On the Concept of Generic Object: A Nonmonotonic Reasoning Approach and Examples

Leopoldo E. Bertossi
Pontificia Universidad Catolica de Chile
Departamento de Ciencia de la Computacion
Escuela de Ingenieria, Casilla 306, Santiago 22, Chile
e-mail: bertossi@ing.puc.cl
Raymond Reiter
Department of Computer Science
University of Toronto
Toronto, Canada M5S 1A4
and
The Canadian Institute for Advanced Research
e-mail: reiter@ai.toronto.edu

Abstract. In this paper we discuss the logical and commonsense intuitions behind the concept of generic object and the possibility of characterizing this concept by means of both classical first–order logic and circumscription. We also show that circumscription can be used in the characterization of the concept of generic point of a curve, as it appears in algebraic geometry. Finally, we show that several occurrences of the concept of generic object in computer science and mathematics can be shaped according to the same pattern of definition.

1 Introduction

We all have intuitions about the concept of generic object, more precisely, of generic element of a class O of objects. Since the notion of generic object seems to be a commonsense concept, it is natural to investigate the application of logical formalizations of commonsense reasoning, as they appear in artificial intelligence [24], to its characterization. In [5], the concept of generic mathematical object, more precisely, of generic triangle in geometry, was investigated from the point of view of circumscriptive commonsense reasoning [17, 21]. Sometimes, the concept has been defined precisely in the context of special mathematical theories. Examples are algebraic geometry [30, 31], set theory [8], and also model theory [27], as an abstraction and generalization of the concept appearing in set theory. The concept also appears in many areas of computer science.

In section 2, we list some of the intuitions behind the concept of generic object, and explore the possibility of formalizing them in classical logical terms. In section 3, we discuss the kind of properties that may be considered for genericity. In section 4, we discuss in general terms the application of circumscription to the characterization of the concept of generic object. In section 5, we investigate the concept of generic object as it appears in some mathematical theories, more specifically, in algebraic geometry, set theory, and model theory. In section 6 we present some examples of generic objects as they appear in computer science.

2 Intuitions Behind the Concept of Generic Object

Let us assume that there is a base theory Σ for the domain of discourse. In general we do not mention it explicitly in the formalizations that follow. We use two distinguished predicates, $O(\cdot)$ and $Gen(\cdot)$, which apply to objects in a specific class and to generic objects in that class, respectively. We further assume that the formula $\forall x(Gen(x) \to O(x))$ is entailed by Σ.

First Intuition: *If a generic object of a class has a property, then all objects in the class have that property.*

A natural attempt to formalize this intuition can be made through the following formula:

$$Gen(a) \wedge P(a) \to \forall y(O(y) \to P(y)). \tag{1}$$

In this sentence we have an implicit (second-order) universal quantification over P. Notice that in case we instantiate P with predicate Gen, all objects in the class would be generic (if there is a generic object a). This conclusion goes against our intuition of a generic object, which shows that in (1) we have to restrict the possibilities for P, in order not to get to trivial or counterintuitive situations. Also, any property appearing in (1) that applies to a generic object would be a logical consequence of the definition of the class of objects. A generic object is then not allowed to have accidental properties that are not part of the specification of the class. According to this result, if generic triangles were not right-angled, we would conclude that there are no right-angled triangles. To avoid this undesirable conclusion, we should reformulate the first intuition by constraining the range of possibilities for properties of generic objects: *If a generic object of a class has a property, then all the objects in the class have that property, except in the case of singular properties that do not qualify for genericity.*

Second Intuition: *A generic object has as properties only those properties that are explicit or implicit in the definition of the class of objects.*

This intuition says that a generic object has as properties only those properties that are logical consequence of the definition of the class of objects to which it belongs. If we try to formalize this intuition, we get first to the same formula (1). We have again a un quantification over properties P. Nevertheless, this formula captures only part of the intuition. If P is a property that is not a logical consequence of the definition, i.e., $\not\models O(x) \to P(x)$, then it is not true that $P(a)$. This part can be seen as a form of Closed World Assumption applied to generic objects. The problem with this formalization, apart from its nonmonotonicity, is that the truth value of a property can be undetermined with respect to the predicate O, i.e., neither $\models (O(x) \rightarrow P(x))$ nor $\models (O(x) \rightarrow \neg P(x))$ holds. In this case, we would jump to both conclusions: $\neg P(a)$ and $\neg\neg P(a)$. We have to restrict the possible P's in the formalization, instead of quantifying universally over all properties.

Third Intuition: *A generic object defines the class of objects to which it belongs.*

This assertion means that $O =_{def} \{x \mid Gen(a) \wedge P(a) \rightarrow P(x)\}$. In the formula defining the class O, there is an implicit universal quantification over P, that is, $O(x) \leftrightarrow \forall P(Gen(a) \wedge P(a) \rightarrow P(x))$. Direction "$\rightarrow$" gives formula (1). The other direction tells us that x belongs to the class O if it possesses all the properties of a generic object. It is clear that we have to restrict the properties P we are considering in the quantification: If a generic object does not have a property that is admissible for an element of the class, i.e., consistent with the definition of the class, then there would be no element with that property in the class. So, we are faced again with the problem of concentrating only on some particular properties for the objects in the class.

Fourth Intuition: *A generic object does not have any singular property.*

First of all, we have to say what is a singular property. Although the concept of singularity seems to import a high degree of commonsense and, accordingly, a treatment of the concept in similar terms to those of the concept of genericity should be attempted, we consider a singular property as an accidental, or contingent, property of objects. An accidental property is a possible but not necessary property of the objects. More precisely, P is singular if $\exists y(O(y) \wedge P(y))$ and $\exists y(O(y) \wedge \neg P(y))$ are both true. The second fact is more relevant because it tells us that property P is not inherent to the definition of class O. Then, we could attempt to formalize the concept in the form $\exists y(O(y) \wedge \neg P(y)) \rightarrow (Gen(a) \rightarrow \neg P(a))$. The contrapositive of the implication gives formula (1). As before, a universal quantification over P is implicit. Then, the problems already mentioned in the formalization of the first intuition appear again.

