Automated Search for Gödel's Proofs

Wilfried Sieg and Clinton Field

Revised September 12, 2003

Technical Report No. CMU-PHIL-144

Philosophy

Methodology

Logic

Carnegie Mellon

Pittsburgh, Pennsylvania 15213

Automated Search for Gödel's Proofs[*]

To appear in: Annals of Pure and Applied Logic

Wilfried Sieg
Clinton Field
Department of Philosophy
Carnegie Mellon University
Pittsburgh

[*] This paper is dedicated to Helmut Schwichtenberg who shares our fascination with automated proof search and who introduced the first author to the intricacies of Gödel's arguments (through lectures in Münster, a long time ago).

Abstract. We present *strategies* and *heuristics* underlying a search procedure that finds proofs for Gödel's incompleteness theorems at an *abstract axiomatic level.* As *axioms* we take for granted the representability and derivability conditions for the central syntactic notions as well as the diagonal lemma for constructing self-referential sentences. The *strategies* are logical ones and have been developed to search for natural deduction proofs in classical first-order logic. The *heuristics* are mostly of a very general mathematical character and are concerned with the goal-directed use of definitions and lemmata. When they are specific to the meta-mathematical context, these heuristics allow us, for example, to move between the object- and meta-theory. Instead of viewing this work as high-level proof search, it can be regarded as a first step in a proof-planning framework: the next refining steps would consist in verifying the axiomatically given conditions. Comparisons with the literature are detailed in section 4. (The general mathematical heuristics are indeed general: in Appendix B we show that they, together with two simple algebraic facts and the logical strategies, suffice to find a proof of "√2 is not rational.")[1]

1. Background. In a genuinely experimental spirit, we extended the *intercalation method for proof search* from pure first-order logic to parts of mathematics by interweaving general logical strategies with specific mathematical heuristics. The guiding question for our investigation was: What is needed, in addition to purely logical considerations, for finding proofs of significant theorems in a fully automated way? We answer the question for Gödel's incompleteness theorems. When proved at an *abstract axiomatic level* they lend themselves naturally to such an investigation; they have intricate, yet not overwhelmingly difficult proofs, and they are obviously significant. During the academic years 1975/77, the first author had taken steps towards establishing them interactively. That work was done for a computer-based course on *Elementary Proof Theory;* a detailed report was given in *Sieg 1978* and a brief summary in *Sieg e.a. 1981.*

Elementary Proof Theory presented the incompleteness theorems for ZF*, that is Zermelo-Fraenkel set theory without the axiom of infinity. Its major innovation consisted in carrying out the meta-mathematical work in a formal theory of binary trees and elementary inductive definitions, called TEM.[2] Without the detour of their arithmetization, the inductively given syntactic notions were shown to be representable in ZF*; the diagonal lemma was established and the proof of the Hilbert-Bernays derivability conditions, central for the second theorem, was sketched. Within that high-level framework the standard material on the incompleteness theorems is compact and the proofs are

[1] Our work was supported by all the members of the current AProS team, in particular by Joseph Ramsey, Orlin Vakarelov, and Ian Kash; we are very grateful.
[2] TEM abbreviates Theory for Elementary Meta-Mathematics. – Feferman systematically investigates in his papers [1982] and [1988] the use of "finitary inductive" definitions in meta-mathematics.

direct. It was natural to ask, whether the proofs can be found via an appropriate extension of the intercalation method.

The arguments for the incompleteness theorems are carried out in the first-order theory TEM: instead of viewing syntactic objects as (having been coded as) natural numbers, we consider them as finitely branching trees; instead of defining syntactic notions recursively, we specify them by elementary inductive definitions, briefly, by eid's. In the language of TEM we have the constant S for the empty tree and the function symbol [,] for the binary operation of building a tree from two given ones. We use X, Y, Z - possibly with indices - as variables ranging over binary trees. The axioms for S and [,] are formulated in analogy to those of Dedekind-Peano arithmetic for zero and successor. The further axioms of TEM include the induction principle for binary trees, and closure and minimality conditions for the eid's. Instead of discussing these axioms in generality – the details do not matter for the current project – we specify some definitions that are actually needed to characterize the formal theory for which the incompleteness theorems are to be proved.

The theory to be considered is ZF*, Zermelo and Fraenkel's theory of sets without the axiom of infinity. The details of its axiomatic formulation do not matter either for the current project. Let us assume that it is formulated in a first-order language with x, y, z – possibly with indices – as variables ranging over sets. To indicate the general character of eid's we specify the generating clauses of the familiar notion of a formula (taking for granted the concepts of atomic formula and of variable); @ stands for any binary sentential connective, Q for the existential or universal quantifier:

If X is an atomic formula, X is a FORMULA;
If X is a FORMULA, [~, X] is a FORMULA;
If X is a FORMULA and Y is a FORMULA, [@, [X, Y]] is a FORMULA;
If X is a variable and Y is a FORMULA, [[Q, X], Y] is a FORMULA.

We write also "FORM(X)" for "X is a FORMULA." TEM contains for such eid's a *closure* and a *minimality* principle. The first principle asserts that FORM is closed under the above clauses and is expressed by

FOR ALL X (if $\mathcal{A}$(FORM, X) then FORM(X)).[3]

The minimality principle claims that FORM is the smallest such class. This is approximated in first-order logic by the usual principle of induction for formulas:

If FOR ALL X (if $\mathcal{A}$ (P, X) then P(X))

then FOR ALL X (if FORM(X) then P(X)).

Formulas are binary trees built up from the empty tree using pairing. In a similar way one can generate inductively the relation X *is a proof of* Y *from assumptions* Z_1, ..., Z_n or from a (n inductively generated) class of axioms; if X is a proof of Y using axioms of ZF*, this relation is denoted by PROOF(X,Y). To indicate that there is a ZF*-proof for Y, we write ZF* | -(Y), ZF* | -Y or THEO(Y).

Using the constant ∅ and the set-theoretic pairing operation < , > one can build up terms in the language of ZF* whose parse trees are isomorphic to the binary trees; they are used as names for the meta-mathematical trees in the same way as numerals in Dedekind-Peano arithmetic are used as names for natural numbers. With every meta-mathematical tree we can directly associate its set-theoretic name or *code*: CODE(S) = ∅ and CODE([X,Y]) = <CODE(X), CODE(Y)>. We also write |_X_| for CODE(X) or indicate it by **X**. This is the apparatus needed to formulate the *representability conditions* for the syntactic notions. We give them paradigmatically for FORM and PROOF:

If FORM(X) then ZF* | - form(**X**), and

If NOT FORM(X) then ZF* | - ~form(**X**);

"form" is a formula in the language of set theory for which these conditions are provable in TEM. Similarly, there is a formula "proof" in the language of ZF* that represents the proof relation PROOF:

If PROOF(X,Y) then ZF* | - proof(**X**,**Y**), and

If NOT PROOF(X,Y) then ZF* | - ~proof(**X**,**Y**).

Using the first representability condition for PROOF one can establish:

If THEO(Y) then ZF* | - theo(**Y**),

[3] $\mathcal{A}$(P,X) is obtained from the generating clauses; it is the disjunction of the following TEM-formulas: (i) X is atomic; (ii) $(X)_0$ is ~ and $P((X)_1)$; (iii) $(X)_0$ is @ and $P(((X)_1)_0)$ and $P(((X)_1)_1)$; (iv) $((X)_0)_0$ is Q and $((X)_0)_1$ is a variable and $P((X)_1)$. P can be viewed as either a meta-variable over TEM-formulas or as a free second-order variable; under the second reading we have an appropriate substitution rule in the logical calculus for TEM.

where theo(y) abbreviates (Ex) proof(x,y).[4] Finally, we will use the *Self-reference Lemma* (or *Diagonal Lemma*) in the form: if F is a formula in the language of set theory (with one free variable), then there is a sentence D_F in that very language such that ZF* proves (D_F <-> F($\mathbf{D_F}$)). Applied to the formula ~theo(y), the self-reference lemma yields the Gödel sentence G that expresses its own unprovability, i.e., ZF* proves (G <-> ~theo(**G**)).

With this systematic background it is not difficult to prove that G is not provable in ZF* assuming, of course, that ZF* is consistent. So let us assume – in order to obtain a contradiction – that ZF* proves G; then, by the diagonal lemma concerning G, ZF* proves ~theo(**G**). On the other hand, by the (semi-) representability of THEO, we can infer from the fact that ZF* proves G, that ZF* establishes theo(**G**). Thus, ZF* proves both ~theo(**G**) and theo(**G**), and we have obtained a contradiction! The independence of G requires a proof that ~G is not provable either; for that a stronger assumption concerning ZF*, stronger than mere consistency, has to be made. Gödel used for that purpose the notion of ω-consistency; the corresponding concept for the context of our meta-mathematical set-up is τ-consistency, thinking of τ as the class of (sets denoted by codes for) binary trees. ZF* is *τ-consistent* is defined by the condition: there is no formula F(y) such that ZF* proves (Ey) (τ(y) & F(y)) and also ~F(**Y**) for all Y; or equivalently, for all formulas F(y), if ZF* proves ~F(**Y**) for all Y, then ZF* does not prove (Ey) (τ(y) & F(y)).

Assuming that ZF* is τ-consistent, we show now that ZF* does not prove the negation of the Gödel sentence G. By what we established already (and the fact that τ-consistency implies ordinary consistency) we know that

FOR ALL X: NOT PROOF(X,G);

the representability of PROOF implies

FOR ALL X: ZF* l - ~proof(**X**,**G**).

But then the τ-consistency of ZF* ensures

NOT ZF* l - (Ey) proof(y,**G**).

As the formula (Ey) proof(y,**G**) is abbreviated by theo(**G**), we can use the self-reference lemma for G to infer that this formula is in ZF* provably equivalent to

[4] The existential quantifier here is E, rather than the standard symbol ∃, to reflect the notation used for ZF* in APROS. In addition, the universal quantifier is A.

~G. Thus, NOT ZF* | -(~G), and the independence of G from ZF* has been established.

Given the axiomatic context provided by the representability of PROOF and THEO and the self-reference lemma applied to ~theo(y), the proofs are direct, yet intricate. To take a first step towards describing the search algorithm that finds proofs of these and related theorems, we present briefly the basic ideas underlying the intercalation method for classical logic; for the theoretical underpinnings we refer to *Sieg 1992*, *Sieg & Byrnes 1998* and *Byrnes 1999*. We should emphasize at this point that, in our view, logical formality per se does not facilitate the finding of proofs. However, logic within a natural deduction framework does help to bridge the gap between assumptions and conclusions by suggesting very rough structures for arguments, i.e., *logical structures* that depend solely on the syntactic form of assumptions and goals. This role of logic, though modest, is the crucial starting-point for moving up to subject-specific considerations that support a theorem. In the case study at hand we will show, how far these logical considerations go, and how they can be extended quite naturally by the *leading mathematical ideas* underlying Gödel's proofs.

2. Intercalation: broad strategies & special heuristics. The intercalation method is a proof search procedure that is goal-directed and guided by the possibly expanding syntactic context of the problem at hand. In first-order logic it is a complete procedure and a basis for broad logical strategies. The fundamental idea is straightforward. In order to bridge the gap between premises $A_1, \ldots, A_n$ and a goal B, one applies *systematically* the rules of the natural deduction calculus. I.e., the elimination rules are applied only from "above," whereas the introduction rules are inverted and applied from "below." Such systematic applications of the rules generate a search space that either contains a proof of B from the assumptions $A_1, \ldots, A_n$ or provides a semantic counterexample to the claim that B is a logical consequence of $A_1, \ldots, A_n$ - tertium non datur; in addition, proofs contained in the search space are necessarily normal. The argument for this sharpened completeness theorem provides a method for searching directly for normal proofs; indeed, it yields also a semantic argument for normal form theorems in natural deduction. Such arguments concerning classical first-order logic were first given in *Sieg 1992,* later also for intuitionistic logic and some modal logics in collaboration with Cittadini.

Normal proofs satisfy a similar *subformula property* as cut-free derivations in the sequent calculus. That, of course, allows a restriction of the systematic search and is basic for broad strategies underlying our proof search: (i) extracting B via elimination rules – if B is a strictly positive subformula of an assumption, (ii) sub-goaling via the appropriate inverted introduction rule – if B is a logically complex formula, (iii) refuting B via the rules for negation – if B is a negation or an atomic formula and if an appropriate pair of contradictory formulas is available. In the latter case there must be a negation that is a strictly positive subformula of an assumption. It is evident that direct proof search is strongly and naturally constrained by the syntactic context of the problem, as only particular subformulas can be intercalated between assumptions and goals.

With these logical strategies in the background let us return to the proof of the first part of the first incompleteness theorem and examine, how the intercalation method might find it with "a little help" (when pure logic is unable to proceed any further). So we begin with the goal NOT (ZF* |-(G)) and the premise ZF*CONS. We also have a definition and a lemma available, namely, the definition

ZF*CONS IFF NOT [ZF* |-(G) AND ZF* |-(~G)]

and the consequence of the diagonal lemma for ~theo(x), i.e.,

ZF* |-(G<->~theo(**G**)).[5]

The goal cannot be extracted from the premises. Thus, the algorithm proceeds indirectly with the assumption ZF* |-(G) and needs a pair of contradictory formulas as new goals. However, no negation occurs as a strictly positive subformula of the premise. As there is a negation in the definition of the premise, we use it and the premise to infer

NOT [ZF* |-(G) AND ZF* |-(~G)].

