Compilability of Abduction

Paolo Liberatore and **Marco Schaerf**
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma - Italy
Email: liberato@dis.uniroma1.it
schaerf@dis.uniroma1.it

Abstract

Abduction is one of the most important forms of reasoning and it has been successfully applied to several practical problems such as diagnosis. In this paper we investigate whether the computational complexity of abduction can be reduced by an appropriate use of preprocessing or compilation. This is motivated by the fact that part of the data of the problem (namely, the set of all possible assumptions and the theory relating assumptions and manifestations) are often known before the rest of the problem. We present a detailed analysis of the computational complexity of abduction when compilation is allowed.

Introduction

Abduction is, along with deduction and induction, one of the main reasoning styles. The first researcher to study abduction in detail has been C. S. Peirce (1955). The simplest presentation of abduction is by comparing it with deduction and induction. Deduction is the process of obtaining conclusions from facts and rules, so, for instance, from a and $a \rightarrow b$ we can conclude b by deduction. Induction is the process of establishing a general rule from the antecedents and consequences, so for instance from a and b we can conclude that $a \rightarrow b$ is true. Abduction is in some sense the inverse process of deduction, since from consequences and rules we try to establish the truth value of antecedents, so for instance given $a \rightarrow b$ and b, we conclude by abduction that a is a possible cause of b. Abduction is the process of finding the most likely explanations of a given set of manifestations, given some rules relating explanations and manifestations.

The most important application of abduction in AI is diagnosis. Indeed, the process of finding the causes of possible abnormal behaviors of systems is in fact a problem of abduction, where we want to find the causes of malfunctions given a set of rules describing the behavior of the system and a description of the observed malfunctions.

The computational complexity of abduction has already been deeply investigated in the literature. The case of hypothesis assembly has been studied by Bylander *et al.* (1989), while the first attempts to investigate the complexity of logic-based abduction has been done by Selman and Levesque (1990), and by Bylander *et al.* (1989). Eiter and Gottlob (1995) presented an extensive analysis of several problems related to abduction, including the problems of relevance and necessity.

In this paper we analyze a different computational aspect of abduction. Each instance of abduction is composed of three parts: the set of explanations, the logical theory defining the connections between explanations and manifestations, and the set of manifestations. In many applications, the set of all possible explanations and the theory are available beforehand, while the set of manifestations is only available when the solution is needed. Take as an example the diagnosis of a digital circuit: The logical theory defining the behavior of the circuit is known when the circuit is built and the set of possible explanations is simply the list of all components that can be faulty. Here we consider the problem of whether faster algorithms can be obtained by allowing the part of the instance that is known in advance to be preprocessed (compiled). We want to know if it is possible to rewrite the theory and the set of possible explanations so that we can provide faster algorithms to solve the problem when a set of manifestations is given.

Kautz, Kearns and Selman (1995) investigated the computational complexity of abduction from model-based representations. Their work is further expanded by Khardon and Roth (1996). In this paper we focus primarily on theory-based abduction, that is, we assume that the representation of the problem is given as a logical theory. In a later section we discuss the relations between our results and those presented in these two papers.

The idea of using compilation for speeding-up the solving of abduction problems is not new. For instance, Console, Portinale, and Duprè (1996) have shown how compiled knowledge can be used in the process of abductive diagnosis. The contribution of the present paper is to give a precise worst-case analysis of the problem.

Preliminaries

A problem of abduction is a triple $\langle H, M, T \rangle$, where H and M are sets of variables, while T is a theory (set of propositional formulae). The set of solutions is defined as follows:

$$SOL(H, M, T) = \{H' \subseteq H \mid H' \cup T \text{ is consistent and } H' \cup T \models M\}$$

Given an ordering $\preceq$ over the subsets of H, the set of minimal solutions of the problem is defined as follows.

$$SOL_{\preceq}(H, M, T) = \min(SOL(H, M, T), \preceq)$$

The ordering $\preceq$ captures the intuitive notion of plausibility of an explanation. That is, $H' \prec H''$ holds if H' is more likely to be the "real" cause of the manifestations than H''. The ordering $\preceq$ represents the concept of "at least as likely as", thus $H' \cong H''$ holds if H' and H'' are equally likely. The definition of $SOL_{\preceq}$ formalizes the idea of choosing only the explanations we consider more likely. Note that we assume that the ordering between two sets of assumptions does not depend on the set of manifestations.

