THE ROYAL SOCIETY

Published online 19 May 2003

A grounded theory of abstraction in artificial intelligence

Jean-Daniel Zucker

LIM&BIO, EPML-CNRS IAPuces, Université Paris XIII, 74 rue Marcel Cachin 93017 Bobigny Cedex, France, and LIP6—Département IA, Université Paris VI, 4 place Jussieu, 75252 Paris Cedex, France (jean-daniel.zucker@lip6.fr)

In artificial intelligence, abstraction is commonly used to account for the use of various levels of details in a given representation language or the ability to change from one level to another while preserving useful properties. Abstraction has been mainly studied in problem solving, theorem proving, knowledge representation (in particular for spatial and temporal reasoning) and machine learning. In such contexts, abstraction is defined as a mapping between formalisms that reduces the computational complexity of the task at stake. By analysing the notion of abstraction from an information quantity point of view, we pinpoint the differences and the complementary role of reformulation and abstraction in any representation change. We contribute to extending the existing semantic theories of abstraction to be grounded on perception, where the notion of information quantity is easier to characterize formally. In the author's view, abstraction is best represented using abstraction operators, as they provide semantics for classifying different abstractions and support the automation of representation changes. The usefulness of a grounded theory of abstraction in the cartography domain is illustrated. Finally, the importance of explicitly representing abstraction for designing more autonomous and adaptive systems is discussed.

Keywords: artificial intelligence; abstraction; reformulation; representation change; machine learning

1. INTRODUCTION: THE NOTION OF ABSTRACTION IN ARTIFICIAL INTELLIGENCE

Abstracting is a pervasive activity in human perception, conceptualization and reasoning. In AI there is a consensus (Ram & Jones 1995) that this ability to 'distil the essence from its superficial trappings' (Goldstone & Barsalou 1998) is a key issue, and that finding an adequate representation is often the hard part of the problem to be solved when building 'intelligent' systems. In fact, in the early 1990s, Brooks did challenge the tenet of the 'good old fashioned AI' stating that finding a good abstraction of a problem was the essence of intelligence and was, in many AI systems, done by the researcher himself (Brooks 1990, 1991). Thus, it is not surprising that a large proportion of AI successes rely as much on intelligent reasoning as on adequate problem representation.

Notwithstanding this fundamental role in AI, little study has been carried out directly on the subject, which has emerged as a side-effect of the investigation of knowledge representation and reasoning. In AI this ability to forget irrelevant details and to find simpler descriptions has been investigated, with few exceptions, either in problem solving (Sacerdoti 1974; Plaisted 1981; Giunchiglia & Walsh 1992; Ellman 1993; Knoblock 1994; Holte *et al.* 1996), or in problem reformulation (Amarel 1983; Lowry 1987; Subramanian 1990). An account of these researches is presented in Holte & Choueiry's (2003) contribution to this issue. In these studies, abstraction is usually not defined in a constructive way but rather *a posteriori* as a particular representation change that reduces the computational complexity of the task at stake.

In problem solving and theorem proving, abstraction is often associated with a transformation of the problem representation that allows a theorem to be proved (or a problem to be solved) more easily, i.e. with a *reduced computational effort*, according to the process described in figure 1. This pragmatic view of abstraction proved very useful to its intended goal and provides a means to characterize a transformation that is or is not an abstraction. The intuitive idea is that a representation change is an abstraction if the computational cost to solve a class of problems or demonstrate a class of theorems is significantly reduced. Representation changes may be recursively applied defining a hierarchy of representations of increasing level of abstraction (Sacerdoti 1974; Christensen 1990; Ellman 1993; Knoblock 1994; Shawe-Taylor *et al.* 1998). However, this view may not be sufficient for building *new* abstractions that rely on the definition of new concepts. Indeed, computational issues, even though important, are subsequent to the establishing of meaningful relations between the 'concepts' and their referents in the world. In concept representation, in fact, the role of abstraction seems more related to 'making sense' of the perception of the world, by transforming it into a set of meaningful 'concepts', prior to an efficient use of them. Abstraction is thus a fundamental mechanism for saving cognitive efforts, by offering a 'higher' level view of our physical and intellectual environment. Goldstone & Barsalou (1998) have recently advocated a stricter link between perception and conceptualization in cognitive

One contribution of 16 to a Theme Issue 'The abstraction paths: from experience to concept'.

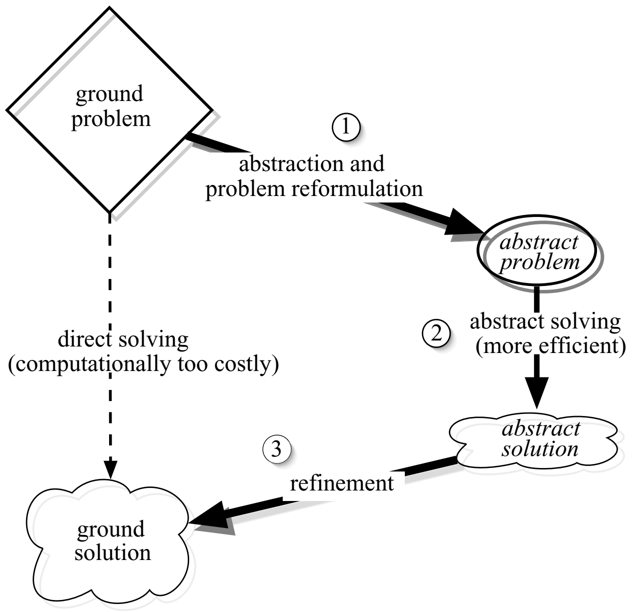

ground problem
1
abstraction and problem reformulation
abstract problem
direct solving (computationally too costly)
2
abstract solving (more efficient)
abstract solution
3
refinement
ground solution

Figure 1. Abstraction process for problem solving. Step 1 concerns a *representation change* justified by the need to reduce the computational complexity to solve a so-called *ground* problem. Step 2 concerns solving the abstracted problem. Step 3 concerns the refinement of the abstract solution back to the ground representation space.

science. Their approach offers a cognitive foundation to our grounded model of abstraction.

In this paper I am interested in the role played by abstraction in a phase preceding problem solving, namely the phase of conceptualizing a domain, when a set of appropriate, possibly interrelated, concepts is defined. In a domain, concepts are used for a variety of different tasks; they must then be internally organized in a flexible and dynamic way, allowing the properties that are relevant for the task at hand to be easily and quickly retrieved. As works on selective attention confirm (Goldstone & Barsalou 1998), humans show an amazing ability to change the representation level of details and choose relevant information. However, when currently irrelevant details are removed, they must not be deleted, but simply hidden, because they may become useful for another task. Hence, a model of the abstraction process should also accommodate linking different representations, in both directions.

Notwithstanding its diffuse presence in human perception and reasoning, abstraction is an elusive notion, difficult to capture in formal terms. Given the multiplicity of goals and tasks in which abstraction is involved, the identification of a unique set of properties which it has to satisfy may not be possible. In fact, there is currently no general framework that provides the means for understanding the different facets of abstraction. A comprehensive review and a unified approach to problem-solving-oriented abstraction have been proposed by Giunchiglia & Walsh (1992) and Choueiry *et al.* (2003).

If abstraction is difficult to formalize, devising different abstractions from a given initial representation is likely to be even more complex. However, the ability of an agent to change or adapt its representation of the world is undoubtedly essential for surviving and solving complex problems. The practical motivation in this article is to

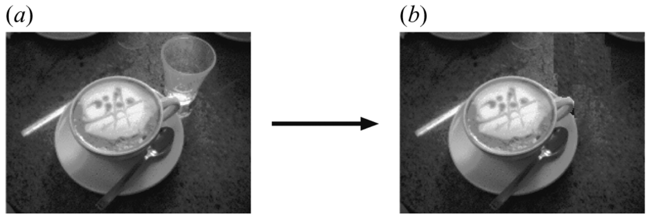


Figure 2. (*a*) A set of objects on a bar table. (*b*) This image is more abstract, because the glass is hidden. This transformation is a domain hiding abstraction.

enquire into the abstraction process in general, and to propose an operational framework to devise systems that automatically and iteratively build their own abstractions. To achieve this goal it is necessary not only to define abstraction(s), but also to propose the means for exploring different abstractions and operationalizing them.

To achieve a better understanding of the abstraction process, several problems remain to be solved. First, a definition of abstraction should be provided, at least in a circumscribed sense. Abstraction is intuitively related to the notion of ‘simplicity’, but this link does not make its definition any easier, as simplicity seems to be an equally elusive notion. Moreover, different properties may be required for abstraction processes involved in different tasks. The problem is thus twofold: (i) to identify what properties are likely to be useful for a task; and (ii) once they have been found, how to formally represent them. A more concrete problem is to define mechanisms to perform abstraction in practice, i.e. to identify operators that carry out transformations between different abstraction levels. Finally, one of the most challenging problems is to explain how useful abstractions are acquired and/or formed. The rest of the paper is organized as follows: § 2 illustrates examples of abstraction, to give the reader an intuitive semantics for the transformations referred to as *abstract*. Section 3 briefly reviews existing theories of abstraction. Section 4 presents a grounded framework for abstraction in concept representation, whereas § 5 introduces the notion of abstract operators. Section 6 gives an illustration of the interest of a grounded theory of abstraction in the cartography domain, and finally, § 7 suggests directions for future work.

2. INFORMAL PRESENTATION OF PERCEPTION-BASED ABSTRACTIONS

In this section I provide some examples of *transformations*[1] that I intend the notion of abstraction to capture. To provide the reader with an intuitive idea of the semantics of these transformations, I have chosen transformations of *images*. The different transformations of images presented differ by their properties, be it their content (*domain* related) or their description (*co-domain* related). Whenever possible, these transformations are related to their cognitive equivalents as described in the seminal work by Goldstone & Barsalou (1998).

(a) ***Domain hiding***

In figure 2, a number of objects are situated on a table in a bar. A filter placed on the top hides the glass. It is still there, but it can no longer be seen unless the filter is

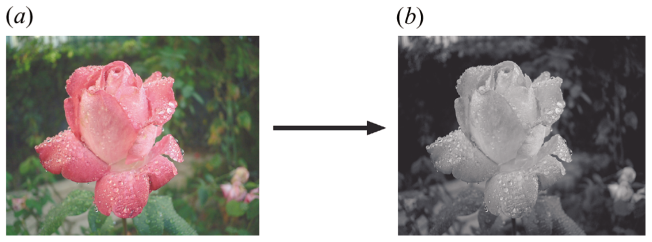


Figure 3. By hiding the colour of an image of a rose (*a*), a more abstract image is obtained (*b*) This transformation, that hides elements of the description, is called co-domain hiding.

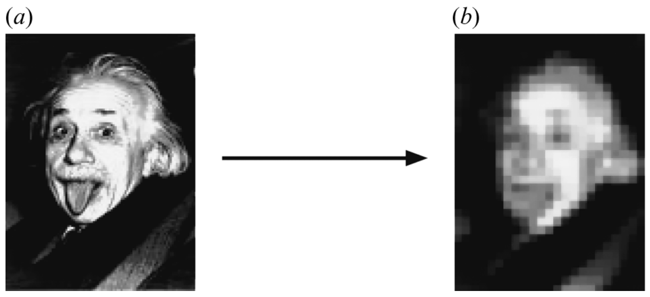


Figure 4. By lowering the resolution of an image (*a*), a more abstract one (*b*) is obtained. This transformation corresponds to a *lossy sub-sampling* in image compression.

removed. From a cognitive point of view, this transformation corresponds to a focalization (Goldstone & Barsalou 1998). Abstraction reduces (hides) part of the domain, to focus, for example, on the word 'Ciao' that has been drawn in the foam of the coffee by the deft Italian barman. Simons & Levin (1997) have shown that when a part of an image is slowly hidden (such as the glass in figure 2*b*), one may be *blind* to this change. This *change blindness* phenomenon in visual perception characterizes the fact that very large changes occurring in full view in a visual scene may go unnoticed (O'Regan 2001). Domain hiding is arguably the most basic and common form of abstraction explored in AI.

