

A Logic for Approximate First-Order Reasoning

Frédéric Koriche

LIRMM, UMR 5506, Université Montpellier II CNRS
161, rue Ada. 34392 Montpellier Cedex 5, France
`koriche@lirmm.fr`

Abstract. In classical approaches to knowledge representation, reasoners are assumed to derive all the logical consequences of their knowledge base. As a result, reasoning in the first-order case is only semi-decidable. Even in the restricted case of finite universes of discourse, reasoning remains inherently intractable, as the reasoner has to deal with two independent sources of complexity: *unbounded chaining* and *unbounded quantification*. The purpose of this study is to handle these difficulties in a logic-oriented framework based on the paradigm of *approximate reasoning*. The logic is semantically founded on the notion of resource, an accuracy measure, which controls at the same time the two barriers of complexity. Moreover, a stepwise technique is included for improving approximations. Finally, both sound approximations and complete ones are covered. Based on the logic, we develop an approximation algorithm with a simple modification of classical instance-based theorem provers. The procedure yields approximate proofs whose precision increases as the reasoner has more resources at her disposal. The algorithm is interruptible, improvable, dual, and can be exploited for anytime computation. Moreover, the algorithm is flexible enough to be used with a wide range of propositional satisfiability methods.

Keywords: approximate reasoning, first-order logic, multi-modal logics, resource-bounded algorithms.

1 Introduction

A widely accepted framework for studying intelligent agents is the knowledge representation approach. Knowledge is described in some logical formalism and stored into a knowledge base. This component is coupled with an inference algorithm, the reasoner, which determines whether a given query is entailed from the knowledge base. One of the main challenges of knowledge representation lies in the computational tradeoff between expressiveness of representation languages and their complexity [21]. On the one hand, knowledge needs to be represented in a very expressive language, such as first-order logic and, on the other, reasoning has to be very efficient, especially for knowledge bases of the size required for human level common-sense. Unfortunately, it is well-known that first-order reasoning is only *semi-decidable*. In other words, if the base and the query are represented in first-order logic, there is no guaranteed way to determine in finite time whether the query is entailed, or not, from the knowledge base.

Since real agents are constrained by finite resources, it seems appropriate to examine first-order reasoning in the setting of *finite universes of discourses*. This assumption has received increasing attention in the communities of database systems [23, 26], planning [11, 12] and theorem proving [13, 25, 28, 29]. This can be expressed by a domain closure axiom or, less restrictively, through a constraint expressing a finite upper limit on the cardinality of the domain of any interpretation. In this setting, every first-order formula can be rewritten to a finite “propositional” formula which, however, is in general exponentially larger. Each atom with m free variables gives rise to (n^m) ground instances for an universe of size n . As a result the complexity of first-order reasoning is (at most) exponentially higher than the complexity of propositional reasoning. Intuitively, this means that a first-order reasoner is confronted with two sources of complexity: *unbounded chaining* and *unbounded quantification*. The first one is related to propositional entailment, which is known to be intractable, while the second one is related to the exponential number of ground instances generated by a first-order formula. So, even in finite universes of discourse, first-order reasoning remains very much demanding from a computational point of view.

Approximate reasoning is an approach advocated in many areas of artificial intelligence to deal with the computational intractability of problems. The motivation behind this paradigm stems from the fact that practical agents have limited time to solve problems and limited memory to remember information. As suggested by Lakemeyer in [16], an approximate reasoning system provides something of middle ground between what is explicit or evident and can be retrieved using few resources and what is implicit and should be inferred given enough time and memory. In case the answer of the reasoner is not satisfactory, one can still decide to continue reasoning. The simplest way is to switch to a conventional reasoning technique. A better way is to use the resource-bounded reasoner to produce better and better answers in a cumulative fashion.

Many such inference algorithms have been proposed, and these algorithms are generally reasonably easy to understand procedurally. For example, one may take an existing theorem prover and bound its execution time and memory in some way. However, understanding inference procedurally is no substitute for understanding what sort of “semantics” and “axiomatrics” underlie the inference. This kind of deeper understanding is the domain of *logic*. A logic for a resource-bounded reasoner gives a clear picture to the notion of resource and tells us what the inference algorithm is, and is not, able to deduce from its knowledge base.

There have been a number of attempts at devising logics for approximate reasoning either proof-theoretically or model-theoretically. On the proof-theoretic side, for example, Dalal defines tractable forms of reasoning by eliminating certain inference rules from propositional logic [5]. The starting point of its framework relies on unit resolution which is tractable but weaker than propositional deduction. Based on this inference rule, the author obtains better and better approximations by imposing incremental bounds on the size of clauses used as lemmas. This technique has been further studied, for example, in [6, 10].

On the model-theoretic side, one of the first framework proposed in the literature is that of Levesque [20]. Based on a multi-valued logic, especially a fragment of Belnap’s relevance logic [1], the author introduces a notion of inference which is weaker than propositional deduction and that captures a tractable form of reasoning. This study has been further extended by Schaerf and Cadoli in [24]. Their framework considers a subset S of the propositional variables, which are deserved a classical interpretation, while the rest of propositions is given a multi-valued interpretation. By increasing the parameter S , the reasoner can regain full logical deduction in an incremental fashion. This treatment of approximate reasoning has been applied to a wide range of reasoning problems [2, 14, 15, 22].

To the best of our knowledge, most of the studies in approximate reasoning have concentrated to propositional logic. On the proof-theoretic camp, Crawford and Etherington in [10] have recently attempted to extend Dalal’s approach to first-order logic but, as they pointed it out, unit resolution alone is undecidable. On the model-theoretic camp, Schaerf and Cadoli have extended their semantics to the description logics $\mathcal{AL}\mathcal{E}$ and $\mathcal{AL}\mathcal{C}$ but these languages remain restricted fragments of first-order logic. A more general framework has been investigated by Lakemeyer in [16, 17]. Based on Levesque’s multi-valued logic, the author defines a inference relation which is weaker than classical first-order implication and that captures a decidable form of reasoning. However, Lakemeyer’s framework is only “one-shot”: if an approximate solution is wrong, the sole thing to do is to switch to a general-purpose reasoning technique. From this perspective, it seems appropriate to pursue investigations in the direction to “improvable” reasoners that would produce better and better solutions in an incremental fashion.

