Semantic Abstraction for Concept Representation and Learning

Lorenza Saitta

Università di Torino
Dipartimento di Informatica
Corso Svizzera 185
10149 Torino (Italy)
saitta@di.unito.it

Jean-Daniel Zucker

Université Paris VI - CNRS
Laboratoire d'Informatique de Paris 6
4, Place Jussieu
75252 Paris (France)
Jean-Daniel.Zucker@lip6.fr

Abstract

So far, abstraction has been mainly investigated in problem solving tasks. In this paper, we are interested in the role of abstraction in representing and learning concepts (i.e., intensional descriptions of classes of objects). We propose a novel perspective on abstraction, originating from the observation that a conceptualization of a domain involves entities belonging to at least three levels. The fundamental level is the *perception of the world,* where concrete objects reside. For memorizing objects, some kind of *structure,* which describes objects and relations perceived in the world, is needed. Finally, to communicate with others, and also to perform reasoning, a *language* has to be used; the language allows both the world and theories about the world to be described *intensionally*.

In previous approaches, abstraction has been frequently defined as a mapping between languages. Our main departure from this view is that abstraction is, originally, a mapping between views of the world, and that the modifications of the structure and of the language are side-effects, necessary to describe what happens at the level of the perceived world. Within the defined framework, we show how the abstraction process can be realized by means of a set of operators, and we formalize a constraint that abstraction mappings should satisfy in order to be useful for Machine Learning, i.e. preserving the generality relation among hypotheses.

1. Introduction

Abstraction is a pervasive activity in human perception and reasoning. With few exceptions, abstraction has been investigated, up to now, in the problem solving context (Sacerdoti, 1973; Plaisted, 1981; Giunchiglia & Walsh, 1992; Ellman, 1993; Holte et al., 1996). In this paper, we are interested in the role played by abstraction in representing and learning concepts. In particular, we consider the task of supervised learning from examples, which consists in inducing an intensional description (a "hypothesis") of a concept from a set of positive and negative instances of it (Michalski, 1983; Mitchell, 1982).

In Machine Learning, abstraction has been used by Knoblock (1992) to learn high level operators in hierarchical planning, by Drastal, Czako and Raatz (1989) to learn concepts in propositional logic, and by Giordana and Saitta (1990), and Zucker and Ganascia (1994) to learn concepts in predicate logic. Giordana and Saitta (1990) provide a discussion about the differences between generalization and abstraction in learning. There are also significant efforts that have been devoted to the "shift of bias" (Utgoff, 1985), a problem related to abstraction. This problem deals with the modification of the language used to represent the learning examples and the concepts. Many approaches follow the methodology called *constructive induction*, which tries to augment the initial language with newly invented features (Muggleton, 1988; Matheus & Rendell, 1989; Wnek & Michalski, 1994; Zucker & Ganascia, 1994).

Unfortunately, there does not yet exist a general framework that could provide the means for better understanding the different types of abstraction, in particular those useful for Machine Learning. The reason is that in Machine Learning not only deductive but also inductive inferences are used. Frameworks such as the ones proposed by Plaisted (1981) or Giunchiglia and Walsh (1992) are very good at capturing important aspects of many abstractions between formal systems using deductive inferences, but they are not meant to take into account inductive inference (a non truth-preserving inference). We aim at defining a framework whose final purpose is the characterization and justification of useful abstractions for Machine Learning.

To account for the importance of the semantic level in inductive inference, we propose to look at the semantic definitions of abstraction proposed by Tenenberg (1987), and by Nayak and Levy (1995) from the perspective of representing and learning concepts. This perspective can shed new light on abstraction.

2. Related Work

As already mentioned, much of the work on abstraction has been carried out w.r.t. problem solving tasks. Plaisted (1981) has provided the a foundation for theorem proving with abstraction, seen as a function from a set of clauses to another one that satisfies some properties related to the resolution principle. Tenenberg (1987) starts from Plaisted's work to define an abstraction as a *mapping* between predicates that preserves consistency. He defines an abstraction either as a *predicate mapping*, or a mapping between clauses, based on predicate mappings, where only consistent clauses are kept. Holte and Zimmer (1988) worked on an original framework, using the category theory, for analyzing the nature of representation and representation shifts. They proposed in their approach to shift away from the study of structures towards the study of *algebraic mappings* between structures.

Giunchiglia and Walsh (1992) have extended Plaisted's approach and reviewed most of the work done in reasoning with abstraction. Their goal is to define a « *general environment for the use of abstraction in automated deduction»*. They are mostly interested in abstractions that preserve consistency. Nayak and Levy (1995) propose a semantic theory of abstraction that extends Tenenberg's approach to Model Level Mapping rather than Predicate Mapping. Instead of viewing abstraction as a purely syntactic mapping, they view abstraction as a two-step process : the first one consists in a limited abstraction of the domain model, and the second one in constructing a set of abstract formulas to capture the abstracted domain model.

Abstraction has also been studied in relation with change of problem representation (Benjamin et al., 1990). This type of abstraction relies upon the assumption that there are some representations that are more "operational" than others for certain kinds of problem (Amarel, 1983). Subramanian (1990) proposed an approach for performing incremental abstraction in reformulating problems for increasing computational efficiency, based on a notion of *relevance*

(Subramanian, Greiner & Pearl, 1997). Korf (1980) views discovering efficient solution strategies as a heuristic search in the space of problem representations. For Lowry (1987) a good problem representation incorporates important problem constraints while hiding superfluous details. Ellman (1993) is focused on improving performance in solving constraint satisfaction problems. Ellman shows how clustering algorithms can be used to construct hierarchies of abstraction spaces to speed up the search for solutions.