3 Properties that Are Relevant to Genericity

The moral that we can draw from the considerations above is that it is necessary to concentrate on some particular properties in order to formalize the concept of generic object. Actually, in all the previous attempts to formalizations, if we allow P to be an arbitrary property, then a generic object is indistinguishable, in the second-order language, from a non generic object, which is not desirable. How we restrict and choose relevant properties depends on the context.

Let us give an example illustrating the previous ideas. Assume that we have a base theory Σ, i.e. a set of sentences in a formal language $\mathcal{L}$. Σ might be the theory that describes in $\mathcal{L}$ a given model (or structure) $\mathcal{M} = \langle M, O^M, a^M, \ldots \rangle$. In $\mathcal{L}$ we have a predicate symbol O and a (tuple of) name(s) a to denote the class of objects O^M and the distinguished individual(s) a^M, respectively. For example, Σ could be the theory of a model of geometry, and a binary predicate $O(\cdot,\cdot)$ could denote the class of pairs of different and intersecting lines. A pair $a^M = (l_1, l_2)$, with $(l_1, l_2) \in O^M$, would be non generic if there is a property (formula) $\psi(x)$ such that: $\psi(a) \in \Sigma$, but $\forall\, \bar{x}(O(\bar{x}) \rightarrow \psi(\bar{x})) \notin \Sigma$.

In the model depicted in Figure 1, the pair of lines in (a) would be considered generic, but not so the pair in (b). Why? The reason is that in the first case we are not able to *say* anything more specific about (l_1, l_2) beyond the fact that $(l_1, l_2) \in O^M$ (i.e., anything not implied by $O(a)$). In the second case, instead, we can *say* that l_1 and l_2 form a right angle. The key in this example can be found in the word "say" above: *What we can say is restricted by the language that is available to us.*

Fig. 1. Pairs of Intersecting Lines

It is so common that our geometric languages allow us to talk about right angles, that we took it for granted that it is possible to do so in $\mathcal{L}$ and, in consequence, that it is possible to singularize specific properties of (l_1, l_2). According to this reasoning, (l_1, l_2) would lose its genericity if we had a way of talking in $\mathcal{L}$ about the measures (in degrees) of the angles that the lines form. Even worse, if we were only allowed to talk about "forming a right angle", the first example would not be generic either because it has the specific property of "not intersecting at a right angle", which our language allows us to state. So, our first case is not generic either.

One could argue that we should not give the same status to the properties "intersecting at right angle" and "not intersecting at right angle". After all, the former is an immediate, positive property, but not the latter. We know that this distinction is rather tricky because it is not difficult to transform negative properties into positive ones by redefining predicates. In addition, we could be interested in properties that are more involved in their definitions, in which case their evaluation in terms of "immediateness" or "positiveness" is less natural.

Despite the problems with this approach, it still seems intuitive and attractive.

If we want to keep it in some form, it seems necessary to think at a metalevel which are the properties we consider for (non)genericity. At this point is where commonsense comes in. Asking for the genesis of the singularization of specific properties in connection with the concept of generic object is more or less the same as asking for the source of defaults or, more generally, for the source of commonsense knowledge. This is an area where much research has still to be done. Nevertheless, a way of achieving a characterization of genericity at an object level, which is at the same time general and uniform, is by using some formal system for doing commonsense reasoning. For example, a system based on circumscription.

4 What about a Nonmonotonic Approach to the Concept of Generic Object?

In the following, we try to be consistent with the ideas and objections raised in the previous sections, and to keep most of the problematic attempts to formalize the concept of generic object given there, but restricting the predicates that are relevant to genericity to a given list. We also detect if there exists some similarity to a circumscriptive approach.

Assume that we have a base theory Σ in a formal language $\mathcal{L}$ that has a predicate symbol O and a name a (or tuple of names) to denote the class of objects and a (tuple of) distinguished individual(s), respectively. Let $S_1, \dots, S_n$ be new predicate symbols. Usually, there are definitions for these predicates in the theory. We are thinking of the S_i's as the singular properties of objects in O. We may further assume that $\Sigma \vdash \exists\, y(O(y) \wedge \neg S_i(y))$ for $i = 1, \dots, n$. That is, the S_i's are accidental properties of objects.

We want a to be a generic object. Intuitively, we do not want a to have any of the singular properties $S_1, \dots, S_n$. The approach followed in [5] —where the base theory was elementary geometry, a was a triangle, and a singular property was right-angledness— consists in circumscribing with scope [9] the singular properties in the sentence given by the base theory plus the fact $O(a)$. Accordingly, in the present case we would consider a as a generic object in O with respect to Σ if it satisfies the specification $\mathit{Circum}_{scoped}(\Sigma';\ W(P,x))$, that is, the scoped circumscription[1] of formula $W(P,x)$ in the theory Σ', where: (1)

[1] We consider a second-order version of scoped circumscription.
$\mathit{Circum}_{scoped}(A;\ W(\bar{P},\bar{x}))$, the *scoped circumscription* of formula $W(\bar{P},\bar{x})$ in the theory A, with variable predicate parameters $\bar{P}$ and *Scope*, is given by the conjunction of A with the second-order sentence:

$$A(\bar{P}', \mathit{Scope}') \wedge \forall\, \bar{x}[W(\bar{P}',\bar{x}) \wedge \mathit{Scope}'(\bar{x}) \supset W(\bar{P},\bar{x}) \wedge \mathit{Scope}(\bar{x})]$$
$$\supset \forall\, \bar{x}[W(\bar{P},\bar{x}) \wedge \mathit{Scope}(\bar{x}) \supset W(\bar{P}',\bar{x}) \wedge \mathit{Scope}'(\bar{x})].$$

With scoped circumscription, the scope of the domain of individuals we are reasoning about, more precisely, the restriction of predicate W to *Scope*, is kept narrow by

$P = (S_1, \ldots, S_n)$, is the tuple of singular predicates; (2) $W(P, x)$ is the formula $S_1(x) \vee \ldots \vee S_n(x)$, i.e., $W(P, x)$ says that x has some singular property; and (3) Σ' is the new theory given by the conjunction of the following sentences: (a) Σ, (b) $\exists x(O(x) \wedge \neg W(P, x))$ (there are nonsingular objects), (c) $O(a)$, and (d) $Scope(a)$.

