ELSEVIER

Artificial Intelligence 88 (1996) 215-251

Artificial Intelligence

Is intractability of nonmonotonic reasoning a real drawback? *

Marco Cadoli *, Francesco M. Donini [1], Marco Schaerf [2]

Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Salaria 113, I-00198 Roma, Italy

Received June 1995; revised November 1995

Abstract

Several studies about computational complexity of nonmonotonic reasoning (NMR) showed that nonmonotonic inference is significantly harder than classical, monotonic inference. This contrasts with the general idea that NMR can be used to make knowledge representation and reasoning simpler, not harder.

In this paper we show that, to some extent, NMR fulfills the *representation* goal. In particular, we prove that nonmonotonic formalisms such as circumscription and default logic allow for a much more compact and natural representation of propositional knowledge than propositional calculus. Proofs are based on a suitable definition of a *compilable* inference problem, and on non-uniform complexity classes. Some results about intractability of circumscription and default logic can therefore be interpreted as the price one has to pay for having such an extra-compact representation.

On the other hand, intractability of inference and compactness of representation are not equivalent notions: we exhibit intractable nonmonotonic formalisms whose nonmonotonic assumptions are representable by few propositional formulae.

Finally, sometimes NMR really makes reasoning simpler. We present prototypical scenarios where closed-world reasoning and well-founded semantics account for a faster, complete and unsound approximation of classical reasoning.

Keywords: Knowledge representation; Nonmonotonic reasoning; Computational complexity; Compact representations

* This is an extended and revised version of a paper presented at AAAI-94 [7].

* Corresponding author. E-mail: cadoli@dis.uniroma1.it.

[1] E-mail: donini@dis.uniroma1.it.

[2] E-mail: schaerf@dis.uniroma1.it.

PII S0004-3702(96)00009-4

1. Introduction

1.1. Motivation

The complexity of nonmonotonic reasoning (NMR) has been extensively analyzed in recent years. Several studies showed that nonmonotonic inference in knowledge bases is significantly harder than classical, monotonic inference. As an example, while deciding if a literal follows from a propositional Horn formula can be done in linear time provided we reason in the classical semantics [11], the same task is co-NP-complete [9] if we reason under circumscription.

Although there are cases in which nonmonotonic inference has a complexity which is comparable to classical inference, the general picture shows that tractable problems may become intractable (e.g., the complexity raises from polynomial to NP-complete [23]), intractable problems may become "more" intractable (e.g., from NP-complete to Σ_2^P-complete [18]), decidable problems may become undecidable [1], and undecidable problems may become "more" undecidable (e.g., from r.e.-complete to Π_2^1-complete [39]). A survey on computational aspects of NMR appears as [10].

This aspect of NMR is acknowledged in the AI community. Brachman [3, p. 1090] writes:

> An irony of work on NMR is that, while the easy adoption and retraction of assumptions is most useful for speeding up natural everyday reasoning, most current NMR proposals drastically compound the already difficult problem of deductive reasoning. We urgently need to determine how NMR can be used to make common-sense inference faster, not slower.

In fact, if we want to make inference more efficient, we have to give up either soundness or completeness. The general idea about NMR is that it can be seen as a fast but *unsound* approximation of ordinary reasoning.

Apart from AI, NMR is sometimes used to represent knowledge in a more compact fashion. Noticeably, negation through *cut* is commonly used among PROLOG programmers for writing more compact and efficient programs [42, Chapter 11]. Moreover, closed-world reasoning [15, 16, 35, 36] allows for effective representation of implicit knowledge in relational as well as deductive databases, and has been widely used among database practitioners for many years now.

Apparently, there is a mismatch between the intuition behind NMR and the theoretical results on its computational complexity (and also, to some extent, the usage of NMR which is done in practice). The following questions naturally arise:

- Is high complexity of NMR a bug or a feature?
- Is the intuition that NMR simplifies reasoning wrong?
- Are there complexity analyses showing that NMR fits its intuitive motivation?
- Can we explain in some formal way why NMR seems to "save space"?

The goal of this paper is to give an answer to the above questions. In particular we address two topics:

- It is clear that NMR captures additional—w.r.t. to classical reasoning—information and that dealing with such information makes reasoning harder. Now suppose

we want to make the same inferences drawn by NMR "without using NMR", i.e., suppose we have a propositional knowledge base K and we want a new knowledge base K' such that, for all queries q, $K \vdash_{\mathrm{NMR}} q$ iff $K' \models q$. What would the monotonic knowledge base K' look like? Clearly, K' should contain explicitly all knowledge implicitly assumed by NMR. Intuitively, such knowledge captured by NMR is "compiled" into K'. But then, how large would K' be?

- It is advocated that NMR makes reasoning faster and unsound. Theoretical results seem to contradict this. Can we show on the contrary specific examples and frameworks which support the above idea?

1.2. Results

The results we obtain belong to the following categories:

(1) For some NMR formalisms and some classes of formulae in which inference is polynomially intractable, we are able to prove that it is not possible to "compile" formulae into polynomially-sized data structures that allow for polynomial-time reasoning, regardless of the time we spend for "compilation". Results of this kind are subject to some widely accepted conjectures on complexity classes.

(2) There are NMR formalisms in which inference is polynomially intractable, but it is possible to "compile" formulae into polynomially-sized data structures that allow for polynomial-time reasoning (clearly, the "compilation" is not done in polynomial time).

(3) There are common-sense reasoning problems—which can be formalized in ordinary propositional logic and whose solution is polynomially intractable—that can be reformulated in some nonmonotonic formalism. The "translation" has the following properties: completeness is always retained, soundness is sometimes lost, and polynomial-time algorithms for NMR can be used.

Intuitively, results of kind (1) above show that in some cases nonmonotonicity can be an extremely powerful tool for representing knowledge in a compact way. Some results about intractability of NMR can therefore be interpreted as the price one has to pay for having such an extra-compact representation. In order to prove such results, we use the notion of *non-uniform complexity class*, a topic whose importance for knowledge representation and reasoning has been recently highlighted in [24].

On the other hand, results of kind (2) show that "non-compilability" is not always related to intractability, since there are polynomially intractable formalisms that are "compilable". Finally, results of the third kind show that, to some extent, NMR formalisms have accomplished the goal posed by Brachman.

When presenting our results, we will make use of several examples, in which we imagine that a student must take some decisions concerning his/her *curriculum.*

The structure of the paper is the following: In Section 2 we recall some definitions of NMR formalisms that will be considered throughout the paper, i.e., circumscription, default reasoning, closed-world reasoning and well-founded semantics. In Section 3 we recall relevant definitions about complexity classes, and we also introduce the notion of "compilable problem"; the main formal tool we need for the new notion are non-uniform complexity classes. Sections 4 and 5 are mainly devoted to presentation of results of kind

(1) for the nonmonotonic formalisms of circumscription and default logic, respectively. In Section 6 we present results of kind (2), while results of kind (3) are presented in Section 7. In Section 8 we compare our research to related work, and we conclude the paper in Section 9.

2. Preliminaries

Throughout the paper we restrict our attention to propositional knowledge bases. We analyze circumscription, closed-world reasoning and default logic, three of the major NMR formalisms. Furthermore, we present a semantics for negation in logic programming known as well-founded semantics. Since they are widely known, in this paper we just mention the main definitions underlying them, restricted to a propositional language.

Given a propositional formula F, we call the set of distinct propositional letters (or atoms) occurring in it the *alphabet* of F. A *literal* is either a letter or its negation. A *clause* is a disjunction of literals. A clause is *positive* if all of its literals are positive, *negative* if all of its literals are negative. A *CNF* formula is a conjunction of clauses. A CNF formula is *positive* if all of its clauses are positive, *negative* if all of its clauses are negative. If each clause has no more than k literals, then we have a *k-CNF* formula. If each clause has at most one positive literal, then we have a *Horn* formula. If each clause has exactly one positive literal, then we have a *definite Horn* formula. If each clause has at most one negative literal, then we have a *dual-Horn* formula.

Interpretations and models of propositional formulae will be denoted as sets of letters (those which are mapped to **true**). If T and γ are propositional formulae such that all models of T are models of γ, i.e., γ is a logical consequence of T, we write $T \models \gamma$. Recall that, if T is a propositional formula which is either Horn, dual-Horn or 2-CNF, and γ is a clause, then there exist well-known algorithms that decide whether $T \models \gamma$ holds in polynomial time.

Following Lifschitz [29], given two models M and N of a propositional formula T and a partition $\langle P; Z\rangle$ of the alphabet of T, we write $M \leqslant_{(P;Z)} N$ if $(M \cap P) \subseteq (N \cap P)$.

A model M is called $(P; Z)$-*minimal* for a formula T if there is no model N of T such that $N \leqslant_{(P;Z)} M$ and $M \not\leqslant_{(P;Z)} N$.

The *circumscription* $CIRC(T; P; Z)$ of T minimizing the atoms in P and varying the atoms in Z denotes the set of $(P; Z)$-minimal models of T. Intuitively a propositional atom is placed in P if we are inclined to assume it false in models. In common-sense reasoning typically such atoms denote abnormality [30]. When an atom is in Z we accept that it is mapped to **true** in models, provided this helps in excluding some of the atoms in P. Through this selection of the minimal models we obtain the nonmonotonic behavior of circumscription.

When $Z = \emptyset$ the $(P; Z)$-minimal models of a formula are called just minimal.

The *closed-world assumption* $CWA(T)$ of a propositional formula T is defined in [36] as follows:

$$CWA(T) = T \cup \{\neg p \mid T \not\models p\}, \tag{1}$$

where p is a propositional letter. This rule has been refined by several authors: Minker [31] introduced the *generalized* CWA, Rajasekar, Lobo and Minker [35] the *weak generalized* CWA, Yahya and Henschen [47] the *extended generalized* CWA, Gelfond and Przymusinska [15] the *careful* CWA, Gelfond, Przymusinski and Przymusinska [16] the *extended* CWA. The notion of varying atoms has been used in the careful and in the extended CWA. Extended CWA, denoted as $ECWA(T; P; Z)$, is the most general of all the above rules and is defined as follows:

$$T \cup \{\neg K \mid \nexists B.\ (T \not\models B) \wedge (T \models K \vee B)\} \tag{2}$$

where $\langle P; Z\rangle$ is a partition of the alphabet of T in minimized/varying atoms, K is any formula not involving letters from Z, and B is a disjunction of atoms from P. The formulae K, whose negations are added to T in the above formula, are called *free for negation*. Careful CWA, denoted as $CCWA(T; P; Z)$, is defined as in formula (2), except that now K is a single atom from P.

Relationships between circumscription and closed-world reasoning were studied by many researchers—see for example Reiter [38], Lifschitz [28], and all work defining closed-world rules. In particular it has been shown that an abstract notion of minimality underlies both of them. For example, $CWA(T)$ is consistent iff T has a unique minimal model; in such a case that model is the unique model of $CWA(T)$. In the propositional case, formulae $ECWA(T; P; Z)$ and $CIRC(T; P; Z)$ are equivalent (cf. [16]).

Default logic has been defined by Reiter in [37]. In default logic knowledge about the world is divided into two parts, representing certain knowledge and defeasible rules, respectively. For propositional default logic, certain knowledge (denoted with W) is a set of propositional formulae, while defeasible rules (denoted with D) are a collection of special inference rules called *defaults*. A default is a rule of the form

$$\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma}$$

where $\alpha, \beta_1, \ldots, \beta_n, \gamma$ are propositional formulae. α is called the *prerequisite* of the default, $\beta_1, \ldots, \beta_n$ $(n \geqslant 0)$ are called *justifications* and γ is the *consequence*.

Default rules of the form $\frac{\alpha:\beta}{\beta}$ are called *normal*. A default theory is a pair $\langle D, W\rangle$, where D and W are as above. The semantics of a default theory $\langle D, W\rangle$ is based on the notion of *extension*, which is a possible state of the world according to the knowledge base. Formally, an extension is a fixpoint of the operator Γ defined as follows. $\Gamma(A)$ is the smallest set such that:

(1) $W \subseteq \Gamma(A)$;
(2) $\Gamma(A) = \{\alpha \mid \Gamma(A) \models \alpha\}$;
(3) if

$$\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma} \in D, \qquad \alpha \in \Gamma(A), \qquad \forall_{j=1}^{n} \neg\beta_j \notin A,$$

then $\gamma \in \Gamma(A)$.

A set of formulae E is an extension of $\langle D, W\rangle$ iff $E = \Gamma(E)$. Note that an extension is a deductively closed set of formulae.

Given a set of defaults D, we denote with $CONS(D)$ the set of consequences of the defaults in D, that is

$$CONS(D) = \left\{ \gamma \mid \left(\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma} \right) \in D \right\}.$$

Each extension E of $\langle D, W \rangle$ is identified by a subset of D, called the set of *generating defaults* of E, defined as:

$$GD(E, \langle D, W \rangle) = \left\{ \frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma} \in D \mid \alpha \in E \text{ and } \neg\beta_1, \ldots, \neg\beta_n \notin E \right\}.$$

The set $GD(E, \langle D, W \rangle)$ of generating defaults has the property (see [37]) that each extension E of $\langle D, W \rangle$ is the deductive closure of

$$W \cup CONS(GD(E, \langle D, W \rangle)).$$

The set of generating defaults gives a compact representation of an extension of a default theory, which by definition is a deductively closed set of formulae, hence infinite.

In this paper we are interested in two forms of inference w.r.t. a default theory. We say that a formula α is a *credulous consequence* of a default theory $\langle D, W \rangle$, denoted as $\langle D, W \rangle \vdash_{CR} \alpha$, if there exists at least one extension E of $\langle D, W \rangle$ such that $\alpha \in E$. Similarly, α is a *skeptical consequence* of a default theory $\langle D, W \rangle$, denoted as $\langle D, W \rangle \vdash_{SK} \alpha$, if for all extensions E of $\langle D, W \rangle$ we have that $\alpha \in E$.

We finally recall the *well-founded semantics* for logic programs with negation. It has been introduced by van Gelder, Ross and Schlipf [44], and it associates to each program a unique intended model, called the well-founded model.