Finally the notion of "granularity" is related to the analysis of possible links between various levels of abstractions. The goal of Hobbs (1985) was to extract smaller theories, computationally more tractable, out of an initial theory, by focusing on the granularity of objects or observations. Imielinski (1987) proposed an approximate reasoning framework for abstraction. He call such kind of reasoning "limited" because it is weaker than the general predicate logic proof method, but it leads to significant computational advantages. Today's research on granularity includes the rough set approach (Pawlak, 1991).

3. Definition of Abstraction Mapping

In this paper we propose a novel perspective on abstraction, originating from the observation that concept representation deals with entities belonging to three different levels. Underlying any source of experience there is the *world (W),* where *concrete* objects (the "real things") reside. However, the world is not really known, because we only have a mediated access to it, through our perception. Then, what is important, for an observer, is not the world *in se,* but the perception *P(W)* that s/he has of it. At this level the percepts "exist" only for the observer and only during their being perceived. Their reality consists in the "physical" stimuli produced on the observer. In order to let these stimuli become available over time, they must be, first of all, memorized into an organized *structure S.* This structure is an *extensional* representation (Van Dalen, 1983) of the perceived world, in which stimuli related one to another are stored together into tables. The set of these tables constitutes a relational database, on which relational algebra operators can be applied. Finally, in order to reason about the perceived world, and to communicate with other agents, a *language* L is needed. L allows the perceived world to be described *intensionally*. At the language level we can operate through inference rules. Let us define R = (P(W), S, L) as a *Reasoning Context.* In this work, we consider only static situations, in which the world, W, does not change with time; this is the usual situation addressed in Machine Learning.

Among the three levels, corresponding to P(W), S and L, respectively, there may be complex links and interplay. The investigation of these links is out of the scope of this paper, because it pertains to philosophy or psychology. We only note that the levels, in the proposed framework, are generated upwards; in fact, P(W) is the primary source of information, S tries to record P(W), and L verbally describes P(W). The relationships among the three considered levels are represented in Figure 4.1.

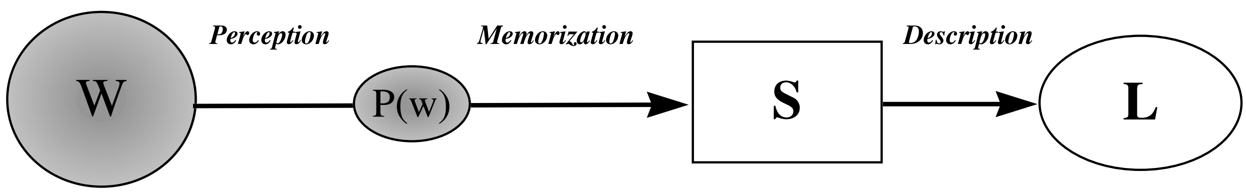

Figure 4.1 — Levels used to reason about the world. In P(W) there are perception of objects and their matter-of-fact links. S is a set of tables, grouping sets of objects sharing some property. L is a formal language, whose semantics is evaluated on S's tables. Finally, T is a theory, in which properties of the world and general knowledge are embedded.

As an instance of the above notions, let us consider the case of several sources, in W, which emit different monochromatic lights. Even though the sources may be unspecified (W is unknown), P(W) contains the retina's stimuli produced by the incoming photons. The visual stimuli can be organized, in S, into a table, where each stimulus (source) is associated, for example, to a wavelength. At this point, symbols may be arbitrarily assigned to each stimulus (e.g., "red", "blue", "green") and to the table itself that contains them (e.g., «color»).

In previous approaches, abstraction has been mostly defined as a mapping between two languages, and the structures have been implicitly considered as sets of models. In our approach, modifications of the structure and of the language must be introduced only to describe the differences occurred in the perception of the world. Perception may change for several reasons: for instance, visual perception may be unwillingly modified by the used tool, be it the eyes, or a pair of sun-glasses, or a binocular. Or, we may purposely ignore details, because we do not consider them relevant for our current goal. In all these cases, the world does not change *per se;* it is only our perception of it that changes. In Machine Learning, the definition of P(W) corresponds to the phase of *feature selection.*

According to the above considerations, we think that abstraction should be defined primarily as a mapping at the level of the perceived world. Given a world W, let P(W) be a perception system consisting of a set of sensors, each one devoted to the reception of a particular signal. Each sensor has a resolution threshold that establishes the minimum difference between two signals in order to consider them as distinct. A set of values provided by the sensors in P is called a *signal pattern* or a *configuration.* Recognizing configurations is important for the observer, because some action may be associated to each configuration. Let Γ be the set of all possible configurations detectable by P(W).

As an example, let us consider a world W constituted by a matrix of $n \times n$ pixels (with n even). Let P(W) be a "retina" with resolution 1 pixel. The retina contains $n \times n$ threshold receptors (pixel is bright or not). Then, the set Γ will be:

$$\Gamma = \{\gamma \mid \gamma : [1, ..., n] \times [1, ..., n] \to \{0, 1\}\} \quad \text{with } |\Gamma| = 2^{n^2}$$

A specific $\gamma \in \Gamma$ is the assignment of a value to each of the n^2 pixels.

