Proving unprovability

M. Tiomkin

Oren str. 35/10
Haifa 34734, Israel*

ABSTRACT

A formal proof system for unprovability in the predicate calculi is developed. This system is shown to be complete with respect to the logic of finite structures. Its applications may be extending the "negation by failure" of Prolog, preventing infinite loops in a deductive data base or Prolog, or proving formulas in a nonmonotonic (default) logic.

1. Introduction

Working with a theorem prover, we are often interested to know when a formula is *not* provable, i.e. when it is not a tautology. First, this allows us to prevent a theorem prover from running forever when a nontheorem is supplied. Second, while investigating the external world, we want to know also what we cannot do, i.e., the negative information, in addition to what we can do. For example, in Prolog or in a deductive data base we can prove negative facts as well as positive ones. But, unfortunately, this negation is defined by the so called "*finite failure*" of the theorem prover (cf. [1]), and this allows the prover to fall into infinite loop beyond any possibility of recognition. The reason for such careless behavior of theorem provers is that, generally, the unprovability in the first order predicate calculus is not recursively enumerable (*RE*), and then we need some nonmechanical oracle activity (for example, human intervention) to detect this. Obviously, if we could detect unprovability, this would help us to define a wider kind of negation, by the rule

$$\frac{\nvdash \varphi}{\vdash \neg\varphi}.$$

This rule is called the *closed world assumption*, or *CWA* (cf. [1]), and such proof systems were widely discussed in the last years by the AI society. For example, the "default" nonmonotonic logics are based on the rules of this type (cf. [2], [3], [4]). Of course, a "non-theorem" prover would be an important part of a system dealing with a nonmonotonic logic.

Clearly, any feasible "non-theorem" prover cannot prove all the "non-theorems" of predicate calculus. Of course, we can restrict ourselves to a part of predicate calculus where the set of "non-theorems" is *RE*. We can deal with some decidable theory, e.g. propositional calculus (cf. [5]), Ackermann formulas (cf. [6]), Pressburger Arithmetic, real closed fields *etc.* Another possibility is to try to prove the non-theorems when possible, leaving some formulas without any information about their (un)provability. This is done, for example, by the "finite failure" Prolog theorem provers.

We intend here to define the "non-theorem" proof system which is more powerful than the "finite failure" one, and where all the "important" non-theorems are provable. We propose a Gentzen type proof system with a natural set of axioms and rules. The most interesting of these rules, where a formula $\exists x\, \varphi(x)$ is proved unprovable if some special Herbrand example $\bigvee_i \varphi(t_i)$ is unprovable, corresponds to Herbrand's theorem (cf. [7]). Very fortunately, it turns out that the non-theorems provable in this system are exactly those not valid in a finite model. The logic of finite models (or finite structures) is recognized by the computer scientists as the natural framework for explaining the world of computations of bounded complexity (cf. [8], [9], [10], [11]). It was frequently claimed that the infinite models (structures) are often meaningless for Computer Science. The non-theorem proof system proposed here seems to be convenient for dealing with finite structures. On the other hand, the logic of finite structures rises another problem - the set of the *valid* sentences is not *RE* (cf. [12]).

The paper is organized as follows. In the next section

*This paper had been written when the author was with the Dept. of Computer Science, Technion, Haifa, Israel.

we give the necessary definitions. In Section 3 we prove the completeness of the non-theorem proof system for the propositional calculus and for the first order logic of finite structures. In Section 4 we apply the non-theorem proof system to the automatic loop detecting for Prolog or deductive data base without function symbols. In Section 5 the open questions and the future work directions are presented.

The author is very grateful to M. Kaminski for fruitful discussions and critical reading the paper.

2. The unprovability proof system

We consider here the usual language of the predicate calculus with logical connectives $\vee, \neg, \forall$, predicates and equality, and without function symbols. c, d denote constants, x, y denote variables; t, s are terms, φ, ψ are formulas, and Γ, Δ, Θ are finite multisets of formulas.

A *sequent* is an expression $\Gamma \to \Delta$. It is valid in the model if one of the formulas from Γ is not valid, or one of the formulas from Δ is valid. The sequent is valid if it is valid in all models. The logical meaning of the sequent $\Gamma \to \Delta$ is that the conjunction of the formulas of Γ implies the disjunction of the formulas of Δ.