Well-founded semantics is based on partial interpretations, rather than complete ones. Partial interpretations are sets of literals that do not contain both a literal and its negation.

Let P be a propositional logic program with negation in the body, let L be the alphabet of P, and let I be a partial interpretation for P. We say $A \subseteq L$ is an *unfounded set* of P with respect to I if for each atom $p \in A$, and for each rule r of P whose head is p, either some subgoal q of the body is false in I or some positive subgoal of the body occurs in A.

Three transformations T_P, U_P and W_P of partial interpretations are defined as follows:

- $p \in T_P(I)$ if and only if there is a rule $r \in P$ such that r has head p and each subgoal literal in the body of r is true in I (this is the well-known immediate consequence operator for logic programs);
- $U_P(I)$, called the greatest unfounded set of P w.r.t. I, is the union of all sets that are unfounded w.r.t. I;
- $W_P(I) = T_P(I) \cup \neg U_P(I)$, where $\neg U_P(I)$ is the set of literals obtained by negating each atom in $U_P(I)$.

The well-founded model of P is defined by the following construction: $I_0 = \emptyset$, $I_{k+1} = W_P(I_k)$. The operator W_P is proved to be monotone, hence there is a unique least fixpoint M satisfying $M = W_P(M)$. Such M is the well-founded model of P, denoted as $WF(P)$.

As for computational complexity, van Gelder, Ross and Schlipf show in [44] that the well-founded model can be computed in time polynomial in the size of P.

3. Complexity and compilability

We refer to the standard notation on complexity classes that can be found in [14, 20, 34]. The size of the representation of a generic object x will be denoted as $|x|$. We deal with the class P (decision problems whose solution can be found in polynomial time by a deterministic Turing machine), with the class NP (same as before, but the Turing machine is non-deterministic) and the class co-NP (which is the set of problems complementary to those in NP). Following the mainstream in complexity theory, we assume that P $\subset$ NP and P $\subset$ co-NP. Therefore, we call a problem which is either NP-hard or co-NP-hard *polynomially intractable* (we remind that a problem P is [co-]NP-hard if each instance of a generic problem R in [co-]NP can be reduced to an instance of P by means of a polynomial-time transformation).

Sometimes we use the symbol of a complexity class for denoting a particular Turing machine. As an example, a P-machine is a deterministic Turing machine whose computing time is bounded by a polynomial p.

Apart from such well-known complexity classes, in this paper we will be concerned with a specific computational notion: that of *compilability* of a polynomially intractable problem. This notion deals with how much efficiency we can gain by doing *off-line reasoning* when trying to solve a computationally difficult problem. Let's consider an example: if our reasoning task is to solve a single instance of an NP-complete problem, and we know the instance a significant amount of time before we need the solution, then we could solve such a problem off-line, save the solution, and use it when it is necessary. A situation of this kind may arise if we have a description of a complex real-world system (e.g., a large instance of a propositional formula) and we want to know if such a description is consistent (i.e., to solve the satisfiability problem) but we don't need the solution immediately. We call a problem of this kind *compilable*.

This situation generalizes when there are few instances of a polynomially intractable problem that we would like to solve. The easy way to proceed is to solve all of them and cache the solutions in a table that records each pair $\langle$instance, solution$\rangle$. In principle, this strategy could be applied even when the number of instances is large, if there is a "compact" way of representing all pairs. Our only concern is that, for each instance I, we must be able to extract (on-line) the solution to I in time which is polynomial in the size of I. Moreover the size of the table must be polynomial in the size of the original problem (otherwise it would be unfeasible to store it in a physical system).

We give in what follows a precise characterization of compilability of a problem. Our intuition tells us that a polynomially intractable problem could be said to be compilable if there is a data structure that grants that all interesting instances of the problem can be solved on-line in polynomial time in the size of the instance itself. Clearly such a data structure must be of polynomial size. In general, we are not interested in how much time it takes to build the data structure, as we are interested only in efficiency of on-line reasoning.

Compilability gains relevance when the instance of a problem comes in two parts: a part which is *fixed* or *off-line* and a part which is *varying* or *on-line*. This can very well be the case for knowledge representation. Recalling the functional approach to

knowledge representation [27], we may suppose we have a knowledge base defined by a sequence of TELL operations; queries to the knowledge base are then posed by ASK operations. Whenever the knowledge base reaches a reasonably stable state, we may want to compile in some way the information contained in it, if this speeds up the process of answering queries. In such a situation, our model of computation is a Turing machine whose input is just the varying (on-line) part.

As an example, in the problem of checking validity of $T \models \alpha$, we might assume that the fixed part is the propositional formula T representing a knowledge base, and that the varying part is the propositional formula α representing a query.

To make explicit fixed and varying part of a problem, we propose the following notation:

Notation (*Problems with fixed and varying part*). We denote a problem P with a fixed part F and varying part V as $[P, F, V]$.

This notation is equivalent to defining a problem P as a subset of the cross-product $F \times V$, where F is the set of instances of the fixed part, and V is the set of instances of the varying part. With abuse of notation, continuing the above example, the logical entailment checking problem $T \models \alpha$, where T is fixed and α is varying, can be denoted as $[T \models \alpha, T, \alpha]$.

Informally, a problem $[P, F, V]$ is compilable if, for each instance f of the fixed part F there is a data structure D_f of size polynomial in $|f|$ such that D_f can be used to solve the problem P in polynomial time.

As an example, the problem $[T \models l, T, l]$, T being a CNF formula and l being a single literal is co-NP-complete, but is definitely compilable. As a matter of fact, the data structure we need is just a table that records, for each literal l occurring in T, whether $T \models l$ or not. If n is the cardinality of the alphabet of T ($n \in O(|T|)$) the size of the table is in $O(n)$, and it can be consulted in $O(n)$ time. Building the table amounts to solving $O(n)$ instances of a co-NP-complete problem; but, as we already mentioned, we are not concerned with the off-line computational burden.

The reason why $[T \models l, T, l]$ is compilable is that the space of possible queries has size polynomial in $|T|$. A problem might be compilable even if this is not true. Let's consider the problem $[T \models conj, T, conj]$, where *conj* is a conjunction of literals. The number of possible conjunctions is $\Omega(2^n)$, but the table mentioned before still suffices for a polynomial-time on-line answer, as for each set $\{l_1, \ldots, l_m\}$ of literals, $T \models l_1 \wedge \cdots \wedge l_m$ iff $T \models l_1, \ldots, T \models l_m$.

We are now ready to give the exact definition of compilability of a problem.

Definition 1 (*Compilable problem*). A problem $[P, F, V]$ is *compilable* if there exist two polynomials p_1, p_2 and an algorithm *ASK* such that for each instance f of F there is a data structure D_f such that:

(1) $|D_f| \leqslant p_1(|f|)$;

(2) for each instance v of V the call $ASK(D_f, v)$ returns yes iff $\langle f, v \rangle$ is a "yes" instance of P;

(3) $ASK(D_f, v)$ requires time $\leqslant p_2(|v| + |D_f|)$.

The definition of compilability is non-constructive, because it does not require a recursive way of building the data structure D_f. Nevertheless, in this paper every time we prove that a problem is compilable we define a data structure in a constructive way. Note that a problem is compilable if and only if its complement is compilable, i.e., the class of compilable problems is closed under complementation.

So far, we provided examples of compilable problems. The definition of non-compilable problem follows from Definition 1: $[P, F, V]$ is non-compilable if there are no polynomials p_1, p_2 with the above properties. As already mentioned, a necessary (but not sufficient) condition for non-compilability is that the set of possible varying parts of the input V is super-polynomial in the size $|f|$ of the fixed part. Intuitively, if a polynomial p_1 with the properties mentioned in Definition 1 does not exist, it means that compilation would take too much (i.e., super-polynomial) space. Dually, if p_1 exists and p_2 does not, then encoding of the problem in a polynomial-space data structure is possible, but it is not possible to extract information from it in polynomial time. In this work we don't prove any problem to be not compilable in this sense. The best we are able to do is to prove non-compilability provided some widely accepted conjectures on complexity classes are true. For clarifying exactly what we do, we need a digression into *Turing machines with advice*. We first report some definitions from [20].

Definition 2 (*Advice-taking Turing machine*). An *advice-taking Turing machine* is a Turing machine that has associated with it a special "advice oracle" A, which can be any function, not necessarily a recursive one. On input x, a special "advice tape" is automatically loaded with $A(|x|)$ and from then on the computation proceeds as normal, based on the two inputs, x and $A(|x|)$.

Definition 3 (*Polynomial advice*). An advice-taking Turing machine uses *polynomial advice* if its advice oracle A satisfies $|A(n)| \leqslant p(n)$ for some fixed polynomial p and all non-negative integers n.

Definition 4 (*Non-uniform complexity classes*). If C is a class of languages defined in terms of resource-bounded Turing machines, then C/poly is the class of languages defined by Turing machines with the same resource bounds but augmented by polynomial advice. Any class C/poly is also known as *non-uniform* C.

Non-uniformity is due to the presence of the advice. Note that the advice is only a function of the size of the input, not of the input itself. Moreover we don't have to care about building the advice oracle: we are free to choose it in the way we like, as long as the condition on the size of the advice is respected. This means, for example, that we can choose the oracle to provide the solution even for some undecidable problems.

Throughout the paper, we will be interested in the class P/poly. Intuitively, a P/poly-machine has some "extra mileage" w.r.t. a P-machine since it can use the advice for free. An appropriate question is therefore what is exactly the "extra mileage" gained by the machine. As an example, it is interesting to compare P/poly to uniform complexity classes such as NP and co-NP. For the sake of the argument, let's assume that co-NP $\subseteq$

P/poly. This would mean that each co-NP-complete problem such as deciding whether $T \models q$, where T is a CNF formula and q is a clause, can be solved by a P/poly-machine $\mathcal{M}$. This would imply that the problem $[T \models q, T, q]$ is compilable, as proved by the following argument (cf. Definition 1): polynomial p_1 is the one bounding the size of the advice in Definition 3, while polynomial p_2 is the one that characterizes the time spent in the computation by $\mathcal{M}$. Remember that we are free to choose any advice oracle we like; if co-NP $\subseteq$ P/poly, we can choose one that is an expert in propositional calculus, and that, for each CNF formula T of size n, provides an advice which is a data structure D_n depending only on n and of size $\leqslant p_1(n)$. Such a data structure can be used by $\mathcal{M}$ along with q for proving whether $T \models q$ holds or not in time $\leqslant p_2(n + p_1(n))$ (this is the algorithm *ASK* alluded to in Definition 1).

The above argument shows that co-NP $\subseteq$ P/poly implies $[T \models q, T, q]$ is compilable. It is possible to prove the other way of the implication, i.e., that if $[T \models q, T, q]$ is compilable then co-NP $\subseteq$ P/poly. This will be done at the end of the present section (Theorem 6).

What we have seen shows that the concept of compilability of a problem goes hand in hand with the notion of non-uniform complexity classes. It is therefore important to know what researchers in theoretical computer science have discovered about such classes. Relations between non-uniform and uniform complexity classes were studied in the literature, cf. e.g., [21,48]. In particular, it is proved that if co-NP $\subseteq$ P/poly, then the collapse of the so-called *polynomial hierarchy* happens. More formally, if co-NP $\subseteq$ P/poly then $\Sigma_2^p = \Pi_2^p$. Collapse of the polynomial hierarchy is widely conjectured to be false in the literature (among other things, $\Sigma_2^p \neq \Pi_2^p$ implies P $\neq$ NP). We are therefore entitled to say that $[T \models q, T, q]$ is likely not to be compilable.

As a matter of fact, in the present paper we go as far as proving that some problems concerning NMR are not compilable provided co-NP $\not\subseteq$ P/poly, or, since P/poly is closed under complement, NP $\not\subseteq$ P/poly. All our proofs make use of a general property, that we state in the next theorem. We assume that a decision problem Π, seen as the infinite set of its instances, can be partitioned according to the size of the instances, for a reasonable encoding (in the sense of [20]) of the instances.

Theorem 5. *Let Π be an* NP*-complete problem, and let $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, where $\Pi_n = \{\pi \in \Pi$ such that $|\pi| = n\}$. Moreover, let $[P, F, V]$ be a problem we want to compile, divided into fixed part F and varying part V. Suppose that there exists a polynomial p such that, for each $n > 0$, there exists an $f_n \in F$ with the following properties*:

(1) $|f_n| \leqslant p(n)$;

(2) *for all $\pi \in \Pi_n$, there exists a $v_\pi \in V$ such that*:

(a) *v_π can be computed from π in polynomial time*;

(b) *$\langle f_n, v_\pi \rangle$ is a "yes" instance of P iff π is a "yes" instance of Π.*

With the above hypothesis, if $[P, F, V]$ is compilable, then NP $\subseteq$ P/poly.

Proof. Suppose that there exists a polynomial p such that, for each n, there exists an $f_n \in F$ with the properties stated. Now suppose that $[P, F, V]$ is compilable. Hence, for each f_n there exists a data structure D_{f_n} with properties stated in Definition 1. Moreover, there is a polynomial-time algorithm *ASK* such that $ASK(D_{f_n}, v)$ returns yes iff $\langle f, v \rangle$

is a "yes" instance of P. Then we can define an advice-taking Turing machine in the following way.

First, define the advice oracle as $A(n) = D_{f_n}$. Observe that $|A(n)| = |D_{f_n}| \leqslant p_1(|f_n|) \leqslant p_1(p(n))$, where p is the polynomial mentioned in the hypotheses of the above theorem, and p_1 is the polynomial mentioned in Definition 1. Hence, the size of the advice is bounded by a polynomial which is the composition of p_1 and p.