(**b**) *Co-domain hiding*

In figure 3, two pictures of a rose are shown: figure 3*a* is a coloured image, and figure 3*b* is a grey-level image. Changing from the first to the second picture, the information about the colour has been hidden. This transformation is related to what Goldstone & Barsalou (1998) call *selectivity*, i.e. the ability not to pay attention to useless perceptual features in a given task. We may notice the asymmetry of the relation between figure 3*a* and *b*: whereas figure 3*b* can be obtained from figure 3*a*, the reverse is not possible (unless the removed information can be memorized). In fact, re-colouring the picture by means of a graphic program is indeed possible, but there is no guarantee that the resulting picture actually corresponds to the original coloured picture of the world that is captured by the same optical instrument. Co-domain hiding corresponds to another widely considered abstraction, where part of the object *description* is hidden.

(**c**) *Domain reduction*

In figure 4 a different case is illustrated. Figure 4*a* is transformed by grouping four adjacent pixels into a single

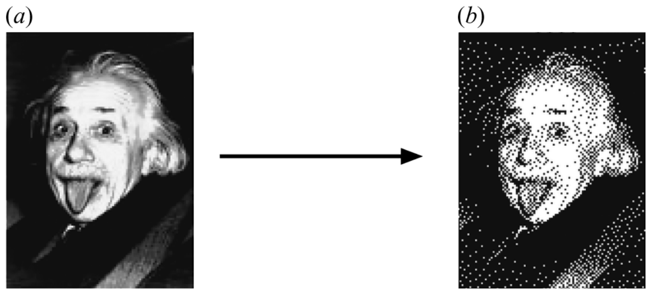


Figure 5. By thresholding with only two levels a 256 grey-level picture (*a*), a much less detailed picture is obtained (*b*). This transformation is called *coarse quantization* in image compression.

one, associating to the compound the mean value of colours and brightness of the four component pixels. The result is a change in resolution, and we consider the image in figure 4*b* more abstract than that in figure 4*a*. Abstraction, in this case, reduces the domain of a function, making sets of points (sixteen adjacent pixels) indistinguishable. This transformation corresponds to the *blurring* cognitive operation defined by Goldstone & Barsalou (1998). In image compression, this transformation is an implementation of the *sub-sampling* technique[2] (Toelg & Poggio 1994). Sub-sampling reduces the number of bits required to describe an image and the *quality* of the sub-sampled image is said to be lower than the quality of the original. It is called a *lossy* compression algorithm.[3] Domain reduction thus corresponds to any abstraction where a group of objects is replaced by a new prototypic object.

(**d**) *Co-domain reduction*

Starting from a grey-level picture of a person (see figure 5*a*), a thresholding can be performed using only two grey levels (black/white) (see figure 5*b*), thus hiding many of the details of the original image. In this case, abstraction reduces the co-domain of a function, making sets of values indistinguishable. In image processing this transformation is called *quantization* (Toelg & Poggio 1994). Coarse quantization is similar to sub-sampling in that information is discarded, but the compression is accomplished by reducing the numbers of bits used to describe each pixel, rather than reducing the number of pixels. Each pixel is reassigned an alternative value and the number of alternative values is less than that in the original image. This type of abstraction is widely used at the level of symbolic description to *discretize* numerical attributes. The intuitive idea is to associate a symbol to a set of values that are considered indiscernible (Dougherty *et al.* 1995; Kohavi & Sahami 1996). Co-domain reduction corresponds to all abstractions where the domain of values taken by object descriptors, function or relation is reduced.

(**e**) *Domain aggregation*

Finally, in figure 6 a very important type of abstraction is described. If we were asked, in front of the desk represented in figure 6*a*, what we see on the table, most of us would answer a 'computer', and not 'a monitor, a keyboard and a mouse'. We have spontaneously grouped together a set of objects that are functionally related into a composite object, i.e. a 'computer'. In figure 6, we may

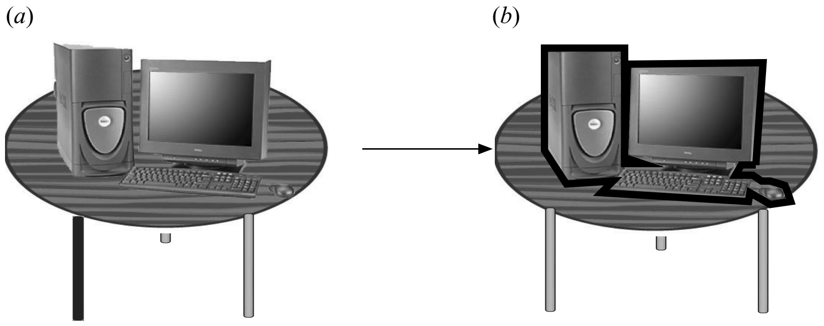
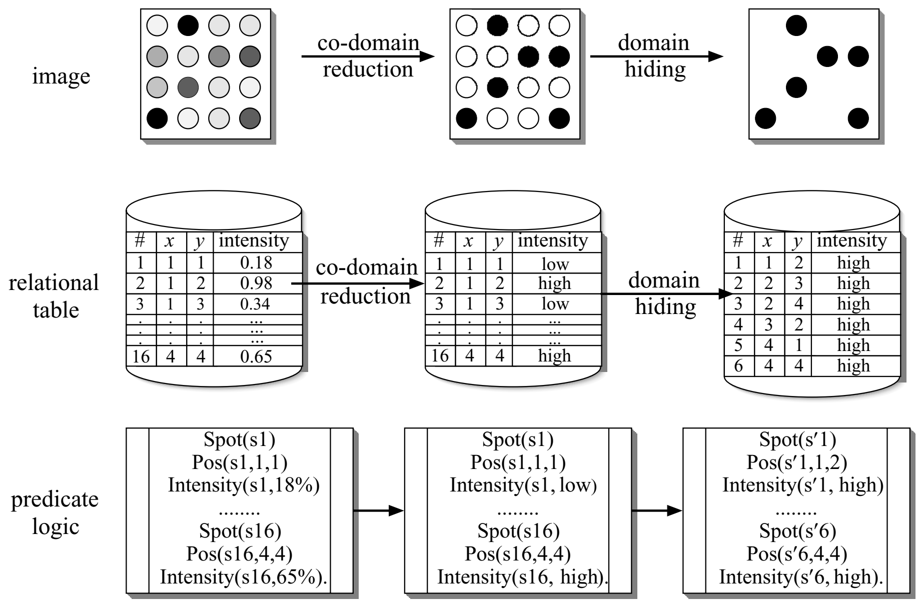


Figure 6. A monitor, a PC tower, a keyboard and a mouse are each represented as individual objects on top of a table (*a*). By aggregating them, a more abstract image—where a computer is viewed as a whole—is obtained (*b*).



#	*x*	*y*	intensity
1	1	1	0.18
2	1	2	0.98
3	1	3	0.34
:	:	:	...
:	:	:	...
16	4	4	0.65

#	*x*	*y*	intensity
1	1	1	low
2	1	2	high
3	1	3	low
:	:	:	...
:	:	:	...
16	4	4	high

#	*x*	*y*	intensity
1	1	2	high
2	2	3	high
3	2	4	high
4	3	2	high
5	4	1	high
6	4	4	high

Spot(s1)
Pos(s1,1,1)
Intensity(s1,18%)
....
Spot(s16)
Pos(s16,4,4)
Intensity(s16,65%).

Spot(s1)
Pos(s1,1,1)
Intensity(s1, low)
....
Spot(s16)
Pos(s16,4,4)
Intensity(s16, high).

Spot(s′1)
Pos(s′1,1,2)
Intensity(s′1, high)
....
Spot(s′6)
Pos(s′6,4,4)
Intensity(s′6, high).

Figure 7. Illustrative examples of a similar abstraction viewed as a transformation of an image, a relational table and facts in a predicate language. This abstraction is at each level a combination of a co-domain reduction (discretize the spot intensity) and domain hiding (hides spot of low intensity).

notice that, even though the whole computer is perceived at first sight, the components do not disappear; in fact, as soon as we speak of 'computer configuration', they can be retrieved and used again. This transformation corresponds to the *productivity* human cognitive operation defined by Goldstone & Barsalou (1998). Domain aggregation is a transformation that corresponds to abstraction where objects are composed to build new ones.

(f) *Abstract transformations in other knowledge representation formalisms*

Domain hiding, co-domain hiding, domain reduction, co-domain reduction and domain aggregation are typical transformations that illustrate the notion of *perception-based* abstraction. In fact, similar transformations are used within non-graphical knowledge representation formalisms, be it logical or not. Relational databases, propositional, or first-order logic are other types of representation formalism where abstractions are typically considered. Figure 7 illustrates an abstraction within three different formalisms. At the perception level, this abstraction corresponds to a combination of both co-domain reduction (as the resolution of the image has been lowered) and domain hiding (as spots of pixels whose intensity is not above a threshold are hidden). At the relational level, this transformation respectively corresponds to the *inclusion* and *select* relational database operations (Goldstein & Storey 1999). At the predicate level, this transformation corresponds to a *value discretization* followed by an *object selection*.

3. ABSTRACTION: DEFINITIONS AND APPROACHES IN ARTIFICIAL INTELLIGENCE

The previous section aimed at illustrating a set of examples of transformations that correspond to abstractions. They illustrated that the notion of abstraction does not have to be solely related to the reduced computational complexity. The notion of *information decrease* does in fact better characterize these different examples. This section defines the notion of abstraction, and contrasts it with that of *reformulation* and *representation change*.

(a) *Representation changes, abstraction and reformulation*

There are many tasks or problems for which there exists one *good representation*,[4] which allows the definition of efficient and sound algorithms. This said, there are also numerous problems for which a good representation is either as yet unknown or exists independently of precise goals. Since the beginning of AI, mechanisms allowing changes of representation (Amarel 1968; Korf 1980; Benjamin *et al.* 1990) and problem reformulation (Riddle 1990; Wneck & Michalski 1994; Lavrac & Flach 2001; Choueiry *et al.* 2003) have been envisioned.

A seminal contribution to the understanding of representation changes in AI comes from Korf (1980). Within problem solving, he distinguishes between transformations that are *isomorphisms* and those that are *homomorphisms*. Isomorphisms preserve the quantity of information of the initial problem but modify the structure that represents this information. Changing the representation of a circle of radius R from $x^2 + y^2 = R^2$ to $\rho = R$ and $\theta \in [0,2\pi]$ is a typical case. By contrast, homomorphisms change the formalism but modify the information quantity.[5]

(b) *A definition of abstraction*

Before giving a general definition of abstraction, it should be noted that the term 'abstraction' has been widely used in AI, and in a wide range of contexts: abstract problem solving and planning (Sacerdoti 1974; Giunchiglia & Walsh 1992), abstract states (Provan 1995; Holte *et al.* 1996), temporal abstraction (Euzenat 1995; Shahar 1997; Bettini *et al.* 1998), space abstraction (Forbus *et al.* 1991), abstracted causal model (Iwasaki 1990), languages, classes and abstract types (Briot & Cointe 1987), abstract interpretation (Cousot 2000), knowledge and abstract reasoning (Hobbs 1985; Imielinski 1987; Smoliar 1991), abstract proof (Giunchiglia & Walsh 1992), abstract lambda-calculus (Orlarey *et al.* 1994), abstract structure (Zucker & Ganascia 1996), abstract conceptual graph (Bournaud *et al.* 2000), abstract data (Smith & Smith 1977; Goldstein & Storey 1999) and abstract genotype (Bentley & Kumar 1999), etc.

The common idea that underlies these various uses of the word abstraction echoes the human capacity to focus on simpler descriptions in perception and conceptualization as well as in reasoning (Goldstone & Barsalou 1998, p. 62). Giunchiglia & Walsh (1992), whose work is among the most cited in the field of abstraction, define abstraction as a process of mapping an initial representation of a problem. The latter is built by hiding details from the ground one, to simplify the exploration space preserving some 'properties' desirable to map back the abstract solution. Definition 1 is an attempt to summarize the common view of abstraction in AI.

