# How to Specialize by Theory Refinement

**Claire Nedellec**

*LRI, Université Paris-Sud, Bt. 490 F-95405 Orsay Cédex France email : cn@lri.lri.fr*
*ISoft Chemin du Moulon F-91190 Gif sur Yvette France*

**Abstract.*** Over-generalization is a well-known problem in empirical learning. Incremental and cautious generalization may limit it, but the concept description language may be incomplete or incorrect. In this case, over-generalization may indicate that the language must be improved. This paper presents an interactive incremental learning method with a "smallest generalization steps" strategy such that whenever a learned concept is over-general, the method specializes it and efficiently helps a user to identify the insufficiencies of the concept language and in improving it if necessary.

## 1 Introduction

Learning concept definitions empirically requires the ability to control the degree of generalization. As the aim of ML is to increase the prediction capability, the learned concept usually verifies additional examples. If some of the covered examples are negative examples, the learned concept is over general.

The method described in this paper uses incremental and cautious generalization as a way to reduce the risk of uncontrolled over-generalization. However, over generalization may nevertheless occur. The training examples may be incomplete and not specific enough and/or the concept description language may be insufficient, so that there is no way to conjunctively express a concept definition that covers all the positive examples and no negative one. As a matter of fact, it is a critical aspect of knowledge acquisition (KA) to design a description language free from such insufficiencies [Shapiro, 83].

In such cases, the learner has no means of detecting the problem, since the learned concept is consistent with everything the learner knows (e.g. the examples and the concept description language). This classical problem in knowledge refinement accepts two main types of solutions. What is learned may be tested and improved through many examples with learning in the limit methods such as CLINT [Bruynooghe et al., 89] or an expert of the domain can directly validate and correct the learned concepts [Sammut & Banerji 86]. In this paper we are interested in the latter approach. On the one hand, it allows to reduce the number of examples required and to simplify the validation phase. But on the other hand the evaluation of the learned concepts may be less reliable than the example evaluation, depending on the domain and the task. This evaluation problem may be partially overcome by KA techniques. It would be also a difficult task for an expert to *correct* the causes of over generalization without further information than the learned concept and the examples. We propose a method to tackle this problem of evaluation and correction. Its cautious generalization strategy provides the capability to efficiently help an expert whenever an over-generalization occurs by restricting the range of the search for the causes of the over generalization to a delimited sub-part of the vocabulary, and by guiding the expert in the identification of the missing knowledge.

* This work is partially supported by CEC, through the ESPRIT-2 contract and the French MRT through PRC-IA.

## 2 APT

### 2.1 The APT Architecture

This method is supported by implementation and experimentation in the learning module of the APT system [Nédellec, 91]. APT architecture is that of DISCIPLE [Tecuci, 90].

APT uses a domain theory in order to generalize problem-solving rules provided by an expert. It is represented by a semantic network where the hierarchies completely determine the generality relation and the property inheritance between the literals of the theory. The concept definitions and the examples are described in the same language that is restricted to conjunctions of the literals defined in the theory. Consequently, the improvement of the concept description language leads to the refinement and the completion of the theory.

### 2.2 Learning method

Initially the expert provides a specific rule. Its application condition is considered as a positive example. It is completed by explanations generated or acquired from the expert [Nédellec, 92a]. Next, an example set is automatically generated by analogy with the completed example. The initial search space of the target concept definition is limited by a lower bound (noted LB) equal to the expert's example and an upper bound (noted UB) equal

to the same formula where all the predicates are replaced by the most general predicates in the corresponding hierarchies of the theory.

Next, the generated examples are successively submitted to the expert. As in [Mitchell, 1978] each negative example leads to a specialization of the UB so that the resulting UB is the most general one that does not cover the negative examples. Each accepted example leads to a generalization of the LB so that the resulting LB is the most specific one that covers all the accepted examples. No dropping rule is applied, but only climbing in the hierarchies of predicates and the problem of the matching choice between literals is supposed being solved. Therefore, the search space converges incrementally on the target concept.