Remark 1 Special and interesting cases are those where: 1. Conjuncts (a) and (b) in Σ' are simultaneously consistent (maybe (b) follows from (a)), and 2. Predicate *Scope* appears only in (d), and tuple a appears only in (c) and (d). ■

With our circumscriptive specification, we hope to obtain:

$$Circum_{scoped}(\Sigma';\ W(P, x)) \models \neg W(P, a), \tag{2}$$

that is, the genericity or nonsingularity of a should be a circumscriptive consequence of the theory. Under certain general circumstances this will be the case:

Proposition 1. ([9]) If $A \not\vdash \exists x(W(P, x) \wedge Scope(x))$, and the following hypothesis **H**: *"A entails a domain-closure axiom and decides the equality predicate for all ground terms of the language"* holds, then $Circum_{scoped}(A;\ W(P, x)) \vdash \forall x(Scope(x) \rightarrow \neg W(P, x))$, provided all predicates are variable. ■

From this proposition we obtain:

Proposition 2. If Σ' satisfies hypothesis **H** in Proposition 1, and also 1., 2. in Remark 1, then (2) holds.

Proof: From 1. and 2. in Remark 1, we obtain that Σ' is consistent. Then, Σ' is also consistent with $\forall x(Scope(x) \rightarrow \neg W(P, x))$, because a model $\mathcal{M} = \langle M, P^M, \ldots \rangle$ of Σ' can be modified, if necessary, in such a way that a^M, the interpretation of a in $\mathcal{M}$, is a point in O^M that does not satisfy $W(P, x)$ in $\mathcal{M}$. We may choose $Scope^M = \{a^M\}$; then, $\Sigma' \not\vdash \exists x(Scope(x) \wedge W(P, x))$. By Proposition 1, $Circum_{scoped}(\Sigma';\ W(P, x)) \models \forall x(Scope(x) \rightarrow \neg W(P, x))$; from which we obtain $Circum_{scoped}(\Sigma';\ W(P, x)) \models \neg W(P, a)$. ■

This result depends on condition **H**. Nevertheless, as pointed out in [9], this condition is sufficient, but not necessary[2]. Whether we will succeed in obtaining (2) will depend on the context.

Assume that $Circum_{scoped}(\Sigma';\ W(P, x)) \models \neg W(P, a)$. That is, $\bigwedge_1^n \neg S_i(a)$ is a circumscriptive consequence of Σ'. This fact will be written in the form $\Sigma' \vdash_{circum} \bigwedge_1^n \neg S_i(a)$. Following our convention of keeping the base theory Σ implicit, we simply write this as follows:

$$O(a) \wedge \exists y(O(y) \wedge \bigwedge_1^n \neg S_i(y)) \vdash_{circum} \bigwedge_1^n \neg S_i(a). \tag{3}$$

minimization.

[2] A counterexample can be found in [5], where an intuitively expected result of this kind —the nonsigularity of a triangle a— was obtained with scoped circumscription. In that application, condition **H** was not only unnecessary, but also false.

Now, we can use "a deduction theorem" for circumscription:

Lemma 3. If $(\Sigma \wedge \varphi) \vdash_{circum} \psi$, then $\Sigma \vdash_{circum} (\varphi \to \psi)$.

Proof: Let $\mathcal{M} = \langle M, O^M, S_1^M, \ldots, S_n^M, Scope^M, a^M, \ldots \rangle$ be a model of the scoped circumscription of $Scope \cap \bigcup_1^n S_i$ in Σ with variable predicates $O, Scope, S_1, \ldots$. Then, $\mathcal{M}$ is a $(Scope \cap \bigcup_1^n S_i;\ Scope, O, S_1, \ldots)$-minimal model of Σ. Assume that $\mathcal{M} \models \varphi$. We have to show that $\mathcal{M} \models \psi$. This proof is easy because in that case $\mathcal{M}$ turns out to be a model of $\Sigma \wedge \varphi$. But it is also a $(Scope \cap \bigcup_1^n S_i;\ Scope, O, S_1, \ldots)$-minimal model since minimality arises from a preorder relation in a class of interpretations. By the hypothesis in this lemma, we have $\mathcal{M} \models \psi$. ∎

Then, (3) can be reformulated in the form $O(a) \vdash_{circum} \exists\, y(O(y) \wedge \bigwedge_1^n \neg S_i(y)) \to \bigwedge_1^n \neg S_i(a)$, which is equivalent to $O(a) \vdash_{circum} \bigvee_1^n S_i(a) \to \forall\, y(O(y) \to \bigvee_1^n S_i(y))$. If we add to the base theory the definition $\forall\, x(S(x) \leftrightarrow \bigvee_1^n S_i(x))$, then this can be written in the form:

$$O(a) \vdash_{circum} S(a) \to \forall\, y(O(y) \to S(y)). \tag{4}$$

Actually, if it is possible to obtain the stronger conclusion $O(a) \vdash_{circum} S_i(a) \to \forall\, y(O(y) \to S_i(y)),\quad i = 1, \ldots, n;$ then, we could have instead of (4)[3]:

$$O(a) \vdash_{circum} \bigwedge_i^n (S_i(a) \to \forall\, y(O(y) \to S_i(y)). \tag{5}$$

The right-hand sides of formulas (4) and (5) can be seen as versions of formula (1), but where only a restricted universal quantification over properties of objects appears. This result means that a circumscriptive characterization of generic objects is, first, close to the initial intuitions behind the concept and, second, close to a natural, but problematic, attempt to a direct (non circumscriptive) specification in usual second-order logic.

5 Genericity in Mathematics

In 5.1, we consider the concept of generic object in algebraic geometry. In 5.2, we consider yet another intuition behind the concept of generic object and its materialization in set theory and model theory.

[3] Of course, this will not be always possible. In geometry, it is not possible to have a triangle that is neither scalene nor obtuse.

