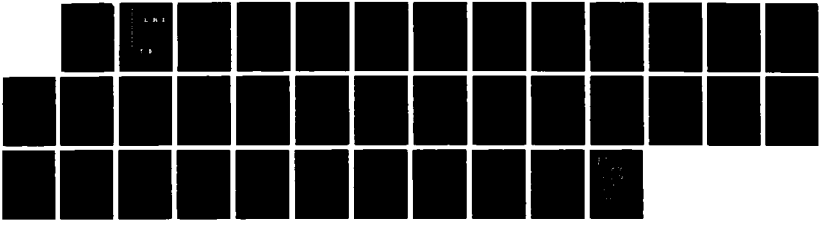
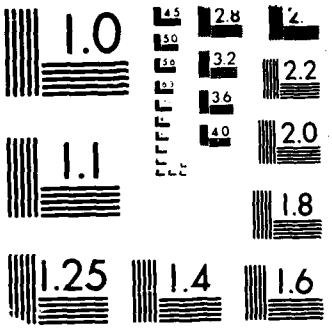
AD-A192 254 IMPROVING THE TOOLS OF SYMBOLIC LEARNING(U) PARIS-11 UNIV ORSAY (FRANCE) LAB DE RECHERCHE EN INFORMATIQUE Y KODRATOFF 01 SEP 87 LRI-321 ARDG(E)-R/D-4624-CC-01 1/1

UNCLASSIFIED DAJA45-85-C-0014 F/G 23/2 NL






MICROCOPY RESOLUTION TEST CHART

RAPPORTS DE RECHERCHE

# L.R.I.

## IMPROVING THE TOOLS OF SYMBOLIC LEARNING

Yves KODRATOFF

*Unité associée au CNRS UA 410:AL KHOWARIZMI*

*Décembre 1986* — *Rapport de Recherche n° 321*





UNIVERSITE DE PARIS-SUD
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bât. 490
91405 ORSAY (FRANCE)

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

ADA192254

# REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188
Exp Date Jun 30, 1986

| | |
|---|---|
| 1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED | 1b RESTRICTIVE MARKINGS: |
| 2a SECURITY CLASSIFICATION AUTHORITY: | 3 DISTRIBUTION/AVAILABILITY OF REPORT: APPROVED FOR PUBLIC RELEASE;DISTRIBUTION UNLIMITED. |
| 2b DECLASSIFICATION/DOWNGRADING SCHEDULE: | |
| 4 PERFORMING ORGANIZATION REPORT NUMBER(S): | 5 MONITORING ORGANIZATION REPORT NUMBER(S): R&D 4624 - CC-01 |
| 6a NAME OF PERFORMING ORGANIZATION: UNIVERSITY OF PARIS SOUTH / 6b OFFICE SYMBOL (If applicable): | 7a NAME OF MONITORING ORGANIZATION: USARDSG (UK) |
| 6c ADDRESS (City, State, and ZIP Code): CENTRE D'ORSAY, BAT 490-91405, ORSAY,CEDEX PARIS FRANCE. | 7b ADDRESS (City, State, and ZIP Code): BOX 65, FPO NY 09510-1500. |
| 8a NAME OF FUNDING/SPONSORING ORGANIZATION: USARDSG(UK) ERO / 8b OFFICE SYMBOL (If applicable): | 9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: DAJA45-85-C-0014 |
| 8c ADDRESS (City, State, and ZIP Code): BOX 65, FPO NY 09510-1500 | 10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. 61102A / PROJECT NO. 1L6LL02 BH57 / TASK NO. 03 / WORK UNIT ACCESSION NO: |

11 TITLE (Include Security Classification): (U)RESEARCH ON TOOLS FOR LEARNING.

12 PERSONAL AUTHOR(S): DR. YVES KODRATOFF

| 13a TYPE OF REPORT | 13b TIME COVERED | 14 DATE OF REPORT (Year, Month, Day) | 15 PAGE COUNT |
|---|---|---|---|
| FINAL | FROM 1985 TO 1987 | 1987 SEPTEMBER 1ST | 41 |

16 SUPPLEMENTARY NOTATION:

| 17 COSATI CODES | | | 18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) |
|---|---|---|---|
| FIELD | GROUP | SUB-GROUP | |
| 09 | 02 | | |
| | | | |

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

CONCEPTS RELATING TO SYMBOLIC MACHINE LEARNING (ML) ARE DISCUSSED IN THIS REPORT. THESE CONCEPTS INCLUDE KNOWLEDGE REPRESENTATION, DESCRIPTIVE NOTATIONS, AND METHODS OF GENERALIZATION ML TECHNIQUES HAVE BEEN APPLIED TO SCENE ANALYSIS THROUGH IMPLEMENTATION OF A SYSTEM THAT LEARNS FEATURES IN ORDER TO RECOGNIZE MULTI-FONT CHARACTERS. HIGHLIGHTS OF THIS RESEARCH ARE DISCUSSED.

| 20 DISTRIBUTION/AVAILABILITY OF ABSTRACT | 21 ABSTRACT SECURITY CLASSIFICATION |
|---|---|
| ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS | UNCLASSIFIED |
| 22a NAME OF RESPONSIBLE INDIVIDUAL: DR. J. ZAVADA | 22b TELEPHONE (Include Area Code): 01-402-9490 / 22c OFFICE SYMBOL: AMXSN-UK-RI. |

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

# IMPROVING THE TOOLS OF SYMBOLIC LEARNING

Yves KODRATOFF

*Unité associée au CNRS UA 410:AL KHOWARIZMI*

*Décembre 1986* *Rapport de Recherche n° 321*

# IMPROVING THE TOOLS OF SYMBOLIC LEARNING

Yves Kodratoff
Equipe Inférence et Apprentissage
Bâtiment 490, Laboratoire de Recherche en Informatique
Université Paris-Sud, UA-410 du CNRS,
F - 91405 Orsay France

**RESUME**

Dans la première partie de cet article, nous donnons quelques conséquences du choix d'une définition de la notion de *Généralisation*. Nous discutons des relations entre définitions fondées sur la déduction et celles fondées sur la substitution.

Dans une seconde partie, nous montrons comment une approche symbolique peut rendre compte, au moins partiellement, du bruit présent dans toute donnée réelle. Nous discutons de cette approche pour l'Analyse des Scènes, l'acquisition de règles et de stratégies de contrôle. Finalement, nous présentons notre idée d' Espace des Versions Polymorphique.

**SUMMARY**

In its first part, this paper presents some consequences of the choice of the definition of *Generalization*. It discusses the definitions based on deduction, versus those based on substitution.

In its second part, it shows how symbolic computations are also able to take into account, at least partly, the noise most real-life data show. It discusses symbolic approaches to noise handling in Scene Analysis, rule learning, strategy learning and, finally, of the idea of polymorphic Version Space.

**Key-words:**

Deductive Generalization, Generalization in an Equational Theory, Learning Strategies, Polymorphy, Resistance to Noise, Rule Learning, Scene Analysis, Version Space. (France, French language)

## INTRODUCTION

In this paper, we shall emphasize two aspects of this part of symbolic Machine Learning which deals with learning from sets of several examples, and the aim of which is "moving from more specific descriptions to more general descriptions" [Langley 1986] ( called here **generalization** ).

One is relative to the practical consequences of a theoretical puzzle.

Among the important techniques used in ML are techniques of generalization, and specialists in ML build up systems that attempt to provide descriptors ( i.e., atomic formulas ) that have the best degree of generalization. For instance, the Version Space [Mitchell 1982] paradigm is a method that helps to find the exact generalization state in which a descriptor must be used in order to optimize the problem solving efficiency of operators making use of this descriptor.

It is then somewhat surprising to see that Classical Logics do not define the generalization state of an atomic formula. The only existing logical tool is relative to disjunctive formulas and is called *subsumption*, while *substitution* defines the relative generality of terms ( i.e., formal functional expressions that are not evaluated ).

We shall attempt to clarify this situation, up to the point where some of the practical consequences of our theoretical choices can be seen.

In section 1, we study definitions of the generalization of implications and conjunctive formulas, and their differences, we study also the practical consequences of choosing *Modus Ponens* instead of the *Generalization Principle* as an inference rule. An other, related, topic of section 1 is the discussion of the use of the properties of the descriptions one wants to learn from.

The other aspect is : how far symbolic methods, as opposed to numeric ones, must be of use ?

In most of the present published works, as soon as some *noise* or some **polymorphy** ( i.e., when concepts have intersecting sets of instances ) has to be taken into account, the authors rush upon numerical representations they claim being the only way to cope with those problems. We have chosen the opposite approach, which is to stick as far as possible to symbolic representations, even if it may first seem absurdly too far. For instance, we would represent polymorphy by putting upper and lower bounds to the properties of concepts, rather than assigning to a given instance so much chances to belong to one concept and so much to belong to an other one.

Section 2 will be devoted to the study of symbolic handling of noise and polymorphy, with, for instance, a presentation of "Polymorphic Version Spaces" which illustrates well how seemingly purely symbolic techniques can be also applied in a wider context.

Our approach aims at improving the provability of each learning steps, and we believe that provability is a necessary ( if not sufficient ) step for explicablity. This last statement is well illustrated by EBG [Mitchell & al. 1986] where explanations are derived from proofs. In our opinion, this point is of much importance, this is why we shall come back to it in conclusion.

## 1. - DIFFERENT DEFINITIONS OF GENERALIZATION

### 1.1 - Intuitive Definition of Generalization

There exists one definition which is agreed upon by all authors, the most intuitive one. We give it under a simplified form where the formulas depend on one variable only. When there are several van-

ables, one has to take into account the fact that each variable is relative to a given *object*. Object oriented generalization is a rather new topic [Manago 1986], we will not go into it because we would like to stick to well-known concepts in this section.

Let P(x) and Q(y) be two formulas.
Let us note by $\{P_{TRUE}\}$ the set of the instances of x such that P(x) = TRUE, and similarly for Q.

$$\{P_{TRUE}\} = \{ x / P(x) = TRUE\}$$
$$\{Q_{TRUE}\} = \{ y / Q(y) = TRUE\}$$

Then one says that P(x) is **more general than** Q(y) iff $\{P_{TRUE}\} \supset \{Q_{TRUE}\}$.

This definition is the one actually used when one wants to show that, say, P(x) is not more general than Q(y). In that case it is enough to exhibit an instance of x such that P(x) is FALSE and Q(y) is TRUE.

The problem, however, is to be able to compute a generalization from its instances, and the above definition gives no way to achieve this goal. This is why alternate definitions, leading to a generalization algorithm, have been developed.

### 1.2 - Vere 's definition of generalization

Let us first consider a conjunction of descriptors. A formula has therefore the form

$$A = A_1 \,\&\, \ldots \,\&\, A_n$$

where each $A_i$ is a descriptor.
Let $\{A\}$ be called the set associated to $A$, defined by

$$\{A\} = \{A_1, \ldots, A_n\}$$

Then $A$ is **more general than** $B$ iff there is

- an expression $B'$ such that $\{B'\} \subseteq \{B\}$
- a substitution $\sigma$ such that $\sigma A = B'$.

Otherwise stated, $\sigma A$ is equal to a subpart of $B$, up to a variable renaming.

For disjunctions of conjunctions, this definition becomes : Let $G_a = g_{a1} \vee \ldots \vee g_{am}$, $G_b = g_{b1} \vee \ldots \vee g_{bn}$, then $G_a$ is **more general than** $G_b$ iff $\forall j \exists i$ such that $g_{ai}$ is more general than $g_{bj}$.

The main drawback of this definition is that it gives no control on the way conjuncts are dropped during the generalization process.

### 1.3 - Existential versus Universal quantification

The state of quantification of the variables introduced during the generalization process depends on

1 - the form of the expressions given as example
2 - the use of the generalized expression.

The form of the expressions given as example depends very much on the way the information is represented.