This negation is one element of a contradictory pair, and the algorithm attempts to prove [ZF* |-(G) AND ZF* |-(~G)]. This formula cannot be extracted: even

[5] We could have chosen one of the more general formulations of consistency, for example, NOT (EXISTS X) (ZF* |-(X) AND ZF* |-(~X)). The quantificational search in the SH-expansion (see Sieg and Byrnes) would find the appropriate instance quickly.

though it is a subformula of a premise, it is not a strictly positive one. So the algorithm inverts the formula and attempts to prove the new goals ZF* I -(G) and ZF* I -(~G). The former goal is already an assumption of the indirect proof, so we examine the latter goal.

It is here that we make the first significant change to the proof search procedure. ZF* I -(~G) cannot be extracted, but as an existential formula it can be inverted. Instead of searching for a term in the language of TEM describing a ZF*-proof of ~G, the search proceeds "inside" ZF*. The claim ZF* I -(~G) can be justified, after all, by the presentation of a proof of ~G within ZF*. The procedure tries now to find a ZF*-proof for the goal ~G. As the formula ~G cannot be extracted, indirect proof is applied to ~G: assume G and find a contradictory pair. There is no negation immediately available in the premises, except through the diagonal lemma for G. Note that this lemma is formulated within TEM as a provability claim for ZF* and should be available for any ZF*-proof. In general, when attempting an extraction or looking for contradictory pairs within a ZF*-proof, strictly positive subformulas of ZF*-formulas A must be considered, where ZF* I -(A) occurs as a strictly positive subformula of a premise or available assumption in TEM. So, the diagonal lemma makes available the formula ~theo(**G**), which is used to construct the contradictory pair. This leaves theo(**G**) as a new goal, which cannot be extracted. The regular proof search procedure would attempt an inversion. But here an additional step can be considered, since theo is a semi-representable relation: we can justify theo(**G**) by establishing ZF* I -(G) in TEM. ZF* I -(G) is an assumption in TEM, so the proof is complete.

The expanded version of the proof search algorithm, which results from the careful examination of the above proof, interweaves mathematical and purely logical considerations in an intercalating and goal-directed manner. It has the following main steps:

Extraction. If the goal is in TEM, then extraction functions as described above for first-order logic. If the goal is in ZF*, then the set of formulas available for extraction is expanded by those formulas A, for which the claim ZF* I -(A) is extractable in TEM and the goal is extractable from A. That is the inference ProvE, which is used to turn A into a part of the ZF*-proof.

Inversion. For the standard connectives inversion is applied as discussed earlier. There are two additional cases where "inversion" is applied. The first case occurs, when the goal in TEM is a statement of the form ZF* | -(A). Here the algorithm tries to find a proof of A in ZF*; that is the inversion of the inference ProvI.[6] In the second case, when the goal is a formula like [~] rel(**X**) in ZF*, and when the relation REL is represented by rel, the procedure tries to prove [NOT] REL(X) in TEM, after having explored indirect strategies in ZF*. For semi-representable relations such as ZF* | -(X), this step is obviously not applied to the negation ~rel(**X**) in ZF*.

Extended extraction and inversion ("Meaning of premises and goals"). Definitional and other mathematical equivalences are used to obtain either a new available formula from which the current goal is extractable or to get an equivalent statement as a new goal. This we would like to do relative to a developing background theory; currently, we just add the definitions and lemmata explicitly to the list of premises.

Indirect strategies are pursued in the same way as in pure first-order logic, with one exception: the set of contradictory pairs for indirect proofs in ZF* is expanded by pairs whose negations are strictly positive subformulas of A in case ZF* | -(A) (and this TEM-statement is itself extractable from an available TEM-claim.)

This completes the informal description of the algorithm that searches for statements surrounding the first incompleteness theorem. The extensions of extraction and inversion mentioned have a very general mathematical character, whereas the extensions via ProvE and ProvI express most directly meta-mathematical content. The former rule reflects, in part, that theorems can be appealed to in proofs, and the latter rule expresses that the search mechanism provides syntactically correct object theoretic proofs.

The extended search procedure evolved out of a probing analysis of the standard proofs for the first incompleteness theorem and incorporates what we take to be the *leading mathematical ideas* for this part of meta-mathematics. It finds proofs not only for the first and second incompleteness theorems (after

[6] If the goal is of the form ZF* | - ([~] rel(X)), the algorithm tries first to prove [NOT] REL(X) directly.

incorporating the derivability conditions), but also for a broader range of theorems and lemmata in this general area; cf. Appendix **A** for a proof of Löb's Theorem and Appendix **D** for two further examples. Even without the specifically meta-mathematical steps the algorithm is of real mathematical interest, as it discovers the structure of the proof for the irrationality of the square root of 2; see Appendix **B**.

3. Machine proofs & new heuristics. We present now the proofs of the first and second incompleteness theorem and start out by explaining the format of proofs. Proofs are presented in a modified Fitch-style format, which can be given using only plain text. We show the scope of assumptions by inserting bars between the number and formula on each line, with nested assumptions being noted by alternating bars and exclamation points. A line of dashes sets off the assumptions themselves. To distinguish the parts of the proof which occur in TEM and those which are embedded ZF*-proofs, we mark every line in the object language with a star. Note that ZF*-proofs retain the scope indications from the meta-language, and appeals to representability will use all available TEM-assumptions.

The rules include the standard natural deduction rules. For example, conjunction introduction has the name "AndI", and the left and right-hand versions of conjunction elimination are named "AndEL" and "AndER" respectively. To these basic rules we add special rule names for every heuristically applied theorem or lemma. "Rep" names the rule for representable or semi-representable relations, where the premise is a representable relation in TEM and the conclusion the corresponding relation in ZF*. "ProvE" and "ProvI" indicate provability elimination and introduction.

We present first the machine proof of non-provability of the Gödel sentence G, assuming that ZF* is consistent. In addition, the machine uses an instance of the diagonal lemma ZF* |-(G <-> ~(theo(**G**))) and the definition of consistency, ZF*CONS IFF NOT(ZF* |-(G) AND ZF* |-(~(G))).

Proof:[7]

[7] When following this argument and all the other machine proofs, the reader should keep in mind the intercalation strategies for bridging the gap between assumptions and goals. After all, they motivate the steps in the arguments.

```
 1.  ZF* |-(G <-> ~(theo(G)))                        Premise
 2.  ZF*CONS                                         Premise
 3.  ZF*CONS IFF NOT(ZF* |-(G) AND ZF* |-(~(G)))     Premise
 4.  | ZF* |-(G)                                     Assumption
     | ------------
*5.  | ! G                                           Assumption
     | ! ------------
*6.  | ! theo(G)                                     Rep 4
*7.  | ! (G <-> ~(theo(G)))                          ProvE 1
*8.  | ! ~( theo(G))                                 IffER 7, 5
*9.  | ~(G)                                          NotI 5, 6, 8
10.  | ZF* |-(~(G))                                  ProvI 9
11.  | ZF* |-(G) AND ZF* |-(~(G))                    AndI 4, 10
12.  | NOT(ZF* |-(G) AND ZF* |-(~(G)))               IffER 3, 2
13.  NOT(ZF* |-(G))                                  NotI 4, 11, 12
```

q.e.d.

To prove the independence of G we have also to establish the non-provability of ~G. As remarked earlier, that requires the stronger hypothesis of τ-consistency. Here are the premises for the non-provability of ~G: the diagonal lemma ZF* |-(G <-> ~(theo(**G**))), ZF*τCONS, ZF*τCONS IMPLIES [(FORALL X)(ZF* |-(~(proof(**X**,**G**))) IMPLIES NOT(ZF* |-(theo(**G**))], ZF*τCONS IMPLIES ZF*CONS, and a reformulation of what was established above, namely ZF*CONS IMPLIES (FORALL X)(NOT(PROOF(X,G))).

Proof:

```
1.  ZF* |-(G <-> ~(theo(G)))                      Premise
2.  ZF*τCONS                                      Premise
3.  ZF*τCONS IMPLIES
    [(FORALL X)(ZF* |-(~(proof(X,G)))
                IMPLIES NOT(ZF* |-(theo(G))]      Premise
4.  ZF*τCONS IMPLIES ZF*CONS                      Premise
5.  ZF*CONS IMPLIES
        (FORALL X)(NOT(PROOF(X,G))))              Premise
6.  | ZF* |-(~(G))                                Assumption
    | ------------
```

*7. | ! ~(theo(**G**)) Assumption

| ! ------------

*8. | ! (G <-> ~(theo(**G**))) ProvE 1

*9. | ! G IffEL 8, 7

*10. | ! ~(G) ProvE 6

*11. | theo(G) NotE 7, 9, 10

12. | ZF* | -(theo(**G**)) ProvI 11

13. | (FORALL X)(ZF* | -(~(proof(**X**,**G**)))
IMPLIES NOT(ZF* | -(theo(**G**))) ImpE 3, 2

14. | ZF*CONS ImpE 4, 2

15. | (FORALL X)(NOT(PROOF(X,G))) ImpE 5, 14

16. | NOT(PROOF(X,G)) AllE 15

*17. | ~(proof(**X**,**G**)) Rep 16

18. | ZF* | -(~(proof(**X**,**G**)) ProvI 17

19. | (FORALL X)(ZF* | -(~(proof(**X**,**G**))) AllI 18

20. | NOT(ZF* | -(theo(**G**))) ImpE 13, 19

21. NOT(ZF* | -(~(G))) NotI 6, 12, 20

q.e.d.

For the proof of the *second incompleteness theorem*, i.e., the non-provability of the formal consistency statement zf*cons under the assumption of the consistency of ZF*, the formalism has to satisfy the *Hilbert-Bernays derivability conditions* D_1 and D_2. D_1 is the formalized semi-representability condition for the theorem predicate [theo(x) -> theo(**theo(X)**)], whereas D_2 is the provable closure under modus ponens [theo(**X -> Y**) -> (theo(**X**) -> theo(**Y**))]. The algorithm makes use of these conditions as rules with one additional heuristic to exploit D_2: if theo(**F**) is the goal and F, as a consequent of a conditional (or biconditional), is a strictly positive subformula of an available purely implicational formula, apply D_2 repeatedly and try to extract theo(**F**).

Proof:

1. ZF* | -(theo(**G**) <-> ~G)) Premise[8]

2. ZF* | -(zf*cons <-> ~(theo(**G**) & theo(**~G**))) Premise

3. NOT(ZF* | -(G)) Premise

[8] Notice that the diagonal lemma is used here in a propositionally equivalent form; the current algorithm does not find the proof, when it is given in its standard form.

4.	| ZF* | -(zf*cons)	Assumption
	| ------------	
*5.	|! ~(G)	Assumption
	|! ------------	
*6.	|! (theo(**G**) <-> ~G)	ProvE 1
*7.	|! theo(**G**))	IffEL 6, 5
*8.	|! theo(**theo(G)**)	Der$_1$ 7
*9.	|! theo(**theo(G)**) -> theo(~**G**))	Der$_2$ 6
*10.	|! theo(~**G**)	ImpE 9, 8
*11.	|! theo(**G**) & theo(~**G**)	AndI 7, 10
*12.	|! (zf*cons <-> ~(theo(**G**) & theo(~**G**)))	ProvE 2
*13.	|! zf*cons	ProvE 4
*14.	|! ~(theo(**G**) & theo(~**G**))	IffEL 12, 13
*15.	| G	NotE 5, 11, 14
16.	| ZF* | -(G)	ProvI 15
17.	NOT(ZF* | -(zf*cons))	NotI 4, 17, 3

q.e.d.

This argument made use of the special character of the Gödel sentence G – in order to obtain the two conjuncts of line *11. Instead, one can exploit the elegant way of proceeding made possible by *Löb's theorem*:

For all sentences F: ZF* | -(theo(**F**) -> F) IFF ZF* | -(F).

Löb's theorem expresses that a sentence F is provable in ZF* if and only if its *reflection formula* (theo(**F**) -> F) can be established in ZF*. Consider a *refutable* sentence H (i.e. a sentence whose negation is provable in ZF*) and assume that ZF* is consistent; then H is not provable in ZF*. Löb's theorem implies that the corresponding reflection formula (theo(**H**) -> H) is not provable either. Thus, the *second incompleteness theorem* amounts to establishing NOT(ZF* | -(zf*cons)) from the premises NOT(ZF* | -(theo(**H**)->H)), ZF* | -(zf*cons<->~(theo(**H**) & theo(~**H**))), and ZF*|-(~H). That is done in the next proof.

