A Theory of Average-Case Compilability in Knowledge Representation

Hubie Chen
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA
hubes@cs.cornell.edu

Abstract

Compilability is a fundamental property of knowledge representation formalisms which captures how succinctly information can be expressed. Although many results concerning compilability have been obtained, they are all "worst-case" results. We develop a theory of average-case compilability which allows for the formal comparison and classification of knowledge representation formalisms "on average."

1 Introduction

By now, a multitude of knowledge representation formalisms have been proposed and studied in the literature, for example, propositional logic, default logic, and circumscription – to name a few. The comparison of these formalisms has been a major research theme over the past decade. In particular, many results have been obtained on the computational complexity of inference and model checking – core reasoning tasks associated with each formalism. The fundamental property of *compilability*, which measures how efficiently or succinctly a formalism represents knowledge, has also been studied.[1] There is now a rich body of results concerning compilability; however, all of these results address the "worst-case", and are thus susceptible to the complaint that they do not address "average" or "typical" compilability. The main contribution of this paper is a theoretical framework providing a language and tools for comparing and classifying the compilability of formalisms *on average*. Our framework is built on notions and insights from the theory of compilability clases due to Cadoli et al. [2000a; 2000b] and the theory of average-case time complexity due to Levin [1986].

1.1 Background

Compilability. Informally, a formalism A is compilable to a formalism B if for every knowledge base x in formalism A, there is a knowledge base in formalism B representing the same information as x with size polynomial in the length of x.[2] Intuitively, this means that formalism B is at least as space efficient as formalism A: whatever can be expressed in formalism A can be expressed, with about the same level of succinctness, in formalism B.

A number of papers compared the compilability of different formalisms [Kautz *et al.*, 1995; Gogic *et al.*, 1995; Khardon and Roth, 1996; Cadoli *et al.*, 1996; 1997; 1999; 2000a; 2000b]. In many cases, these papers rigorously prove that one formalism B is strictly more succinct than another formalism A – that is, A is compilable to B, but B is not compilable to A. This means that there is *no* way to translate knowledge bases in B to knowledge bases in A, unless the translation is allowed to increase the size of knowledge bases by a super-polynomial amount. In other words, any translation of knowledge bases in B to knowledge bases in A is *inherently exponential* in size.[3] Observe that this statement has the same flavor as the famous "P does not equal NP" conjecture from classical complexity theory. This conjecture holds that no algorithm can solve an NP-complete problem, unless the algorithm is allowed to take super-polynomial time on some inputs – or, put differently, any algorithm solving a NP-complete problem is *inherently exponential* in time.

Notice that the above definition of "formalism A is compilable to formalism B" does not take into account the difficulty or complexity of computing the translation between knowledge bases in A and knowledge bases in B. This non-uniformity is a key feature of the definition: the *existence* of a succinct translation is sufficient; an efficiently computable translation is not necessary. It turns out that many proofs of non-compilability – that is, proofs of statements of the form "formalism B is not compilable to formalism A" – rely on results from *non-uniform* complexity theory. Such proofs often are not unconditional, but are contingent upon the widely believed complexity-theoretic assumption that the polynomial hierarchy does not collapse.[4]

[1] In previous work, compilability has also been called *space efficiency* and *succinctness*.

[2] In the technical portion of this paper, this definition will be captured formally by the $\leq_{\mathrm{nu-comp}}$ reducibility.

[3] Here, by "exponential" we mean exceeding every polynomial infinitely often.

[4] The polynomial hierarchy is a collection of complexity classes which includes P, NP, co-NP, and other classes which are in essence generalizations of these three classes. For the intents and purposes of this paper, this assumption can be thought of as being similar to the (perhaps better known) assumption that P does not equal NP. (In