Definition 3.1. *An* abstraction *is a change of representation, in a* same *formalism, that hides* details *and preserves* desirable properties.

This definition is more restrictive than that proposed by Giunchiglia & Walsh (1992), as it requires that the formalism remain unchanged in the transformation. The usefulness of this restriction is better in distinguishing the essence of an abstraction from the general notion of representation change. Nevertheless, frequently an abstraction and a change of formalism are combined into a *representation change* as described in figure 8*a*. Indeed, abstraction makes it possible to change the quantity of information represented, allowing a reformulation in a less expressive or *simpler* formalism to become possible. In this context, the term reformulation is restricted to precisely characterize a change of the sole representation formalism.[6]

Definition 3.2. *A* reformulation *is a change of representation, from one to another formalism,* preserving *the quantity of information involved.*

A review of the relation between the notion of abstraction and 'approximation' is beyond the scope of this article. However, there is a large body of research on approximate reasoning[7] (Imielinski 1987; Lesser *et al.* 1988; Subramanian 1990), 'qualitative reasoning' (Kuipers 1986), temporal reasoning (Bettini *et al.* 1998) and spatial reasoning (Forbus *et al.* 1991). Allen's famous set of relations between interval diagrams is a good illustration of links between approximation and abstractions (Allen 1984). It describes all possible relations among time intervals according to a logical framework: before, meets, overlaps, during, starts, finishes and equals.

(c) *The difficulty to define the notions of 'simplicity', 'details' and 'desirable properties'*

Definition 3.1 remains elusive, and, in effect, when one tries to formalize the definition of an abstraction one encounters the formalization of the notions of *details*, *desirable property* and *simplicity*. The following is a quick survey, more particularly in the context of a machine learning task, which clearly shows the variety of characterization given for these three notions upon which the definition of abstraction relies. This diversity underlines the difficulty in formalizing definition 3.1.

The notion of 'detail' is often associated with that of *relevance* to a class of tasks (Subramanian *et al.* 1997; Lavrac *et al.* 1998). Details that are hidden are indeed defined as being 'less relevant' to these tasks. In machine learning, for example, Blum & Langley (1997) have given several definitions of attribute *relevance*. Their definitions are related to measures that quantify the information brought by an attribute with respect to other attributes, the class or the sample distribution. Based on these definitions a *filter approach* to feature selection (a kind of *co-domain hiding* at the language level) may be applied to an initial representation of examples to ease the learning task by reducing the number of features. Domain hiding is also used in machine learning to reduce the number of examples to speed up instance-based algorithms such as *k*-Nearest Neighbors (kNN) (Wilson & Martinez 1998). In problem solving where a basic mechanism to speed up the search is to hide parts of the operators' pre-conditions, the 'criticity' to satisfy an operator pre-condition may be used to rank the pre-condition and decide the ones that will be dropped (Bundy *et al.* 1996; Yang 1997). It is also possible to define a domain-independent notion of detail. In that case, it is related to the notion of *scale* or *granularity* of descriptions (Hobbs 1985; Imielinski 1990; Smoliar 1991).

The notion of 'desirable properties' depends on the field where abstractions are used. In problem solving, for example, a classic desired property is the 'monotonicity' (Sacerdoti 1974; Knoblock 1994). This property states that operators' pre-condition does not interfere once

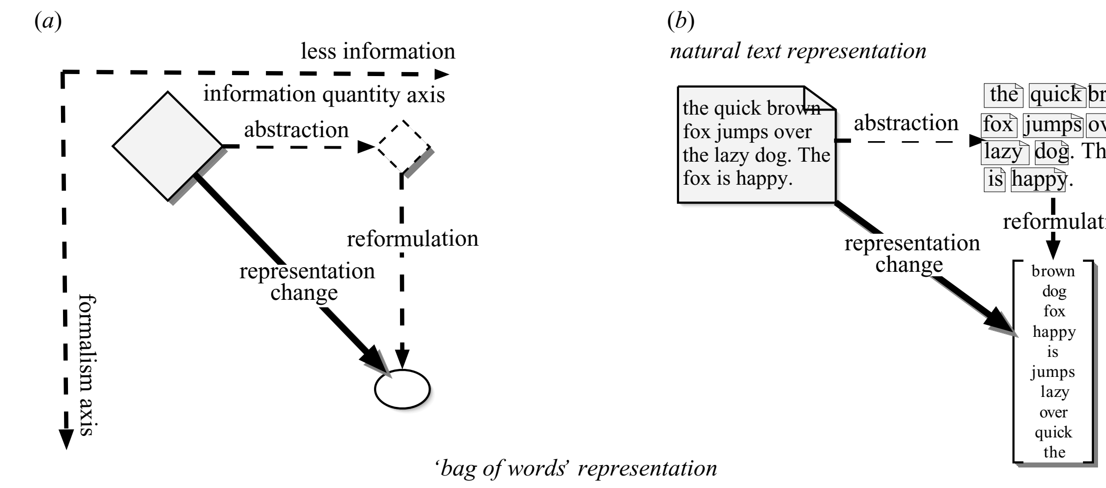

(*a*)
less information
information quantity axis
abstraction
reformulation
representation change
formalism axis
(*b*)
natural text representation
the quick brown fox jumps over the lazy dog. The fox is happy.
abstraction
the quick brown fox jumps over the lazy dog. The fox is happy.
reformulation
representation change
brown dog fox happy is jumps lazy over quick the
'bag of words' representation

Figure 8. (*a*) Change of representation viewed as a combination of abstraction and reformulation. (*b*) A change of representation from a natural text representation to a 'bag of words' representation. The abstraction assumption is that the presence of a word is relevant and not its position. This representation change is often used in text indexing.

abstracted. Another useful property is the 'downward refinement' (Bacchus & Yang 1994) that states that no backtrack in a hierarchy of abstract space is necessary to build the refined plan. In theorem proving, a desirable property states that for each theorem in the ground representation there exists a corresponding abstract one in the abstract representation (this property is called TI abstraction). In machine learning, a desirable property states that generalization order between hypotheses is preserved (Giordana & Saitta 1990). In constraint satisfaction programming, a desirable property is that the set of variables that are abstracted into one have 'interchangeable supports' (Freuder 1991). More generally, there is a domain-independent desirable property: it states that a given *order relation* is preserved by an abstraction (Hobbs 1985).

The notion of 'simplicity' plays a key role in the characterization of abstracted representation. There is abundant literature discussing the *simplicity* of a representation. In machine learning, this notion is fundamental because it guides the choice between equally accurate concept descriptions. Indeed, the 'simplest' concept is invariably favoured following the *parsimony principle* or Occam's razor[8] (Blumer *et al.* 1987; Domingos 1998). There are different measures of the simplicity of a hypothesis space that have been proposed. The *capacity*,[9] for example, is a measure of the power of a collection of functions to discriminate between classes. Some other principles or measures of simplicity are used in machine learning: the Minimum Description Length (Rissanen 1985) and the Minimum Message Length (Wallace & Boulton 1968). Finally, mention should be made of the rich literature on information-theoretic approaches to the notion of simplicity (Kolmogorov 1965; Li & Vitányi 1993), computational learning (Blumer *et al.* 1987; Kearns 1990) and statistical learning (Vapnik 1995).

4. THEORIES OF ABSTRACTION

Section 3 has provided a workable definition of a transformation that is or is not an abstraction. An abstraction theory goes beyond a definition, and is a step towards understanding the reasons that justify the choice of a particular abstraction and the search for better abstractions. A comprehensive theory of the principles underlying abstractions is useful for a number of reasons. From a practical point of view it may provide: (i) the means for clearly understanding the different types of abstraction used in past work; (ii) semantic and computational justifications for using abstractions; and (iii) justifications to automatically construct useful abstractions.

Moreover, *an understanding of different abstractions within a common framework can allow the transfer of techniques between disparate domains* (Nayak & Levy 1995). There have been several attempts to propose such a type of theory. Many of them have been carried out in theorem proving, where the role of abstraction is to find a sketch of a proof, whose details can be filled in later. This section gives a brief account of existing theories of abstraction and their associated definition of abstraction.

(a) *Abstraction as syntactical mapping*

(i) *Abstraction as predicate mapping*

Predicate mapping (Plaisted 1981; Tenenberg 1987) is a class of abstractions based on the observation that the distinctions between a set of predicates $P_1, \ldots, P_n$ in a theory may become irrelevant for certain inferences. An abstract theory can be constructed by replacing all occurrences of the predicates P_i in the base theory by a single abstract predicate P_a. Let us consider, for example, the following base theory that states that Hondas are Japanese cars, BMW are European cars, and both European and Japanese cars (among others) are types of car:

$$\begin{array}{lll} \text{BASE THEORY} & \text{HONDA}(x) \Rightarrow \text{JAPANESECAR}(x), & (4.1)\\ & \text{BMW}(x) \Rightarrow \text{EUROPEANCAR}(x), & (4.2)\\ & \text{JAPANESECAR}(x) \Rightarrow \text{CAR}(x), & (4.3)\\ & \text{EUROPEANCAR}(x) \Rightarrow \text{CAR}(x). & (4.4) \end{array}$$

The distinction between the predicates $P_1 = \text{JAPANESECAR}$ and $P_2 = \text{EUROPEANCAR}$ may become irrelevant (e.g. when trying to answer a query CAR(A)), and therefore these predicates can both be mapped to $P_a = \text{JAPEUROCAR}$.

PREDICATE MAPPING JAPANESECAR → JAPEUROCAR,
EUROPEANCAR → JAPEUROCAR.

This abstraction applied to axioms (4.1)–(4.4) yields the following simpler abstract theory:

SYNTACTICALLY ABSTRACTED THEORY HONDA(x) ⇒ JAPEUROCAR(x), (4.5)

BMW(x) ⇒ JAPEUROCAR(x), (4.6)

JAPEUROCAR(x) ⇒ CAR(x). (4.7)

However, suppose the base theory also includes the following axioms stating that European cars are fast and Japanese cars are reliable:

BASE THEORY EUROPEANCAR(x) ⇒ FAST(x), (4.8)

JAPANESECAR(x) ⇒ RELIABLE(x). (4.9)

Applying the same syntactical mapping to axioms (4.8) and (4.9) would result in the following:

ABSTRACT THEORY JAPEUROCAR(x) ⇒ FAST(x), (4.10)

JAPEUROCAR(x) ⇒ RELIABLE(x). (4.11)

However, combining these two axioms with axioms (4.5) and (4.6), leads to undesirable false proofs (Plaisted 1981). For example, one can infer that Honda cars are fast, and BMWs are reliable, inferences not sanctioned by the base theory (Nayak & Levy 1995). To avoid this problem, Tenenberg (1987) has proposed a *restricted* type of predicate mapping. After replacement of the predicates in the clauses, only the clauses that do not bring inconsistency are kept.

(ii) *Abstraction as mapping between formal systems*

Giunchiglia & Walsh (1992) have extended the syntactic view of abstraction of Plaisted (1981) to a mapping between *formal systems*. A formal system Σ is defined by a triple (Λ,Δ,Ω), where Λ is the language, Δ is the deductive mechanism and Ω is the set of axioms. They define an abstraction as follows:

Definition 4.1 (formal abstraction; Giunchiglia & Walsh 1992). *An abstraction, denoted f: $\Sigma_1 \rightarrow \Sigma_2$, is the union of three functions $f_\Lambda : \Lambda_1 \rightarrow \Lambda_2$, $f_\Delta : \Delta_1 \rightarrow \Delta_2$, and $f_\Omega : \Omega_1 \rightarrow \Omega_2$, which respectively transform the formulas of the initial language, inference rules and axioms respectively into the abstract formulas, the abstract rules and the abstract axioms.*

According to Giunchiglia & Walsh (1992), the majority of abstractions in problem solving and theorem proving may be represented within this framework. They also notice that most abstractions modify neither the axioms nor the inference rules. Abstractions are therefore, in most cases, mapping between languages. They have introduced a useful distinction between abstraction: TD and TI. This distinction is used to classify the various abstractions found in problem-solving and theorem-proving literature. TI abstractions are those whose image by f_Λ of each theorem from the ground space is a theorem in the abstract space. The TD abstractions are, on the contrary, abstractions such that images of certain theorems in the ground space are no longer a theorem in the abstract space.