Proof:

1.	NOT(ZF* | -(theo(**H**) -> H))	Premise
2.	ZF* | -(zf*cons <-> ~(theo(**H**) & theo(~**H**)))	Premise
3.	ZF* | -(~H)	Premise
4.	| ZF* | -(zf*cons)	Assumption

```
      | ------------
 *5.  |! theo(H)                                  Assumption
      |! ------------
 *6.  |!| ~(H)                                    Assumption
      |!| ------------
 *7.  |!| theo(~H))                               Rep 3
 *8.  |!| theo(H) & theo(~H)                      AndI 5, 7
 *9.  |!| (zf*cons <-> ~(theo(H) & theo(~H)))     ProvE 2
*10.  |!| zf*cons                                 ProvE 4
*11.  |!| ~(theo(H) & theo(~H))                   IffER 9, 10
*12.  |! H                                        NotE 6, 8, 11
*13.  | theo(H) -> H                              ImpI 5, 12
 14.  | ZF* |-(theo(H) -> H)                      ProvI 13
 15.  NOT(ZF* |-(zf*cons))                        NotI 4, 14, 1
```

q.e.d.

This proof of the second incompleteness theorem uses Löb's Theorem only in the discussion leading up to the precise derivational problem. In Appendix **A** the preliminary considerations are incorporated into the proof; there we also show an elegant machine proof of Löb's Theorem.

4. Comparisons. A number of researchers have pursued goals similar to ours, but with interestingly different programmatic perspectives and strikingly different computational approaches. We focus on work by Ammon, Quaife, Bundy e.a. [1996], and Shankar. We first discuss Ammon's and Quaife's work, as theirs is programmatically closest to ours: Ammon aims explicitly for a *fully automatic* proof of the first incompleteness theorem, and Quaife establishes the incompleteness theorems and Löb's theorem in a setting that is similarly "abstract" as ours.

In his 1993 Research Note *An automatic proof of Gödel's incompleteness theorem,* Ammon describes the SHUNYATA program and the proof it found for the first incompleteness theorem. SHUNYATA's proof is structurally identical with the proof in Kleene's book *Introduction to Metamathematics* (pp. 204-8); the latter proof is discussed in great detail in sections 4 and 5 of Ammon's note. Two main claims are made: (i) Gödel's undecidable sentence is "constructed" by the program "on the basis of elementary rules for the formation of formulas," and this is taken as evidence for the subsidiary claim (on p. 305) that the program "implicitly rediscovered Cantor's diagonal method;" (ii) the proof of its undecidability is found by a heuristically guided *complete proof procedure* involving Gentzen's natural deduction rules for full first-order logic. The first claim (made on p. 291 and reemphasized on p. 295) is misleading: the Gödel sentence is of course constructible by the elementary rules for the (suitably extended) language of number theory, but that the formula so constructed expresses its unprovability has to be ensured by other means (and is "axiomatically" required to do so by Ammon's definition 3 and lemma 1).[9] As to the second claim (made on p. 294), the paper contains neither a logical calculus nor a systematic proof procedure using the rules of the calculus. What one finds are local heuristics for analyzing quantified statements and conditionals together with directions to prove the negation of a statement, i.e., to use the not introduction rule. These latter directions are quite open-ended, as there is no mechanism for selecting appropriate contradictory pairs. (Cf. Ammon's discussion of the "contradiction heuristic" on p. 296.)

In 1988 Quaife had already published a paper on *Automated proofs of Löb's Theorem and Gödel's two incompleteness theorems*. The paper presents proofs of the theorems mentioned in its title[10] "at a suitable level of abstraction" - as the author emphasizes on p. 219 - "from the underlying details of Gödel numbering and of recursive functions." The suitable level of abstraction is provided by the provability logic K4. That well-known logic contains as special axioms the derivability conditions and as its special rule (beyond modus ponens) the rule of "necessitation;" the additional rule corresponds to the semi-representability of the theorem predicate. In order to make use of the resolution theorem proving system ITP, the first-order metatheory of K4 is represented in ITP by five "clauses," which are listed in Appendix C. Four of the clauses correspond to the axioms and rules just mentioned, whereas the very first clause guarantees that all tautologies are obtained. The tautologies are established by "applying properly

[9] Our assessment of this claim is in full agreement with that found in the *Letter to the Editor* by Brüning e.a..

[10] Quaife establishes only the unprovability of G, not of its negation under the assumption of ω-consistency. On p. 229 he asserts, "With the right axioms, its proof [i.e., the other half of the first incompleteness theorem, S&F] could be reproduced about as easily as the principal half above."

specified demodulators" and transforming given sentential formulas into conjunctive normal form; the underlying procedure is complex and involves particular weighting schemes. Quaife illustrates the procedure by presenting on pp. 226-7 a derivation of a "reasonably complex tautology;" the derivation uses a sequence of 73 demodulation steps. Quaife concludes the discussion of this derivation by saying: "ITP can also be asked to print out the line-by-line application of each demodulator, but that detailed proof is too long for this article." We present this tautology and its direct (and easily found) natural deduction proof in Appendix **C**.

In contrast to Ammon's paper, we find here a conceptually and technically straightforward meta-mathematical and logical set-up: representability and derivability conditions are axiomatically assumed, and the logical inference machinery is precisely and carefully described. However, it is very difficult to understand, how the syntactic context of axioms, theorems and assumptions directs the search in a way that is motivated by the leading ideas of the mathematical subject.[11] The proofs use in every case "axioms and previously proven theorems" in addition to the standard hypotheses for the theorem under consideration. It is clear that the "previously proven theorems" are strategically selected, and it is fair to ask, whether the full proof – from axioms through intermediate results to the meta-mathematical theorems – should be viewed as "automated" or rather as "interactive" with automated large logical steps. So the direct computational question is, would proofs of the main theorems be found, if only the axioms were available?

The answer is most likely "No." OTTER, the resolution theorem prover that developed out of ITP, was not able to prove, under appropriately similar conditions, the full first incompleteness theorem in 1996; that is reported in Bundy, Giunchiglia, Villafiorita and Walsh's paper *An incompleteness theorem via abstraction.*[12] It was precisely this computational problem that motivated their paper, namely to show how "abstraction" can be useful to attack it. They present a proof of Gödel's theorem, where the real focus is not on the particular meta-

[11] A similar reservation is articulated by Fearnley-Sander in his review of Quaife's book.

[12] On p. 10 they write: "This proof [of the full first incompleteness theorem; S&F] turns out to be a considerable challenge to an unguided theorem prover. We have given these axioms to OTTER (v. 3.0) ... but it blew up."

mathematical proof, but rather on the process of abstraction and refinement that aids proof planning. This process is not a fully automated one, since both the choice of the abstraction and the subsequent refinement of the abstract proof into the original language require external guidance. While we share the ultimate goal of limiting the search space for mathematical proofs by "abstraction," their semi-automated abstraction process is a very different, though complementary approach.

The three approaches we have been discussing are as "abstract" as ours in the sense that the diagonal lemma, the representability condition and, in Quaife's and our case, the derivability conditions are taken for granted. Shankar's book *Metamathematics, Machines, and Gödel's Proof* focuses on an interactive proof of (the Rosser version of) the first incompleteness theorem.[13] The explicit goal was to find out, whether the full proof could *in practice be checked* using a computer program, i.e., the Boyer-Moore theorem prover. In the preface to his book Shankar points out that "A secondary goal was to determine the effort involved in such a verification, and to identify the strengths and weaknesses of automated reasoning technology." The crucial meta-mathematical task and most significant difficulty consisted in verifying the representability conditions - for a particular theory (the system Z_2 for number theory in Cohen's book) and a particular way of making computability precise (via McCarthy's Lisp). That required, of course, a suitable formalization of all meta-mathematical considerations within, what Shankar calls on p. 141, "a constructive axiomatization of pure Lisp." In sections 5.4 and 5.5 Shankar gives a very informative analysis of, and an excellent perspective on, the work presented.

Moving back from interactive theorem proving to automated proof search, it is clear that the success of our search procedure results from carefully interweaving mathematical and logical considerations, which lead from explicitly formulated principles to a given conclusion. Proofs provide *explanations* of what they prove by putting their conclusions in a context that shows them to be correct. This need not be a global context providing a foundation for all of mathematics, but it can be a rather more restricted one as

[13] In addition, Shankar provides a "mechanical proof" of the Church-Rosser Theorem in Chapter 6.

here for the presentation of the incompleteness theorems. Such a local deductive organization is *the* classical methodology of mathematics with two well-known aspects: the formulation of principles and the reasoning from such principles; we have illustrated only the latter aspect by using suitable strategic considerations and appropriate heuristic "leading mathematical ideas."

The task of considering a part of mathematics, finding appropriate basic notions, and explicitly formulating principles – so that the given part can be systematically developed – is of a quite different character. For Dedekind the need to introduce new and more appropriate notions arises from the fact that human intellectual powers are imperfect. The limitation of these powers leads us, Dedekind argues, to frame the object of a science in different forms or different systems. To introduce a notion, "as a motive for shaping the systems," means in a certain sense to formulate a hypothesis concerning the inner nature of a science, and it is only the further development that determines the real value of such a notion by its greater or smaller *efficacy* (Wirksamkeit) in recognizing general truths. In the part of meta-mathematics we have been considering, Hilbert and Bernays did just that: their formulation of representability and derivability conditions ultimately led to more "abstract" ones and, in particular, to the principles for the provability logic K4 and related systems; see (Boolos 1993).[14]

5. Concluding remarks. No matter how one might mechanize an attempt of gaining such a principled deeper understanding of a part of mathematics, the considerations for a systematic and efficient automated development would still be central. In our given meta-mathematical context, there is an absolutely natural step to be taken next. As we emphasized earlier, there is no conflict or even sharp contrast between proof search and proof planning: proof search is hierarchically and heuristically organized through the use of "axioms" and their subsequent verification (or refutation). The guiding idea for verification in the intercalation approach is to generate sequences of formulas, reduce differences,

[14] In a different, though closely related case, Hilbert and Bernays succeeded in providing "recursiveness conditions" for the informal concept of calculability in a deductive formalism; that was done in a supplement of the second volume of their *Grundlagen der Mathematik.*

and arrive ultimately at syntactic identities. Such difference reduction also underlies the techniques for inductive theorem proving that have been developed by Bundy e.a. in their recent book. We conjecture that those techniques can be seamlessly joined with the intercalation method to take the *next step* and prove the representability conditions. The strictly formal proof in TEM might then be transformed into a ZF* proof of the first derivability condition, automatically. – From a different, more proof-theoretic perspective one might wish to compare the intercalation method for natural deduction calculi with appropriately formulated methods for sequent calculi with and without cuts. That might lead to interesting heuristics for choosing suitable cut formulas (to make proof search more efficient).[15]

[15] This issue was suggested as a good research direction by an anonymous referee.

BIBLIOGRAPHY

Ammon, Kurt

1993 An automatic proof of Gödel's incompleteness theorem; Artificial Intelligence 61, 291-306.

Boolos, George

1993 *The logic of provability*; Cambridge University Press.

Brüning, S., M. Thielscher, and W. Bibel

1993 Letter to the editor; Artificial Intelligence 61, 353-4.

Bundy, Alan, Fausto Giunchiglia, Adolfo Villafiorita, and Toby Walsh

1996 An incompleteness theorem via abstraction; Technical Report #9302-15; Istituto per la ricerca scientifica e tecnologica, Trento.

Bundy, Alan, David Basin, Dieter Hutter, and Andrew Ireland

2003 *Rippling: Meta-level guidance for mathematical reasoning*; Book manuscript.

Byrnes, John

1999 Proof search and normal forms in natural deduction; Ph.D. Thesis; Department of Philosophy, Carnegie Mellon University.

Cohen, Paul J.

1966 *Set theory and the continuum hypothesis*; Benjamin, Reading, Mass.

Dedekind, Richard

1854 Über die Einführung neuer Funktionen in der Mathematik; Habilitationsrede; 428-38; in: *Gesammelte mathematische Werke* (Fricke, Noether and Ore, editors), vol. 3, Vieweg, 1933.

Fearnley-Sander, Desmond

Review of *Quaife 1992*; http://psyche.cs.monash.edu.au/

Feferman, Solomon

1982 Inductively presented systems and the formalization of meta-mathematics; in: *Logic Colloquium '80*, van Dalen, Lascar, Smiley (eds.), North-Holland Publishing Company, 95-128.

1988 Finitary inductively presented logics; in: *Logic Colloquium '88*, Ferro e.a. (eds.), North-Holland Publishing Company, 191-220.

Fitch, F.

1952 *Symbolic Logic*; The Ronald Press Company, New York.

Gödel, Kurt

1931 Über formal unentscheidbare Sätze der Principia mathematica und verwandter Systeme I; Monatshefte für Mathematik und Physik 38, 173-198.

Löb, M.

1955 Solution of a problem of Leon Henkin; J. Symbolic Logic, 20, 115-8.

Quaife, Art

1988 Automated proofs of Löb's theorem and Gödel's two incompleteness theorems; Journal of Automated Reasoning 4, 219-231.

1992 *Automated Development of Fundamental Mathematical Theories*; Kluwer Academic Publishers.

Shankar, N.

1994 *Metamathematics, Machines, and Gödel's Proof*; Cambridge Tracts in Theoretical Computer Science 38, Cambridge University Press.

Sieg, Wilfried

1978 *Elementary proof theory*, Technical Report 297, 104 pp., Institute for Mathematical Studies in the Social Sciences, Stanford.

1992 *Mechanisms and Search* (Aspects of Proof Theory); AILA Preprint.