Let us now consider another perception system P'(W), acting on the same world, but consisting of a retina with a lower resolution: the retina can only analyze squares of 2×2 pixels and outputs 1 iff at least 2 of the 4 pixels are bright. In this case we have:

$$\Gamma' = \{\gamma' \mid \gamma' : [1, 3, ..., n\text{-}1] \times [1, 3, ..., n\text{-}1] \to \{0, 1\}\} \quad \text{with } |\Gamma'| = 2^{\left(\frac{n}{2}\right)^2}$$

Each configuration γ' can be uniquely obtained by a corresponding configuration γ by a majority voting algorithm. However, given γ', it is not possible to recover uniquely the configuration γ from which γ' has been derived, unless additional information is provided (for instance the explicit link between γ' and γ). We say that P'(W) is "simpler" than P(W), because it offers to the observer less information to manipulate. The price to be paid for this lower cognitive burden (computational load, for an artificial system) is a decreasing in the information about the world. This lack of information can be either good or bad: if the correct action to be performed in response to the recognition of the state of W has to be very fine-tuned, this lack of information may result in the wrong action. On the contrary, if this is not the case, there is no point is handling more information than is needed, because a part of the observer's cognitive effort can be freed for attending another task.

Definition 4.1 – Given a world W, let $P_1(W)$ and $P_2(W)$ be two perception systems for W, and let Γ_1 and Γ_2 be the corresponding configuration sets. The perception system $P_2(W)$ will be said *simpler* than $P_1(W)$ iff:

a) There is a function $f : \Gamma_1 \rightarrow \Gamma_2$ that starting from any configuration $\gamma_1 \in \Gamma_1$ uniquely determines a configuration $\gamma_2 \in \Gamma_2$

b) The inverse function f^{-1} does not exist, because there is at least some configuration γ_1 that cannot be uniquely reconstructed from γ_2, without additional information. ❐

Intuitively, "simpler" means that some information, apparent in P_1, is hidden in P_2. The above definition has the advantage of linking simplicity to its semantic meaning of "work to be done" to manipulate the information, rather to its syntactic expression. Obviously, the syntactic complexity of the language may enter the simplicity evaluation of a perceptual system, when syntactic complexity implies more work to handle the conveyed information.

According to the above considerations, we are now in the position to provide a formal definition of abstraction.

Definition 4.2 – Given a world W, let $R_g = (P_g(W), S_g, L_g)$ and $R_a = (P_a(W), S_a, L_a)$ be two reasoning contexts, which we conventionally label as *ground* and *abstract.* An *Abstraction* is a functional mapping $A: P_g(W) \rightarrow P_a(W)$ between a perception $P_g(W)$ and a simpler perception $P_a(W)$ of the same world W. ❐

The function A allows any abstract configuration to be uniquely determined from a ground one, but not vice versa, without additional information.

For the sake of clarification, we introduce in Figure 2 an illustrative example.

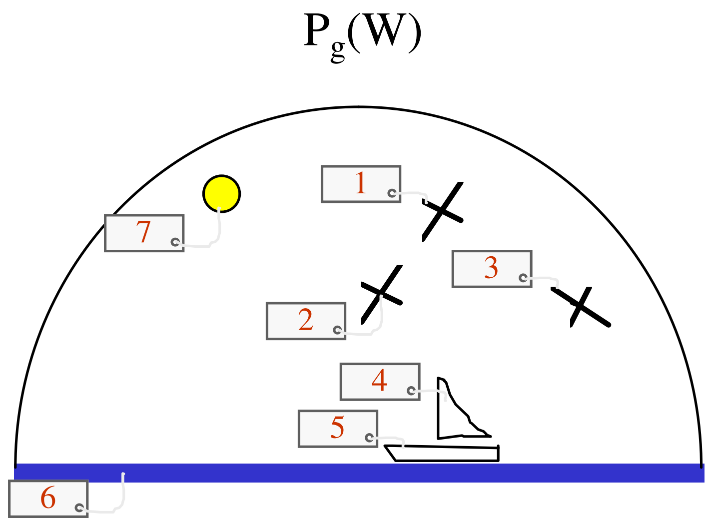

Figure 2 – Example of a $P_g(W)$, the environment perceived by an air traffic controller situated in a tower on a sea shore. S/he is aware of seven entities, each identified with a unique number.

A world may be, in principle, infinitely rich in information. It contains various kinds of objects (*compound,* when composed by several components, or *atomic*), each kind with specific properties. Thus, P(W) can be informally defined as a 4-tuple P(W) = <***OBJ, ATT, FUNC, REL***>, where ***OBJ*** is a set of perceived objects, ***ATT*** is a set of object attributes, ***FUNC*** is a set of functional links, and ***REL*** is a set of relations. For instance, in the perceived world of Figure 2, ***OBJ*** contains seven objects, ***ATT*** contains the attributes "color", "owner" and "type", ***FUNC*** contains a function that gives the distance between two planes and a function that returns the country of an owner, and finally, possible relations in ***REL*** are spatial ones, such that a plane is "close" to another one and that three planes are in priority conflict for landing.

For memorization purpose, the implicit information available in P(W) is organized as a set of tables, forming a relational database, namely the structure S. S extensively identifies attribute values, functions and relations. To each physical object a unique identifier (denoted by an integer or roman number in the ground and abstract world respectively) is associated. While S describes the world extensionally, an intensional description is also needed, in order to facilitate reasoning and communication. Then a logical language L is introduced. The truth of a formula of L is evaluated on the structure S. Figure 3, provides a possible structure S_g and language L_g for the perceived world of Figure 2.