Secondly, the machine operates as follows: given an instance π of Π, with $|\pi| = n$, the machine loads $A(|\pi|) = A(n) = D_{f_n}$, then computes v_π from π in time polynomial w.r.t. $|\pi|$, then decides whether $\langle D_{f_n}, v_\pi \rangle$ is a "yes" instance of P by calling $ASK(D_{f_n}, v_\pi)$. If $[P, F, V]$ is compilable, the last decision can be made in time bounded polynomially by $p_2(|D_{f_n}| + |v_\pi|)$, where p_2 is the other polynomial mentioned in Definition 1. Therefore, the advice-taking Turing machine would globally work in time polynomial in $|\pi|$. Since Π is an NP-complete problem, this would imply NP $\subseteq$ P/poly. □

Theorem 5 provides a general schema for the non-compilability proofs we present in the paper. All our non-compilability proofs use the same technique, even though with major differences. Most of the times, we choose as the NP-complete problem Π the problem 3SAT [14, Problem LO2], and partition the set of instances as $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, according to their size n. For each n, we know that the number of distinct propositional letters occurring in each instance of Π_n is at most n, since there cannot be more than n literals in an instance of 3SAT of size n. Without loss of generality, we assume that all formulae of Π_n are built on the same set of atoms $L = \{a_1, \ldots, a_n\}$. Observe that for each n, Π_n has the following properties:

(1) Each instance $\pi \in \Pi_n$ is a subset of a maximum instance $\pi_{\max(n)} \in \Pi$. $\pi_{\max(n)}$ is defined as the set of all the three-literal clauses on L. Clearly, all instances of 3SAT of size n are subsets of $\pi_{\max(n)}$.

(2) $\pi_{\max(n)}$ has size polynomial in n; in particular, it has size $\Theta(n^3)$. Observe that $\pi_{\max(n)} \notin \Pi_n$.

In our proofs we use the following conventions. We denote with a, b and c propositional atoms, with w, x and y literals (i.e., either atoms or negated atoms), with $\overline{L}$ a new set of literals, one-to-one with L, defined as

$$\overline{L} = \{\overline{a} \mid a \in L\}. \tag{3}$$

We exploit the fact that each three-literal clause over L can be given a name: we denote with C a set of atoms one-to-one with possible three-literal clauses of L

$$C = \{c_i \mid \gamma_i \text{ is a three-literal clause of } L\}. \tag{4}$$

Sometimes we use the variant NOT-ALL-EQUAL-3SAT [14, Problem LO3] of 3SAT: given a set of 3-CNF clauses, where each clause contains only positive literals, does there exist an assignment to the atoms such that in each clause at least one atom is mapped to **true** and at least one is mapped to **false**? In these cases, we denote with G a set of atoms one-to-one with possible three-atoms clauses (i.e., positive clauses) of L

$$G = \{g_i \mid \gamma_i \text{ is a three-atoms clause of } L\}. \tag{5}$$

We conclude the section showing a result that we anticipated before. In this way we demonstrate how the previously defined formal machinery works.

Theorem 6. *If the problem* $[T \models q, T, q]$, *where* T *is a propositional formula in CNF and* q *is a clause, is compilable, then* NP $\subseteq$ P/poly.

Proof. By Theorem 5 we only need to choose an appropriate NP-complete problem $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, a CNF formula T_n of size polynomial in n and show that for all $\pi \in \Pi_n$ there exists a clause q_π, computable from π in polynomial time such that $T_n \models q_\pi$ iff π is a "yes" instance of Π. We choose the NP-complete problem 3SAT as our Π.

Let L be the alphabet of n atoms used in the 3-CNF formulae of size n, i.e., in Π_n. We define L' as the new alphabet $L \cup C$, where C is a set of atoms one-to-one with possible three-literal clauses of L, as defined in formula (4).

We define T_n on the alphabet L' as follows: for each clause $\gamma_i \in \pi_{\max(n)}$, T_n contains the clause $c_i \vee \gamma_i$, where $c_i \in C$.

Note that the size of T_n is $O(n^3)$, and T_n is a 4-CNF formula. This completes the definition of T_n.

Let π be a generic 3-CNF formula over L, and let $c_1, \ldots, c_h$ be all the atoms of C corresponding to the clauses in π. Define $q_\pi = c_1 \vee \cdots \vee c_h$. Note that q_π is computed in time polynomial in $|\pi|$.

We now show that $T_n \not\models q_\pi$ iff π is satisfiable.

($\Leftarrow$) Assume that π is satisfiable and let $M \subseteq L$ be a model of π. An interpretation I of C is built as follows: for each $c \in C$, if c corresponds to a clause in π, then c is mapped to **false**, otherwise it is mapped to **true**. Interpretation $M \cup I$ is a model of T_n, but is not a model of q_π.

($\Rightarrow$) Assume $T_n \not\models q_\pi$, then there is a model M of T_n which maps all atoms of q_π into **false**. As a consequence, $M \models \gamma_i$ for each clause γ_i in π, i.e., the restriction of M to atoms of L is a model of π. □

The technique used in the above proof has been proposed by Kautz and Selman in the context of Horn least upper bounds [24] (cf. Section 8).

4. NMR for compact representation of knowledge I: circumscription

In the previous section we introduced the notion of compilability of a reasoning problem. In this section, and in the following one, we use such a notion to prove some computational properties of NMR. In particular, we show that NMR formalisms can be used to represent information in a compact way, where compactness is measured w.r.t. the classical representation of the same knowledge.

We introduce an example dealing with a student who needs to plan his/her *curriculum*. First of all, we show that the natural way the student can do this is to reason using circumscription. Given that the reasoning problem the student is faced with is computationally intractable, we address the following question: if the student was to reason in a classical fashion, how much implicit information would he/she need to represent ex-

plicitly in the new knowledge base? In other words, is it feasible to compile the original knowledge base—dealt with NMR—into a new one—dealt with a classical inference engine—such that the same inferences are possible in polynomial time? The results we prove show that, in general, this is not feasible.

Example 7 (*The lazy student*). The faculties impose constraints on the possible *curricula*: the set of admissible *curricula* is represented by means of the models of a propositional formula T which might look like the following:

$$\texttt{DataBases} \vee \texttt{Algebra},$$

$$\texttt{Algebra} \vee \texttt{NonMonotonicReasoning},$$

$$\texttt{DataBases} \vee \texttt{ReasoningAboutKnowledge},$$

$$\texttt{Algebra} \vee \texttt{ReasoningAboutKnowledge},$$

$$\texttt{Algebra} \vee \texttt{ComputationalComplexity}.$$

Throughout this section we assume that such formulae are always in 2-CNF. A *curriculum* is just a model of T, e.g., {DataBases, Algebra, NonMonotonicReasoning}. Such a definition does not capture the preferences that a student might have. As an example, the student might be better off by doing both courses {ReasoningAboutKnowledge, ComputationalComplexity} than doing just one of the courses {DataBases, Algebra, NonMonotonicReasoning}. Moreover, the student may prefer to take as few courses as possible.

This suggests to use the idea of $(P; Z)$-minimal models, where all exams that the student dislikes are in P; let's say that

$$P = \left\{ \begin{array}{l} \texttt{DataBases}, \\ \texttt{Algebra}, \\ \texttt{NonMonotonicReasoning} \end{array} \right\}, \quad Z = \left\{ \begin{array}{l} \texttt{ReasoningAboutKnowledge}, \\ \texttt{ComputationalComplexity} \end{array} \right\}.$$

A *minimal curriculum* is a $(P; Z)$-minimal model of T. In the above situation there are three minimal *curricula*:

$M1$ {Algebra, ReasoningAboutKnowledge},

$M2$ {Algebra, ReasoningAboutKnowledge, ComputationalComplexity},

$M3$ {DataBases, NonMonotonicReasoning, ReasoningAboutKnowledge, ComputationalComplexity}.

We say that a course c is *mandatory* if it has to be done in all *curricula*, i.e., $T \models c$. A course c is *preferred* if it has to be done in all minimal *curricula*, i.e., $CIRC(T; P; Z) \models c$. Clearly, a mandatory course is always preferred, but not vice versa.

In the above situation there are no mandatory courses, although ReasoningAboutKnowledge is preferred.

The above example suggests that there are at least four possible computational services that a student might ask:

Problem 1. Find a *curriculum*. The student has no specific preferences, a *curriculum* is just as good as any other one.

Problem 2. Find a minimal *curriculum*. This is an improvement w.r.t. the previous service, as the student can express some preferences. Nevertheless, there is no explanation why a *curriculum* is provided. In Example 1, $M2$ is as good as $M1$. Even if $M1$ is provided as an answer, the student does not know whether `ReasoningAboutKnowledge` is preferred or not.

Problem 3. Decide whether a course is mandatory. The student—with no preferences—wants to know whether he/she must attend a specific course. The shortcoming of this service is in that it does not give any information on non-mandatory courses that are false only in *curricula* that the student would never accept because of his/her preferences.

Problem 4. Decide whether a course is preferred (preferences decided in advance). Same as above, but the student has preferences.

This sophisticated form of reasoning is mostly relevant in this situation: the lazy student might take course c, but he/she does not want to commit until he/she is completely convinced that c is preferred. In fact not all courses suggested by an answer to Problem 2 are preferred.

The complexity of the above problems has been already studied in the literature. When T is 2-CNF we have the following figures:

- Problem 1 is polynomial [13].
- Problem 2 is polynomial [4].
- Problem 3 is polynomial [13].
- Problem 4 is co-NP-complete [9].

In the above scenario NMR seems to do exactly the form of reasoning the student needs. In fact a solution to Problem 4 gives some extra information that solutions to the other three problems miss, therefore we may be willing to accept its extra complexity. Furthermore, this extra complexity can be dealt with by compilation, as outlined in the beginning of Section 3: since the set of courses the student could ever take is limited—let's say they are $c_1, \ldots, c_n$—he/she might compute off-line which of them is preferred, i.e., decide whether $CIRC(T; P; Z) \models c_i$ holds for each i $(1 \leqslant i \leqslant n)$. Even if this amounts to solve n instances of a co-NP-complete problem, the queries can be posed off-line and their answers cached. Then on-line query answering just amounts to table look-up, which is clearly polynomial.

The next step is now to consider the scenario where *several* students are interested in preparing their *curricula*. Clearly this is not an additional difficulty if all of them have the same preferences. What happens if each student has his/her own preferences? The complexity of answering a single query to Problem 4 is still co-NP-complete, but there is an exponential number of possible queries, as the possible choices for P are 2^n. Is it still possible to do off-line reasoning for making on-line reasoning efficiently?

We can think of automatically solving Problem 4, if several lazy students want to decide whether a course is preferred. In this case, students may or may not have the same preferences.

Problem 4.1. Decide whether a course is preferred, when all students have the same preferences.

Problem 4.2. Decide whether a course is preferred, when each student has his/her own preferences.

Using the terminology and notations introduced in Section 3, Problem 4.1 amounts to the problem $[CIRC(T;P;Z) \models l,(T,P,Z),l]$, while Problem 4.2 corresponds to $[CIRC(T;P;Z) \models l,T,(P,Z,l)]$ (note that P, Z moved from the fixed to the varying part in the second problem).

The compilation argument for Problem 4 proves the following property of Problem 4.1.

Proposition 8. *The problem* $[CIRC(T;P;Z) \models l,(T,P,Z),l]$, *where* T *is* 2*-CNF and* l *is a single literal, is compilable.*

Compilation for Problem 4.1 can be done simply by caching. Even if such caching cannot be done for Problem 4.2 (there are exponentially many different preferences) one may wonder if the problem is compilable in some smarter way.

However, we are able to show that it is very unlikely that such a compilation may exist. For this first "non-compilability" result in the context of NMR we provide a full proof in the text. Proofs of the other theorems appear in the Appendix.

Theorem 9. *If the problem* $[CIRC(T;P;Z) \models l,T,(P,Z,l)]$, *where* T *is a positive* 2*-CNF and* l *is a single literal, is compilable, then* NP $\subseteq$ P/poly.

Proof. We apply Theorem 5, so we need to choose an appropriate NP-complete problem $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$ and for each n we define a positive 2-CNF formula T_n of size polynomial in n. Thus we show that for all $\pi \in \Pi_n$ there exists a partition $\langle P_\pi; Z_\pi \rangle$ of the alphabet of T_n and a literal l_π, all computable from π in polynomial time, such that $CIRC(T_n;P_\pi;Z_\pi) \models l_\pi$ iff π is a "yes" instance of Π.

We choose the NP-complete problem 3SAT as our Π. If a is an atom, $\overline{a}$ is its negation and if x is the literal $\overline{a}$, then $\overline{x}$ is a. Our proof is based on a reduction given in [9, Theorem 16], which showed that circumscriptive inference of literals in positive 2-CNF formulae is co-NP-hard.

Let L be the alphabet of n atoms used in the 3-CNF formulae of size n, i.e., in Π_n. We define L' as the new alphabet $L \cup \overline{L} \cup C \cup \{z\}$, where atoms of $\overline{L}$ are as in formula (3), C is a set of atoms one-to-one with possible three-literal clauses of L, as defined in formula (4) and z is a new atom. Observe that now $\overline{a}$ denotes both a negated atom of L and an atom of L'. Since formulae over L' will never use negated atoms, we can retain this ambiguity to simplify the notation in the reduction.

We define T_n on the alphabet L' according to the following rules:

(1) For each letter a of L, there is a clause $a \vee \overline{a}$ in T_n.

(2) For each clause $\gamma_i = w \vee x \vee y$ in $\pi_{\max(n)}$, where w, x and y are literals, there are four clauses in T_n. The first three are $c_i \vee w$, $c_i \vee x$, $c_i \vee y$; observe that if x is a negated literal in π, then x is an atom in T_n. As an example, let $\gamma_i = s \vee \neg t \vee u$, T_n contains $c_i \vee s$, $c_i \vee \overline{t}$, $c_i \vee u$. The fourth clause is $c_i \vee z$.

Note that the size of T_n is $O(n^3)$, and T_n is a positive 2-CNF formula. This completes the definition of T_n.

Let π be a generic 3-CNF formula over L, and let $c_1, \ldots, c_h$ be all the atoms of L' corresponding to the clauses in π. Define $P_\pi = \{c_1, \ldots, c_h\} \cup L \cup \overline{L}$, $Z_\pi = L' - P_\pi$ and $l_\pi = z$. Note that P_π, Z_π and l_π are computed in time polynomial in $|\pi|$ and l_π is the same for all instances π.