Sieg, Wilfried and John Byrnes

1998 Normal natural deduction proofs (in classical logic); Studia Logica 60, 67-106.

Sieg, Wilfried and Saverio Cittadini

2002 Normal natural deduction proofs (in non-classical logics); Technical Report No. CMU-PHIL-130, 29 pp.

Sieg, Wilfried, Ingrid Lindstrom, and Sten Lindstrom

1981 Gödel's incompleteness theorems - a computer-based course in elementary proof theory; in: *University-Level Computer-Assisted Instruction at Stanford 1968-80*, P. Suppes (ed.), Stanford, 1981, 183-193.

APPENDICES

A. Löb's theorem. The context of the theorem is given in section 3. Here we present an argument obtained by our automated proof search and re-prove the second incompleteness theorem; in the latter proof, the appeal to Löb's theorem is explicitly built into the argument.

In order to prove Löb's theorem in TEM, one faces two claims, namely,

(i) ZF* I -(theo(**F**) -> F) IMPLIES ZF* I -(F)

and

(ii) ZF* I -(F) IMPLIES ZF* I -(theo(**F**) -> F).

The last claim is immediate, whereas the first is difficult: its proof uses the instance of the diagonal lemma for the formula (theo(x) -> F). Here is the precise derivational problem at the heart of Löb's theorem: ZF* I -(F) can be proved from the premises ZF* I -(theo(**F**) -> F) and ZF* I -(L <-> (theo(**L**) -> F)).

We actually have two proofs of Löb's theorem, which differ in the presentation of the derivability conditions. In the first proof the conditions are formulated as premises and are instantiated for this problem. They enter the search through the standard extraction procedure. In the second proof heuristics guide their application. The heuristics were described above and have a fairly general character; they are designed to apply each condition when it may be useful. The resulting proofs are very similar, differing mainly in the greater number of extraction rule applications necessary in the first proof to make use of the axiomatically given derivability conditions. We present only the first proof.

Proof

1. ZF* I -(L <-> (theo(**L**) -> F)) — Premise

2. ZF* I -(theo(**L**) -> (theo(**theo(L)**) -> theo(**F**))) — Premise

3. ZF* I -(theo(**L**) -> theo(**theo(L)**)) — Premise

4. I ZF* I -((theo(**F**) -> F)) — Assumption
I ------------

*5. I ! theo(**L**) — Assumption
I ! ------------

*6. I ! theo(**L**) -> (theo(**theo(L)**) -> theo(**F**)) — ProvE 2

*7. I ! (theo(**theo(L)**) -> theo(**F**)) — ImpE 6, 5

*8. I ! (theo(**L**) -> theo(**theo(L)**)) — ProvE 3

*9. | ! theo(**theo(L)**) ImpE 8, 5

*10. | ! theo(**F**) ImpE 7, 9

*11. | ! (theo(**F**) -> F) ProvE 4

*12. | ! F ImpE 11, 10

*13. | (theo(**L**) -> F) ImpI 5, 12

*14. | (L <-> (theo(**L**) -> F)) ProvE 1

*15. | L IffEL 14, 13

16. | ZF* | -(L) ProvI 15

*17. | theo(**L**) Rep 16

*18. | F ImpE 13, 17

19. | ZF* | -(F) ProvI 18

20. (ZF* | -((theo(**F**) -> F)) IMPLIES ZF* | -(F)) ImpI 4, 19

21. | ZF* | -(F) Assumption
| ------------

*22. | ! theo(**F**) Assumption
| ! ------------

*23. | ! F ProvE 21

*24. | (theo(**F**) -> F) ImpI 22, 23

25. | ZF* | -((theo(**F**) -> F)) ProvI 24

26. (ZF* | -(F) IMPLIES ZF* | -((theo(**F**) -> F))) ImpI 21, 25

27. (ZF* | -((theo(**F**) -> F)) IFF ZF* | -(F)) IffI 20, 26

q.e.d.

Now we present the proof of the second incompleteness theorem with the explicit use of Löb's Theorem.

Proof

1. ZF*CONS Premise

2. ZF* | -(~(H)) Premise

3. (ZF*CONS IFF NOT((ZF* | -(H) AND ZF* | -(~(H))))) Premise

4. ZF* | -(zf*cons <-> ~((theo(**H**) & theo(**~(H)**)))) Premise

5. (ZF* | -(H) IFF ZF* | -((theo(**H**) -> H))) Premise

6. | ZF* | -(zf*cons) Assumption
| ------------

7. | NOT((ZF* | -(H) AND ZF* | -(~(H)))) IffER 3, 1

```
*8.    | ! theo(H)                                          Assumption
       | !------------
*9.    | ! | ~(H)                                           Assumption
       | ! | ------------
*10.   | ! | (zf*cons <-> ~((theo(H) & theo(~(H)))))        ProvE 4
*11.   | ! | zf*cons                                        ProvE 6
*12.   | ! | ~((theo(H) & theo(~(H))))                      IffER 10, 11
*13.   | ! | theo(~(H))                                     Rep 2
*14.   | ! | (theo(H) & theo(~(H)))                         AndI 8, 13
*15.   | !H                                                 NotE 9,14, 12
*16.   | (theo(H) -> H)                                     ImpI 8, 15
 17.   | ZF* | -((theo(H) -> H))                            ProvI 16
 18.   | ZF* | -(H)                                         IffEL 5, 17
 19.   | (ZF* | -(H) AND ZF* | -(~(H)))                     AndI 18, 2
 20.   NOT(ZF* | -(zf*cons))                                NotI 6, 19, 7
```

B. The square root of 2 is not rational. - The logical search algorithm uncovers directly the following proof of the claim from the premises:

(1) $\sqrt{2}$ is rational $\leftrightarrow (\mathrm{E}\ x)\ (\mathrm{E}\ y)\ (\sqrt{2}*x = y\ \&\ \sim(\mathrm{E}z)\ (z \mid x\ \&\ z \mid y))$

(2) $(\mathrm{A}\ x)(\mathrm{A}\ y)\ (2*x^2 = y^2 \rightarrow 2 \mid x\ \&\ 2 \mid y)$

(3) $(\mathrm{A}\ x)(\mathrm{A}\ y)\ (\sqrt{2}*x = y \rightarrow 2*x^2 = y^2)$

The universe of discourse consists of the set of all reals or just the algebraic ones, but the range of the quantifiers consists just of the sort of positive integers. Here is the translation of the automatically generated proof; "translation," as the parser understands only a more restricted language.

1. $\sqrt{2}$ is rational $\leftrightarrow (\mathrm{E}\ x)(\mathrm{E}\ y)\ (\sqrt{2}*x = y\ \&\ \sim(\mathrm{E}z)\ (z \mid x\ \&\ z \mid y))$ — Premise
2. $(\mathrm{A}\ x)(\mathrm{A}\ y)\ (2*x^2 = y^2 \rightarrow 2 \mid x\ \&\ 2 \mid y)$ — Premise
3. $(\mathrm{A}\ x)(\mathrm{A}\ y)\ (\sqrt{2}*x = y \rightarrow 2*x^2 = y^2)$ — Premise
4. | $\sqrt{2}$ is rational — Assumption
   |------------
5. | $(\mathrm{E}\ x)(\mathrm{E}\ y)\ (\sqrt{2}*x = y\ \&\ \sim(\mathrm{E}z)\ (z \mid x\ \&\ z \mid y))$ — IffER 1, 4
6. | ! $(\mathrm{E}\ y)\ (\sqrt{2}*u = y\ \&\ \sim(\mathrm{E}z)\ (z \mid u\ \&\ z \mid y))$ — Assumption
   | !-----------
7. | ! | $(\sqrt{2}*u = v\ \&\ \sim(\mathrm{E}z)\ (z \mid u\ \&\ z \mid v))$ — Assumption
   | ! |-----------
8. | ! | $(\mathrm{A}\ y)\ (2*u^2 = y^2 \rightarrow 2 \mid u\ \&\ 2 \mid y)$ — AllE 2
9. | ! | $(2*u^2 = v^2 \rightarrow 2 \mid u\ \&\ 2 \mid v)$ — AllE 8
10. | ! | $(\mathrm{A}\ y)\ (\sqrt{2}*u = y \rightarrow 2*u^2 = y^2)$ — AllE 3
11. | ! | $(\sqrt{2}*u = v \rightarrow 2*u^2 = v^2)$ — AllE 10
12. | ! | $\sqrt{2}*u = v$ — AndEL 7
13. | ! | $2*u^2 = v^2$ — ImpE 11, 12
14. | ! | $2 \mid u\ \&\ 2 \mid v$ — ImpE 9, 13
15. | ! | $(\mathrm{E}\ z)(z \mid u\ \&\ z \mid v)$ — ExI 14
16. | ! | $\sim(\mathrm{E}\ z)(z \mid u\ \&\ z \mid v)$ — AndER 7
17. | ! | $\bot$ — $\bot$I 15, 16
18. | ! $\bot$ — ExE 6, 7, 17
19. | $\bot$ — ExE 5, 6, 18
20. $\sim(\sqrt{2}$ is rational$)$ — NotI 4, 19

$\bot$ is taken as a placeholder for an appropriate contradiction, say, (P & ~P).

C. In *Quaife 1988*, pp. 226-227, this "reasonably complex tautology" is presented:

[(P->(Q->R)) -> ((Q->(R->S)) -> (Q->(P->S)))]

Its proof, however, is considered to be too long for incorporation into the article. In our natural deduction framework the proof is absolutely canonical and direct; here it is – in twelve lines:

```
1.   | (P -> (Q -> R))                                    Assumption
     | ------------
2.   | ! (Q -> (R -> S))                                  Assumption
     | ! ------------
3.   | ! | Q                                              Assumption
     | ! | ------------
4.   | ! | ! P                                            Assumption
     | ! | ! ------------
5.   | ! | ! (R -> S)                                     ImpE 2, 3
6.   | ! | ! (Q -> R)                                     ImpE 1, 4
7.   | ! | ! R                                            ImpE 6, 3
8.   | ! | ! S                                            ImpE 5, 7
9.   | ! | (P -> S)                                       ImpI 4, 8
10.  | ! (Q -> (P -> S))                                  ImpI 3, 9
11.  | ((Q -> (R -> S)) -> (Q -> (P -> S)))               ImpI 2, 10
12.  ((P -> (Q -> R)) -> ((Q -> (R -> S)) -> (Q -> (P -> S))))  ImpI 1, 11
```

As mentioned in section 4, Quaife's framework is a formulation of the first-order metatheory of K4 within ITP. The predicate ThmK4(x) expresses that the formula x is a theorem of K4. Here are the clauses generating theorems (from p. 223):

(ITP.A1) If taut(x) then ThmK4(x);

(ITP.A2) ThmK4((b(x->y) -> (b(x)->b(y))));

(ITP.A3) ThmK4(b(x) -> b(b(x)));

(ITP.R1) If ThmK4((x->y)) & ThmK4(x) then ThmK4(y);

(ITP.R2) If ThmK4(x) then ThmK4(b(x)).

A1 guarantees that all tautologies are theorems; A2 and A3 correspond to the derivability conditions; R1 is modus ponens, and R2 expresses the semi-representability of the theorem predicate.

D. Here we present two further computer-generated proofs surrounding the incompleteness theorems. The first claim is a version of the first half of the first incompleteness theorem, asserting the unprovability of the reflection formula for the Gödel sentence.

(i) ZF*CONS IMPLIES NOT(ZF* |-(theo(**G**) -> G))

Proof

```
 1.    (ZF*CONS IFF NOT((ZF* |-(G) AND ZF* |-(~(G)))))        Premise
 2.    ZF* |-((G <-> ~(theo(G))))                             Premise
 3.    | ZF*CONS                                              Assumption
       | ------------
 4.    | ! ZF* |-((theo(G) -> G))                             Assumption
       | !------------
 5.    | ! NOT((ZF* |-(G) AND ZF* |-(~(G))))                  IffER 1, 3
 *6.   | ! (G <-> ~(theo(G)))                                 ProvE 2
 *7.   | ! | theo(G)                                          Assum
       | ! | ------------
 *8.   | ! | (theo(G) -> G)                                   ProvE 4
 *9.   | ! | G                                                ImpE 8, 7
 *10.  | ! | ~(theo(G))                                       IffER 6, 9
 *11.  | ! ~(theo(G))                                         NotI 7, 7, 10
 *12.  | ! G                                                  IffEL 6, 11
 13.   | ! ZF* |-(G)                                          ProvI 12
 *14.  | ! | G                                                Assumption
 *15.  | ! | theo(G)                                          Rep 13
 *16.  | ! | ~theo(G)                                         IffER 6, 14
 *17.  | ! ~(G)                                               NotI 14,15,16
 18.   | ! ZF* |-(~(G))                                       ProvI 17
 19.   | ! (ZF* |-(G) AND ZF* |-(~(G)))                       AndI 13, 18
 20.   | NOT(ZF* |-((theo(G) -> G)))                          NotI 4, 19, 5
 21. (ZF*CONS IMPLIES NOT(ZF* |-((theo(G) -> G))))            ImpI 3, 20
```

q.e.d.

The argument is perfectly canonical – up to the extraction step in line *12; at this point G could have been extracted from the formula (theo(**G**) -> G) in line 4. The resulting proof is "symmetric" to the given one.

The second claim asserts that for any refutable sentence R, the formula expressing its unprovability, i.e., ~(theo(**R**)), is in ZF* equivalent to its reflection formula (theo(**R**) -> R)).

