented Induction** (**Basic AO Induction**) method has been developed for knowledge discovery in databases. It has been implemented in the database learning system **DBLearn** [5]. Both the applicability and performance of the system are encouraging.

A key in Basic AO Induction is the *attribute-oriented concept tree ascension* for concept generalization, a concept hierarchy is used to represent background

* Email: dcheung@csd.hku.hk

** The research of the third author was supported in part by grants from the Natural Sciences and Engineering Research Council of Canada and the Centre for Systems Science of Simon Fraser University.

knowledge supplied by domain experts. The most commonly used concept hierarchies are *concept tree* and *lattice*, in which concepts are generalized to higher level concepts *unconditionally*. For example, in a concept tree defined for the attribute *GPA* of a student database, a *3.6 GPA* (in a 4 points system) can be generalized to a higher level concept, perhaps to the concept *excellent.* This depends only on the GPA value but not any other information of a student. However, in some cases, domain experts (users) may need to represent the background knowledge in such a way that concept generalization would depend not only on the concept itself but also other conditions. For example, in some institutions, they may want to apply different rules to different types of students. The same *3.6 GPA* may only deserve a *good*, *if the student is a graduate*; and it may be *excellent*, *if the student is an undergraduate.*

This suggests that a more general concept generalization scheme should be *conditional* or *rule-based.* To this end, we propose a rule-based hierarchy for background knowledge representation called **Rule-Based Concept Graph**.

The adoption of rule-based concept graph substantially increase the representation and induction power of attribute-oriented generalization. To support this model, a **Rule-Based Attribute-Oriented Induction** (**Rule-Based AO Induction**) has been designed. Both the general principles and an efficient method for Rule-Based AO Induction are described in this paper.

The paper is organized as follows. The primitives of knowledge discovery and the principle of Basic AO Induction are briefly reviewed in Section 2. The general notions of Rule-Based Concept Generalization and Rule-Based Concept Hierarchy are discussed in Section 3. The principles of Rule-based AO induction is presented in Section 4. Applications of rule-based knowledge discovery are discussed in Section 5.

2 Basic Attribute-Oriented Induction

Task-relevant data, background knowledge, and *expected representation of learning results* are the three primitives that specify a learning task in Basic AO Induction [5]. A query that specifies a learning task is used to retrieve the task-relevant set of data from a database. In Basic AO Induction, the retrieved task-relevant tuples are stored in an **initial relation**, and concept tree is being used to represent background knowledge. Many kinds of rules, such as *characteristic rules*, and *discriminant rules* can be discovered by induction processes [5]. A tuple in a relation can be viewed as a logic formula in conjunctive normal form, and a relation is characterized by a large set of disjunctions of such conjunctive forms [3, 8]. Thus, both the data for learning and the rules discovered can be represented in either relational form (tuples) or first-order predicate calculus.

The fundamental idea in AO Induction is to generalize the initial relation to a **prime relation** and then to the **final relation**. During this process, attribute values in the relation are converted iteratively to higher level attribute values with respect to generalization paths in a concept tree. Tuples that have the same higher level attribute values are merged to form higher level tuples. The prime relation is generated when the number of distinct values of each attribute in the relation is less than the predefined **desirable attribute threshold**. Then the prime relation will be reduced further by selectively generalize some attributes until the size of the final relation is less than the predefined **generalization relation threshold**[5]. At the end, tuples in the final relation which represent certain regularities are converted into knowledge rules. Note that the two notions *attribute value* and *concept* are used interchangeably here. When a generalized tuple is converted into a rule, attribute values (generalized) are converted to a predicates in the rule which are intrepreted as high level concepts.

3 Rule-Based Concept Generalization and Concept Hierarchies

Concept generalization is a key component in database inductive learning. Concepts are ordered in a concept hierarchy by levels from specific (lower level) to general (higher level). Generalization is achieved by ascending concepts along the paths of a concept hierarchy.

In general, a concept can ascend via more than one path. A **generalization rule** can be assigned to a path in a concept hierarchy to determine whether a concept can be generalized along that path. For example, the generalization of a student's GPA in (2.0 - 2.49) can be performed along two paths with the following two generalization rules:

If a student's GPA is in the range (2.0 - 2.49) and he is an undergrad, then it is an average GPA.