One can easily define a sequent calculus for proving the valid sequents. The idea belongs to Gentzen (cf. [13]), and the formulation of the proof system in English may be found in [14]. The intuitionistic sequent calculus is obtained by allowing only those sequents $\Gamma \to \Delta$ with $|\Delta| = 0,1$. *The axioms and rules remain the same!* The Prolog resolution calculus is obtained from the intuitionistic sequent calculus by allowing only atomic formulas, i.e. formulas without logical connectives. Then the sequent $\Gamma \to \{\varphi\}$ is a Prolog clause, and the sequent $\Gamma \to$, with the empty Δ, is a set of goals (the aim is, as usual in Prolog, to prove the *empty* sequent $\to$). The only rules remaining after this restriction are the substitution rule and the *cut* (it is yet very different from the Prolog cut, see below), which in this case make up exactly the resolution rule.

Similarly, we define an *antisequent* as $\Gamma \not\to \Delta$. The meaning of this expression is that the sequent $\Gamma \to \Delta$ is *not valid* (provable). We define the natural axioms and proof rules for the antisequents which result from the various theorems for the usual sequent calculus. We call this system UPC (unprovability calculus).

Axioms.

(*Ax*) $\Gamma \not\to \Delta$, where Γ, Δ contain only atomic formulas, Γ does not contain equality, Δ does not contain $t=t$, and $\Gamma \cap \Delta = \varnothing$.

Introduction of $\vee$.

$$(\not\to\vee)\ \frac{\Gamma \not\to \Delta, \varphi, \psi}{\Gamma \not\to \Delta, \varphi \vee \psi} \qquad (\vee\not\to)\ \frac{\Gamma, \varphi \not\to \Delta \quad \Gamma, \psi \not\to \Delta}{\Gamma, \varphi \vee \psi \not\to \Delta}$$

Introduction of $\neg$.

$$(\not\to\neg)\ \frac{\Gamma, \varphi \not\to \Delta}{\Gamma \not\to \Delta, \neg\varphi} \qquad (\neg\not\to)\ \frac{\Gamma \not\to \Delta, \varphi}{\Gamma, \neg\varphi \not\to \Delta}$$

The above axioms and rules comprise the complete unprovability proof system for the propositional calculus (see Section 3). Notice how beautiful and reasonable they are. We call this system PUPC (propositional unprovability calculus). The proof system for the first order unprovability looks much more ugly. First, we need to use the usual provability proof system. Second, the system proposed here cannot be easily extended to deal with the function symbols.

We recall the well-known *cut* rule (cf. [13], [14]), which is similar to the usual *modus ponens* rule for the Hilbert type calculus (the proof systems for proving single formulas with the help of axioms and rules). The cut may be formulated as follows.

$$\frac{\Gamma \to \Delta, \varphi \quad \Gamma, \varphi \to \Delta}{\Gamma \to \Delta}$$

Reversing this rule, we obtain two cuts for the unprovability.

$$(\not\to cut)\ \frac{\Gamma \not\to \Delta \quad \Gamma, \varphi \to \Delta}{\Gamma \not\to \Delta, \varphi} \qquad (cut \not\to)\ \frac{\Gamma \not\to \Delta \quad \Gamma \to \Delta, \varphi}{\Gamma, \varphi \not\to \Delta}$$

The *thinning* rule is exactly the reversed thinning rule for the usual sequent calculus.

$$(thin)\ \frac{\Gamma \not\to \Delta, \quad \Gamma' \subseteq \Gamma, \ \Delta' \subseteq \Delta}{\Gamma' \not\to \Delta'}$$

For the equality we need only a rule for adding an inessential equality to the left.

$$(=\not\to)\ \frac{\Gamma \not\to \Delta}{\Gamma, t=s \not\to \Delta},$$

where t or s does not appear freely in $\Gamma \cup \Delta$.

For dealing with quantifiers, we need two rules for introducing $\forall$ - to the left and to the right size of a sequent. While the right introducing of $\forall$ is very natural, and its soundness easily follows from the tautology $\forall x\, \varphi(x) . \supset . \varphi(t)$, the left introducing of $\forall$ needs the Herbrand's theorem for its justification.

$(\not\to\forall)$ $\dfrac{\Gamma\not\to\Delta, \varphi(x|t)}{\Gamma\not\to\Delta, \forall x\,\varphi}$ $\qquad$ $(\forall\not\to)$ $\dfrac{\Gamma, \{\varphi(x|t_i)\}\not\to\Delta}{\Gamma, \forall x\,\varphi\not\to\Delta}$,

where Γ, Δ contain only atomic formulas, φ is a universal formula, and the nonempty $\{t_i\}$ comprises of all the free variables and constants from $\Gamma, \forall x\,\varphi, \Delta$.

