In Proceedings of the AAAI-91 Workshop
Knowledge Acquisition: From Science to Technology to Tools, Anaheim, CA, 1991

Steps Toward Automating Knowledge Acquisition for Expert Systems

Gheorghe Tecuci*
Center for Artificial Intelligence, Department of Computer Science
George Mason University, 4400 University Drive, Fairfax, VA 22030-4444
email: tecuci@aic.gmu.edu

Abstract

This paper presents a learning-based approach to the automation of knowledge acquisition for expert systems. An expert system is viewed as an explicit model of a human expert's competence and performance. We distinguish three phases in the development of such a model. The first one consists of defining a framework for the model, in terms of a knowledge representation formalism and an associated problem solving method. The second phase consists of defining a preliminary model that describes the basic concepts of the expertise domain. The last phase consists of incrementally extending and improving the domain model through learning from the human expert. The paper describes the learning system NeoDISCIPLE which illustrates the usefulness of six principles for automating the knowledge acquisition process: expert system building as a three-phase modeling of human expertise, understanding-based knowledge extension, knowledge acquisition through multistrategy learning, consistency-driven concept formation and refinement, closed-loop learning, and cooperation between the human expert and the learning system.

1 Introduction

Recently there was a growing interest in devising methods, techniques and systems for automating knowledge acquisition for expert systems. On one side, the knowledge acquisition community tries to automate the existing techniques for knowledge elicitation and domain modeling (Marcus, 1988; Boose et al., 1989). On the other side, the machine learning community tries to apply the learning methods and techniques to the knowledge acquisition task (Wilkins et al., 1986; Morik, 1989; Bareiss et al., 1990).

One approach to the automation of knowledge acquisition is to build tools that make a strong assumption about the problem solving method used by the expert systems they create (Marcus, 1988; Klinker, 1988). In this paper we present a complementary approach that exploits several general principles emerging from the field of machine learning. These principles form the basis of a general learning system shell called NeoDISCIPLE, which is an extension and a generalization of the DISCIPLE system (Tecuci, 1988; Tecuci and Kodratoff, 1990). We present these principles and their implementation into NeoDISCIPLE.

2 Principles for automating the knowledge acquisition process

2.1 Expert system building as a three-phase modeling of human expertise

* Joint appointment with the Research Institute for Informatics, 71316, Bd.Miciurin 8-10, Bucharest 1, Romania

An expert system may be viewed as an explicit model of a human expert's competence and performance (Morik, 1989). We distinguish three phases in the development of such a model. The first one consists of defining a suitable framework for the model, in terms of a knowledge representation formalism and an associated problem solving method. The second phase consists of defining a preliminary model that describes the basic concepts of the expertise domain. The last phase consists of incrementally extending and improving the domain model through learning.

A domain model to be built with the help of NeoDISCIPLE consists of two basic kinds of knowledge. The first one is a hierarchical semantic network, describing explicitly the object concepts in the world, together with their properties and relationships. These object concepts are hierarchically organized according to the "more-general-than" (or "isa") relationship which states that all the instances of a concept C are also instances of any concept that is more general than C. The second kind of knowledge consists of general rules. The meaning of these rules depends of the application domain. They may be inference rules for inferring new properties and relations of objects from other properties and relations, general problem solving rules as, for instance, rules that indicate the decomposition of complex problems into simpler subproblems, or even action models that describe the actions that could be performed by an agent (for instance, a robot), in terms of their preconditions, effects and involved objects. The important feature of these rules is that they refer to the objects and relations from the hierarchical semantic network, and their generality depends of the generality of the involved object concepts.

To build a model of an application domain one has first to define the problem solving method and the corresponding types of the problem solving rules. These, together with the hierarchical semantic network of object concepts, will represent the framework of the domain model.

Next one has to define a preliminary domain model. The goal of this phase is to extract from a human expert whatever knowledge he/she may describe easily and correctly. In general, this knowledge will consist of incomplete descriptions of some basic object concepts from the expertise domain, object concepts that define the initial hierarchical semantic network.

The third phase, which is actually supported by NeoDISCIPLE, consists of incrementally extending and improving the domain model through learning from examples provided by the human expert. During this phase, the expert shows NeoDISCIPLE new facts or specific problem solving episodes (represented by problems and their solutions). From each such input, NeoDISCIPLE may learn an inference rule or a problem solving rule that is a generalization of the input. As a side effect of this learning process, NeoDISCIPLE may develop the hierarchical semantic network by defining new properties, new relationships or even new object concepts.

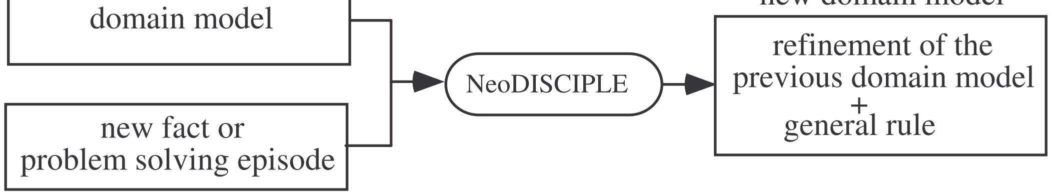

domain model
new fact or problem solving episode
NeoDISCIPLE
new domain model
refinement of the previous domain model + general rule

Figure 1: Incremental development of the domain model

2.2 Understanding-based knowledge extension

The initial model provided by the human expert allows NeoDISCIPLE to react to new inputs with the goal of developing and updating the model so as to consistently integrate them. The general strategy is to try to "understand" (explain to itself) the input in terms of the current domain model. By this we mean that NeoDISCIPLE will try to build a plausible proof which demonstrates that the input is a consequence of the knowledge it already has. For instance, if the input is a new fact then the system will try to prove that this fact derives from other facts which are explicitly represented into the domain model. Or, if the input is an example of a problem solving episode, then the system will try to prove that the episode is correct, by using the facts from the domain model. If successful, the understanding process will determine the facts from the domain model from which the input can be derived. These facts represent the explanation of the input (Mitchell et al., 1986; Tecuci and Kodratoff, 1990).