At the end of the generalization phase, the learned concept is evaluated by the expert. If it is over general, a search for the causes of the over-generalization begins. APT gives the expert the possibility to select the specialization direction by pointing out the too general predicate of the learned concept definition in order to avoid an exhaustive search. Next, it specializes the learned concept by backtracking on the intermediate steps that it has performed in the course of the generalization and submits them to the expert until the expert accepts one. This way, APT identifies the most specific over general LB state (noted MSOG concept) that covers negative examples, when the previous states do not. Notice that the all previous states are over specific, since they do not cover the positive example that has led to the validation of the MSOG concept

APT's aim is now to correct the definition of the MSOG concept to exclude all the negative example from its scope without excluding positive ones. The inconsistent example subset that contain negative examples unduly covered is defined as the subset of examples that are *newly* covered by the MSOG concept and not covered by any previous state. Moreover the vocabulary of the theory that may have to be corrected in order to suppress the over generalization belong to the subpart of the language that is involved in this generalization step. So, the size of this subset is critical : it is obvious that the smaller the inconsistent example subset is, the easier it will be to identify the negative examples to exclude from the concept definition. Reducing the size of the subset means reducing as much as possible the difference between two successive states because the size of this subset directly depends on the difference between two successive states. This may be done by applying the "smallest generalization steps strategy" (SGSS) while generalizing whenever there exists a risk of over generalization .

### 2.3 Cautiously generalizing

The SGSS consists at each step, in generalizing *only one predicate* of the current LB *on one degree* according to the generality relation of the theory, that is replacing a predicate by its "parent" in the corresponding theory hierarchy. As the SGSS determines a partial order of the successive states of the LB, it also determines the processing order of the examples.

Formally, at a stage T, the current LB is a conjunction of literals, the predicates of which are ($P_1$, $P_2$, ... $P_i$, ... $P_n$). At the stage T+1, APT tries to validate one more general state LB' ($P_1$, $P_2$, ... $\mathbf{\Pi_i}$ , ... $P_n$) where $\Pi_i$ is the direct parent of $P_i$ in the corresponding hierarchy of the theory. APT submits one of the generated example ($p_1$, $p_2$, ... $p_i$, ... $p_n$) to the expert. All the example predicates ($p_1$, $p_2$, ... $p_{i-1}$, $p_{i+1}$... $p_n$), except $p_i$ must be less general than the corresponding predicates in the current LB ($P_1$, $P_2$, ... $P_{i-1}$, $P_{i+1}$, ... $P_n$), and the predicate $p_i$ must be less general than $\Pi_i$.

If this example is accepted, it forms a "near success" (by analogy with "near miss") and the consequent generalization of the LB is one of the most elementary ones, since just one predicate of the LB has to be generalized on only one degree to cover it.

If the example is rejected by the expert, this near miss is excluded from the search space by the specialization of the UB as with the INBF algorithm [Smith, Rosenbloom, 90], who shows the efficiency of pruning the search space in that way. Since the guilty predicate $p_i$ in the negative example is completely identified, (this is the only predicate that does not match the current LB), the corresponding predicate $P_i'$ in the UB is specialized the most to the corresponding predicate $P_i$ in the LB, without any risk of over specialization.

Therefore, by the SGSS, the example subset that is newly covered by any LB state and no previous one is as small as possible. The consequences are of three types. First, disjunctive concepts are learned. The search space of the target concepts is represented by an oriented generalization lattice. By applying the SGSS, the search space converges through the lattice on the learned concepts which are represented by the disjunction of the most general nodes of the lattice that have been validated by the system. If some of them are over general, a reevaluation phase is executed for each of them.

Second, as APT has performed as small as possible steps according to its generalization operator, it cannot build new intermediate concepts between the over general MSOG concept and the over specific previous states. So, the over generalization indicates that the theory is incorrect or incomplete or that the training examples where incomplete. The only solution in order to exclude all the negative examples from the inconsistent subset without excluding any positive ones is thus to acquire new knowledge in order to constrain the definition of the MSOG concept. So, APT needs to interact with the expert to acquire the needed knowledge.

Third, the characteristics of the inconsistent set are particularly useful because the examples are semantically close since their predicates have common direct parents in the theory. This permits the application of KA techniques based on the study and the comparison and the opposition of close examples, in order to find the missing constraints and vocabulary allowing to describe separately the negative and positive examples. This allows APT to interact with the expert is in a significantly more user friendly and helpful way than if only studying of abstract generalization steps.

**2.4 Refinement and revision**

At that stage, APT knows only one positive example from the inconsistent example subset, this is the example that had led to the validation of the MSOG concept during the generalization phase. In order to guide the expert in the correction of the MSOG concept APT needs to identify negative examples, then it submits to the expert other examples from the inconsistent subset. To avoid having to study all the examples from this set which may be numerous, the expert has the possibility to select a subset by choosing predicates which seems to be problematic in the list of predicates that APT knows being involved in the over generalization problem.