*Consider the English sentence "That particular crow, named Jack, is black".*
*It can be interpreted either as an implication, or as a conjunction. Disputing on which is the best would be outside of the scope of this paper.*

*In the first case, its first order logic representation will be :*

$$CROW(JACK) \Rightarrow BLACK(JACK).$$

*in the second case, it will be :*

$$CROW(JACK) \;\&\; BLACK(JACK).$$

**When one is learning from implications** ( or, more generally, from theorems ) the intuitive behaviour consists in introducing universally quantified variables [Plotkin 1970].

*From the knowledge*

$$CROW(JACK) \Rightarrow BLACK(JACK)$$
$$CROW(JOCK) \Rightarrow BLACK(JOCK),$$

*one is tempted to infer*

$$\forall x\ [CROW(x) \Rightarrow BLACK(x)]$$

*because it gives a good representation of the sentence "All crows are black".*

**When one is learning from conjunctions,** it is counter-intuitive to introduce universal quantifiers.

*From the knowledge*

$$CROW(JACK) \;\&\; BLACK(JACK)$$
$$CROW(JOCK) \;\&\; BLACK(JOCK),$$

*one is not tempted to infer*

$$\forall x\ [CROW(x) \;\&\; BLACK(x)]$$

*because it represents the sentence "All objects are black crows" which is nowhere in the examples. Even more convincingly, one cannot learn that*

$$\forall x \forall y\ [BLACK(x) \;\&\; WHITE(y)]$$

*from*

$$BLACK(CROW) \;\&\; WHITE(SWAN)$$
$$BLACK(JAY) \;\&\; WHITE(DOVE)$$

*since the examples contain no contradiction while $\forall x \forall y\ [BLACK(x) \;\&\; WHITE(y)]$ does.*

*Nevertheless, it may seem a bit awckward to "infer" from them*

$$\exists x \exists y\ [BLACK(x) \;\&\; WHITE(y)]$$

*since this existential theorem is nothing but a mere logical deduction from either example.*

Suppose that you start from a relation R(A, B) among instances. It is trivial to understand that, most often, the relation $\forall x \forall y\ [R(x, y)]$ is wrong. One has to find a relation of the type

$$\forall x\ \forall y\ [P(x) \;\&\; Q(y) \Rightarrow R(x, y)]$$

where P and Q describe those variables for which R is TRUE, but in general, one has no way to find P and Q.

That explains why some authors define

$$P(A) \text{ generalizes into } \exists x\ P(x) \text{ iff } \exists x\ [P(A) \Rightarrow P(x)]$$

Since this implication is a tautology, this definition is also very much disputable. The idea of generalization conveys some increase in the information content of the generalized formula. Here, on the contrary, generalization would take place, and seemingly decrease the information content of the generalized formula. This last point will be detailed in section 1.3.2.3 below.

Let us now see how these problems are handled in each particular case.

### 1.3.1 - Theorem Learning

When one is learning from example theorems, one will introduce universally quantified variables. This gives rise to two different difficulties. Both of them are extremely deep problems and their answer belong to long term research. Nevertheless, we shall now describe them briefly.

Firstly, there exist indeed theorems that contain existential quantifiers, and the recognition of this existential quantifier is very difficult problem which amounts to function synthesis.

Secondly, the examples usually do not specify what is the domain of validity of the theorem ( i.e., one learns usually false theorems from examples ) and the determination of this domain amounts to predicate synthesis.

1.3.1.1 - Inventing Skolem Functions

When some variables are existentially quantified, there is always a hidden function which will be extremely difficult to put into evidence.

*Suppose that one is learning from set of examples like : 0 + 1 = 1, 0 + 2 = 2, ..., 1 + 0 = 1, ..., 1 + 1 = 2, etc ..., where + is an unknown symbol. It would be wrong to infer formulae all the variables of which are universally quantified like : $\forall x \forall y \forall z\ [x + y = z]$.*

*Let us now suppose that it has been possible, say by using suitable counter-examples, to guess that one possible formula is $\forall x \forall y \exists z\ [x + y = z]$.*
*Obviously, this last theorem, although true, does not solve the learning problem implicitly stated by the above sequence of examples : "invent a definition of a function + that fits with this set of input-output examples".*

When a theorem contains existential quantifiers, the first goal is, of course, to recognize which are the variables under their scope. In general, as the example shows, this is not the ultimate goal which is rather : "remove those existentially quantified variables by synthesizing a suitable skolem function that fits with the examples".

*Instead of $\forall x \forall y \exists z\ [x + y = z]$, one rather wants to find a function f such that $\forall x \forall y\ [x + y = f(x, y)]$, and f realizes the operation +.*

Several methodologies that propose an approach to the solution of this problem can be found in [Biermann & al. 1984]. Recently, an original approach has been developed and implemented in our group [Franova 1985, 1986].

1.3.1.2 - Finding Domain Definitions

Let us suppose now that we are in the simpler case where all quantifications are universal ones. It does not mean that the theorem is true in all possible interpretations : one must also find the domain of definition of the variables.

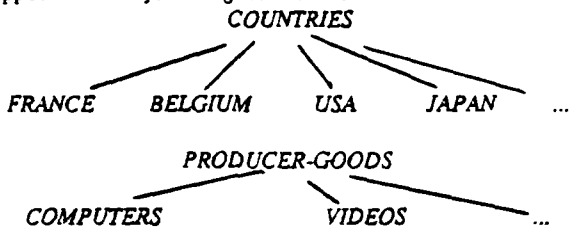
*Suppose that the system is to learn rules concerning the economic relationships between countries.*
*For example, it will be told that :*
*If France is a buyer of video recorder, and Japan produces them, then France is a potential buyer of video recorder from Japan.*
*A formal way of representing this sentence is :*
*$E_1$ : NEEDS(FRANCE, VIDEOS) & PRODUCES(JAPAN, VIDEOS) --> POSSBUY(FRANCE, VIDEOS, JAPAN).*

*Assume that we also have the second example :*
*$E_2$ : NEEDS(BELGIUM, COMPUTERS) & PRODUCES(USA, COMPUTERS) --> POSSBUY(BELGIUM, COMPUTERS, USA).*

*It is then easy to find the following generalization :*
*G : $\forall x \forall y \forall u$ NEEDS(x, u) & PRODUCES(y, u) --> POSSBUY(x, u, y).*

*This generalization is still not correct since, for instance, it would allow x and u to be instantiated by the same value.*

*In this simple case, suppose that the following taxonomies are available :*




*These hierarchies describe the possible domains of the variables, this information can be introduced as a condition to the application of the rule :*

*∀x∀y∀u [ IF COUNTRY(x) & COUNTRY(y) & PRODUCER-GOODS(u)*
*THEN { NEEDS(x, u) & PRODUCES(y, u) --> POSSBUY(x, u, y) } ].*

A greater refinement is of course possible when information is available, more detailed than the two above taxonomies [Kodratoff 1985], [Kodratoff 1986a].

### 1.3.2 - **Concept Learning or Different Ways to Use a Recognition Function**

In this section, let us assume that we do not learn rules or theorems, but conjuncts of atoms.
This kind of learning aims at obtaining a formula, called **recognition function**, that characterizes the micro-world to which the examples belong.

When quantifiers are introduced, the recognition process will work by using a deduction principle.
In our example, we shall use refutation and write the recognition as the deduction in a PROLOG program, and use Edinburgh notation [Clocksin & Mellish 1981]

*Suppose that we start from*

$E_1$ : *"This scene contains KOKO which is a white swan",*
$E_2$ : *"This scene contains KIKI which is a white swan".*

*These examples are interpreted as a description of some scene "This scene".*
*They are then given the form :*

$E_1'$ : *SWAN(KOKO) & WHITE(KOKO)*
$E_2'$ : *SWAN(KIKI) & WHITE(KIKI).*

*Obviously, one aims here at recognizing scenes that contain a white swan.*
*This example has been chosen on purpose to be opposed to the "black crow" one since all swans are not white.*

#### 1.3.2.1 - Universal quantification

All variables are universally quantified, the recognition "function" has therefore the form : ∀x P(x).
It will be used as a recognition function of a scene, say $S_1$, as defined :
One says that

**∀x P(x) recognizes $S_1$ when one can prove ∀x P(x) ⇒ $S_1$.**

Using refutation for the proof of ∀x P(x) ⇒ $S_1$ amounts to prove that its negation leads to a contradiction, i.e. that

$$\forall x\ P(x)\ \&\ \neg S_1$$

leads to a contradiction.

*$E_1'$ and $E_2'$ will generalize to*

$$\forall x\ P(x) = \forall x\ [SWAN(x)\ \&\ WHITE(x)]$$

*which will be used to recognize a scene made of a white swan.*
*Suppose that we want to check that*

$$S_1 = [SWAN(JACKO)\ \&\ WHITE(JACKO)]$$

*is recognized.*
*One has to prove that*

*SWAN(x) :-*
*WHITE(x) :-*
*:- SWAN(JACKO), WHITE(JACKO)*

*leads to a contradiction, which is of course the case.*
*Therefore, the scene is recognized by this recognition function.*

This kind of generalization has the property ( in some cases it is a drawback, in some others it may be an advantage ) that it will fail to recognize a scene with additional details.

*Consider a scene with a white swan and a Renault car, one will have to find a contradiction in the set:*

*SWAN(x) :-*
*WHITE(x) :-*
*:- SWAN(JACKO), WHITE(JACKO), CAR(RENAULT)*

*and this will not be possible.*

In conclusion, one must use universally quantified variables when one looks for a recognition function that recognizes whole scenes. One must not use them when the recognition function is supposed to recognize sub-parts of a scene.

1.3.2.2 - Existential quantification

All variables are existentially quantified, the recognition "function" has therefore the form : $\exists x\ P(x)$.
It will be used as a recognition function of a scene, say $S_1$, as defined :
One says that

$$\exists x\ P(x) \text{ recognizes } S_1 \text{ when } S_1 \vdash \exists x\ P(x),$$

i.e., when one can deduce $\exists x\ P(x)$ from $S_1$.
Using refutation for the proof of $S_1 \vdash \exists x\ P(x)$ amounts to prove that deducing the negation of $\exists x\ P(x)$ from $S_1$ leads to a contradiction, i.e. that

$$S_1\ \&\ \neg\ \exists x\ P(x)$$

leads to a contradiction.

*$E_1'$ and $E_2'$ will generalize to*

$$\exists x\ P(x) = \exists x\ [SWAN(x)\ \&\ WHITE(x)],$$

*therefore one has :*

$$\neg\ \exists x\ P(x) = \forall x\ \neg P(x) = \forall x\ \neg[SWAN(x)\ \&\ WHITE(x)]$$

*Suppose that we want to check that*

$$S_1 = [SWAN(JACKO)\ \&\ WHITE(JACKO)]$$

*is recognized.*
*One has to prove that*

*SWAN(JACKO) :-*
*WHITE(JACKO) :-*
*:- SWAN(x), WHITE(x)*

*leads to a contradiction, which is of course the case.*
*Therefore, the scene is recognized by this recognition function.*

1.3.2.3 - Conclusion

Existential quantification might have been felt as counter-intuitive because "nothing is learned" from it. This is not true for the following reasons.

- The existential theorem $\exists x\ P(x)$ learned from a set of examples $\{E_1, ..., E_n\}$ must be deducible from each $E_i$. It therefore catches, as it should be, some of the common features of the examples.
It will recognize a scene by its sub-parts.

*Consider again a scene with a white swan and a Renault car, one will have to find now a contradiction in the set :*

*SWAN(JACKO) :-*
*WHITE(JACKO) :-*
*CAR(RENAULT) :-*
*:- SWAN(x), WHITE(x)*

*and presence of a renault car is no longer harmful.*

- This definition is actually very near to the intuitive one. In particular, it contains Michalski's generalization rules [Michalski & Stepp 1983], [Michalski 1984]. For instance, the example of section 1.3.2.2 shows how it contains the "dropping condition" rule.

- This approach has been used in [Kodratoff 1985] for the specific case of counter-examples. It has been generalized by Nicolas [Nicolas 1986a, 1986b] who uses a theorem prover in order to perform inductive learning, which may seem surprizing at first sight.

We have developed an other way to define generalization, by extending the classical definition of term *generalization, as seen in the next section.*

In order to prove the necessity of introducing these new concepts, let us consider the following counter-example to the methods issued from Modus Ponens, as presented in sections 1.3.1 and 1.3.2.