In the sequel of the paper we assume that the ordering $\preceq$ is "well-founded". That is, if the set $SOL(H, M, T)$ is non empty, then there exists at least one minimal element (i.e., $\min(SOL(H, M, T), \preceq)$ is not empty).

In real applications we have very different forms of information on the plausibility of an explanation. The two simplest orderings are $\subseteq$-preference and $\leq$-preference. While $\subseteq$-preference selects only irredundant explanations, the $\leq$-preference selects explanations with the least possible number of assumptions.

$\subseteq$-preference $H' \preceq H''$ if and only if $H' \subseteq H''$;

$\leq$-preference $H' \preceq H''$ if and only if $|H'| \leq |H''|$;

where $|X|$ denotes the cardinality of the set X. In the sequel, we assume that the set of all possible manifestations is identical to the set of all variables. Moreover, without loss of generality, we assume that T is a 3CNF formula. The reasoning problems we consider are the following ones:

Existence: is there an explanation of the observed manifestations? That is, $SOL(H, M, T) \neq \emptyset$?

Relevance: given a variable $h \in H$, is there a minimal solution containing h? That is, $\exists H' \subseteq H$ such that $H' \in SOL_{\preceq}(H, M, T)$ and $h \in H'$?

Necessity: is $h \in H$ in all, and at least one, minimal solution? That is, $SOL(H, M, T) \neq \emptyset$ and $\forall H' \subseteq H$ we have that $H' \in SOL_{\preceq}(H, M, T)$ implies $h \in H'$?

Clearly, the ordering does not matter for the problem of existence, since we consider only well-founded orderings that guarantee that $\min(A, \preceq) \neq \emptyset$ whenever A is not empty. For all other problems, the ordering must be taken into account. Different orderings may lead to different computational properties.

Complexity and Compilability

We assume the reader is familiar with basic complexity classes, such as P, NP and the classes of the polynomial hierarchy (Stockmeyer 1976; Garey & Johnson 1979). In the sequel, C, C$'$, etc. denote arbitrary classes of the polynomial hierarchy. We assume that the input instances of problems are strings built over an alphabet Σ. The *length* of a string $x \in \Sigma^*$ is denoted by $||x||$.

We summarize some definitions and results proposed to formalize the on-line complexity of problems (Cadoli *et al.* 1996). Following the intuition that an abductive problem is composed of a part that is known in advance (T and H) and a part that is only known at run-time (M), we divide a reasoning problem into two parts: one part is *fixed* or *accessible off-line*, and the second one is *varying*, or *accessible on-line*. We present a way of formalizing the *on-line complexity*, or *compilability*, of solving a problem composed of such inputs, i.e., complexity when the first input can be preprocessed.

A function f is called *poly-size* if there exists a polynomial p such that for all strings x it holds $||f(x)|| \leq p(||x||)$. An exception to this definition is when x represents a natural number: in this case, we impose $||f(x)|| \leq p(x)$.

A function g is called *poly-time* if there exists a polynomial q such that for all x, $g(x)$ can be computed in time less than or equal to $q(||x||)$. These definitions easily extend to binary functions as usual.

We define a *language of pairs* S as a subset of $\Sigma^* \times \Sigma^*$. Using the above definitions we introduce a new hierarchy of classes of languages of pairs, the *non-uniform compilability classes*, denoted as $\|\!\rightarrow$C, where C is a generic uniform complexity class, such as P, NP, coNP, or Σ_2^p.