If a student's GPA is in the range (2.0 - 2.49) and he is a grad, then it is a poor GPA.

Concept hierarchies whose paths have associated generalization rules are called **rule-based concept hierarchies**, and there are three types of rule-based concept hierarchies. In the first type, **deductive generalization rules** are associated with the generalization paths. A deductive generalization rule has the form: $A(x) \wedge B(x) \longrightarrow C(x)$. For a tuple x, concept (attribute value) A can be generalized to concept C (higher level attribute values) if condition B can be satisfied by x. The condition $B(x)$ can be a simple predicate or a very complex logic formula involving different attributes and relations. In the second type, **computational rules** are associated with the hierarchy. Each rule is represented by a condition which is value-based and can be computed from an attribute or a tuple or the database. The truth value of the condition would then determine whether a concept can be generalized via a path. The third type is a hybrid hierarchy in which either one of the above two types of rules can be associated with a path. In the scope of database induction, the same technique can be used on all the different types of rule-based hierarchies. Therefore, in the rest of this paper, we will use deductive-rule-based concept graph as a typical concept hierarchy.

4 Rule-Based Attribute-Oriented Induction

A rule-based AO induction system is defined by (EDB, RCH, DS, TS, KRS, ATh, RTh). EDB is the underlying extended relational databases, RCH is a set of deductive-rule-based concept hierarchies, one for each attribute, DS is a deductive system to support concept generalization, it includes all the deductive rules in RCH and a set of supporting rules. TS is a set of task specifications and KRS is a scheme, which is the first order predicate calculus, to represent knowledge. ATh and RTh are respectively the *desirable attribute threshold* and the *generalization relation threshold.* The goal of the induction system (ERD,RCH,DS,TS,KRS,ATh,RTh) is to extract general and useful knowledge from data in ERD that satisfy a task specification in TS by doing generalization along the direction provided by RCH with the support of DS. The learned result will be represented by the scheme defined by KRS.

Example 1. In this example, the EDB is a university student database defined by the schema *Student (Name, Status, Sex, Major, Age, Birthplace, GPA).* The learning task in Ts is to discover the characteristic rules for CS students. The concept graph in RCH for the attribute GPA is defined in Fig. 1 and the associated deductive rules defined in Fig. 2. An equation $A \wedge B \rightarrow C$ in Fig. 2 represents the conditional rule: *If x satisfies condition B, then its attribute value A can be generalized to the higher level concept C.* For example, both R_6 and R_7 denote the rules:

R_6*: If a student's GPA is in (3.49 - 3.8) and (s)he is a graduate, then it can be classified as a good GPA.*

R_7*: If a student's GPA is in (3.49 - 3.8) and (s)he is an undergrad, then it can be classified as an excellent GPA.*

During the generalization process, one of these two rules would be applied to generalize the GPA in the range $(3.49 - 3.8)$ to either *good* or *excellent*. □

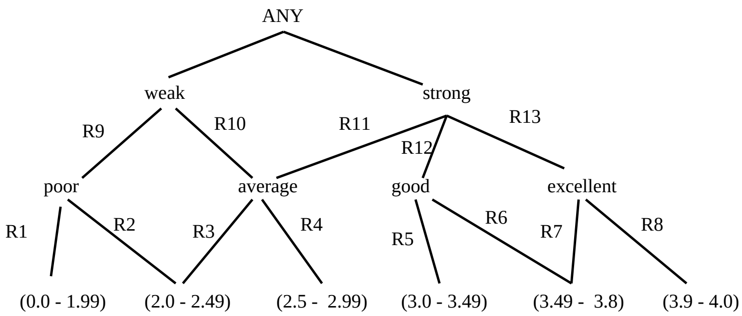

Fig. 1. A rule-based concept graph for *GPA*

$R_1 : \{0.0 - 1.99\} \rightarrow poor$
$R_2 : \{2.0 - 2.49\} \wedge \{graduate\} \rightarrow poor$
$R_3 : \{2.0 - 2.49\} \wedge \{undergraduate\} \rightarrow average$
$R_4 : \{2.5 - 2.99\} \rightarrow average$
$R_5 : \{3.0 - 3.49\} \rightarrow good$
$R_6 : \{3.49 - 3.8\} \wedge \{graduate\} \rightarrow good$
$R_7 : \{3.49 - 3.8\} \wedge \{undergraduate\} \rightarrow excellent$
$R_8 : \{3.9 - 4.0\} \rightarrow excellent$
$R_9 : \{poor\} \rightarrow weak$
$R_{10} : \{average\} \wedge \{senior, graduate\} \rightarrow weak$
$R_{11} : \{average\} \wedge \{freshman, sophomore, junior\} \rightarrow strong$
$R_{12} : \{good\} \rightarrow strong$
$R_{13} : \{excellent\} \rightarrow strong.$