When negative examples have been identified, APT asks the expert to compare them with the positive example that he has previously accepted. These examples are semantically close since their predicates have the same direct parents in the theory and consequently it is much easier to find what knowledge description would allow to distinguish them. This technique of comparing close examples to elicit knowledge stems from psychology field and is applied with interesting results in KA .

APT is able to propose to the expert relevant means to use this elicited knowledge in order to correct the MSOG concept by exploiting the subpart of the vocabulary that is involved in the over generalization problem, the positive examples and the negative examples classified by the expert.

We present here the tools which seems to be the most significant and powerful. They are detailed through an example in [Nédellec, 92b]. APT gives to the expert the possibility to complete the theory if the available vocabulary is insufficient and/or to add new constraints to the MSOG concept. So, some tools use the examples that APT knows in order to modify the theory when other tools allow to specialize the MSOG concept and to modify the theory if needed, without using the examples. APT reacts to each modification by indicating if they are sufficient or not in order to exclude the negative examples from the MSOG concept scope. Let us suppose that,

- the *expert concept* is $e_1(x)$, $e_2(y)$, $e_3(x, y)$
- the *MSOG concept* is $\Pi_1(x)$, $\Pi_2(y)$, $\Pi_3(x, y)$
- the *previous concept* is $P_1(x)$, $\Pi_2(y)$, $\Pi_3(x, y)$
- the *positive example* $p_1(x)$, $p_2(y)$, $p_3(x, y)$

has led to the generalization into MSOG

- The following examples from the inconsistent set have been classified by the expert

*Positive examples* :

$Ex_1$ : $P_1,^1(x)$ , $P_2,^1(y)$ , $P_3,^1(x, y)$

....

$Ex_i$ : $P_1,^i(x)$ , $P_2,^i(y)$ , $P_3,^i(x, y)$

*Negative examples* :

$Ex_{i+1}$ : $P_1,^{i+1}(x)$ $P_2,^{i+1}(y)$ $P_3,^{i+1}(x, y)$

....

$Ex_n$ : $P_1,^n(x)$ , $P_2,^n(y)$ , $P_3,^n(x, y)$

*a. Modifying the Domain Theory*

We present four tools improving the theory so that positive examples may be described separately from negative ones by using the learning context. The first step that APT performs consists in listing separately the "positive" predicates that occur in positive examples and the "negative" predicates that occur in negative examples from the inconsistent set. The aim while modifying the theory is to find a way to separate the positive predicates and the negative predicates that define the same variable, such as separating the positive predicates $P_{1,}^{1}(x)$ ... $P_{1,}^{i}(x)$ and the negative predicates $P_{1,}^{i+1}(x)$ ... $P_{1,}^{n}(x)$ defining the same variable x in the examples.

These group of predicates belong to the same hierarchy. They are less general than the predicate defining the same variable in the MSOG concept and more general than the predicate defining the variable in the expert example (Figure 1).

• First, for each variable, APT proposes to add intermediate predicates in the theory to gather in two different clusters the positive predicates and the negative predicates. Suppose that the expert adds the predicates $q_{+},^1$ and $q_{-},^1$ as sons of $\Pi_1$ such as in figure 1. Then, if the expert adds also the constraint $q_{+},^1(x)$ to the MSOG concept, this is sufficient to exclude all the negative examples while covering all the positive ones.

• Second, APT proposes to move the cluster of positive predicates or the cluster of negative predicates along their hierarchy. For instance, if $q_{-},^1$ with its "negative" sons $P_{1,}^{i+1}(x)$ ... $P_{1,}^{n}(x)$ is attached to $G_1$ instead of $\Pi_1$, it is not necessary to change the MSOG concept for the negative examples to be excluded, because $G_1$ is more general than $\Pi_1$.