Definition 1 ($\|\!\rightarrow$C classes) *A language of pairs $S \subseteq \Sigma^* \times \Sigma^*$ belongs to $\|\!\rightarrow$C iff there exists a binary poly-size function f and a language of pairs $S' \in$ C such that for all $\langle x, y\rangle \in S$ it holds:*

$$\langle x, y\rangle \in S \text{ iff } \langle f(x, ||y||), y\rangle \in S'$$

A problem in $\|\!\rightarrow$C is a problem that is in C after a suitable polynomial-size preprocessing. Clearly, any problem whose time complexity is in C is also in $\|\!\rightarrow$C (just take $f(x, ||y||) = x$ and $S' = S$). Compilation is useful if a problem in C is in $\|\!\rightarrow$C$'$, where C$' \subset$ C, that is, preprocessing decreases the complexity of the problem. There are problems for which such reduction of complexity is possible (Cadoli *et al.* 1996).

Definition 2 (nucomp reductions) *A nucomp reduction from a problem A to a problem B is a triple $\langle f_1, f_2, g\rangle$, where f_1 and f_2 are poly-size functions, g is a polynomial function, and for every pair $\langle x, y\rangle$ it holds that $\langle x, y\rangle \in A$ if and only if $\langle f_1(x, ||y||), g(f_2(x, ||y||), y)\rangle \in B$.*

If there exists a nucomp reduction from A to B we say that A is nucomp reducible to B, denoted as $A \leq_{nucomp} B$.

For these classes it is possible to define the notions of *hardness* and *completeness*.

Definition 3 ($\|\!\rightarrow$C-completeness) *Let S be a language of pairs and* C *a complexity class. S is* $\|\!\rightarrow$C-hard *iff for all problems $A \in \|\!\rightarrow$C we have that $A \leq_{nucomp} S$. Moreover, S is* $\|\!\rightarrow$C-complete *if S is in $\|\!\rightarrow$C and is* $\|\!\rightarrow$C-hard.

It is important to point out that the hierarchy formed by the compilability classes is proper if and only if the polynomial hierarchy is proper (Cadoli *et al.* 1996; Karp & Lipton 1980; Yap 1983) — a fact widely conjectured to be true.

Informally, we may say that $\|\not\rightsquigarrow$NP-hard problems are "not compilable to P". Indeed, if such compilation were possible, then it would be possible to define f as the function that takes the fixed part of the problem and gives the result of compilation (ignoring the size of the input), and S' as the language representing the on-line processing. This would implies that a $\|\not\rightsquigarrow$NP-hard problem is in $\|\not\rightsquigarrow$P, and this implies the collapse of the polynomial hierarchy. In general, a problem which is $\|\not\rightsquigarrow$C-complete for a class C can be regarded as the "toughest" problem in C, even after arbitrary preprocessing of the fixed part.

Representative Equivalence

While the definitions of nucomp-reduction and $\|\not\rightsquigarrow$C-completeness are adequate to show the compilability level of a given reasoning problem, they require the definition of a new reduction that satisfies all the required properties. Now we show a technique that let us reuse, with only simple modifications, the reductions shown when the complexity result is first proved. We present conditions under which a polynomial reduction from an arbitrary problem implies the existence of a nucomp reduction.

Definition 4 (Classification Function) *A* classification function *for a problem A is a polynomial function $Class$ from instances of A to nonnegative integers, such that $Class(y) \leq ||y||$.*

Definition 5 (Representative Function) *A* representative function *for a problem A is a polynomial function $Repr$ from nonnegative integers to instances of A, such that $Class(Repr(n)) = n$, and that $||Repr(n)||$ is bounded by some polynomial in n.*

Definition 6 (Extension Function) *An* extension function *for a problem A is a polynomial function from instances of A and nonnegative integers to instances of A such that, for any y and $n \geq Class(y)$, the instance $y' = Exte(y, n)$ satisfies the following conditions:*

1. *$y \in A$ if and only if $y' \in A$;*
2. *$Class(y') = n$.*

Let, for example, A be the problem of propositional satisfiability. We can take $Class(F)$ as the number of variables in the formula F and $Repr(n)$ to be the set of all clauses of three literals over an alphabet of n variables. Finally, a possible extension function is obtained by adding tautological clauses to an instance.