Notice that the requirement of atomicity of Γ, Δ in the $(\forall\not\to)$ rule may be replaced by the weaker one that the antisequent $\Gamma, \forall x\,\varphi\not\to\Delta$ is "existential", i.e. that all the formulas in $\Gamma, \forall x\,\varphi$ are universal, and all the formulas in Δ are existential. The rule will remain sound (admissible), but we do not need this for the completeness results.

Recall that the cut rule is eliminable in the first order sequent calculus (the well known Gentzen's Hauptsatz) giving the so called subformula property, i.e. that if a formula is provable then it is provable from its subformulas. But, in our first order unprovability system the cut is not eliminable. One can easily see that the provable antisequent $\forall x\neg\forall y(x=y)\not\to$ is not provable without the cut. However, the unprovability cut does have the subformula property, and the only "problematic" rule without the subformula property is the thinning rule. Yet, the problem with the cut is that it is not "pure" unprovability rule, because it mixes the provability and unprovability proofs.

3. Completeness results

In order to justify our proof system, we should present some completeness results. I.e., that an antisequent is provable if and only if the corresponding sequent is not provable. Although it is true for the propositional version (see Theorems 1 and 2.1), one can easily show the nonexistence of a complete proof system for the first order unprovability. The reason is that the set of unprovable first order formulas is not RE, and so the first order unprovability cannot be completely determined by a finite proof. Thus we give only a partial completeness result in the first order case. That is to say, we show the completeness of the first order antisequent calculus with respect to the logic of finite structures.

First, we note that all the rules of our unprovability calculus derive an antisequent from at most one other antisequent. Thus, in every unprovability proof there is a unique path of antisequents from the root (the derived antisequent) to a leaf (an axiom). Also, for any axiom of unprovability we can easily construct a (finite) model "satisfying" it. This allows us to prove the soundness of the unprovability calculus in the constructive way.

Theorem 1. *If the antisequent $\Gamma\not\to\Delta$ is provable in* UPC (PUPC), *then there exists a finite (propositional) model M where all the formulas from Γ are valid and all the formulas from Δ are not. Moreover, this model is constructed by a simple (polynomial or linear time) algorithm.*

Proof. Given a proof of an antisequent $\Gamma\not\to\Delta$, it has a unique axiom $\Gamma'\not\to\Delta'$ in it. On the Herbrand universe of the constants and variables from Γ', Δ' (cf. [15]), we define the model M' as follows. A predicate $p(t_1, \ldots, t_n)$ is true in the model if the corresponding formula belongs to Γ', otherwise it is false. Trivially, this model (with assigning of the variables and constants to themselves) does not satisfy the sequent $\Gamma'\to\Delta'$. Then, following the path of antisequents from the axiom to the root, we add an assignment for a new constant or variable when the $(=\not\to)$ rule is applied, and reduce the domain to those elements having an assignment from the Herbrand universe of the current antisequent when the $(\forall\not\to)$ rule is applied. Then we convince ourselves that these models "satisfy" the corresponding antisequents appearing in the proof, in particular $\Gamma\not\to\Delta$. If we define finite model as a finite domain D with a set of tuples $P\subseteq D^n$ for every n-place predicate p, then this algorithm is linear. For a different definition of a model it is only polynomial. ■

The proof of completeness, i.e. that the existence of a model implies the existence of an unprovability proof, is a bit more complicated than that of soundness.

Theorem 2.1 (PUPC). *If there exists a propositional model M where all the formulas from Γ are valid and all the formulas from Δ are not, then the propositional antisequent $\Gamma\not\to\Delta$ is provable in* PUPC. *Moreover, this proof is constructed in the $O(nl\times|\Gamma\not\to\Delta|)$ time, where nl is the number of logical connectives in $\Gamma\not\to\Delta$.*

Proof. First, we mark all the subformulas in $\Gamma\not\to\Delta$ which are valid in the model.

Afterwards we build up the proof top-down, dealing each time with a nonatomic formula. We change the antisequent in accordance with the main logical connective of this formula, by the corresponding introduction rule. The new antisequent has always less logical connectives than the old one. The only nontrivial case is $\Gamma, \varphi\vee\psi\not\to\Delta$. Then, if the formula φ is marked as valid, we change the antisequent to $\Gamma, \varphi\not\to\Delta$, otherwise we change it to $\Gamma, \psi\not\to\Delta$.

One can easily show that all the antisequents obtained in this way are "valid" in the model. Then, the atomic an-

tisequent obtained at the end of the process is an axiom. Clearly, this sequence of antisequents is a skeleton for a proof in PUPC. ■

Theorem 2.2 (UPC). *If there exists a finite model M where all the formulas from Γ are valid and all the formulas from Δ are not, then the antisequent $\Gamma \not\to \Delta$ is provable in* UPC.