In this paper, we introduce a model-theoretic framework for approximate first-order reasoning. The framework is based on a multi-modal logic which contains a well-founded semantics and a correct and complete axiomatization. To some extent, our logic combines ideas from Schaerf and Cadoli’s approximation technique and Lakemeyer’s system for limited reasoning. In essence, our framework integrates the following features.

- The logic is founded on the notion of *resource*, an accuracy measure which semantically captures bounded approximations of first-order inference. The measure reflects both the quality and the cost of the approximations.
- The framework enables *improvable reasoning*: the quality of approximations is an increasing function of the resources that have been spent.
- The framework covers *dual reasoning*: both sound but incomplete and complete but unsound solutions are returned at any computation step.

The rest of the paper is organized as follows. In section 2, we define the syntax, the semantics, and a sound and complete axiomatization for the logic. In section 3, we investigate the semantical properties of approximate reasoning. In section 4, we show how to transform a traditional instance-based theorem prover into a resource-bounded algorithm which is interruptible, improvable, dual, and that can be exploited for anytime computation [30]. The approximation schema is flexible enough to be used with a wide range of efficient satisfiability methods. Finally, in section 5, we suggest some topics for future research.

2 The Logic

In this section we present a logic, named **AFOR**, for approximate first-order reasoning. We begin to define the syntax, next we examine the semantics in detail, and then we present a sound and complete axiomatization for the logic.

2.1 Syntax

The basic building block of our framework is Levesque’s first-order logic with standard names presented in [19] and further examined by Lakemeyer in [16, 17]. A *first-order signature* consists of denumerable sets P, F, N of symbols called *predicates*, *functions* and *standard names* respectively. Each predicate and function symbol has a fixed *arity* which is defined by the number of its arguments. Function symbols with arity zero are called *constants*. The set of standard names corresponds to the universe of discourse over which quantifiers range.

A *term* is either a variable, a standard name, or a function symbol whose arguments are themselves terms. A *ground term* is a term not containing any variable. A *primitive term* is either a constant or a function symbol whose arguments are standard names. An *atom* is a predicate whose arguments are terms and a *literal* is an atom or its negation. A *primitive atom* (resp. *primitive literal*) is an atom (resp. literal) with standard names as arguments. The sets of primitive atoms and primitive literals generated from the signature are denoted A and L , respectively. A *standard formula* is either an atom or can be obtained by the usual rules for the connectives \neg and \wedge , and the quantifier \forall . Other connectives such as \vee , \supset and \equiv , and the quantifier \exists are defined in the usual way.

We now turn to the concept of resource. The key point behind this notion is to control the two aforementioned sources of complexity of first-order reasoning, namely unbounded chaining and unbounded quantification. To this end, a *resource parameter* is defined as a pair $S = (P_S, N_S)$ where P_S is a finite subset of P and N_S is a nonempty finite subset of N . A parameter can be seen as the collection of primitive atoms and standard names which are relevant for chaining and quantification for a given problem instance. The sets of primitive atoms and primitive literals generated from S are denoted A_S and L_S , respectively. The “empty” parameter, which doesn’t contain any predicate, is denoted S_0 .

A *formula* is either a standard formula or can be obtained by the following rules: if α is a formula, then $\neg\alpha$ is a formula, if α and β are formulas then $\alpha \wedge \beta$ is a formula, and if α is a standard formula and S a resource parameter then $\Box_S \alpha$ is a formula. Notice that the syntax does not allow quantifying-in or nested modalities. The modality \Diamond_S is used as an abbreviation of $\neg\Box_S\neg$. A formula such as $\Box_S \alpha$ is read “the reasoner necessarily infers α given the resources S ”; dually $\Diamond_S \alpha$ is read “the reasoner possibly infers α given the resources S ”.

A *sentence* is a closed formula, a *declaration* is a closed standard formula, and a *ground declaration* is a quantifier free declaration. In the following, sequences of terms are written in vector notion. For example, (t_1, \dots, t_k) is abbreviated as \mathbf{t} . If a formula α contains the free variables x_1, \dots, x_k , then the notation $\alpha[\mathbf{x}/\mathbf{t}]$ will denote the result of replacing each occurrence of x_i by t_i in α .

2.2 Semantics

The semantics of **AFOR** combines ideas from Belnap's four-valued logic [1] with possible world interpretations that allow varying domains of quantification [7].

We first assign semantics to ground terms. To this point, the logic makes the assumption that the universe of discourse is isomorphic to the set of standard names, that is, a ground term is identified with a unique name. A *denotation function* is defined as a mapping d from primitive terms to standard names. A denotation function determines a unique map, also denoted d , on the set of all ground terms, according to the following conditions: $d(n) = n$, where n is a standard name, and $d(f(\mathbf{t})) = d(f(\mathbf{n}))$, where $n_i = d(t_i)$.

We now examine the semantics for all non-logical symbols. Our approach rests on an extension of the standard notion of possible world which we call *valuations*. While worlds use fixed domains of quantification and assign a truth value to every primitive atoms, valuations, in contrast, allow varying domains of quantification and assign truth values to all the literals. In formal terms, a *valuation* is a structure $v = (N_v, L_v, d_v)$, where N_v is a subset of N , L_v is a subset of L , and d_v is a denotation function. A *world* is a valuation w , where N_w is the set of all standard names N and L_w is a subset of L such that for every primitive atom a , $a \in L_w$ if and only if $\neg a \notin L_w$. We say that a valuation v is more *specific* than v' , and write $v \subseteq v'$, if $N_v = N_{v'}$, $L_v \subseteq L_{v'}$ and $d_v = d_{v'}$. We remark that the specificity relation is a partial order on the space of all valuations. Moreover it induces a lattice structure on each subspace of valuations defined over the same domain of quantification and the same denotation function.