Note that these functions are related to the problem A only, and do not involve the specific problem B we want to prove hard, neither the specific reduction used.

Once proved that for a given problem A it is possible to define three functions satisfying the above requirements, what is needed is a condition over the reduction from A to B that implies the hardness of the problem B. A polynomial reduction from a problem A to a problem of pairs B is a pair of polynomial functions $\langle r, h\rangle$ such that $x \in A$ if and only if $\langle r(x), h(x)\rangle \in B$. The following condition is what is needed for proving that a problem is nucomp-hard.

Definition 7 (Representative Equivalence) *Given a problem A (satisfying the above three conditions), a problem of pairs B, and a polynomial reduction $\langle r, h\rangle$ from A to B, the condition of representative equivalence holds if, for any instance y of A, it holds:*

$$\langle r(y), h(y)\rangle \in B \quad \textit{iff} \quad \langle r(Repr(Class(y)), h(y)\rangle \in B$$

It can be proved that the condition of representative equivalence implies that the problem B is $\|\not\rightsquigarrow$C-hard, if A is C-hard. Indeed, the following theorem shows that representative equivalence is a sufficient condition for proving the nucomp-hardness of a problem.

Theorem 1 *Let A be a problem for which there exists a classification, a representative and an extension function. If there exists a polynomial reduction from A to a problem of pairs B that satisfies representative equivalence, then there exists a nucomp reduction from $*A$ to B.*

We give an high-level explanation of the method we use to prove the incompilability of the considered problems of abduction. We begin by applying the method to the problem of existence of explanations, and then we sketch how it can be used for relevance and necessity.

The Method

Let us consider the problem of deciding whether there exists an explanation for a set of manifestations. It is already known that this problem is Σ_2^p-hard (Eiter & Gottlob 1995). A way for proving that this problem is also $\|\not\rightsquigarrow\Sigma_2^p$-hard is to find a polynomial reduction from a Σ_2^p-hard problem to the one under consideration, satisfying the condition of representative equivalence (of course, we have first to prove that the Σ_2^p-hard problem has a classification, representative, and extension functions).

The most easy way to do this is to consider reductions already known. For instance, Eiter and Gottlob (1995) have shown a reduction from $\exists\forall$QBF to the problem of existence of solutions. However, while $\exists\forall$QBF has the three needed functions, the reduction itself does not satisfy the condition of representative equivalence. As a result, we have to look for another reduction, either from $\exists\forall$QBF or from some other Σ_2^p-hard problem.

We are of course looking for a reduction that is as simple as possible. In general, the more similar two problems are, the easier it is to find a reduction with some given properties. The question now is: what is the Σ_2^p-hard problem that is most similar to the problem of existence of explanation? Clearly, the problem itself is the most similar one.

The theorem of representative equivalence says that, if we have a reduction from an *arbitrary* Σ_2^p-hard problem A to B, satisfying representative equivalence, then B is $\|\not\rightsquigarrow\Sigma_2^p$-hard. Nothing, however, prevent us from choosing $A = B$, if B is known to be Σ_2^p-hard. This technique can be formalized as follows:

- show that there exists a classification, representative, and extension functions for the problem B;
- show that there exists a reduction from B to B satisfying representative equivalence.

The most obvious reduction from a problem to itself is the identity. In our case, however, identity does not satisfy the condition of representative equivalence. As a result, we have to look for some other reduction.

Before showing the technical details of the reductions used, we notice that this technique allows for determining the compilability of problems for which a precise characterization of complexity is not known, since it allows to prove that a problem is $\|\!\rightsquigarrow$C-hard for any class C of the polynomial hierarchy for which B is C-hard.