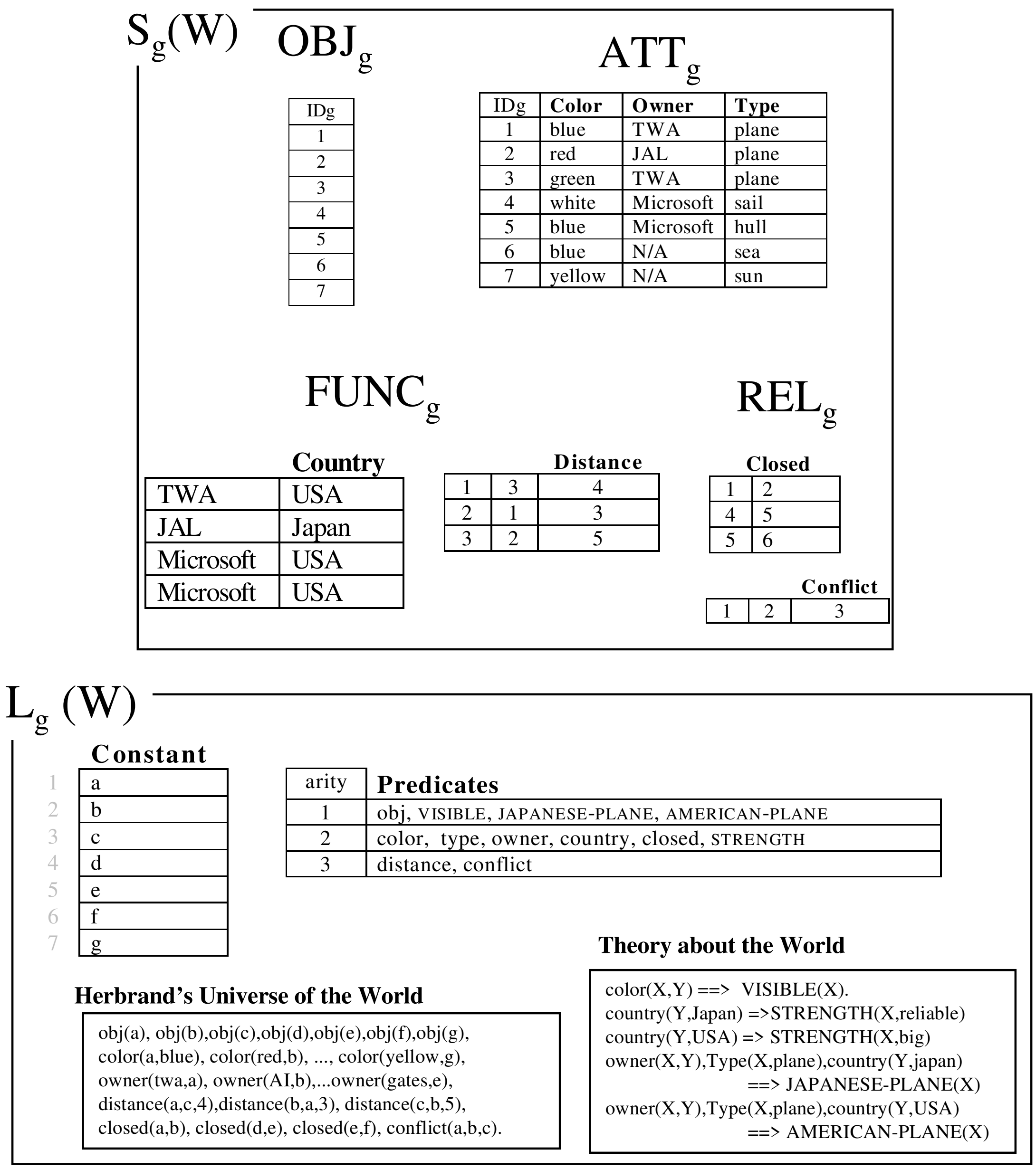

Figure 3 – Example of $S_g(W)$ and $L_g(W)$ corresponding to the perceived world of Figure 2.

In Figure 4, the view of the abstraction process presented in this paper is synthetically described. The symbols ω, σ, and λ denote abstraction operators acting at the three considered levels, respectively.