This "understanding" process depends of the inferential capabilities of the system with respect to the input. We distinguish between three types of such capabilities:

• *Poor knowledge about the input*

The system has no knowledge to build any plausible proof of the input. In such a case, NeoDISCIPLE uses heuristics to propose facts as plausible pieces of explanations, to be validated by the user who may himself indicate additional pieces of explanation or even the entire explanation (which becomes new knowledge to be included into the domain model).

• *Incomplete knowledge about the input*

The system has knowledge allowing it to build a plausible (incomplete) proof of the input. In this case, the building of the proof tree requires the abduction of explanatory facts or inference steps which represent new knowledge to be added to the domain model.

• *Complete knowledge about the input*

The system has knowledge allowing it to build a complete deductive proof of the input.

If an explanation of the input is found, then the system will learn from it a general rule (see next section). This rule will allow it to directly derive, not only the input from which it was learned, but also similar knowledge. However, if no explanation is found, then the input is considered to represent entirely new knowledge that is added as such into the current domain model.

As a result of the understanding process, the input will become (explicit or implicit) part of the domain model. Moreover, as a result of rule learning, other similar knowledge (facts or problem solving episodes) will become implicit part of the domain model.

It is hoped that, through successive learning steps, the domain model of the system will evolve from poor (i.e. without inferential capabilities) to incomplete (i.e. with incomplete inferential capabilities), and from incomplete to complete (i.e. with complete inferential capabilities).

2.3 Knowledge acquisition through multistrategy learning

From each received input NeoDISCIPLE is trying to learn a general rule. This may be a general inference rule, if the input is a specific fact, or it may be a general problem solving rule, if the input is a specific problem solving episode. The learning method of the system integrates synergistically a whole range of learning strategies (explanation-based learning, learning by analogy, empirical inductive learning, learning by asking questions and by being told, abduction and conceptual clustering) and consists of the following steps:

• *Understand the input and find an explanation of it*

NeoDISCIPLE starts learning by trying to understand the input. This process depends of the inferential capabilities of the system with respect to the input and may involve induction, deduction and/or analogy. If successful, the understanding process will determine the facts from the domain model from which the input can be derived. These facts represents the explanation of the input.

• *Generalize the found explanation to an analogy criterion*

Next the system generalizes the explanation to an analogy criterion that would allow the recognition of similar explanation structures in the domain model, and the generation of new knowledge pieces analogous with the input. Depending of the system knowledge, the analogy criterion is obtained by an inductive and/or deductive generalization of the explanation.

• *Apply the analogy criterion to the domain model to generate examples analogous with the input*

The instances of the analogy criterion in the current domain model represent explanations similar with the explanation of the input. From each such explanation the system may generate a fact (or problem solving episode) analogous with the input. The generated knowledge is shown to the human expert which has to characterize it as true or false (see Figure 2).

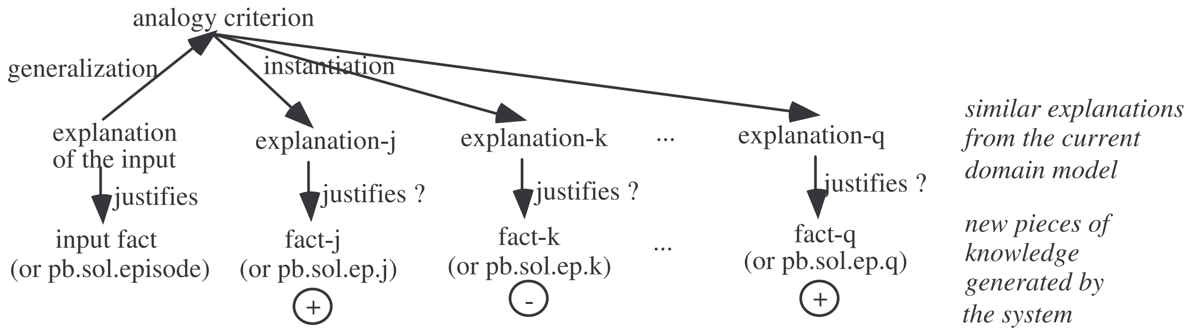


Figure 2: Generation of examples analogous with the input

• *Learn from the generated examples*

The generated facts (or problem solving episodes) that have been characterized as true by the human expert represent positive examples of the rule to be learned. The others represent negative examples. From these examples, NeoDISCIPLE learns a general rule that covers as many of the positive examples as possible and as few of the negative examples as possible.

• *Improve the current domain model as a by product of rule learning*

During rule learning, the current domain model may be improved by adding new properties, relationships or concepts into the hierarchical semantic network and/or by improving some of the inference rules used in the learning process.

2.4 Consistency-driven concept formation and refinement

NeoDISCIPLE starts learning with a preliminary model which usually consists of incomplete descriptions of some basic object concepts that define an initial language for representing and learning new object concepts, facts, rules etc. Because of this incompleteness, the general knowledge pieces learned by NeoDISCIPLE may have exceptions. For instance, a learned rule may cover invalid problem solving episodes. In order to eliminate these exceptions, new concepts have to be defined, or the definitions of the existing concepts have to be refined. For instance, one may eliminate the negative exceptions of a rule by defining a new concept discriminating between the positive examples and the negative exceptions, and by introducing it into the applicability condition of the rule (Wrobel, 1989; Tecuci, 1991). Alternatively, one may refine the definition of a concept with a new feature or relationship shared only by the positive examples of the rule (Tecuci, 1991). In this way, the hierarchical semantic network of object concepts is iteratively developed with the goal of improving the consistency of the learned rules.

2.5 Closed-loop learning

As shown in Figure 1, the knowledge learned from an input become background knowledge which is used in the subsequent learning process, increasing the quality of learning. Therefore, NeoDISCIPLE illustrates a general case of closed-loop learning (Michalski, 1990a).