In order to simplify the following proofs, we denote with $\Pi(Y)$ the set of all distinct clauses of length 3 on a given alphabet $Y = \{y_1, \ldots, y_n\}$. Since the theory T is in 3CNF by assumption, we have that $T \subseteq \Pi(V)$, where V is the set of variables appearing in T.

Existence of Solutions

The following lemma will be used to show a reduction from the problem of existence of solution to itself having the property of representative equivalence. Given H, M and $T = \{\gamma_1, \ldots, \gamma_m\}$, we define H', M', and T' as follows:

$$\begin{array}{rcl} H' & = & H \cup C \cup D \\ M' & = & M \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\} \\ T' & = & \{\neg c_i \vee \neg d_i \mid \gamma_i \in \Pi(H \cup X)\} \cup \\ & & \{c_i \rightarrow \gamma_i \mid \gamma_i \in \Pi(H \cup X)\} \end{array}$$

where X is the alphabet of T, while C and D are sets of new variables in one-to-one correspondence with the clauses (γ) in $\Pi(H \cup X)$. Note that, by definition, T is a subset of $\Pi(H \cup X)$. We now define f to be the function:

$$f(\langle H, M, T\rangle) = \langle H', M', T'\rangle$$

The following lemma relates the solutions of $\langle H, M, T\rangle$ with the solutions of $\langle H', M', T'\rangle$.

Lemma 1 *Let f be the function defined above. For any H, M, T, it holds:*

$$SOL(f(\langle H, M, T\rangle)) = \{S \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\} \mid S \in SOL(\langle H, M, T\rangle)\}$$

Proof. We divide the proof in three parts. First, we prove that any solution of $f(\langle H, M, T\rangle)$ contains exactly the literals c_i and d_i that are in M'. Then, we prove that if S' is a solution of $f(\langle H, M, T\rangle)$ then $S \backslash (C \cup D)$ is a solution of $\langle H, M, T\rangle$, and then we prove the converse.

1. We prove that $S \cap (C \cup D) = \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\}$. Let R be the set of literals $\{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\}$. Since $R \subseteq M'$, we have that $S' \cup T' \models R$. Let us consider an arbitrary variable c_i or d_i in R. The point is that T' does not contain any positive occurrence of c_i or d_i (indeed, $c_i \rightarrow \gamma_i$ is $\neg c_i \vee \gamma_i$ in CNF form). As a result, if $c_i \in R$, then S' must contain a positive occurrence of c_i. But S' is a set of variables, thus $c_i \in S'$. The same holds for any $d_i \in R$.

2. Let S' be an element of $SOL(\langle H', M', T'\rangle)$. We prove that $S = S' \backslash (C \cup D) \in SOL(\langle H, M, T\rangle)$. The point proved above shows that, for each i, S' contains either c_i or d_i, depending on whether $\gamma_i \in T$. As a result:

$$\begin{array}{rcl} S' \cup T' & \equiv & S \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\} \cup \\ & & \{\neg c_i \vee \neg d_i\} \cup \{c_i \rightarrow \gamma_i\} \\ & \equiv & S \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\} \cup T \end{array}$$

   As a result, $S \cup T$ is consistent because the above formula is. Moreover, since the above formula implies M, and the variables in $C \cup D$ appears only once, it also holds $S \cup T \models M$. As a result, S is a solution of $\langle H, M, T\rangle$.

3. Let $S \in SOL(\langle H, M, T\rangle)$. We show that $S' = S \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\}$ is a solution of $\langle H', M', T'\rangle$. This is an easy consequence of the fact that $S' \cup T'$ is equivalent to $S \cup T \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\}$ □

The aim of this lemma is to show the existence of reductions satisfying the condition of representative equivalence. Another lemma is needed.