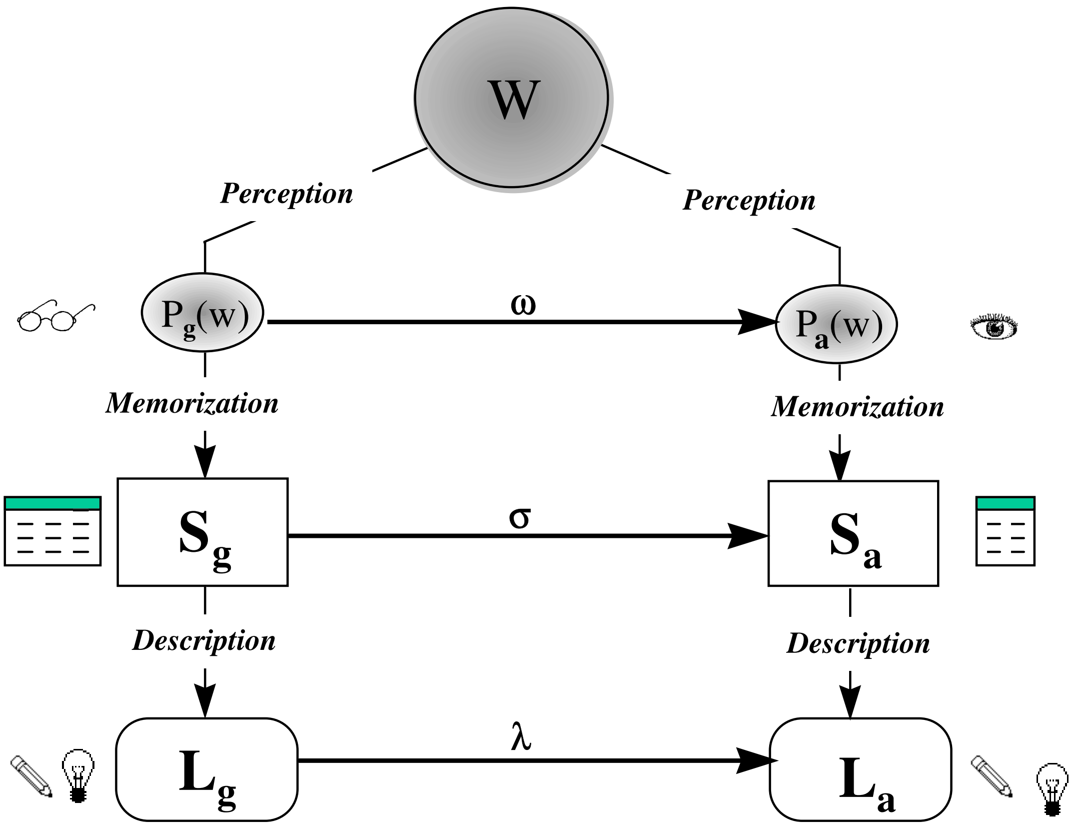

Figure 4 — Scheme of the abstraction process.

4. Abstraction Operators

In this section we first introduce some operators that we consider abstraction operators at the level of the world perception P(W): they constitute a set Ω. We present the corresponding operators at the other two levels, as well.

4.1. World Perception Level

The set Ω contains the abstraction operators at the P(W) level. The set proposed here is not necessarily exhaustive. In other contexts, Ω can be either reduced or augmented with domain-specific abstraction types. In this paper, we consider the following operators:

$$\Omega = \{\omega_{ind}, \omega_{ta}, \omega_{hide}, \omega_{eq\text{-}val}, \omega_{red\text{-}arg}, \omega_{prop}\}$$

The first operator applies to ***OBJ***. In particular, ω_{ind} specifies a *grouping of indistinguishable objects*. This operation corresponds to Imielinski's domain abstraction, which groups objects into *equivalence classes* (Imielinski, 1987). The arguments of the operator are the objects that shall be considered the same. The operator ω_{ta} also acts on ***OBJ***: it specifies how a set of ground objects can be *grouped to form a new compound object.* The primitive objects disappear from the abstract world. This operation, introduced first by Giordana and Saitta (1990) and called *term construction*, is a major novelty of this approach. It offers the most significant promises for limiting the complexity of learning in a first order logic setting, because it simplifies the matching process between hypotheses and examples (Giordana, Saitta & Roverso, 1991; Zucker & Ganascia, 1996; Giordana, Lo Bello & Saitta, 1993).

The arguments of the operator ω_{hide} are subsets of ***OBJ*, *ATT*, *FUNC*** or ***REL***. The specified arguments can be *ignored* in the abstract world, where they disappear. The operator $\omega_{eq\text{-}val}$'s arguments are also subsets of ***ATT*, *FUNC*** or ***REL***. In particular, for an attribute or a function, $\omega_{eq\text{-}val}$ specifies what subset of the attribute or function values can be merged, because they are considered indistinguishable. If the values belong to a function range, then the operator can be seen as an *approximation* of the function (Bennett, 1987).

Finally, the last two operators apply to subsets of either ***FUNC*** or ***REL,*** whose elements' arity is not less than 2. In particular, $\omega_{red\text{-}arg}$ specifies a function (or relation) and a subset of its arguments, which must be dropped from the function (or relation), obtaining thus a function (or relation) with reduced arity. The operator ω_{prop} is the extreme applicaton of $\omega_{red\text{-}arg}$, because ω_{prop} eliminates all the arguments of a function (or relation). Then, the function moves from a predicate logic to a propositional logic setting. This abstraction corresponds to the *propositional abstraction,* proposed by Plaisted (1981) at the language level.

Let us consider, in Figure 5, the three operators ω_{ind}, ω_{hide} and ω_{ta} applied to the example of Figure 2.