Compilability classes. The initial papers that demonstrated non-compilability results [Kautz *et al.*, 1995; Gogic *et al.*, 1995; Khardon and Roth, 1996; Cadoli *et al.*, 1996; 1997; 1999] were based on *ad hoc* proofs, each of which isolated two particular formalisms and then demonstrated non-compilability of one to the other. In [Cadoli *et al.*, 2000a; 2000b], new complexity classes measuring compilability, called *compilability classes*, were introduced; for every classical complexity class C, it is possible to define a compilability class analog of C. The theory of compilability classes made it possible to systematize proofs of non-compilability much in the way classical complexity theory makes it possible to systematize proofs of intractability. A proof that a language is NP-complete is demonstration that there is no polynomial-time algorithm for the language, and that the language has the same time complexity (up to a polynomial) as all other NP-complete languages. Likewise, a proof that a formalism is complete for the compilability class analog of NP is demonstration that it is not compilable to any formalism in the compilability class analog of P, and that the language is compilable to and from all other formalisms complete for the compilability class analog of NP.[5] (More generally, when C is a class from the polynomial hierarchy, a proof that a formalism is complete for the compilability class analog of C is demonstration that it is not compilable to any formalism in the compilability class analog of C', if C' is below C in the polynomial hierarchy.)

In addition to providing a methodology for comparing formalisms with respect to compilability, the compilability classes capture formally the notion of off-line *preprocessing*. Preprocessing a knowledge base off-line can be of great utility if the resulting, processed knowledge base is in a form that allows for quick, on-line response to queries (and if many queries are expected). Membership of a formalism A in the compilability class analog of P will mean that any knowledge base x of A can be preprocessed into a form that does not unreasonably increase the size of x, but permits queries to x to be processed efficiently (that is, in polynomial time).

The compilability classes are part of a formal framework for discussing compilability, which includes robust notions of reduction and completeness. Not only is this framework extremely appealing from a theoretical point of view, but the compilability classes are rife with *natural complete problems* – the *sine qua non* of complexity classes purporting to be useful in performing problem classification. Indeed, there do not appear to be any knowledge representation formalisms which defy classification as complete for a compilability class. On the downside, many of the existing classification results are quite negative, showing that a formalism is complete for (the compilability class analog of) NP, coNP, or a higher level of the polynomial hierarchy.

fact, if the polynomial hierarchy does not collapse, then P does not equal NP.)

[5] Our discussion presumes that the polynomial hierarchy does not collapse.

1.2 Motivations and Approach

Worst-case versus average-case. Although crisp theoretical results can be obtained concerning compilability and non-compilability, these are *worst-case* notions: non-compilability of formalism B to formalism A implies that there is an infinite family of knowledge bases in B that cannot be succinctly translated into knowledge bases in A. Non-compilability does *not* say anything about the density or frequency of instances from the family of untranslatable knowledge bases. This observation calls into question the real-world utility of studying compilability (as defined above); if formalism B is compilable to formalism A for all but a pathological family of knowledge bases arising infrequently in practice, then formalism B is, for pragmatic purposes, compilable to formalism A. Therefore, a theory which permits formal results of compilability *on average* is necessary. This paper lays the foundations for a theory of average-case compilability.

Notice that our objection to the worst-case nature of non-compilability is not truly novel. It has long been observed that NP-completeness of a language L does not imply hardness of L on typical or real-world instances. Our objection is really this old observation, masquerading in the new context of compilability.

One theory that was developed in response to this old observation is the theory of average-case time complexity (ACTC), initiated by Levin [1986]. Our theory of average-case compilability will be built on a key notion of ACTC – that of "polynomial on average." Moreover, there are useful analogies between our theory and the theory of ACTC. Consequently, we provide a brief overview of ACTC.

Average-case time complexity. By definition, a language L (consisting of strings) is in P if there is an algorithm deciding membership for L in polynomial time. The idea behind ACTC is to relax the requirement in this definition that a suitable algorithm is one that *always* runs in polynomial time; this is done by placing a probability distribution on all strings and allowing an algorithm to take super-polynomial times on unlikely strings.

A language paired with a probability distribution over all strings is called a *distributional language*. While languages are the objects classified by classical complexity theory, distributional languages are the objects classified by ACTC. A distributional language (L, μ) is in the average-case version of P, *average-P*, if there is an algorithm deciding membership for L in time *polynomial on μ-average* – a concept to be discussed formally later in this paper.

There is a vast literature on ACTC; for more information, we recommend the overviews/surveys [Johnson, 1984; Gurevich, 1989; 1991a; 1991b; Impagliazzo, 1995; Wang, 1997; Goldreich, 1997] as starting points.