The concept of resource parameter S is semantically captured by an accessibility relation on valuations \mathcal{R}_S . Given two valuations v and v' and any resource parameter S , $v' \in \mathcal{R}_S(v)$, if $N_{v'} = N_S$, $L_{v'} \cap L_S = L_v \cap L_S$ and $d_{v'} = d_v$. In other words, $\mathcal{R}_S(v)$ is the set of valuations that share the same domain of quantification N_S , that assign the same truth value as v to each literal in L_S , and that use the same denotation functions. We remark that, for any valuation v , $\mathcal{R}_S(v)$ is a complete lattice under the specificity ordering \subseteq . The smallest and the largest valuations of $\mathcal{R}_S(v)$ are denoted $\cap \mathcal{R}_S(v)$ and $\cup \mathcal{R}_S(v)$, respectively.

We now turn to the semantics of sentences. Since valuations assign independent truth values to literals and their complements, the semantic rules for sentences must define truth support for both sentences and their negation.

$$v \models p(\mathbf{t}) \quad \text{iff} \quad p(\mathbf{n}) \in L_v \text{ and } \mathbf{n} = d_v(\mathbf{t}), \quad (1)$$

$$v \models \neg p(\mathbf{t}) \quad \text{iff} \quad \neg p(\mathbf{n}) \in L_v \text{ and } \mathbf{n} = d_v(\mathbf{t}), \quad (2)$$

$$v \models \neg\neg\alpha \quad \text{iff} \quad v \models \alpha, \quad (3)$$

$$v \models \alpha \wedge \beta \quad \text{iff} \quad v \models \alpha \text{ and } v \models \beta, \quad (4)$$

$$v \models \neg(\alpha \wedge \beta) \quad \text{iff} \quad v \models \neg\alpha \text{ or } v \models \neg\beta, \quad (5)$$

$$v \models (\forall x)\alpha \quad \text{iff} \quad \text{for all } n \in N_v, v \models \alpha[x/n], \quad (6)$$

$$v \models \neg(\forall x)\alpha \quad \text{iff} \quad \text{for some } n \in N_v, v \models \neg\alpha[x/n], \quad (7)$$

$$v \models \Box_S \alpha \quad \text{iff} \quad \text{for all } v' \in \mathcal{R}_S(v), v' \models \alpha, \quad (8)$$

$$v \models \neg\Box_S \alpha \quad \text{iff} \quad v \not\models \Box_S \alpha. \quad (9)$$

A sentence α is called *satisfiable* iff $w \models \alpha$ for some world w . We say that a sentence α is *valid*, and write $\models \alpha$, iff $w \models \alpha$ holds for all worlds w . Finally, given two sentences α and β , we say that β is a *logical consequence* of α iff $\models \alpha \supset \beta$ holds. The following lemmas capture important structural properties of the semantics. They will be frequently used in the remaining sections.

Lemma 1. *For any declaration α and any valuations v, v' such that $v \subseteq v'$,*

$$\text{if } v \models \alpha \text{ then } v' \models \alpha.$$

Lemma 2. *For any declaration α and any world w ,*

$$w \models \Box_S \alpha \text{ iff } \cap \mathcal{R}_S(w) \models \alpha, \quad (1)$$

$$w \models \Diamond_S \alpha \text{ iff } \cup \mathcal{R}_S(w) \models \alpha. \quad (2)$$

Proof (1). Suppose that $w \models \Box_S \alpha$. By semantic rule 8, we obtain $v \models \alpha$ for all $v \in \mathcal{R}_S(w)$. It follows that $\cap \mathcal{R}_S(w) \models \alpha$. Dually, suppose that $w \not\models \Box_S \alpha$. By semantic rule 9, we obtain $v \not\models \alpha$ for some $v \in \mathcal{R}_S$. Since $\cap \mathcal{R}_S(w) \subseteq v$, by contraposition of lemma 1, it follows that $\cap \mathcal{R}_S(w) \not\models \alpha$.

Proof (2). Suppose that $w \models \Diamond_S \alpha$. By semantic rule 9, we obtain $v \not\models \neg \alpha$ for some $v \in \mathcal{R}_S(w)$. If $v \models \alpha$ then by lemma 1 we have $\cup \mathcal{R}_S(w) \models \alpha$. Otherwise, by contraposition of lemma 1 we obtain $\cap \mathcal{R}_S(w) \not\models \alpha \vee \neg \alpha$. By induction on the structure of α , it follows that $\cup \mathcal{R}_S(w) \models \alpha \wedge \neg \alpha$. So $\cup \mathcal{R}_S(w) \models \alpha$. Now suppose that $w \not\models \Diamond_S \alpha$. By semantic rule 8, we have $v \models \neg \alpha$ for all $v \in \mathcal{R}_S(w)$. If $\cup \mathcal{R}_S(w) \models \alpha$ then by lemma 1 we obtain $\cup \mathcal{R}_S(w) \models \alpha \wedge \neg \alpha$. By induction on the structure of α , it follows that $\cap \mathcal{R}_S(w) \not\models \alpha \vee \neg \alpha$, but this contradicts the former hypothesis. Therefore, we must obtain $\cup \mathcal{R}_S(w) \not\models \alpha$.

2.3 Axiomatization

We now focus on obtaining a sound and complete axiomatization for our logic. An *axiom system* consists of a collection of *axioms* and *inferences rules*. A *proof* in an axiom system is a finite sequence of sentences, each of which is either an instance of an axiom or follows by an application of an inference rule. Finally, we say that a sentence α is a *theorem* of the axiom system and write $\vdash \alpha$ if there exists a proof of α in the system. The axiom system of **AFOR** is the following.