1.3.3 - A **Counter-example**

Let us now give a "counter-example" to deductive definition, in that sense that a best generalization is not found by it.

1.3.3.1 - A definition of "best generalization" issued from Modus Ponens

There is an obvious way to define a best generalization when one quantifies existentially the variables.
The best generalization is the one which is the "nearest" to all the examples, but contains the information they have in common.
Let $\{E_i\}$ be a set of examples, and $\{G_j\}$ be a set of possible generalizations, i.e., $\forall i$, one must be able to prove that $E_i \vdash G_k$, for each of the $G_k$'s.
Since one infers the generalizations from the examples, it is obvious that one must define the best generalization among the $G_k$'s by being the mosy specific one, i.e. the one, if it exists, from which all others can be inferred.

1.3.3.1 - The counter-example

*Suppose that one starts from the two examples*

$E_1$ : *ON(A, B) & NEAR(B, C)*
$E_2$ : *ON(D, E)*

*with the theorems*

$\forall x\ \forall y\ [ON(x, y) \Rightarrow NEAR(x, y)]$
$\forall x\ \forall y\ [NEAR(x, y) \Leftrightarrow NEAR(y, x)]$

*Using these theorems, one can show that the two following Potential Generalizations*

$$G_1 : \exists x\ \exists y\ [ON(x, y)]$$
$$G_2 : \exists x\ \exists y\ \exists z\ [ON(x, y)\ \&\ NEAR(y, z)]$$

*are equivalent relative to our definition, since* $G_1 \Leftrightarrow G_2$.

*Nevertheless, the associated generalizations obtained by substitution techniques, as seen below, are :*

$$f_1 : ON(x, y)$$
$$f_2 : ON(x, y)\ \&\ NEAR(y, z)$$

*They are not equivalent since, using the theorems, one can show that* $f_1$ *is equivalent to*

$$f_1' : ON(x, y)\ \&\ NEAR(y, x).$$

$f_2$ *is clearly ( from definition 1.5.2 below ) more general than* $f_1$ *since the substitution* $\sigma = \{x \leftarrow x, y \leftarrow y, z \leftarrow x\}$ *is such that* $\sigma f_2 = f_1'$.

## 1.4 - Term generalization

### 1.4.1 - Terms

Let V be a countable set of variables and F a family of functions indexed by the natural numbers. When a function f belongs to $F_n$, one says that the arity of f is n. The set $F_0$ of functions of arity zero is called the set of the constants.

The set of terms on V and T, is defined by
(i) $v \in V$ is a term
(ii) $f(t_1,.....t_n)$ is a term iff $f \in F_n$ and $t_1,.....t_n$ are terms.

Intuitively, the set of terms is a set of expressions built with functions of some arity, constants and variables.

### 1.4.2 - Generalization

The term $t_1$ **is more general than the term** $t_2$, denoted by $t_1 \leq t_2$, iff there exists a substitution $\sigma t_1 = t_2$.
This definition does not take into account the properties of the functions. One describes these properties by a "theory" ε, and one defines a generalization modulo this theory.

### 1.4.3 - ε - generalization

Let ε be a set of axioms which express the properties of the functions.
When one needs to use these axioms in order to recognize the equality of two terms, one says that they are **ε-equal**
*For instance, the two terms* $t_1 = (2 + 3)$ *and* $t_2 = (3 + 2)$ *are not considered as "equal" but as "ε-equal" because one needs to use the axiom of + commutativity :*

$$\forall x\ \forall y\ [(x + y) = (y + x)],$$

*in order to recognize that* $t_1 =_\varepsilon t_2$.

This definition may seem counter-intuitive but is it necessary to single out the use of axioms in the context of an automatic generation of generalizations because their use may lead to infinite computation loops ( using the axiom in one direction and then in the other one ). This kind of problems have been very much studied, see for instance [Stickel 1981], [Hsiang 1982].

Let $=_\varepsilon$ denote ε-equality.
A term $t_1$ **is more general than a term** $t_2$ in the theory ε iff there exist $t_1' =_\varepsilon t_1$ and $t_2' =_\varepsilon t_2$ **and a** σ

such that $\sigma t_1' = t_2'$.

Depending on $\varepsilon$, it may be that the above definition of $\varepsilon$-generalization is not consistent with its implicit future use for definition an ( at least partial ) order. Using some of the properties one may find $t_1'$ and $t_2'$ such that $t_1 =_\varepsilon t_1'$ and $t_2 =_\varepsilon t_2'$ and there is a $\sigma$ such that $\sigma_1 t_1' = t_2'$.
Nevertheless, it may well also be that, using other properties, one can find $t_1''$ and $t_2''$ such that $t_1 =_\varepsilon t_1''$ and $t_2 =_\varepsilon t_2''$, and there exists $\sigma_2$ such that $\sigma_2 t_2'' = t_1''$ even when $t_1' \neq_\varepsilon t_2''$ [Kodratoff & Ganascia 1986].

Since we want to use the properties of the functions, and further define the generality of formulas (therefore using the properties of our connectors) it is necessary to find a definition of $\varepsilon$ - generalization that avoids this difficulty.

1.4.4 - **Example of $\varepsilon$ - generalization** (where atomic formulas are treated like terms)

Let us suppose that we work in a world of objects which have a color and the the following knowledge is available

$$\forall x \; \exists y \; \text{COLOR}(y, x)$$

It states that each object x has a color named y. In addition, RED is a kind of COLOR and this information is supposed to be also known. This knowledge allows us to transform any atomic formula like RED(x) into an instance of more general atomic formula COLOR(RED, x).
Let us compare the generality of the concept "red square" $C_1$ and "square" $C_2$.

$$C_1 = \text{SQUARE}(x) \;\&\; \text{RED}(x)$$
$$C_2 = \text{SQUARE}(x)$$

Applying the above theorem, one knows that for any x of $C_2$ , it has an unknown color , say y. Therefore $C_2$ is equivalent to $C_2' = \text{SQUARE}(x) \;\&\; \text{COLOR}(y, x)$. Based on the fact that RED is more particular than COLOR, one can find $C_1 =_\varepsilon C_1'$, $C_1' = \text{SQUARE}(x) \;\&\; \text{COLOR}(\text{RED}, x)$. Now, the usual term definition of generality can be applied since $\sigma C_2' = C_1'$ with $\sigma = (y \leftarrow \text{RED})$. Therefore $C_2$ is more general than $C_1$ in the theory which contains the above information.

**1.5 - Definition of Formula Generalization modulo a theory**

Let $E_1$ and $E_2$ be two formulas and $\varepsilon$ an equational theory.

1.5.1 - **Generalized formula**

We say that formula $E_1$ is a **generalization** of formula $E_2$ if Condition 1 is fulfilled.

Condition 1 : there exists $E_1'$ such that $E_1' =_\varepsilon E_1$ and there is $\sigma_2$ such that $\sigma_2 E_1' =_\varepsilon E_2$.

This condition states that there exists $E_1'$, equivalent to $E_1$ and that $E_1'$, considered as a term, is more general than $E_2$ considered as a term.
The next definition gives an other condition which insures that formula generality is a partial ordering.

1.5.2 - **Generality relation between two formulas.**

We shall say that $E_1$ **is more general than** $E_2$ when Condition 1 and Condition 2 are fulfilled. Condition 1 is as above and

Condition 2 : For all $E_2'$ such that $E_2' =_\varepsilon E_2$ , if there exists $\sigma_1$, such that $\sigma_1 E_2' =_\varepsilon E_1$, then $E_2' =_\varepsilon E_1$.

This second condition states that the first condition can actually be used for ordering the formulas.

It says that if there is a $E_2'$ which is equivalent to $E_2$ and which is more general ( as a term ) than $E_1$, then all three $E_1$, $E_2$, $E_2'$ must be equivalent.

Some theoretical consequences of condition 2 have been studied in [Kodratoff & Ganascia 1986] under the name of *i-implication*.

We shall rather explain how one can make an algorithm out of definition 1.5.2.
One needs to find out the transformed $E_1$ and $E_2$, called $E_1'$ and $E_2'$ in the above definition. We have called this work : **Structural Matching** [Kodratoff 1983].

### 1.6 - Structural Matching ( SM )

#### 1.6.1 - Definition

Two formulas structurally match if they are identical except for the constants and the variables that instantiate their predicates.

More formally :
Let $E_1$ and $E_2$ be two formulas.
$E_1$ **structurally matches** $E_2$ iff there exists a C and there exist $\sigma_1$ and $\sigma_2$ such that
1- $\sigma_1 C = E_1$ and $\sigma_2 C = E_2$ .
2- $\sigma_1$ and $\sigma_2$ never substitute a variable by a formula or a function.

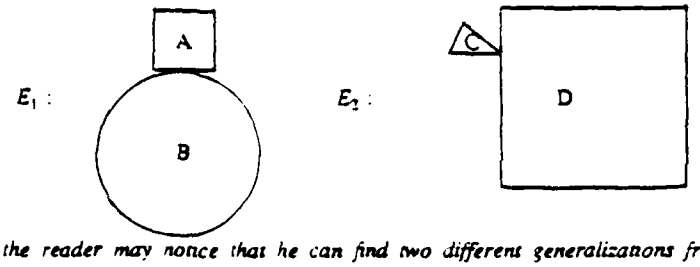
It must be understood that SM may be difficult up to undecidable. Nevertheless, in most cases, one can use the information coming from the other examples, in order to know how to orientate the proofs necessary to the application of this definition.

#### 1.6.2 - SMizing two formulas

SM may well fail, whereas the effects of the attempt to put into SM may still be interesting.
We say that two formula have been SMized when every possible property has been used in order to put them into SM.
When the SM is a success, then SMizing is identical to putting into SM.
When the SM is a failure, SMizing keeps the best possible result in the direction of matching formulas.

#### 1.6.3 - A simple example of ( successful ) Structural Matching

*Consider the two following examples.*




*Using his intuition, the reader may notice that he can find two different generalizations from these examples.*
*He sees that either*
- *there are two different objects touching each other, and a small polygon*

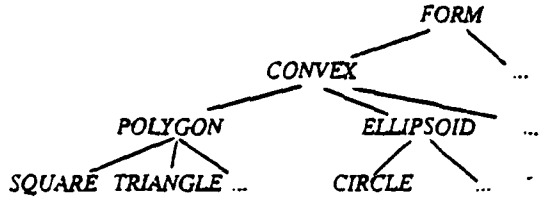
*- there are two different objects touching each other, one of them is a square.*

*Both generalizations are true and there is no reason why one of them should be chosen rather than the other. We shall now see that one of the interesting features of SM is that it keeps all the available information, and therefore constructs a formula containing both the above two "concepts".*

*The examples can be described by the following formulas*

$$E_1 = SQUARE(A) \;\&\; CIRCLE(B) \;\&\; ON(A, B) \;\&\; SMALL(A) \;\&\; BIG(B)$$
$$E_2 = TRIANGLE(C) \;\&\; SQUARE(D) \;\&\; TOUCH(C, D) \;\&\; SMALL(C) \;\&\; BIG(D)$$

*Let us suppose that the following hierarchy is provided to the system.*




*together with the theorems*

$$\forall x \, \forall y \; [ON(x, y) \Rightarrow TOUCH(x, y)]$$
$$\forall x \, \forall y \; [TOUCH(x, y) \Leftrightarrow TOUCH(y, x)]$$

This taxonomy and the theorems represent our semantical knowledge about the micro-world in which learning is taking place.

The SM of $E_1$ and $E_2$ proceeds by transforming them into equivalent formulas $E_1'$ and $E_2'$, such that $E_1'$ is equivalent to $E_1$, and $E_2'$ is equivalent to $E_2$ in this micro-world ( i.e., taking into account its semantics ).
When the process is completed, $E_1'$ and $E_2'$ are made of two parts.
One is a variabilized version of $E_1$ and $E_2$. It is called the **body** of the SMized formulas. When SM succeeds, the bodies of $E_1'$ and $E_2'$ are identical.
The other part, called the **bindings** ( of the variables ), gives all the conditions necessary for the body of each $E_i'$ to be identical to the corresponding $E_i$.

