KNOWLEDGE REPAIR MECHANISMS: EVOLUTION vs. REVOLUTION

by

Ryszard S. Michalski

Report #946, Department of Computer Science, University of Illinois, Urbana, Illinois, July, 1985.

File No. UIUCDCS-F-85-946

KNOWLEDGE REPAIR MECHANISMS:
Evolution vs Revolution

by

Ryszard S. Michalski

Intelligent Systems Group
Department of Computer Science
University of Illinois Urbana-Champaign

This research was supported in part by the National Science Foundation under grant DCR 84-06801, and in part by the Office of Naval Research under grant N00014-82-K-0186.

KNOWLEDGE REPAIR MECHANISMS:
Evolution vs. Revolution

Ryszard S. Michalski
Massachussetts Institute of Technology*

ABSTRACT

When new facts contradict established knowledge, this knowledge can be repaired by an evolutionary approach or by a revolutionary approach. The evolutionary approach makes incremental modifications to the appropriate segments of knowledge, while the revolutionary approach replaces old knowledge with new knowledge, generated from scratch. Within the evolutionary approach two methods are discussed: a *premise-based* method, and an *exception-based* method. Assuming that knowledge is represented in the form of rules, the premise-based method modifies the main conditions (premises) of the rules, while the exception-based method accumulates exceptions to the rules and formulates preconditions for rule application. In the context of machine learning (as opposed to human learning) a *full memory* evolutionary approach is advocated, which incrementally improves knowledge structures, but does not forget facts. An exception-based learning method is discussed that represents knowledge in the form of *censored production rules*. Such rules are created by augmenting ordinary rules by an *unless* or *provided* condition.

INTRODUCTION

Almost all human knowledge is fluid. It changes because the world around of us changes, or because new facts we learn about the world call for a modification of what we know. When our knowledge of some subject is inconsistent with newly observed facts, we may choose one of several options. We may ignore the inconsistency, hoping that it is insignificant or accidental, and retain our knowledge unaltered. Or we may make incremental modifications to the appropriate part of our knowledge. This is an *evolutionary* approach. Another option is to throw away this piece of knowledge altogether and develop another one from scratch. This is a *revolutionary* approach.

The revolutionary approach may bring about new and significantly better knowledge. The new knowledge so obtained may be radically different from the old knowledge, or closely related (as Einstein's reformulation is related to Newton's second law of dynamics). It is also relatively easy to implement such an approach on a computer, because it does not require an intimate understanding of the current body of knowledge, nor does it need sophisticated knowledge repair mechanisms.

* On leave from the University of Illinois at Urbana-Champaign.

On other hand, the revolutionary approach requires starting from first principles and using original facts and observations. It may also involve adopting new viewpoints, discovering unexpected relationships, formulating new ideas. Therefore it is difficult, often inefficient and time consuming. This approach does not take advantage of the parts of the old knowledge structures that are correct. Its difficulty grows rapidly with the extent of knowledge to be developed anew. The revolutionary approach to knowledge improvement seems thus to be most advantageous for small bodies of knowledge.

Until now, our machine learning programs have been primarily oriented toward acquiring only small amounts of knowledge, using relatively simple inductive learning techniques. Consequently, it is not surprising that most implemented up-to-date rule learning techniques use an non-incremental, revolutionary approach.

As knowledge bases of our AI systems grow larger and larger, and the problem of knowledge acquisition bottleneck worsens, machine learning research needs to put greater emphasis on the development of efficient methods for incremental learning and knowledge repair. This includes developing knowledge representations that not only facilitate processes of inference derivation, but are also amenable to easy modification and improvement.

The evolutionary approach to knowledge repair can be accomplished in two basic ways: by incrementally refining the main body of knowledge (the *premise-based* method), or by accumulating exceptions and formulating preconditions for applying knowledge currently held (the *exception-based* method). The latter method can be illustrated by a simple way in which Newton's second law of dynamics can be improved. Instead of replacing it with Einstein's more precise (but also more complicated) equation, one can formulate the area of applicability of Newton's equation, and use it within this area. And this is what we usually do for simple physics problems on earth.

An important question is how to decide when to use the evolutionary approach and when to use the revolutionary approach. The decision can be made on the basis of estimates of costs and benefits of applying either approach in a given situation. For the revolutionary approach, there is the cost of developing a new body of knowledge from scratch. A potential benefit is that the new knowledge may be simpler and/or better. For the evolutionary approach, there is the cost of modifying already possesed knowledge. This cost depends on many factors, such as the complexity of this body of knowledge, amount of repair needed and the availability of appropriate skills and tools. A potential benefit is that in some situations a modification can be made simply and quickly.