Axioms:

$$\text{All tautologies of first-order logic} \quad (\text{A1})$$

$$\Box_S \neg \neg \alpha \equiv \Box_S \alpha \quad (\text{A2})$$

$$\Box_S (\alpha \wedge \beta) \equiv \Box_S \alpha \wedge \Box_S \beta \quad (\text{A3})$$

$$\Box_S \neg (\alpha \wedge \beta) \equiv \Box_S \neg \alpha \vee \Box_S \neg \beta \quad (\text{A4})$$

$$\Box_S (\forall x) \alpha \equiv \Box_S \bigwedge_{n \in N_s} \alpha[x/n] \quad (\text{A5})$$

$$\Box_S \neg (\forall x) \alpha \equiv \Box_S \bigvee_{n \in N_s} \neg \alpha[x/n] \quad (\text{A6})$$

$$\Box_S (a \vee \neg a), \text{ where } a \in A_S \quad (\text{A7})$$

$$\Diamond_S (a \wedge \neg a), \text{ where } a \notin A_S \quad (\text{A8})$$

$$\Box_S \alpha \supset \alpha \text{ where } \alpha \text{ is a ground declaration} \quad (\text{A9})$$

Inference rules:

$$\text{From } \vdash \alpha \text{ and } \vdash \alpha \supset \beta \text{ infer } \vdash \beta \quad (\text{R1})$$

$$\text{From } \vdash \alpha[x/n] \text{ infer } \vdash (\forall x) \alpha \quad (\text{R2})$$

The axiom system may be divided into two categories. The first one is concerned by axiom (A1) and rules (R1) and (R2) which come from standard first-order logic. Hence, the first-order fragment of **AFOR** is correctly handled. The specificity of our logic lies on the second category. Axioms (A2)-(A4) capture the properties of double negation, conjunction and disjunction, respectively. Axioms (A5)-(A8) are the key point of resource-bounded reasoning. The first two axioms introduce a limitation on the *quantification* capabilities of the reasoner. Specifically, they state that any universal (existential) quantifier can be rewritten to a finite number of conjunctions (disjunctions) which is bounded by the size of N_S . From an orthogonal point of view, the last two axioms impose a limitation on the *chaining* capabilities of the reasoner. Axiom (A7) says that the system necessarily infers the tautology $a \vee \neg a$, whenever a is in A_S . Dually axiom (A8) says that the system can infer the antilogy $a \wedge \neg a$, if a is not in A_S . Finally, axiom (A9) claims that reasoning under the scope of \Box_S is sound, provided that the declaration α is quantifier-free.

It is interesting to analyse the axiomatization from the standpoint of the so-called *logical omniscience problem* [9]. A reasoner is called logically omniscient if its inference capabilities are closed under logical consequence. By inference rule (R1) and axioms (A2)-(A4), we remark that $\Box_S \alpha \supset (\Box_S (\alpha \supset \beta) \supset \Box_S \beta)$ is a theorem of the axiom system. So, the inference capabilities of the reasoner are closed under *material* implication. However, we also remark that the sentence $\Box_S \alpha \supset ((\alpha \supset \beta) \supset \Box_S \beta)$ is *not* a theorem of the axiom system. Hence, the inference capabilities of the reasoner are not closed under *logical* implication. The following result gives soundness and completeness for the axiom system.

Theorem 1 (Soundness and Completeness). *For any sentence α ,*

$$\vdash \alpha \text{ iff } \models \alpha.$$

Proof (sketch). The soundness of the axiom system is easily demonstrated from the semantic rules and lemma 2. The proof of completeness is based on the technique of saturated sets presented in [7]. A set of sentences is called *saturated* if it is ω -complete and consistent. A set of sentences E is ω -complete if $E \cup p[x/n]$ is consistent whenever $E \cup \{\neg(\forall x) p\}$ is consistent. Completeness follows if we can show that any saturated set is satisfiable. We begin by extending a given ω -complete set E to a maximally consistent set using the Lindenbaum procedure. Then we build a world w_E as follows. d_E is an injective morphism from the ground terms in E to N , and $a \in L_E$ iff $a \in E$, for every primitive atom a . The central lemma in the proof shows that for any sentence α , we have $\alpha \in E$ iff $w_E \models \alpha$. The only difficulty is the case where α is of the form $\Box_S \beta$. We begin to rewrite β to an equivalent ground declaration into conjunctive normal form, by using (A2)-(A6). The “if” part of the proof is based on axioms (A8) and (A9). Dually, the “only if” part of the proof is built from axioms (A7) and (A9).

3 Approximate Reasoning

After an excursion into the logic **AFOR**, we now apply our results to the formalization of approximate first-order reasoning. In the knowledge representation paradigm, the main task for a reasoner is to decide whether a query is entailed, or not, from the knowledge base. In general, this task is divided into two steps. First, convert the knowledge base and the negation of the query into clausal form and next, determine whether the resulting declaration is satisfiable or not. In this study, we concentrate on the second step of the reasoning process.

From this perspective, we specify a resource-bounded reasoner as an “abstract type” that takes in input a clausal declaration α and an increasing sequence of resources (S_0, \dots, S_n) , and that approximates the problem of deciding whether α is satisfiable, or not, by means of two dual families of modal operators $(\Box_{S_0}, \dots, \Box_{S_n})$ and $(\Diamond_{S_0}, \dots, \Diamond_{S_n})$. If we prove that $\Box_{S_i} \alpha$ is satisfiable for any index i , then we have proved that α is satisfiable. Dually, if we prove that $\Diamond_{S_i} \alpha$ is unsatisfiable for any i , then we have proved that α is unsatisfiable. This stepwise process has the important advantage that the iteration may be stopped when a confirming answer is already obtained for a small index i .

Before examining into detail the properties of approximate reasoning, we introduce some useful definitions. An *Herbrand signature* is a first-order signature such that the set of function symbols F contains at least one constant symbol, and the set of standard names N is the set of all ground terms built from F . In other words, N is the *Herbrand universe* of the signature. Such a context greatly simplifies the technical aspects of the semantics. Specifically, in the language defined over a Herbrand signature, there exists exactly one denotation function d from ground terms to standard names, namely the identity function. Thus, any valuation v is uniquely determined by its components N_v and L_v .

A *clause* is a disjunction of literals. A *clausal declaration* is a declaration in prenex normal form containing only universal quantifiers, whose matrix is a disjunction of clauses. When clear from the context, such sentences will be respectively modeled as sets of literals and sets of clauses. With each clausal declaration α , we can uniquely associate a Herbrand signature whose function and predicate symbols are those occurring in α , with the additional condition that if α does not contain any constant, then we introduce a new constant in its signature. To avoid some complicated notations, we assume from now that the underlying representation language of a clausal declaration α is the language built from the Herbrand signature of α .

With these notions in hand, we can now examine the semantical properties of approximations. First, we show that resource-bounded reasoning is *improvable* and *dual*. The quality of approximations improves as we increase the resources. Moreover, both sound approximations and complete ones are improvable.