Fig. 2. Conditional Generalization Rules for GPA

4.1 Induction Phase One : From Initial Relation to Prime Relation

The goal to the first phase is to generalize and reduce the initial relation to the prime relation. We describe here an efficient two passes algorithm for inducing the prime relation from the initial relation.

First Pass - Determination of the Desirable Level The goal of this pass is to make use of the deductive system to find out, for each attribute A, the **minimum desirable level**[6] L_A in an efficient way.

In one scan of the initial relation, for each attribute A, the deductive system DS will compute, from the bottom level upward, for each level of the concept hierarchy, how many distinct values can be generalized from the ground facts in the initial relation. The level, in which the number of distinct values that can be generalized from the initial relation is just smaller than the threshold ATh, becomes the *minimum desirable level* L_A.

Second Pass - Tuple Generalization and Merge The initial relation is scanned again in this pass and every tuples and its attributes are generalized by DS to the desirable levels computed in the first pass. At the same time, the generalized tuples are compared and identical tuples are merged. The resulting generalized relation becomes the prime relation.

Example 2. The two passes algorithm is applied to the initial relation and the learning task described in Example 1 and the induction is done with respect to the rule-based concept graph described in Example 1 and Fig. 1. The prime relation generated is presented in Table 1. □

Status	Sex	Age	Birthplace	GPA	Vote
undergrad	M	16_25	Canada	average	40
undergrad	M	16_25	Canada	good	20
undergrad	F	16_25	Canada	excellent	10
...	...	...	...	...	...
grad	M	25_30	Canada	poor	6
grad	M	25_30	Canada	good	4
grad	F	25_30	Canada	excellent	4

Table 1. A Prime Relation from the Rule-Based Generalization.

4.2 Induction Phase Two : From Prime Relation to Final Relation

In the second phase, some selected attributes can be generalized further and the generalization-comparison-merge process will be repeated until the size of the relation is smaller than the **generalization relation attribute** RTh. (Some selected attributes can also be removed before the generalization starts.)

In the case of rule-based induction, an **information loss** problem can become an issue in reducing a prime relation to the final relation. In the basic AO induction, attributes in a prime relation can always be generalized further by concept tree ascension because the generalization is based only on the current generalized attribute values. However, this is not the case in the rule-based induction because the application of a rule may depend on the information not currently available in the prime relation. Generalization may sometimes result in *the loss of information*, which could be crucial in the following cases:

(1) A rule may depend on an attribute which has been removed.

(2) A rule may depend on an attribute whose concept level in the prime relation has been generalized too high to match the condition of the rule.

(3) A rule may depend on a condition which can only be evaluated against the initial relation, e.g., the number-of-tuples in the relation.

For example, one cannot decide whether the first tuple in Table 1, which represents "undergrad with an average GPA", should be generalized according to R_{10} to *weak* or R_{11} to *strong* because the status information (*freshman/sophomore/junior/senior*) has been lost during the previous generalization. In fact, the 40 student tuples which are merged into this tuple may have all the different status.

To tackle the *information loss problem*, a **Backtracking Algorithm** is proposed here. In a generalized relation (intermediate or prime relation), a generalized tuple is the result of merging of a set of tuples in the initial relation. This set of tuples in the initial relation is called the **source set** of the generalized tuple; whereas the generalized tuple is called their **covering tuple**. When the GPA attribute with the values {*poor, average, good, excellent*} in Table 1 are to be further generalized to {*weak, strong*}, more than one rule may be applicable to a tuple in Table 1. For the first tuple in Table 1, both R_{10} and R_{11} may be applicable to different tuples in its source set. Therefore, tuples in its source set may have to be generalized to the two different concepts *weak* and *strong* separately. In order to do this, the covering tuples have to be backtracked to the tuples in their source set.