Lemma 2 *Let c be a positive integer number, and let g_c be the following function:*

$$\begin{array}{rcl} g_c(\langle H, M, T\rangle) & = & \langle H \cup \{h_{|H|+1}, \ldots, h_c\}, M, \\ & & T \cup \{x_{r+1} \vee \neg x_{r+1}, \ldots, x_c \vee \neg x_c\}\rangle \end{array}$$

where $r = |Var(T) \backslash H|$. It holds

$$\begin{array}{rl} SOL(g_c(\langle H, M, T\rangle)) = & \{S \cup H' \mid S \in SOL(\langle H, M, T\rangle) \\ & \text{and } H' \subseteq \{h_{|H|+1}, \ldots, h_c\}\} \end{array}$$

We now define the classification, representative, and extension functions for the basic problems of abduction. First, the classification function is given by the maximum between the number of variables in H and the number of variables in T but not in H:

$$Class(\langle H, M, T\rangle) = \max(|H|, |Var(T) \backslash H|)$$

The representative instance of the class c is given by an instance with c possible assumptions, c other variables, and T composed by all possible clauses of three literals over these variables:

$$Repr(c) = \langle \{h_1, .., h_c\}, \emptyset, \Pi(\{h_1, \ldots, h_c\} \cup \{x_1, \ldots, x_c\})\rangle$$

The extension function is also easy to give. For example, we may add to T a set of tautologies with new variables.

$$\begin{array}{l} Ext(\langle H, M, T\rangle, m) = \\ \quad \langle H, M, T \cup \{x_{r+1} \vee \neg x_{r+1}, \ldots, x_m \vee \neg x_m\}\rangle \\ \quad \text{where } r = |Var(T) \backslash H| \end{array}$$

It is easy to check that these three functions are valid classification, representative, and extension functions for the problem of existence of explanation. It is also easy to prove that the same three functions are valid for the problems of relevance and necessity.

We are now able to show a reduction satisfying the condition of representative equivalence. Let i be the reduction defined as follows:

$$i(\langle H, M, T\rangle) = f(g_{Class(\langle H, M, T\rangle)}(\langle H, M, T\rangle))$$

By the above two lemmas, $i(\langle H, M, T\rangle)$ has solutions if and only if $\langle H, M, T\rangle$ has, thus this is a polynomial reduction. Moreover, the fixed part of $i(\langle H, M, T\rangle)$ depends only on the class of the instance $\langle H, M, T\rangle$. As a result, this reduction satisfies the condition of representative equivalence. The obvious consequence is the hardness of the problem of existence of solutions, which follows from the Σ_2^p-hardness of this problem (Eiter & Gottlob 1995).

Theorem 2 *The problem of establishing the existence of solution of an abductive problem is $\|\!\rightsquigarrow\Sigma_2^p$-hard.*

Relevance and Necessity

Let us now analyze is the problems of relevance and necessity. We make the following simplifying assumption: given an instance of abduction $\langle H, M, T\rangle$, where $H = \{h_1, \ldots, h_m\}$, we assume that we want to decide whether the first assumption h_1 is relevant. Clearly, the complexity of these two problems is the same, as we can always rename the variables appropriately.

By the two Lemmas above, it is easy to see that $i(\langle H, M, T\rangle)$ is also a reduction from the problem of relevance to the problem of relevance. It also satisfies representative equivalence, thus we have the compilability results for the problem of relevance.

Theorem 3 *The problem of relevance with no ordering is $\|\!\rightsquigarrow\Sigma_2^p$-hard.*

Let now consider the case in which an ordering $\preceq$ is used. The following properties on $\preceq$ are defined.

Meaningful. The ordering $\preceq$ is meaningful if, for any variable h and any pair of sets H' and H'' such that $h \notin H' \cup H''$, $H' \cup \{h\} \preceq H'' \cup \{h\}$ iff $H' \preceq H''$.

Irredundant The ordering $\preceq$ is irredundant if, for any pair of sets H' and H'', if $H' \subset H''$ then $H' \prec H''$.

It can be proved that the reduction i also works for the case of relevance and necessity, and satisfies representative equivalence, if the ordering $\preceq$ is irredundant. The basic point of the proof of incompilability of existence of solution is Lemma 1, which holds when no ordering is defined. In the case in which we consider $SOL_\preceq$ instead of SOL, it is possible to prove a similar lemma, if the given ordering is meaningful.