Theorem 2 (Monotonicity). *For any clausal declaration α and any resource parameters R and S such that $R \subseteq S$,*

$$\text{if } \Box_R \alpha \text{ is satisfiable, then } \Box_S \alpha \text{ is satisfiable,} \quad (1)$$

$$\text{if } \Diamond_R \alpha \text{ is unsatisfiable, then } \Diamond_S \alpha \text{ is unsatisfiable.} \quad (2)$$

Proof (1). Suppose that $\Box_R \alpha$ is satisfiable. Then $w \models \Box_R \alpha$ for some world w . Let v denotes $\cap \mathcal{R}_R(w)$. By construction, $N_v = N_R$ and $L_v = L_w \cap L_R$. Moreover, by lemma 2, it follows that $v \models \alpha$. Now, let us define a total mapping τ from N_S to N_R such that $\tau(n) = n$ for every $n \in N_R$. Let v' be a new valuation where $N_{v'} = N_S$ and $L_{v'}$ is the set of all literals $l(\mathbf{n})$ such that $l(\tau(\mathbf{n})) \in L_v$. Suppose that $v' \not\models \alpha$. Then there exists at least one ground clause $\gamma(\mathbf{n})$ of α generated from N_S and such that $\gamma(\mathbf{n}) \cap L_{v'} = \emptyset$. It follows that $\gamma(\tau(\mathbf{n})) \cap L_v = \emptyset$. Since $\gamma(\tau(\mathbf{n}))$ is a ground clause of α generated from N_R , it follows that $v \not\models \alpha$, but this contradicts the former hypothesis. Hence, $v' \models \alpha$. Let w' be a new world such that $L_{w'} = L_{v'} \cup (L_w / L_S)$ and let v'' denotes $\cap \mathcal{R}_S(w')$. By construction, $N_{v''} = N_S$ and $L_{v''} = L_{v'} \cap L_R$. Thus $v' \subseteq v''$. By lemma 1, it follows that $v'' \models \alpha$. Hence, $w' \models \Box_S \alpha$. Therefore, $\Box_S \alpha$ is satisfiable.

Proof (2). Suppose that $\Diamond_S \alpha$ is satisfiable. Then $w \models \Diamond_S \alpha$ for some world w . Let v and v' denote $\cup \mathcal{R}_S(w)$ and $\cup \mathcal{R}_R(w)$, respectively. By application of lemma 2, we have $v \models \alpha$. Now, let us define a new valuation v'' such that $N_{v''} = N_v$ and $L_{v''} = L_v \cup (L - L_R)$. By construction, $v \subseteq v''$. Thus, by lemma 1 it follows that $v'' \models \alpha$. Moreover, it is clear that $L_{v''} = L_{v'}$. Suppose that $v' \not\models \alpha$. Then, there exists at least one ground clause γ of α generated from N_R and such that $\gamma \cap L_{v'} = \emptyset$. Since $N_R \subseteq N_S$, it follows that γ is a ground clause of α generated from N_S . Moreover, since $L_{v''} = L_{v'}$, it follows that $\gamma \cap L_{v''} = \emptyset$. Hence $v'' \not\models \alpha$, but this contradicts the former hypothesis. So, $v' \models \alpha$ and by lemma 2, it follows that $w \models \Diamond_R \alpha$. Therefore, $\Diamond_R \alpha$ is satisfiable.

Second, we demonstrate that there exists a systematic adequacy relationship between approximate reasoning and classical first-order reasoning.

Theorem 3 (Adequacy). *For any clausal declaration α and parameter S ,*

$$\text{if } \Box_S \alpha \text{ is satisfiable, then } \alpha \text{ is satisfiable,} \quad (1)$$

$$\text{if } \Diamond_S \alpha \text{ is unsatisfiable, then } \alpha \text{ is unsatisfiable.} \quad (2)$$

Proof (1). Let R be a new parameter such that $N_R = N_R$ and P_R is the set of all predicates occurring in the formula α . Suppose that $\Box_S \alpha$ is satisfiable. By theorem 2 it follows that $\Box_R \alpha$ is satisfiable. Then $w \models \Box_R \alpha$ for some world w . Let τ be a function from N to N_R such that $\tau(n) = n$ for every $n \in N_R$. Let w' be a world where $L_{w'} = \{l(\mathbf{n}) : l(\tau(\mathbf{n})) \in L_w\}$. Suppose that $w' \not\models \alpha$. Then there exists a ground clause $\gamma(\mathbf{n})$ of α such that $w' \not\models \gamma(\mathbf{n})$. It follows that $w \not\models \gamma(\tau(\mathbf{n}))$. However, by contraposition of axiom (A9) we obtain $w \not\models \Box_R \gamma(\tau(\mathbf{n}))$. Since $\gamma(\tau(\mathbf{n}))$ is a ground clause generated from N_R , it follows that $w \not\models \Box_R \alpha$, hence contradiction. So, $w' \models \alpha$ and therefore α is satisfiable.

Proof (2). We use the parameter R defined in part (1). Suppose that $\Diamond_S \alpha$ is unsatisfiable. By theorem 2 it follows that $\Diamond_R \alpha$ is unsatisfiable. Let E be the set of all ground clauses of α generated by N_R . Clearly, $\Diamond_R E$ is unsatisfiable. Moreover, from axiom (A9) and rule (R1) we infer that $E \supset \Diamond_R E$ is a theorem of our logic. By contraposition of this theorem, it follows that E is unsatisfiable. However, E is a finite subset of all ground instances of clauses of α . By application of Herbrand's theorem [3] (only if part), it follows that α is unsatisfiable.

Third and finally, we guarantee the convergence of approximate unsatisfiability: if a declaration is unsatisfiable, then using enough resources, we are guaranteed to find the correct solution. Notice that we cannot hope obtaining an analogue result for the dual part: since first-order unsatisfiability is recursively but *not* co-recursively denumerable, a infinite amount of resources may be necessary for determining satisfiability of a first-order clausal declaration.