Average-case compilability. In laying down a theory for average-case compilability, we want to "soften" the definition of "formalism A is compilable to formalism B" by relaxing the requirement that there needs to be a translation of knowledge bases of strictly polynomial size. This is done roughly in analogy to the described development of ACTC. We first define a distributional formalism to be a formalism paired with a probability distribution over all knowledge bases. Then, a distributional formalism (A, μ) is (informally) *compilable on average* to a formalism B if there is a translation of knowledge bases of size polynomial on μ-average.[6]

Using the notion of "compilable on average," we define average-case analogs of compilability classes. These average-case analogs contain distributional formalisms, in contrast to the compilability classes themselves, which contain pure formalisms. Intuitively, membership of a distributional formalism (A, μ) in the average-case compilability class analog of P will mean the following: any knowledge base x of A can be preprocessed in a way that tends not to increase the size of x by more than a polynomial (with respect to μ), and that allows for rapid processing of queries to x.

2 Preliminaries

In this section, we present notation and assumptions that will be used throughout the paper.

We assume Σ to be a fixed finite alphabet which is used to form strings. We will at times assume that pairs of strings (that is, elements of $\Sigma^* \times \Sigma^*$) are represented as strings (that is, elements of Σ^*); when this assumption is made, we assume that the representation is via a pairing function $\langle \cdot, \cdot \rangle$ such that the length of $\langle x, y \rangle$ is linear in $|x| + |y|$. For a string $x \in \Sigma^*$, we let $|x|_u$ denote $1^{|x|}$, that is, the length of x written in unary notation.

We assume that the reader has familiarity with basic notions of computational complexity theory – in particular, the classes of the polynomial hierarchy (P, NP, $\mathsf{co\text{-}NP}$, Σ_2^p, Π_2^p, etc.) and the polynomial many-one reduction $\leq_m^p$ [Balcázar *et al.*, 1995]. A *language* is a subset of Σ^*, that is, a set of strings. A *complexity class* is a set of languages. When C is a complexity class and $\leq$ is a reduction, we say that C is *compatible* with $\leq$ if for all languages A and B, $A \leq B$ and $B \in C$ imply that $A \in C$. We will assume throughout that every complexity class C is compatible with the polynomial many-one reduction $\leq_m^p$. We say that a language B is complete for a complexity class C under $\leq$ reductions if $B \in C$ and for all $A \in C$, $A \leq B$.

A function $f : \Sigma^* \to \Sigma^*$ is *polynomial-size* if there exists a polynomial p such that for all $x \in \Sigma^*$, $|f(x)| \leq p(|x|)$. A function $f : \Sigma^* \to \Sigma^*$ is *polynomial-time computable* if there exist a polynomial p and a Turing machine M such that for all $x \in \Sigma^*$, the Turing machine M, on input x, produces $f(x)$ in time less than $p(|x|)$.[7]

Definition 2.1 *A* knowledge representation formalism *(KRF) is a subset of* $\Sigma^* \times \Sigma^*$. *When F is a KRF and $x \in \Sigma^*$, we let F_x denote the set* $\{y \in \Sigma^* : (x, y) \in F\}$.

Intuitively, each $x \in \Sigma^*$ can be thought of as a knowledge base (KB), representing the information F_x. The following are examples of KRFs which capture model checking and inference for 3-SAT formulas.

$$\mathsf{Propositional\text{-}Logic\text{-}MC} \stackrel{\text{def}}{=}$$
$$\{(x, y) : x \text{ is a 3-SAT formula and } y \text{ is a model of } x\}$$

$$\mathsf{Clause\text{-}Inference} \stackrel{\text{def}}{=}$$
$$\{(x, y) : x \text{ is a 3-SAT formula}, y \text{ is a 3-clause, and } x \models y\}$$

Notice the generality of the definition of a KRF; the knowledge represented (the various F_x) may be models, formulas, or some altogether different combinatorial structures.

3 Compilability classes and reductions

This section reviews the theory of compilability classes. There are two different types of compilability classes, uniform and non-uniform, but our focus is on the latter, for reasons discussed below. *We emphasize that none of the definitions, theorems, or insights in this section are our own, but rather, are due to [Cadoli* et al.*, 2000a; 2000b], on which our presentation is based.*