Giunchiglia and Walsh argue that useful abstractions for the resolution of problems are the TI abstractions because they preserve all theorems. In problem solving, given S the description of a situation, i.e. a set of ground facts, a widely used abstraction consists in hiding part of precondition operators as it happens in STRIPS.[10] The intuition is to build an abstract plan that ignores the non-critical preconditions of operators and then refines it. Such abstraction may also be categorized as a TI abstraction as all the plans valid in the ground space are also valid in the abstract space. Their theory of abstraction[11] is not directly related to the notion of simplicity, and although it is a powerful framework to describe contributions, it is not meant to support the construction of new abstractions.

(b) *Abstraction as semantic mapping*

(i) *Semantic mapping of interpretation models*

The fundamental shortcomings of the syntactic theory of abstraction are that: (i) while it captures the final result of an abstraction, it does not capture the underlying justification that leads to the abstraction; (ii) TI abstractions potentially lead to false proofs; and (iii) the abstracted theory may not be the strongest one given the abstraction. For example, it appears that adding axiom (4.12) yields the strongest theory that removes predicates JAPANESECAR and EUROPEANCAR and still does not admit false proofs:

SEMANTICALLY ABSTRACT THEORY JAPEUROCAR(x)
⇒ (FAST(x) ∨ RELIABLE(x)). (4.12)

Nayak & Levy (1995)—extending Tenenberg's work—have proposed a semantic theory of abstraction. Their theory defines abstraction as a model-level mapping rather than predicate mapping. Instead of viewing abstraction as a syntactic mapping (like Giunchiglia & Walsh 1992), they view abstraction as a two-step process: a kind of semantic mapping. The first step consists in abstracting the 'domain', and the second one in constructing a set of abstract formulae to capture the abstracted domain. This semantic theory yields abstractions that are weaker than the base theory, i.e. they are strictly a subset of TD. Recall that Giunchiglia and Walsh do not advocate TD abstractions (for problem solving), because, although sound, they 'lose completeness' in comparison with TC abstractions, which preserve completeness, and TI, which lose soundness. Nayak and Levy introduce two important notions: *MI abstractions* (MI that are a strict subset of TD, see definition 4) and *simplifying assumptions* (which allow one to evaluate the utility of an abstraction depending on the reliability of this assumption). They defined abstraction as follows: an abstraction mapping Π: interpretations (L_{base}) ⇒ interpretations (L_{abs}) is a model-level specification of how the interpretations of L_{base} are to be abstracted to interpretations of L_{abs}.

Definition 4.2 (model-increasing abstraction; Nayak & Levy 1995). *The theory T_{abs} is a model-increasing abstraction for T_{base} with respect to an abstraction mapping Π if for every model M_{base} of T_{base} (the set of sentences in L_{base}), $\Pi(M_{base})$ is a model of T_{abs} (the set of sentences in L_{base}).*

The work of Nayak and Levy, although only considering a limited subset of the possible abstractions (i.e. 'domain abstraction'), is, in my view, a fundamental contribution to a theory of abstraction.

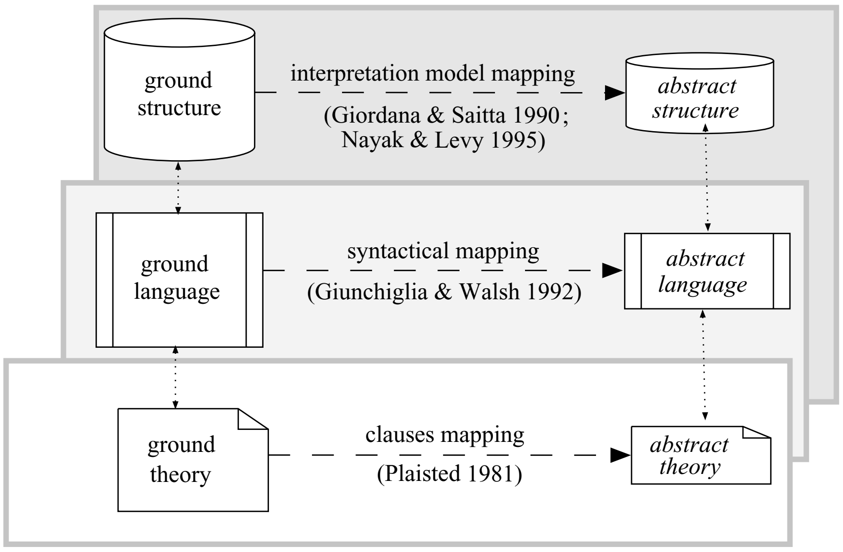


Figure 9. Theoretical approaches to abstraction according to the level of representation they operate at: mapping of theories (Plaisted 1981), mapping of clauses (Giunchiglia & Walsh 1992) and mapping of interpretation models (Giordana & Saitta 1990; Nayak & Levy 1995).

(ii) *Semantic mapping of formulae*

Historically, Giordana & Saitta (1990) were precursors in the semantic approach to predicate mapping. Indeed, they did not consider abstraction as a purely syntactic transformation but as a semantic one that preserves the generality order between formulae.[12] The abstraction of a clause C, expressed in a language $\mathscr{L}$, is another clause C′, where relations in C have been renamed or grouped into new ones defined over an abstract language $\mathscr{L}'$. In their theory, an abstraction is defined as follows:

Definition 4.3 (abstraction; Giordana & Saitta 1990). *An abstract theory, mapping a language of clauses $\mathscr{L}$ to an abstract language $\mathscr{L}'$, is a consistent set of definitions of the type $\forall x \exists\, y = (y_{i=1,n})$ such that $[\Psi(x) \equiv f_1(x,y) \vee \ldots \vee f_n(x,y)]$, where Ψ is an abstract predicate of $\mathscr{L}'$ and $f_{i=1,n}$ are formulas based on predicates in $\mathscr{L}$ applied to the variables x and $y_{i=1,n}$.*

An abstraction is, in their view, the definition of new concepts at the extensional level: a specific correspondence between the variables of the abstract formula and those of the initial one. These abstractions are a subset of the TD abstractions and are particularly useful to define the notion of *term abstraction* as described in § 2e and figure 6. In their theory, an abstraction of five pixels centred on a pixel x into a 'cross'-shaped larger pixel would be represented as follows:

$$\forall x \exists y,z,t,u\, Cross(x) \equiv (Pixel(x) \wedge Pixel(y) \wedge Pixel(z) \wedge Pixel(t) \wedge Pixel(u) \wedge Above(y,x) \wedge Below(z,x) \wedge Right(t,x) \wedge Left(u,x)).$$

(c) *Abstract view on abstraction theories*

Although the theories presented in this section bear similar titles, they are quite different according to the representation level to which the abstraction is considered. The Plaisted (1981) theory considers the abstraction at the level of logical theories. Giunchiglia & Walsh (1992) have adopted a syntactical level and are mainly interested in TI abstractions. Nayak & Levy (1995) as well as Giordana & Saitta (1990) have proposed to define semantical abstractions at the interpretation model level. These latter approaches give the means to capture what makes the essence of the abstraction: a 'simplifying assumption'. Figure 9 summarizes these different contributions by contrasting the type of representation level at which they operate.

(d) *Practical utility of abstraction within representation change*

To conclude this brief state of the art of theories of abstraction, it is worth assessing the utility of abstractions in practice. The vast majority of works that mention 'abstraction' underline its positive effects, in particular performance increase. Reported speed-ups are often exponential (Giunchiglia 1996). However, in certain cases abstraction can lead to reductions in performance[13] that are worse than the best possible benefits (Backstrom & Jonsson 1995). Although Giunchiglia has shown that the negative results of Backstrom and Jonsson correspond to pathological problems (Giunchiglia 1996), it is nevertheless highly probable that there exists, for abstraction, an analogue of the no-free-lunch-theorem (Wolpert 1995). It is thus essential to choose abstractions carefully.[14] To summarize, the utility of an abstraction is related to the utility of the representation change that is associated with it. Several factors must be taken into account, including:

(i) the computational cost of the reformulation from the ground problem to the new representation;
(ii) the ratio between the complexity of the abstract problem and that of the initial problem;
(iii) the computational cost of the reformulation of the abstract solutions back to the initial representation;
(iv) the density of solutions of the initial problem within the new representation.

5. A GROUNDED DEFINITION OF ABSTRACTION

Most of the representation changes called abstraction in AI reduce the quantity of information. If the theories presented in § 4 are good at characterizing and classifying existing abstractions, they fail to offer a constructive approach to the problem. In this section, a somewhat novel perspective on abstraction is proposed that originates from the observation that the conceptualization of a domain involves, *simultaneously*, at least four different levels. Given these four levels, a theory is proposed that is grounded in the lowest level (the perception level) but offers the means to build compatible abstractions at any upper level of representation.

(a) *Representation levels and description framework*

Underlying any source of experience there is the world W, which for the sake of simplicity we assume is not changing over time. The world is not really known, because we, or artificial systems, only have a mediated access to it through our/their perception. Thus, what is important for an observer is not the world *per se*, but the perception P that she/he has of it. P specifies the nature of the elements that constitute the result of perception; for example, P may say that the perception consists of the brightness and colour of a matrix of pixels, or of a given number of objects, described by shape and size. An actual world perception P is obtained through a process $\mathscr{P}$ of signal/information acquisition from/about the world:

$$P = \mathscr{P}(W).$$

The above notation synthetically indicates that the perception process $\mathscr{P}$ applied to the world W provides particular content to the elements specified by P. For the sake of exemplification, let us consider a sensor used in molecular biology to measure the expression of genes, such as the one reported in figure 10.

P defines the elements of the world that we consider elementary (or atomic) percepts. To specify the reasons why these elements are selected is beyond the scope of this paper. Even though these elements may be rather complex, they are considered the basic building blocks of any further conceptualization. In the case of figure 10, for instance, the process $\mathscr{P}$, which underlies the Microarray detector, is the hybridization of probes with sample and control cDNA, and the perception P consists of the signal intensity generated by the sum of the fluorescence green corresponding to the concentration of sample and red corresponding to the concentration of control. A 'natural' object to consider is a spot. Nevertheless, the spot is itself built of pixels. They could equally well have been selected as basic objects.[15]

The basic stimuli in P can be categorized according to ontologies, including objects, attributes, functions and relations. More precisely, objects can be atomic or compound; atomic objects do not have parts, whereas compound objects do have parts that are themselves objects: a part-of hierarchy relates compound objects to their constituents. Both atomic and compound objects have properties, which we call *attributes*. Other types of properties involve groups of objects, resulting in functions and relations. The percepts in P, generated by $\mathscr{P}(W)$, can be grouped into four classes:

$$P = < \text{OBJ, ATT, FUNC, REL} > .$$

OBJ is the set of perceived objects, ATT is the set of perceived object attributes, FUNC is the set of perceived functional links and REL is the set of perceived relations. We underline that in W there is no explicit notion of attribute or relation, but that these are meta-notions developed by the observer through a previous, separate process, in order to avoid dealing with the original stimuli for each new task. In the case of figure 10, an object x is the signal on a spot, an attribute can be its intensity, its colour or its surface, a function might be a link between the coordinate of the spot and the identifier of the corresponding gene. Finally, a relation could specify the relative positions of two probes (this information is in fact useful to spatially normalize the signal-to-noise ratio on the chip).