Principles of the Backtracking Algorithm

1. *Backtrack the tuples in the prime relation to the tuples in their source set.* This can be implemented efficiently by adding a virtual attribute *covering-tuple-id* in the initial relation to record the identity of its corresponding covering tuple.
2. *Further generalize some selected attributes to some higher levels.* This generalization has to be done by the deductive system DS on the the initial relation, not on the prime relation as in the case of Basic AO Induction.
3. *Compare and merge the generalized tuples.* Comparison is performed among the tuples associated with the same covering-tuple-id and relying only on the attributes which are selected for further generalization. Such a comparison and merge of generalized tuples results in an **enhanced-prime relation**.
4. *Map the merged tuples back to the prime relation and split the tuples in the prime relation accordingly.* All the tuples in the enhanced-prime relation sharing the same covering-tuple-id will be mapped to the corresponding covering tuple in the prime relation. Tuples in the prime are then split into several tuples according to this mapping and votes are adjusted. The split tuples are then generalized on the selected attributes to the corresponding values in the enhanced-prime relation.
5. *Merge the generalized tuples in the split prime relation.* If the size of the tuple-merged relation is still larger than the generalization relation threshold, repeat the generalization process until the size of the generalized relation is within the threshold.

Name	Status	Sex	Age	Birthplace	GPA	Covering-tuple-id
–	Junior	M	20	Vancouver	2.3	1
...	...	...	...	...	...	...
–	Sophomore	M	21	Calgary	2.3	1
–	Freshman	M	18	Toronto	2.4	1
...	...	...	...	...	...	...
–	Junior	M	19	Ottawa	3.1	2
–	Ph.D.	M	30	Waterloo	3.9	40

Table 2. Extend the Initial Relation with the attribute Covering-tuple-Id.

Example 3. The Backtracking Algorithm is illustrated by further generalizing the attribute GPA from $\{poor, average, good, excellent\}$ to $\{weak, strong\}$ for the prime relation in Table 1. The initial relation in Example 1, with a virtual attribute *covering-tuple-id* associated, is presented in Table 2. For example, tuples whose *covering-tuple-id* are 1 are those that have been generalized and merged into the first tuple in Table 1. Apply the rules R_9 to R_{13} in Fig. 2 to the initial relation in Table 2 and generalize the GPA attributes to $\{weak, strong\}$. After comparison and merge, the enhanced-prime relation (Table 3) in generated. Since the first two tuples in Table 3 (covering-tuple-id = 1) are mapped to the first tuple in Table 1, the first tuple in the prime relation Table 1 should split into two. The result of tuple splitting is presented in Table 4. After backtracking and tuple splitting, identical tuples in Table 4 can be merged again to reduce its size. □

Covering-tuple-id	GPA	Vote
1	strong	10
1	weak	30
2	strong	20
...	...	...
40	strong	4

Table 3. Enhanced Prime Relation

Status	Sex	Age	Birthplace	GPA	Vote
undergrad	M	16_25	Canada	strong	10
undergrad	M	16_25	Canada	weak	30
undergrad	M	16_25	Canada	strong	20
...	...	...	...	...	...
grad	F	25_30	Canada	strong	4

Table 4. Result of Splitting the Prime Relation (Table 1)

The Rule-Based Attribute-Oriented Induction process is summarized in the following algorithm:

4.3 Algorithm 1

Rule-Based Attribute-Oriented Induction

Input : *A learning task specification Ts /* R is the initial relation */*
Output: *The final relation R_f.*
Method:
Phase One: *Inducing the prime relation R_p from R*

Step 1: *$R_t := R$; /* R_t is a temporary relation */*

Step 2: *for each attribute A_i of R_t, compute its minimum desirable level L_i;*

Step 3: *for each tuple in R_t and each attribute A_i, deduce its generalized value in level L_i and replace the old value of A_i by the deduced value; /* merge identical tuples and register the number of tuples merged as votes; a virtual attribute covering-tuple-id is added to R to record identity of the corresponding covering tuple in R_t. */*

Step 4: *$R_p := R_t$;*

Phase Two: *Inducing the final relation R_f from R_p by backtracking*
/ Let N be the size of R_p */*
WHILE $\mathsf{N} > \mathsf{RTh}$ **DO**
BEGIN