The formal definition of non-uniform compilability may by itself look non-intuitive, so we first attempt to describe some of the ideas behind this definition, before giving the actual definition. Intuitively, we want to say that a KRF F is compilable if membership queries $y \in F_x$ can be decided efficiently after the KB x is preprocessed into a new KB $f(x)$. We want to constrain the size of the new KB $f(x)$; otherwise, it may be prohibitively large to store. We arrive at a candidate definition of compilability: say that a KRF F is in $\mathsf{comp}\text{-}C$ if for some polynomial-size function f and a second KRF $F' \in C$, the property

$$\text{for all pairs } (x, y) \in \Sigma^* \times \Sigma^*, (x, y) \in F \text{ if and only if } (f(x), y) \in F'$$

holds. Notice that the class C constrains the difficulty of deciding a post-processing query $y \in F'_{f(x)}$; of course, such queries can be decided efficiently when $C = \mathsf{P}$. However, to permit fine classification of KRFs, we leave C as a parameter to the definition of $\mathsf{comp}\text{-}C$.

The above definition misses one important detail – non-uniformity. If F is a KRF such that F_x is always a finite set, F can never be complete for $\mathsf{comp}\text{-}C$ when C is a class of the polynomial hierarchy above P. At the same time, there are KRFs with this "finiteness" property that are provably *not* in $\mathsf{comp\text{-}P}$. As an example, $\mathsf{Clause\text{-}Inference}$ is in $\mathsf{comp\text{-}co\text{-}NP}$, but is neither $\mathsf{comp\text{-}co\text{-}NP}$-complete nor in $\mathsf{comp\text{-}P}$.[8] Hence, the classes $\mathsf{comp}\text{-}C$ fail to capture the compilability of some very natural KRFs. (For more information on these matters, as well as proofs of the claims, we refer the reader to [Cadoli *et al.*, 2000a].) To resolve this issue, we allow the *length* of a query to be known to the compilation mapping f.

Definition 3.1 *($\mathsf{nu\text{-}comp}\text{-}C$) Let C be a complexity class. A KRF F belongs to $\mathsf{nu\text{-}comp}\text{-}C$ if there exists a binary polynomial-size function $f : \Sigma^* \times 1^* \to \Sigma^*$ and a KRF F' in C such that the following property holds:*

[6] In the technical portion of this paper, this definition will be captured formally by the $\leq_{\mathrm{dist-nu-comp}}$ reducibility.

[7] Notice that every polynomial-time computable function is polynomial-size, but not every polynomial-size function is polynomial-time computable.

[8] This discussion presumes that the polynomial hierarchy does not collapse.

for all pairs $(x, y) \in \Sigma^* \times \Sigma^*, (x, y) \in F$ *if and only if* $(f(x, |y|_u), y) \in F'$.

The fact that the compilation mapping may use the length of y may seem a bit strange; after all, we wish to capture, after preprocessing on a knowledge base x, the difficulty of answering queries of the form $y \in F_x$. However, it is reasonable to assume that we will never be interested in such a query when the length of y greatly exceeds that of x – since such a query requires a large amount of time to even write down. Under this assumption, Definition 3.1 is equivalent to the "uniform" definition given above. (Formally, if there exists a polynomial p such that $(x, y) \in F$ implies that $|y| \leq p(|x|)$, then F is in nu-comp-C if and only if F is in comp-C.)

The following notion of reduction, associated with the nu-comp-C classes, allows one to compare the compilability of different KRFs.

Definition 3.2 *(nu-comp reducibility) A KRF F is nu-comp reducible to a KRF F' (denoted by $F \leq_{\mathrm{nu-comp}} F'$) if there exist binary polynomial-size functions $f_1, f_2 : \Sigma^* \times 1^* \to \Sigma^*$ and a binary polynomial-time computable function $g : \Sigma^* \times \Sigma^* \to \Sigma^*$ such that the following property holds:*

for all pairs $(x, y) \in \Sigma^* \times \Sigma^*, (x, y) \in F$ *if and only if* $(f_1(x, |y|_u), g(f_2(x, |y|_u), y)) \in F'$.

Theorem 3.3 *The nu-comp reduction is transitive and is compatible with the class nu-comp-C (for every complexity class C).*

We note that Clause-Inference is nu-comp-co-NP-complete, under nu-comp reductions.

Finally, we observe that the nu-comp-C analog of the polynomial hierarchy does not collapse.

Theorem 3.4 *If C_1 and C_2 are classes of the polynomial hierarchy such that C_2 is higher than C_1, then nu-comp-C_1 is properly contained in nu-comp-C_2 (under the assumption that the polynomial hierarchy does not collapse).*

4 Average-case compilability

In this section, we present our new theory of average-case compilability.