Lemma 3 *If $\preceq$ is a meaningful ordering, it holds:*

$$SOL_\preceq(f(\langle H, M, T\rangle)) = \{S \cup \{c_i \mid \gamma_i \in T\} \cup \{d_i \mid \gamma_i \notin T\} \mid S \in SOL_\preceq(\langle H, M, T\rangle)\}$$

It is also possible to prove the analogous of Lemma 2.

Lemma 4 *Let c be a positive integer number, and let g_c be the function of Lemma 2. If $\preceq$ is an irredundant ordering, it holds:*

$$SOL_\preceq(g_c(\langle H, M, T\rangle)) = SOL_\preceq(\langle H, M, T\rangle)$$

From the above lemmas, it follows that the reduction i also works in the case in which $\preceq$ is meaningful and irredundant, and we consider the problems of relevance and necessity.

Theorem 4 *If $\preceq$ is a meaningful irredundant ordering, then the problem of relevance is $\|\!\rightsquigarrow$C-hard for any class* C *of the polynomial hierarchy such that the problem is* C*-hard.*

A similar theorem holds for necessity. Since $\subseteq$ and $\leq$ are meaningful irredundant orderings, from their complexity we are able to find their corresponding characterization of compilability.

Corollary 5 *Relevance using $\subseteq$ is $\|\!\rightsquigarrow\Sigma_2^p$-hard, while using $\leq$ it is* nucomp$\Delta_3^p[\log n]$*-hard.*

Related Work

In this paper we have analyzed whether the complexity of abduction can be decreased by a preprocessing. All along the paper we have assumed that the input is provided as a triple $\langle H, M, T\rangle$, where T is a propositional theory. In (Kautz, Kearns, & Selman 1995), a different problem is analyzed, where the background knowledge (that in their case corresponds to a Horn theory) is not given in terms of a propositional theory, but as a set of *characteristic models*. In the paper it is shown that computing abduction from this representation only requires polynomial time (Kautz, Kearns, & Selman 1995, Theorem 13), while for the representation in terms of a propositional Horn theory the same problem is NP-complete (Selman & Levesque 1990, Theorems 2,3,4)). This result has been further strengthened by Khardon and Roth (1996) where they show a model-based representation for general propositional theories (not just Horn ones) and prove (Khardon & Roth 1996, Theorem 9.1) that abduction can be computed in polynomial time for general propositional theories (when represented by their set of characteristic models).

It seems the case that, by preprocessing the input $\langle H, M, T\rangle$ into a model-based representation the complexity decreases from NP-complete to polynomial-time. However, this is not the case. In fact, as shown by Kautz, Kearns and Selman in the Horn case (Kautz, Kearns, & Selman 1995, Theorem 4) and by Khardon and Roth for the more general case (Khardon & Roth 1996, Claim 7.2 and Theorem 7.4), there is no way to rewrite a propositional representation into a model-based one without (in the worst case) increasing the size exponentially. Our definition of preprocessing does not permit such a rewriting since it imposes that the function f of Definition 1 is poly-size. Hence, the complexity results for model-based representations do not contradict the results presented in this paper.

The result of incompilability as the ones given in this paper left open two possibilities: first, compilation may be used to generate a polynomial-sized output that reduces complexity in some cases (but now always); second, we may always want a reduction of complexity, possibility giving up the requirement of polynomiality of the result of compilation. Work in practical compilation has shown that this second option may lead to compilation algorithm with a good space performance in practical cases. The results of this paper are not in contradiction with such results, but rather complement them. Indeed, saying that $\|\!\rightsquigarrow$NP-hard problems are not compilable to P is more or less like saying that NP-hard problems are not tractable. In the case of NP-hardness, the

theoretical results show that even worst-case exponential algorithms are perfectly reasonable, and, in some sense, support the research in the direction of non-polynomial algorithms. In the same way, the incompilability results prove that an exponential output of the compilation phase is not a drawback of such an algorithm, but a necessity.