We now show that $CIRC(T_n; P_\pi; Z_\pi) \not\models l_\pi$ iff π is satisfiable.

($\Leftarrow$) Assume that π is satisfiable and let $M \subseteq L$ be a model for π and $\overline{M}$ be a subset of $\overline{L}$ such that $\overline{a} \in \overline{M}$ iff $a \in M$. Define MM as $(L - M) \cup \overline{M} \cup C$. It can be verified that MM is a model of T_n, because it includes C, and for every atom $a \in L$ it contains either a or $\overline{a}$, but not both. Moreover, l_π is false in MM.

To prove that MM is $(P_\pi; Z_\pi)$-minimal for T_n, we consider separately atoms in $L \cup \overline{L}$, and atoms in $\{c_1, \ldots, c_h\}$. No atom x from $L \cup \overline{L}$ can be eliminated from MM without introducing the corresponding atom $\overline{x}$, since in T_n there is the clause $x \vee \overline{x}$. Now consider the elimination of (say) c_i, and let $w \vee x \vee y$ be the ith clause of π. Since M is a model of π, M satisfies at least one literal, say, w. By construction of MM, $w \notin MM$ because if w is positive, then $w \in M$, and therefore $w \notin (L - M)$; if w is negative, then $w \notin \overline{M}$. Since in T_n there is the clause $c_i \vee w$, the atom c_i cannot be eliminated without introducing w. Therefore, MM is minimal. Since $l_\pi \notin MM$, it follows that $CIRC(T_n; P_\pi; Z_\pi) \not\models l_\pi$.

($\Rightarrow$) Assume $CIRC(T_n; P_\pi; Z_\pi) \not\models l_\pi$ and let MM be a $(P_\pi; Z_\pi)$-minimal model of T_n, such that l_π is false in MM. Since for each $c \in C$, the clause $c \vee l_\pi$ is in T_n, $C \subseteq MM$. Consider an atom $c_i \in P_\pi$ and let $\gamma_i = s \vee \neg t \vee u$. There are in T_n the three clauses $c_i \vee s$, $c_i \vee \overline{t}$, $c_i \vee u$.

We prove that at least one of s, $\overline{t}$, u is not in MM. By contradiction: if s, $\overline{t}$, u were in MM, then c_i could be eliminated from MM (introducing l_π to satisfy the fourth clause $c_i \vee l_\pi$), obtaining a model MM' with fewer atoms of P_π, and MM would not be minimal, contradicting the hypothesis.

Now we prove that $a \in MM$ iff $\overline{a} \notin MM$. First, because in T_n there is the clause $a \vee \overline{a}$, at least one of a and $\overline{a}$ is in MM. If both were in MM, either a or $\overline{a}$ could be eliminated, and the resulting model $MM - \{a\}$ would still satisfy T_n.

Define $M = \{a \in L \mid \overline{a} \in MM\}$. M is a model of π because it satisfies all the clauses γ_i. In fact, at least one of s, $\overline{t}$, u is not in MM, therefore at least one of $\overline{s}$, t, $\overline{u}$ is in MM. As a consequence, either one of s, u is in M or t is not in M, thus M satisfies γ_i. We conclude that π is satisfiable. □

So far we considered atomic queries. Non-atomic clauses are nevertheless necessary for posing more complex queries such as disjunction (e.g., is it necessary to take at least one course in a given set?) or implication (e.g., is it necessary to take course a provided course b is taken?). Theorem 9 implies that Problem 4.2 is not compilable also

Table 1
Are 2-CNF knowledge bases under circumscription compilable?

	All queries with same preferences	Different preferences
"Short" queries	compilable (Proposition 8)	non-compilable (Theorem 9)
"Long" queries	non-compilable (Theorem 10)	non-compilable (Theorems 9, 10)

for non-atomic queries. Instead, since in its proof it is crucial that P and Z are given as part of the input, one could think that when P and Z are fixed—like in Problem 4.1—compilation of all non-atomic queries is still possible. The next theorem shows that this conjecture is very unlikely to hold.

Theorem 10. *If the problem* $[CIRC(T;P;Z) \models q,(T,P,Z),q]$, *where* T *is 2-CNF and* q *is a clause, is compilable, then* NP $\subseteq$ P/poly.

At a first sight, the above two theorems seem to give just a negative result: NMR is not compilable. But the result is in fact twofold: the theorems show that if one wants to represent the circumscription of a 2-CNF knowledge base K with a new knowledge base K', either inference in K' is intractable, or (if inference has to be kept tractable) K' has super-polynomial size w.r.t. K, in the worst case. As a consequence, we know that circumscription allows one to derive a super-polynomial number of new consequences which are not derivable from K with classical inference: if the number of new consequences were polynomial, they could be simply cached. It might be possible that such consequences could be compacted in one formula of polynomial size; but in this case, Theorems 9 and 10 show that extracting consequences from such a formula is an intractable task, unless NP $\subseteq$ P/poly. Hence the positive aspect of Theorems 9 and 10 is that circumscription is an extremely powerful tool for representing problems in a compact way.

We summarize the results in Table 1, where we divide cases between the two services, and between "short" and "long" queries. By "short" we mean clauses of fixed length (e.g., length 1 to ask for preferred exams in the lazy student example), while "long" means clauses of arbitrary length.

5. NMR for compact representation of knowledge II: default logic

In this section we show that the results presented in Section 4 are not peculiar of circumscription, but also apply to default logic.

We introduce a new example dealing with a student who wants to choose his/her *curriculum* to maximize his/her chances of obtaining a fellowship. We show that the natural way the student can do this is by formalizing the choice of the *curriculum* using default logic. Given that the problem is computationally intractable for simple default theories, we address the question of whether reasoning in such theories is a compilable problem. We show in this section that compilation is not feasible. Again, looking at the positive side of this results, we conclude that also default logic allows one to save a super-polynomial amount of space, in the best case.

Example 11 (*Fellowships for students*). A student wants to obtain a fellowship from HAL-9000, Inc. to pay for his/her university fees. In order to obtain it, he/she must present a *curriculum* that complies with the school rules and the requirements imposed by HAL-9000. The set of constraints imposed by the school officials is represented as a propositional formula W which might look like the following:

$$\texttt{DataBases} \wedge \texttt{Algebra} \rightarrow \texttt{NonMonotonicReasoning},$$

$$\neg\texttt{Algebra} \vee \neg\texttt{ComputationalComplexity},$$

$$\texttt{Algebra} \rightarrow \texttt{ReasoningAboutKnowledge},$$

$$\texttt{ComputationalComplexity} \rightarrow \texttt{ReasoningAboutKnowledge}.$$

The above formula is Horn, and the intended meaning of its clauses is that you must attend `NonMonotonicReasoning` if you attend `DataBases` and `Algebra`, you cannot attend both `Algebra` and `ComputationalComplexity` (e.g., because their contents overlap), you must attend `ReasoningAboutKnowledge` if you attend `Algebra`, etc. In addition to these constraints the student must satisfy the requirements imposed by HAL-9000. Since HAL-9000 people are interested in a specific field of computer science (say, algebra and computational complexity) they prefer applications containing as many courses as possible on this subject. Let $S = \{s_1, \ldots, s_k\}$ be the courses offered by the university on this specific subject. We can formalize these preferences via the set of default rules $D = \{\frac{:s_1}{s_1}, \ldots, \frac{:s_k}{s_k}\}$. Defaults of this kind are called prerequisite-free positive normal unary (PFPNU) in [43]. Intuitively, an extension of the default theory $\langle D, W\rangle$ is a *curriculum* which maximizes courses in S, while still satisfying constraints in W.

In our fellowship example, we have $S = \{\texttt{Algebra}, \texttt{ComputationalComplexity}\}$ and, therefore,

$$D = \left\{ \frac{:\texttt{Algebra}}{\texttt{Algebra}}, \frac{:\texttt{ComputationalComplexity}}{\texttt{ComputationalComplexity}} \right\}.$$

We call a course c *mandatory* (for the faculties) if taking it is implied by the constraints imposed, i.e., $W \models c$. A course c is *relevant* (for the fellowship) if there exists an extension E of $\langle D, W\rangle$ such that $c \in E$, i.e., if $\langle D, W\rangle \vdash_{\mathrm{CR}} c$. A course c is *necessary* (for the fellowship) if for all extensions E of $\langle D, W\rangle$ we have that $c \in E$, i.e., if $\langle D, W\rangle \vdash_{\mathrm{SK}} c$. In our specific case, there are no mandatory courses, `Algebra`, `ComputationalComplexity` and `ReasoningAboutKnowledge` are relevant while only `ReasoningAboutKnowledge` is necessary.

In the above scenario there are three possible computational services that a student might ask:

Problem 5. Decide whether a course is mandatory. The student has no choice but to take all the mandatory courses.

Problem 6. Decide whether a course is relevant. Taking such a course does not violate the constraints. Moreover it belongs to at least one *curriculum* that satisfies requirements imposed by HAL-9000.

Problem 7. Decide whether a course is necessary. Similar to the previous problem, but now the course belongs to all *curricula* satisfying the requirements imposed by HAL-9000.

The complexity of some of the above problems has been already studied in the literature. When W is Horn and D is PFPNU, we have the following figures:

- Problem 5 is polynomial [11].
- Problem 6 is NP-complete [43].
- Problem 7 is co-NP-complete (Corollary A.4 of the present paper).

Observe that Problems 6 and 7, although intractable, could be compiled by caching the answers to $\langle D, W\rangle \vdash_{\mathrm{SK}} c_i$ and $\langle D, W\rangle \vdash_{\mathrm{CR}} c_i$ for each course c_i. We now investigate what happens if we allow complex queries to such a default theory. Non-atomic queries are necessary for posing complex queries such as disjunction (e.g., is there at least one necessary course in a given set?) and conjunction (e.g., are all the courses in a set relevant?). The next theorem shows that it is very unlikely that a generalized version of Problem 6, where the query is either a positive conjunction or a negative disjunction, can be compiled.

Theorem 12. *If the problem* $[\langle D, W\rangle \vdash_{\mathrm{CR}} q, \langle D, W\rangle, q]$, *where W is a Horn theory, D is a set of PFPNU defaults, and q is either a conjunction of positive literals or a disjunction of negative literals, is compilable, then* $\mathrm{NP} \subseteq \mathrm{P/poly}$.

As for relevance queries, the only case where compilation is effective is when we restrict to disjunctive positive queries, that is, queries of the kind: "Is there at least one relevant course in the set S?".

Theorem 13. *The problem* $[\langle D, W\rangle \vdash_{\mathrm{CR}} q, \langle D, W\rangle, q]$, *where W is a Horn theory, D is a set of PFPNU defaults, and q is a disjunction of positive literals, is compilable.*

We now turn our attention to the problem of deciding whether courses are necessary to obtain the fellowship. If our problem is to decide whether at least one course in a given set is necessary, the following result tells us that a generalized version of Problem 7 is unlikely to be compilable.

Theorem 14. *If the problem* $[\langle D, W\rangle \vdash_{\mathrm{SK}} q, \langle D, W\rangle, q]$, *where W is a Horn theory, D is a set of PFPNU defaults, and q is either a disjunction of positive literals or a disjunction of negative ones, is compilable, then* $\mathrm{NP} \subseteq \mathrm{P/poly}$.

However, compilation is effective in reducing the complexity of on-line inference if only conjunctive queries are allowed.

Table 2
Compilability results for W Horn, D PFPNU; C = Compilable; NC = Not Compilable

CREDULOUS			SKEPTICAL		
conjunctive	disjunctive		conjunctive	disjunctive	
positive	positive	negative		positive	negative
NC	C	NC	C	NC	NC
Theorem 12	Theorem 13	Theorem 12	Theorem 15	Theorem 14	Theorem 14

Theorem 15. *The problem* $[\langle D, W\rangle \vdash_{SK} q, \langle D, W\rangle, q]$, *where* W *is a Horn theory,* D *is a set of PFPNU defaults, and* q *is a conjunction of literals, is compilable.*

Summing up, the theorems presented in this section show that, apart from restricted cases, if one wants to represent a default theory with a Horn W and a PFPNU set of defaults D with a propositional knowledge base K, either inference in K is intractable, or (if inference has to be kept tractable) K has super-polynomial size w.r.t. $|\langle D, W\rangle|$ in the worst case. Similarly to circumscription, default logic allows for compact representation of knowledge. Hence the positive aspect of Theorems 12 and 14 is that even simple default theories are extremely powerful for representing problems in a compact way.

We summarize the results in Table 2, where we divide cases between the two forms of reasoning (credulous and skeptical) and between conjunctive and disjunctive queries.

We conclude the section by comparing these results with those of the previous section. It is well known [12] that skeptical default logic easily simulates circumscription, using a default $\frac{:\neg a}{\neg a}$ for each atom a to be minimized. That is, $CIRC(T; P; Z) \models \alpha$ iff $\langle D, T\rangle \vdash_{SK} \alpha$, where $D = \{\frac{:\neg a}{\neg a} \mid a \in P\}$. By renaming each atom a with $\neg a$ in both the default theory and α, one can easily prove that circumscription can be simulated by a PFPNU default theory. Hence, the results of the previous section apply also to default logic: the problem $[\langle D, W\rangle \vdash_{SK} q, \langle D, W\rangle, q]$, where W is 2-CNF and q is a clause, is compilable for "short" queries, while it is not compilable for "long" queries (cf. first column of Table 1). Instead, the problem $[\langle D, W\rangle \vdash_{SK} q, W, (D, q)]$ (i.e., W is a fixed 2-CNF formula and both the set D of PFPNU defaults and the clause q are in the varying part) is not compilable, independently of the form of the query (cf. second column of Table 1). Note that results presented in this section refer to W Horn, hence they are incomparable with the ones derivable from Section 4. Moreover, credulous reasoning is peculiar to default logic and has no counterpart in circumscription.