2.6 Cooperation between the human expert and the learning system

The knowledge acquisition method of NeoDISCIPLE is based on a cooperation between the human expert and the learner which exploits their complementary abilities. That is, each part contributes to the knowledge acquisition process with what he (it) can do better than the other.

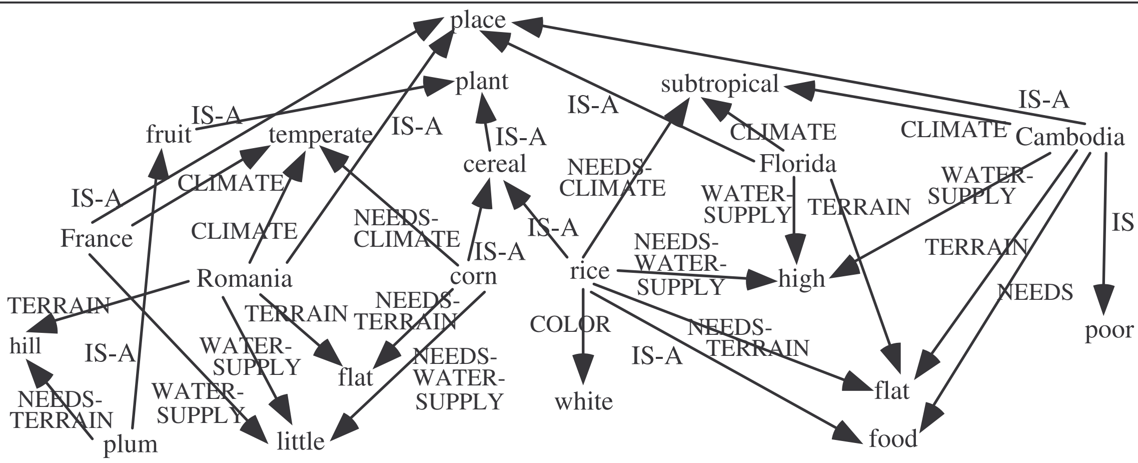
The human expert, for instance, provides an initial imperfect elementary description of his domain. He is particularly good at providing suitable solutions to problems. He may judge if a solution to a problem is good or not, or if a fact is true or false. He is less good at providing an explanation of why a particular solution to a problem is good or not, but can easily accept or reject tentative explanations proposed by the system. What is particularly difficult for the human expert is to provide general pieces of information and to maintain the consistency of the domain model. On the other hand, NeoDISCIPLE suggests justifications of the observed facts or examples of problem solving episodes, generalizes them, and iteratively develops and updates the model of the expertise domain, so that to consistently integrate the learned knowledge.

3 Intuitive illustration of the knowledge acquisition methodology

3.1 Question-answering in geography

We shall illustrate our approach to the automation of knowledge acquisition by considering the building an expert system able to answer questions about geography. A possible framework for the domain model consists of an expert system shell that implements a backward chaining theorem prover. The knowledge base consists of a hierarchical semantic network (describing explicitly properties and relations of the geographical objects) and of inference rules (for inferring new properties and relations). To answer a question of the form "Does (corn GROWS-IN Romania) ?", the system will first look into the semantic network. If the above fact is not explicitly represented, then the system will try to infer it from the explicitly represented facts.

Once the framework has been defined, the human expert has to provide whatever domain knowledge he may easily express. In general, this will consist of incomplete descriptions of some basic geographical objects, represented in the form of a hierarchical semantic network like the one from the top of Figure 3. These object concepts constitute a preliminary domain model used to learn new geographical concepts and inference rules. Although we plan to investigate the automation of definition of this preliminary domain model, the current version of NeoDISCIPLE does not support the human expert more than accepting any model, as incomplete as it may be.



```
R1:  IF
        (y HAS-METEO-COND-FOR x)&(y HAS-TERRAIN-COND-FOR x)      ; if y has meteorological
     THEN ; and terrain conditions for x
        (x GROWS-IN y)                                          ; then x grows in y

R2:  IF
        upper bound
        (x IS-A something)&(y IS-A something)&                  ; the water supply of y
        (t IS-A something)&(u IS-A something)&                  ; is that needed by x,
        (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&            ; and the climate of y
        (y CLIMATE u)&(x NEEDS-CLIMATE u)                       ; is that needed by x

        lower-bound
        (x IS-A fruit)&(y IS-A place)&                          ; the water supply of the place y
        (t IS-A little)&(u IS-A temperate)&                     ; is little, as needed by the fruit x,
        (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&            ; and the climate of y is temperate,
        (y CLIMATE u)&(x NEEDS-CLIMATE u)                       ; as needed by x
     THEN
        (y HAS-METEO-COND-FOR x)                                ; y has meteorological cond for x
     with the positive examples
        (x<-plum, y<-Romania, t<-little, u<-temperate)&(x<-grape, y<-France, t<-little, u<-temperate)

R3:  IF
        upper bound
        (x IS-A something)&(y IS-A something)&(z IS-A something)&     ; the terrain of y
        (y TERRAIN z)&(x NEEDS-TERRAIN z)                   ; is that needed by x

        lower-bound
        (x IS-A fruit)&(y IS-A place)&(z IS-A hill)&                  ; the terrain of a place y is hill,
        (y TERRAIN z)&(x NEEDS-TERRAIN z)                   ; as needed by the fruit y
     THEN
        (y HAS-TERRAIN-COND-FOR x)                                    ; y has terrain conditions for x
     with the positive examples
        (x<-plum, y<-Romania, z<-hill)&(x<-grape,y<-France,z<-hill)
```

Figure 3: A sample domain model

The preliminary domain model is extended and improved by NeoDISCIPLE through successive interactions with the human expert. During these interactions, the human expert provides new geographical facts, and the system learns inference rules and develops the hierarchical semantic network. Some of the rules may be incompletely learned as, for instance, the rules R2 and R3 in Figure 3. Instead of an exact condition they specify a version space (Mitchell, 1978) for the condition, represented by a conjunctive expression that is more general than the exact condition (the upper bound), and a conjunctive expression that is less general than the exact condition (the lower bound).