Proof. The proof is constructed bottom-up. Given a finite model M with the domain $D=\{a_1,\ldots,a_n\}$, we choose n new variables $x_1,\ldots,x_n$, assigning x_i to a_i. Then we define four provable and valid in M antisequents $\Gamma 0 \not\to \Delta 0$, $\Gamma 1 \not\to \Delta 0$, $\Gamma 2 \not\to \Delta 0$, and $\Gamma 3 \not\to \Delta 0$, which will serve as the definition of the model.

$\Delta 0$ is the set of all the nonvalid atomic formulas on $x_1,\ldots,x_n$,

$\Gamma 0$ is the set of all the valid atomic formulas on $x_1,\ldots,x_n$, without equality,

$\Gamma 1$ is $\Gamma 0 \cup \{x_i = x_i\}$,

$\Gamma 2$ is $\Gamma 1$, $\forall y \bigvee_{i=1}^{n} y = x_i$, and

$\Gamma 3$ is $\Gamma 2 \cup \{t = x_i : t$ appears freely in Γ, Δ and is assigned to a_i in the model$\}$.

Trivially, the antisequent $\Gamma 0 \not\to \Delta 0$ is an axiom. The antisequent $\Gamma 1 \not\to \Delta 0$ is provable from $\Gamma 0 \not\to \Delta 0$ by n $(cut \not\to)$s with the sequents $\cdots \to \cdots, x_i = x_i$. The antisequent $\Gamma 2 \not\to \Delta 0$ is provable from $\Gamma 1 \not\to \Delta 0$ by the $(\forall \not\to)$ rule, and the antisequent $\Gamma 3 \not\to \Delta 0$ is provable from $\Gamma 2 \not\to \Delta 0$ by several applications of the $(= \not\to)$ rule.

One can easily see that M is the only model of $\Gamma 3 \not\to \Delta 0$. We construct by induction on a formula φ in the signature of $\Gamma 3, \Delta 0$ the proof of the sequent $\Gamma 3 \to \Delta 0, \varphi$, if $M \models \varphi$, and the proof of the sequent $\Gamma 3, \varphi \to \Delta 0$, if $M \not\models \varphi$.

Notice that the formulas from Γ are valid in M, whereas those from Δ are not. Then we construct the proof of $\Gamma 3 \cup \Gamma \not\to \Delta 0 \cup \Delta$ by several *cut*s with the formulas from Γ, Δ, and prove $\Gamma \not\to \Delta$ by the *thinning* rule. ■

Remark. For the predicate calculus without equality the proof of Theorem 2.2 should be slightly changed. We have to extend $x_1,\ldots,x_n$ by the free variables and constants from the $\Gamma \not\to \Delta$ giving $(t_1,\ldots,t_m)$, and to change every equality $y = x_i$ in $\Gamma 2$ by its finite "definition" by a formula $\&_{P,k} \forall z_1 \ldots P(z_1,\ldots,z_{k-1}, y, z_{k+1},\ldots) \equiv P(z_1,\ldots,z_{k-1}, x_i, z_{k+1},\ldots)$.
Whenever the above universal formula holds, y and x_i are indistinguishable in the model. This is the usual definition of equality without equality itself. The antisequent $\Gamma 2 \not\to \Delta 0$ is provable in UPC without equality from $\Gamma 0 \not\to \Delta 0$ by a lot of applications of the $(\forall \not\to)$ rule. Of course, in this case we do not need the odd Γ's, i.e. $\Gamma 1$ and $\Gamma 3$.

Notice also that the propositional unprovability system PUPC has the subformula property, and the complexity of a proof computation is very close to that of finding a model. All these make PUPC a bit trivial, whereas the first order unprovability calculus seems, despite its complicacy and unusuality, more promising.

4. Example: loop detecting in Prolog

Let's consider Prolog or deductive data base without function symbols. We can easily find an example of an infinitely looping system of rules. The *transitive closure* seems to be the simplest one. Define for the two-place predicate a its transitive closure tca by the following rules.

$tca(X,X) \leftarrow$

$tca(X,X) \leftarrow a(X,Z), tca(Z,Y)$

The goal we want is $\leftarrow tca(c,d)$, with the ground parameters c, d. Clearly, if $tca(c,d)$ is not true and the relation a has two cycles C_1 and C_2 such that C_1 is accessible from c and d is accessible from C_2, then any resolution based computation of this goal falls into infinite loop. The reason is that there is no finitely failed SLD-tree for these clauses.