6. NMR for compact representation of knowledge III: CWR

In Section 4 we showed that, under some conditions, circumscription allows one to save a super-polynomial amount of space when representing a body of knowledge. Because of the relations between propositional circumscription and closed-world reasoning (cf. Section 2), this holds for extended closed-world assumptions as well. More specifically, Theorems 9 and 10 hold if we replace $CIRC(T; P; Z)$ with $ECWA(T; P; Z)$.

What can be said about the other forms of closed-world reasoning that we mentioned in Section 2? Not all of them are able to capture enough knowledge to save super-polynomial space w.r.t. a purely propositional formula. As an example both the closed-world assumption $CWA(T)$ and the careful closed-world assumption $CCWA(T;P;Z)$ of a propositional formula T can always be represented with a propositional formula of size $O(|T|)$, since they are equivalent to T plus a set of literals. Therefore, if we consider classes of formulae in which deciding logical entailment is a polynomial task, such as Horn, dual-Horn and 2-CNF, we have compilable NMR problems:

Proposition 16. *The problems*

$$[CWA(T) \models q, T, q] \quad \textit{and} \quad [CCWA(T;P;Z) \models q, (T;P;Z), q],$$

where T is either Horn, dual-Horn or 2-CNF and q is a clause, are compilable.

In fact, we can compute off-line $CWA(T)$ or $CCWA(T)$ and decide on-line logical entailment. Note that adding negative literals to T does not increase its size by more than a linear factor and preserves its syntactic form (e.g., the *CCWA* of a Horn formula is still Horn).

While this result is not surprising, it nevertheless points out that compilability is not directly related to complexity of inference. We show this fact by focusing on CCWA, since it is more similar to circumscription as it allows one to express preferences among atoms. Inference in such a formalism can be intractable for classes of formulae in which classical inference is polynomial. As an example, in [9, Corollary 6] it is shown that checking whether a literal follows from the CCWA of a dual-Horn formula is co-NP-hard. Instead, testing if a dual-Horn formula logically implies a clause is a polynomial-time problem [11].

Therefore, our results provide a new tool for the comparison of nonmonotonic formalisms, apart from semantical considerations. In particular, CCWA and ECWA exhibit different computational behaviors: On-line reasoning in both of them is polynomially intractable, but only in the latter this is "justified" by the capability of capturing large amounts of knowledge with a compact formula.

7. Unsound and fast inference with NMR

As mentioned in Section 1, it has been frequently argued that one of the expected features of NMR was that it could account for a form of unsound, but *fast*, inference. Results on the computational complexity of NMR seem to contradict the possibility of NMR being faster than classical reasoning. In this section we show that there exist interesting scenarios where NMR is more efficient than classical reasoning even according to worst-case analysis, sometimes at the cost of losing soundness. We introduce this aspect by means of an example. This time the student must face constraints of a different syntactic form on his/her *curriculum*.

Example 17 (*The cautious student*). The faculty members represent the requirements needed to attend a course with a set of dependencies R:

$$\texttt{NonMonotonicReasoning} \longrightarrow \texttt{Algebra},$$

$$\texttt{NonMonotonicReasoning} \longrightarrow \texttt{Logic},$$

$$\texttt{DataBases} \longrightarrow \texttt{Algebra},$$

$$\texttt{ComputerArchitectures} \longrightarrow \texttt{Algebra},$$

$$\texttt{ReasoningAboutKnowledge} \longrightarrow \texttt{Logic},$$

$$\texttt{ComputationalComplexity} \longrightarrow \texttt{Logic},$$

i.e., to attend NonMonotonicReasoning a student should also attend Algebra, etc. Throughout this example we assume that such formulae are always definite Horn. A set such as $C = \{\texttt{NonMonotonicReasoning}, \texttt{ComputerArchitectures}\}$ can represent courses the student has attended. The models of $C \cup R$ represent all admissible completions of the *curriculum*.

A conjunct like

$$g_1 = \texttt{Algebra} \wedge \texttt{ReasoningAboutKnowledge} \wedge \neg\texttt{DataBases} \wedge \neg\texttt{ComputationalComplexity}$$

represents a plan of the student: courses the student may be interested in taking (positive literals) or avoiding (negative literals), but he/she has not committed yet. The student may also have alternative plans, such as

$$g_2 = \texttt{Logic} \wedge \texttt{ComputationalComplexity} \wedge \texttt{DataBases} \wedge \neg\texttt{ReasoningAboutKnowledge},$$

or

$$g_3 = \texttt{Algebra} \wedge \texttt{ComputerArchitectures} \wedge \neg\texttt{ReasoningAboutKnowledge}.$$

A plan g may be satisfied by a model of $C \cup R$ or it may not. The student wants to know if in all models at least one of his/her plans will be satisfied. This could be represented as a goal $G = g_1 \vee g_2 \vee g_3$.

The scenario could be modified if the faculties add new requirements to R or if the student makes further commitments, thus adding atoms to C.

Let us formalize the computational services the student may be interested in:

Problem 8. Decide whether the set of courses in C plus the courses required by R satisfy the goal G.

This service provides information on the current situation but it does not give any guarantee on the future. The set containing all and only the courses that the student is

obliged to attend in order to respect the present requirements corresponds to the minimal model of the definite Horn theory $C \cup R$. Since $CWA(C \cup R)$ corresponds to the minimal model of $C \cup R$, solving Problem 8 amounts to deciding whether $CWA(C \cup R) \models G$ holds.

Problem 9. Decide whether the goal G will be satisfied no matter which new requirements are imposed (in addition to R) by the faculties and which new courses (in addition to C) the student decides to attend.

This service provides information on the current and the future situation. We now have many distinct admissible completions of the curriculum. If we want to be sure that, no matter which new requirements are imposed and new courses are taken, the goal is satisfied, we must check that in all possible completions the goal is satisfied. Since the completions correspond to the models of $C \cup R$, solving Problem 9 amounts to decide whether $C \cup R \models G$ holds.

Coming back to the example, we have that $CWA(C \cup R) \models G$ while $C \cup R \not\models G$. Therefore, if the faculties do not change the set of requirements and the student does not decide to take additional courses, the goal will be satisfied.

Let's consider an alternative goal of the student: $G' = g'_1 \vee g'_2 \vee g'_3$, where

$$g'_1 = \texttt{Algebra} \wedge \texttt{ReasoningAboutKnowledge} \wedge \texttt{Logic} \wedge \neg\texttt{ComputationalComplexity},$$

$$g'_2 = \texttt{Logic} \wedge \texttt{ComputationalComplexity} \wedge \neg\texttt{ReasoningAboutKnowledge},$$

$$g'_3 = \texttt{Algebra} \wedge \neg.\texttt{ReasoningAboutKnowledge}.$$

Both $CWA(C \cup R) \models G'$ and $C \cup R \models G'$ hold. As a consequence, the student is sure that, whatever new requirements and courses are added, the goal G' will always be satisfied.

We consider now the complexities of the above problems.

- Problem 8 is polynomial: First compute the minimal model M of the Horn formula $R \wedge C$, then check whether $M \models G$. Both steps can be accomplished in polynomial time [11].
- Problem 9 is co-NP-complete: hardness follows from the co-NP-completeness of tautology checking of a formula in disjunctive normal form.

NMR is faster than classical reasoning in this specific case. Here NMR can be seen as a fast, complete but unsound approximation of classical reasoning, as for any pair of formulae Σ and γ, $\Sigma \models \gamma$ implies $CWA(\Sigma) \models \gamma$, but not the other way around.

This behavior of NMR is not restricted to this particular situation. We take a further example from the logic programming field. Let P be a propositional general logic program—where negation is allowed in the body of the rules—and γ be a clause. Deciding whether γ is true in all the (classical) models of P, i.e., $P \models \gamma$ interpreting `not` (negation) as classical negation, is a co-NP-complete problem. On the other hand, deciding whether γ is a consequence of P under the well-founded semantics [44], i.e., γ

is satisfied by the well-founded model of P (written $WF(P) \models \gamma$), is a polynomial-time problem.

Also in this case NMR accounts for an approximation of classical reasoning which is fast, complete, but unsound in general, as $P \models \gamma$ implies $WF(P) \models \gamma$, but not the other way around. In fact, in [44] it is noted that well-founded semantics does not use the excluded middle rule (i.e., an atom is mapped either to **true** or to **false**, and not into both), which can be seen precisely as the source of complexity in propositional calculus.

Let us consider another example clarifying the kind of approximation made by the well-founded semantics.

Example 18 (*The short-sighted student*). Recall the previous example, and suppose the faculty adopts more liberal requirements about courses, letting the student make some choices. The new requirements are represented as a set of disjunctive rules R:

$$\texttt{ReasoningAboutKnowledge} \longrightarrow \texttt{DataBases} \vee \texttt{NonMonotonicReasoning},$$

$$\texttt{DataBases} \longrightarrow \texttt{Algebra} \vee \texttt{Logic},$$

$$\texttt{NonMonotonicReasoning} \longrightarrow \texttt{ComputationalComplexity},$$

$$\texttt{ComputationalComplexity} \longrightarrow \texttt{Logic}.$$

Again, a set such as $C = \{\texttt{ReasoningAboutKnowledge}\}$ can represent courses the student has attended. The models of $C \cup R$ represent all admissible completions of the *curriculum*. As defined in Section 4, a course c is *mandatory* if it has to be done in all *curricula*, i.e., $C \cup R \models c$.

Problem 10. Decide whether a course is mandatory.

It can be shown that unsatisfiability of 3-CNF formulae can be reduced to this problem, hence we can conclude that the problem is co-NP-complete (cf. e.g., [19]).

Of course, Problem 10 can be answered "no" if one finds an admissible *curriculum* not containing the given course. Now suppose the student wants to find such a *curriculum*. Since the (intractable) problem involves some hidden combinatorics, the student uses the following heuristic. The student has some local preferences when a choice has to be made, i.e., assume that for each rule with a disjunctive head, the student has a (local) preference about the courses he/she must choose among: one is better than the other ones in the head. In the example, `NonMonotonicReasoning` could be preferred to `DataBases` in the first rule, and `Algebra` could be preferred to `Logic` in the second rule. Starting from C, the student applies each rule whose body is satisfied, adding one of the courses in the head of a rule.

When a choice in a disjunction has to be made, the student adds the preferred course, but only if none of the other courses has already been added—this criterion of minimal addition corresponds to the reasonable assumption that the student does not want to add unnecessary courses. In the example, starting from $C = \{\texttt{ReasoningAboutKnowledge}\}$, the student finds the *curriculum*

$$\left\{\begin{array}{ll} \texttt{ReasoningAboutKnowledge}, & \texttt{NonMonotonicReasoning}, \\ \texttt{ComputationalComplexity}, & \texttt{Logic} \end{array}\right\}.$$

Finally, the student decides if a given course is mandatory or not based on the presence of the course in the *curriculum* built in the above way. E.g., the student decides that `Logic` is mandatory.

Observe that the preferences expressed by the student are just local to a rule, and not global (this justifies the title of the example). In fact, the student prefers `Algebra` to `Logic`, but decides that `Logic` is mandatory, even if there is a propositional model {`ReasoningAboutKnowledge`, `DataBases`, `Algebra`} where `Logic` is not chosen while `Algebra` is. In this respect the approximation made by the above reasoning is unsound, i.e., a "yes" answer cannot be trusted.

We also want to point out that the above approximate reasoning is also nonmonotonic. Nonmonotonicity arises as the student commits to more courses, e.g., starting from $C \cup \{\texttt{DataBases}\}$, the student finds the other *curriculum* {`ReasoningAboutKnowledge`, `DataBases`, `Algebra`}, and now decides that `Logic` is no longer mandatory.

The construction of the *curriculum* can be mimicked (and formalized) by well-founded semantics in the following way: First, R is transformed into a new set of rules R', where local preferences in disjunctions are expressed by shifting all courses but the preferred one in the body of the rule:

$$\texttt{ReasoningAboutKnowledge} \wedge \neg\texttt{DataBases} \longrightarrow \texttt{NonMonotonicReasoning},$$

$$\texttt{DataBases} \wedge \neg\texttt{Logic} \longrightarrow \texttt{Algebra},$$

$$\texttt{NonMonotonicReasoning} \longrightarrow \texttt{ComputationalComplexity},$$

$$\texttt{ComputationalComplexity} \longrightarrow \texttt{Logic}.$$

Even if $C \cup R'$ is propositionally equivalent to $C \cup R$, the syntactic form suggests now a logic programming approach. In fact, the process by which the student builds the model starting from C can be captured by well-founded semantics: the well-founded model $WF(C \cup R')$ is exactly

$$\left\{\begin{array}{ll} \texttt{ReasoningAboutKnowledge}, & \texttt{NonMonotonicReasoning}, \\ \texttt{ComputationalComplexity}, & \texttt{Logic}, \\ \neg\texttt{DataBases}, & \neg\texttt{Algebra} \end{array}\right\}.$$

Now we can approximate the problem of deciding whether a course c is mandatory in the following way: we assume that c is mandatory if $c \in WF(C \cup R')$, c is not mandatory if $\neg c \in WF(C \cup R')$ while we do not decide when neither c nor $\neg c$ belong to $WF(C \cup R')$. Hence, the approximation we obtain is unsound, but complete: in fact, if a course appears as a negative atom in the well-founded model, then it is a non-mandatory course for sure.

Once formalized with well-founded semantics, we recognize that the above heuristic procedure is a fast and unsound approximation of classical reasoning. In fact, it will

find an answer in polynomial time, but may not provide us with a completely defined and accurate *curriculum*.

What we obtain is that negative atoms in the well-founded model correspond to definitely avoidable courses, while undefined atoms—if any—correspond to those courses whose necessity in the *curriculum* being built we have been unable to settle.

8. Related work

Our notion of compilability bears some resemblance with the notion of *expression complexity* introduced by Vardi in [45], and further studied in [46], to characterize the complexity of relational query languages. However, expression complexity captures the complexity of applying a query to a *fixed* data or knowledge base, while in our compilability approach we consider a compiled (not fixed) knowledge base. Therefore, differently from Vardi's work, our analysis takes into account the size of both the original knowledge base and the compiled one.