At the perception level, the percepts 'exist' only for the observer, and only during the act of perceiving. Their reality consists in the stimuli sent to the observer. In the considered example of figure 10, the stimuli come from the microarray, and each occurrence of a signal in a spot will decay without leaving any long-term effect (fluorescence is naturally decaying). To allow the stimuli to become available over time, for retrieval and further reasoning, they must be memorized and organized into a structure S (Van Dalen 1983). This structure is an extensional representation of the perceived world, in which stimuli perceptually related to one another are stored together. In the example of figure 10, signals must be memorized with their attributes and organized into some structure, in order to lend themselves to their intended elaboration. A way of organizing them is to record the measures of the associate attributes values in a table inside a database. Obviously, in a natural system, stimuli from the world are elaborated by their natural perceptual systems, which have the definition of both $\mathscr{P}$ and P embedded in their anatomo-physiology, as well as suitable memorization mechanisms and appropriate relations with action. In an artificial system, storage occurs in a relational database, manipulated via relational algebra operators (Ullman 1983). We will denote by $\mathscr{M}$ the memorization process, which generates a memory structure S:

$$S = \mathscr{M}(P).$$

Finally, to describe, in a symbolic way, the perceived world, and to communicate with other agents, a language L is needed. L allows the perceived world to be described intentionally. Assigning names to the tables themselves, to the objects, attributes, functions and relations is then a process of description:

$$L = \mathscr{D}(S).$$

In the example of figure 10, for instance, the intensity of the signal on a spot will be called 'intensity', the positions of the detectors will be related to their 'x- and y-coordinates', the physical/biological contiguity of detectors will be called 'adjacency', and so on. Finally, a theory enables reasoning about the world. The theory may also contain general background knowledge, which does not belong to any specific domain. At the theory level, inference rules are used. We call theorization the process of expressing the theory in the language L (possibly enriched to accommodate domain-independent background knowledge):

$$T = \mathscr{T}(L).$$

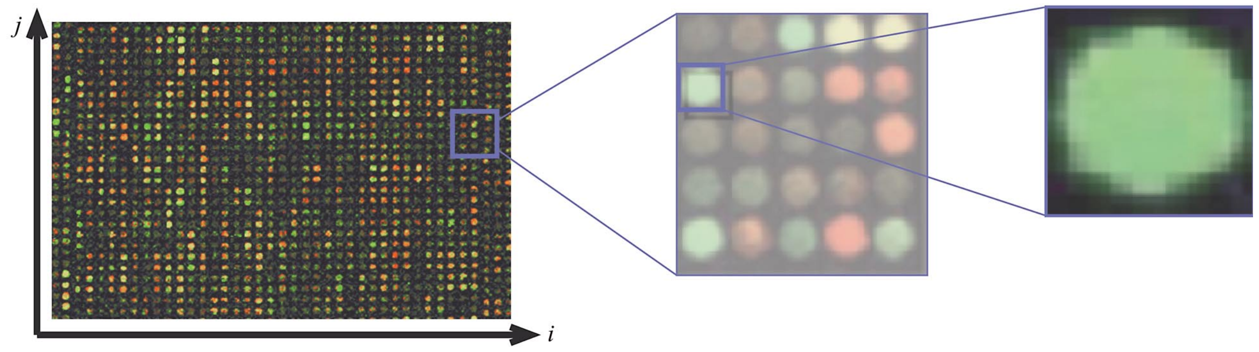


Figure 10. Picture of a microarray, a modern tool for analysing gene expression. It consists of a small membrane, or glass slide, containing samples of many genes arranged in a regular two-dimensional array (up to 40 000 genes). Each spot on an array is associated with a particular gene. The location and intensity of a colour provides an estimate of the expression level of the gene(s) in the sample (diseased) and in the control biological extract (healthy). Green and red spots represent underexpressed and overexpressed genes, respectively, in the sample versus the control.

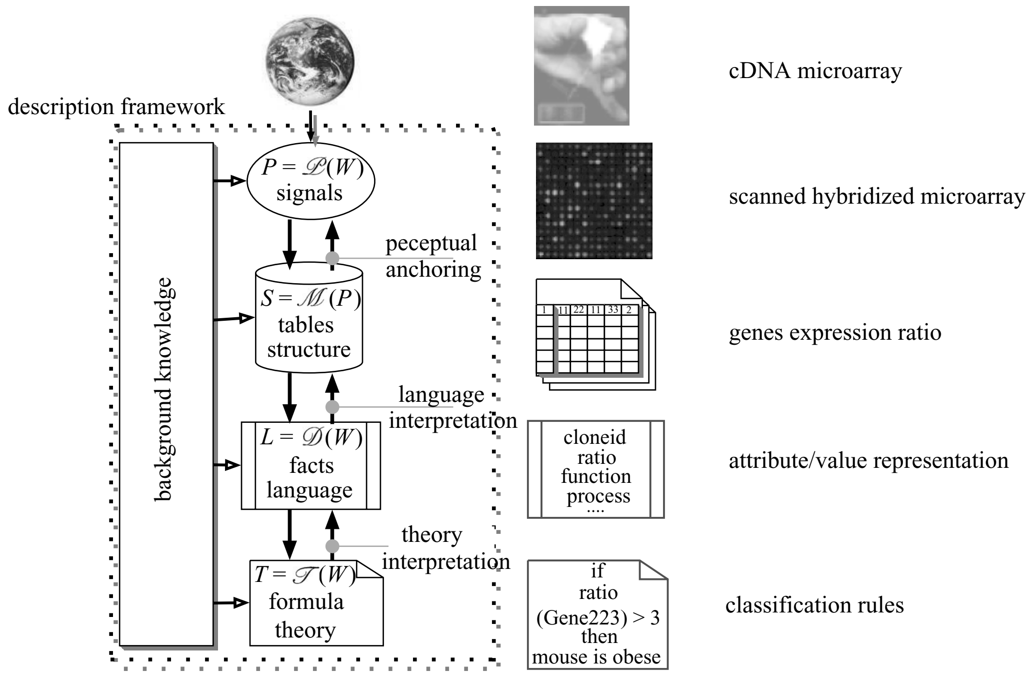

description framework
$P = \mathscr{P}(W)$ signals
peceptual anchoring
$S = \mathscr{M}(P)$ tables structure
language interpretation
$L = \mathscr{D}(W)$ facts language
theory interpretation
$T = \mathscr{T}(W)$ formula theory
background knowledge
cDNA microarray
scanned hybridized microarray
genes expression ratio
cloneid ratio function process ....
attribute/value representation
if ratio (Gene223) > 3 then mouse is obese
classification rules

Figure 11. The four levels involved in representing and reasoning about a world W in the KRA model. P denotes a perception of objects and their physical links in W. S is a set of tables, each one grouping objects sharing some property. L is a formal language, whose semantics is evaluated on the tables of S. Finally, T is a theory formulated using L, in which the properties of the world and general knowledge are embedded. General background knowledge may provide inputs at any level.

In the example of figure 10, a theory is needed to interpret the gene expression variation, hence transforming pure measurements into experimental evidence. Moreover, formal knowledge, such as the symmetry of the adjacency relation, can be added to T.

The four levels are ordered as in figure 11. They represent the basis of our KRA model. An ascending (or descending) arrow from a box X to Y on figure 11 indicates that the syntactic and semantic definition of level Y must be interpreted (or is a description) based on the content of level X. As no world is totally isolated, a body of background knowledge provides, in principle, input to each level, especially to the theory, where general laws and domain-independent facts may be needed. The primary role of perception (level P in figure 11) in biasing the upper levels is strongly advocated by Goldstone & Barsalou (1998). This dependency, however, does not mean that the content of the levels is generated strictly bottom-up: a complex bi-directional interplay may exist, and conceptualization or task information may even affect perception (Goldstone & Barsalou 1998). In figure 11, background knowledge may be the database that contains the clone identifier of each gene of each spot on the microarray.

A deeper analysis of the relations among the introduced representation levels is out of the scope of computer science, because it requires contributions from philosophy and cognitive science, at the very least. Thus, in this paper the levels are considered as given, and the nature of the $\mathscr{P}$, $\mathscr{M}$, $\mathscr{D}$ and $\mathscr{T}$ processes are not discussed. Instead, we concentrate on representational issues and define a description framework $\mathscr{D}(W)$, over a world W, as the 4-

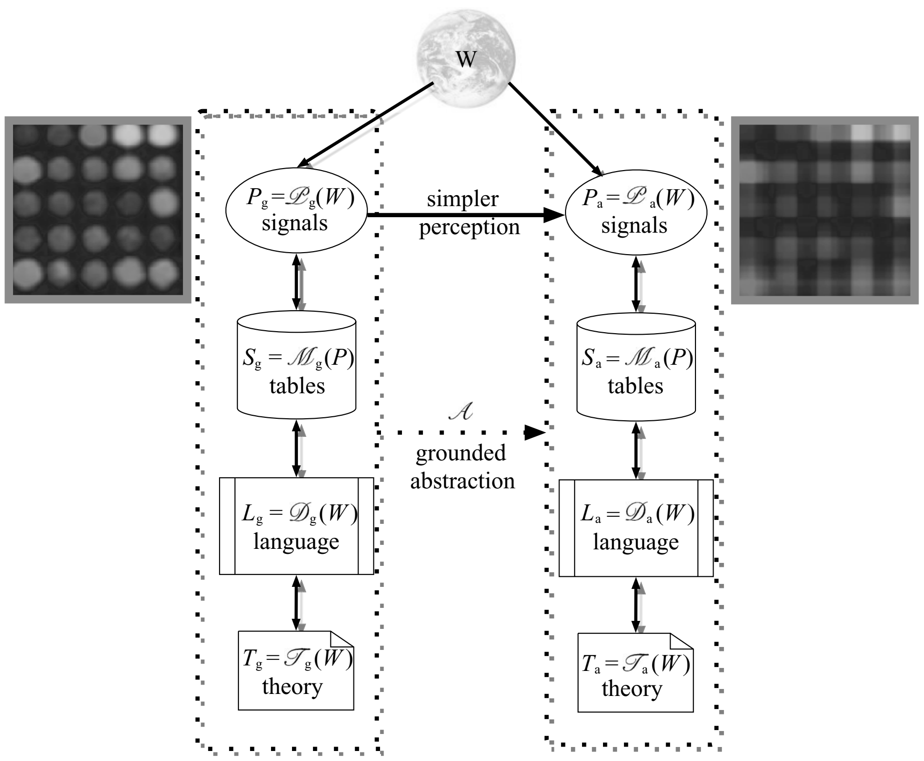

W
$P_g = \mathscr{P}_g(W)$ signals
simpler perception
$P_a = \mathscr{P}_a(W)$ signals
$S_g = \mathscr{M}_g(P)$ tables
$S_a = \mathscr{M}_a(P)$ tables
$\mathscr{A}$
grounded abstraction
$L_g = \mathscr{D}_g(W)$ language
$L_a = \mathscr{D}_a(W)$ language
$T_g = \mathscr{T}_g(W)$ theory
$T_a = \mathscr{T}_a(W)$ theory

Figure 12. A grounded abstraction is a mapping between a ground representation framework $D_g(W) = (P_g, S_g, L_g, T_g)$ to another $D_a(W) = (P_a, S_a, L_a, T_a) = \mathscr{A}(D_g(W))$ such that the perception of the latter (P_a) is simpler than that of the first (P_g). The perception on the right is simpler than the one on the left according to definition 3.2.

tuple $\mathscr{D}(W) = (P,S,L,T)$. Before proceeding to any formal definitions, it is worth spending some time on the notion of world W. In fact, it is by no means intended that the world W be restricted to the physical one, and that the perception P and the perception process $\mathscr{P}$ are only related to our five senses. A world W could be a conceptual one, for instance, a text, in which a reading process identifies words and phrases, which form a 'perception' $\mathscr{P}$ of the text (see figure 8*b*).

(b) *Formal definition of simplicity in a description framework*

Given a world W, let P be a perception of W resulting from a process $\mathscr{P}$ that uses a set of sensors, each one tailored to capture 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 $\mathscr{P}$ is called a signal pattern or a *configuration*. Let Γ be the set of possible configurations detectable by $\mathscr{P}$.