The algorithm that constructs $E_1'$ and $E_2'$ is explained in [Kodratoff 1983, Kodratoff & Ganascia 1986, Kodratoff & al. 1984]. It has been implemented several times under the name of AGAPE or MAGGY.

*In our example, it would find*
*Body of* $E_1'$ =
$POLYGON(u, y) \;\&\; SQUARE(x) \;\&\; CONVEX(v_1, v_2, z) \;\&\; ON(y, z) \;\&\; TOUCH(y, z) \;\&\; SMALL(y) \;\&\; BIG(z)$
*Bindings of* $E_1'$ =
$((x = y) \;\&\; (y \neq z) \;\&\; (x \neq z) \;\&\; (v_1 = ELLIPSOID) \;\&\; (v_2 = CIRCLE) \;\&\; (u = SQUARE) \;\&\; (x = A) \;\&\; (z = B))$

*Body of* $E_2'$ =
$POLYGON(u, y) \;\&\; SQUARE(x) \;\&\; CONVEX(v_1, v_2, z) \;\&\; TOUCH(y, z) \;\&\; SMALL(y) \;\&\; BIG(z)$
*Bindings of* $E_2'$ =
$((x \neq y) \;\&\; (y \neq z) \;\&\; (x = z) \;\&\; (v_1 = POLYGON) \;\&\; (v_2 = SQUARE) \;\&\; (u = TRIANGLE) \;\&\; (x = D) \;\&\; (y = C))$

*The reader can check that $E_1'$ and $E_2'$ are equivalent to $E_1$ and $E_2$.*

*$E_1'$ and $E_2'$ contain exactly the information extracted from the hierarchy and the theorems which is necessary to put the examples into SM.*

*For instance, in $E_1'$, the expression ' (POLYGON(u, y) ' means that there is a polygon in $E_1$, and since we have the binding (u = SQUARE), it says that this polygon is a square, which is redundant in view of the fact that SQUARE(x) & (x = y) says that x is a square and is the same as y. This redundancy is not artificial when one considers the polygon in $E_2$ which is a TRIANGLE.*

This example shows well that, once this SM step has been performed, the generalization step itself becomes trivial : we keep in the generalization all the bindings common to the SMized formulas and drop all those not in common.
In other words, this SM technique allows to reduce the well-known generalization rules [Michalski 1983, 1984] to the only "dropping condition rule" which becomes legal on SMized formulas. All the induction power is in the dropping condition rule, all other rules are purely deductive. We must confess that formal proof of the above statement is still under research.

*The generalization $E_1$ and $E_2$ is therefore*

*$E_g$ : POLYGON(u, y) & SQUARE(x) & CONVEX($v_1$, $v_2$, z) & TOUCH(y, z) & SMALL(y) & BIG(z)*
*with bindings (y ≠ z).*

*In "English", this formula means that there are two different objects ( named y and z ), y and z touch each other, y is a small polygon, z is a big convex, and there is a square ( named x ) which may be identical to y or z.*

It can be easily guessed that using theorems can lead to many difficulties, since one enters the realm of Theorem Proving, which is well-known for being a good source of yet unsolved problems.
In the case of SM, one is driven by the need to put the examples into a similar form, and the usual difficulties of Therorem Proving are somewhat smoothered out.
We cannot formally prove this point, but the following example, taken from [Vrain 1986] can at least illustrate our claim.

### I.7 - Using theorems to improve generalization

Starting from two examples that have no common predicates, we show that they nevertheless have a common generalization, found by using theorems that link the predicates.

*Let the examples be*

$$E_1 = MAMMALIAN(A) \;\&\; BRED\_ANIMAL(A)$$
$$E_2 = TAME(B) \;\&\; VIVIPAROUS(B)$$

*to which the following theorems are joined*

$$R_1: \forall x\ [MAMMALIAN(x) \;\&\; BRED\_ANIMAL(x) \Rightarrow TAME(x)]$$
$$R_2: \forall x\ [TAME(x) \;\&\; VIVIPAROUS(x) \Rightarrow MAMMALIAN(x)]$$
$$R_3: \forall x\ [TAME(x) \Rightarrow HARMLESS(x)]$$

*The first step of SM is here trivial : we replace the constants by a variable x, and obtain the equivalent examples :*

$E_1'$ = *MAMMALIAN(x) & BRED_ANIMAL(x) [EQ(x, A)]*
$E_2'$ = *TAME(x) & VIVIPAROUS(x) [EQ(x, B)]*

*Since the predicates have no common occurence, we consider the first ( this ordering is not significant, and just follows the one in which the examples are given ) predicate of $E_1'$ : MAMMALIAN. We see that we can deduce this predicate from $E_2$, using the rule $R_2$. We get:*

$E_1''$ = *MAMMALIAN*(x) & BRED_ANIMAL(x) [EQ(x, A)]*
$E_2''$ = *TAME(x) & VIVIPAROUS(x) & MAMMALIAN**(x) [EQ(x, B)]*

*The MAMMALIAN of $E_1'$ has been treated, this why it is marked by an * in $E_1''$. The one of $E_2''$ is issued from the use of theorems, this is why it is marked by **.*

*Using again the order in which the examples are given, the next non-marked predicate is BRED_ANIMAL.*
*-- No rule can be applied to $E_2''$ to make explicit the presence of BRED_ANIMAL in it.*
*-- Nevertheless, we remark that appiying the rule $R_1$ to $E_1''$ uses the concerned predicate : BRED_ANIMAL. Checking the effect of this application, we see that it generates the atomic formula TAME(x) and that there is an occurence of x in $E_2''$ which matches this occurence. Therefore, we conclude that we* **must** *apply $R_1$ to $E_1''$.*
*One obtains*

$E_1'''$ = *MAMMALIAN*(x) & BRED_ANIMAL*(x) & TAME**(x) [EQ(x, A)]*
$E_2'''$: *TAME*(x) & VIVIPAROUS(x) & MAMMALIAN**(x) [EQ(x, B)]*

*Now, the only un-matched predicate is VIVIPAROUS in $E_2'''$.*
*-- No rules can be applied to $E_1'''$ to make its presence explicit.*
*-- The only rule which can be applied in $E_2'''$, relative to VIVIPAROUS is $R_1$. But, it would introduce the atomic formula MAMMALIAN(x), which is already matched since its instances are starred.*

*No other rule can be applied, we star the predicate VIVIPAROUS to remember that it has already been dealt with, obtaining :*

$E_1''''$ = *MAMMALIAN*(x) & BRED_ANIMAL*(x) & TAME**(x) [EQ(x, A)]*
$E_2''''$ = *TAME*(x) & VIVIPAROUS*(x) & MAMMALIAN**(x) [EQ(x, B)]*

*All possible occurences have been dealt with, a complete SM is not possible, therefore the SMizing operation stops here.*

*Now, the generalization step is trivial : one drops the non-common occurences, obtaining the generalization*

*G = TAME(x) & MAMMALIAN(x)*

This example shows well how potential infinite proof loops can be easily avoided, simply because they do not improve the SMizing state of the examples.

More generally, one can use theorem proving techniques in order to improve the degree of similarity detected among the examples.
Such a system is under development in our group [Vrain 1987]. It is not the concatenation of a classical theorem prover and of generalization algorithms, but is rather strongly adapted to the kind of proofs required by Machine Learning.
As an instance of its peculiarity ( and of its incompleteness ), it will not allow to use twice the same theorem during a given derivation. This is of course a crude way to avoid infinite loops but, as the above example show, the corresponding incompleteness is not so wide as one could fear.

# 2 - SYMBOLIC LEARNING IN A NOISY ENVIRONMENT

In the presence of noise or polymorphy, numeric techniques have proven their usefulness. On this topic, we want to stress two points.

1 - applying numeric techniques too soon always spoils the understandability of the results, and may even hamper the efficiency.
2 - when used at proper time, they become a wonderful tool.

In other words, we do not criticize the use itself of numeric techniques, but their too early use.
The aim of this section is to show that one should, and this is possible, stick to symbolic techniques as far as possible before beginning to compute coefficients combinations.
This having been said, we are also quite conscious of the importance of a proper combination of the coefficients. We are simply a bit puzzled by the huge amount of research done about coefficient combination, and the tiny one done to find when and where they must combine. More details on this last point can be found in [Duval & Kodratoff 1986] in the case where one draws inferences from uncertain or noisy clauses.

Recently, and independantly, Michalski [Michalski 1986] has introduced the idea of "two-tiered concept meaning" which is a close parent of the ones presented here.

## 2.1 - Learning recognition functions in Scene Analysis

Scene Analysis is very typical of a huge development of numerical techniques and of what we shall call, and try to prove to be, a "hidden" use of symbolic method.

### 2.1.1 - Domain Independant Scene Analysis

There is of course a need for methods to go from the pixel level to the level of some descriptors ( like segments, curvature changes, etc ... ). Up to now, all available methods are purely numeric.

In our own research group, all symbolic learning for scene analysis that has been done, either starts from an already known symbolic description [Kodratoff & Lemerle-Loisel 1984] or from supposedly noise-free pixel descriptions [Cannat & Kodratoff 1986], [Cannat & al. 1986].
In order to fill up the gap between real images and ideal ones, we are presently using a numeric method due to [Mokhtarian & Mackworth 1986] which seems very promising.

This shows that we have followed a quite classical pattern : start from real pixel images, use numerical techniques to get a noise-free description in terms of high level descriptors, use then symbolic methods for interpreting these descriptors.

This approach to Scene Analysis seems to us justified if, and only if, one is supposed to simulate a system entering a brand-new domain, and forced to discover all forms each time it sees them. The methods issued from this approach should then be domain independant.

### 2.1.1 - Domain Dependant Scene Analysis

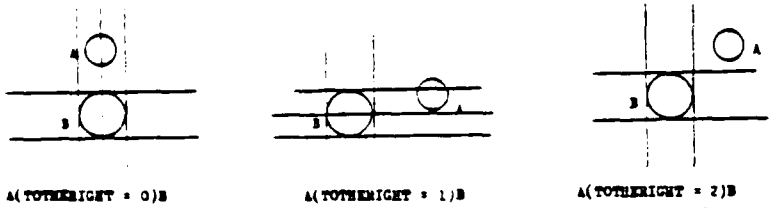
On the contrary, when one is working in a specific domain, one is always, implicitly or explicitly ( our point is : too often implicitly ) using high level knowledge relative to the domain. For instance, if some kind of curves are likely to appear, one will develop a special pixel-to-descriptor method to detect

them. Besides, one will introduce special descriptors that will take into account some subtle differences among those curves, that would be otherwise confused.

This is what we call "hidden use of symbolic methods".

*As an example of this undesirable feature, we shall self-criticize and cite [Kodratoff & Lemerle-Loisel 1984] where, for instance, the spatial relationships* TOTHERIGHT *among forms ( that are represented by circles ) are described as follows.*

*We define the* **horizontal strip** *associated to a circle ( respectively, its* **vertical strip** *) by the portion of the plane which is between two horizontal ( resp. vertical ) parallels tangent to the circle.*
*We then define 6 different sorts of operators describing different ways to express that a circle stands "to the right of" an other circle.*
*For instance, $TOTHERIGHT_0(A, B)$ says that circle A is actually directly above circle B, i.e. that the center of A is inside the vertical strip of B. $TOTHERIGHT_1(A, B)$ says that the center of A is inside the horizontal strip of B. $TOTHERIGHT_2(A, B)$ says that the center of A is outside both the horizontal and vertical strip of B, etc ...*




*By defining a strip and using it in the operator definitions, we ( in a hidden symbolic way ) handle the noise relative to the position of circle A when its center is approximately on a vertical ( resp. horizontal ) line, since the precision with which our operator is defined includes the width of B. On the contrary, when the center of A is around the limit of the strip, it becomes then extremely noise sensitive since the least difference may make decide that it is inside or outside the strip.*

This describes a partial handling of the noise, efficient in some situations, very poor in others, which is quite typical of hidden symbolic noise handling, and gives its limits.
Besides our own, most papers describing a specific application fall as well into the same trap.