Corollary 1 (Convergence). *For any clausal declaration α , if α is unsatisfiable then there exists a resource parameter S such that $\diamond_S \alpha$ is unsatisfiable.*

Proof. Suppose that α is unsatisfiable. Then, by application of Herbrand's theorem [3] (if part), there must exist a finite unsatisfiable set E of ground clauses of α . Let R_S and N_S be the set of all predicates and ground terms that occur in E . Clearly enough, $\diamond_S E$ is unsatisfiable. Since E is a subset of the set of all ground clauses of α generated by N_S , it follows that $\diamond_S \alpha$ is unsatisfiable.

Example 1. Suppose we are given the following declaration:

$$\alpha = \{\{p(x, y), r(x)\}, \{\neg q(f(b)), r(x)\}, \{\neg r(a), q(f(x))\}, \{\neg p(a, b), q(x)\}\}.$$

The Herbrand signature of α is defined by the sets $P = \{p, q, r\}$, $F = \{a, b, f\}$ and $N = \{a, b, f(a), f(b), \dots\}$. We want to show that α is satisfiable. Hence, we need to find a parameter S such that $\square_S \alpha$ is satisfiable. In fact, this happens with $P_S = \{q, r\}$ and $N_S = \{a, f(a)\}$ which are restricted subparts of the signature.

Example 2. Suppose we are given the following declaration:

$$\alpha = \{\{p(x)\}, \{\neg p(a), q(x)\}, \{r(g(x), y), q(f(a))\}, \{\neg p(b), \neg q(x)\}, \{\neg r(x, f(y))\}\}.$$

The Herbrand signature of the formula is defined by $P = \{p, q, r\}$, $F = \{a, b, f, g\}$ and $N = \{a, b, f(a), g(a), \dots\}$. We want to show that α is unsatisfiable. So we need to find a parameter S such that $\diamond_S \alpha$ is unsatisfiable. In fact this holds with $P_S = \{p, q\}$ and $N_S = \{a, b\}$ which are restricted subparts of the signature.

4 Approximate Computation

In this section, we investigate the computational aspects of approximate reasoning. We present an original algorithm, named AFOS, for approximate first-order satisfiability. We begin to specify the algorithm, next we prove its soundness and completeness and then we analyse its computational complexity.

Before exploring the algorithm into detail, we introduce some additional definitions. A *substitution* is a mapping θ from variables to terms. Given a resource parameter S , a *S-substitution* is a substitution θ such that the range of θ is a subset of N_S . Given two parameters R and S such that $R \subseteq S$, a *(R, S)-substitution* is a S -substitution θ such that the range of θ contains a nonempty subset of $N_S - N_R$. A clause δ is called a *S-instance* (resp. *(R, S)-instance*) of a clause γ if $\delta = \gamma\theta$ for some S -substitution (resp. *(R, S)-substitution*) θ . In the following, α_S denotes the set of all S -instances of α generated by S .

The algorithm AFOS, presented in figure 1, can be thought as an iterative instance-based theorem prover. The three major parts of the algorithm are: first the choice of new resources, second the resource-bounded instance generation, and third the satisfiability test by a standard propositional prover. The algorithm basically carries out these three steps until a proof for satisfiability or unsatisfiability is found or a time-space limit (i.e. interruption) is reached. It is important to remark that the procedure both returns the solution and the resources that have been spent for computing the solution. For example, suppose that $\text{AFOS}(\alpha)$ returns (true, S) for some clausal declaration α . The intuitive reading of this result is “ α can be shown satisfiable using the resources S ”. Such an information not only provides knowledge about the solution but also meta-knowledge about the resources needed to compute the solution.

```

Input : a clausal declaration  $\alpha$ ;
Output: a resource parameter  $S$  and the truth-value true if  $\Box_S \alpha$  is satisfiable, false if  $\Diamond_S \alpha$  is unsatisfiable and unknown otherwise;

 $S \leftarrow S_0$ ;
 $\alpha_S^\Box \leftarrow \emptyset$ ;
 $\alpha_S^\Diamond \leftarrow \emptyset$ ;

while not interruption() do
    | Choice of resource parameter;
    |  $R \leftarrow S$ ;
    |  $S \leftarrow \text{choose}()$ ;
    | Instantiation;
    | if  $P_R = P_S$  then
    | | foreach  $(R, S)$ -instance  $\gamma$  of  $\alpha$  do
    | | |  $\alpha_S^\Box \leftarrow \alpha_S^\Box \cup \{\gamma \cap L_S\}$ ;
    | | | if  $\gamma \subseteq L_S$  then  $\alpha_S^\Diamond \leftarrow \alpha_S^\Diamond \cup \{\gamma\}$ ;
    | | else
    | | |  $\alpha_S^\Box \leftarrow \emptyset$ ;
    | | |  $\alpha_S^\Diamond \leftarrow \emptyset$ ;
    | | | foreach  $S$ -instance  $\gamma$  of  $\alpha$  do
    | | | |  $\alpha_S^\Box \leftarrow \alpha_S^\Box \cup \{\gamma \cap L_S\}$ ;
    | | | | if  $\gamma \subseteq L_S$  then  $\alpha_S^\Diamond \leftarrow \alpha_S^\Diamond \cup \{\gamma\}$ ;
    | | Satisfiability;
    | | if  $\alpha_S^\Box$  is satisfiable then return  $(\text{true}, S)$ ;
    | | if  $\alpha_S^\Diamond$  is unsatisfiable then return  $(\text{false}, S)$ ;
    | return  $(\text{unknown}, S)$ ;

```

Figure 1: Approximate First-Order Satisfiability (AFOS)

Interestingly, our algorithm incorporates several major features. First, the algorithm is *interruptible*: it can be stopped at any time and provide some answer. Second, the procedure is *dual*: it can compute at the same time the satisfiability and the unsatisfiability of a generic formula. Third, the instance generation step can be shown *progressive*. Specifically, if R is a subset of S , then any clause in the declaration α_R^\square is a subset of some clause in the declaration α_S^\square ; furthermore, the declaration α_R^\diamond is a subset of the declaration α_S^\diamond . Fourth and finally, our algorithm can be shown *incremental* and *anytime* [30]. In particular, the reasoner can decide to fix the choice of predicates P_S for a certain number of iterations. During these iterations, the set of ground terms N_S is progressively increased and the declarations α_S^\square and α_S^\diamond are progressively expanded in an incremental fashion. If a solution is not found then the reasoner can choose a new set P_S , reinitialize the set N_S , and apply again the same strategy.