Some work related to issues dealt with in the present paper—e.g., NMR for saving representational space, trading off-line computation for on-line efficiency—appeared in the literature. The approach taken in such papers is now briefly compared to ours.

Borgida and Etherington [2] propose a system for representing hierarchical knowledge, i.e., subset relations among classes of individual objects. In the system it is possible to say, for example, that cats, snakes and ferrets are rodent eaters and all rodent eaters are carnivore. Moreover it is possible to assign individual objects to classes, e.g., Tom is a cat. Both kinds of knowledge are represented as logical formulae. Inference of positive facts is governed by the hierarchy between classes, e.g., it can be inferred that Tom is a carnivore. Inference of negative facts is ruled by the generalized CWA of the logical formula: as an example, if dogs are not cats, and the most specific fact about Tom we know is that he is a cat, then we infer Tom is not a dog. Clearly this nonmonotonic feature makes reasoning unsound; nevertheless, since negative facts must not be explicitly stored, storage space is reduced (very roughly, an improvement by a $\sqrt{n}$ factor).

When disjunctive knowledge about individuals is to be represented, e.g., Sid is either a cat or a ferret, the system looks for the most specific class that can represent the disjunction. In the present case, the system records the fact that Sid is a rodent eater. Clearly this is a source of incompleteness in reasoning—as rodent eaters include snakes as well—and also in this case space is saved, as no further class is ever created. Summing up, the knowledge representation system allows for saving storage space, at the cost of losing both soundness and completeness.

Selman and Kautz are the first authors to introduce the idea of compilation in the knowledge representation field. In [24,41] they propose to compile a propositional theory for obtaining two Horn formulae, called respectively *Horn least upper bound* and *Horn greatest lower bound*. The main difference with our approach is in that we require that the compilation process completely preserves the informational content of the original theory, while in their approach only approximations are obtained. In fact, the Horn greatest lower bound allows for complete and unsound reasoning w.r.t. the original formula, while the Horn least upper bound allows for sound and incomplete reasoning.

The impact of the preprocessing in terms of the complexity (space and time) of reasoning in the Horn approximations is a bit complicated. Reasoning with a Horn formula is a polynomial-time problem in the size of the formula itself, therefore Horn compilation should allow for faster on-line reasoning. Anyway the Horn least upper bound may have super-polynomial size w.r.t. the original formula, as they prove in [24]. This proof is the first one, to the best of our knowledge, that uses non-uniform complexity classes to prove non-compilability.

We want to point out that some semantical aspects of Horn approximations can be explained in terms of nonmonotonic reasoning, as shown in [5].

Other papers discuss the possibility of reducing the complexity of query answering through off-line preprocessing, specifically focusing on inference in classical propositional logic.

Moses and Tennenholtz [32] analyze the possibility of speeding up the complexity of query answering through a previous off-line analysis of the knowledge base. Their goal can be considered as a special case of our notion of compilation: they consider a particular subset of all queries, which they call *efficient basis*, whose answers enable to answer all queries in polynomial time. Some query languages may not admit an efficient basis. Our results complement theirs, as we consider any possible preprocessing (even non-recursive preprocessing) with the only restriction that the new representation can answer queries in time polynomial in the size of the original knowledge base. For each entry of Tables 1 and 2 marked "non-compilable", we proved that not only no efficient basis exists but also that no other compilation is possible.

The possibility of rewriting off-line a propositional theory, was addressed by Kautz, Kearns and Selman in [22]. In particular, they investigate the reformulation of a Horn formula into the set of its *characteristic* models, where characteristic models are independent models that cannot be obtained as intersection of the others. As it turns out, the compactness of the representation using Horn clauses and characteristic models cannot be compared, as it may be the case that one is exponentially more succinct than the other in specific cases. Nevertheless, some forms of inference are simpler if a formula is represented via the set of its characteristic models.

This idea has been expanded by Khardon and Roth in two papers [25,26] where they present a new framework for learning and reasoning. In particular, in [26] they analyze the possibility of rewriting a propositional theory in a new form that admits a polynomial-time inference algorithm. More precisely, they generalize the notion of characteristic models introduced by Kautz, Kearns and Selman, to apply to all formulae, not just Horn ones. When knowledge bases are represented in this format, inference can be performed in polynomial time. Note, however, that not all boolean functions have a small set of characteristic models: there exist boolean functions whose CNF representation has size polynomial in the number of propositional variables but whose set of characteristic models has exponential size.

Selman studied planning problems structured in fixed and varying parts in [40], analyzing the impact in terms of computational complexity of such a structure. An instance of a planning problem is defined by an initial state, a final (i.e., desired) state, and a set of operators that can change the state, where a state is a value assignment to a set of state variables. The framework in which the three components are part of the input

of the planning problem is called *domain-independent* planning. If the set of operators is fixed, then we have *domain-dependent* planning. Finally, if just the initial state is part of the input, then we have *reactive* planning. Some planning problems, formulated in the "domain-independent version", are polynomially intractable, while they become polynomial in the "domain-dependent version". Trade-offs between space and time arise when reactive planning is considered. The notion of *universal* plan is introduced, which is a function that is able to compute "the next step to do", given the description of the current state. Since the set of possible situations is exponential in general, this may cause a blow-up in the size of the universal plan. In fact, Selman is able to prove that there exist cases in which such a super-polynomial blow-up must happen, unless NP $\subseteq$ P/poly.

Recently, Nerode et al. [33] investigated reasoning in deductive databases where predicates can be minimized with circumscription. Their explicit goal is to compile the database off-line, trading the space needed to store the compiled database with the time gained in on-line query answering. Since they use the ground representation of the database, which is a propositional formula, our results apply. In particular, on the practical side, the propositional theories we use in our proofs yield databases which can be used as worst-case benchmarks for compiling algorithms.

9. Conclusions

Recent theoretical results on the computational complexity of NMR seem to contradict two of the main reasons for the development of nonmonotonic formalisms, namely that defeasible assumptions should allow for: (1) faster, although unsound, inference and (2) more compact representation of knowledge. We have shown in this paper that, to some extent, NMR fulfills separately its goals.

Regarding the second goal, we have proven that circumscription and default logic do indeed allow for more compact representation of knowledge. These results can also be extended to circumscription of general propositional formulae and to many formalisms for belief revision and update (as shown in [8, 17] and in [6], respectively). In fact, in the present work we do not prove that the space of a purely propositional representation of a nonmonotonic formula must be super-polynomial (this is proven in [8, 17]). Rather we prove that either there is no such polynomially-sized formula, or, if it exists, it is impossible to extract information from it in polynomial time (cf. Definition 1). Some results about intractability of NMR can therefore be interpreted as the price one has to pay for having extra-compact representation of knowledge. It is therefore unfair to say that NMR is harder than classical reasoning. In fact the input of a nonmonotonic inference problem could be super-polynomially smaller than the input for solving with classical inference the same problem. On the other hand the implicit assumption in saying that NMR is harder is that the sizes of the inputs are the same.

Regarding the first goal, we have given prototypical scenarios where NMR accounts for a faster and unsound approximation of classical reasoning. Therefore NMR is not always computationally harder, and may even be simpler than classical reasoning.

Acknowledgements

The authors are grateful to the anonymous referees for their comments and suggestions. This work has been supported by Agenzia Spaziale Italiana (ASI), Ministero dell'Università e della Ricerca Scientifica e Tecnologica (MURST), and Consiglio Nazionale delle Ricerche (SARI Project).

Appendix A. Proof of theorems

Proofs of Section 4

Theorem 10. *If the problem* $[CIRC(T;P;Z) \models q, (T,P,Z), q]$, *where* T *is 2-CNF and* q *is a clause, is compilable, then* NP $\subseteq$ P/poly.

Proof. To apply Theorem 5, we first need to choose an appropriate NP-complete problem $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, and for each n to define a 2-CNF formula T_n of size polynomial in n and a partition $\langle P_n; Z_n \rangle$ of the alphabet of T_n. Then, we need to show that for all $\pi \in \Pi_n$ there exists a clause q_π computable in polynomial time such that $CIRC(T_n; P_n; Z_n) \not\models q_\pi$ iff π is a "yes" instance of Π.

We choose the NP-complete problem 3SAT as our Π. Our proof is based on a reduction given in [9, Theorem 16], which showed that circumscriptive inference of literals in positive 2-CNF formulae is co-NP-hard. Let L be the alphabet of n atoms used in Π_n. Let L' be the alphabet $L \cup \overline{L} \cup C \cup D \cup E \cup \{z\}$, where $\overline{L}$ and C are defined in formulae (3) and (4), respectively, and D and E are isomorphic to C, i.e., $D = \{d_i \mid \gamma_i$ is a three-literal clause of $L\}$, and similarly for E. Negative literals of L' are denoted by $\neg x$, where $x \in L'$.

We define T_n on the alphabet L' according to the following rules:

(1) For each atom a of L, there is a clause $a \vee \overline{a}$ in T_n.

(2) For each clause $\gamma_i = w \vee x \vee y$ in $\pi_{\max(n)}$, where w, x and y are literals of L, there are seven clauses in T_n: $c_i \vee w$, $c_i \vee x$, $c_i \vee y$, $c_i \vee z$, $c_i \vee d_i$, $d_i \vee e_i$, $\neg d_i \vee \neg e_i$. Observe that the last two clauses express an exclusive-or between d_i and e_i.

Define the sets of atoms as: $P_n = L \cup \overline{L} \cup C \cup D \cup E$, and $Z_n = \{z\}$, i.e., all atoms of L' but z are minimized. Note that T_n, P_n and Z_n have size polynomial in the size n of L, and that T_n is a 2-CNF formula.

Given a generic $\pi \in \Pi_n$, let q_π be the clause

$$q_\pi = z \vee \left(\bigvee_{\gamma_i \in \pi} \neg d_i \right) \vee \left(\bigvee_{\gamma_i \notin \pi} d_i \right).$$

Note that q_π can be computed in time polynomial in the size of π. Apart from z, negative literals of q_π correspond to clauses appearing in π, whereas positive literals correspond to the ones not present in π.

To make the proof more readable, we define the sets

$$C_\pi = \{c_i \mid \gamma_i \in \pi\}, \qquad D_\pi = \{d_i \mid \gamma_i \in \pi\}, \qquad E_\pi = \{e_i \mid \gamma_i \in \pi\}.$$

We now prove the theorem by showing that π is satisfiable iff $CIRC(T_n; P_n; Z_n) \models q_\pi$.

($\Rightarrow$) Let $M \subseteq L$ be a model for π, and let $\overline{M} = \{\overline{a} \in \overline{L} \mid a \in M\}$. Define MM as $(L - M) \cup \overline{M} \cup C \cup D_\pi \cup (E - E_\pi)$. It can be verified that MM is a model of T_n, because it includes C, for every atom $a \in L$ it contains either a or $\overline{a}$, and the exclusive-or's between the D's and the E's are satisfied. Moreover, q_π is false in MM.

To prove that MM is $(P_n; Z_n)$-minimal, we consider separately atoms in $L \cup \overline{L} \cup D \cup E$, atoms in C_π, and atoms in $C - C_\pi$.

First, no atom x from $L \cup \overline{L}$ can be eliminated from MM without introducing the corresponding atom $\overline{x}$, since in T_n there is the clause $x \vee \overline{x}$. Moreover, no atom in D can be eliminated without introducing the corresponding atom in E, and vice versa, because of the exclusive-or's.

Secondly, consider the elimination of (say) $c_i \in C_\pi$, and let $w \vee x \vee y$ be the ith clause of π. Since M is a model of π, M satisfies at least one literal, say, w. By construction of MM, the atom $w \notin MM$. Since the clause $c_i \vee w$ is in T_n, the atom c_i cannot be eliminated without introducing w.

Finally, an atom $c_i \in C - C_\pi$ cannot be eliminated without introducing the corresponding $d_i \in D - D_\pi$, because there is the clause $c_i \vee d_i$, and $d_i \notin MM$ by construction. Therefore, MM is minimal.

($\Leftarrow$) Let MM be a $(P_n; Z_n)$-minimal model of T_n, such that q_π is false in MM. From the falsity of q_π we know that $z \notin MM$, $D_\pi \subseteq MM$, and $(D - D_\pi) \cap MM = \emptyset$. From this and the exclusive-or's between atoms of D and E, we know that $E - E_\pi \subseteq MM$, and $E_\pi \cap MM = \emptyset$. Moreover, since for each $c_i \in C$, the clause $c_i \vee z$ is in T_n, $C \subseteq MM$. Finally, for each atom $a \in L$ at least one of a and $\overline{a}$ is in MM, because of the clause $a \vee \overline{a}$, and not both because of minimality of MM.

Consider an atom $c_i \in C_\pi$ such that $\gamma_i = w \vee x \vee y$. There are in T_n the three clauses $c_i \vee w$, $c_i \vee x$, $c_i \vee y$. We prove, by contradiction, that at least one of w, x and y is not in MM. If all of w, x and y were in MM, then c_i could be eliminated from MM (introducing z to satisfy the fourth clause $c_i \vee z$), obtaining a model MM' with fewer atoms of P_n, and MM would not be minimal, contradicting the hypothesis. Note that since $D_\pi \subseteq MM$, the clause $c_i \vee d_i$ is still satisfied in MM'.

Define $M = \{a \in L \mid \overline{a} \in MM\}$. Observe that $a \in M$ if and only if $a \notin MM$, because $a \in MM$ iff $\overline{a} \notin MM$. Moreover, M is a model of π because for each clause $w \vee x \vee y$ in π, at least one of w, x and y is not in MM, and therefore at least one of them is mapped to **true** by M. □

Proofs of Section 5

The following lemma, stating a property of Horn theories, is useful in the proofs concerning default theories.