Definition 4.1 *A* probability distribution μ *is a real-valued function from $\Sigma^* \times 1^*$ to $[0, 1]$ such that $\sum_{(x,l) \in \Sigma^* \times 1^*} \mu(x, l) = 1$.*

The nu-comp-C classes are used to classify and compare KRFs; our new classes will be used to classify and compare what we call distributional KRFs.

Definition 4.2 *A* distributional KRF (DKRF) *is a pair (F, μ) consisting of a KRF F and a probability distribution μ.*

We now define the notion of "polynomial on average." This definition is exactly that used in the theory of average-case time complexity (up to appropriate changes of the domain and range of the functions whose size we wish to measure).

Definition 4.3 *Let μ be a probability distribution. A function $f : \Sigma^* \times 1^* \to \Sigma^*$ is of size* polynomial on μ-average *if there exists an $\epsilon > 0$ such that $\sum_{(x,l) \in \Sigma^* \times 1^*} |f(x, l)|^\epsilon |\langle x, l\rangle|^{-1} \mu(x, l) < \infty$.*

We are now ready to define our new "average-compilability" classes; formally, each class is a set of DKRFs. The definition can be viewed as a relaxation of Definition 3.1. The difference is that the translation mapping $f : \Sigma^* \times 1^* \to \Sigma^*$ need no longer be of strictly polynomial size, but is now only required to be of size polynomial on average.

Definition 4.4 *(avg-nu-comp-C) Let C be a complexity class. A DKRF (F, μ) belongs to avg-nu-comp-C if there exists a binary function $f : \Sigma^* \times 1^* \to \Sigma^*$ of size polynomial on μ-average and a KRF F' in C such that the property of Definition 3.1 holds.*

There is an alternative way to define avg-nu-comp-C, in terms of the following type of reduction, which relates DKRFs to KRFs.

Definition 4.5 *(dist-nu-comp reducibility) A DKRF (F, μ) is dist-nu-comp reducible to a KRF F' (denoted by $F \leq_{\mathrm{dist-nu-comp}} F'$) if there exist binary polynomial-size functions $f_1, f_2 : \Sigma^* \times 1^* \to \Sigma^*$ and a binary polynomial-time computable function $g : \Sigma^* \times \Sigma^* \to \Sigma^*$ such that $\langle f_1, f_2\rangle$ is of size polynomial on μ-average*[9] *and the property of Definition 3.2 holds.*

Theorem 4.6 *Suppose that (F, μ) is a DKRF and that C is a complexity class. The DKRF (F, μ) is in avg-nu-comp-C if and only if $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F'$ for some F' in nu-comp-C.*

This theorem has an important corollary: if nu-comp-C has a complete KRF F', then the DKRFs contained in avg-nu-comp-C are exactly those which reduce to F'. In order to establish the corollary, the following lemma is needed.

Lemma 4.7 *Suppose that (F, μ) is a DKRF and that F' and F'' are KRFs. If $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F'$ and $F' \leq_{\mathrm{nu-comp}} F''$, then $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F''$.*

Roughly, Lemma 4.7 can be viewed as a proof of transitivity: if (F, μ) reduces to F' and F' reduces to F'', then (F, μ) reduces to F'' (under the right notions of reduction).

Corollary 4.8 *Suppose that (F, μ) is a DKRF, C is a complexity class and the KRF F' is nu-comp-C-complete under nu-comp reductions. Then, the DKRF (F, μ) is in avg-nu-comp-C if and only if $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F'$.*

It is worth noting here that the class nu-comp-C has a complete KRF under nu-comp reductions whenever the underlying complexity class C has a complete language under $\leq^p_m$ reductions [Cadoli *et al.*, 2000a].

We now give a notion of reduction for the comparison of DKRFs. When μ and ν are probability distributions, we say that ν *dominates* μ if there exists a polynomial p such that for all $(x, l) \in \Sigma^* \times 1^*$, $\mu(x, l) \leq p(|\langle x, l\rangle|)\nu(x, l)$.