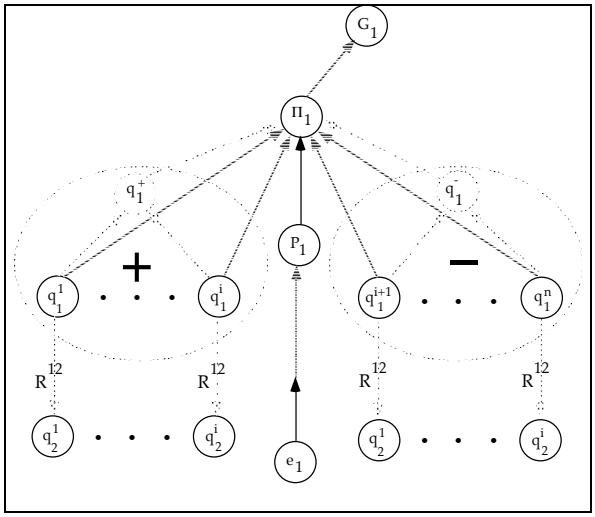


**Figure 1:** Part of the theory

• Third, APT proposes to add to the theory new binary predicates to link together positive predicates separately from negative ones. For instance, it proposes to link each of the positive predicates $P_{1,}^{1}(x)$ ... $P_{1,}^{i}(x)$ that define x in the positive examples to the corresponding positive predicate $P_{2,}^{1}(y)$ ... $P_{2,}^{i}(y)$ that define y in the same examples (figure 1). For instance, $R_{+,}^{12}$ links $P_{1,}^{1}$ with $P_{2,}^{1}$, ... $P_{1,}^{i}$ with $P_{2,}^{i}$ in the theory. Next, if the expert adds the constraint $R_{+,}^{12}(x, y)$ to the MSOG concept, that is sufficient to exclude all the negative examples while covering all the positive ones.

• Fourth, APT also proposes to rename positive binary predicates separately from negative ones.

APT verifies if the modifications performed by the expert lead to the exclusion of the negative examples from the scope of the MSOG concept, but it is the role of the expert to provide the knowledge needed by comparing the examples and to use these different tools in order to solve the over generalization problem.

*b. Adding new constraints to the rule*

APT gives also the possibility to add conjunctions of litterals with unary and binary predicates and known and unknown arguments in order to modify precisely the scope of the MSOG concept. APT helps the expert to integrate the unknown litterals in the theory by using the learning context to propose the most probable modifications [Nédellec, 92b].

• First, the expert may add a literal with a binary predicate such as $\Pi_4(x, y)$. If it is already defined in the theory, APT only verifies that the negative examples are no longer covered by the MSOG concept.

If the new binary predicate is unknown, APT looks which predicates in the MSOG concept define the argument of the new predicate, such as here, $\Pi_1$ defines x and $\Pi_2$ defines y. So, instead of directly linking $\Pi_1$ and $\Pi_2$ by $\Pi_4$ in the theory, what would not exclude any negative example, APT proposes to the expert to choose at which level in each hierarchies of $\Pi_1$ and $\Pi_2$ the expert wants to add the relation $\Pi_4$. When the expert has chosen the relevant generality level, APT proposes to the expert to select, at that level, the predicates that he wants to be linked by the relation $\Pi_4$.

The expert has thus not to directly modify the theory but only to choose which seems to him to be the right level of modifications. Notice that by selecting some sons of $\Pi_1$ and $\Pi_2$ to be linked by $\Pi_4$, the expert determines the examples covered by the new definition of the MSOG concept. The predicates that are not linked by $\Pi_4$ are excluded from the scope of the MSOG concept (just as with the third tool described above).

• Second, the expert may add to the MSOG concept a literal with an unary predicate. APT looks if the argument of the new predicate is already defined by a predicate of the MSOG concept. For instance, the argument x of a new literal $\Pi_5(x)$ is already defined by $\Pi_1$. If the new predicate is known, APT only informs the expert whether these modification of the MSOG concept does or not exclude any negative example. It obviously depends of the generality degree of the new predicate.

If the new unary predicate is unknown, APT asks the expert to choose at which level in the hierarchy of the expert wants to add it. When the expert has chosen the relevant generality level, APT proposes to the expert to select the predicates that he wants to be covered by the new predicate at that level. The predicates that are not covered by $\Pi_5$ are excluded from the scope of the MSOG concept such as it is done by the second tool described in the previous section.

If the argument of the new literal is not defined in the MSOG concept, APT cannot propose any mean to add it in the theory and it asks to the expert to do it by himself.

So, APT gives the expert the possibility to modify the theory as a consequence of the modification of the MSOG concept, but it is the role of the expert to provide the needed knowledge and to make the right choice among the propositions that APT makes.

Therefore, by using the SGSS method, APT is able to closely cooperate with the expert and propose relevant modifications on the over general concept and on the theory, and to carry on these modifications under the control of the expert.