Lemma A.1. *Let H be a Horn theory, C a conjunction of positive literals and D a disjunction of negative literals, where the set of atoms appearing in C is disjoint from the set of atoms appearing in D and the atoms in D occur only positively in H. Furthermore, assume that $H \not\models D$ and that $H \cup \{C \rightarrow D\} \models D$. This implies that $H \models C$.*

Proof. First of all, note that $\{C \rightarrow D\}$ is a Horn clause. From $H \cup \{C \rightarrow D\} \models D$ it follows that $H \models (C \rightarrow D) \rightarrow D$. With simple rewritings, we obtain $H \models C \vee D$, and, therefore $H \cup \{\neg D\} \cup \{\neg C\}$ is unsatisfiable. Since $H \not\models D$, $H \cup \{\neg D\}$ is satisfiable. Observe that $\{\neg D\}$ is a set of positive atoms. Hence the set $Res(H, \{\neg D\})$ of clauses obtained by resolving all clauses of H with all atoms in $\neg D$ does not contain the empty clause, while the set $Res(Res(H, \{\neg D\}), \neg C)$ contains the empty clause. By hypothesis, atoms of D occur only positively in H, thus resolving atoms of $\neg D$ with H will not generate any new clause; in other words, $Res(H, \{\neg D\}) = H \cup \{\neg D\}$. Since atoms of D do not appear in $\neg C$, it follows that $Res(Res(H, \{\neg D\}), \neg C) = Res(H, \neg C) \cup \{\neg D\}$. Since $Res(Res(H, \{\neg D\}), \neg C)$ contains the empty clause, so does $Res(H, \neg C) \cup \{\neg D\}$. But $\{\neg D\}$ does not contain the empty clause, therefore $Res(H, \neg C)$ contains it. As a consequence, $H \wedge \neg C$ is unsatisfiable, i.e., $H \models C$. □

Theorem 12. *If the problem* $[\langle D, W\rangle \vdash_{CR} q, \langle D, W\rangle, q]$, *where* W *is a Horn theory,* D *is a set of PFPNU defaults, and* q *is either a conjunction of positive literals or a disjunction of negative literals, is compilable, then* NP $\subseteq$ P/poly.

Proof. We apply Theorem 5. We choose an appropriate NP-complete problem $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, a set of PFPNU defaults D_n, and a Horn theory W_n of size polynomial in n, and show that for all $\pi \in \Pi_n$ there exists a query q_π computable in polynomial time such that $\langle D_n, W_n\rangle \vdash_{CR} q_\pi$ iff π is a "yes" instance of Π.

We choose the NP-complete problem NOT-ALL-EQUAL-3SAT as our Π. Our proof is based on a reduction given in [43, Theorem 1], which showed that credulous inference in default theories where W is Horn and D is PFPNU is NP-hard.

Let L be the alphabet of n atoms used in the 3-CNF formulae of size n, i.e., in Π_n. Using the conventions introduced at the end of Section 3, we denote with a, b, c propositional atoms. Let L' be the alphabet $L \cup G \cup S \cup R$, where G is defined as in formula (5) and S, R are two sets of new literals one-to-one with atoms in G.

We define W_n and D_n on the alphabet L' according to the following rules:

(1) For each three-atoms clause $\gamma_i = a \vee b \vee c$, where $a, b, c \in L$, there are five clauses in W_n. The first one is $\neg a \vee \neg b \vee \neg c \vee \neg s_i$, and the others are $\neg a \vee g_i$, $\neg b \vee g_i$, $\neg c \vee g_i$, $\neg s_i \vee r_i$, where $s_i \in S$, $r_i \in R$, $g_i \in G$ are the atoms corresponding to γ_i.

(2) W_n also contains the clause $\Gamma = (\bigvee_{g \in G} \neg g) \vee (\bigvee_{r \in R} \neg r)$.

(3) For each atom $l \in L$ there is a default $d_l = \frac{:l}{l}$ in D_n.

(4) For each atom $s \in S$ there is a default $d_s = \frac{:s}{s}$ in D_n.

Note that the size of $\langle D_n, W_n\rangle$ is $O(n^3)$, W_n is a Horn theory and D_n is PFPNU.

Remark. From the above definition of D_n, it follows that an extension E of $\langle D_n, W_n\rangle$ is complete over $L \cup S$, i.e., for any atom $l \in L \cup S$ either $l \in E$ or $\neg l \in E$.

Given a generic $\pi \in \Pi_n$, that is a 3-CNF positive formula over L, a *maximal solution* for π is a set L_π of atoms mapped to **true** such that every superset of L_π is not a solution for π. Clearly, if π has a solution it has a maximal solution. Furthermore, we

denote with G_π the set of atoms of G corresponding to the clauses in π and similarly for S_π and R_π.

Since the theorem holds for both positive conjunctions and negative disjunctions, we define two distinct q_π. We call CPQ_π the conjunctive positive query and DNQ_π the disjunctive negative query:

$$CPQ_\pi = \bigwedge\{s \mid s \in S_\pi\} \wedge \bigwedge\{g \mid g \in G_\pi\}, \tag{A.1}$$

$$DNQ_\pi = \bigvee\{\neg r \mid r \notin R_\pi\} \vee \bigvee\{\neg g \mid g \notin G_\pi\}. \tag{A.2}$$

Note that both queries can be computed in time polynomial in the size of π. We need now to prove a lemma relating CPQ_π with DNQ_π.

Lemma A.2. *Let E be an extension of $\langle D_n, W_n\rangle$. Then $CPQ_\pi \in E$ iff $DNQ_\pi \in E$.*

Proof. Observe that the clause Γ in W_n is equivalent to the formula $((\bigwedge G_\pi) \wedge (\bigwedge R_\pi)) \rightarrow DNQ_\pi$. Let $\Sigma = CONS(GD(E, \langle D_n, W_n\rangle))$.

($\Rightarrow$) If $CPQ_\pi \in E$, then $G_\pi \subseteq E$. Moreover, also $R_\pi \subseteq E$ because $S_\pi \subseteq E$ and because of the clauses $\neg s_i \vee r_i$ in W_n. Since $((\wedge G_\pi) \wedge (\wedge R_\pi)) \rightarrow DNQ_\pi \in E$, also $DNQ_\pi \in E$.

($\Leftarrow$) If $DNQ_\pi \in E$, then by definition of generating defaults, we have $W_n \cup \Sigma \models DNQ_\pi$. Now, observe that $(W_n - \Gamma) \cup \Sigma \cup \{\neg DNQ_\pi\}$ is satisfiable, hence $(W_n - \Gamma) \cup \Sigma \not\models DNQ_\pi$. Since $W_n \cup \Sigma$ is a Horn theory, $G_\pi \wedge R_\pi$ is a conjunction of positive literals, DNQ_π is a disjunction of negative literals and the other conditions of Lemma A.1 are satisfied, we have $W_n \cup \Sigma \models G_\pi \wedge R_\pi$. Let $SR_\pi = \{\neg s_i \vee r_i \mid \gamma_i \in \pi\}$. Obviously $(W_n - SR_\pi) \cup \Sigma \not\models R_\pi$ but $W_n \cup \Sigma \models R_\pi$, thus applying again Lemma A.1 for each clause in SR_π we have $W_n \cup \Sigma \models S_\pi$. Therefore, $W_n \cup \Sigma \models G_\pi \wedge S_\pi$, that is $CPQ_\pi \in E$. □

Proof of Theorem 12 (*continued*). We now prove that $\langle D_n, W_n\rangle \vdash_{CR} CPQ_\pi$ iff π has a solution.

($\Leftarrow$) Let L_π be a maximal solution for π. We show that there exists an S_x such that $\emptyset \subseteq S_x \subseteq (S - S_\pi)$ and the set $E_\pi = \{\alpha \mid W_n \cup L_\pi \cup S_\pi \cup S_x \models \alpha\}$ is an extension of $\langle D_n, W_n\rangle$. Note that $S_\pi \subseteq E_\pi$ by construction of E_π. Moreover, since L_π is a solution of π, for all clauses $\gamma_i = a \vee b \vee c \in \pi$ at least one of a, b and c is in E_π. Therefore, $g_i \in E_\pi$ and consequently, $CPQ_\pi \in E_\pi$.

S_x is simply found by first applying all defaults d_l, d_s such that $l \in L_\pi$ and $s \in S_\pi$. Now apply as many defaults d_s such that $s \in S - S_\pi$ as possible, while keeping the resulting theory consistent. S_x is formed by the conclusions of the applied defaults d_s. If none of them was applicable S_x is empty.

We now prove that E_π is an extension of $\langle D_n, W_n\rangle$, i.e., we show that E_π is consistent and that all applicable defaults of D_n have been applied. Consistency is ensured by the fact that L_π is a solution for π.

By definition of E_π, all applicable defaults of the form $d = \frac{:s_i}{s_i}$, where $s_i \in S_\pi$, have been applied. Moreover, by construction of S_x, all applicable defaults d_s, where $s \in S - S_\pi$, have been applied. Let us assume that there exists an $l \in L - L_\pi$ such

that $d_l = \frac{:l}{l}$ is applicable. We prove that this implies $L_\pi \cup \{l\}$ is a solution for π, thus contradicting the maximality of L_π. For each clause $\gamma_i = a \vee b \vee c \in \pi$ at least one of a, b, c is in E, since L_π is a solution. Moreover, since both $\neg a \vee \neg b \vee \neg c \vee \neg s_i$ and s_i are in E, at least one of $\neg a$, $\neg b$, $\neg c$ must be in E also after $\frac{:l}{l}$ has been applied. As a consequence, $L_\pi \cup \{l\}$ is a solution for π, thus contradicting the maximality of L_π.

($\Rightarrow$) Let us assume that $\langle D_n, W_n\rangle \vdash_{CR} CPQ_\pi$. As a consequence, there exists an extension E of $\langle D_n, W_n\rangle$ such that $CPQ_\pi \in E$. Hence, for any i such that $\gamma_i = a \vee b \vee c \in \pi$ we have that $g_i \in E$ and $s_i \in E$. Since g_i appears only positively in W_n, except for Γ, from Lemma A.1 it follows that at least one atom of a, b, c is in E. Furthermore, since $s_i \in E$, at least one atom of a, b, c is not in E. Thus, $L \cap E$ is a solution for π.

We now prove that $\langle D_n, W_n\rangle \vdash_{CR} DNQ_\pi$ iff π has a solution. This immediately follows from the fact that $\langle D_n, W_n\rangle \vdash_{CR} CPQ_\pi$ and from Lemma A.2 it follows that $\langle D_n, W_n\rangle \vdash_{CR} CPQ_\pi$ iff $\langle D_n, W_n\rangle \vdash_{CR} DNQ_\pi$. This completes the proof. □

Theorem 13. *The problem* $[\langle D, W\rangle \vdash_{CR} q, \langle D, W\rangle, q]$, *where W is a Horn theory, D is a set of PFPNU defaults, and q is a disjunction of positive literals, is compilable.*

Proof. We prove this theorem by first showing a decomposability property of default theories with Horn W and PFPNU D.

Lemma A.3. *Let W be a Horn theory, D a PFPNU set of defaults and $\gamma = a_1 \vee \cdots \vee a_k$ a positive disjunction. $\langle D, W\rangle \vdash_{CR} \gamma$ iff there exists an a_i $(1 \leqslant i \leqslant k)$ such that $\langle D, W\rangle \vdash_{CR} a_i$.*

Proof. First of all, note that any extension E of $\langle D, W\rangle$ can be represented by the Horn theory $T = W \cup \{l \mid \frac{:l}{l} \in GD(E, \langle D, W\rangle)\}$. Since T is Horn and γ is a positive disjunction, it is well known that $T \models \gamma$ iff there exists an a_i $(1 \leqslant i \leqslant k)$ such that $T \models a_i$. Hence, if $\langle D, W\rangle \vdash_{CR} \gamma$ then there exists an a_i $(1 \leqslant i \leqslant k)$ such that $\langle D, W\rangle \vdash_{CR} a_i$. The other direction is trivial, thus the lemma follows. □

Proof of Theorem 13 (*continued*). This problem can now be solved by constructing off-line a table where for every atom $l \in L$ we store whether $\langle D, W\rangle \vdash_{CR} l$ or not. By Lemma A.3, deciding $\langle D, W\rangle \vdash_{CR} \gamma$ can be reduced to k look-ups to the above mentioned table, obviously in polynomial time. □

Theorem 14. *If the problem* $[\langle D, W\rangle \vdash_{SK} q, \langle D, W\rangle, q]$, *where W is a Horn theory, D is a set of PFPNU defaults, and q is either a disjunction of positive literals or a disjunction of negative ones, is compilable, then* NP $\subseteq$ P/poly.

Proof. We exploit Theorem 5 by first choosing an appropriate NP-complete problem $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$, a set D_n of PFPNU defaults, a Horn theory W_n of size polynomial in n, and then by showing that for all $\pi \in \Pi_n$ there exists a disjunction q_π computable in polynomial time such that $\langle D_n, W_n\rangle \vdash_{SK} q_\pi$ iff π is a "yes" instance of Π. We first focus on positive disjunctions.

We choose the NP-complete problem NOT-ALL-EQUAL-3SAT as our Π. Let L be the alphabet of n atoms used in the 3-CNF formulae of size n, i.e., in Π_n. $\overline{L}$ is a set of new atoms one-to-one with the atoms of L, as in formula (3). Let L' be the alphabet $L \cup \overline{L} \cup G$, where G is defined as in formula (5).

We define W_n and D_n on the alphabet L' according to the following rules:

(1) For each three-atoms clause $\gamma_i = a \vee b \vee c$ there are two clauses in W_n:
  (a) $\neg a \vee \neg b \vee \neg c \vee g_i$;
  (b) $\neg \overline{a} \vee \neg \overline{b} \vee \neg \overline{c} \vee g_i$.

(2) For each atom $l \in L$ the clause $\neg l \vee \neg \overline{l}$ is in W_n.

(3) For each atom $l \in L$ there are two defaults in D_n:
  (a) $\frac{:l}{l}$;
  (b) $\frac{:\overline{l}}{\overline{l}}$.

Note that the size of $\langle D_n, W_n \rangle$ is $O(n^3)$, W_n is a Horn theory and D_n is PFPNU.

Let π be a generic 3-CNF positive formula over L, and G_π be the subset of the atoms of G corresponding to the clauses in π. Let $q_\pi = \bigvee\{g \mid g \in G_\pi\}$. It is clear that q_π can be computed in time polynomial in the size of π.