For instance, from the input fact

```
                              (rice GROWS-IN Cambodia)                                   (1)
```

the system learned the following inference rule:

```
IF                                                                                           (2)
    (x IS-A plant)&(y IS-A place)&(z IS-A terrain-type)&          ; If
    (t IS-A quantity)&(u IS-A climate-type)&(v IS-A soil-type)&   ; the water supply of the place y
    (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&                  ; is that needed by the plant x, and
    (y CLIMATE u)&(x NEEDS-CLIMATE u)&                            ; the climate of y is that needed by x, and
    (y TERRAIN z)&(x NEEDS-TERRAIN z)&                            ; the terrain of y is that needed by x, and
    (y SOIL v)&(x NEEDS-SOIL v)                                   ; the soil of y is that needed by y
THEN                                                          ; then
    (x GROWS-IN y)                                                ; x grows in y
```

Such a rule allows the system to derive the original input fact, but also other related facts as, for instance, "(rice GROWS-IN Tunisia)" or "(corn GROWS-IN Romania)".

As a by product of rule learning, the system has also learned new relevant geographical relationships, like "SOIL" and "NEEDS-SOIL", as well as new basic geographical facts like "(Cambodia SOIL fertile)" and "(rice NEEDS-SOIL fertile)".

It has also improved the rule R3 in Figure 3 by adding necessary predicates to both bounds of the condition:

```
R3': IF                                                                                       (3)
        upper bound
        (x IS-A something)&(y IS-A something)&(z IS-A something)&   ; the terrain of y
        (y TERRAIN z)&(x NEEDS-TERRAIN z)&                          ; is that needed by x and
        (v IS-A something)&(y SOIL v)&(x NEEDS-SOIL v)              ; the soil of y is that needed by x

        lower-bound
        (x IS-A plant)&(y IS-A place)&(z IS-A terrain-type)&        ; the terrain of a place y is the terrain-
        (y TERRAIN z)&(x NEEDS-TERRAIN z)&                          ; type needed by the plant y, and the
        (v IS-A soil-type)&(y SOIL v)&(x NEEDS-SOIL v)              ; soil of y is the soil-type needed by x
     THEN
        (y HAS-TERRAIN-COND-FOR x)                                  ; y has terrain conditions for x
     with the positive examples
        (x<-plum, y<-Romania, z<-hill, v<-normal)&(x<-grape,y<-France,z<-hill, v<-normal)&
        (x<-rice, y<-Cambodia, z<-flat, v<-fertile)
     with the negative example
        (x<-rice, y<-Florida, z<-flat, v<-normal)
```

The next section illustrates how this learning process took place.

3.2 Illustration of the rule learning method

Once the system has received the input fact "(rice GROWS-IN Cambodia)" it tried to understand it by showing that it is a consequence of the knowledge explicitly represented into the domain model. This "understanding" process depends of the inferential capabilities of the system with respect to the input fact: poor, incomplete, or complete. We shall illustrate all these three cases.

Let us first suppose that the current domain model is the one from Figure 3. In this case the system has incomplete knowledge about the input "(rice GROWS-IN Cambodia)" because it is able to build the plausible proof tree from the top of Figure 4. We called this tree "plausible" because it was built by using the upper bound conditions of the rules R2 and R3. Therefore, the inferences made are only plausible and should be validated by the expert. In general, to build a plausible proof tree, the system may need to abduce new facts or even new inference steps. The leaves of the plausible tree represent the facts from the knowledge base that imply the input, i.e. the explanation of the input. The next step is to generalize the explanation to an analogy criterion. To this purpose, the system generalizes the plausible proof tree as much as allowed by the inference rules used. The generalization technique is similar to that of (Mooney and Benett, 1986). NeoDISCIPLE first replaces each instantiated inference rule with its general pattern (by using the upper bound of the rule's condition if the rule has not an exact condition), and then unifies these patterns. In this way the system builds the generalized tree from the bottom of Figure 4, the leaves of which represent the analogy criterion.