It has been applied to real world problems [Nedellec, 92b] for design [Tecuci, 90], for hypertension treatment and for loan analysis [Bento, 91ab].

## 3 Related Work

MARVIN, described in [Sammut and Banerji 86] has some similarities with our work, although it does not address the problem of identifying the causes of an over-generalization nor the problem of correcting them. MARVIN submits examples to an oracle in order to perform elementary generalization steps by applying one rewrite rule at a time. Unlike MARVIN's rules, the rules used by APT to generalize have only one premise because they are represented by a semantic network. One the one hand, the representation by rewrite rules that can have more than one premise is more powerful, but on the other hand, the size of the generalization steps are larger.

The SGSS method used by APT substitutes only *one predicate* for a given predicate in the current hypothesis, while MARVIN substitutes one predicate for all the premises of the chosen rule. The consequences are significant. First, for APT there is no intermediate step between two successive generalization steps. Then, if an example is refused, APT does never need to specialize in the direction of the previous generalization as MARVIN does, but the generalization is definitively stopped in this direction. Moreover, the examples that are newly covered by a generalization step are much less numerous and semantically very close since two successive steps differ by only one predicate. APT is thus able to provide powerful means to identify the cause of the over generalization and to provide adaptive KA tools to acquire new knowledge in order to constrain the over general learned concept. MARVIN does not provide such tools because its search space is too large, it asks the expert to correct the theory by himself.

[Utgoff, 86] shows that the language in which concepts are described is one of the biases that determine how generalizations are performed. As the description language determines the limits of the hypothesis space, extending it to express more concepts means in effect to complete the language. Then, as we do, Utgoff studies how to change an incomplete language, with the aim of expressing concepts that are consistent with example sets. The syntax remains the same.

Utgoff's method named STABB is applied to LEX [Mitchell, 82]. It consists in automatically integrating new symbol in the language so that the learned concepts may be expressed in a conjunctive form.

APT generalizes in a quite similar way but it does not automatically create new symbols. It seems that expert abilities are generally required to modify the language because the needed modifications may be far more important than adding intermediate features at an arbitrary level in hierarchies as STABB does. So by interacting with an expert APT is able do perform more sophisticated changes. Rather than completing the concept language, APT refines it.

## 5 Conclusion

In this paper, we have presented a method implemented into APT to limit over-generalization when empirically learning. ML and KA techniques have been closely integrated into APT in order to identify the insufficiencies of the concept language and to improve it if necessary. So, the method is based on the close copperation between the learning system and an expert of the domain.

**References**


Bento C., Costa E., Fereira J.L., "*APT as a knowledge elicitation tool*", Esprit Project reports, 1991.

Berwick R.C., "Learning from positive-only examples : The subset principles and three cases studies", in *Machine Learning II* , Morgan Kaufmann, 1986.

Bruynooghe M., De Raedt L. & De Schreye d. C. "Explanation based program transformation", *IJCAI-89*, pp. 407-412, 1989.

Mitchell, T. M., "*Version Spaces: An Approach to Concept Learning*", Ph.D. Standford University, 1978.

Nédellec C. "Smallest generalization step strategy", IWML-91 , pp. 529-533, 1991.

Nédellec C. "Knowledge Refinement using Knowledge Acquisition and Machine Learning methods", *AAAI Spring Symposium*, 1992.

Nédellec C. "Knowledge Acquisition and Machine Learning Methods to control Over Genralization", *EKAW-92*.

Sammut, C.A and Banerji, R.B., "Learning Concepts by asking Questions", in *Machine Learning II*, pp 167-192, Morgan Kaufmann, 1986.

Shapiro, E. Y. "*Algorithmic program debugging*", MIT press, Cambridge, MA, 1983.

Smith B. D. and Rosenbloom P. S., "Incremental Non-Backtracking Focusing: A Polynomially Bounded Generalization Algorithm for Version Space.", *AAAI-90*, pp 848-853, 1990.

Tecuci G. & Kodratoff Y., "Apprenticeship Learning in Nonhomogeneous Domain Theories", in *Machine Learning III* , pp 514-552, Kodratoff Y. & Michalski R. (eds), Morgan Kaufmann, 1990.

Utgoff P.E., "Shift of bias for inductive concept learning", in *Machine Learning II*, pp 107-148, Morgan Kaufmann, 1986.