(ii) ZF* I -(~(R)) IMPLIES ZF* I -((~(theo(**R**)) <-> (theo(**R**) -> R)))

Proof

```
  1.   ZF* I -(~(R))                                 Premise
 *2.   I ~(theo(R))                                  Assumption
       I ------------
 *3.   I ! theo(R)                                   Assumption
       I !------------
 *4.   I ! I ~(R)                                    Assumption
       I ! I ------------
 *5.   I ! R                                         NotE 2, 3
 *6.   I (theo(R) -> R)                              ImpI 5
 *7.   (~(theo(R)) -> (theo(R) -> R))                ImpI 6
 *8.   I (theo(R) -> R)                              Assumption
       I ------------
 *9.   I ! theo(R)                                   Assumption
       I !------------
*10.   I ! ~(R)                                      ProvE 1
*11.   I !R                                          ImpE 8, 9
*12.   I ~(theo(R))                                  NotI 10, 11
*13.   ((theo(R) -> R) -> ~(theo(R)))                ImpI 12
*14.   (~(theo(R)) <-> (theo(R) -> R))               IffI 7, 13
 15.   ZF* I -((~(theo(R)) <-> (theo(R) -> R)))      ProvI 14
```

q.e.d.

Automated Search for Gödel's Proofs

Wilfried Sieg and Clinton Field

July 16, 2003

Technical Report No. CMU-PHIL-144

Philosophy

Methodology

Logic

Carnegie Mellon

Pittsburgh, Pennsylvania 15213

Automated Search for Gödel's Proofs[*]

Wilfried Sieg
Clinton Field
Department of Philosophy
Carnegie Mellon University
Pittsburgh

[*] This paper is dedicated to Helmut Schwichtenberg who shares our fascination with automated proof search and who introduced the first author to the intricacies of Gödel's arguments (through lectures in Münster, a long time ago).

Abstract. We present *strategies* and *heuristics* underlying a search procedure that finds proofs for Gödel's incompleteness theorems at an *abstract axiomatic level.* As *axioms* we take for granted the representability and derivability conditions for the central syntactic notions as well as the diagonal lemma for constructing self-referential sentences. The *strategies* are logical ones and have been developed to search for natural deduction proofs in classical first-order logic. The *heuristics* are mostly of a very general mathematical character and are concerned with the goal-directed use of definitions and lemmata. When they are specific to the meta-mathematical context, these heuristics allow us, for example, to move between the object- and meta-theory. Instead of viewing this work as high-level proof search, it can be regarded as a first step in a proof-planning framework: the next refining steps would consist in verifying the axiomatically given conditions. Comparisons with the literature are detailed in section 4. (The general mathematical heuristics are indeed general: in Appendix B we show that they, together with two simple algebraic facts and the logical strategies, suffice to find a proof of "√2 is not rational.")[1]

1. Background. In a genuinely experimental spirit, we extended the *intercalation method for proof search* from pure first-order logic to parts of mathematics by interweaving general logical strategies with specific mathematical heuristics. The guiding question for our investigation was: What is needed, in addition to purely logical considerations, for finding proofs of significant theorems in a fully automated way? We answer the question for Gödel's incompleteness theorems. When proved at an *abstract axiomatic level* they lend themselves naturally to such an investigation; they have intricate, yet not overwhelmingly difficult proofs, and they are obviously significant. During the academic years 1975/77, the first author had taken steps towards establishing them interactively. That work was done for a computer-based course on *Elementary Proof Theory*; a detailed report was given in *Sieg 1978* and a brief summary in *Sieg e.a. 1981*.

Elementary Proof Theory presented the incompleteness theorems for ZF*, that is Zermelo-Fraenkel set theory without the axiom of infinity. Its major innovation consisted in carrying out the meta-mathematical work in a formal theory of binary trees and elementary inductive definitions, called TEM.[2] Without the detour of their arithmetization, the inductively given syntactic notions were shown to be representable in ZF*; the diagonal lemma was established and the proof of the Hilbert-Bernays derivability conditions, central for the second theorem, was sketched. Within that high-level framework the standard material on the incompleteness theorems is compact and the proofs are

[1] Our work was supported by all the members of the current AProS team, in particular by Joseph Ramsey, Orlin Vakarelov, and Ian Kash; we are very grateful.
[2] TEM abbreviates Theory for Elementary Meta-Mathematics. – Feferman systematically investigates in his papers [1982] and [1988] the use of "finitary inductive" definitions in meta-mathematics.

direct. It was natural to ask, whether the proofs can be found via an appropriate extension of the intercalation method.

The arguments for the incompleteness theorems are carried out in the first-order theory TEM: instead of viewing syntactic objects as (having been coded as) natural numbers, we consider them as finitely branching trees; instead of defining syntactic notions recursively, we specify them by elementary inductive definitions, briefly, by eid's. In the language of TEM we have the constant S for the empty tree and the function symbol [,] for the binary operation of building a tree from two given ones. We use X, Y, Z - possibly with indices - as variables ranging over binary trees. The axioms for S and [,] are formulated in analogy to those of Dedekind-Peano arithmetic for zero and successor. The further axioms of TEM include the induction principle for binary trees, and closure and minimality conditions for the eid's. Instead of discussing these axioms in generality – the details do not matter for the current project – we specify some definitions that are actually needed to characterize the formal theory for which the incompleteness theorems are to be proved.

The theory to be considered is ZF*, Zermelo and Fraenkel's theory of sets without the axiom of infinity. The details of its axiomatic formulation do not matter either for the current project. Let us assume that it is formulated in a first-order language with x, y, z – possibly with indices – as variables ranging over sets. To indicate the general character of eid's we specify the generating clauses of the familiar notion of a formula (taking for granted the concepts of atomic formula and of variable); @ stands for any binary sentential connective, Q for the existential or universal quantifier:

If X is an atomic formula, X is a FORMULA;
If X is a FORMULA, [~, X] is a FORMULA;
If X is a FORMULA and Y is a FORMULA, [@, [X, Y]] is a FORMULA;
If X is a variable and Y is a FORMULA, [[Q, X], Y] is a FORMULA.

We write also "FORM(X)" for "X is a FORMULA." TEM contains for such eid's a *closure* and a *minimality* principle. The first principle asserts that FORM is closed under the above clauses and is expressed by

FOR ALL X (if $\mathcal{A}$(FORM, X) then FORM(X)).[3]

The minimality principle claims that FORM is the smallest such class. This is the usual principle of induction for formulas:

If FOR ALL X (if $\mathcal{A}$ (P, X) then P(X))

then FOR ALL X (if FORM(X) then P(X)).

Formulas are binary trees built up from the empty tree using pairing. In a similar way one can generate inductively the relation X *is a proof of* Y *from assumptions* $Z_1, \ldots, Z_n$ or from a (n inductively generated) class of axioms; if X is a proof of Y using axioms of ZF*, this relation is denoted by PROOF(X,Y). To indicate that there is a ZF*-proof for Y, we write ZF* | -(Y), ZF* | -Y or THEO(Y).

Using the constant ∅ and the set-theoretic pairing operation < , > one can build up terms in the language of ZF* whose parse trees are isomorphic to the binary trees; they are used as names for the meta-mathematical trees in the same way as numerals in Dedekind-Peano arithmetic are used as names for natural numbers. With every meta-mathematical tree we can directly associate its set-theoretic name or *code*: CODE(S) = ∅ and CODE([X,Y]) = <CODE(X), CODE(Y)>. We also write | _X_ | for CODE(X) or indicate it by **X**. This is the apparatus needed to formulate the *representability conditions* for the syntactic notions. We give them paradigmatically for FORM and PROOF:

If FORM(X) then ZF* | - form(**X**), and

If NOT FORM(X) then ZF* | - ~form(**X**);

"form" is a formula in the language of set theory for which these conditions are provable in TEM. Similarly, there is a formula "proof" in the language of ZF* that represents the proof relation PROOF:

If PROOF(X,Y) then ZF* | - proof(**X**,**Y**), and

If NOT PROOF(X,Y) then ZF* | - ~proof(**X**,**Y**).

Using the first representability condition for PROOF one can establish:

If THEO(Y) then ZF* | - theo(**Y**),

[3] $\mathcal{A}$(P,X) is obtained from the generating clauses; it is the disjunction of the following TEM-formulas: (i) X is atomic; (ii) $(X)_0$ is ~ and $P((X)_1)$; (iii) $(X)_0$ is @ and $P(((X)_1)_0)$ and $P(((X)_1)_1)$; (iv) $((X)_0)_0$ is Q and $((X)_0)_1$ is a variable and $P((X)_1)$. P can be viewed as either a meta-variable over TEM-formulas or as a free second-order variable; under the second reading we have an appropriate substitution rule in the logical calculus for TEM.

where theo(y) abbreviates (Ex) proof(x,y).[4] Finally, we will use the *Self-reference Lemma* (or *Diagonal Lemma*) in the form: if F is a formula in the language of set theory (with one free variable), then there is a sentence D_F in that very language such that ZF* proves (D_F <-> F($\mathbf{D}_F$)). Applied to the formula ~theo(y), the self-reference lemma yields the Gödel sentence G that expresses its own unprovability, i.e., ZF* proves (G <-> ~theo(**G**)).

With this systematic background it is not difficult to prove that G is not provable in ZF* assuming, of course, that ZF* is consistent. So let us assume – in order to obtain a contradiction – that ZF* proves G; then, by the diagonal lemma concerning G, ZF* proves ~theo(**G**). On the other hand, by the (semi-) representability of THEO, we can infer from the fact that ZF* proves G, that ZF* establishes theo(**G**). Thus, ZF* proves both ~theo(**G**) and theo(**G**), and we have obtained a contradiction! The independence of G requires a proof that ~G is not provable either; for that a stronger assumption concerning ZF*, stronger than mere consistency, has to be made. Gödel used for that purpose the notion of ω-consistency; the corresponding concept for the context of our meta-mathematical set-up is τ-consistency, thinking of τ as the class of (sets denoted by codes for) binary trees. ZF* is *τ-consistent* is defined by the condition: there is no formula F(y) such that ZF* proves (Ey) (τ(y) & F(y)) and also ~F(**Y**) for all Y; or equivalently, for all formulas F(y), if ZF* proves ~F(**Y**) for all Y, then ZF* does not prove (Ey) (τ(y) & F(y)).

Assuming that ZF* is τ-consistent, we show now that ZF* does not prove the negation of the Gödel sentence G. By what we established already (and the fact that τ-consistency implies ordinary consistency) we know that

FOR ALL X: NOT PROOF(X,G);

the representability of PROOF implies

FOR ALL X: ZF* I - ~proof(**X**,**G**).

But then the τ-consistency of ZF* ensures

NOT ZF* I - (Ey) proof(y,**G**).

As the formula (Ey) proof(y,**G**) is abbreviated by theo(**G**), we can use the self-reference lemma for G to infer that this formula is in ZF* provably equivalent to

[4] The existential quantifier here is E, rather than the standard symbol ∃, to reflect the notation used for ZF* in APROS. In addition, the universal quantifier is A.

~G. Thus, NOT ZF* | -(~G), and the independence of G from ZF* has been established.

Given the axiomatic context provided by the representability of PROOF and THEO and the self-reference lemma applied to ~theo(y), the proofs are direct, yet intricate. To take a first step towards describing the search algorithm that finds proofs of these and related theorems, we present briefly the basic ideas underlying the intercalation method for classical logic; for the theoretical underpinnings we refer to *Sieg 1992*, *Sieg & Byrnes 1998* and *Byrnes 1999*. We should emphasize at this point that, in our view, logical formality per se does not facilitate the finding of proofs. However, logic within a natural deduction framework does help to bridge the gap between assumptions and conclusions by suggesting very rough structures for arguments, i.e., *logical structures* that depend solely on the syntactic form of assumptions and goals. This role of logic, though modest, is the crucial starting-point for moving up to subject-specific considerations that support a theorem. In the case study at hand we will show, how far these logical considerations go, and how they can be extended quite naturally by the *leading mathematical ideas* underlying Gödel's proofs.

2. Intercalation: broad strategies & special heuristics. The intercalation method is a proof search procedure that is goal-directed and guided by the possibly expanding syntactic context of the problem at hand. In first-order logic it is a complete procedure and a basis for broad logical strategies. The fundamental idea is straightforward. In order to bridge the gap between premises $A_1, \ldots, A_n$ and a goal B, one applies *systematically* the rules of the natural deduction calculus. I.e., the elimination rules are applied only from "above," whereas the introduction rules are inverted and applied from "below." Such systematic applications of the rules generate a search space that either contains a proof of B from the assumptions $A_1, \ldots, A_n$ or provides a semantic counterexample to the claim that B is a logical consequence of $A_1, \ldots, A_n$ - tertium non datur; in addition, proofs contained in the search space are necessarily normal. The argument for this sharpened completeness theorem provides a method for searching directly for normal proofs; indeed, it yields also a semantic argument for normal form theorems in natural deduction. Such arguments concerning classical first-order logic were first given in *Sieg 1992*, later also for intuitionistic logic and some modal logics in collaboration with Cittadini.