The rest of this paper is concerned with the evolutionary approach to knowledge repair. It addresses both, the premise-based and the exception-based method of incremental learning, assuming that knowledge is represented in the form of rules. It describes a *full memory* rule refinement method, which modifies premises of the rules using both, the newly observed facts and those previously employed. An exception-based refinement method is also briefly described, which uses *censored production rules* as a knowledge representation. Such rules are created by augmenting ordinary rules by an *unless* or *provided* condition.

NEED FOR INCREMENTAL KNOWLEDGE REFINEMENT

Typically, in ordinary life matters and decision making, humans (as individuals, not as a group) employ incremental learning methods. They tend to use any newly obtained information to refine what they already know, rather than

to completely change or reformulate their knowledge. There seem to be several reasons for such an incremental style of repairing knowledge.

One reason is that we live in a world which continuously changes, and we must react to these changes as they occur. The information comes to us sequentially, and we have to process it and relate it to our other knowledge as we receive it. Also, we cannot consciously erase what we know, as our memory does not have an *erase* command. When some new information contradicts a part of our knowledge, we may try to modify this part appropriately, but we cannot simply remove it from memory, and replace it with a new better part.

Another important factor is the limitation of our memory. We cannot store and have easy access to all the information we have ever been been exposed to. We seem to store and have access to only the most prominent facts and generalizations. One may also observe, that the personal knowledge base of an adult appears to be quite large, and it would be difficult to make any radical modification of it.

The first factor, the sequential flow of information, reflects the intrinsic nature of the world, and cannot be changed. Other factors, however, the lack of erase instruction and the memory limitation, apply to people but not to modern computers. For contemporary computers, storage, fast retrieval, and deletion of vast amounts of information is not infeasible. It is therefore argued that when implementing learning methods on a computer, it may be advantageous in certain situations to employ a *full memory* method, which incrementally refines knowledge structures, but does not forget facts (Reinke and Michalski, 1985). The strength of such a method lies in its ability to use all the original facts for guiding the process of modifying and generalizing knowledge stuctures, and selecting alternative solutions. Also, such a learning method guarantees the completeness and consistency of the modified knowledge with all the facts.

PREMISE-ORIENTED FULL MEMORY LEARNING:
Incremental AQ Algorithm

The basic top-level algorithm used in many of our learning programs is AQ (the quasi-minimal algorithm), or some simplified version of it. It was originally developed for solving the general covering problem very efficiently (Michalski, 1969), and subsequently adapted to problems of inductive learning. The algorithm generates a cover, that is, a set of rules that generally describe all positive learning events and none of the negative events. To do so, it employs the concept of a *star* of one positive event against all negative events, i.e., the set of all most general and consistent concepts or rules that explain a single positive event (Michalski, 1983). One advantage of this algorithm is that elements of any star can always be conjunctive concepts (and thus very simple concepts), and another that it allows to estimate the difference between the generated cover and the minimum cover.

In a simplified version, algorithm AQ generates a star of a randomly chosen (positive) event, and selects the best concept from the star according to a flexible criterion. If the selected concept does not cover (explain) all the positive events, then a new star is generated for some so far uncovered learning event, and the process repeats until all positive events are covered. In this form it is a non-incremental learning algorithm.

There have been two incremental versions of this algorithm, one which does not remember past learning events (Michalski and Larson, 1978), and one employing *full memory* of past events (Reinke, 1985; Reinke and Michalski, 1985). Here we will briefly describe only the version utilizing full memory.

The problem can be formulated as follows. Given is a set of rules, a set of newly acquired facts (new learning events), and a set of previous facts from which the rules were induced (old learning events). Suppose that some of the new learning events contradict the rules. The goal is to transform the original set of rules to a new set, such that the new rules are consistent and complete with regard to all, new and old learning events. Moreover, the new set of rules should be the most preferred one (according to some assumed criterion) among all alternative such sets of rules.

We will assume that the premise of any rule is either a single conjunctive concept, or a disjunction of such concepts. To make rules consistent and complete with regard to all newly acquired events, some rule premises may have to be specialized (to uncover new negative events that are incorrectly covered), and some premises may have to be generalized (to cover new positive events that are not covered).

The incremental algorithm starts by determining a set of rule components (conjunctive concepts in rule premises) that cover new negative events. These components need to be specialized. Each such component covers some (old) positive events and some (new) negative events. The specialization is accomplished by applying the non-incremental AQ algorithm to such a "local" learning problem (to determine one or more components that cover only positive events).

The next step generalizes all rule components (the newly specialized ones and the original ones) to cover those new events that were not covered by the original rules. This is done by reapplying the non-incremental version of the algorithm, treating each rule component as a generalized learning event.

This incremental learning algorithm was tested on a series of problems in the domains of insect classification, chess endgames and plant disease diagnosis (Reinke, 1985; Reinke and Michalski, 1985). Experiments have shown that the incremental learning method was between 5 to 100 times faster than the non-incremental method. The complexity and the performance of rules learned incrementally were on the avarage only slightly worse than that of the rules learned in one step (i.e., non-incrementally).