The approximation schema is general enough to be combined with a wide range of satisfiability testers. The underlying interest is to use methods which are appropriate for the problem at hand and that have been shown powerful enough for solving large size instances of the problem. Complete methods such as depth first search enumeration [27] can be used to compute at the same time the satisfiability of α_S^\square and the unsatisfiability of α_S^\diamond . On the other hand, incomplete methods such as local search algorithms [8, 12] can be exploited if we concentrate on the satisfiability of α_S^\square .

The two following results clarify the interest of approximate computation. The first theorem gives soundness and completeness for the algorithm. The second theorem states its computational complexity. To this point, the last result claims that the two barriers of complexity in first order reasoning, that is “quantification” and “chaining”, are bounded by the resource parameter S .

Theorem 4 (Soundness and Completeness for AFOS). *Given a clausal declaration α a resource parameter S and no interruption of the algorithm,*

$$\text{AFOS}(\alpha) \text{ returns } (true, S) \text{ iff } \Box_S \alpha \text{ is satisfiable,} \quad (1)$$

$$\text{AFOS}(\alpha) \text{ returns } (false, S) \text{ iff } \Diamond_S \alpha \text{ is unsatisfiable.} \quad (2)$$

Proof. We only examine part (1) as a dual strategy applies to part (2). we know that $\Box_S \alpha$ is satisfiable iff $w \models \Box_S \alpha$ for some world w . By lemma 2, $w \models \Box_S \alpha$ iff $\cap \mathcal{R}_S(w) \models \alpha$. Let v denotes $\cap \mathcal{R}_S(w)$. By semantic rules (4) and (6), $v \models \alpha$ iff $v \models \gamma$ for every S -instance γ of α . By semantic rules (1) and (5), $v \models \gamma$ iff $\gamma \cap L_v \neq \emptyset$. So, $v \models \alpha$ iff $v \models \alpha_S^\square$ and hence, $\Box_S \alpha$ is satisfiable iff $\Box_S \alpha_S^\square$ is satisfiable. Now suppose that $\text{AFOS}(\alpha)$ returns $(true, S)$. Then α_S^\square is satisfiable. Since the relations and terms that occur in α_S^\square are subsets of R_S and N_S it follows that $\Box_S \alpha_S^\square$ is satisfiable. Therefore, $\Box_S \alpha$ is satisfiable. Dually, assume that $\Box_S \alpha$ is satisfiable. So, $\Box_S \alpha_S^\square$ is satisfiable. By axiom (A9), it follows that α_S^\square is satisfiable. Provided that no interruption occurred, $\text{AFOS}(\alpha)$ returns $(true, S)$.

Theorem 5 (Complexity). *For any clausal declaration α and any resource parameter S , deciding whether $\Box_S \alpha$ is satisfiable and $\Diamond_S \alpha$ is satisfiable can be computed in $O(|\alpha_S| \cdot 2^{|P_S|})$ time.*

Proof. Let us examine the sentence $\Box_S \alpha$. By application of theorem 4, $\Box_S \alpha$ is satisfiable iff AFOS(α) returns $(true, S)$. The instantiation part of the algorithm must, at most, generate all S -instances of α which is $O(|\alpha_S|)$. So, the time complexity of this part is $O(|\alpha_S|)$. In the worst case, all the clauses of α are stored in α_S^\square , so the size of α_S^\square is $O(|\alpha_S|)$. Moreover, the cardinality of all distinct ground atoms in α_S^\square is $O(|P_S|)$. So, the worst-case time complexity of the satisfiability part is $O(|\alpha_S| \cdot 2^{|P_S|})$. A dual argument applies to the sentence $\Diamond_S \alpha$.

Example 3. We consider again example 1. Suppose that AFOS chooses the resource parameter S defined by $P_S = \{q, r\}$ and $N_S = \{a, f(a)\}$. We obtain $\alpha_S^\square = \{\{r(a)\}, \{r(f(a))\}, \{\neg r(a), q(f(a))\}, \{q(a)\}, \{q(f(a))\}\}$. Clearly, α_S^\square is satisfiable. Therefore, it follows that $\Box_S \alpha$ is satisfiable.

Example 4. Now consider again example 2 and suppose that AFOS selects the resource parameter S with $P_S = \{p, q\}$ and $N_S = \{a, b\}$. We obtain: $\alpha_S^\diamond = \{\{p(a)\}, \{p(b)\}, \{\neg p(a), q(a)\}, \{\neg p(a), q(b)\}, \{\neg p(b), \neg q(a)\}, \{\neg p(b), \neg q(b)\}\}$. It is clear that α_S^\diamond is unsatisfiable. Hence, it follows that $\Diamond_S \alpha$ is unsatisfiable.

Example 5. We assume that the following knowledge base E is part of a very large ontology of group theory and other algebraic structures. E uses a small signature which we assume to be part of a larger vocabulary.

$$E = \begin{cases} \{p(x, y, f(x, y))\}, \\ \{p(e, x, x)\}, \\ \{p(x, e, x)\}, \\ \{p(i(x), x, e)\}, \\ \{p(x, i(x), e)\}, \\ \{\neg p(x, y, v), \neg p(y, z, v), \neg p(x, w, u), p(v, z, u)\}, \\ \{\neg p(x, y, v), \neg p(y, z, v), \neg p(v, z, u), p(x, w, u)\}, \\ \{\neg s(x), \neg s(y), \neg p(x, i(y), z), s(z)\} \end{cases}$$

Suppose we are given the following query: for any given element in a subgroup, show that its inverse is still in the subgroup. This can be denoted by $(\forall x)(s(x) \supset s(i(x)))$. Let $\alpha = E \cup \{\{s(a)\}, \{\neg s(i(a))\}\}$. We want to show that α is unsatisfiable. So the algorithm needs to find a resource parameter S such that α_S^\diamond is unsatisfiable. In fact, this holds with $R_S = \{s, p\}$ and $N_S = \{a, i(a), e\}$. We remark that the number of clauses and atoms in α_S^\diamond is 1475 and 30, respectively. By combining iterative instantiation with an improved version of the Davis-Putnam procedure [27], unsatisfiability should be inferred in short time.