We will not present here a complete solution but simply underline that it can be of two different kinds.

1 - Classically, one can introduce rude force belief coefficients and assign a belief to each descriptor.

2 - As recommended here, one should try to keep available as much as possible of the symbolic information. Here, this symbolic information can be represented by the fact that some operators are po-

lymorphic and some others are not.

*In our example, $TOTHERIGHT_0(A, B)$ and $TOTHERIGHT_2(A, B)$ are polymorphic since the center of A can be near the limit of the strip of B, in which case the slightest error may make switch from one to the other.*
*$TOTHERIGHT_0(A, B)$ and $TOTHERIGHT_1(A, B)$ are not polymorphic when the two circle do not intersect, they are when they intersect.*

Polymorphy would then be a better way to treat noise than numeric coefficients since it allows to keep more of the semantic of the domain.
In section 2.4 below we describe how polymorphy can even be used to retain most of the information provided by Mitchell's version spaces.

### 2.2 - Learning noise-resistant recognition functions

Once more, we want to stress the point that coefficients of some sort are not the exclusive solution to noise handling.

In this section, the generalization issued from the examples will be seen as a recognition function of the examples.

When noise is present in a data base, it often introduces some contradictions in it.
We shall study now the special case where these contradictions are actualized by the fact that sets of positive and negative instances are not disjoint. This case has been studed by [Fu & Buchanan 1985]. We present here a solution, first given in [Kodratoff & al. 1986], which is completely different from the one given by [Fu & Buchanan 1985].

Suppose that one starts from a set of examples $\{E\} = \{E_1, ..., E_n\}$ and counter-examples $\{CE\} = \{CE_1, ... CE_m\}$.
Let G be a conjunctive generalization of $\{E\}$ and let us suppose that some of the counter-examples, say the sub-set $\{CE'\}$, are recognized by G.

*We shall use the following example, inspired by plant pathology rules [Kodratoff & al. 1986].*
*Let the positive examples be :*
$E_1$ : *(COLOR = RED) & (SIZE = VERY-BIG) & (TEXTURE = SOFT)*
$E_2$ : *(COLOR = GREEN) & (SIZE = BIG) & (TEXTURE = HARD)*
$E_3$ : *(COLOR = GREEN) & (SIZE = VERY-BIG) & (TEXTURE = HARD)*

*and let a counter-example be :*
$CE_1$ : *(COLOR = RED) & (SIZE = BIG) & (TEXTURE = HARD)*

*Supposing that we know that BIG and VERY-BIG can generalize to LARGE, a conjunctive generalization of $\{E_1, E_2, E_3\}$ is*
G : *(COLOR = ANY) & (SIZE = LARGE) & (TEXTURE = ANY)*

*This generalization is also a recognition function when one says that it recognizes its instances. Unfortunately, it recognizes also $CE_1$ which is one of its instances.*

We suggest to treat this noise effect by partitioning $\{E_1, ..., E_n\}$ in two.
Let $\{E'\} = \{E_1, ..., E_r\}$ and $\{E''\} = \{E_{r+1}, ..., E_n\}$ be two disjoint subsets of $\{E\}$. Let G' and G" be generalizations of $\{E'\}$ and $\{E''\}$ such that G' $\vee$ G" does not recognize $\{CE'\}$.

*Let $\{E'\} = \{E_1, E_3\}$ and $\{E''\} = E_2$.*
*$G' =$ (COLOR = ANY) & (SIZE = VERY-BIG) & (TEXTURE = ANY)*
*$G'' = E_2$.*
*The disjunction $G' \vee G''$ does not recognize CE.*

Therefore, we have solved our initial problem of recognition of counter-examples by a generalization of the examples.
This solution, which seems to be a little artificial, allows also to favor noise-resistant generalizations. This follows from the fact that there are usually many different partitions of {E} that will generate disjunctive generalizations rejecting the counter-examples.

*Let $\{E'\} = \{E_2, E_3\}$ and $\{E''\} = E_1$. This partition generates an other disjunctive generalization that rejects CE, namely :*
*$GG' =$ (COLOR = GREEN) & (SIZE = LARGE) & (TEXTURE = HARD)*
*$GG'' = E_1$.*
*and $GG' \vee GG''$ rejects CE.*

*On the contrary, the last partition, $\{E'\} = \{E_1, E_2\}$ and $\{E''\} = E_3$ generates the generalization $G \vee E_3$, which of course recognizes CE.*

Let $\{G_1', \ldots, G_t'\}$ be the set of such disjunctive generalizations.

*In our example, $G_1' = G' \vee G''$, $G_2' = GG' \vee GG''$.*

Some of the descriptors, out of which the generalizations are made, may be more or less noise-resistant.
One will choose in this set the generalization that is the most noise resistant, as shown by the two following rules. Both of them rely on the fact that one can discriminate the noise resistance of the descriptors.

*In our example, we consider only the noise issued from descriptor polymorphy, and follow everyday intuition. Let us accept that there is no polymorphy between RED and GREEN, some between HARD and SOFT, and much between BIG and VERY-BIG. It follows that we consider that the colors are not noisy, the texture somewhat noisy, and the sizes very noisy.*

**Rule-1** ( purely symbolic ).

For each $G_i'$, consider the set of descriptors that discriminate {E} against {CE}. They are the important descriptors that contain the features typical to {E}, and atypical to {CE}.
Rule-1 is then : choose the $G_i'$ whose discriminant descriptors are the most noise-resistant.

*In $G_1'$ : [(G' = (COLOR = ANY) & (SIZE = VERY-BIG) & (TEXTURE = ANY)) $\vee$ (G'' = $E_2$)], G' rejects CE because of the descriptor* SIZE, *and G'' rejects CE because of the descriptor* COLOR.
*In $G_2'$ : [(GG' = (COLOR = GREEN) & (SIZE = LARGE) & (TEXTURE = HARD)) $\vee$ (GG'' = $E_1$)], GG' rejects CE because of the descriptor* COLOR *and GG'' rejects CE because of the descriptors* SIZE *and* TEXTURE
*It follows that $G_2'$ uses more noise-resistant descriptors, since the SIZE used in GG'' is helped by a supplementary difference in TEXTURE.*

*Rule-1 would lead us to choose $G_2'$ as correct, noise-resistant, disjunctive generalization.*

**Rule-2** ( purely numeric ).

It may happen that Rule-1 is not operative because the disjunctive generalizations use the same

descriptors.

*Imagine, in our example, that TEXTURE does not appear in $G_2'$.*

Let $G_i'$ and $G_j'$ be two disjunctive generalizations that cannot be ordered by Rule-1.
As we have already seen in our example, different discriminating descriptors are issued from the different clusters where the generalisations come from.
Call $\{\{E_i'\}, \{E_i''\}\}$ the partition of {E} which generates $G_i'$ and call $\{\{E_j'\}, \{E_j''\}\}$ the partition of {E} which generates $G_j'$.
Let us call P' the descriptor, common to $\{E_i'\}$ and $\{E_j''\}$, that discriminates {E} from {CE}, and call P" the descriptor, common to $\{E_i''\}$ and $\{E_j'\}$, that discriminates {E} from {CE}.

For the sake of clarity, suppose further that P' is less noisy than P", and that
$\{E_i'\}$ and $\{E_j''\}$ contain more elements than $\{E_i''\}$ and $\{E_j'\}$.

Then P' is less noisy than P", and it is issued from a statistically more significant subset of examples.

Rule-2 is : whenever possible, choose the disjunctive generalization that makes use of discriminant descriptors that are both the most statistically significant and the less noise-sensitive.

*Imagine, in our example, that TEXTURE does not appear in $G_2'$, the two disjunctive generalizations would then use the same set of descriptors.*

*In $G_1'$, the more noisy descriptor, SIZE, is the one which is issued from $E_1$ and $E_3$. It does not fit the conditions of Rule-2.*
*In $G_2'$, the less noisy descriptor, COLOR, is issued from $E_2$ and $E_3$. This gives us a second reason for choosing $G_2'$ as noise-resistant disjunctive generalization.*

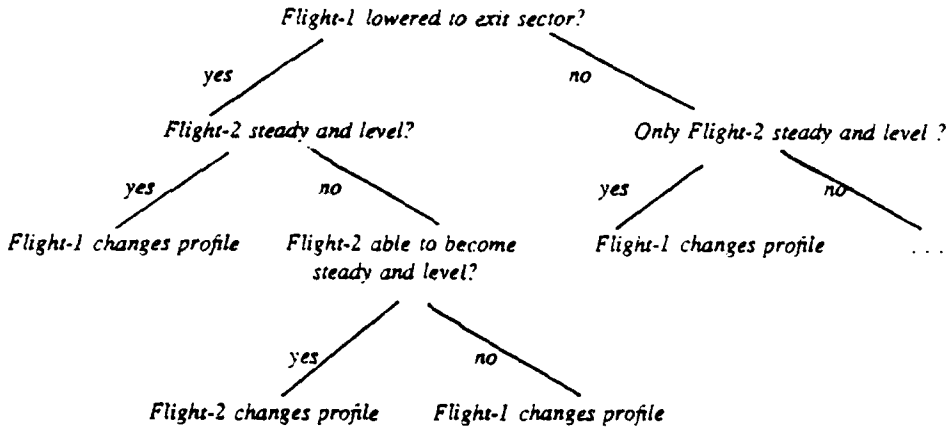
The above technique allows to combine numeric data about the number of examples covered by a description, numeric or symbolic data about the noise associated to each descriptor, and symbolic data about disjunctive generalizations.

### 2.3 - Learning noise-resistant strategies

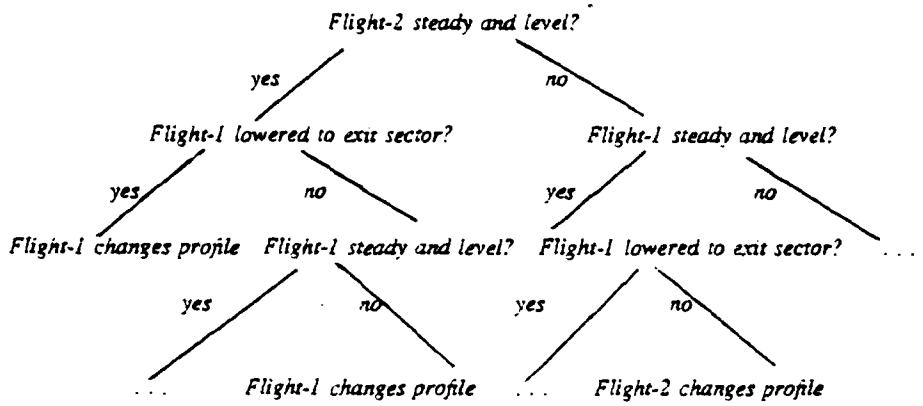
When a sequence of commutative operators has to be applied to achieve a goal, it is a matter of strategy to decide in which order the operators must be applied.
Similarly, the classical "conflict resolution" of the System Experts is nothing but a strategical choice about what is the operator to apply next, when several are available.

*This problem can be illustrated by the following example, taken from [Bisseret & Girard 1973]. It simulates the controls necessary when two planes are exiting an air-traffic sector. There are two conflicting flights, and the problem is to find which must change its flight profile.*

*One can ask as first question wether Flight-1 must be lowered to exit sector. A part of the decision tree is then :*




*One can also ask first wether Flight-2 is steady and level. A part of the corresponding decision tree is then the following.*




*The choice between these two strategies, so claim the specialists in air-traffic control, is not due to any noise ( and so we hope ! ), but to an estimation of the complexity of the exact calculations in each case.*
*Form our AI point of view, noise and calculation complexity can well be confused.*

As the above example shows, depending of the computation complexity, or noise, relative to the answer to *Flight-2 steady and level?*, it is wise or not to ask it as first question.

More generally, it is quite evident that strategies should be adapted to the noise of the descriptors they use. The less noisy descriptors should be used as early as possible, and the most noisy ones, as late as possible.

Before discussing some solutions to the problem of the obtainment of these strategies, let us first point out that, again, a "pure" symbolic problem, viz. the obtainement of variable strategies, is one of the solutions to noise handling.