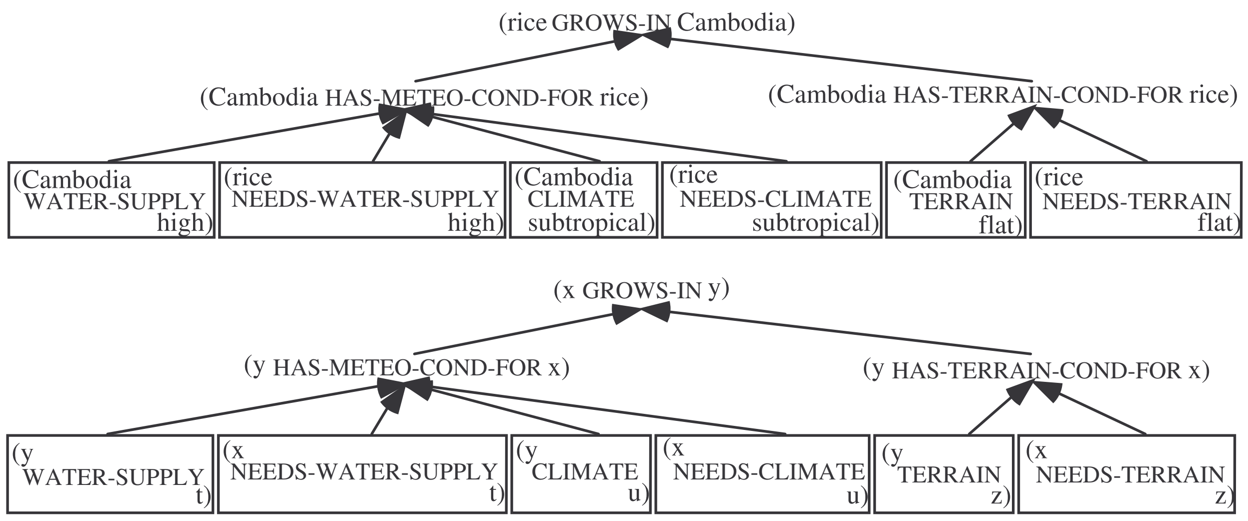

(rice GROWS-IN Cambodia)
(Cambodia HAS-METEO-COND-FOR rice)
(Cambodia HAS-TERRAIN-COND-FOR rice)
(Cambodia WATER-SUPPLY high)
(rice NEEDS-WATER-SUPPLY high)
(Cambodia CLIMATE subtropical)
(rice NEEDS-CLIMATE subtropical)
(Cambodia TERRAIN flat)
(rice NEEDS-TERRAIN flat)
(x GROWS-IN y)
(y HAS-METEO-COND-FOR x)
(y HAS-TERRAIN-COND-FOR x)
(y WATER-SUPPLY t)
(x NEEDS-WATER-SUPPLY t)
(y CLIMATE u)
(x NEEDS-CLIMATE u)
(y TERRAIN z)
(x NEEDS-TERRAIN z)

Figure 4: Finding an explanation of the input and
generalizing it to an analogy criterion, in the case of incomplete knowledge

Let us now suppose that the current domain model consists of only the semantic network from the top of Figure 3. This represents an example of poor knowledge about the input "(rice GROWS-IN Cambodia)" because this knowledge is not enough for inferring anything about the validity of the input. However, the system makes the hypothesis that the input fact is a *direct* consequence of other facts that are explicitly represented into the semantic network. It therefore uses heuristics to select such facts, and to propose them as partial explanations to be validated by the user, who may himself indicate other pieces of explanations. One used heuristic is to propose as plausible explanations the relations between the objects from the input (rice and Cambodia):

Are the following relations explanations for '(rice GROWS-IN Cambodia)':
(rice NEEDS-TERRAIN flat) & (Cambodia TERRAIN flat) ?*Yes*
(rice IS-A food) & (Cambodia NEEDS food) ? *No*
(rice NEEDS-WATER-SUPPLY high) & (Cambodia WATER-SUPPLY high) ? *Yes*
(rice NEEDS-CLIMATE subtropical) & (Cambodia CLIMATE subtropical) ?*Yes*

All the pieces of explanations marked by a user's yes form the explanation of the input. In the case of poor knowledge, this explanation is inductively generalized to an analogy criterion by simply transforming the objects into variables. As one may notice, the system has found the same explanation and the same analogy criterion as in the case of incomplete knowledge.

Let us finally suppose that the semantic network in Figure 3 has been augmented with the relations "(rice NEEDS-SOIL fertile)" and "(Cambodia SOIL fertile)", and that the rules R2 and R3 have been completely learned. The resulting domain model is "complete" with respect to the input fact "(rice GROWS-IN Cambodia)" because it allows the system to build a deductive proof of the input. In such a case, the learning method reduces to pure explanation-based learning (Mitchell et al., 1986; DeJong and Mooney, 1986). Indeed, NeoDISCIPLE builds a tree similar to the one from the top of Figure 4, except that each inference step is a deduction, and the tree is a logical proof. Then, by using the general form of the inference rules R1, R2, and R3, it builds a generalized proof tree, similar to the one from the bottom of Figure 4. Because this generalized tree is a logical proof, its leaves imply the input and represent the exact condition of the rule to be learned. Therefore NeoDISCIPLE learned at once the rule shown in (2). The other steps of the method are no longer necessary.

In the following, we shall show how learning proceeds when the system does not have "complete" knowledge about the input fact "(rice GROWS-IN Cambodia)". NeoDISCIPLE has found an explanation of the input and has generalized it to an analogy criterion (see Figure 5). The instances of the analogy criterion in the domain model represent explanations similar with the explanation of the input. Each such explanation may account for a fact analogous with the input one. Because analogy is a weak inference, these facts could be, however, true or false. The goal of the system is to learn a general rule that covers the true facts and rejects the false ones.