Definition 5.1 (perception simplicity; Saitta & Zucker 2001). *Given a world W, let $\mathscr{P}_1$ and $\mathscr{P}_2$ be two perception processes generating P_1 and P_2, respectively. Let Γ_1 and Γ_2 be the corresponding configuration sets. The perceived world P_2 will be said simpler than P_1 if and only if $K(\Gamma_2) \leqslant K(\Gamma_1)$, where K is the Kolmogorov complexity of a configuration set (Kolmogorov 1965; Li & Vitányi 1993).*

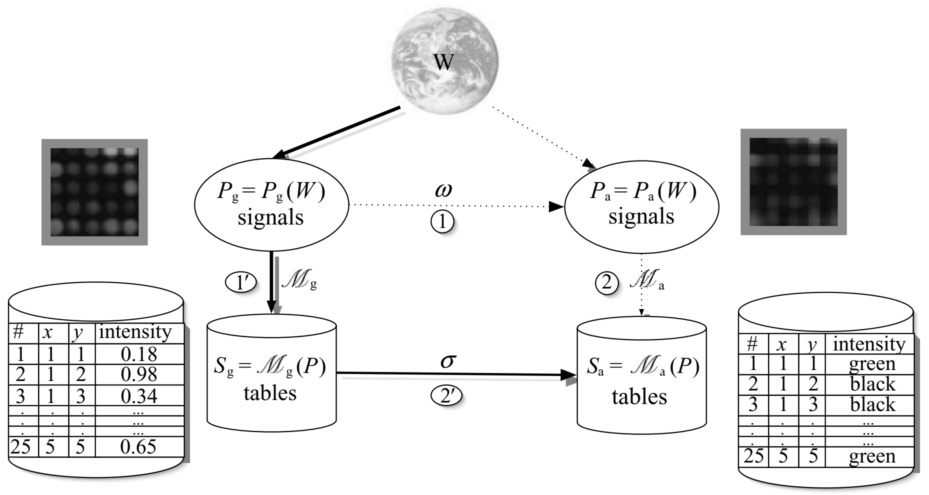
The above definition has the advantage of linking simplicity to its semantic meaning of the cognitive effort of information processing, rather than to its syntactic expression. Obviously, syntactic complexity may have an effect on the simplicity of a perceived world, when a higher syntactic complexity implies more work to handle the conveyed information. This definition is general. It does not impose any semantic link between Γ_1 and Γ_2; it only states that Γ_2 is *simpler* to describe.

This definition respects the transitive aspect of an abstraction (Iwasaki 1990; Yoshida & Motoda 1990), and chains of abstraction mappings can be considered. A discussion of what concerns the relation between simplicity and information content can be found elsewhere (Li & Vitányi 1993). Here, it is sufficient to say that abstraction is not concerned with the probability distribution of the configurations belonging to a set of Γ: just one configuration, the observed one, γ, is relevant. Moreover, the relevant problem is not recognizing the configuration, but describing it.

Given a world W, let $D_g(W) = (P_g, S_g, L_g, T_g)$ and $D_a(W) = (P_a, S_a, L_a, T_a)$ be two description frameworks over the same world W, which we conventionally label as *ground* and *abstract*, respectively. An abstraction is a mapping $\mathscr{A}$ defined as follows:

Definition 5.2 (grounded abstraction; Saitta & Zucker 2001). *An abstraction is a mapping $\mathscr{A}$ from a ground description framework $D_g(W)$ onto an abstraction $D_a(W)$ such that $P_a = \mathscr{P}_a(W)$ is simpler—according to definition 5.1—than $\mathscr{P}_g P_g(W)$.*

This definition is illustrated in figure 12. According to this definition, all the transformations introduced in § 2a–d are formally defined as abstractions, as can be demonstrated using definition 5.1.



#	x	y	intensity
1	1	1	0.18
2	1	2	0.98
3	1	3	0.34
.	.	.	...
.	.	.	...
.	.	.	...
25	5	5	0.65

#	x	y	intensity
1	1	1	green
2	1	2	black
3	1	3	black
.	.	.	...
.	.	.	...
.	.	.	...
25	5	5	green

Figure 13. An abstract structure S_a may be obtained by applying an *S*-operator σ to the ground structure S_g, given that it is compatible with the *P*-operator ω defined at the perception level.

6. ABSTRACTION OPERATORS

Section 5 provides a formal definition of abstraction as a functional mapping $\mathscr{A}$ between a given perception of a world and a simpler one. To build abstractions that are grounded on perception in a constructive manner, this section introduces the notion of abstraction operators at the levels of structure, language and theory. To guarantee that the operators at the different levels of the description framework are compatible with the abstraction defined at the perceptual level, a notion of *compatibility* is introduced.

(a) P-*operators,* S-*operators,* L-*operators and* T-*operators*

Given an abstraction $\mathscr{A}$, as defined in definition 5.2, an abstract *P*-operator ω is defined as follows:

Definition 6.1 (abstract perception operator; Saitta & Zucker 2001). *An abstraction P-operator ω denotes a procedure that takes as input a perception $P_g(W)$ of a world W and outputs a simpler perception $P_a(W)$ of the same world.*

In other words, a *P*-operator represents a *generic transformation* of world perceptions, which, applied to a particular perception, produces an abstraction of it. There exist an infinite number of transformations of a given world perception that satisfy the above definition. Among all *P*-operators, we are interested in the ones that have a certain degree of generality (universality), i.e. the ones that have larger domains of application. *P*-operators classically operate at the level of signals or a stream (be it an image, a sound, a text, …). Filters in image analysis are typical example of *P*-operators (e.g. the ones that transform the images of figures 2*a*, 3*a*, 4*a* and 5*a*).

The notion of abstract *S*-operators, *L*-operators and *T*-operators are defined in an analogous fashion (Saitta & Zucker 2001). The key idea is that each of these operators represents a type of algorithm that takes input knowledge represented at a given formalism level and produces a more abstract representation of this knowledge in the same formalism. SELECTION and DELETE are typical abstract *S*-operators on relational databases (Goldstein & Storey 1999). At the language level, XSLT[16] is an example of programming language that supports transformations. As for *T*-operators that simplify theories, the Prolog language or other dedicated *rewriting* languages are good examples of programming languages that support the writing of transformation algorithms.

(b) *Compatible operators*

Given an abstraction *P*-operator ω, there are two ways to build an abstract structure S_a. The first path consists in applying ω to the ground perception to obtain the abstract perception P_a, then 'memorizing' ($\mathscr{M}_a(P_a)$) it into the abstract structure S_a (see path 1–2 in figure 13). A second path corresponds to the use of an *S*-operator σ that operates directly on the ground structure S_g (see path 1′–2′ in figure 13).

The first path requires one to explicitly build the abstract world perception P_a and then to memorize it. Because the memorization step of a new perception is difficult to automate fully, it is more useful to have a transformation that works directly on the ground structure S_g. The notion of operator compatibility is introduced to express the fact that both the mentioned paths must lead to the same structure. This compatibility provides the semantic foundation to the abstract *S*-operators at the structure level.

Definition 6.2 (*S*-operator compatibility). *An S-operator σ, applicable at the structure level, is compatible with a P-operator ω at the world perception level, if and only if $\sigma(\mathscr{M}_g(P_g)) = \mathscr{M}_a(\omega(P_a))$.*

The compatibility of *L*-operators and *T*-operators is defined in an analogous manner. It is important to insist on the fact that abstraction operators are usually combined with reformulation operators. Whereas in approaches to abstraction based on mappings at the language level the consistency problem may emerge (Tenenberg 1987; Nayak & Levy 1995), in the grounded approach presented, consistency is guaranteed by the fact that the considered abstractions correspond to actually possible perceptions. However, a caveat is that it may be impossible to define compatible transformation rules at the language level. Goldstone & Barsalou (1998) have also mentioned this point in relation to human perception.

7. ILLUSTRATIVE EXAMPLES OF GROUNDED ABSTRACTION IN *CARTOGRAPHY*

In this section, an example is provided of the use of the presented grounded theory of abstraction in the *cartography* domain. It concerns: (i) the modelling of the knowledge acquisition process of map design; and (ii) the partial automation of a part of this process called cartographic generalization.

(a) *Abstraction in the map design process*

A domain where the description framework introduced in the previous sections finds a natural application is the *cartography* domain. Consider the process of map design (Brassel & Weibel 1988; Armstrong 1991; Mustière & Zucker 2002), described in figure 14 (steps 1–4). The map production process closely follows the model of abstraction presented in this paper. The creation of maps from geographical data is a multiple-step process that involves several intermediate representations. The first step of cartography is to collect data from the geographical world (W), or part of it. This is usually done through aerial photographs ($\mathscr{P}_g(W)$) or satellite images. The photographs produced are the perceived world P. From this picture, an expert (a photogrammetrist or a stereoplotter operator) manually extracts a GDB (also called Digital Landscape Model or DLM by Brassel & Weibel (1988)). A GDB contains the coordinates of all the points and lines that were identified by the photogrammetrist on the images. The photogrammetrist performs, simultaneously, an abstraction and a reformulation of the image. In addition, during this second step, she/he associates a category (road, building, river, field, etc.) to the lines identified. In the third step, the GDB is displayed by means of cartographic symbols applied to objects stored in it. This step is performed by a cartographer and corresponds to the choice of a language, L, which, in this case, is an iconic one, consisting of symbols such as roads, cities, buildings, rivers, etc. Finally, maps are not an end in themselves; maps are created for space and landscape analysis, for itinerary search, measuring distances, estimating population density, or other geographical theory construction (T). The map may therefore be completed in a fourth step by various types of information corresponding to the theme of the map (geology, rainfall, population, tourist, history, etc.). The distinction into levels, represented in figure 14 by the vertical axis, is only one aspect of the cartographic process of designing maps. All steps of map creation involve both knowledge representation and knowledge abstraction, because each step retains only part of the information available; a GDB does not contain all the information seen on the image by the photogrammetrist.[17] Thus, map creation is best represented as a process combining, at each step, an abstraction (a simplification) and a reformulation (a change of formalism). The KRA model introduced in § 5 supports the modelling of both the process of abstraction (change of level of detail) and the process of reformulation (change of language). Indeed, usually these two processes are closely intertwined in cartography and difficult to distinguish from one another. This distinction provides the basis for automating cartographic knowledge acquisition as a combined acquisition of specific knowledge for abstraction and knowledge for reformulation, as explained below.

(b) *Abstraction in the cartographic generalization process*

The third step is usually repeated for each desired scale (see step 3′ in figure 14), and consists in using the GDB to define the objects to be symbolized on the map (e.g. which sets of lines are considered as houses), their position (e.g. a house may have to be moved to be more readable), their level of detail (e.g. the sinuosity of a road). This step is called cartographic generalization (Muller *et al.* 1995). This operation is far more complex than a simple reduction, and it involves abstracting details so that the map is readable at the chosen scale. The basic operation that an expert uses to perform cartographic generalization is to apply various transformation algorithms to the GDB. We have developed a machine learning approach to learn how to apply these operators to produce acceptable maps. Because of the complexity of the task, it appears (both theoretically and experimentally) that a direct approach, consisting in learning from the GDB, is inappropriate. To address this problem, the modification of the representation language is a promising approach that increases both accuracy and rule comprehensibility.

The abstraction process is to progress from a detailed description of each part of a geographical object to a more global description, containing only those properties of the object relevant to the map users' needs. For example, an abstraction is to progress from a complete description of the geometry of a set of streets in a town to their description as a grid (Mustière *et al.* 1999). A concrete description of the proposed model has been done in a cartographic generalization of buildings and roads (Mustière *et al.* 2000), leading to a partial automation of the process and very significant results. As a tangible proof of the cartographic quality of the results obtained, the algorithms developed are now incorporated into the mass production of maps at the French National Geographic Institute (IGN).