Step 1: *Select a set of attributes A_j, ($1 \leq j \leq k$, where $k < n$), from R_p for further generalization; for each A_j, a desirable level L_j is also decided;*

Step 2: *$R_t := R$;*

Step 3: *For each tuple in R_t and for each selected attribute A_j, deduce its generalized value in level L_j; /* Compare tuples in R_t by the selected attributes and their covering-tuple-id; merge and project the identical tuples by the selected attributes and the covering-tuple-id into a temporary relation R_{ep} (enhanced prime relation). */*

Step 4: *$R_{tp} := R_p$;*

Step 5: *For each tuple t in R_{tp}, split it into a set of tuples corresponding to those in R_{ep} whose covering-tuple-id are equal to the identity of t; /* Merge identical tuples in the split R_{tp}. */*

Step 6: *$R_p := R_{tp}$;*

END □

The complexity of the above algorithm can be decomposed into the cost of induction and that of deduction. The induction portion of the algorithm is very efficient. However, the cost of the deduction portion will depend on the complexity of the rules and the efficiency of the deductive system DS. The deduction rules in the algorithm are the conditional rules associated with a concept graph, which in most cases, are very simple conditional rules. Therefore, it is practical to assume each deduction process is bounded by a constant in the analysis of the algorithm.

Theorem 1. *If the cost of generalizing an attribute to any level is bounded, the complexity of the Rule-Based Attribute-Induction Algorithm is $O(n \log n)$, where n is the number of tuples in the initial data relation.* □

5 Discussion and Conclusion

A rule-based attribute-oriented induction technique is proposed and studied in this paper. The technique extends the previously developed basic attribute-oriented induction method and make the method more generally applicable to databases containing both data and rules. Knowledge discovery in databases has wide applications. First, the knowledge rules discovered are based on a large data set. They represent important regularities in the database. Second, knowledge discovery helps construction of large knowledge-base. The huge numbers of on-line databases with broad applications serve as an enormously rich source for discovering knowledge rules. Third, another important application of knowledge discovery is to support cooperative query answering [7].

This study develops a general model for Rule-Based AO Induction which handles induction on a rule-based concept hierarchy. Even though a rule-based concept graph is substantially more complex than a concept tree, the complexity of the rule-based AO induction remains at the same order to that of the basic induction. Experiments on medium sized data sets have demonstrated the effectiveness and efficiency of the induction method. Our proposed system extends the database learning system to a rule-based induction system. Future experimental work has been planned in this direction. Furthermore, efficient extraction of knowledge from multidatabases, friendly graphical user interface, and efficient knowledge browser are among the interesting topics in knowledge discovery in databases for future research.

References

1. M.L. Brodie and S. Ceri. On Intelligent and Cooperative Information Systems: A Workshop Summary. *International Journal of Intelligent and Cooperative Information Systems*, V1 N2:233–248, 1992.
2. W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus. Knowledge discovery in databases: An overview. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 1–27. AAAI/MIT Press, 1991.
3. H. Gallaire, J. Minker, and J. Nicolas. Logic and databases: A deductive approach. *ACM Comput. Surv.*, 16:153–185, 1984.
4. J. Han, Y. Cai, and N. Cercone. Knowledge discovery in databases: An attribute-oriented approach. In *Proc. 18th Int. Conf. Very Large Data Bases*, pages 547–559, Vancouver, Canada, August 1992.
5. J. Han, Y. Cai, and N. Cercone. Data-driven discovery of quantitative rules in relational databases. *IEEE Trans. Knowledge and Data Engineering*, 5:29–40, 1993.
6. J. Han, Y. Cai, N. Cercone, and Y. Huang. DBLEARN: A knowledge discovery system for databases. In *Proc. 1st Int. Conf. on Information and Knowledge Management*, pages 473–481, Baltimore, Maryland, Nov. 1992.
7. A. Motro and Q. Yuan. Querying database knowledge. In *Proc. 1990 ACM-SIGMOD Int. Conf. Management of Data*, pages 173–183, Atlantic City, NJ, June 1990.
8. J. D. Ullman. *Principles of Database and Knowledge-Base Systems, Vols. 1 & 2.* Computer Science Press, 1989.