5.1 Generic Points in Algebraic Geometry

In algebraic geometry, we find the concept of generic point of a curve [30, 31]. We show, first, that the intuition behind this concept corresponds to the general intuitions we presented before and, second, that this notion can be characterized by means of circumscription.

Let $\mathbb{C}$ be the field of complex numbers, $\mathbf{K}$ an extension field, and $\mathcal{C}$ a curve in, say, $\mathbf{K}^2$, with equation $F(\bar{x}) = 0$, where $F(\bar{x}) \in \mathbb{C}[\bar{x}]$ is an irreducible polynomial with coefficients in $\mathbb{C}$. A point $\bar{a} \in \mathbf{K}^2$ is a *generic point* of $\mathcal{C}$ if: (a) $F(\bar{a}) = 0$ (i.e., $\bar{a}$ belongs to the curve $\mathcal{C}$), and (b) for all $G(\bar{x}) \in \mathbb{C}[\bar{x}]$, if $G(\bar{a}) = 0$, then for all points $\bar{p} \in \mathbf{K}^2$ that belong to the curve $\mathcal{C}$, $G(\bar{p}) = 0$.

Example:[4] Let us start from the field of complex numbers $\mathbb{C}$. We may introduce a new parameter (or indeterminate) t and pass to the extension field $\mathbf{K} = \mathbb{C}(t)$ of rational functions in t, with complex coefficients. Let $F(x, y) = x^2 + y^2 - 1$, be a polynomial in x and y with complex coefficients. This polynomial is irreducible in the ring $\mathbb{C}[x, y]$ of polynomials in x, y with complex coefficients, that is, there is no pair of polynomials in $\mathbb{C}[x, y]$ of degrees smaller than 2 that factorize $F(x, y)$. Furthermore, $F(x, y)$ defines a curve $\mathcal{C}$ on the plane $\mathbf{K}^2$, namely the set of points in $\mathbf{K}^2$ that satisfy the equation $x^2 + y^2 - 1 = 0$. For example, the point $(\frac{2t}{t^2+1}, \frac{t^2-1}{t^2+1}) \in \mathbf{K}^2$ lies on the curve. Actually, this point is a generic point of the curve. ■

We can see that this definition of generic object corresponds to our first intuition: A point on the curve $\mathcal{C}$ (our class of objects) is generic if *all its properties* are also possessed by the other points on the curve. In this case "*all its properties*" refers to the algebraic properties that are definable by polynomial equations with coefficients in the ground field $\mathbb{C}$. A circumscriptive characterization —of the form indicated in (2)— of generic points of a curve should coincide with the algebraic definition.

The formal language has the following symbols: (a) Constants: $0, 1, a_1, a_2$ ($\bar{a} = (a_1, a_2)$ stands for the generic point); (b) Predicates: C (a unary predicate for the set of complex numbers or the base field), $\mathcal{C}$ (a binary predicate for the bidimensional curve), and S (a binary predicate for a set of singular points); (c) Functions: $+$ and $\cdot$ (for the usual field operations), and $p(x, y)$ for a fixed polynomial in two variables.

We consider the following axioms: (1) Axioms for fields; (2) Axioms for fields relativized to C; (3) Axioms stating that C is an algebraic substructure of the intended field (the extension field of C), that is, the axioms: $C(0)$, $C(1)$, $\forall xy(C(x) \wedge C(y) \rightarrow C(x + y) \wedge C(x \cdot y))$; (4) $\exists x \neg C(x)$ (C is a proper subfield); (5) An infinite, but recursive, list of axioms (a schema) stating that C is algebraically closed, that is, for each natural number $n > 0$, the axiom $\forall x_0 x_1 \cdots x_n (\bigwedge_{i=0}^{n} C(x_i) \wedge x_n \neq 0 \rightarrow \exists x(C(x) \wedge x_0 + x_1 \cdot x + \cdots + x_n \cdot x^n =$

[4] John McCarthy made the authors aware of the concept of generic point in algebraic geometry; he gave us this example.

0)); (6) An infinite, but recursive, list of axioms (a schema) stating that C has characteristic zero, that is, for each natural number $n > 0$, the axiom $1 + \cdots + 1$ (n times) $\neq 0$; (7) A definition of $\mathcal{C}$ as an algebraic curve, i.e., the axiom $\forall\, x, y(\mathcal{C}(x, y) \leftrightarrow p(x, y) = 0)$, where $p(x, y)$ is an operation symbol standing for a fixed polynomial in x, y; (8) Axioms stating that $p(x, y)$ is a polynomial with coefficients in C: $\exists\, x_0 \cdots x_m (\bigwedge_{i=0}^{m} C(x_i) \wedge \forall\, xy(p(x, y) = \sum_{i=0}^{m} x_i \cdot x^{h_i} \cdot y^{k_i}))$ ($m, h_0, \ldots, h_m, k_0, \ldots, k_m$ are fixed, we might also use some extra constants for the coefficients of the polynomial); (9) A finite list of axioms stating that $p(x, y)$ is nontrivial, i.e., $\forall\, z(C(z) \rightarrow \exists\, xy(p(x, y) \neq z))$, and irreducible, that is, there is no pair of polynomials with coefficients in C of degree smaller than the degree of $p(x, y)$ that factorize $p(x, y)$.

This is our base theory Σ. In addition, we introduce two extra binary predicates S and *Scope*. By circumscribing the singularity predicate S, we expect to obtain the genericity of a point $\bar{a}$ on the curve $\mathcal{C}$. For this purpose, we will assume that the axiom $\forall\, x\, \forall\, y(S(x, y) \leftrightarrow \exists\, z(z \neq 0 \wedge C(z \cdot y) \wedge C(z \cdot y)))$ that defines predicate S is included in theory Σ. S is the set of *ground points* of the plane based on the extension field. Ground points in $\mathbf{K}^2$ can be reduced, by multiplication of its coordinates by a common non-zero factor in $\mathbf{K}$, to points with both coordinates in $\mathbb{C}$. For example, $(t + ti, t)$ is a ground point of $\mathbb{C}(t) \times \mathbb{C}(t)$. Non ground points are called *transcendental*. The point $(\frac{2t}{t^2+1}, \frac{t^2-1}{t^2+1})$ is transcendental in $\mathbb{C}(t) \times \mathbb{C}(t)$. For all practical (and theoretical) purposes, we may assume that the ground points are those of the form (c, d) with c and d in the ground field $\mathbb{C}$.