RULE REPAIR BY DETERMINING EXCEPTIONS AND PRECONDITIONS

Suppose that a given set of rules (or a theory) works well most of the time, but occasionally misfires. We may collect cases when rules do not work, and apply the above or other incremental learning method to develop correct rules. Or we may develop new correct rules from scratch. In science and other areas where standards for consistency and precision are very high, rules (or theories) that are only partially correct are not acceptable. Efforts will be made to determine correct rules through either an evolutionary or revolutionary approach. If the problem is sufficiently important, these efforts will be extended regardless of cost.

In commonsense reasoning, or in solving complex practical problems, such as the ones for which we are developing expert systems, it is not always possible to have perfect rules. Due to the lack of precise knowledge of the domain, the cost of obtaining all needed facts, or the time and other limitations, we often use approximate rules and uncertain theories. In these situations, an exception-based method can be especially useful.

In such a method, the cases for which a given rule does not work are collected together and treated as exceptions. Various issues related to representing and reasoning with exceptions are discussed by Etherington and Reiter (1983), Winston (1981 and 1983) and Minsky (1985). In addition (or alternatively),

situations in which the rule works well are characterized and generalized into a precondition for the rule application.

In a recent work on *variable precision logic*, Michalski and Winston (1985) developed a representation system for expressing and reasoning with exceptions and preconditions (Michalski and Winston, 1985). The system employs *censored production rules*, which are in the form:

If	*<premise>*	
then	*<decision>*	
unless	<censor>	(1)

The *censor* states conditions, which when satisfied reverse the *decision*. It is assumed that the censored conditions occur rarely. It has been shown, that from a *logical viewpoint*, the **unless** operator is equivalent to the exclusive-or operator. From an *expository viewpoint*, the **if-then** part is assumed to carry the important causal or other information, while the **unless** part acts as a switch that changes the polarity of the the decision. A complementary form of a censored production rule uses the **provided** operator, which states the preconditions for the rule. From the logical viewpoint the rule (1) is equivalent to the following rule with the **provided** operator:

If	*<decision>*	
then	*<decision>*	
provided	*<~censor>*,	(2)

where ~ denotes the negation operator.

A preliminary method for learning using censored production rules is described by Becker (1985). In one of the experiments, his program was given a number of cases when after turning on the key the car started, and cases when it did not start. The program learned the following rule (expressed here in a slightly editted form):

If	the ignition key turned on	
then	car starts	
unless	gas tank is empty or battery is dead.	(3)

The rule seems to reflect well our commonsense reasoning about starting a car.

ACKNOWLEDGEMENT

The author thanks Gail Thornburg and Marek Holynski for useful comments and discussions. This research was done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's research is provided in part by the Advanced Research Projects Agency under the Office of Naval Research Contract N00014-80-C-0505.

This work was also supported in part by grants awarded to the University of Illinois at Urbana-Champaign from the National Science Foundation, grant No. DCR-8406801, and by the Office of Naval Research, grant No. N00014-12-K-0186.

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-F-85-946	2.	3. Recipient's Accession No.
4. Title and Subtitle Knowledge Repair Mechanisms: Evolution vs Revolution			5. Report Date July 1985 6.
7. Author(s) Ryszard S. Michalski			8. Performing Organization Rept. No.
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, IL 61801			10. Project/Task/Work Unit No. 11. Contract/Grant No. DCR 84-06801 N00014-82-K-0186
12. Sponsoring Organization Name and Address National Science Foundation Washington, DC Office of Naval Research Arlington, VA			13. Type of Report & Period Covered 14.
15. Supplementary Notes			

16. Abstracts

When new facts contradict established knowledge, this knowledge can be repaired by an evolutionary approach or by a revolutionary approach. The evolutionary approach makes incremental modifications to the appropriate segments of knowledge, while the revolutionary approach replaces old knowledge with new knowledge, generated from scratch. Within the evolutionary approach two methods are discussed: a premise-based method, and an exception-based method. Assuming that knowledge is represented in the form of rules, the premise-based method modifies the main conditions (premises) of the rules, while the exception-based method accumulates exceptions to the rules and formulates preconditions for rule application. In the context of machine learning (as opposed to human learning) a full memory evolutionary approach is advocated, which incrementally improves knowledge structures, but does not forget facts. An exception-based learning method is discussed that represents knowledge in the form of censored production rules. Such rules are created by augmenting ordinary rules by an unless or provided condition.

17. Key Words and Document Analysis. 17a. Descriptors

Machine Learning
Incremental Learning
Production Rules
Knowledge Acquisition

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 9
	20. Security Class (This Page UNCLASSIFIED	22. Price