Definition 4.9 *(avg-nu-comp reducibility) A DKRF (F, μ) is avg-nu-comp reducible to a DKRF (F', μ') if there exist a nu-comp reduction (f_1, f_2, g) from F to F' and a probability distribution μ_1 dominating μ such that $\mu'(y, m) \geq \sum \mu_1(x, l)$, where the sum is over all (x, l) such that [$f_1(x, l) = y$ and there exists $z \in \Sigma^*$ with $|z| = l$ and $|g(f_2(x, l), z)| = m$].*

[9] By $\langle f_1, f_2\rangle$, we denote the function which maps a pair $(x, y) \in \Sigma^* \times 1^*$ to $\langle f_1(x, y), f_2(x, y)\rangle$.

Theorem 4.10 *The avg-nu-comp reduction is transitive and is compatible with the class* avg-nu-comp-C *(for every complexity class* C*).*

5 Discussion

We now discuss some of the considerations behind the definitions in the previous section. When $\mu : \Sigma^* \times 1^* \to [0, 1]$ is a probability distribution and S is a subset of $\Sigma^* \times 1^*$, let $(\mu|S)$ denote the conditional distribution of μ on S, that is, the function defined on $(x, l) \in S$, with value equal to $\mu(x, l)/[\sum_{s \in S} \mu(s)]$.

• *Why is Levin's notion of polynomial-on-average used instead of the naive formulation of expected polynomial size?*

According to the naive formulation, a function $f : \Sigma^* \times 1^* \to \Sigma^*$ has expected polynomial size (with respect to μ) if there exists $k \geq 1$ such that for all n,

$$\sum_{|\langle x,l \rangle|=n} |f(x, l)|(\mu(x, l)|\{(x', l') : |\langle x', l' \rangle| = n\}) = O(n^k).$$

It is well known that this notion of expected polynomial is *not* closed under polynomials (see for example [Goldreich, 1997]). For instance, there exist functions $f, f' : \Sigma^* \times 1^* \to \Sigma^*$ and a distribution μ such that $|f'(x, l)| = |f(x, l)|^2$ and f has expected polynomial size, but f' fails to have expected polynomial size.

In our average-case compilability theory, this "lack of closure under polynomials" problem manifests itself in the following way. Suppose that, using the alternative definition of avg-nu-comp-C given in Theorem 4.6, the KRF F' is witness to the membership of DKRF (F, μ) in avg-nu-comp-C, that is, $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F'$. Surely, if F'' is at least as space efficient as F' (that is, $F' \leq_{\mathrm{nu-comp}} F''$), then one expects that F'' would also witness the membership of (F, μ) in avg-nu-comp-C; this is the content of Lemma 4.7, on which Corollary 4.8 relies. However, if expected polynomial size is used instead of polynomial on average in Definition 4.3, Lemma 4.7 breaks down – then, there would exist a DKRF (F, μ) and KRFs F' and F'' such that $(F, \mu) \leq_{\mathrm{dist-nu-comp}} F'$, $F' \leq_{\mathrm{nu-comp}} F''$, and (F, μ) does *not* $\leq_{\mathrm{dist-nu-comp}}$-reduce to F''.

• *Why does a DKRF have one probability distribution – over all elements of* $\Sigma^* \times 1^*$ *– as opposed to an ensemble of distributions, each of which is over some finite subset of* $\Sigma^* \times 1^*$*?*

Defining a DKRF to be a KRF paired with an ensemble of distributions (each of which is, say, defined on a different string length) may seem more natural than the given definition, where there is only one distribution, on all strings.

However, the literature on ACTC contains many alternative formulations of "polynomial on average" (equivalent to that given in Definition 4.3) and sufficient conditions for a function to be polynomial on average. (For example, there is a different formulation of Definition 4.3 in terms of ensembles of distributions, each having finite support, in [Impagliazzo, 1995].) These give rise to alternative formulations of avg-nu-comp-C, and so the particular characterizations we give for avg-nu-comp-C (in Definition 4.4 and Theorem 4.6) were, in some sense, chosen over provably equal characterizations on purely aesthetic grounds.

• *Why is a probability distribution over* $\Sigma^* \times 1^*$ *as opposed to over* Σ^**? After all, the knowledge bases are represented by elements of* Σ^**, and an element of* $\Sigma^* \times 1^*$ *is just a "slice" of a knowledge base.*

Let F be a KRF conforming to the "bounded-query-length" assumption discussed in Section 3, that is, suppose that there exists a polynomial p such that $(x, y) \in F$ implies $|y| \leq p(|x|)$. Then, a distribution over Σ^* naturally induces a distribution over $\Sigma^* \times 1^*$ for the padded version of F defined as $F' = \{(x, z) : z = y\Box^k, |z| = p(|x|), (x, y) \in F\}$ (where $\Box$ is an extra padding symbol), which has the property that all strings in F'_x have the same length, for any x. Note that all of the example KRFs in this paper already have this property.