Normal proofs satisfy a similar *subformula property* as cut-free derivations in the sequent calculus. That, of course, allows a restriction of the systematic search and is basic for broad strategies underlying our proof search: (i) extracting B via elimination rules – if B is a strictly positive subformula of an assumption, (ii) sub-goaling via the appropriate inverted introduction rule – if B is a logically complex formula, (iii) refuting B via the rules for negation – if B is a negation or an atomic formula and if an appropriate pair of contradictory formulas is available. In the latter case there must be a negation that is a strictly positive subformula of an assumption. It is evident that direct proof search is strongly and naturally constrained by the syntactic context of the problem, as only particular subformulas can be intercalated between assumptions and goals.

With these logical strategies in the background let us return to the proof of the first part of the first incompleteness theorem and examine, how the intercalation method might find it with "a little help" (when pure logic is unable to proceed any further). So we begin with the goal NOT (ZF* |-(G)) and the premise ZF*CONS. We also have a definition and a lemma available, namely, the definition

ZF*CONS IFF NOT [ZF* |-(G) AND ZF* |-(~G)]

and the consequence of the diagonal lemma for ~theo(x), i.e.,

ZF* |-(G<->~theo(**G**)).[5]

The goal cannot be extracted from the premises. Thus, the algorithm proceeds indirectly with the assumption ZF* |-(G) and needs a pair of contradictory formulas as new goals. However, no negation occurs as a strictly positive subformula of the premise. As there is a negation in the definition of the premise, we use it and the premise to infer

NOT [ZF* |-(G) AND ZF* |-(~G)].

This negation is one element of a contradictory pair, and the algorithm attempts to prove [ZF* |-(G) AND ZF* |-(~G)]. This formula cannot be extracted: even

[5] We could have chosen one of the more general formulations of consistency, for example, NOT (EXISTS X) (ZF* |-(X) AND ZF* |-(~X)). The quantificational search in the SH-expansion (see Sieg and Byrnes) would find the appropriate instance quickly.

though it is a subformula of a premise, it is not a strictly positive one. So the algorithm inverts the formula and attempts to prove the new goals ZF* | -(G) and ZF* | -(~G). The former goal is already an assumption of the indirect proof, so we examine the latter goal.

It is here that we make the first significant change to the proof search procedure. ZF* | -(~G) cannot be extracted, but as an existential formula it can be inverted. Instead of searching for a term in the language of TEM describing a ZF*-proof of ~G, the search proceeds "inside" ZF*. The claim ZF* | -(~G) can be justified, after all, by the presentation of a proof of ~G within ZF*. The procedure tries now to find a ZF*-proof for the goal ~G. As the formula ~G cannot be extracted, indirect proof is applied to ~G: assume G and find a contradictory pair. There is no negation immediately available in the premises, except through the diagonal lemma for G. Note that this lemma is formulated within TEM as a provability claim for ZF* and should be available for any ZF*-proof. In general, when attempting an extraction or looking for contradictory pairs within a ZF*-proof, strictly positive subformulas of ZF*-formulas A must be considered, where ZF* | -(A) occurs as a strictly positive subformula of a premise or available assumption in TEM. So, the diagonal lemma makes available the formula ~theo(**G**), which is used to construct the contradictory pair. This leaves theo(**G**) as a new goal, which cannot be extracted. The regular proof search procedure would attempt an inversion. But here an additional step can be considered, since theo is a semi-representable relation: we can justify theo(**G**) by establishing ZF* | -(G) in TEM. ZF* | -(G) is an assumption in TEM, so the proof is complete.

The expanded version of the proof search algorithm, which results from the careful examination of the above proof, interweaves mathematical and purely logical considerations in an intercalating and goal-directed manner. It has the following main steps:

Extraction. If the goal is in TEM, then extraction functions as described above for first-order logic. If the goal is in ZF*, then the set of formulas available for extraction is expanded by those formulas A, for which the claim ZF* | -(A) is extractable in TEM and the goal is extractable from A. That is the inference ProvE, which is used to turn A into a part of the ZF*-proof.

Inversion. For the standard connectives inversion is applied as discussed earlier. There are two additional cases where "inversion" is applied. The first case occurs, when the goal in TEM is a statement of the form ZF* |-(A). Here the algorithm tries to find a proof of A in ZF*; that is the inversion of the inference ProvI.[6] In the second case, when the goal is a formula like [~] rel(**X**) in ZF*, and when the relation REL is represented by rel, the procedure tries to prove [NOT] REL(X) in TEM, after having explored indirect strategies in ZF*. For semi-representable relations such as ZF* |-(X), this step is obviously not applied to the negation ~rel(**X**) in ZF*.

Extended extraction and inversion ("Meaning of premises and goals"). Definitional and other mathematical equivalences are used to obtain either a new available formula from which the current goal is extractable or to get an equivalent statement as a new goal. This we would like to do relative to a developing background theory; currently, we just add the definitions and lemmata explicitly to the list of premises.

Indirect strategies are pursued in the same way as in pure first-order logic, with one exception: the set of contradictory pairs for indirect proofs in ZF* is expanded by pairs whose negations are strictly positive subformulas of A in case ZF* |-(A) (and this TEM-statement is itself extractable from an available TEM-claim.)

This completes the informal description of the algorithm that searches for statements surrounding the first incompleteness theorem. The extensions of extraction and inversion mentioned have a very general mathematical character, whereas the extensions via ProvE and ProvI express most directly meta-mathematical content. The former rule reflects, in part, that theorems can be appealed to in proofs, and the latter rule expresses that the search mechanism provides syntactically correct object theoretic proofs.

The extended search procedure evolved out of a probing analysis of the standard proofs for the first incompleteness theorem and incorporates what we take to be the *leading mathematical ideas* for this part of meta-mathematics. It finds proofs not only for the first and second incompleteness theorems (after

[6] If the goal is of the form ZF* |- ([~] rel(X)), the algorithm tries first to prove [NOT] REL(X) directly.

incorporating the derivability conditions), but also for a broader range of theorems and lemmata in this general area; cf. Appendix **A** for a proof of Löb's Theorem and Appendix **D** for two further examples. Even without the specifically meta-mathematical steps the algorithm is of real mathematical interest, as it discovers the structure of the proof for the irrationality of the square root of 2; see Appendix **B**.

3. Machine proofs & new heuristics. We present now the proofs of the first and second incompleteness theorem and start out by explaining the format of proofs. Proofs are presented in a modified Fitch-style format, which can be given using only plain text. We show the scope of assumptions by inserting bars between the number and formula on each line, with nested assumptions being noted by alternating bars and exclamation points. A line of dashes sets off the assumptions themselves. To distinguish the parts of the proof which occur in TEM and those which are embedded ZF*-proofs, we mark every line in the object language with a star. Note that ZF*-proofs retain the scope indications from the meta-language, and appeals to representability will use all available TEM-assumptions.

The rules include the standard natural deduction rules. For example, conjunction introduction has the name "AndI", and the left and right-hand versions of conjunction elimination are named "AndEL" and "AndER" respectively. To these basic rules we add special rule names for every heuristically applied theorem or lemma. "Rep" names the rule for representable or semi-representable relations, where the premise is a representable relation in TEM and the conclusion the corresponding relation in ZF*. "ProvE" and "ProvI" indicate provability elimination and introduction.

We present first the machine proof of non-provability of the Gödel sentence G, assuming that ZF* is consistent. In addition, the machine uses an instance of the diagonal lemma ZF* |-(G <-> ~(theo(**G**))) and the definition of consistency, ZF*CONS IFF NOT(ZF* |-(G) AND ZF* |-(~(G))).

Proof:[7]

[7] When following this argument and all the other machine proofs, the reader should keep in mind the intercalation strategies for bridging the gap between assumptions and goals. After all, they motivate the steps in the arguments.

1. ZF* |-(G <-> ~(theo(**G**))) Premise
2. ZF*CONS Premise
3. ZF*CONS IFF NOT(ZF* |-(G) AND ZF* |-(~(G))) Premise
4. | ZF* |-(G) Assumption
   | ------------
*5. |! G Assumption
   |! ------------
*6. |! theo(**G**) Rep 4
*7. |! (G <-> ~(theo(**G**))) ProvE 1
*8. |! ~(theo(**G**)) IffER 7, 5
*9. | ~(G) NotI 5, 6, 8
10. | ZF* |-(~(G)) ProvI 9
11. | ZF* |-(G) AND ZF* |-(~(G)) AndI 4, 10
12. | NOT(ZF* |-(G) AND ZF* |-(~(G))) IffER 3, 2
13. NOT(ZF* |-(G)) NotI 4, 11, 12

q.e.d.

To prove the independence of G we have also to establish the non-provability of ~G. As remarked earlier, that requires the stronger hypothesis of τ-consistency. Here are the premises for the non-provability of ~G: the diagonal lemma ZF* |-(G <-> ~(theo(**G**))), ZF*τCONS, ZF*τCONS IMPLIES [(FORALL X)(ZF* |-(~(proof(**X**,**G**))) IMPLIES NOT(ZF* |-(theo(**G**))], ZF*τCONS IMPLIES ZF*CONS, and a reformulation of what was established above, namely ZF*CONS IMPLIES (FORALL X)(NOT(PROOF(X,G))).

Proof:

1. ZF* |-(G <-> ~(theo(**G**))) Premise
2. ZF*τCONS Premise
3. ZF*τCONS IMPLIES
   [(FORALL X)(ZF* |-(~(proof(**X**,**G**)))
   IMPLIES NOT(ZF* |-(theo(**G**))] Premise
4. ZF*τCONS IMPLIES ZF*CONS Premise
5. ZF*CONS IMPLIES
   (FORALL X)(NOT(PROOF(X,G)))) Premise
6. | ZF* |-(~(G)) Assumption
   | ------------

*7. | ! ~(theo(**G**)) Assumption
| ! ------------
*8. | ! (G <-> ~(theo(**G**))) ProvE 1
*9. | ! G IffEL 8, 7
*10. | ! ~(G) ProvE 6
*11. | theo(G) NotE 7, 9, 10
12. | ZF* | -(theo(**G**)) ProvI 11
13. | (FORALL X)(ZF* | -(~(proof(**X**,**G**)))
IMPLIES NOT(ZF* | -(theo(**G**))) ImpE 3, 2
14. | ZF*CONS ImpE 4, 2
15. | (FORALL X)(NOT(PROOF(X,G))) ImpE 5, 14
16. | NOT(PROOF(X,G)) AllE 15
*17. | ~(proof(**X**,**G**)) Rep 16
18. | ZF* | -(~(proof(**X**,**G**)) ProvI 17
19. | (FORALL X)(ZF* | -(~(proof(**X**,**G**))) AllI 18
20. | NOT(ZF* | -(theo(**G**))) ImpE 13, 19
21. NOT(ZF* | -(~(G))) NotI 6, 12, 20

q.e.d.

For the proof of the *second incompleteness theorem*, i.e., the non-provability of the formal consistency statement zf*cons under the assumption of the consistency of ZF*, the formalism has to satisfy the *Hilbert-Bernays derivability conditions* D_1 and D_2. D_1 is the formalized semi-representability condition for the theorem predicate [theo(x) -> theo(**theo(X)**)], whereas D_2 is the provable closure under modus ponens [theo(**X -> Y**) -> (theo(**X**) -> theo(**Y**))]. The algorithm makes use of these conditions as rules with one additional heuristic to exploit D_2: if theo(**F**) is the goal and F, as a consequent of a conditional (or biconditional), is a strictly positive subformula of an available purely implicational formula, apply D_2 repeatedly and try to extract theo(**F**).

Proof:

1. ZF* | -(theo(**G**) <-> ~G)) Premise[8]
2. ZF* | -(zf*cons <-> ~(theo(**G**) & theo(**~G**))) Premise
3. NOT(ZF* | -(G)) Premise

[8] Notice that the diagonal lemma is used here in a propositionally equivalent form; the current algorithm does not find the proof, when it is given in its standard form.

4. | ZF* | -(zf*cons) Assumption

| ------------

*5. | ! ~(G) Assumption

| ! ------------

*6. | ! (theo(**G**) <-> ~G) ProvE 1

*7. | ! theo(**G**)) IffEL 6, 5

*8. | ! theo(**theo(G)**) Der_1 7

*9. | ! theo(**theo(G)**) -> theo(**~G**)) Der_2 6

*10. | ! theo(**~G**) ImpE 9, 8

*11. | ! theo(**G**) & theo(**~G**) AndI 7, 10

*12. | ! (zf*cons <-> ~(theo(**G**) & theo(**~G**))) ProvE 2

*13. | ! zf*cons ProvE 4

*14. | ! ~(theo(**G**) & theo(**~G**)) IffEL 12, 13

*15. | G NotE 5, 11, 14

16. | ZF* | -(G) ProvI 15

17. NOT(ZF* | -(zf*cons)) NotI 4, 17, 3

q.e.d.

This argument made use of the special character of the Gödel sentence G – in order to obtain the two conjuncts of line *11. Instead, one can exploit the elegant way of proceeding made possible by *Löb's theorem*:

For all sentences F: ZF* | -(theo(**F**) -> F) IFF ZF* | -(F).