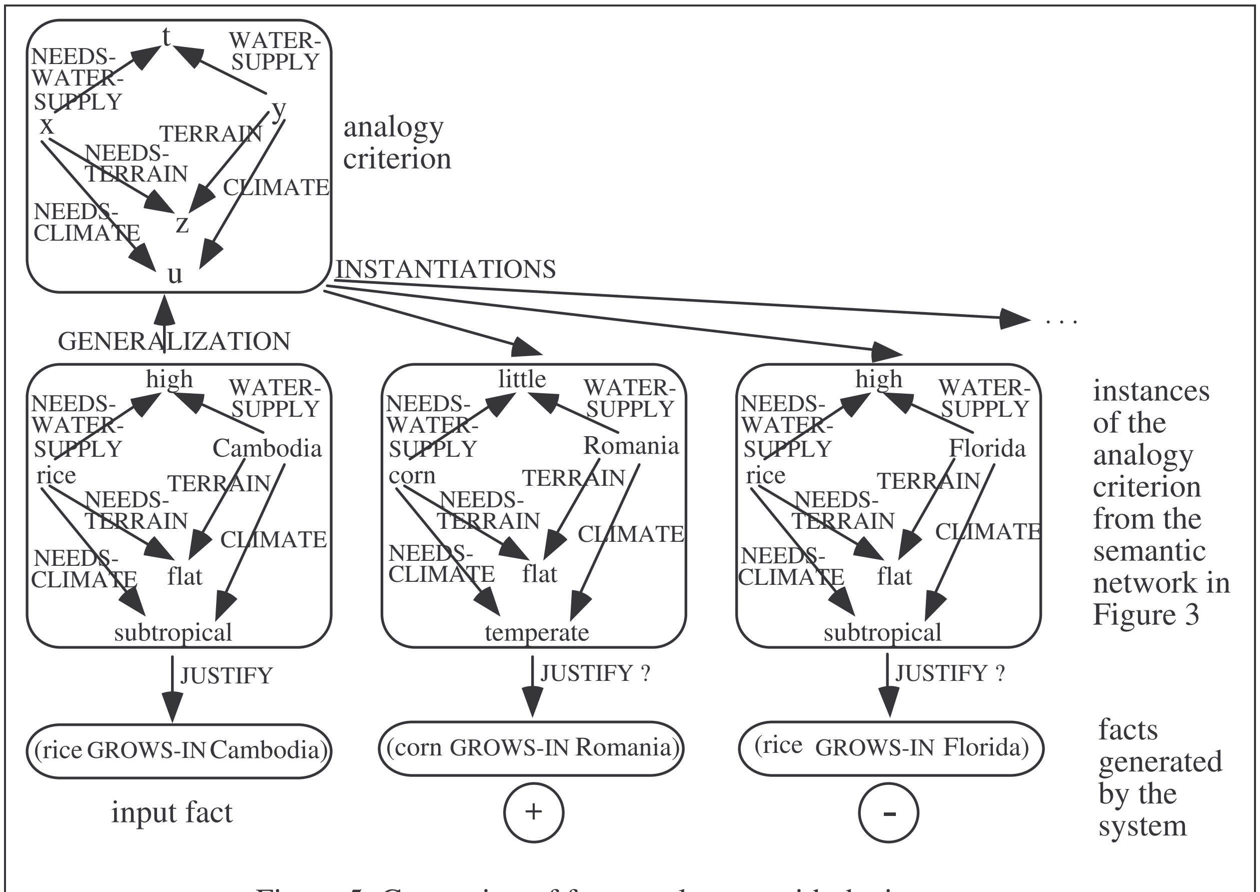


Figure 5: Generation of facts analogous with the input one

First of all, from the input fact, its explanation and the analogy criterion, the system builds an initial version space for the rule to be learned. This version space is shown in Figure 6. Its lower bound covers only the input fact and its upper bound covers all the facts that may be generated (Tecuci and Kodratoff, 1990).

Then, the system applies the analogy criterion to the semantic network in Figure 3 and generates, one after the other, the facts from Figure 5, asking the expert to validate them:

Does (corn GROWS-IN Romania)? Yes

The validated facts represent positive examples of the rule to be learned and the rejected ones represent negative examples.

```
IF
  upper-bound(analogy criterion)
  (x IS-A something)&(y IS-A something)&(z IS-A something)&(t IS-A something)&(u IS-A something)&
  (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&(y CLIMATE u)&(x NEEDS-CLIMATE u)&
  (y TERRAIN z)&(x NEEDS-TERRAIN z)
  lower-bound(explanation)
  (x IS-A rice)&(y IS-A Cambodia)&(z IS-A flat)&(t IS-A high)&(u IS-A subtropical)&
  (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&(y CLIMATE u)&(x NEEDS-CLIMATE u)&
  (y TERRAIN z)&(x NEEDS-TERRAIN z)
THEN
  (x GROWS-IN y)
with the positive example
  (x<-rice, y<-Cambodia, z<-flat, t<-high, u<-subtropical)
```

Figure 6: The initial version space for the rule to be learned

The generated facts from Figure 5 are used to shrink the version space from Figure 6, by using the method presented in (Tecuci and Kodratoff, 1990).

For each positive example as, for instance, "(corn GROWS-IN Romania)", the system generalizes the lower bound of the version space (see Figure 6) to the most specific generalization that covers the explanation of the example (see (4) below), and is less general then the upper bound.

Explanation i (4)

(x IS-A corn)&(y IS-A Romania)&(z IS-A flat)&(t IS-A little)&(u IS-A temperate)
(y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&(y CLIMATE u)&(x NEEDS-CLIMATE u)&
(y TERRAIN z)&(x NEEDS-TERRAIN z)

For each negative example as, for instance, "(rice GROWS-IN Florida)", the system and the expert determines the explanation of the failure:

Failure explanation j

Not(rice GROWS-IN Florida) because (Florida SOIL normal)&(rice NEEDS-SOIL fertile) (5)

and the corresponding piece of explanation of the initial input:

Explanation j

(rice GROWS-IN Cambodia) because (Cambodia SOIL fertile)&(rice NEEDS-SOIL fertile)
(6)

This last explanation is added to both bounds of the version space of the rule to be learned as if it were found during the initial understanding of the input fact.

As a consequence of the above two examples, the version space in Figure 6 becomes:

```
IF
    upper-bound
    (x IS-A something)&(y IS-A something)&(z IS-A something)&(t IS-A something)&(u IS-A something)&
    (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&(y CLIMATE u)&(x NEEDS-CLIMATE u)&
    (y TERRAIN z)&(x NEEDS-TERRAIN z)&(v IS-A something)&(y SOIL v)&(x NEEDS-SOIL v)
    lower-bound
    (x IS-A cereal)&(y IS-A place)&(z IS-A flat)&(t IS-A quantity)&(u IS-A climate-type)&
    (y WATER-SUPPLY t)&(x NEEDS-WATER-SUPPLY t)&(y CLIMATE u)&(x NEEDS-CLIMATE u)&
    (y TERRAIN z)&(x NEEDS-TERRAIN z)&(v IS-A soil-type)&(y SOIL v)&(x NEEDS-SOIL v)
THEN
    (x GROWS-IN y)
with the positive examples
    (x<-rice, y<-Cambodia, z<-flat, t<-high, u<-subtropical v<- fertile)
    (x<-corn,y<-Romania,z<-flat,t<-little,u<-temperate,v<-normal)
with the negative example
    (x<-rice, y<-Florida, z<-flat,t<-high, u<-subtropical, v<-normal)
```

Figure 7: The new version space of the rule to be learned

The learning process decreases the distance between the two bounds of the version space. It continues until the bounds become identical or all the generated examples have been used.

3.3 Improving the current domain model

As a result of learning from the input "(rice GROWS-IN Cambodia)", the domain model is developed in several respects by:

- the addition of the learned rule;
- the addition of new facts or even of new concepts, in the semantic network;
- the improvement of the rules used in the "understanding" of the input.