6 Example: model checking for circumscription

In this section, we illustrate the use of our new theory by showing that model checking for circumscription on 3-SAT formulas (formalized below and denoted by Circumscription-MC(d)) is in avg-nu-comp-P, under a natural probability distribution where formulas are generated by including each clause independently with identical probability. Since Circumscription-MC(d) is nu-comp-co-NP-hard,[10] this result gives a natural DKRF which is contained in avg-nu-comp-P, but which has a KRF which is provably *not* in nu-comp-P (Theorem 3.4).

Suppose that ϕ is a 3-SAT formula over $V_n \stackrel{\mathrm{def}}{=} \{v_1, \ldots, v_n\}$, and S is a subset of V_n. We say that a model $a : V_n \to \{0, 1\}$ of ϕ is a *S-minimal model* (of ϕ) if for all other models $b : V_n \to \{0, 1\}$ of ϕ, $[b(s) \leq a(s)$ for all $s \in S]$ implies that $[b(s) = a(s)$ for all $s \in S]$.[11] In addition, a model $a : S \to \{0, 1\}$ of ϕ is said to be a *minimal model* if it is the restriction of a S-minimal model $a : V_n \to \{0, 1\}$ (of ϕ).

We define Circumscription-MC(d) to be the set

$$\{(\phi, a) : \phi \text{ is a 3-SAT formula on variable set on } V_n \text{ and } a : \{v_1, \ldots, v_{\lfloor dn \rfloor}\} \to \{0, 1\} \text{ is a minimal model of } \phi\}.$$

Let W_n denote the set $\{\phi : \phi$ is a 3-SAT formula on V_n and $a : \{v_1, \ldots, v_{\lfloor dn \rfloor}\} \to \{0, 1\}$ is an assignment$)\}$, which is the set of all syntactically well-formed pairs that may or may not be in Circumscription-MC(d).

Theorem 6.1 *Let* $\nu_{n,c}$ *denote the distribution on 3-SAT formulas over* V_n *where a formula is generated by including each of the* $8\binom{n}{3}$ *3-clauses independently with probability* c/n^2 *(for all real numbers* $c > 0$ *and natural numbers* $n \geq 1$*). Let* $\mu_{c,d}$ *(with* $c > 0$ *and* $d \in (0, 1)$*) be any distribution such that*

- $(\mu_{c,d}|W_n) = \nu_{n,c}$ *for all* n*, and*
- $\mu_{c,d}(\phi, y) = 0$ *if* ϕ *is a 3-SAT formula on* V_n *and* $|y| \neq \lfloor dn \rfloor$.

[10] This can be shown using a result in [Gogic *et al.*, 1995].

[11] Here, $\leq$ denotes the usual total ordering on $\{0, 1\}$ where $0 \leq 1$.

If the inequality $2^{1-d}e^{-c/6}(2 - e^{-c/2})^d < 1$ *holds, then* $($*Circumscription-MC*$(d), \mu_{c,d})$ *is in avg-nu-comp-*P*.*

In particular, $($*Circumscription-MC*$(0.9), \mu_{3.73,0.9})$ *is in avg-nu-comp-*P*.*

This theorem is established using techniques from probabilistic combinatorics. It is worth noting that at $c = 3.73$, the expected number of satisfying assignments that a random 3-SAT formula has is exponential.

7 Conclusions and future work

In [Papadimitriou, 1996], Papadimitriou writes:

> The ultimate and most conclusive criterion for comparing knowledge representation formalisms is to compare their expressive power not on arbitrary sets of models, but on the "interesting" sets of models, the ones that come up in the "real world." We still hope that a meaningful and convincing formulation of this important problem may be possible.

In this paper, we presented a robust and flexible theory of average-case compilability in which the intuitive notion of "interesting sets of models" can be formalized as a DKRF (Definition 4.2), DKRFs can be classified (Definition 4.4), and DKRFs can be compared (Definition 4.9). We illustrated the use of this theory by showing that, under a natural probability distribution, model checking for circumscription is compilable to the analog of the complexity class P, in our theory.