8. CONCLUSION

Abstraction is an elusive notion that plays a fundamental role in both human intelligence and AI. In the latter, the common idea that lies behind most abstractions is that of a change of the representation that reduces the computational cost for solving a class of task. There are countless illustrative examples of changes of representation. They often lead to an exponential increase in problem-solving performances. In this paper, it has been argued that this pragmatic definition of abstraction does not support building representation changes that lead to simpler representation. By analysing the notion of simplicity from an information quantity point of view, the complementary roles of reformulation and abstraction in any representation change have been considered. The model we have introduced captures some important aspects of the previously proposed theory of abstractions and limits itself to grounded abstraction, i.e. abstraction that may be characterized by an information loss at the perception level. Although inherently limited, this definition of abstraction supports the definition of *abstraction operators* at different

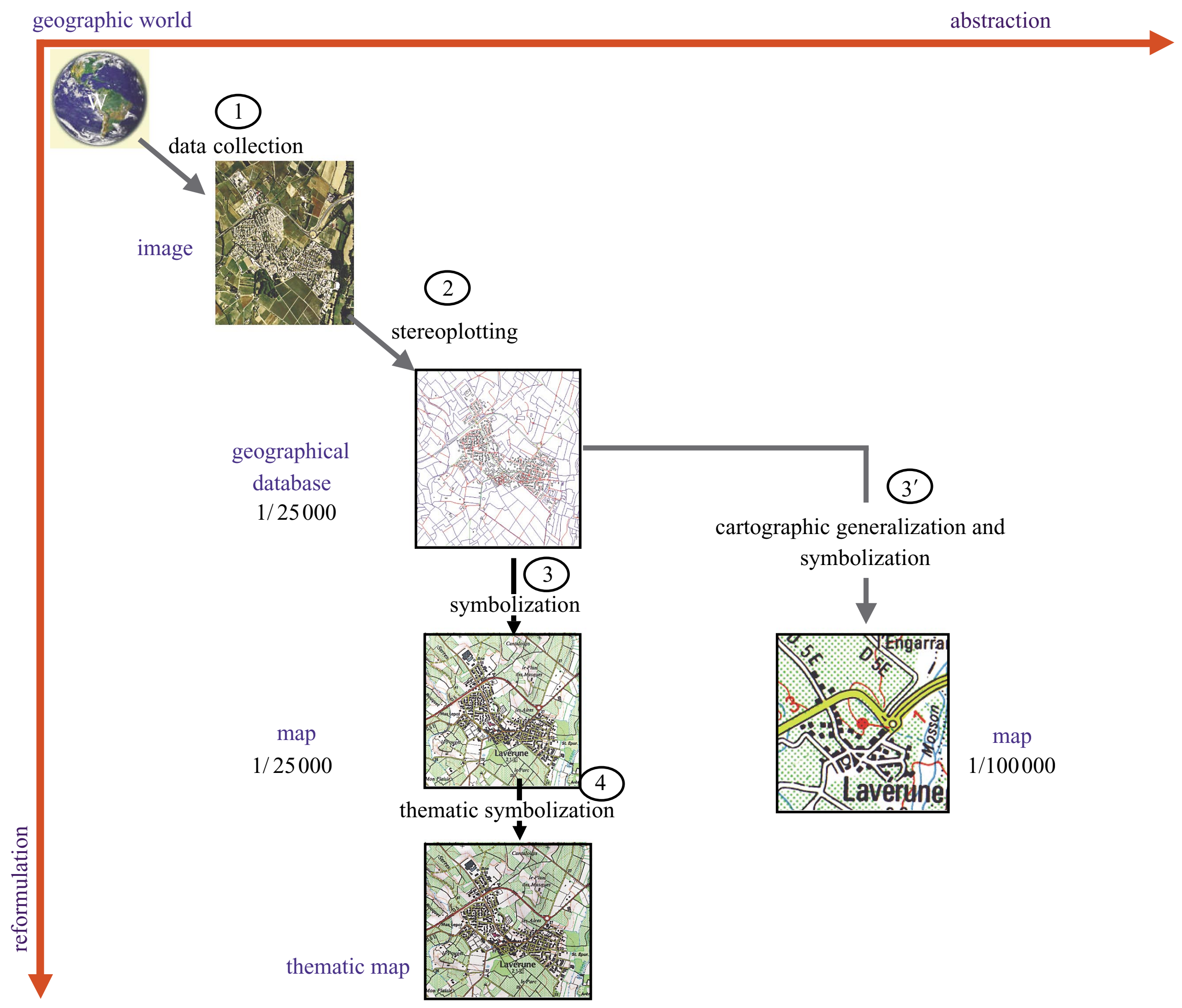


Figure 14. Two types of process are intermixed in the cartographic process of conceiving and drawing a map. They are representation changes through the different levels of representation involved in the process of map drawing (Mustière & Zucker 2002). Abstractions are represented by horizontal arrows, reformulations by vertical arrows, and other arrows represent representation changes. Such diagrams are called Reformulation/Abstraction Description (RAD) diagrams.

levels of knowledge representation (raw data, relational tables, languages, theories, etc.). These operators capture the essence of abstraction and provide a means of both characterizing existing abstraction and operationalizing the building of new abstraction at the database, language and theory level, provided compatible operators exist. Operators have proved particularly useful in formalizing the abstraction performed in cartographic generalization and significantly improved its partial automation (Mustière *et al.* 2000).

This grounded approach to abstraction still leaves open many fundamental questions. One question concerns the study of the most relevant property that operators ought to preserve for deductive, abductive and inductive reasoning. Another question concerns an algebraic formalization of abstraction operators. Such a formalization would be useful to combine abstraction operators and define the properties to be preserved. Furthermore, a comprehensive description of useful operators in the different fields of AI would offer a way to classify existing contributions to abstraction. An exciting direction for research includes the automatic change of representation by composing abstract and reformulation operators. Early experiments (Bredèche *et al.* 2003), in applying abstraction operators to explore a space of representations to improve the learning of *anchors* in autonomous robots, are a promising step for designing more autonomous and adaptive systems.

This work is a synthesis of different contributions made in collaboration with Lorenza Saitta. The author benefits from a grant to his EPML-32 CNRS IAPuces team.

ENDNOTES

[1]The words ‘mapping’, ‘function’, ‘map’, ‘operator’, ‘transformation’ and ‘morphism’ are all synonyms; they are often used in the representation change literature to denote the relation between an initial and a changed representation.

[2]Sub-sampling can be implemented by keeping only a fraction of the pixels from the original. This latter transformation is the most basic of all image compression techniques and may be described as domain hiding.

[3]The decompression of a lossy compressed image does not result in an image similar to the original one.

[4]Let us underline the ambiguity of the term *representation*, which refers to both the notion of representation *formalism* as well as a specific *description* in a given formalism.

[5]‘Changes of [logical] representation are characterized as isomorphisms and homomorphisms, correspondingly to changes of information structure and information quantity, respectively’ (Korf 1980).

[6]Definitions 3.1 and 3.2 are related to the notions of isomorphisms and homomorphisms introduced by Korf (1980).

[7]Imielinski (1987) has proposed an *approximate reasoning* framework for abstraction. He called such a kind of reasoning 'limited', because it is weaker than the general predicate logic-proof method, but it leads to significant computational advantages.

[8]Occam's razor is a logical principle attributed to the medieval philosopher William of Occam (or Ockham). The principle states that one should not make more assumptions than the minimum needed.

[9]The capacity may be esimated using the Vapnik–Chervonenkis dimension (VC-dim).

[10]A STRIPS operator is a triple (Precondition, Deletion, Addition), respectively corresponding to a set of ground facts, denoting the operator's preconditions, deleted facts and added facts. This type of representation is widely used in planning, and ABSTRIPS is a well-known abstract planner that extends STRIP abilities (Yang 1997).

[11]'Notice that we do not formally study the requirement for simplicity or any more global requirements for increased efficiency. This would require some complexity arguments which will be discussed in subsequent papers' (Giunchiglia & Walsh 1992, p. 5).

[12]This property is indeed much desired in machine learning as it used to explore the space of hypothesis that generalize examples and not counter-examples of a concept.

[13]Backstrom & Jonsson (1995) have showed that there are domains for which the ALPINE (Knoblock 1994) and HIGHPOINT (Bacchus & Yang 1994) algorithms may generate exponentially longer plans than optimal, despite the *downward refinement property*.

[14]The no-free-lunch-theorem states that there is no such thing as a universal 'best' algorithm for all possible search problems.

[15]The problem of defining what is and what is not an object is too difficult a problem. If, historically, the primary goal of computer vision has been to identify an object (Stone 1993), a unique operational definition of what an object is has not emerged. In the field of human vision, several researchers suggest that anything perceived that we can either act or talk upon may be considered as an object (Buser & Imbert 1992).

[16]XSLT (eXtensible Stylesheet Language Transformations) is a language for transforming XML documents into other XML documents.

[17]The ideal geographical dataset that would be obtained without any abstraction is called the *nominal ground*. It is considered to represent the true value for the values contained in the geographical dataset, and is used to evaluate the quality of the geographical dataset.

REFERENCES

Allen, J. 1984 Towards a general theory of action and time. *Artif. Intell.* **23**, 123–154.

Amarel, S. 1968 On representations of problems of reasoning about actions. *Machine Intell.* **3**, 131–171.

Amarel, S. 1983 Problems of representation in heuristic problem solving: related issues in the development of expert systems. In *Methods of heuristics* (ed. M. Groner, R. Groner & W. Bischov), pp. 245–350. Hillsdale, NJ: Lawrence Erlbaum.

Armstrong, M. P. 1991 Knowledge classification and organisation. In *Map generalization: making rules for knowledge representation* (ed. B. P. Buttenfield & R. B. McMaster), pp. 86–102. Essex, UK: Longman.

Bacchus, F. & Yang, Q. 1994 Downward refinement and the efficiency of hierarchical problem solving. *Artif. Intell.* **71**, 43–100.

Backstrom, C. & Jonsson, P. 1995 Planning with abstraction hierarchies can be exponentially less efficient. In *Proc. 15th Int. Joint Conf. on A.I.* (ed. M. E. Pollack), pp. 1599–1604. San Mateo, CA: Morgan Kaufmann.

Benjamin, D. P., Dorst, L., Mandhyan, I. & Rosar M. 1990 An algebraic approach to abstraction and representation change. In *Proc. AAAI Workshop on Automatic Generation of Approximations and Abstractions, Boston, MA*, pp. 47–52.

Bentley, P. & Kumar, S. 1999 Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem. In *Proceedings of the Genetic and Evolutionary Computation Conf., Orlando, USA*, vol. 1 (ed. W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela & R. E. Smith), pp. 35–43. San Mateo, CA: Morgan Kaufmann.

Bettini, C., Wang, X. S., Jajodia, S. & Jia-Ling, L. 1998 Discovering temporal relationships with multiple granularities in time sequences. *IEEE Trans. Know. Data Engng* **10**, 222–237.

Blum, A. & Langley, P. 1997 Selection of relevant features and examples in machine learning. *Artif. Intell.* **1–2**, 245–271.

Blumer, A., Ehrenfeucht, A., Haussler, D. & Warmuth, M. K. 1987 Occam's razor. *Inf. Process. Lett.* **24**, 377–380.

Bournaud, I., Courtine, M. & Zucker, J.-D. 2000 Abstractions for knowledge organization of relational descriptions. In *Lectures Notes in Computer Science 1864* (ed. B. Y. Choueiry & T. Walsh), pp. 86–106. New York: Springer.

Brassel, K. & Weibel, R. 1988 A review and conceptual framework of automated map generalization. *Int. J. Geogr. Inf. Syst.* **2**, 229–244.

Bredèche, N., Chevaleyre, Y., Zucker, J.-D., Drogoul, A. & Sabah, G. 2003 A meta-learning approach to anchor visual percepts. *Robot. Auton. Syst. J.* (Special issue on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems.)

Briot, J.-P. & Cointe, P. 1987 A uniform model for object-oriented languages using the class abstraction. In *Int. Joint Conf. on Artificial Intelligence (IJCAI '87), Milan, Italy* (ed. J. McDermott), pp. 40–43. San Mateo, CA: Morgan Kaufmann.

Brooks, R. A. 1990 Elephants don't play chess. *Robot. Auton. Syst.* **6**, 3–15.

Brooks, R. A. 1991 Intelligence without representation. *Artif. Intell.* **47**, 139–159.

Bundy, A., Giunchiglia, F., Sebastiani, R. & Walsh, T. 1996 Calculating criticalities. *Artificial Intell.* **88**, 39–67.

Buser, P. & Imbert, M. 1992 *Vision*. Cambridge, MA: MIT Press.