5 Conclusion

This main motivation behind this work has been to obtain a model approximate reasoning that defines a computationally more attractive reasoner than classical first order logic. We have stressed on a multi-modal logic which contains a well-founded semantics and a correct and complete axiomatization. Based

on this logic, we have shown that our framework integrates several major features: bounded resources, improvability and dual reasoning. Finally, we designed a resource-bounded algorithm with a simple modification of classical instance-based theorem provers, and we have discussed the quality and complexity guarantees of this approximate deduction mechanism.

There are various avenues of research that come out of this work. On the logic side, a first investigation is to extend the language with equality. In particular, the question whether first-order logic with equality can be embedded in our framework should be settled. A secondary, but important, investigation is to examine decidable sub-fragments of first-order logic such as, for instance, Schönfinkel-Bernays expressions. To this point, it would be interesting to obtain a convergence result for the satisfiability problem of these sub-fragments. On the algorithmic side, a number of open problems remain to be explored. In particular, an important issue is the development of “intelligent” strategies for the incremental choice of resources. This choice may be heuristic; search strategies advocated in the literature of instance-based theorem proving should play a major role in the global efficiency of the framework [4, 13, 18, 29]. For example, a well-known principle is to iteratively choose predicates and terms according to their increasing arity. More sophisticated heuristics can be developed by combining unification techniques used for instantiation and strategies employed in propositional testers. Alternatively, the choice of the resources may be guided by *control knowledge*; it is a commonplace that knowledge about the structure of a base is important for efficient inferences [24]. To this very point, we recall that AFOS communicates at the same time knowledge about the solution but also meta-knowledge about the computation. In the query-answering process this meta-knowledge should be automatically learned to perform an appropriate choice of the resources which guarantees a high degree of confidence. These criteria are under study and we hope that an intelligent control strategy will be possible for approximate first-order reasoning.

References

1. N. D. Belnap. A useful four-valued logic. In *Modern Uses of Multiple-Valued Logic*, pages 8–37. Reidel, Dordrecht, 1977.
2. M. Cadoli and M. Schaerf. Approximate inference in default reasoning and circumscription. *Fundamenta Informaticae*, 2:123–143, 1995.
3. C. Chang and R. C. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press Inc., 1973.
4. H. Chu and D. A. Plaisted. CLIN-S: A semantically guided first-order theorem prover. *Journal of Automated Reasoning*, 18(2):183–188, 1997.
5. M. Dalal. Anytime clausal reasoning. *Annals of Mathematics and Artificial Intelligence*, 22(3-4):297–318, 1998.
6. A. del Val. Tractable databases: How to make propositional unit resolution complete through compilation. In *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, pages 551–561, 1994.

7. J. W. Garson. Quantification in modal logic. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic, Volume II: Extensions of Classical Logic*, pages 249–307. D. Reidel Publishing Co., 1984.
8. J. Gu, P. W. Purdom, J. Franco, and B. W. Wah. Algorithms for satisfiability problem: A survey. In *Satisfiability Problem: Theory and Applications*, volume 35, pages 19–152. American Mathematical Society, 1997.
9. J. Hintikka. *Knowledge and Belief*. Cornell University Press, 1962.
10. D. W. Etherington J. M. Crawford. A non-deterministic semantics for tractable inference. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 286–291, 1998.
11. H. Kautz, D. McAllester, and B. Selman. Encoding plans in propositional logic. In *Principles of Knowledge Representation and Reasoning*, pages 374–384, 1996.
12. H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic and stochastic search. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 1194–1201, 1996.
13. S. Kim and H. Zhang. ModGen: Theorem proving by model generation. In *Proc. of the 12th National Conference on Artificial Intelligence*, pages 162–167, 1994.
14. F. Koriche. Approximate reasoning about combined knowledge. In *Intelligent Agents Volume IV*, volume 1365 of *LNAI*, pages 259–273. Springer Verlag, 1998.
15. F. Koriche. A logic for anytime deduction and anytime compilation. In *Logics in Artificial Intelligence*, volume 1489 of *LNAI*, pages 324–342. Springer Verlag, 1998.
16. G. Lakemeyer. Limited reasoning in first-order knowledge bases. *Artificial Intelligence*, 71:213–255, 1994.
17. G. Lakemeyer. Limited reasoning in first-order knowledge bases with full introspection. *Artificial Intelligence*, 84:209–255, 1996.
18. S. J. Lee and D. A. Plaisted. Problem solving by searching for models with a theorem prover. *Artificial Intelligence*, 69(1-2):205–233, 1994.
19. H. J. Levesque. Foundations of a Functional Approach to Knowledge Representation. *Artificial Intelligence*, 23:125–212, 1984.
20. H. J. Levesque. A logic of implicit and explicit belief. In *Proceedings of the 6th National Conference on Artificial Intelligence*, pages 198–202, 1984.
21. H. J. Levesque and R. J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3(2):78–93, 1987.
22. F. Massacci. Anytime approximate modal reasoning. In *Proceedings of 15th National Conference on Artificial Intelligence*, pages 274–279, 1998.
23. R. Reiter. What should a database know? In *Proceedings of the 7th ACM Symposium on Principles of Database Systems*, pages 302–304, 1988.
24. M. Schaerf and M. Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74:249–310, 1995.
25. J. Slaney. SCOTT: A model-guided theorem prover. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 109–115, 1993.
26. M. Y. Vardi. On the complexity of bounded-variable queries. In *Proceedings of the 14th ACM Symposium on Principles of Database Systems*, pages 266–276, 1995.
27. H. Zhang and M. E. Stickel. Implementing the davis-putnam method. *Journal of Automated Reasoning*, 24(1/2):277–296, 2000.
28. J. Zhang and H. Zhang. SEM: a system for enumerating models. In *Proc. of the 14th International Joint Conference on Artificial Intelligence*, pages 298–303, 1995.
29. Y. Zhu. *Efficient First-Order Semantic Deduction Techniques*. PhD thesis, University of North Carolina at Chapel Hill (USA), 1998.
30. S. Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996.