We depart from the approach in [5], in the sense that, in this case, we consider point $\bar{a}$ as variable in the scoped circumscription.

Proposition 4. $\mathit{Circum}_{scoped}(\Sigma \cup \{\mathcal{C}(\bar{a}), \mathit{Scope}(\bar{a})\};\ S;\ \bar{a}) \models \neg S(\bar{a})$.

Remark 2 1. By definition of scoped circusmcription, predicate *Scope* and the parameters that appear in the definition of predicate S are variable in the circumscription. In this case, we also make explicit the fact that point $\bar{a}$ is allowed to vary in the minimization process.
2. Proposition 4 tells us that the scoped circumscription of Σ plus "$\bar{a}$ belongs to the curve" and "$\bar{a}$ is in the scope of our default reasoning" entails that $\bar{a}$ is not a singular point.
3. As was shown in [5], this circumscription is the same as circumscribing, in the classical form, the defined predicate $S \cap \mathit{Scope}$ in the theory obtained from $\Sigma \cup \{\mathcal{C}(\bar{a}), \mathit{Scope}(\bar{a})\}$ plus the definition of predicate S.
4. Since the base theory is infinite, our circumscription is not given in terms of a finitary first–order formula[5]. Nevertheless, there is no problem in defining such

[5] See [15] for some remarks on circumscription for infinitary theories. But, if we are willing to give up a first–order base theory, we may obtain a syntactical version for the circumscription by starting from a finite theory Σ' corresponding to a categorical second–order axiomatization of the complete ordered field of real numbers. This approach would require a new unary predicate R for real numbers. We might elim-

a circumscription in semantical terms, that is, in terms of minimal models of the infinite theory. ∎

Proof: (of Proposition 4) Let $\mathfrak{K} = \langle K, C^K, +^K, \ldots, \mathcal{C}^K, S^K, Scope^K, \bar{a}^K \rangle$ be a $(S \cap Scope)$-minimal model of $\Sigma \cup \{\mathcal{C}(\bar{a}), Scope(\bar{a})\}$ plus the definition of predicate S. Assume that $\bar{a}^K \in S^K$, that is, a_1^K, and $a_2^K \in C^K$. Then, $\bar{a}^K \in S^K \cap Scope^K$.

Since $\mathcal{C}^K$ correspond to an irreducible C-algebraic curve in $K \times K$, we know by results from algebraic geometry that $\mathcal{C}$ has a transcendental (non ground) point, say, $\bar{b} = (b_1, b_2)$. We can use this point to give a new model $\mathfrak{K}'$ for the initial theory: Leave all the interpretations as in $\mathfrak{K}$ except for $\bar{a}$ and $Scope$, which are interpreted as $\bar{b}$ and $\{\bar{b}\}$, respectively.

Now, we have: $S^{K'} \cap Scope^{K'} = \varnothing \subsetneqq \{\bar{a}^K\} \subseteq S^K \cap Scope^K$. This inclusion contradicts the minimality of model $\mathfrak{K}$. ∎

The proof of Proposition 4 was simple because we allowed point $\bar{a}$ to vary in the circumscription. Despite the variability of $\bar{a}$, the use of *Scope* was decisive in the proof. The same proof does not work without using this auxiliary predicate. A natural question is whether we could get by without scoped circumscription, but allowing the point to vary[6]. In [5], some heuristic reasons were given for the failure of this alternative approach. The proof above provides additional evidence in that direction. Since these are not conclusive answers, the matter should be explored further.

From Proposition 4 and the fact that a point is a generic point of a curve if and only if it is transcendental (see [30]), we obtain the genericity of point $\bar{a}$:

Corollary 5. For every $n > 0$ and formula $\varphi(x, y, x_1, \ldots, x_n)$ of the form $\sum_{i=1}^{n} x_i \cdot x^u \cdot y^v = 0$, which defines an algebraic curve in x, y with coefficients $x_1, \ldots, x_n$,

$$Circum_{scoped}(\Sigma \cup \{\mathcal{C}(\bar{a}), Scope(\bar{a})\};\ S;\ \bar{a}) \models \forall x_1, \ldots, x_n(\varphi(\bar{a}, x_1, \ldots, x_n) \rightarrow \forall x, y(\mathcal{C}(x, y) \rightarrow \varphi(x, y, x_1, \ldots, x_n))).$$

∎

In [32], Thom shows connections between the concept of generic point in algebraic geometry and the concept of generic object in catastrophe theory and differential dynamical systems. Thus, we foresee applications of circumscription to the characterization of this central concept in two other important areas of mathematics.

5.2 Another Intuition and Forcing

Besides the intuitions behind the concept of generic object that were discussed in section 2, there is yet another one, with stronger mathematical contents:

inate schemas (5) and (6) in Σ by introducing a new constant i and the axioms: $(C(i) \wedge i \cdot i + 1 = 0)$, and $\forall x(C(x) \rightarrow \exists yz(R(y) \wedge R(z) \wedge x = y + z \cdot i)$.

[6] This question was raised to the authors by Vladimir Lifschitz. This idea was influential in letting point $\bar{a}$ vary in our scoped circumscription.

Fifth Intuition: *If it is possible to prove something using (the properties of) a generic object, then it is possible to prove the same without the object*[7].

This is the intuition behind Cohen's forcing method for establishing the independence of some axioms of set theory, but in his case the concept of generic set has a precise meaning inside the theory. Actually, his idea consists in constructing a model, say of the Zermelo-Fraenkel axioms ZF of set theory plus the negation of the axiom whose independence is to be established, by adjoining a *generic* set G to a given model $\mathcal{M}$ of ZF, and thus obtaining a new model $\mathcal{N}$ with the desired properties. The intuition that "what can be proved with the generic G can also be proved without G" is made precise and established in the so-called Cohen's Truth Lemma that tells us that a proposition is true in the extended model $\mathcal{N}$ if and only if it is forced to hold by a condition imposed on G in the original model $\mathcal{M}$.