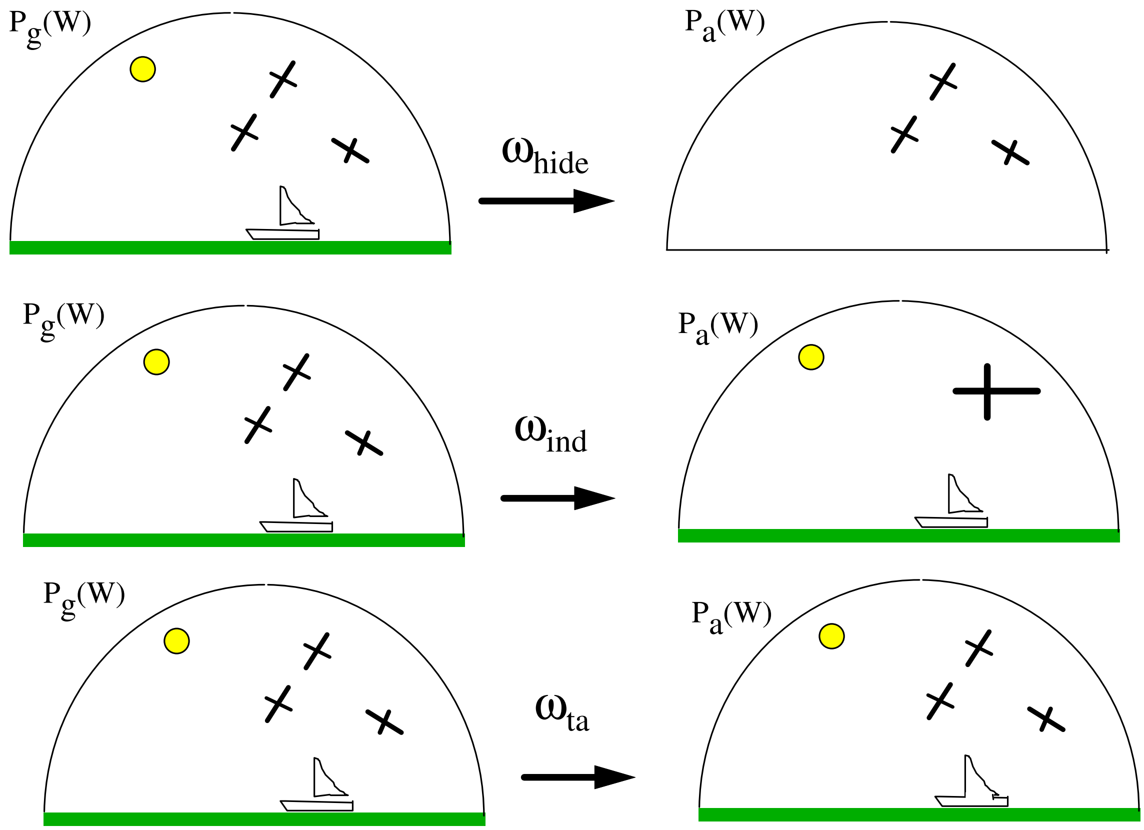

Figure 5 — Applying the operators ω_{ind}, ω_{hide} and ω_{ta} to Pg(W).

4.2. Structure Level

Each of the operators defined in Ω has an associated operator in a set Σ, at the structure level:

$$\Sigma = \{\sigma_{ind}, \sigma_{ta}, \sigma_{hide}, \sigma_{eq\text{-}val}, \sigma_{red\text{-}arg}, \sigma_{prop}\}$$

Each operator σ in Σ is defined on S by means of relational algebra's operators, such as projection, selection, join and product (Smith & Smith, 1977). For example, if $\omega_{red\text{-}arg}(r(x,y), y)$ belongs to Ω and r is a binary relation, then $\omega_{red\text{-}arg}(r^*(x,y), y)$ is a projection on x of the table r* associated to r. In an analogous way, σ_{ind}, applied to objects, is a selection operation. Other operators, such as σ_{ta}, are more complex to define but can all be related to database manipulations.

In Figure 6, the application of σ_{ind} (of Figure 5) to the example of Figure 2 is represented.

$S_g(W)$

OBJ_g

IDg
1
2
3
4
5
6
7

ATT_g

IDg	Color	Owner	Type
1	blue	TWA	plane
2	red	JAL	plane
3	green	TWA	plane
4	white	Microsoft	sail
5	blue	Microsoft	hull
6	blue	N/A	sea
7	yellow	N/A	sun

$FUNC_g$

Country

TWA	USA
JAL	Japan
Microsoft	USA
Microsoft	USA

Distance

1	3	4
2	1	3
3	2	5

REL_g

Closed

1	2
4	5
5	6

Conflict

1	2	3

σ_{ind}

$S_a(W)$

OBJ_a

IDg
i
ii
iii
iv
v

ATT_a

IDg	Color	Owner	Type
i	?	TWA or JAL	a plane
ii	white	Microsoft	a sail
iii	blue	Microsoft	a hull
iv	blue	N/A	a sea
v	yellow	N/A	a sun

$FUNC_a$

Country

TWA	USA
JAL	Japan
Microsoft	USA
Microsoft	USA

REL_a

Close

ii	iii
iv	v

Figure 6 – Application of the operators σ_{ind} (of Figure 5) to the example of Fig 2. In the abstract structure, the three planes are represented by the unique identifyer (i), whose properties, according to the semantics of ω_{ind}, are « undefined » color and "TWA or JAL" owner. Tables and Functions are updated w.r.t. to σ_{ind}.

4.3. Language Level

At this level, predicate and function symbols are associated to tables in S. We must differentiate, at the language level, abstractions of atomic predicates from abstractions of formulas. For each operator defined in Σ, let us define a corresponding operator in a set Λ:

$$\Lambda = \{\lambda_{ind}, \lambda_{ta}, \lambda_{hide}, \lambda_{eq\text{-}val}, \lambda_{red\text{-}arg}, \lambda_{prop}\}$$

The operator λ_{ta} can only be applied to formulas, whereas the other ones can be applied to atomic predicates, to functions and to formulas. The operator $\lambda_{eq\text{-}val}$ has been analyzed in several works on abstraction, because it corresponds to the case of *predicate mapping*, when it is applied to atomic predicates (Plaisted, 1981; Tenenberg, 1987; Giunchiglia & Walsh, 1992; Nayak & Levy, 1995), or to climbing a taxonomy (Utgoff, 1985), when it is applied to an attribute values. As mentioned before, both operators $\lambda_{red\text{-}arg}$ and λ_{prop} have been proposed by Plaisted (1981).

Let us consider, for instance, the operator λ_{ind}. If the planes from Japan and USA should be merged as in an analogous example from (Nayak & Levy, 1995), we can define in our framework the inference rule:

$$\lambda_{ind}\,(\text{JAPANESE-PLANE, AMERICAN-PLANE} \mid \text{FOREIGN-PLANE}) \equiv$$
$$\forall x\,[\text{JAPANESE-PLANE}(x) \vee \text{AMERICAN-PLANE}(x) \rightarrow \text{FOREIGN-PLANE}(x)]$$

More complex is the case of abstracting formulas. Let us consider, for instance, the operator ω_{ta} that combines a "hull" and a "sail" to form a yacht. In the world, ω_{ta} corresponds to actually building up a yacht. In S it corresponds to defining a new table, $f_{yacht}(x,y)$, with three columns: each row contains a triple $<hull_j, sail_j, yacht_j>$, which asserts that $hull_j$ is combined with $sail_j$ to form $yacht_j$. The column containing the yacht is the only one to be exported to the abstract world. At the language level, a corresponding operator λ_{ta} could be defined as follows:

$$\lambda_{ta}\,(\text{hull, sail} \mid \text{yacht}) \equiv$$
$$\forall x,y,z\,[\text{type}(x,\text{hull}) \wedge \text{type}(y,\text{sail}) \wedge \text{close}\,(y,x) \rightarrow \text{type}(z,\text{yacht})]$$
$$\text{where } z = f_{yacht}(x,y)$$

When a theory is available in the ground world, several difficulties emerge when trying to abstract it. A discussion of the issue, and a preliminary solution, has been proposed by Giordana, Saitta and Roverso (1991).