Conclusions and Future Work

In this paper we have shown that for most forms of logic-based abduction it is not possible to decrease the complexity of the most common computational tasks even if an arbitrarily long preprocess phase on the theory T and the set of hypothesis H is allowed.

We are currently extending their work in two directions. First of all, some more refined definitions of preference can be given, for instance prioritization and penalty. The use of these preference relations increase the complexity of the considered problems. Since these extensions are very relevant in practice, it make sense to study their properties w.r.t. compilability.

Another direction we are currently investigating is by restricting the theory T to be a set of Horn clauses. Indeed, since the complexity of abduction often goes down one level in the polynomial hierarchy when T is Horn, this restriction is relevant to implementation. It is thus useful to characterize the problem also from the point of view of compilation. Some preliminary results from the study of these two problems suggest that, even in these cases, the complexity of the abductive problems does not decrease when a compilation step is applied to T and H.

A practical question is whether there exist constraints on the form of the abductive problem allowing compilation to make the problem polynomial. Recent work in other areas (Cadoli *et al.* 1996) suggest that the most "natural" restrictions on logical formalisms (e.g. CNF form) do not lead to compilability.

Finally, it seems possible to give a definition of expressivity to logical-based abduction formalisms, which allows to classify different formalisms w.r.t. the set of abductive problems they are able to express. A preliminary analysis suggests that compilation (and not complexity) classes are useful to prove some results on this problem.

References

[Bylander *et al.* 1989] Bylander, T.; Allemang, D.; Tanner, M. C.; and Josephson, J. R. 1989. Some results concerning the computational complexity of abduction. In *Proc. of KR'89*, 44–54.

[Cadoli *et al.* 1996] Cadoli, M.; Donini, F. M.; Liberatore, P.; and Schaerf, M. 1996. Feasibility and unfeasibility of off-line processing. In *Proc. of ISTCS'96*, 100–109. IEEE Computer Society Press. URL = FTP://FTP.DIS.UNIROMA1.IT/PUB/AI/PAPERS/CADO-ETAL-96.PS.GZ.

[Console, Portinale, & Theseider Dupré 1996] Console, L.; Portinale, L.; and Theseider Dupré, D. 1996. Using compiled knowledge to guide and focus abductive diagnosis. *IEEE Trans. on Knowledge and Data Engineering* 8(5):690–706.

[Eiter & Gottlob 1995] Eiter, T., and Gottlob, G. 1995. The complexity of logic-based abduction. *J. of the ACM* 42(1):3–42.

[Garey & Johnson 1979] Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, Ca: W.H. Freeman and Company.

[Karp & Lipton 1980] Karp, R. M., and Lipton, R. J. 1980. Some connections between non-uniform and uniform complexity classes. In *Proc. of STOC'80*, 302–309.

[Kautz, Kearns, & Selman 1995] Kautz, H. A.; Kearns, M. J.; and Selman, B. 1995. Horn approximations of empirical data. *Artificial Intelligence* 74:129–145.

[Khardon & Roth 1996] Khardon, R., and Roth, D. 1996. Reasoning with models. *Artificial Intelligence* 87:187–213.

[Peirce 1955] Peirce, C. S. 1955. Abduction and induction. In Buchler, J., ed., *Philosophical Writings of Peirce*. Dover, New York. chapter 11.

[Selman & Levesque 1990] Selman, B., and Levesque, H. J. 1990. Abductive and default reasoning: A computational core. In *Proc. of AAAI'90*, 343–348.

[Stockmeyer 1976] Stockmeyer, L. J. 1976. The polynomial-time hierarchy. *Theor. Comp. Sci.* 3:1–22.

[Yap 1983] Yap, C. K. 1983. Some consequences of non-uniform conditions on uniform classes. *Theor. Comp. Sci.* 26:287–300.