Cohen's idea of using forcing for the construction of models of set theory was developed by Abraham Robinson [27] in the more general context of model theory [7][8]. In this model-theoretic forcing, generic objects are outside the object level language or outside the universe of the model; they are *generic sets of sentences* in the language expanded with new names for individuals. The "generic model theorem" shows that, starting from a generic set of sentences G, it is possible to construct a *generic* model $M(G)$, such that each element of the universe is denoted by one of the new constants, and the sentences satisfied by $M(G)$ are exactly those sentences that are forced by a condition in G. Another theorem establishes that $M(G)$ is a model of the base theory Σ if Σ is an inductive theory, that is, if it can be axiomatized by $\forall\exists$-sentences. A detailed treatment of model-theoretic forcing can be found in [14].

A natural question that arises is whether generic models of theories correspond to models obtained by means of formalisms for commonsense reasoning. A generic model has the "problem" of being too big, in the sense of being existentially closed, that is, it must satisfy every existential sentence that is satisfied by any algebraic superstructure. On the other hand, models obtained by circumscription are minimal in a precise sense. For example, consider the very simple inductive theory $\Sigma = \{P(a) \vee P(b)\}$. A P-minimal model of Σ is $\mathcal{M} = \langle \{a\}, \{a\}, a, a \rangle$.

---

[7] This intuition is not far from a common mathematical practice: When we try to prove, under given hypothesis, that all objects have a given property, we usually consider a "new" object a that does not appear explicitly mentioned in the hypothesis, and prove that a has the given property. Accordingly, we consider as proven that all objects have the property since they inherit the property from that "generic" object. This kind of reasoning is sanctioned in some formal deductive systems through a "generalization rule".

[8] This is not surprising, if we think that set–theoretical forcing reminds us of the introduction of generic points in algebraic geometry, an older subject. It is also interesting that Robinson made a generalization and abstraction of the ideas behind generic points in algebraic geometry and introduced them in model theory, before the emergence of the concept of forcing [25, 26, 28].

This model cannot be generic because it is not existentially closed: Its superstructure $\langle\{a,b\},\{a,b\},a,b\rangle$ satisfies the sentence $\exists\, x(x \neq a \wedge P(x))$.

6 Some Examples of Genericity in Computer Science

In this section we show how the concept of generic object appears in different areas of computer science, either explicitly or implicitly. We also try to make clear that, in essence, all those occurrences follow the same pattern of definition, namely, that a generic object having certain — well determined — properties implies that all other objects in the class have those properties.

6.1 Generic Databases

In [4], Armstrong gives a set of inference rules for functional dependencies in databases. This system of inference rules is complete and sound in the sense that it is possible to obtain all and only the functional dependencies that are logical (semantical) consequences of a given set of functional dependencies, by application of the rules. This result can be obtained [10, 19, 29] from the following preliminary result due to Armstrong: "If F is a set of functional dependencies and F^+ denotes the deductive closure of F with respect to the inference rules, then there is a relation R such that for any functional dependency φ, $\varphi \in F^+$ if and only if φ is true in R".

According to this proposition, R satisfies all the dependencies in F, and we can check whether an arbitrary functional dependency deductively follows from F by checking its validity in the relation R. R is called an Armstrong's relation for F. We have the following situation: (1) Relation R satisfies F, and (2) For every functional dependency φ: If R satisfies φ, then every relation R' that satisfies F also satisfies φ.

Thus, we can say that R is a *generic* relation in the class $\mathcal{O}$ of relations that satisfy F: All the relations in the class inherit the functional dependencies (the properties) from the generic relation.

6.2 Generic Models of Sets of Clauses

Let $L(S)$ be a first-order language, U_H the Herbrand Universe, and B_H the Herbrand Base [18]. A Herbrand interpretation for L(S) has U_H as its universe, and each n-ary function symbol $f \in S$ is interpreted by the function that sends each $(t_1,\ldots,t_n) \in U_H^n$ to the ground term $f(t_1,\ldots,t_n)(\in U_H)$. The interpretation of the predicate symbols is determined by a subset of the Herbrand Base. We have the following theorem that tells us that we can check the consistency of sets of clauses by considering only Herbrand interpretations: A set of closed clauses

$\mathcal{C} \subseteq L(S)$ is satisfiable if and only if it is satisfied by a Herbrand interpretation for $L(S)$.

Let us consider the class $\mathcal{I}(S)$ of interpretations for language $L(S)$. Our objects, collected in a class $\mathcal{O}$, are the subsets of $\mathcal{I}(S)$. The class H of Herbrand interpretations for $L(S)$ is a particular object in the class. H is a generic object in the following sense: For each set $\mathcal{C}$ of closed clauses in $L(S)$, we may define the following property $P_{\mathcal{C}}$ of sets of interpretations: $P_{\mathcal{C}}(A) \;:\Leftrightarrow\; \forall \mathfrak{A} \in A : \mathfrak{A} \not\models \mathcal{C}$.

The non trivial direction of the previous theorem tells us: For all $P_{\mathcal{C}}$: $P_{\mathcal{C}}(H) \Rightarrow \forall A \in \mathcal{O} : P_{\mathcal{C}}(A)$. That is, if the generic object H has any of the properties $P_{\mathcal{C}}$, then all objects in the class $\mathcal{O}$ have that property.

6.3 Horn Clauses Admit Generic Models

We consider here only propositional formulas, say, of a language $L(P)$. According to Makowsky [20], a truth valuation σ is *generic* for a set of propositional formulas Σ iff: (1) $\sigma \models \Sigma$, and (2) for each propositional variable $p \in P$: $\sigma(p) = 1 \Leftrightarrow \Sigma \models p$. This definition means that a generic model of a set of formulas makes true only those propositional variables that are forced to be true by Σ. It is extremely economical in making propositional variables true.