**How to obtain sets of strategies?**

1 - From Human Experts.

The above strategies for flights exiting a sector was directly given by the expert.
More generally, our experience shows that human experts quite dislike providing rules, which they are almost exclusively asked, and just love providing strategies, which they are not asked.
This is why we have devised a system called DISCIPLE [Kodratoff & Tecuci 1986a, 1986b] which is oriented towards the learning of strategies on the conditions in which rules must be applied.

In this system, the rules must be given, at least in an instantiated form, and conditions for their applications are learned through a conversational interaction with an expert.
The system "guesses" the condition of application of a rule, then, from its data basis, it applies the guessed condition to its knowledge. It therefore proposes instantiated rules to the expert. When the expert accepts them, the conditions of application are confirmed and accordingly generalized, when the expert rejects them, the condition of application are accordingly particularized.

2 - Automatic generation of strategies

When one generates automatically recognition functions, they can be, as done by [Michalski & Chilanski 1980], [Michalski & al. 1982] used as conditions for rule application.

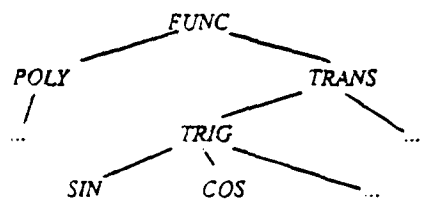
In these references, it is very clear that large rules, concluding to an action from a large set of conditions, are looked for. It could be very useful to look for intermediary clusters of examples and counter-examples that could provide intermediary rules, as for instance done by [Fu & Buchanan 1985].
In this way, which merges automatic generation of recognition functions and conceptual clustering, it could be quite possible to generate automatically sets of possible strategies, that could be used to be adapted to the noise condition in each particular application.

### 2.4 - Polymorphic Version Space

The notion of Version Space has been introduced by T. Mitchell [Mitchell 1982] who describes it as a set of possible generalization states. Let us recall briefly Mitchell's results.

One generalizes from the examples, and the subset of "maximally specific generalizations" obtained by generalization from the examples is called the S-set. In this paper, we shall use the extension of Mitchell's ideas due to Utgoff [Utgoff 1986], and suppose that intermediary concepts ( called "bias" by Utgoff ) are always available.
One particularizes from counter-examples, and the subset of "maximally general generalizations" obtained from the counter-exemples is called the G-set.

*In order to illustrate it, let us use the following example, from [Mitchell, Utgoff & Banerji 1983].*




*If the examples are instances of SIN and COS, then the S-set is made of all the sons of TRIG by fol-*

*lowing Mitchell, while Utgoff allows us to suppose that there is always some intermediary concept ( let us call it here : SIN ∨ COS ) that makes the S-set nearer to the examples.*

*If the counter-examples are instances of POLY, then the G-set is made of the sons of TRANS.*

A first order logic presentation of Mitchell's ideas will allow us to discuss their generalization to noisy data.

Let P be the first order logic predicates that expresses the success of action done is the situation $A_1$.

*In the above example, imagine that one is concerned with symbolic integration, and that one disposes of a set of possible operators, among them, an operator of integration by parts :*

$$OP_2 : \int udv = uv - \int vdu.$$

*Then $A_1$ = [Functional part of dv = SIN], and P = [Success of Integration by Parts by applying $OP_2$]. For the sake of brevity, we leave implicit in the rest this section that the Integration by Parts is always done by using $OP_2$.*

Let $\{A_i\}$ be the set of the situations that insure success during the training phase, then each $A_i$ is such that $A_i \Rightarrow P$. Therefore, each $A_i$ is a sufficient condition for the validity of P.
The S-set ( with the bias extension ) is therefore a set of sufficient conditions for a success.

*Functional part of dv = SIN is a sufficient condition for the integration by part to succeed. Functional part of dv = COS also.*

Let $\{A_i'\}$ be the set of the actions that insure failure during the training phase, and let $\{CA_i'\}$ the complementary set to $\{A_i'\}$ . The set $\{CA_i'\}$ is the G-set for the value of u in the Integration by Parts.

Otherwise stated, given $\{A_i'\}$, one tries to find an other subset $\{CA_i'\}$ such that, for each i,

$$A_i' \Leftrightarrow \neg CA_i'$$

Since each $A_i'$ is a failure, it is such that

$$A_i' \Rightarrow \neg P.$$

It trivially follows that

$$P \Rightarrow CA_i'$$

Therefore, each $CA_i'$ is a necessary condition for the validity of P. The G-set is therefore a set of necessary conditions for a success.

*Consider the above hierarchy. Since POLY and TRANS are two different sons of the same father, one knows that, in well-behaved taxonomies, they exclude each other, i.e. that POLY ⇔ ¬TRANS.*

*Since POLY is a counter-example to the success of the integration by parts with Functional part of dv = POLY, the G-set of the integration by part is Functional part of dv = TRANS and its sons.*

*In greater details, one can see that*

$$[\text{Functional part of } dv = POLY] \Rightarrow \neg[\text{success of Integration by Parts}].$$

*Because of the taxonomy, one has*

$$POLY \Leftrightarrow \neg TRANS.$$

*Using the classical fact that*

$$A \Rightarrow B \text{ is equivalent to } \neg A \vee B.$$

*one has*

$$\neg[\text{Functional part of } dv = TRANS] \vee \neg[\text{success of Integration by Parts}]$$

*which is equivalent to*

*[Functional part of dv = TRANS] ∨ ¬[success of Integration by Parts]*

*and, therefore, to*

*[success of Integration by Parts] ⇒ [Functional part of dv = TRANS].*

The interest of this small theoretization lies in the fact that it gives the main two hypothesis under which Version Spaces are tractable.

**On the one hand**, one must have ¬¬A = A, i.e., one must not use intuitionistic logics. This restriction is not so strong in practice.

Nevertheless, the following example will show that one must be careful while using negation in a reasoning step.

*Suppose that one is working with red, green, and blue colors.*
*Let r be a particular red color. Then a possible negation of this red color, ¬r, may be a particular green color, say g. Now, possible negations of g are of course r, but also any other color which not green, for example a particular blue, b.*
*In that case, ¬¬r may take the value b instead of r as one could expect.*
*One has to suppose that a special care is taken for tracking the origin of the negations when double negation is applied, in order to insure the validity of classical logics.*

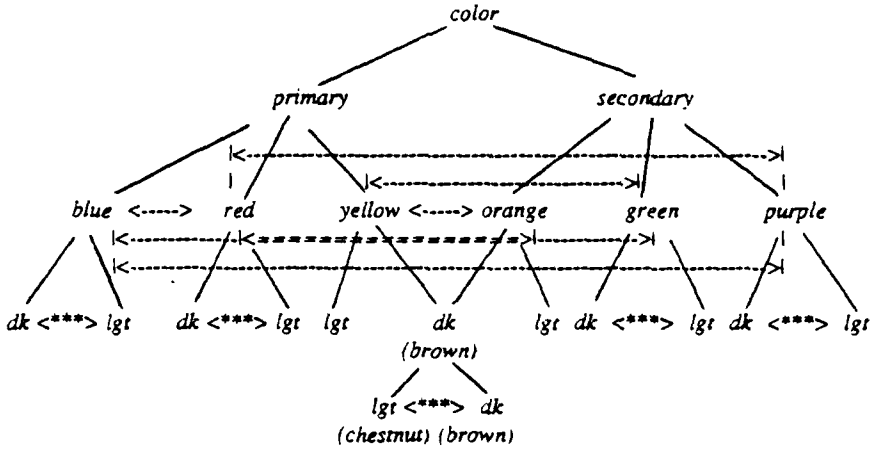
**On the other hand**, in order to build the G-set, one must find counter-examples, $A'_i$, that exclude some other predicates, i.e. such that $A_i \Leftrightarrow \neg CA'_i$.
In this case, concept polymorphy, i.e. the fact that concepts are not always disjoint, will prevent an easy building of the G-set.

*We shall illustrate now our claims with concepts of colors, which clearly are partially polymorphic.*
*For instance, red and rose are polymorphic, because some of their instances may be confused, but red and green are not.*

*The colors are then not represented by a unique point in a state space, but by a set of the possible instances of each color.*

One will have to construct a taxonomic-like tree, similar to the one of the Version Space. Many more links, indicating a partial polymorphy will have to be added to the taxonomy. Coefficients can be associated to each link, indicating how much important the polymorphy is. We insist that the existence of coefficients, and the way they are combined, is not the main issue. On the contrary, the main issue is to keep track of the successive steps of reasoning, in order to be able to provide explanation to the user.
This last point has been explained in [Duval & Kodratoff 1986] in the context of uncertain reasoning.

*Consider the following taxonomy for colors, and the associated links of polymorphy. No horizontal link between two concepts means no polymorphic links.*
*Strong polymorphy is marked by a mere confusion of concepts as* brown *below. Medium polymorphy is marked by a horizontal line of - . Small polymorphy is marked by a horizontal line of * . dk stands for dark, lgt for light.*

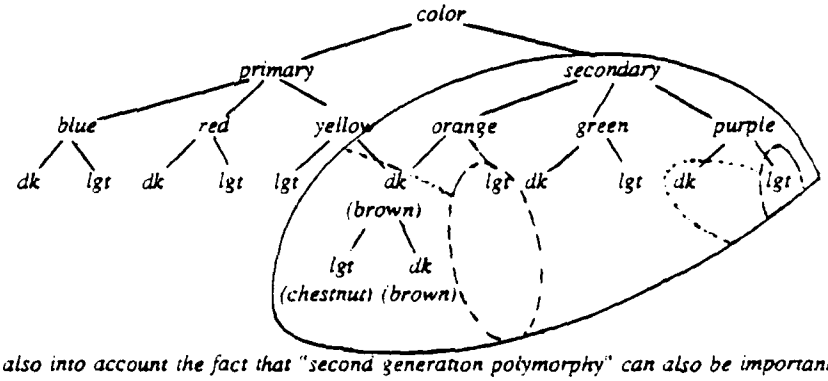



*It is generally understood that, for instance, primary and secondary colors cannot be confused, i.e. that* primary ⟺ ¬secondary. *This is why there is no horizontal line between* primary *and* secondary. *Nevertheless, polymorphy of their sons will induce some ( implicit ) polymorphy between them.*

When a counter-example is given, one will have to check which predicates can be truly considered as behaving classically, i.e. they have no polymorphy with the counter-example predicate. One will have also to keep track of partial polymorphy that tells that they are partially only rejected by the counter-example.

*Suppose that the above taxonomy is used to allow or reject some action P, and that the color light red is a counter-example : light red ⟹ ¬P.*
*The construction of the G-set will proceed as follows.*

*Let us first suppose that "first generation" polymorphy only is considered.*
*Light red is a primary color, polymorphic with some secondary colors, namely purple light and orange light. It follows that the G-set contains all secondary colors, except these two, as in the figure below.*




*Let us now take also into account the fact that "second generation polymorphy" can also be important.*
*Light red is also feebly polymorphic with dark red which, in turn, is polymorphic with dark purple and dark orange.*
*Therefore, dark purple and dark orange must be also excluded of the G-set, but with much less strength than their light counterparts.*

*Now, and only now, some way of combining coefficients will be necessary to have a correct modelling of the necessary knowledge.*

In Mitchell's Version Space, the G-set is a one-dimensional entity since one point, the node where it starts, is necessary.
If no combination of polymorphy is allowed, a two dimensional G-set must be used, in order to tell which predicates are excluded from it, as seen in the above figure.
If one allows combinations of polymorphy coefficients, a three dimensional G-set becomes necessary. The third dimension tells the intensity with which the predicates belong to the G-set.

The same kind of work can be done for the the S-set. One will obtain in a similar way a three dimensional Version Space.

One knows that the S-set and the G-set must coincide in order to obtain hat we call here : necessary and sufficient conditions for the application of the rules.
In the case of polymorphic Version Space, the same is true, but the coincidence can be approximate only, and must hold between complicated shapes.