There are many open questions for future investigation; to conclude the paper, we list a few.

- Can additional DKRFs be classified as being inside avg-nu-comp-P? We conjecture in particular that Circumscription-MC(d) – along with the "standard" 3-SAT probabilistic model with $p = c/n^2$ for a sufficiently *low* c – is in avg-nu-comp-P.
- Are there DKRFs complete for avg-nu-comp-C (where C is from the polynomial hierarchy) under the avg-nu-comp reduction, or some other notion of reduction compatible with the avg-nu-comp-C classes?
- Can any questions concerning the new avg-nu-comp-C classes defined here be related to better known complexity-theoretic hypotheses?

References

[Balcázar *et al.*, 1995] J. L. Balcázar and J. Díaz and J. Gabarró. *Structural Complexity I*. Springer-Verlag, Berlin, 1995.

[Cadoli *et al.*, 2000a] Marco Cadoli, Francesco M. Donini, Paolo Liberatore, and Marco Schaerf. Preprocessing of intractable problems. *Information and Computation*, 176(2):89–120, 2000.

[Cadoli *et al.*, 1999] Marco Cadoli, Francesco M. Donini, Paolo Liberatore, and Marco Schaerf. The size of a revised knowledge base. *Artificial Intelligence*, 115(1), 25-64.

[Cadoli *et al.*, 2000b] Marco Cadoli, Francesco M. Donini, Paolo Liberatore, and Marco Schaerf. Space Efficiency of Propositional Knowledge Representation Formalisms. *Journal of Artificial Intelligence Research*, 13:1–31, 2000.

[Cadoli *et al.*, 1997] Marco Cadoli, Francesco M. Donini, Marco Schaerf, and Riccardo Silvestri. On Compact Representations of Propositional Circumscription. In *Theoretical Computer Science*, 182:183-202.

[Cadoli *et al.*, 1996] Marco Cadoli, Francesco M. Donini, and Marco Schaerf. Is Intractability of Non-Monotonic Reasoning a Real Drawback? *Artificial Intelligence*, 88:215-251.

[Gogic *et al.*, 1995] Goran Gogic, Henry Kautz, Christos Papadimitriou, and Bart Selman. The Comparative Linguistics of Knowledge Representation. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 862–869, Montreal, Canada, 1995.

[Goldreich, 1997] Oded Goldreich. Notes on Levin's Theory of Average-Case Complexity. *Electronic Colloquium on Computational Complexity (ECCC)* 4(58): 1997.

[Gurevich, 1991a] Yuri Gurevich. Average case completeness. *Journal of Computer and System Sciences*, 42:346-398, 1991.

[Gurevich, 1991b] Yuri Gurevich. Average case complexity. In *Proceedings of the 18th International Colloquium on Automata, Languages and Programming*, volume 510 of Lecture Notes in Computer Science, Springer, pages 615-628, 1991.

[Gurevich, 1989] Yuri Gurevich. The challenger-solver game: variations on the theme of P =? NP. *EATCS Bulletin*, pages 112-121, 1989.

[Impagliazzo, 1995] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the 10th Conference on Structure in Complexity Theory*, IEEE Computer Society Press, pages 134-147, 1995.

[Johnson, 1984] David Johnson. The NP-completeness column: an ongoing guide. *Journal of Algorithms*, 5:284-299, 1984.

[Kautz *et al.*, 1995] Henry Kautz, Michael Kearns, and Bart Selman. Horn approximations of empirical data. *Artificial Intelligence*, 75:129-145.

[Khardon and Roth, 1996] Roni Khardon and Dan Roth. Reasoning with Models. *Artificial Intelligence* 87:187–213, November 1996.

[Levin, 1986] Leonid Levin. Average case complete problems. *SIAM Journal on Computing*, 15:285-286, 1986.

[Papadimitriou, 1996] Christos Papadimitriou. The Complexity of Knowledge Representation. In *Proceedings of the Eleventh Annual IEEE Conference on Computational Complexity*, pages 244-248, Philadelphia, Pennsylvania, May 1996.

[Wang, 1997] Jie Wang. Average-case computational complexity theory. (L. Hemaspaandra and A. Selman, eds.), Complexity Theory Retrospective II, Springer-Verlag, pages 295-328, 1997.