Let us denote by $\mathcal{O}$ the class of valuations that satisfy Σ, and define the following properties S_p $(p \in P)$ on valuations: $S_p(\sigma) \;:\Leftrightarrow\; \sigma(p) = 1$. We can easily prove the following:

Proposition 6. A valuation σ_0 is generic for Σ iff:

(a) $\sigma_0 \in \mathcal{O}$, and
(b) for all S_p: $S_p(\sigma_0) \;\Rightarrow\;$ for all σ: $(\mathcal{O}(\sigma) \Rightarrow S_p(\sigma))$. ■

Our proposition shows that Makowsky's definition captures one of the usual intuitive notions of genericity: If the generic valuation has the distinguished properties of the class to which it belongs, then all valuations in the class have those properties. We can also easily show from results in [20] that the generic model of a set of formulas can be obtained as the parallel propositional circumscription (see [15]) of all the propositional variables in the formula. As Makowsky points out, generic models give a semantics to Reiter's Closed World Assumption [23] applied to propositional knowledge bases (see also [13]). The relationship between CWA and circumscription has been explored in [16].

6.4 Data Types and Generic Functions

There is a notion of generic function that appears in relation to the concept of parametric polymorphic function in data type specifications [6]. Polymorphic functions are functions whose operands can be of more than one type. Polymorphic types may be defined as as types whose operations are applicable to operands of more than one type. An example of a polymorphic function is the length of a list of objects, which returns integer values independently of the types of the objects. This function is not only an example of a polymorphic function, but also an example of a generic function: It works for arguments of many types, generally doing the same kind of work independently of the argument's type. Other examples of generic polymorphic functions (or procedures) are provided by the usual comparison–based sorting algorithms [22]; they work by comparison of keys, without considering the type and contents of the records.

Let us consider a fixed set of data types A and the class $\mathcal{O}$ of functions that, for some type $a \in A$, and for each $x \in a$, compute the unique value y, such that $R(x, y)$ holds. $R(\cdot, \cdot)$ is a functional relation that exists, say, in a broader setting, e.g. set theory, and supports objects of different types. For example, $R([x_1, \ldots, x_n], n)$ could be the functional relation that defines the length of a list of objects $x_1, \ldots, x_n$. These objects could be integers, reals, or objects of any type. Notice that according to our definition of the class $\mathcal{O}$, there might be more than one type associated to a function in the class, in the sense that the function computes the function values for different types. Let us denote with f_a a function that computes the function for arguments of type a.

A function $g \in \mathcal{O}$ is *generic* if the following holds: $\forall\, a \in A\, \forall\, x \in a\, \forall\, y (f_a(x) = y \Leftrightarrow g(x) = y)$.

If we denote with upper-case letters the functional relations corresponding to the functions in $\mathcal{O}$, then some facts follow from the definition of generic function: (a) $\bigcup_a F_a = G$, (b) $\forall\, a\; F_a = G \cap a$, and (c) $\forall\, x, y\; ((x, y) \notin G \Rightarrow \forall\, a\; (x, y) \notin F_a)$. In particular, item (c) expresses something of the form *"if a generic function has a property, then all functions in the class have that property"*.

6.5 Most General Unifiers Are Generic

Let us consider the class $\mathcal{O}$ of unifiers for a set of expressions E as they appear in logic programming [18]. A unifier μ is a most general unifier (an mgu) in the class if any unifier λ in the class can be represented in the form $\lambda = \mu \circ \delta$, for some assignment δ. That is, we obtain any unifier in the class by starting with the mgu μ and applying to the remaining variables the assignment δ. In this sense, the mgu is a unifier that assigns exactly those ground terms to variables in E that it is forced to assign. We have the following property for the mgu: For all variables x in E and ground term t: $\mu : x \mapsto t \;\Rightarrow\; \forall\, \lambda \in \mathcal{O} :\; \lambda : x \mapsto t$,

Again, this statement follows a known pattern: *"If a mgu has the property of assigning a ground term to a variable, then all the unifiers in the class have the same property"*. We can see that the notions "(most) general" and "generic" can be sensibly seen as interchangeable.

6.6 Generic Query Languages for Databases

In database theory there is the concept of generic query language [2, 3, 12]. The idea is the following: Databases usually do not show the internal representation of the data. Furthermore, when we query or update the database, the kind of computations that are performed treat data in an abstract, generic, form. On the other side, query languages usually refer only to logical properties of the data without referring to internal data representation. Database transformations should treat values as essentially uninterpreted objects [1]. More precisely [3], a query is a mapping ϕ from finite structures that are instances of an input database schema to finite structures over an output database schema. The mapping must be computable, and generic, in the sense that for each automorphism ν of the input I, $\varphi(\nu(I)) = \nu(\phi(I))$.

Again, we find the common pattern behind this notion of genericity. Results that can be obtained by means of languages that provide generic queries, or by means of computational transformations that are generic, can be obtained by means of any other (correct) query language or computational transformation, in particular, by means of semantics– or internal representation–based query languages or transformations.

6.7 Generic Points for Negation in Logic Programming

Assume that we have a normal program in logic P [18]. It is known that there are problems in the presence of a non ground negated subgoal of the form "$\leftarrow$ *not* $g(x)$".

The first-order declarative meaning of a program clause of the form "$f(x) \leftarrow$ *not* $g(x)$", containing "*not* $g(x)$" as a subgoal, can be captured by the formula "$\forall\, x(f(x) \leftarrow \neg g(x))$". Accordingly, if our original query is "$\leftarrow f(x)$", asking if there exists some value for x that makes f true, then we should ask if there exists a value for x that makes $\neg g(x)$ true. On its turn, this leads us to ask if *for all values of* x, $g(x)$ is true. There have been some attempts to capture in logic programming this classical meaning for negated uninstantiated subgoals. Nevertheless, in current implementations of PROLOG the procedural meaning is quite different, and is closer to "negation as failure". Actually, if we ask the query ": $-$ *not* $g(X)$.", then: *not* $g(X)$ succeeds $\Leftrightarrow$ query $g(X)$ finitely fails.