5. Characterizing Abstractions for Concept Learning

What makes our proposed framework different from previous works on abstraction is that an abstraction is not anymore defined starting from a syntactic mapping (Plaisted, 1981; Tenenberg, 1987; Giunchiglia & Walsh, 1992) but as a true semantic mapping. Indeed, we define an abstraction as a mapping between two perceived worlds $P_g(W)$ and $P_a(W)$. The modifications of the structure and of the language necessary to describe what happened at the level of the perceived world are, in some sense, side-effects.

In order to see why our approach may be useful for characterizing abstractions in concept learning, let us consider both a reasoning context $R_g = (P_g(W), S_g, L_g)$ and an abstraction between $P_g(W)$ and $P_a(W)$. If we know the operators in Ω, we can generate both the abstract structure S_a using the corresponding operators in Σ, and the language L_a using the operators in Λ.

Central questions for abstraction in Machine Learning (ML) include the following ones:

1) If concepts and examples are defined at a "ground" level $P_g(W)$, what are the features of an abstraction mapping that make the corresponding abstract concepts learnable (and, hopefully, more "easily" learnable) in the abstract space using L_a ?
2) How a concept definition learnt in L_a can be mapped back into L_g?

Here we restrict the discussion to the first question. Informally, in concept learning the definition (a "hypothesis") of an unknown concept "c" is searched for in a space defined by the language L_g. The most important structure on L_g is the so-called "generality" relation among hypotheses. Given two hypothesis h_1 and h_2, h_1 is said more general than h_2 if the set of models of h_2 is included in that of h_1. Given the centrality of this property, we would like to consider abstractions useful for ML that preserve it.

Definition 5.1 – (Generality-preserving Language Abstraction Mapping (GLAM))

A language level abstraction mapping Π, between L_g and L_a, compatible with an abstraction mapping M between $P_g(W)$ and $P_a(W)$, is a GLAM, if the more-general-than relation existing between any two hypotheses in L_g is preserved between the corresponding hypotheses in L_a, and vice-versa. The generality relations are evaluated on S_g and S_a, respectively. ❐

Examples of abstraction mappings that preserve generality can be found in (Giordana, Saitta & Roverso, 1991). The above definition does not set any restriction on the GLAMs, except that the abstracted structure S_a be used for evaluating the truth of the abstracted hypotheses. An open question is the identification of the operators which are GLAM and of the set of operators that preserved the GLAM property by composition.

6. Conclusions

In this paper we have briefly introduced a new perspective on abstraction, specifically oriented to the representation and learning of concepts. An important difference between learning and theorem proving is that learning exploits non-truth-preserving mechanisms, such as induction and abduction, which have different properties with respect to deduction.

As already mentioned, Nayak and Levy (1995) proposed a two-step semantic theory of abstraction. This semantic theory yields abstractions that are weaker than the base theory, i.e. they are strictly a subset of TD (theorem decreasing) abstractions (Giunchiglia & Walsh, 1992). In fact, Nayak and Levy introduce the notions of *model increasing abstractions* that are a strict subset of TD-abstractions. As Giunchiglia and Walsh state it, TD-abstractions are not really interesting for automated reasoning, because TD-abstractions are loosing part of the theorems. Indeed, some theorems in the base space may not be proved in the abstract one.

On the contrary, in Machine Learning, we do not mind loosing theorems, as long as the relation of generality is preserved. In facts, Nayak and Levy's Model Increasing (MI) abstractions are particular GLAMs. Nevertheless, MI-abstractions are still defined starting from the language. As such, they capture abstractions at the model level that can be defined by formulas in the language. But there are abstractions at the domain level that cannot be expressed by a formula, and therefore cannot be represented by a MI abstraction. In contrast, in our framework, the abstractions are firstly defined from the domain point of view : they do not need necessarily to be expressible in the language.

6. References

Amarel, S. (1983). “Representation in problem solving”. In *Methods of Heuristics* (pp. 131-171). Palo Alto, CA: Lawrence Erlbaum.

Benjamin D.P., Dorst L., Mandhyan I. and Rosar M. (1990). “An Algebraic Approach to Abstraction and Representation Change”. *Proc. AAAI Workshop on Automatic Generation of Approximations and Abstractions* (Boston, MA), pp. 47-52.

Bennett, S. (1987). "Approximation in Mathematical Domains", *Proc. IJCAI-87* (Milano, Italy), pp. 239-241.

Drastal, G., Czako G., & Raatz S. (1989). Induction in an abstraction space. *Proc. IJCAI-89* (Detroit, MI), pp. 708-712.

Ellman T. (1993). “Synthesis of Abstraction Hierarchies for Constraint Satisfaction by Clustering Approximately Equivalent Objects”. *Proc. Int. Conf. on Machine Learning* (Amherst, MA), pp. 104-111.

Enderton H.B. (1972). *Mathematical Introduction to Logic,* Academic Press.