Löb's theorem expresses that a sentence F is provable in ZF* if and only if its *reflection formula* (theo(**F**) -> F) can be established in ZF*. Consider a *refutable* sentence H (i.e. a sentence whose negation is provable in ZF*) and assume that ZF* is consistent; then H is not provable in ZF*. Löb's theorem implies that the corresponding reflection formula (theo(**H**) -> H) is not provable either. Thus, the *second incompleteness theorem* amounts to establishing NOT(ZF* | -(zf*cons)) from the premises NOT(ZF* | -(theo(**H**)->H)), ZF* | -(zf*cons<->~(theo(**H**) & theo(**~H**))), and ZF*|-(~H). That is done in the next proof.

Proof:

1. NOT(ZF* | -(theo(**H**) -> H)) Premise

2. ZF* | -(zf*cons <-> ~(theo(**H**) & theo(**~H**))) Premise

3. ZF* | -(~H) Premise

4. | ZF* | -(zf*cons) Assumption

```
      | ------------
 *5.  |! theo(H)                                    Assumption
      |! ------------
 *6.  |!| ~(H)                                      Assumption
      |!| ------------
 *7.  |!| theo(~H))                                 Rep 3
 *8.  |!| theo(H) & theo(~H)                        AndI 5, 7
 *9.  |!| (zf*cons <-> ~(theo(H) & theo(~H)))       ProvE 2
*10.  |!| zf*cons                                   ProvE 4
*11.  |!| ~(theo(H) & theo(~H))                     IffER 9, 10
*12.  |! H                                          NotE 6, 8, 11
*13.  | theo(H) -> H                                ImpI 5, 12
 14.  | ZF* |-(theo(H) -> H)                        ProvI 13
 15.  NOT(ZF* |-(zf*cons))                          NotI 4, 14, 1
```

q.e.d.

This proof of the second incompleteness theorem uses Löb's Theorem only in the discussion leading up to the precise derivational problem. In Appendix **A** the preliminary considerations are incorporated into the proof; there we also show an elegant machine proof of Löb's Theorem.

4. Comparisons. A number of researchers have pursued goals similar to ours, but with interestingly different programmatic perspectives and strikingly different computational approaches. We focus on work by Ammon, Quaife, Bundy e.a. [1996], and Shankar. We first discuss Ammon's and Quaife's work, as theirs is programmatically closest to ours: Ammon aims explicitly for a *fully automatic* proof of the first incompleteness theorem, and Quaife establishes the incompleteness theorems and Löb's theorem in a setting that is similarly "abstract" as ours.

In his 1993 Research Note *An automatic proof of Gödel's incompleteness theorem,* Ammon describes the SHUNYATA program and the proof it found for the first incompleteness theorem. SHUNYATA's proof is structurally identical with the proof in Kleene's book *Introduction to Metamathematics* (pp. 204-8); the latter proof is discussed in great detail in sections 4 and 5 of Ammon's note. Two main claims are made: (i) Gödel's undecidable sentence is "constructed" by the program "on the basis of elementary rules for the formation of formulas," and this is taken as evidence for the subsidiary claim (on p. 305) that the program "implicitly rediscovered Cantor's diagonal method;" (ii) the proof of its undecidability is found by a heuristically guided *complete proof procedure* involving Gentzen's natural deduction rules for full first-order logic. The first claim (made on p. 291 and reemphasized on p. 295) is misleading: the Gödel sentence is of course constructible by the elementary rules for the (suitably extended) language of number theory, but that the formula so constructed expresses its unprovability has to be ensured by other means (and is "axiomatically" required to do so by Ammon's definition 3 and lemma 1).[9] As to the second claim (made on p. 294), the paper contains neither a logical calculus nor a systematic proof procedure using the rules of the calculus. What one finds are local heuristics for analyzing quantified statements and conditionals together with directions to prove the negation of a statement, i.e., to use the not introduction rule. These latter directions are quite open-ended, as there is no mechanism for selecting appropriate contradictory pairs. (Cf. Ammon's discussion of the "contradiction heuristic" on p. 296.)

In 1988 Quaife had already published a paper on *Automated proofs of Löb's Theorem and Gödel's two incompleteness theorems*. The paper presents proofs of the theorems mentioned in its title[10] "at a suitable level of abstraction" - as the author emphasizes on p. 219 - "from the underlying details of Gödel numbering and of recursive functions." The suitable level of abstraction is provided by the provability logic K4. That well-known logic contains as special axioms the derivability conditions and as its special rule (beyond modus ponens) the rule of "necessitation;" the additional rule corresponds to the semi-representability of the theorem predicate. In order to make use of the resolution theorem proving system ITP, the first-order metatheory of K4 is represented in ITP by five "clauses," which are listed in Appendix C. Four of the clauses correspond to the axioms and rules just mentioned, whereas the very first clause guarantees that all tautologies are obtained. The tautologies are established by "applying properly

[9] Our assessment of this claim is in full agreement with that found in the *Letter to the Editor* by Brüning e.a..

[10] Quaife establishes only the unprovability of G, not of its negation under the assumption of ω-consistency. On p. 229 he asserts, "With the right axioms, its proof [i.e., the other half of the first incompleteness theorem, S&F] could be reproduced about as easily as the principal half above."

specified demodulators" and transforming given sentential formulas into conjunctive normal form; the underlying procedure is complex and involves particular weighting schemes. Quaife illustrates the procedure by presenting on pp. 226-7 a derivation of a "reasonably complex tautology;" the derivation uses a sequence of 73 demodulation steps. Quaife concludes the discussion of this derivation by saying: "ITP can also be asked to print out the line-by-line application of each demodulator, but that detailed proof is too long for this article." We present this tautology and its direct (and easily found) natural deduction proof in Appendix **C**.

In contrast to Ammon's paper, we find here a conceptually and technically straightforward meta-mathematical and logical set-up: representability and derivability conditions are axiomatically assumed, and the logical inference machinery is precisely and carefully described. However, it is very difficult to understand, how the syntactic context of axioms, theorems and assumptions directs the search in a way that is motivated by the leading ideas of the mathematical subject.[11] The proofs use in every case "axioms and previously proven theorems" in addition to the standard hypotheses for the theorem under consideration. It is clear that the "previously proven theorems" are strategically selected, and it is fair to ask, whether the full proof – from axioms through intermediate results to the meta-mathematical theorems – should be viewed as "automated" or rather as "interactive" with automated large logical steps. So the direct computational question is, would proofs of the main theorems be found, if only the axioms were available?

The answer is most likely "No." OTTER, the resolution theorem prover that developed out of ITP, was not able to prove, under appropriately similar conditions, the full first incompleteness theorem in 1996; that is reported in Bundy, Giunchiglia, Villafiorita and Walsh's paper *An incompleteness theorem via abstraction.*[12] It was precisely this computational problem that motivated their paper, namely to show how "abstraction" can be useful to attack it. They present a proof of Gödel's theorem, where the real focus is not on the particular meta-

[11] A similar reservation is articulated by Fearnley-Sander in his review of Quaife's book.

[12] On p. 10 they write: "This proof [of the full first incompleteness theorem; S&F] turns out to be a considerable challenge to an unguided theorem prover. We have given these axioms to OTTER (v. 3.0) ... but it blew up."

mathematical proof, but rather on the process of abstraction and refinement that aids proof planning. This process is not a fully automated one, since both the choice of the abstraction and the subsequent refinement of the abstract proof into the original language require external guidance. While we share the ultimate goal of limiting the search space for mathematical proofs by "abstraction," their semi-automated abstraction process is a very different, though complementary approach.

The three approaches we have been discussing are as "abstract" as ours in the sense that the diagonal lemma, the representability condition and, in Quaife's and our case, the derivability conditions are taken for granted. Shankar's book *Metamathematics, Machines, and Gödel's Proof* focuses on an interactive proof of (the Rosser version of) the first incompleteness theorem.[13] The explicit goal was to find out, whether the full proof could *in practice be checked* using a computer program, i.e., the Boyer-Moore theorem prover. In the preface to his book Shankar points out that "A secondary goal was to determine the effort involved in such a verification, and to identify the strengths and weaknesses of automated reasoning technology.". The crucial meta-mathematical task and most significant difficulty consisted in verifying the representability conditions - for a particular theory (the system Z_2 for number theory in Cohen's book) and a particular way of making computability precise (via McCarthy's Lisp). That required, of course, a suitable formalization of all meta-mathematical considerations within, what Shankar calls on p. 141, "a constructive axiomatization of pure Lisp." In sections 5.4 and 5.5 Shankar gives a very informative analysis of, and an excellent perspective on, the work presented.

Moving back from interactive theorem proving to automated proof search, it is clear that the success of our search procedure results from carefully interweaving mathematical and logical considerations, which lead from explicitly formulated principles to a given conclusion. Proofs provide *explanations* of what they prove by putting their conclusions in a context that shows them to be correct. This need not be a global context providing a foundation for all of mathematics, but it can be a rather more restricted one as

[13] In addition, Shankar provides a "mechanical proof" of the Church-Rosser Theorem in Chapter 6.

here for the presentation of the incompleteness theorems. Such a local deductive organization is *the* classical methodology of mathematics with two well-known aspects: the formulation of principles and the reasoning from such principles; we have illustrated only the latter aspect by using suitable strategic considerations and appropriate heuristic "leading mathematical ideas."

The task of considering a part of mathematics, finding appropriate basic notions, and explicitly formulating principles – so that the given part can be systematically developed – is of a quite different character. For Dedekind the need to introduce new and more appropriate notions arises from the fact that human intellectual powers are imperfect. The limitation of these powers leads us, Dedekind argues, to frame the object of a science in different forms or different systems. To introduce a notion, "as a motive for shaping the systems," means in a certain sense to formulate a hypothesis concerning the inner nature of a science, and it is only the further development that determines the real value of such a notion by its greater or smaller *efficacy* (Wirksamkeit) in recognizing general truths. In the part of meta-mathematics we have been considering, Hilbert and Bernays did just that: their formulation of representability and derivability conditions ultimately led to more "abstract" ones and, in particular, to the principles for the provability logic K4 and related systems; see (Boolos 1993).[14]

5. Concluding remarks. No matter how one might mechanize an attempt of gaining such a principled deeper understanding of a part of mathematics, the considerations for a systematic and efficient automated development would still be central. In our given meta-mathematical context, there is an absolutely natural step to be taken next. As we emphasized earlier, there is no conflict or even sharp contrast between proof search and proof planning: proof search is hierarchically and heuristically organized through the use of "axioms" and their subsequent verification (or refutation). The guiding idea for verification in the intercalation approach is to generate sequences of formulas, reduce differences,

[14] In a different, though closely related case, Hilbert and Bernays succeeded in providing "recursiveness conditions" for the informal concept of calculability in a deductive formalism; that was done in a supplement of the second volume of their *Grundlagen der Mathematik.*

and arrive ultimately at syntactic identities. Such difference reduction also underlies the techniques for inductive theorem proving that have been developed by Bundy e.a. in their recent book. We conjecture that those techniques can be seamlessly joined with the intercalation method to take the *next step* and prove the representability conditions. The strictly formal proof in TEM might then be transformed into a ZF* proof of the first derivability condition, automatically.

BIBLIOGRAPHY

Ammon, Kurt

1993 An automatic proof of Gödel's incompleteness theorem; Artificial Intelligence 61, 291-306.

Boolos, George

1993 *The logic of provability*; Cambridge University Press.

Brüning, S., M. Thielscher, and W. Bibel

1993 Letter to the editor; Artificial Intelligence 61, 353-4.

Bundy, Alan, Fausto Giunchiglia, Adolfo Villafiorita, and Toby Walsh

1996 An incompleteness theorem via abstraction; Technical Report #9302-15; Istituto per la ricerca scientifica e tecnologica, Trento.

Bundy, Alan, David Basin, Dieter Hutter, and Andrew Ireland

2003 *Rippling: Meta-level guidance for mathematical reasoning*; Book manuscript.

Byrnes, John

1999 Proof search and normal forms in natural deduction; Ph.D. Thesis; Department of Philosophy, Carnegie Mellon University.

Cohen, Paul J.

1966 *Set theory and the continuum hypothesis*; Benjamin, Reading, Mass.

Dedekind, Richard

1854 Über die Einführung neuer Funktionen in der Mathematik; Habilitationsrede; 428-38; in: *Gesammelte mathematische Werke* (Fricke, Noether and Ore, editors), vol. 3, Vieweg, 1933.

Fearnley-Sander, Desmond

Review of *Quaife 1992*; http://psyche.cs.monash.edu.au/

Feferman, Solomon

1982 Inductively presented systems and the formalization of meta-mathematics; in: *Logic Colloquium '80*, van Dalen, Lascar, Smiley (eds.), North-Holland Publishing Company, 95-128.

1988 Finitary inductively presented logics; in: *Logic Colloquium '88*, Ferro e.a. (eds.), North-Holland Publishing Company, 191-220.

Fitch, F.

1952 *Symbolic Logic*; The Ronald Press Company, New York.

Gödel, Kurt

1931 Über formal unentscheidbare Sätze der Principia mathematica und verwandter Systeme I; Monatshefte für Mathematik und Physik 38, 173-198.

Löb, M.

1955 Solution of a problem of Leon Henkin; J. Symbolic Logic, 20, 115-8.

Quaife, Art

1988 Automated proofs of Löb's theorem and Gödel's two incompleteness theorems; Journal of Automated Reasoning 4, 219-231.