Now, query $g(X)$ finitely fails if the control strategy makes possible to check exhaustively whether for all possible ground terms $\mathbf{t}$, $g(\mathbf{t})$ (finitely) fails. Then, the last double implication above can be put in equivalent terms: *not* $g(X)$ succeeds $\Leftrightarrow$ *for all* $\mathbf{t}$, $g(\mathbf{t})$ finitely fails.

That is: *not* $g(X)$ succeeds $\Leftrightarrow$ *for all* $\mathbf{t}$, *not* $g(\mathbf{t})$ succeeds. This equivalence reminds us of our notion of genericity: X seems to be a generic element with respect to the property "*not* $g(\cdot)$". This suggests that, in order to evaluate a negated non ground query ": $-$ *not* $g(X)$", we might attempt to evaluate the query" : $-$ *not* $g(\mathbf{e})$", where $\mathbf{e}$ is a generic element.

It seems that in [11], Ginsberg attempts to materialize this kind of approach to the evaluation of uninstantiated negated subgoals, by defining appropriate test elements **e**.

Acknowledgments: We are grateful to Tania Bedrax-Weiss, Alvaro Campos, Vladimir Lifschitz, Jorge Lobo, John McCarthy, and Pablo Straub for discussions, suggestions, and pointers to the literature; and to Gabriel Kuper, Rene Peralta, and Moshe Vardi for pointers to the literature and for making some papers available. The first author gratefully acknowledges the financial support of several grants (FONDECYT # 1930554 , DIUC, Fundacion DICTUC).

References

1. Aho, A.V. and Ullman, J.D. "Universality of Data Retrieval Languages". Proc. 6th ACM Symposium on Principles of Programming Languages (San Antonio, Texas), 1979, 110–120.
2. Abiteboul, S. and Vianu, V. "Procedural Languages for Database Queries and Updates", J. Computer and Systems Sciences, 41, 1990, 181–229.
3. Abiteboul, S. and Vianu, V. "Generic Computation and its Complexity", Proceedings STOC'91, ACM, 209–219.
4. Armstrong, W.W. "Dependency Structures of Database Relationships". Proc. IFIP 74, North Holland, 1974, 580–583.
5. Bertossi, L.E. and Reiter, R. "Circumscription and Generic Mathematical Objects". Working Notes of the 4th International Workshop on Nonmonotonic Reasoning, Vermont, May 1992, AT&T and AAAI. To appear in Fundamenta Informaticae.
6. Cardelli, L. and Wegner, P. "On Understanding Types, Data Abstraction, and Polymorphism". ACM Computing Surveys, 17, 4, 1985, 471–522.
7. Chang, C.C. and Keisler, J. "Model Theory". North-Holland, 1978.
8. Cohen, P.A. "Set Theory and the Continuum Hypothesis". Benjamin, 1966.
9. Etherington, D.W., Krauss, S. and Perlis, D. "Nonmonotonicity and the Scope of Reasoning". Artificial Intelligence 52, 1991, 221–261.
10. Fagin, R. "Functional Dependencies in a Relational Database and Propositional Logic". Revision of IBM Research Report RJ 1776, San Jose, California, 1976.
11. Ginsberg, M. "Negative Subgoals with Free Variables". Journal of Logic Programming, 11, 1991.
12. Hull, R. and Yap, C.K. "The Format Model: A Theory of Database Organization", J. ACM, 31, 3, 1984, 518–537.
13. Hodges, W. "Logical Features of Horn Clauses". Handbook of Logic in Artificial Intelligence and Logic Programming, vol. 1, 449–503, Oxford University Press, 1993.
14. Keisler, J.H. "Fundamentals of Model Theory". In *Handbook of Mathematical Logic*, Barwise, J. (ed.), North-Holland, 1977.
15. Lifschitz, V. "Circumscription". In 'Handbook of Logic in Artificial Intelligence and Logic Programming', Oxford University Press. To appear.
16. Lifschitz, V. "Closed-World Databases and Circumscription". Artificial Intelligence, 27, 1985, 229–235.

17. Lifschitz, V. “Computing Circumscription”. Proc. 9th IJCAI 1985, Morgan Kaufmann Publishers, 1985, 121–127.
18. Lloyd, J.W. “Foundations of Logic Programming”. Springer, 2nd edition, 1987.
19. Maier, D. “The Theory of Relational Databases”. Computer Science Press, 1983.
20. Makowsky, J.A. “Why Horn Formulas Matter in Computer Science: Initial Structures and Generic Examples”. Journal of Computer and Systems Sciences 34, 1987, 266–292.
21. McCarthy, J. “Circumscription – a form of non–monotonic reasoning”. Artificial Intelligence, 13, 1980, 27–39.
22. Mehlhorn, K. “Algorithms and Data Structures”. Springer, vol. 1, 1984.
23. Reiter, R. “On Closed World Data Bases”. In ‘Logic and Data Bases’, Gallaire, H. and Minker, J. (eds.), Plenum Press, 1978, 55–76.
24. Reiter, R. “Nonmonotonic Reasoning”. Annual Reviews in Computer Science, vol. 2, 1987.
25. Robinson, A. “On the Metamathematics of Algebra”. North Holland, 1951.
26. Robinson, A. “Théorie Métamathématique des Idéaux”. Gauthier-Villars, Paris, 1955.
27. Robinson, A. “Forcing in Model Theory”. Symposia Mathematica, vol. 5, Academic Press, New York, 1971, 69–82.
28. Robinson, A. “Introduction to Model Theory and to the Metamathematics of Algebra”. North-Holland, 1974.
29. Sagiv, Y., Delobel, C., Parker, D.S. and Fagin, R. “An Equivalence between Relational Database Dependencies and a Fragment of Propositional Logic”. Journal of the ACM, 48, 3, 1981, 435–453.
30. Semple, J.G. and Kneebone, G.T. “Algebraic Curves”. Oxford University Press, 1959.
31. Seidenberg, A. “Elements of the Theory of Algebraic Curves”. Addison-Wesley, 1968.
32. Thom, R. “Structural Stability and Morphogenesis”, W.A. Benjamin, Inc., 1975.

This article was processed using the LaTeX macro package with LLNCS style