Let Π (a "program") comprise of universal closures of all the rules in the rule base. If for proving a goal we apply in parallel two processes, the first one is the usual resolution based proof search, and the second is the unprovability proof search, then the unprovability proof for the antisequent $\Pi \not\to tca(c,d)$ will be found. Notice that the unprovability proof needs participation of *all* the rules from the rule base. This is the consequence of the nonmonotonicity of unprovability.

Looking closer at the model supplied by the unprovability proof, we can see that the predicate tca will be an extension of the transitive closure of a, and the sets $\{x : tc(c,x)\}$ and $\{y : tc(y,d)\}$ may serve as the loop invariants for proving that the resolution proof does not succeed.

Notice however, that yet the unprovability proof system does not give us much more than the usual searching a refutation model.

5. Conclusion

One can see that in the propositional case the unprovability proof system has all the nice properties of the usual sequent or natural deduction calculus. I.e., an antisequent is provable from its subformulas, and the axioms and rules are very natural and intuitively clear. In this sense, the first order unprovability calculus leaves much to be desired. However, considering finite structures as the models for proving unprovability seems to be the most natural, taking into account that the set of all the unprovable first order formulas is not RE. Therefore, introducing a different unprovability proof system it should be advantageous to care for its equivalence to UPC.

Another important problem is developing the unprovability proof system where function symbols are allowed. In this case the Herbrand universe is infinite, and the $(\forall\not\rightarrow)$ rule presented here can not be used. The set of unprovable formulas (provable antisequents) remains RE, and we can easily construct the trivial proof system just taking all the finitely "satisfiable" antisequents as axioms. Still, it will be worth determining the natural unprovability proof system based on a finite number of axiom and rule schemata, similarly to UPC.

The unprovability calculus which is complete in the logic of finite structures allows us to deal more "proof theoretically" with the logics of the *closed world assumption* (CWA), more precisely, with the logics of the *closed finite world assumption* (CFWA). In these logics we are able to prove a "positive" formula φ if it is valid in all the models, or to prove $\neg\varphi$ by proving that φ is not valid in the finite models. Of course, there remains a "hole" of those formulas which are valid in the finite structures, but are refutable in an infinite one. But the fact that the set of provable formulas is not recursive convinces us that there always must be such a hole.

References

[1] J.W. LLoyd, *Foundations of Logic Programming,* Springer-Verlag, Berlin, 1984.

[2] D. McDermott, NonMonotonic Logic II: Nonmonotonic Modal Theories, *Journal of ACM* **29** (1982), 33-57.

[3] R. Reiter, A Logic for Default Reasoning, *Artificial Intelligence* **13** (1980), 81-132.

[4] M. Tiomkin and M. Kaminski, Nonmonotonic default modal logics, TR #487, Dept. of Comp. Sc., Technion, Haifa, Israel, Jan. 1988.

[5] X. Caicedo, A Formal System for the non-Theorems of the Propositional Calculus, *Notre Dame Journal of Formal Logic* **XIX** (1978), 147-151.

[6] W. Schönfeld, Proof Search for Unprovable Formulas, *Proceedings of 7th German Workshop on Artificial Intelligence,* Dasel/Solling, 1983, 207-215.

[7] E. Mendelson, *Introduction to Mathematical Logic*, Van Nostrand Reinhold Co., New York, 1964.

[8] R. Fagin, Generalized First Order Spectra and Polynomial-Time Recognizable Sets, in: R. Carp, ed., *Complexity of Computation, SIAM-AMS Proceedings* (1974), 43-73.

[9] N. Immerman, Languages which Capture Complexity Classes, *Proceedings of 15th ACM Symposium on Theory of Computing,* 1983, 147-152.

[10] M.Y. Vardi, The Complexity of Relational Query Languages, *Proceedings of 14th ACM Symposium on Theory of Computing,* 1982, 137-146.

[11] Y. Gurevich, Toward Logic Tailored for Computational Complexity, in: A. Dold and B. Eckmann, ed., *Computation and Proof Theory, Lecture Notes in Mathematics* **1104** (1983), 175-216.

[12] B.A. Trachtenbrot, Impossibility of an algorithm for the decision problem in finite classes, *Dokladi Akademii Nauk SSSR* **70** (1950), 569-572, in Russian.

[13] G. Gentzen, Untersuchungen über das logische Schliessen, *Mathematische Zeitschrift* **39** (1934-5), 176-210, 405-431, in German.

[14] S.C. Kleene, *Introduction to Metamathematics,* North-Holland, Amsterdam, 1962, 3rd edition.

[15] B. Dreben & W.D. Goldfarb, *The Decision Problem,* Addison-Wesley, Reading, MA, 1979.