We now prove that $\langle D_n, W_n \rangle \nvdash_{SK} q_\pi$ iff π is a "yes" instance of NOT-ALL-EQUAL-3SAT.

($\Leftarrow$) We assume π has solutions, and L_π is one of them. We show that $\langle D_n, W_n \rangle \nvdash_{SK} q_\pi$. It is sufficient to exhibit an extension E of $\langle D_n, W_n \rangle$ such that $E \not\models q_\pi$.

We prove that $E_\pi = \{\alpha \mid W_n \cup L_\pi \cup \{\overline{l} \mid l \notin L_\pi\} \models \alpha\}$ is such an extension. Let $GD(E_\pi, \langle D_n, W_n \rangle)$ be the set of generating defaults of E_π. We show that:

(1) $GD(E_\pi, \langle D_n, W_n \rangle)$ is maximal: from constraints $\neg l \vee \neg \overline{l}$ in W_n, no other default can be applied without generating a contradiction.

(2) $W_n \cup L_\pi \cup \{\overline{l} \mid l \notin L_\pi\}$ is consistent: the only way to generate an inconsistency is to add to W_n both l and $\overline{l}$, for some $l \in L$, which is not the case of E.

(3) $W_n \cup L_\pi \cup \{\overline{l} \mid l \notin L_\pi\} \not\models q_\pi$: if this is not the case, L_π would not be a solution to π, as we prove now. First of all, $W_n \cup L_\pi \cup \{\overline{l} \mid l \notin L_\pi\}$ is a Horn theory; therefore it implies the positive disjunction q_π iff it implies at least one of its disjuncts $g \in G_\pi$. If we want W_n to imply some positive literal g_i, Lemma A.1 gives us only two possibilities:
  (a) Add to W_n three literals a, b and c corresponding to the formula $\neg a \vee \neg b \vee \neg c \vee g_i$ (see point (1a) in the construction of W_n). But this would imply $a, b, c \in \{l \mid l \in L_\pi\}$, i.e., L_π is not a solution of π.
  (b) Add to W_n three literals $\overline{a}$, $\overline{b}$, $\overline{c}$ corresponding to formula $\neg \overline{a} \vee \neg \overline{b} \vee \neg \overline{c} \vee g_i$ (see point (1b) in the construction of W_n). But this would imply $\overline{a}, \overline{b}, \overline{c} \in \{\overline{l} \mid l \notin L_\pi\}$, i.e., L_π is not a solution of π.

($\Rightarrow$) Let us assume that $\langle D_n, W_n \rangle \nvdash_{SK} q_\pi$. As a consequence, there exists an extension E of $\langle D_n, W_n \rangle$ such that $E \not\models q_\pi$.

Let L_π be the set $\{l \mid \frac{:l}{l} \in GD(E, \langle D_n, W_n \rangle)\}$. Since E is an extension, for each $l \in L$ it holds that:

(1) either $l \in E$ or $\overline{l} \in E$, since E is maximal;

(2) not both $l \in E$ and $\overline{l} \in E$, since E is consistent.

Since $E \not\models q_\pi$, and $\neg a \vee \neg b \vee \neg c \vee g_i$ belongs to W_n there is no clause $a \vee b \vee c$ in π such that all of $a, b, c \in L_\pi$.

Analogously, it's impossible that all of $a, b, c \notin L_\pi$, otherwise $\overline{a}, \overline{b}, \overline{c} \in L_\pi$, hence $E \models g_i$ because of clause $\neg\overline{a} \vee \neg\overline{b} \vee \neg\overline{c} \vee g_i$ of W_n. Therefore, L_π is a solution of π.

This completes the proof for positive disjunctive queries. For negative disjunctive queries, just substitute every occurrence of an atom $g \in G$ with $\neg g$ in W_n and q_π. □

Theorem 14 has the following corollary, which extends the work of Stillman in [43].

Corollary A.4. *Given a Horn theory W, a PFPNU set of defaults D and a positive literal l, deciding whether $\langle D, W\rangle \vdash_{\mathrm{SK}} l$ is* co-NP*-complete.*

Proof (*Sketch*). Let π be an instance of NOT-ALL-EQUAL-3SAT, L the atoms used in π, $\overline{L}$ a set of new atoms one-to-one with atoms of L, and z an atom not in L. Define W and D as follows:

(1) For each three-atom clause $a \vee b \vee c \in \pi$ there are two clauses in W:
   (a) $\neg a \vee \neg b \vee \neg c \vee z$;
   (b) $\neg\overline{a} \vee \neg\overline{b} \vee \neg\overline{c} \vee z$.

(2) For each atom $l \in L$ there is a clause in W: $\neg l \vee \neg\overline{l}$.

(3) For each atom $l \in L$ there are two defaults in D:
   (a) $\frac{:l}{l}$;
   (b) $\frac{:\overline{l}}{\overline{l}}$.

Note that $\langle D, W\rangle$ can be constructed in time polynomial in π, W is a Horn theory and D is PFPNU.

It follows from the proof of Theorem 14 that π is a "yes" instance of NOT-ALL-EQUAL-3SAT iff $\langle D, W\rangle \not\vdash_{\mathrm{SK}} z$. □

Theorem 15. *The problem $[\langle D, W\rangle \vdash_{\mathrm{SK}} q, \langle D, W\rangle, q]$, where W is a Horn theory, D is a set of PFPNU defaults, and q is a conjunction of literals, is compilable.*

Proof. First of all, a trivial property of skeptical reasoning in default logic is that $\langle D, W\rangle \vdash_{\mathrm{SK}} \alpha \wedge \beta$ iff $\langle D, W\rangle \vdash_{\mathrm{SK}} \alpha$ and $\langle D, W\rangle \vdash_{\mathrm{SK}} \beta$. The problem can be solved by off-line constructing a table where we store for every atom $l \in L$ whether or not $\langle D, W\rangle \vdash_{\mathrm{SK}} l$. Deciding $\langle D, W\rangle \vdash_{\mathrm{SK}} c_1 \wedge \cdots \wedge c_k$ can be reduced to k look-ups of such a table, and this can obviously be done in polynomial time. □

References

[1] F. Baader and B. Hollunder, Embedding defaults into terminological knowledge representation formalisms, in: *Proceedings Third International Conference on the Principles of Knowledge Representation and Reasoning*, Cambridge, MA (1992) 306–317.

[2] A. Borgida and D.W. Etherington, Hierarchical knowledge bases and efficient disjunctive reasoning, in: *Proceedings First International Conference on the Principles of Knowledge Representation and Reasoning*, Toronto, Ont. (1989) 33–43.

[3] R.J. Brachman, The future of knowledge representation, in: *Proceedings AAAI-90*, Boston, MA (1990) 1082–1092.

[4] M. Cadoli, On the complexity of model finding for nonmonotonic propositional logics, in: A. Marchetti Spaccamela, P. Mentrasti and M. Venturini Zilli, eds., *Proceedings of the Fourth Italian Conference on Theoretical Computer Science* (World Scientific Publishing, Teaneck, NJ, 1992) 125–139.

[5] M. Cadoli, Semantical and computational aspects of Horn approximations, in: *Proceedings IJCAI-93*, Chambery (1993) 39–44.

[6] M. Cadoli, F.M. Donini, P. Liberatore and M. Schaerf, The size of a revised knowledge base, in: *Proceedings Fourteenth ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems* (1995) 151–162.

[7] M. Cadoli, F.M. Donini and M. Schaerf, Is intractability of non-monotonic reasoning a real drawback?, in: *Proceedings AAAI-94*, Seattle, WA (1994) 946–951; extended version: RAP.09.95 DIS, University of Roma "La Sapienza", Rome (1995).

[8] M. Cadoli, F.M. Donini and M. Schaerf, On compact representations of propositional circumscription, in: *Proceedings Twelfth Symposium on Theoretical Aspects of Computer Science* (1995) 205–216; extended version: RAP.14.95 DIS, University of Roma "La Sapienza", Rome (1995); also: *Theoret. Comput. Sci.* (to appear).

[9] M. Cadoli and M. Lenzerini, The complexity of propositional closed world reasoning and circumscription, *J. Comput. Syst. Sci.* **48** (1994) 255–310.

[10] M. Cadoli and M. Schaerf, A survey of complexity results for non-monotonic logics, *J. Logic Programming* **17** (1993) 127–160.

[11] W.P. Dowling and J.H. Gallier, Linear-time algorithms for testing the satisfiability of propositional Horn formulae, *J. Logic Programming* **1** (1984) 267–284.

[12] D.V. Etherington, Relating default logic and circumscription, in: *Proceedings IJCAI-87*, Milan (1987) 489–494.

[13] S. Even, A. Itai and A. Shamir, On the complexity of timetable and multicommodity flow problems, *SIAM J. Comput.* **5** (1976) 691–703.

[14] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, CA, 1979).

[15] M. Gelfond and H. Przymusinska, Negation as failure: careful closure procedure, *Artif. Intell.* **30** (1986) 273–287.

[16] M. Gelfond, H. Przymusinska and T. Przymusinski, On the relationship between circumscription and negation as failure, *Artif. Intell.* **38** (1989) 75–94.

[17] G. Gogic, H. Kautz, C. Papadimitriou and B. Selman, The comparative linguistics of knowledge representation, in: *Proceedings IJCAI-95*, Montreal, Que. (1995) 862–869.

[18] G. Gottlob, Complexity results for nonmonotonic logics, *J. Logic Comput.* **2** (1992) 397–425.

[19] T. Imielinski, Incomplete deductive databases, *Ann. Math. Artif. Intell.* **3** (1991) 259–294.

[20] D.S. Johnson, A catalog of complexity classes, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science Vol. A* (North-Holland, Amsterdam, 1990) Chapter 2.

[21] R.M. Karp and R.J. Lipton, Some connections between non-uniform and uniform complexity classes, in: *Proceedings Twelfth ACM Symposium on Theory of Computing* (1980) 302–309.

[22] H.A. Kautz, M.J. Kearns and B. Selman, Reasoning with characteristic models, in: *Proceedings AAAI-93*, Washington, DC (1993) 34–39.

[23] H.A. Kautz and B. Selman, Hard problems for simple default logics, *Artif. Intell.* **49** (1991) 243–279.

[24] H.A. Kautz and B. Selman, Forming concepts for fast inference, in: *Proceedings AAAI-92*, San Jose, CA (1992) 786–793.

[25] R. Khardon and D. Roth, Learning to reason, in: *Proceedings AAAI-94*, Seattle, WA (1994) 682–687; extended version: TR-02-94, Aiken Computation Laboratory, Harvard University, Cambridge, MA (1994).

[26] R. Khardon and D. Roth, Reasoning with models, in: *Proceedings AAAI-94*, Seattle, WA (1994) 1148–1153; extended version: TR-01-94, Aiken Computation Laboratory, Harvard University, Cambridge, MA (1994); also: *Artif. Intell.* **87** (1996) 187–213.

[27] H.J. Levesque, Foundations of a functional approach to knowledge representation, *Artif. Intell.* **23** (1984) 155–212.

[28] V. Lifschitz, Closed-world databases and circumscription, *Artif. Intell.* **27** (1985) 229–235.
[29] V. Lifschitz, Computing circumscription, in: *Proceedings IJCAI-85*, Los Angeles, CA (1985) 121–127.
[30] J. McCarthy, Applications of circumscription to formalizing common-sense knowledge, *Artif. Intell.* **28** (1986) 89–116.
[31] J. Minker, On indefinite databases and the closed world assumption, in: *Proceedings Sixth Conference on Automated Deduction*, New York (1982) 292–308.
[32] Y. Moses and M. Tennenholtz, Off-line reasoning for on-line efficiency, *Artif. Intell.* **83** (1996) 229–239.
[33] A. Nerode, R.T. Ng and V.S. Subrahmanian, Computing circumscriptive databases I: theory and algorithms, *Inform. Comput.* **116** (1995) 58–80.
[34] C.H. Papadimitriou, *Computational Complexity* (Addison-Wesley, Reading, MA, 1994).
[35] A. Rajasekar, J. Lobo and J. Minker, Weak generalized closed world assumption, *J. Autom. Reasoning* **5** (1989) 293–307.
[36] R. Reiter, On closed world data bases, in: H. Gallaire and J. Minker, eds., *Logic and Data Bases* (Plenum, New York, 1978) 119–140.
[37] R. Reiter, A logic for default reasoning, *Artif. Intell.* **13** (1980) 81–132.
[38] R. Reiter, Circumscription implies predicate completion (sometimes), in: *Proceedings AAAI-82*, Pittsburgh, PA (1982) 418–420.
[39] J.S. Schlipf, Decidability and definability with circumscription, *Ann. Pure Appl. Logic* **35** (1987) 173–191.
[40] B. Selman, Near-optimal plans, tractability, and reactivity, in: *Proceedings Fourth International Conference on Principles of Knowledge Representation and Reasoning*, Bonn (1994) 521–529.
[41] B. Selman and H.A. Kautz, Knowledge compilation using Horn approximations, in: *Proceedings AAAI-91*, Anaheim, CA (1991) 904–909.
[42] L. Sterling and E. Shapiro, *The Art of Prolog* (MIT Press, Cambridge, MA, 1986).
[43] J. Stillman, It's not my default: the complexity of membership problems in restricted propositional default logics, in: *Proceedings AAAI-90*, Boston, MA (1990) 571–578.
[44] A. van Gelder, K.A. Ross and J.S. Schlipf, The well-founded semantics for general logic programs, *J. ACM* **38** (1991) 620–650.
[45] M.Y. Vardi, The complexity of relational query languages, in: *Proceedings Fourteenth ACM Symposium on Theory of Computing* (1982) 137–146.
[46] M.Y. Vardi, On the complexity of bounded-variable queries, in: *Proceedings Fourteenth ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems* (1995).
[47] A. Yahya and L.J. Henschen, Deduction in non-Horn databases, *J. Autom. Reasoning* **1** (1985) 141–160.
[48] C.K. Yap, Some consequences of non-uniform conditions on uniform classes, *Theoret. Comput. Sci.* **26** (1983) 287–300.