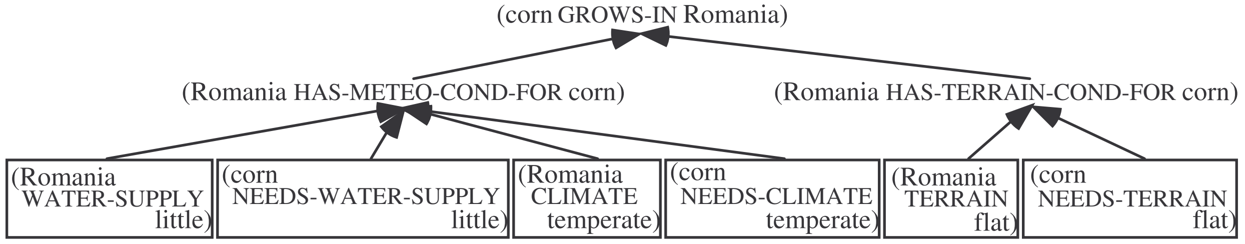
The important outcome of this learning process is that it may change the nature of the system's knowledge, with respect to other inputs, from poor to incomplete, or even to complete.

For instance, **Failure explanation j** and **Explanation j** (see (5) and (6) above), are added to the semantic network as new basic geographical knowledge.

Also, forcing the new lower bound of the rule in Figure 7 (which now includes **Explanation j**) to cover the previously encountered positive example "(corn GROWS-IN Romania)", the system acquired new basic knowledge about the soil of Romania and the soil needed by corn:

(Romania SOIL normal) & (corn NEEDS-SOIL normal)

When the system had incomplete knowledge about the input it also improved the rules used in the understanding process. The guidance for such an improvement is provided by the generalized plausible proof from the bottom of Figure 4. Indeed, to each new fact generated by the system corresponds an instance of this tree, as shown in Figure 8.


(corn GROWS-IN Romania)
(Romania HAS-METEO-COND-FOR corn)
(Romania HAS-TERRAIN-COND-FOR corn)
(Romania WATER-SUPPLY little)
(corn NEEDS-WATER-SUPPLY little)
(Romania CLIMATE temperate)
(corn NEEDS-CLIMATE temperate)
(Romania TERRAIN flat)
(corn NEEDS-TERRAIN flat)

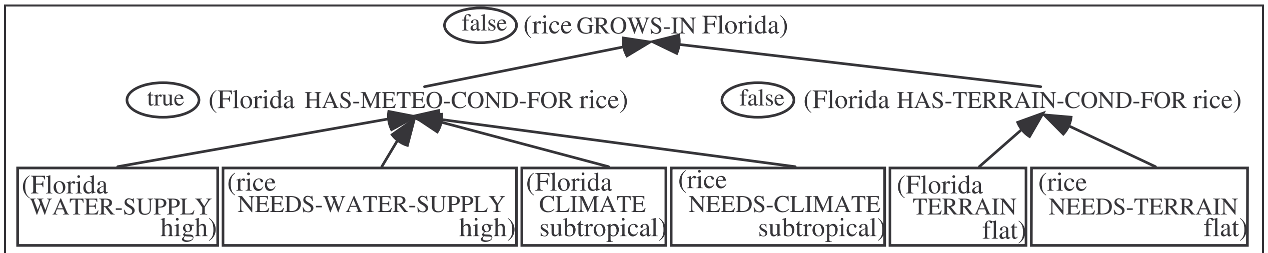

false (rice GROWS-IN Florida)
true (Florida HAS-METEO-COND-FOR rice)
false (Florida HAS-TERRAIN-COND-FOR rice)
(Florida WATER-SUPPLY high)
(rice NEEDS-WATER-SUPPLY high)
(Florida CLIMATE subtropical)
(rice NEEDS-CLIMATE subtropical)
(Florida TERRAIN flat)
(rice NEEDS-TERRAIN flat)

Figure 8: Instances of the generalized tree in Figure 4, corresponding to the true fact "(corn GROWS-IN Romania)" and to the false fact "(rice GROWS-IN Florida)"

The tree corresponding to the true fact "(corn GROWS-IN Romania)" is a correct proof tree that shows new positive instances of the rules R2 and R3 in Figure 3. Therefore, the lower bounds of these rules are generalized to cover the corresponding instances.

The tree corresponding to the false fact "(rice GROWS-IN Florida)" is a wrong proof tree (the leaf predicates are true but the top predicate is not). This means that some of the inferences made are incorrect. To detect them, the system and the user follow the proof tree from bottom up. If the user states that the consequent of a certain inference step is not true, then the corresponding inference rule may be the faulty one. In this case, the wrong inference step is:

(Florida TERRAIN flat)&(rice NEEDS-TERRAIN flat) --> (Florida HAS-TERRAIN-COND-FOR rice)

The detection of this wrong inference step allowed the user and the system to discover the failure explanation in (5) above. It has also provided a negative example for the rule R3 in Figure 3. As a consequence, the system transformed the rule R3 from Figure 3 into the rule R3' (see (3)). It is important to notice that the positive examples and the negative examples of the rules R2 and R3 are used to improve these rules in the same way as the positive and the negative examples of the rule to be learned.

4 Discussion and conclusions

NeoDISCIPLE is implemented in Common Lisp and runs on Macintosh. Until now it was used to build small knowledge bases for the following types of expertise domains: planning (in robotics), planning combined with design (in manufacturing), prediction (in chemistry), and question-answering (in geography). From these experiments, we have concluded that it is not very difficult to build a small knowledge base with NeoDISCIPLE.

The dialogue with the user is based on "intelligent", specific, and easy to answer questions. Also, NeoDISCIPLE minimizes the number of questions asked by carefully generating only those facts or problem solving episodes that are most likely to advance the learning process. In the experiments performed, NeoDISCIPLE usually needed to generate less than 10 examples (see Figure 2) for learning a rule. However, the knowledge base may sometimes allow the generation of thousands of such examples. If, in a certain critical application, all these examples need to be tested, then this may require a lot of time from the human expert. Therefore, new methods have to be devised to test the examples independently of the expert (for instance, by comparing them with a database of cases), or to select for testing only the examples that have the highest likelihood of contradicting the rule being learned.