In general, the G-set and the S-set will not coincide but simply intersect. In most cases, one will have even nothing but information about the likeliness of this intersection.

Therefore, the global information about noise or polymorphy will not be totally contained in belief coefficients only.

Numeric coefficients are of course necessary to convey the information about the likeliness of the intersection, but one must be aware that it would be wrong to forget the essential information conveyed by our extension of the Version Spaces, which can be described as follows.
Let $P_1$ be a predicate belonging to the G-set only, $P_2$ be a predicate belonging to the S-set only, and $[P_1, P_2]$ be the set of predicates that are sons of $P_1$ and fathers of $P_2$.
Then, the exact generalization state is unknown, but belongs to $[P_1, P_2]$.
This last sentence is a way of describing uncertainty by a purely symbolic method which could never have been imagined without Mitchell's noise-free Version Spaces.

## CONCLUSION

In a recent paper, we claim that AI is not a sub-field of Computer Science, but a new Science by itself [Kodratoff 1986b], independant from its parents Mathematics, Logics, and of course Computer Science.
We shall simply recall here our main argument : AI has its self well-identified field of research, namely the definition, measurement and applications of *explanations* given in the own language of its user. In other words, while all other Sciences provide explanations in their own language ( very often they are even able to become rather esoteric ! ), the topic of AI is to reach a point where it can provide explanations in the own language of the user of AI.

We do not want to argue this point here, but would rather try to show that the rest of this paper, in a perhaps indirect way, tries to help achieving this goal. Even if the reader disagree with our position that AI is the science of explanations, he can still discuss our point that a better definition of generalization ( section 1 of this paper ) and a systematic use of symbolic techniques ( section 2 of this paper ) are good tools to achieve a better explicativeness of AI systems.

**About section 1**, one may well wonder what its content may have to do with explicableness, since it looks like theoretical discussions about a formal definition of generalization. Of course, one can argue in a very abstract way that better definitions always lead to better understandability. In the case of generalization, it is by itself a kind of explanation of why are the examples *similar*. Refining generalization may help to remember some hidden common feature which can be an explanation. Counter-examples will be necessary to decide what is and what is not an explanation.

*Recall the examples of section 1.7, $E_1$ = MAMMALIAN(A) & BRED_ANIMAL(A), $E_2$ = TAME(B) & VIVIPAROUS(B), from which we could find the generalization G = TAME(x) & MAMMALIAN(x).*
*Suppose now that a counter-example to G is CE = DANGEROUS(LION-1). From it, we can now tell that eventhough ( implicitly ) present in both examples, MAMMALIAN and TAME are not the good explanation of the link between the examples. One has to use $R_3$: $\forall x$ [TAME(x) $\Rightarrow$ HARMLESS(x)], and some knowledge of the kind $\forall x$ [DANGEROUS(x) $\Rightarrow$ $\neg$HARMLESS(x)], to be able to explain that this examples are about harmless animals. Without introducing TAME in $E_1$, by the use of $R_1$, one would have been unable to find this explanation.*

Our refinements to generalization are not explanatory by themselves, but they may allow to start explanatory processes.

**About section 2**, its content is much more evidently linked to explicableness. Symbolic techniques keep the kind of information that provides explanations while numeric ones ( and especially coefficient combinations ) do not.

*As an example, consider the LEX system which is capable of carrying out formal integrations [Mitchell, Utgoff & Banerji 1983]. As seen in section 2.4, the learning part of LEX as been trying to make identical the G-set and the S-set of ' u ' and ' dv ' in integration by parts. Suppose that it succeeded by finding that these common G-and-S-sets are ' polynomial ' for ' u ' and ' trigonometric ' for ' dv '.*
*Suppose now the system is asked to integrate* $3x \cos x \, dx$ *and that it chooses to integrate by parts with* $u = 3x$ *and* $dv = \cos x \, dx$.
*It is, at least in principle, capable of explanations in the sense that, it is capable, when asked the question: "Why have you chosen this way of integrating?", of giving the answer: "Because I had the option of choosing a ' u ' which is a polynomial and a ' dv ' whose functional part is a trigonometric function.*

The symbolic handling of knowledge about necessary and sufficient conditions makes possible this kind of explanation.

*As a kind of counter-example, imagine that it could be quite possible to achieve also very good results in symbolic integration by asserting coefficients to the possible ' u ' and ' dv ' in integration by part, and learning by increasing the coefficents in case of success, and decreasing them in case of failure. No explanations can been given from this kind of learning.*

**Acknowledgements**

The research described in this paper has been partly funded by ESPRIT-AIP project INSTIL, partly by GRECO & PRC Intelligence Artificielle.
Marta Franova, Michel Manago and Jean-Francois Puget have nicely helped me to put my ideas in a clearer form.

## REFERENCES

[Biermann & al. 1984] Biermann A.W., Guiho G., Kodratoff Y. ( eds ) *Automatic Program Construction Techniques*, Macmillan Publishing Company, New York 1984.

[Bisseret & Girard 1973] Bisseret A., Girard Y., "Organigramme de raisonnement du contrôleur, détection de conflits et recherche de solutions", Rapport IRIA CO-7303-R37, 1973.

[Cannat & Kodratoff 1986], J-J. Cannat, Y. Kodratoff, "Learning Techniques Applied to Multi-font Character Recognition", Applications of Artificial Intelligence III, John F. Gilmore, Editor, Proc. SPIE 635, p. 462-479, 1986

[Cannat & al. 1986] J.J. Cannat, S. Moscatelli, Y. Kodratoff, "Learning Techniques Applied to Multi-font Character Recognition", Proc ICPR-8, Paris 1986, pp. 123-125.

[Cloksin & Mellish 1981] Cloksin W.F., Mellish C.S. *Programming in Prolog*, Spronger-Verlag, Berlin 1981.

[Duval & Kodratoff 1986] Duval B., Kodratoff Y., "Automated Deduction in an Uncertain and Inconsistent Data Basis", Proc. ECAI-86, Brighton 1986, pp. 101-108.

[Fu & Buchanan 1985] Fu L.-M., Buchanan B. G., "Learning Intermediate Concepts in Constructing a Hierarchical Knowledge Base", Proc. IJCAI-85, Los Angeles 1985, pp. 659-666.

[Franova 1985] Franova M. " CM-strategy : A Methodology for Inductive Theorem Proving or Constructive Well-Generalized Proofs " A. Joshi (ed), Proceedings of the Ninth International Joint Conference on Artificial Intelligence, August 1985, Los Angeles, pp. 1214-1220.

[Franova 1986] Franova M. " Proving Theorems by Construction of Valid Formulae ", *Information Processing 86*, in Proceedings of the IFIP'86 Congress, H.J. Kluger editor, North Holland 1986, pp 41-46.

[Hsiang 1982] Hsiang J., "Topics in Automated Theorem Proving and Program Generation", *Ph.D. thesis*, Dept. Comp. Sci. Univ. Illinois, Urbana, November 1982.

[Kodratoff 1983] Kodratoff Y., "Generalizing and Particularizing as the Techniques of Learning", *Computers and Artificial Intelligence 2*, pp. 417-441, 1983 .

[Kodratoff & Lemerle-Loisel 1984] Kodratoff Y., Lemerle-Loisel R., "Learning complex structural descriptions from examples". Computer Vision, Graphics, and Image Processing 27, pp. 266-290 (1984).

[Kodratoff & al. 1984] Kodratoff Y., Ganascia J.-G., Clavieras B., Bollinger T., Tecuci G., "Careful generalization for concept learning", Proc. ECAI-84, pp. 483-492, Pisa 1984. Now also available in *Advances in Artificial Intelligence*, T. O'Shea ed., North-Holland 1985, pp.229-238.

[Kodratoff 1985] Kodratoff Y., "Une théorie et une méthodologie de l'apprentissage symbolique", Actes *COGNITIVA 85*, pp. 639-651, Paris Juin 1985.

[Kodratoff 1986a] Kodratoff Y. *Leçons d'Apprentissage Symbolique Automatique*, Ed. CEPADUES, Toulouse 1986.

[Kodratoff 1986b] Kodratoff Y. "Is AI a Sub-Field of Computer Science ? or AI is the Science of Explanations", Rapport de Recherche LRI n0 312, 1986.

[Kodratoff & Ganascia 1986] Kodratoff Y., Ganascia J.G. "Improving the generalization step in learning",in *Machine Learning : An artificial intelligence approach, Vol. 2*, Michalski R. S., Carbonell J. G., Mitchell T. M. eds, Morgan Kaufmann 1986, pp. 215-244.

[Kodratoff & Tecuci 1986a] Kodratoff Y., Tecuci G., "Rule Learning in DISCIPLE", Proc. EWSL-86, Orsay, Feb. 1986.

[Kodratoff & Tecuci 1986b] Kodratoff Y., Tecuci G., "DISCIPLE : An Interactive Approach to Learning Apprentice Systems", LRI Internal publication n0 293, Aug. 86.

[Kodratoff & al. 1986] Kodratoff Y., Manago M., Blythe J., Smallman C. "Generalization in a Noisy Environment", Proc. AAAI workshop on Knowledge Acquisition, Banff 1986.

[Langley 1986] Langley P. "The terminology of Machine Learning" Machine Learning 1, 1986, 141-144.

[Manago 1986] Manago M., "Object Oriented Generalization : A Tool for Improving KNowledge-Based Systems", Proc. IMAL-86, Les Arcs, 1986, pp.93-104.

[Michalski & Chilauski 1980] Michalski R. M., Chilauski R. L. "Learning by Being Told and Learning from Examples : An Experimental Comparison of the Two Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybean Disease Diagnosis", *Internatl. J. of Policy Analysis and Information Systems 4*, 1980.

[Michalski & al. 1982] Michalski R. M., Davis J. H., Bisht V. S., Sinclair J. B. "PLANT/ds : An Expert Consulting System for the Diagnostic of Soybean Diseases", Proc. ECAI-82, Orsay 1982, pp. 133-138.

[Michalski & Stepp 1983] Michalski R. S., Stepp R. E. "Learning from Observation : Conceptual Clustering", in *Machine Learning, an Artificial Inteiligence Approach*, Michalski R.S., Carbonell J.G., Mitchell T.M. eds, Tioga Publishing Company 1983, pp. 163-190.

[Michalski 1984] Michalski R.S., "Inductive Learning as Rule-guided Transformation of Symbolic Descriptions : a Theory and Implementation", in *Automatic Program Construction Techniques*, Biermann A.W., Guiho G., Kodratoff Y. eds, Macmillan Publishing Company, 1984, pp. 517-552.

[Michalski 1986] Michalski R.S., "Two-Tiered Concept Meaning, Inferential Matching and Conceptual Cohesiveness", *J. of Education Comp. Res. 2*, 1986, to appear.

[Mitchell 1982] Mitchell T.M. "Generalization as Search", Artificial Intelligence 18, 1982, 203-226.

[Mitchell, Utgoff & Banerji 1983] Mitchell T.M., Utgoff P.E., Banerji R. "Learning by experimentation, acquiring and refining problem-solving heuristics", in *Machine Learning, an Artificial Intelligence Approach*, Michalski R.S., Carbonell J.G., Mitchell T.M. eds, Tioga Publishing Company 1983, pp. 163-190.

[Mitchell & al. 1986] Mitchell T. M., Keller R. M., Kedar-Cabelli S. T., "Explanation-Based Generalizations : A Unifying View", Machine Learning 1, 47-80, 1986.

[Mokhtarian & Mackworth 1986] Mokhtarian F., Mackworth A., "Scale-Based Description and Recognition of Planar Curves and Two-dimensional Shapes" IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-8, 34-43, 1986.

[Nicolas 1986a] Nicolas J., "Les stratégies de contrôle dans l'apprentissage à partir d'exemples",

Comptes-Rendus JFA-1986, Rapport de Recherche LRI 259, 1986.

[Nicolas 1986b] Nicolas J., "Learning as Search : A Logical Approach", Proc. CIIAM'86, Hermes Paris 1986, pp.443–459.

[Plotkin 1970] Plotkin, G.D., "A Note on Inductive Generalization", in *Machine Intelligence*, Meltzer, B. and Michie, D. (Eds.), American Elsevier, New York, 1970.