Choueiry, B., Iwasaki, Y., McIlraith, S. 2003 Towards a practical theory for reformulation for reasoning about physical systems. *Artif. Intell.* (In the press.)

Christensen, J. 1990 A hierarchical planner that generates its own hierarchies. In *Proc. 8th National Conference on Artificial Intelligence, 29 July–3 August 1990*. Boston, MA: AAAI Press.

Cousot, P. 2000 Abstract interpretation based formal methods and future challenges. In *Lecture notes in computer science 2000* (ed. R. Wilhelm), pp. 138–156. Heidelberg: Springer-Verlag.

Domingos, P. 1998 Occam's two razors: the sharp and the blunt. In *Proc. 4th International Conference on Knowledge Discovery and Data Mining (KDD-98), New York, 27–31 August 1998* (ed. R. Agrawal, P. E. Stolorz & G. Piatetsky-Shapiro). Boston, MA: AAAI Press.

Dougherty, J., Kohavi, R. & Sahami, M. 1995 Supervised and unsupervised discretization of continuous features. In *Proc. 12th Int. Conf. on Machine Learning* (ed. A. Prieditis & S. Russell), pp. 194–202. San Mateo, CA: Morgan Kaufmann.

Ellman, T. 1993 Synthesis of abstraction hierarchies for constraint satisfaction by clustering approximately equivalent objects. In *Proc. 10th Int. Conf. on Machine Learning, Amherst, MA* (ed. C. Sammut & A. Hoffmann), pp. 104–111. San Mateo, CA: Morgan Kaufmann.

Euzenat, J. 1995 An algebraic approach for granularity in qualitative time representation. In *Actes 14th Int. Joint Conference Artif. Intelligence, Montreal, Canada, 21–25 August 1995*, pp. 894–900. San Mateo, CA: Morgan Kaufmann.

Forbus, K., Nielsen, P. & Faltings, B. 1991 Qualitative spatial reasoning: the CLOCK project. *Artif. Intell.* **51**, 95–143.

Freuder, E. 1991 Eliminating interchangeable values in constraint satisfaction problems. In *Proc. Third Int. Conference on Innovative Applications of Artificial Intelligence* (ed. R. G. Smith & A. C. Scott), pp. 227–233. Anaheim, CA: MIT Press.

Giordana, A. & Saitta, L. 1990 Abstraction: a general framework for learning. In *Working Notes of Workshop on Automated Generation of Approximations and Abstractions, Boston, MA*, pp. 245–256.

Giunchiglia, F. 1996 Using abstrips abstractions. Where do we stand? *Artif. Intell. Rev.* **13**, 201–213.

Giunchiglia, F. & Walsh, T. 1992 A theory of abstraction. *Artif. Intell.* **56**, 323–390.

Goldstein, R. C. & Storey, V. C. 1999 Data abstractions: why and how? *Data Knowl. Engng* **29**, 293–311.

Goldstone, R. & Barsalou, L. 1998 Reuniting perception and conception. *Cognition* **65**, 231–262.

Hobbs, J. 1985 Granularity. In *International Joint Conference on Artificial Intelligence, Los Angeles, CA, USA, 1985* (ed. A. Toshi), pp. 432–435. San Mateo, CA: Morgan Kaufmann.

Holte, R. C. & Choueiry, B. Y. 2003 Abstraction and reformulation in artificial intelligence. *Phil. Trans. R. Soc. Lond.* B **358**, 1197–1204. (DOI 10.1098/rstb.2003.1317.)

Holte, R. C., Mkadmi, T., Zimmer, R. M. & MacDonald, A. J. 1996 Speeding up problem-solving by abstraction: a graph-oriented approach. *Artif. Intell.* **85**, 321–361.

Imielinski, T. 1987 Domain abstraction and limited reasoning. In *Proc. 10th International Joint Conference on Artificial Intelligence, Milan, Italy, 1987*, pp. 997–1003. San Mateo, CA: Morgan Kaufmann.

Imielinski, T. 1990 Abstraction in query processing. *J. Assoc. Comput. Machinery (JACM)* **38**, 534–558.

Iwasaki, Y. 1990 Reasoning with multiple abstraction models. In *Proc. AAAI Workshop on Automatic Generation of Approximations and Abstractions, Boston, MA*, pp. 122–133.

Kearns, M. J. 1990 *The computational complexity of machine learning*. Cambridge, MA: MIT Press.

Knoblock, C. 1994 Automatic generation of abstraction for planning. *Artif. Intell.* **68**, 243–302.

Kohavi, R. & Sahami, M. 1996 Error-based and entropy-based discretization of continuous features. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pp. 114–119. Menlo Park, CA: AAAI Press.

Kolmogorov, A. N. 1965 Three approaches to the quantitative definition of information. *Prob. Inf. Transmission* **1**, 4–7.

Korf, R. E. 1980 Towards a model for representation change. *Artif. Intell.* **14**, 41–78.

Kuipers, B. 1986 Qualitative simulation. *Artif. Intell.* **29**, 289–338.

Lavrac, N. & Flach, P. A. 2001 An extended transformation approach to inductive logic programming. *ACM Trans. Comput. Logic* **2**, 458–494.

Lavrac, N., Gamberger, D. & Turney, P. D. 1998 A relevancy filter for constructive induction. In *Feature extraction, construction, and selection: a data mining perspective* (ed. H. Liu & H. Motoda), pp. 137–154. Dordrecht, The Netherlands: Kluwer.

Lesser, V. R., Pavlin, J. & Durfee, E. H. 1988 Approximate processing in real-time problem solving. *Artif. Intell. Magazine* **9**, 49–61.

Li, M. & Vitányi, P. 1993 *An introduction to Kolmogorov complexity and its applications*. New York: Springer.

Lowry, M. 1987 The abstraction/implementation model of problem reformulation. In *International Joint Conference on Artificial Intelligence, Milan, Italy, 1987*, pp. 1004–1010. San Mateo, CA: Morgan Kaufmann.

Muller, J.-C., Lagrange, J.-P., Weibel, R. & Salgé, F. 1995 Generalization: state of the art and issues in. In *GIS and generalization* (ed. J.-C. Muller, J.-P. Lagrange & R. Weibel), pp. 3–17. London: Taylor & Francis.

Mustière, S. & Zucker, J.-D. 2002 Généralisation cartographique et apprentissage automatique partir d'exemples. In *Généralisation et représentations multiples* (ed. A. Ruas), pp. 353–368. Paris: Hermès.

Mustière, S., Zucker, J.-D. & Saitta, L. 1999 Cartographic generalization as a combination of representing and abstracting knowledge. In *Proc. ACM/GIS'99*. New York: ACM Press.

Mustière, S., Zucker, J.-D. & Saitta, L. 2000 An abstraction-based machine learning approach to cartographic generalization. In *Proceedings of the 9th International Symposium on Spatial Data Handling, Beijing, China, August 2000* (ed. A. Jeh), pp. 50–63.

Nayak, P. P. & Levy, A. Y. 1995 A semantic theory of abstraction. In *Proc. Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, 20–25 August 1995* (ed. A. Toshi), pp. 196–202.

O'Regan, J. K. 2001 Thoughts on change blindness. In *Vision and attention* (ed. L. R. Harris & M. Jenkin), pp. 281–302. Berlin: Springer.

Orlarey, Y., Fober, D., Letz, S. & Bilton, M. 1994 Lambda calculus and music calculi. In *Proc. Int. Computer Music Conf.*, pp. 243–250. San Francisco, CA: International Computer Music Association.

Plaisted, D. 1981 Theorem proving with abstraction. *Artif. Intell.* **16**, 47–108.

Provan, G. M. 1995 Abstraction in belief networks: the role of intermediate states in diagnostic reasoning. In *Conf. on Uncertainty in Artificial Intelligence, Montreal, Canada* (ed. P. Mesnard & S. Hanks), pp. 464–471. San Mateo, CA: Morgan Kaufmann.

Ram, A. & Jones, E. 1995 Foundations of foundations of artificial intelligence. *Phil. Psychol.* **8**, 193–199.

Riddle, P. 1990 Automating problem reformulation. In *Change of representation and inductive bias* (ed. D. P. Benjamin), pp. 105–124. Boston, MA: Kluwer.

Rissanen, J. 1985 Minimum description length principle. In *Encyclopedia of statistical sciences*, vol. 5 (ed. S. Kotz & N. L. Johnson), pp. 523–527. New York: Wiley.

Sacerdoti, E. 1974 Planning in a hierarchy of abstraction spaces. *Artif. Intell.* **5**, 115–135.

Saitta, L. & Zucker, J.-D. 2001 A model of abstraction in visual perception. *Appl. Artif. Intell.* **15**, 761–776.

Shahar, Y. 1997 A framework for knowledge-based temporal abstraction. *Artif. Intell.* **90**, 79–133.

Shawe-Taylor, J., Bartlett, P. L., Williamson, R. & Anthony, M. 1998 Structural risk minimization over data-dependent hierarchies. NeuroCOLT technical report, NC-TR-96-053, ftp://ftp.dcs.rhbnc.ac.uk /pub/neurocolt/tech reports.

Simons, D. J. & Levin, D. T. 1997 Change blindness. *Trends Cogn. Sci.* **1**, 261–267.

Smith, J. M. & Smith, D. 1977 Database abstractions: aggregation and generalization. *ACM Trans. Database Syst.* **2**, 105–133.

Smoliar, S. 1991 Algorithms for musical composition: a question of granularity. *Computer* (July), 54–56.

Stone, J. V. 1993 Computer vision: what is the object? In *Artificial intelligence and simulation of behaviour, Birmingham, England* (ed. A. Sloman, D. Hogg, G. Humphrey, A. Ramsay & D. Partridge), pp. 199–208. Amsterdam: IOS Press.

Subramanian, D. 1990 A theory of justified reformulations. In *Change of representation and inductive bias* (ed. D. P. Benjamin), pp. 147–167. Boston, MA: Kluwer.

Subramanian, D., Greiner, R. & Pearl, J. (eds) 1997 *Artif. Intell.* **97**. Special Issue on Relevance.

Tenenberg, J. 1987 Preserving consistency across abstraction mappings. In *Proc. IJCAI-87, Milan, Italy, 1987* (ed. J. McDermott), pp. 1011–1014.

Toelg, S. & Poggio, T. 1994 Towards an example-based image compression architecture for video conferencing. A.I. memo no. 1494, MIT, Boston, MA, June.

Ullman, J. D. 1983 *Principles of database systems*, vol. 1, 2. Rockville, MD: Computer Science Press.

Van Dalen, D. 1983 *Logic and structure*. Berlin: Springer.

Vapnik, V. N. 1995 *The nature of statistical learning theory*. New York: Springer.

Wallace, C. S. & Boulton, D. M. 1968 An information measure for classification. *Comput. J.* **11**, 185–194.

Wilson, D. R. & Martinez, T. R. 1998 Reduction techniques for exemplar-based learning algorithms. Machine learning. *Mach. Learn.* **38**, 257–268.

Wneck, J. & Michalski, R. 1994 Hypothesis-driven constructive induction in AQ17-HCI: a method and experiments. *Mach. Learn.* **14**, 139–168.

Wolpert, D. H. (ed.) 1995 *The mathematics of generalization*. Santa Fe Institute Studies in the Sciences of Complexity and Addison Wesley.

Yang, Q. 1997 *Intelligent planning: a decomposition and abstraction based approach*. Berlin: Springer.

Yoshida, K. & Motoda, H. 1990 Towards automatic generation of hierarchical knowledge bases. In *Proc. AAAI Workshop on Automatic Generation of Approximations and Abstractions, Boston, MA*, pp. 98–109.

Zucker, J.-D. & Ganascia, J.-G. 1996 Changes of representation for efficient learning in structural domains. In *Proc. 13th Int. Conf. on Machine Learning, Bari, Italy, 3–6 July 1996*, pp. 543–551. San Mateo, CA: Morgan Kaufmann.

GLOSSARY

AI: artificial intelligence
GDB: geographical database
KRA: knowledge reformulation and abstraction
MI: model increasing
TD: theorem decreasing
TI: theorem increasing