A basic source of knowledge for learning is the hierarchical semantic network which provides the generalization language. Therefore, an application domain for which one cannot define a "rich enough" semantic network is not suitable for NeoDISCIPLE.

The integrated learning method of NeoDISCIPLE outperforms any of the constituent single-strategy methods in that it is able to learn in situations in which they were insufficient. However, NeoDISCIPLE still suffers from the basic limitation of the learning systems: if the bias built into the system is incorrect, the system will fail to learn properly. In the current version of NeoDISCIPLE, the initial semantic network (which contributes significantly to the system's learning bias) is supposed to be incomplete but correct. During learning, the definition of the object concepts may be refined and even new concepts may be defined (Tecuci, 1991). While this may improve the initial bias, it will not modify it drastically. A better way to surmount this limitation is to perform not only additions to the semantic network, but also deletions. We therefore plan to develop the learning method of NeoDISCIPLE so that to start with an imperfect domain model (which is not only incomplete, but also partially incorrect), and to gradually improve it.

The presented methodology divides the process of building an expert system (viewed as a model of an expertise domain) into three phases: defining a framework for the domain model, providing a preliminary domain model, and incrementally improving the domain model. Only the third phase is automated by the present version of NeoDISCIPLE. In the future, we plan to evolve NeoDISCIPLE into a system that will assist an expert user during all the three phases.

For the automation of the first phase, we plan to identify suitable expert system shells (i.e. systems the knowledge base of which could be learned by NeoDISCIPLE) and to couple each of them with a customized version of NeoDISCIPLE.

The definition of the preliminary domain model could be automated by using an approach similar to that of the BLIP and MOBAL systems (Morik, 1989; Wrobel, 1989). These systems are able to build such an initial model from user provided facts.

NeoDISCIPLE itself could be further improved by improving the existent learning strategies and by adding new ones, in order to increase the learning capabilities of the system. This research direction is closely related to an on going effort at the Center for Artificial Intelligence to define a unifying theory of machine learning and a general multistrategy task-adaptive learning methodology based on this theory (Michalski, 1990a,b; Tecuci and Michalski, 1991).

Acknowledgements

The author thanks Yves Kodratoff and Ryszard Michalski for interesting and useful discussions that influenced this work, and an anonymous referee for useful comments. This research was done in the Artificial Intelligence Center of George Mason University. Research activities of the Center are supported in part by the Defence Advanced Research Projects Agency under grant No. N00014-87-K-0874, administrated by the Office of Naval Research, and in part by the Office of Naval Research under grants No. N00014-88-K-0226 and No. N00014-88-K-0397.

References

Bareiss, R., Porter B., and Wier C., PROTOS: An Exemplar-based Learning Apprentice, in Kodratoff Y., and Michalski R.S. (eds), *Machine Learning: An Artificial Intelligence Approach,* vol. III, Morgan Kaufmann, 1990.

Boose, J.H., Gaines, B.R., and Ganascia, J.G.(eds), *Proceedings of the Third European Workshop on Knowledge Acquisition for Knowledge-based Systems,* Paris, July, 1989.

DeJong G., and Mooney R., Explanation-Based Learning: An Alternative View, in *Machine Learning,* vol.1, no. 2, pp. 145-176, 1986.

Klinker G., KNACK: Sample-Driven Knowledge Acquisition for Reporting Systems, in S. Marcus (ed), *Automating Knowledge Acquisition for Expert Systems*, Kluwer Publishers, 1988.

Kodratoff, Y., and Michalski, R.S. (eds), *Machine Learning: An Artificial Intelligence Approach,* Morgan Kaufmann, vol.III, 1990.

Marcus S.(ed), *Automating Knowledge Acquisition for Expert Systems*, Kluwer, 1988.

Michalski R. S., Toward a Unified Theory of Learning: Multistrategy Task-adaptive Learning, *Research Report MLI 90-1,* Artificial Intelligence Center, George Mason University, 1990a.

Michalski R. S., A Methodological Framework for Multistrategy Task-adaptive Learning, in Ras Z., Zemankova M. (eds), *Methodologies for Intelligent Systems,* North Holland, 1990b.

Mitchell, T. M., Version Spaces: An Approach to Concept Learning, *Doctoral dissertation*, Stanford University, 1978.

Mitchell T.M., Keller R.M., and Kedar-Cabelli S.T., Explanation-Based Generalization: A Unifying View, *Machine Learning,* vol.1, no.1, pp. 47-80, 1986.

Mooney R., Bennet S., A Domain Independent Explanation Based Generalizer, in *Proceedings AAAI-86*, Philadelphia, 1986.

Morik K., Sloppy modeling, in Morik K. (ed), *Knowledge Representation and Organization in Machine Learning*, Springer Verlag, 1989.

Tecuci G., DISCIPLE: A Theory, Methodology, and System for Learning Expert Knowledge, *Ph.D. Thesis,* University of Paris-South, 1988.

Tecuci, G. and Kodratoff Y., Apprenticeship Learning in Imperfect Theory Domains, in Kodratoff Y., and Michalski R.S. (eds), *Machine Learning,* vol. III, Morgan Kaufmann, 1990.

Tecuci G., A Multistrategy Learning Approach to Domain Modeling and Knowledge Acquisition, in *Proc. of the European Conference on Machine Learning*, Porto, Springer-Verlag, 1991.

Tecuci, G. and Michalski R., A Method for Multistrategy Task-adaptive Learning Based on Plausible Justifications, *Proceedings of the Eight International Workshop on Machine Learning,* Chicago, Morgan Kaufmann, 1991.

Wilkins, D.C., Clancey, W.J., and Buchanan, B.G., *An Overview of the Odysseus Learning Apprentice*, Kluwer Academic Press, 1986.

Wrobel S., Demand-Driven Concept Formation, in Morik K.(ed), *Knowledge Representation and Organization in Machine Learning*, Springer Verlag, 1989.