[Stickel 1981] Stickel M.E., "A Unification Algorithm for Associative-Commutative Functions", *J. ACM* 28, pp. 423–43, 1981.

[Utgoff 1986] Utgoff P.E., "Shift of Bias for Inductive Concept Learning", in *Machine Learning : An artificial intelligence approach. Vol. 2*, Michalski R. S., Carbonell J. G., Mitchell T. M. eds, Morgan *Kaufmann* 1986, pp. 107-148.

[Vrain 1987] Vrain C. Forthcoming thesis, Univ. Paris-Sud Feb. 1987

DE JANVIER 86 A



257 ALGEBRAIC SEMANTICS OF EXCEPTION HANDLING
G. BERNOT, M. BIDOIT, C. CHOPPY (Janvier 1986) (28 pages)

258 PROCEEDINGS OF THE EUROPEAN WORKING SESSION ON LEARNING : EWSL 86
3-4/02/1986 (260 pages) 130 F

259 COMPTE RENDUS DE LA JOURNEE FRANCAISE SUR L'APPRENTISSAGE : JFA 86
5/02/1986 (60 pages)

260 SERIALIZATION OF CONCURRENT PROGRAMS
M.P. FLE (Janvier 1986) (35 pages)

261 ARETE-VULNERABILITY DU DIAMETRE DANS LES RESEAUX D'INTERCONNEXION
R. KERJOUAN (Janvier 1986) (43 pages)

262 k-LINKED AND k-CYCLIC DIGRAPHS
Y. MANOUSSAKIS (Janvier 1986) (22 pages)

263 ON FINDING A PARTIAL SOLUTION TO A LINEAR SYSTEM OF EQUATIONS IN POSITIVE INTEGERS
J. L. LAMBERT (Février 1986) (13 pages)

264 FREE CHOICE FIFO NETS
A. FINKEL (Février 1986) (19 pages)

265 SUFFICIENT CONDITIONS FOR MAXIMALLY CONNECTED DENSE GRAPHS
T. SONEOKA, H. NAKADA, M. IMASE, C. PEYRAT (Mars 1986) (20 pages)

266 HAMILTON CYCLES IN REGULAR 2-CONNECTED GRAPHS
J. A. BONDY, M. KOUIDER (Mars 1986) (12 pages)

267 USING DETERMINISTIC FUNCTIONS IN PROBABILISTIC ALGORITHMS
M. SANTHA (Mars 1986) (15 pages)

268 METHODE DE TRANSFORMATION DE PROGRAMMES DE BURSTALL-DARLINGTON APPLIQUEE A LA PROGRAMMATION LOGIQUE
N. AZIBI, Y. KODRATOFF (Mars 1986) (86 pages) 43 F

269 ON THE EFFECT OF JOIN OPERATIONS ON RELATION SIZES
D. GARDY, C. PUECH (Avril 1986) (35 pages)

270 AUTOMATIC PROGRAMMING TECHNIQUES APPLIED TO SOFTWARE DEVELOPPEMENT AN APPROACH BASED ON EXCEPTION HANDLING
M. BIDOIT, C. GRESSE, F. LOSAVIO, P. SCHLIENGER (Avril 1986) (28 page

271 CIRCUMFERENCE AND HAMILTONISM IN K. ,-FREE GRAPHS
E. FLANDRIN, I. FOURNIER, A. GERMA (Avril 1986) (16 pages)

272 N-TANGO OU LE TRAITEMENT DU CONNECTEUR NON DANS TANGO
G. KASSEL (Avril 1986) (41 pages)

273 COVERING THE VERTICES OF A GRAPH BY CYCLES OF PRESCRIBED LENGTH
D. AMAR, I. FOURNIER, A. GERMA (Avril 1986) (17 pages)

274 HAMILTONIAN DICYCLES AVOIDING PRESCRIBED ARCS IN TOURNAMENTS
P. FRAISSE, C. THOMASSEN (Mai 1986) (13 pages)

275 A NOTE ON CIRCUMFERENCE AND INDEPENDENT SETS
P. FRAISSE (Mai 1986) (10 pages)

276 $D_\lambda$-CYCLES AND THEIR APPLICATIONS FOR HAMILTONIAN GRAPHS
P. FRAISSE (Mai 1986) (26 pages)

277 DEGREE MINIMUM ET CYCLES DE LONGUEUR DONNEE
D. AMAR, E. FLANDRIN, I. FOURNIER, A. GERMA (Mai 1986) (14 pages)

278 APPLICATIONS OF RESIDUES FOR THE ANALYSIS OF PARALLEL SYSTEMS COMMUNICATING BY FIFO CHANNELS
A. CHOQUET, A. FINKEL (Mai 1986) (25 pages)

279 PANCYCLISM IN CHVATAL-ERDOS'S GRAPHS
D. AMAR, I. FOURNIER, A. GERMA (Mai 1986) (20 pages)

280 DIFFUSION D'INFORMATIONS DANS UN RESEAU DISTRIBUE APPLICATION A L'EXCLUSION MUTUELLE
J. C. KONIG (Mai 1986) (27 pages)

281 FAULT TOLERANT ROUTINGS IN KAUTZ AND DE BRUIJN NETWORKS
N. HOMOBONO, C. PEYRAT (Mai 1986) (13 pages)

282 PANCYCLISM IN $K_{t,3}$-FREE GRAPHS
E. FLANDRIN, I. FOURNIER, A. GERMA (Mai 1986) (21 pages)

283 OBJECT ORIENTED TOOLS FOR THE DESIGN OF HIGH LEVEL INTERFACES : THE KEY FOR ADAPTABILITY
S. KARSENTY (Juin 1986) (17 pages)

284 CONNECTIVITY OF IMASE AND ITOH DIGRAPHS
N. HOMOBONO, C. PEYRAT (Juin 1986) (15 pages)

285 CORRECTNESS PROOFS FOR ABSTRACT IMPLEMENTATIONS
G. BERNOT (Juin 1986) (33 pages)

286 INCORPORATING FUNCTIONAL DEPENDENCIES IN DEDUCTIVE QUERY ANSWERING
N. SPYRATOS, C. LECLUSE (Juin 1986) (25 pages)

287 GRAPHS SUCH THAT EVERY TWO EDGES ARE CONTAINED IN A SHORTEST CYCLE
N. HOMOBONO, C. PEYRAT (Juin 1986) (13 pages)

288 ON A CONCURRENCY MEASURE
J. BEAUQUIER, B. BERARD, L. THEMONIER (Juin 1986) (17 pages)

289 INTERCONNECTION NETWORK WITH EACH NODE ON TWO BUSES
J. C. BERMOND, J. BOND, C. PEYRAT (Juillet 1986) (13 pages)

290 PROCEEDING OF THE INTERNATIONAL MEETING ON ADVANCES IN LEARNING IMAL 86 (Juillet 1986)

291 THE SEMANTICS OF QUERIES AND UPDATES IN RELATIONAL DATABASES
N. SPYRATOS, C. LECLUSE (Juillet 1986) (48 pages)

292 NOMBRE MAXIMUM DE CYCLES DE LONGUEUR EGALE A LA MAILLE
J. BOND, H. THUILLIER (Juillet 1986) (12 pages)

293 DISCIPLE : AN INTERACTIVE APPROACH TO LEARNING APPRENTICE SYSTEMS
Y. KODRATOFF, G. TECUCI (Août 1986) (40 pages)

294 TWO NOTES ON GRAPHS AND SECURITY
C. DELORME, J. J. QUISQUATER (Août 1986) (8 pages)

295 AUTOMATIC PROOFS BY INDUCTION IN THEORIES WITHOUT CONSTRUCTORS
J. P. JOUANNAUD, E. KOUNALIS (Septembre 1986) (28 pages)

296 $\omega$ - LANGUAGES OF PETRI NETS AND LOGICAL SENTENCES
E. PELZ (Septembre 1986) (19 pages)

297 CLOSURE PROPERTIES OF DETERMINISTIC PETRI NETS
E. PELZ (Septembre 1986) (18 pages)

298 LARGE FAULT-TOLERANT INTERCONNECTION NETWORKS
J. C. BERMOND, N. HOMOBONO, C. PEYRAT (Septembre 1986) (22 pages

299 CONCEPTUAL DISTANCE-BASES LEARNING
Y. KODRATOFF, C. TECUCI (Septembre 1986) (27 pages)

300 AN EXTENSION OF FREE CHOICE NETS TO FIFO
A. FINKEL, A. CHOQUET (Septembre 1986) (7 pages)

301 SYSTEMES DISTRIBUES ET AUTOMATES FINIS
J. BEAUQUIER (Septembre 1986) (27 pages)

302 SERPE : AN EXTENSIBLE STRUCTURE FOR ANALYSIS OF PETRI-NETS
P. FRAISSE, C. JOHNEN, N. TREVES (Septembre 1986) (14 pages)

303 NUMBER OF ARCS, CYCLES AND PATHS IN BIPARTITE DIGRAPHS
M. MANOUSSAKIS, Y. MANOUSSAKIS (Octobre 1986) (37 pages)

304 MINIMALISM, JUSTIFICATION AND NON-MONOTONICITY IN DEDUCTIVE DATABASES
N. BIDOIT, R. HULL (Octobre 86)(52 pages)

305 CONCEPTUAL HIERARCHICAL ASCENDING CLASSIFICATION
N. BENAMOU, Y. KODRATOFF (Octobre 1986)(29 pages)

306 ON A CONCURRENCY MEASURE
J. BEAUQUIER, B. BERARD, L. THIMONNIER (Octobre 1986) (16 pages)

307 ON SOME CONJECTURES ON CUBIC 3-CONNECTED GRAPHS
J. L. FOUQUET, H. THUILLIER (Octobre 1986) (21 pages)

308 UNE GENERALISATION DE L'ARBRE DE COUVERTURE AUX SYSTEMES DE TRANSITIONS BIEN STRUCTURES
A. FINKEL, (Octobre 1986) (20 pages)

309 CYCLES AND PATHS OF MANY LENGTHS IN BIPARTITE DIGRAPHS
D. AMAR, Y. MANOUSSAKIS (Octobre 1986) (20 pages)

310 CIRCULATION OF SEMI-FLOWS OF PR/T-SYSTEMS
J. VAUTHERIN (Octobre 1986) (21 pages)

311 A LOGIC FOR DATA AND KNOWLEDGE BASES
N. SPYRATOS, C. LECLUSE (Novembre 1986) (27 pages)

312 IS AI A SUB-FIELD OF COMPUTER SCIENCE ? OR AI IS THE SCIENCE OF EXPLANATIONS
Y. KODRATOFF (Novembre 1986) (21 pages)

313 CONSEQUENCIES OF THE DECIDABILITY OF THE REACHABILITY IN PETRI NETS
J. L. LAMBERT (Novembre 1986) (50 pages)

314 ISOMORPHISMS OF CAYLEY MULTIGRAPHS OF DEGREE 4 ON FINITE ABELIAN GROUPS
C. DELORME, O. FAVARON, M. MAHEO (Novembre 1986) (15 pages)

315 A COMPILER FOR CONDITIONAL TERM REWRITING SYSTEMS
S. KAPLAN (Décembre 1986) (22 pages)

316 SIMPLIFYING CONDITIONAL TERM REWRITING SYSTEMS : UNIFICATION, TERMINATION AND CONFLUENCE
S. KAPLAN (Décembre 1986) (48 pages)

317 SOME NEW COMPOUND GRAPHS
C. DELORME, J. J. QUISQUATER (Décembre 1986) (19 pages)

318 QUATRE ARTICLES SUR L'ETAT DE L'ART EN APPRENTISSAGE SYMBOLIQUE AUTOMATIQUE
FOUR PAPERS ON THE STATE OF THE ART IN MACHINE LEARNING
Y. KODRATOFF, N. GRANER (Décembre 1986) (56 pages)

319 ANALYSING NETS BY THE INVARIANT METHOD
G. MEMMI, J. VAUTHERIN (Décembre 1986) (40 pages)

END

DATE

FILMED

DTIC

JULY 88