Giordana A. and Saitta L. (1990). “Abstraction: a General Framework for Learning”. *Working Notes of Workshop on Automated Generation of Approximations and Abstractions* (Boston, MA), pp. 245-256.

Giordana A., Lo Bello G. and Saitta L. (1993). “Abstraction in Propositional Calculus”. *Proc. Workshop on Knowledge Compilation and Speed Up Learning* (Amherst, MA), pp. 56-64.

Giordana A., Saitta L., Roverso D. (1991). “Abstracting Concepts with Inverse Resolution”, Proc. *8th Int. Machine Learning Workshop* (Evanston, IL), pp. 142-146.

Giunchiglia, F., & Walsh, T. (1992). A Theory of Abstraction. *Artificial Intelligence*, 56, 323-390.

Hobbs, J. (1985). "Granularity", *Proc. IJCAI-85*, pp. 432-435.

Holte R.C., Mkadmi, T., Zimmer, R.M., MacDonald, A.J. (1996). "Speeding Up Problem-Solving by Abstraction: A Graph-Oriented Approach." *Artificial Intelligence, 85,* 321-361.

Holte, R.C. and Zimmer, R.M. "A Mathematical Framework for Studying Representation" Proceedings of the Sixth International Workshop on Machine Learning, Cornell University, Ithaca, New York, Segre A.M. (eds.). Morgan Kaufmann.

Imielinski, T. (1987). Domain Abstraction and Limited Reasoning. In Proc.*IJCAI*, (pp. 997-1003), Milano, Italy.

Iwasaski Y. (1990). “Reasoning with Multiple Abstraction Models”. *Proc. AAAI Workshop on Automatic Generation of Approximations nd Abstractions* (Boston, MA), pp. 122-133.

Knoblock, C. (1992). “Automatic Generation of Abstraction for Planning”.*Artificial Intelligence, 68 (2).*

Korf, R. E. (1980). “Towards a Model for Representation Change”. *Artificial Intelligence, 14,* 41-78.

Lowry, M. (1987). “The Abstraction/Implementation Model of Problem Reformulation”. *Proc. IJCAI-87* (Milano, Italy), pp. 1004-1010.

Matheus, C., Rendell, L. (1989). "Constructive Induction in Decision Trees", *Proc. IJCAI-89*, (Detroit, USA), pp. 645-650.

Michalski, R. (1983). “A Theory and Methodology of Inductive Learning”. In R. Michalski, T. Mitchell and J. Carbonell (Eds.), *Machine Learning, Vol. I*, Tioga Publ. Co., Palo Alto, CA, pp. 83-134.

Mitchell, T. (1982). "Generalization as Search", *Artificial Intelligence,18,* 203-226.

Muggleton, S., Buntine, W. (1988). "Machine Invention of First-Order Predicates by Inverting Resolution", *Proc. Fifth Int. Conf. on Machine Learning,* (Ann Arbor, MI), pp.339-352.

Nayak P.P. and Levy A.Y. (1995). “A Semantic Theory of Abstraction”. *Proc. IJCAI-95* (Montréal, Canada), pp. 196-202.

Pawlak Z. (1991). Rough Sets : Theoretical Aspects of Reasonning about Data. Kluwer Academic Publisher., Norwell, MA.

Pearl, J. (1978). "On the Connection between the Complexity and Credibility of Inferred Models", *Int. J. of General Systems*, 4, 255-264.

Plaisted, D. (1981). Theorem Proving with Abstraction. *Artificial Intelligence*, 16, 47-108.

Sacerdoti, E. (1973). Planning in a hierarchy of abstraction spaces. In *Proceedings of the 3rd International Jointed Conference in Artificial Intelligence (IJCAI)*, (pp. 412-422).

Smith J.M. and Smith D. (1977). “Database Abstractions: Aggregation and Generalization”. *ACM Trans. on Database Systems, 2,* 105-133.

Subramanian, D. (1990). “A Theory of Justified Reformulations”. In D. P. Benjamin (Eds.), *Change of Representation and Inductive Bias* (pp. 147-167). Boston: Kluwer Academic Publishers.

Subramanian D., Greiner R. and Pearl J. (eds) (1997). *Artificial Intelligence*, *97 (1-2).* Special Issue on Relevance.

Tenenberg J. (1987). “Preserving Consistency across Abstraction Mappings", *Proc. IJCAI-87* (Milan, Italy), pp. 1011-1014.

Utgoff, P. (1985). "Shift of Bias For Inductive Concept Learning", in Michalski R., Carbonell J. & Mitchell T.(Eds) *Machine Learning: An AI Approach,* Vol. *II.* Los Altos, CA: Morgan Kaufmann), pp.107-148.

Van Dalen, D. (1983). *Logic and Structure*, Springer-Verlag, Berlin, Germany.

Wneck J. and Michalski R. (1994). “Hypothesis-Driven Constructive Induction in AQ17-HCI: A Method and Experiments”. *Machine Learning, 14,* 139-168.

Yoshida K. and Motoda H. (1990). “Towards Automatic Generation of Hierarchical Knowledge Bases”. *Proc. AAAI Workshop on Automatic Generation of Approximations nd Abstractions* (Boston, MA), pp. 98-109.

Zucker, J.-D., and Ganascia, J.-G. 1994. Selective Reformulation of Examples in Concept Learning. Proc. *International Conference on Machine Learning*, pp. 352-360, New-Brunswick, Morgan Kaufmann.

Zucker, J.-D., and Ganascia, J.-G. 1996. Changes of Representation for Efficient Learning in Structural Domains. Proc. *International Conference in Machine Learning*, pp. 543-551, July 3-6, Bari, Italy. Morgan Kaufmann.