1992 *Automated Development of Fundamental Mathematical Theories*; Kluwer Academic Publishers.

Shankar, N.

1994 *Metamathematics, Machines, and Gödel's Proof*; Cambridge Tracts in Theoretical Computer Science 38, Cambridge University Press.

Sieg, Wilfried

1978 *Elementary proof theory*, Technical Report 297, 104 pp., Institute for Mathematical Studies in the Social Sciences, Stanford.

1992 *Mechanisms and Search* (Aspects of Proof Theory); AILA Preprint.

Sieg, Wilfried and John Byrnes

1998 Normal natural deduction proofs (in classical logic); Studia Logica 60, 67-106.

Sieg, Wilfried and Saverio Cittadini

2002 Normal natural deduction proofs (in non-classical logics); Technical Report No. CMU-PHIL-130, 29 pp.

Sieg, Wilfried, Ingrid Lindstrom, and Sten Lindstrom

1981 Gödel's incompleteness theorems - a computer-based course in elementary proof theory; in: *University-Level Computer-Assisted Instruction at Stanford 1968-80*, P. Suppes (ed.), Stanford, 1981, 183-193.

APPENDICES

A. Löb's theorem. The context of the theorem is given in section 3. Here we present an argument obtained by our automated proof search and re-prove the second incompleteness theorem; in the latter proof, the appeal to Löb's theorem is explicitly built into the argument.

In order to prove Löb's theorem in TEM, one faces two claims, namely,
(i) ZF* | -(theo(**F**) -> F) IMPLIES ZF* | -(F)
and
(ii) ZF* | -(F) IMPLIES ZF* | -(theo(**F**) -> F).
The last claim is immediate, whereas the first is difficult: its proof uses the instance of the diagonal lemma for the formula (theo(x) -> F). Here is the precise derivational problem at the heart of Löb's theorem: ZF* | -(F) can be proved from the premises ZF* | -(theo(**F**) -> F) and ZF* | -(L <-> (theo(**L**) -> F)).

We actually have two proofs of Löb's theorem, which differ in the presentation of the derivability conditions. In the first proof the conditions are formulated as premises and are instantiated for this problem. They enter the search through the standard extraction procedure. In the second proof heuristics guide their application. The heuristics were described above and have a fairly general character; they are designed to apply each condition when it may be useful. The resulting proofs are very similar, differing mainly in the greater number of extraction rule applications necessary in the first proof to make use of the axiomatically given derivability conditions. We present only the first proof.

Proof

```
 1.  ZF* | -(L <-> (theo(L) -> F))                    Premise
 2.  ZF* | -(theo(L) -> (theo(theo(L)) -> theo(F)))   Premise
 3.  ZF* | -(theo(L) -> theo(theo(L)))                Premise
 4.  | ZF* | -((theo(F) -> F))                        Assumption
     | ------------
*5.  | ! theo(L)                                      Assumption
     | ! ------------
*6.  | ! theo(L) -> (theo(theo(L)) -> theo(F))        ProvE 2
*7.  | ! (theo(theo(L)) -> theo(F))                   ImpE 6, 5
*8.  | ! (theo(L) -> theo(theo(L)))                   ProvE 3
```

*9. | ! theo(**theo(L)**) ImpE 8, 5

*10. | ! theo(**F**) ImpE 7, 9

*11. | ! (theo(**F**) -> F) ProvE 4

*12. | ! F ImpE 11, 10

*13. | (theo(**L**) -> F) ImpI 5, 12

*14. | (L <-> (theo(**L**) -> F)) ProvE 1

*15. | L IffEL 14, 13

16. | ZF* | -(L) ProvI 15

*17. | theo(**L**) Rep 16

*18. | F ImpE 13, 17

19. | ZF* | -(F) ProvI 18

20. (ZF* | -((theo(**F**) -> F)) IMPLIES ZF* | -(F)) ImpI 4, 19

21. | ZF* | -(F) Assumption

| ------------

*22. | ! theo(**F**) Assumption

| ! ------------

*23. | ! F ProvE 21

*24. | (theo(**F**) -> F) ImpI 22, 23

25. | ZF* | -((theo(**F**) -> F)) ProvI 24

26. (ZF* | -(F) IMPLIES ZF* | -((theo(**F**) -> F))) ImpI 21, 25

27. (ZF* | -((theo(**F**) -> F)) IFF ZF* | -(F)) IffI 20, 26

q.e.d.

Now we present the proof of the second incompleteness theorem with the explicit use of Löb's Theorem.

Proof

1. ZF*CONS Premise

2. ZF* | -(~(H)) Premise

3. (ZF*CONS IFF NOT((ZF* | -(H) AND ZF* | -(~(H))))) Premise

4. ZF* | -(zf*cons <-> ~((theo(**H**) & theo(**~(H)**)))) Premise

5. (ZF* | -(H) IFF ZF* | -((theo(**H**) -> H))) Premise

6. | ZF* | -(zf*cons) Assumption

| ------------

7. | NOT((ZF* | -(H) AND ZF* | -(~(H)))) IffER 3, 1

*8. | ! theo(**H**) Assumption
| !------------

*9. | ! | ~(H) Assumption
| ! | ------------

*10. | ! | (zf*cons <-> ~((theo(**H**) & theo(**~(H)**)))) ProvE 4

*11. | ! | zf*cons ProvE 6

*12. | ! | ~((theo(**H**) & theo(**~(H)**))) IffER 10, 11

*13. | ! | theo(**~(H)**) Rep 2

*14. | ! | (theo(**H**) & theo(**~(H)**)) AndI 8, 13

*15. | !H NotE 9,14, 12

*16. | (theo(**H**) -> H) ImpI 8, 15

17. | ZF* | -((theo(**H**) -> H)) ProvI 16

18. | ZF* | -(H) IffEL 5, 17

19. | (ZF* | -(H) AND ZF* | -(~(H))) AndI 18, 2

20. NOT(ZF* | -(zf*cons)) NotI 6, 19, 7

B. The square root of 2 is not rational. - The logical search algorithm uncovers directly the following proof of the claim from the premises:

(1) $\sqrt{2}$ is rational <-> (E x) (E y) ($\sqrt{2}*x = y$ & ~(Ez) ($z \mid x$ & $z \mid y$))

(2) (A x)(A y) ($2*x^2 = y^2$ -> $2 \mid x$ & $2 \mid y$)

(3) (A x)(A y) ($\sqrt{2}*x = y$ -> $2*x^2 = y^2$)

The universe of discourse consists of the set of all reals or just the algebraic ones, but the range of the quantifiers consists just of the sort of positive integers. Here is the translation of the automatically generated proof; "translation," as the parser understands only a more restricted language.

1.	$\sqrt{2}$ is rational <-> (E x)(E y) ($\sqrt{2}*x = y$ & ~(Ez) ($z \mid x$ & $z \mid y$))	Premise
2.	(A x)(A y) ($2*x^2 = y^2$ -> $2 \mid x$ & $2 \mid y$)	Premise
3.	(A x)(A y) ($\sqrt{2}*x = y$ -> $2*x^2 = y^2$)	Premise
4.	$\mid$ $\sqrt{2}$ is rational $\mid$ ------------	Assumption
5.	$\mid$ (E x)(E y) ($\sqrt{2}*x = y$ & ~(Ez) ($z \mid x$ & $z \mid y$))	IffER 1, 4
6.	$\mid$! (E y) ($\sqrt{2}*u = y$ & ~(Ez) ($z \mid u$ & $z \mid y$)) $\mid$!------------	Assumption
7.	$\mid ! \mid$ ($\sqrt{2}*u = v$ & ~(Ez) ($z \mid u$ & $z \mid v$)) $\mid ! \mid$ ------------	Assumption
8.	$\mid ! \mid$ (A y) ($2*u^2 = y^2$ -> $2 \mid u$ & $2 \mid y$)	AllE 2
9.	$\mid ! \mid$ ($2*u^2 = v^2$ -> $2 \mid u$ & $2 \mid v$)	AllE 8
10.	$\mid ! \mid$ (A y) ($\sqrt{2}*u = y$ -> $2*u^2 = y^2$)	AllE 3
11.	$\mid ! \mid$ ($\sqrt{2}*u = v$ -> $2*u^2 = v^2$)	AllE 10
12.	$\mid ! \mid$ $\sqrt{2}*u = v$	AndEL 7
13.	$\mid ! \mid$ $2*u^2 = v^2$	ImpE 11, 12
14.	$\mid ! \mid$ $2 \mid u$ & $2 \mid v$	ImpE 9, 13
15.	$\mid ! \mid$ (E z)($z \mid u$ & $z \mid v$)	ExI 14
16.	$\mid ! \mid$ ~(E z)($z \mid u$ & $z \mid v$)	AndER 7
17.	$\mid ! \mid$ $\perp$	$\perp$I 15, 16
18.	$\mid !$ $\perp$	ExE 6, 7, 17
19.	$\mid$ $\perp$	ExE 5, 6, 18
20.	~($\sqrt{2}$ is rational)	NotI 4, 19

$\perp$ is taken as a placeholder for an appropriate contradiction, say, (P & ~P).

C. In *Quaife 1988*, pp. 226-227, this "reasonably complex tautology" is presented:

[(P->(Q->R)) -> ((Q->(R->S)) -> (Q->(P->S)))]

Its proof, however, is considered to be too long for incorporation into the article. In our natural deduction framework the proof is absolutely canonical and direct; here it is – in twelve lines:

```
1.   | (P -> (Q -> R))                                        Assumption
     | ------------
2.   | ! (Q -> (R -> S))                                      Assumption
     | ! ------------
3.   | ! | Q                                                  Assumption
     | ! | ------------
4.   | ! | ! P                                                Assumption
     | ! | ! ------------
5.   | ! | ! (R -> S)                                         ImpE 2, 3
6.   | ! | ! (Q -> R)                                         ImpE 1, 4
7.   | ! | ! R                                                ImpE 6, 3
8.   | ! | ! S                                                ImpE 5, 7
9.   | ! | (P -> S)                                           ImpI 4, 8
10.  | ! (Q -> (P -> S))                                      ImpI 3, 9
11.  | ((Q -> (R -> S)) -> (Q -> (P -> S)))                   ImpI 2, 10
12.  ((P -> (Q -> R)) -> ((Q -> (R -> S)) -> (Q -> (P -> S))))   ImpI 1, 11
```

As mentioned in section 4, Quaife's framework is a formulation of the first-order metatheory of K4 within ITP. The predicate ThmK4(x) expresses that the formula x is a theorem of K4. Here are the clauses generating theorems (from p. 223):

(ITP.A1) If taut(x) then ThmK4(x);

(ITP.A2) ThmK4((b(x->y) -> (b(x)->b(y))));

(ITP.A3) ThmK4(b(x) -> b(b(x)));

(ITP.R1) If ThmK4((x->y)) & ThmK4(x) then ThmK4(y);

(ITP.R2) If ThmK4(x) then ThmK4(b(x)).

A1 guarantees that all tautologies are theorems; A2 and A3 correspond to the derivability conditions; R1 is modus ponens, and R2 expresses the semi-representability of the theorem predicate.

D. Here we present two further computer-generated proofs surrounding the incompleteness theorems. The first claim is a version of the first half of the first incompleteness theorem, asserting the unprovability of the reflection formula for the Gödel sentence.

(i) ZF*CONS IMPLIES NOT(ZF* |-(theo(**G**) -> G))

Proof

```
 1.    (ZF*CONS IFF NOT((ZF* |-(G) AND ZF* |-(~(G)))))      Premise
 2.    ZF* |-((G <-> ~(theo(G))))                           Premise
 3.    | ZF*CONS                                            Assumption
       |------------
 4.    | ! ZF* |-((theo(G) -> G))                           Assumption
       | !------------
 5.    | ! NOT((ZF* |-(G) AND ZF* |-(~(G))))                IffER 1, 3
*6.    | ! (G <-> ~(theo(G)))                               ProvE 2
*7.    | ! | theo(G)                                        Assum
       | ! |------------
*8.    | ! | (theo(G) -> G)                                 ProvE 4
*9.    | ! | G                                              ImpE 8, 7
*10.   | ! | ~(theo(G))                                     IffER 6, 9
*11.   | ! ~(theo(G))                                       NotI 7, 7, 10
*12.   | ! G                                                IffEL 6, 11
 13.   | ! ZF* |-(G)                                        ProvI 12
*14.   | ! | G                                              Assumption
*15.   | ! | theo(G)                                        Rep 13
*16.   | ! | ~theo(G)                                       IffER 6, 14
*17.   | ! ~(G)                                             NotI 14,15,16
 18.   | ! ZF* |-(~(G))                                     ProvI 17
 19.   | ! (ZF* |-(G) AND ZF* |-(~(G)))                     AndI 13, 18
 20.   | NOT(ZF* |-((theo(G) -> G)))                        NotI 4, 19, 5
 21. (ZF*CONS IMPLIES NOT(ZF* |-((theo(G) -> G))))          ImpI 3, 20
```

q.e.d.

The argument is perfectly canonical – up to the extraction step in line *12; at this point G could have been extracted from the formula (theo(**G**) -> G) in line 4. The resulting proof is "symmetric" to the given one.

The second claim asserts that for any refutable sentence R, the formula expressing its unprovability, i.e., ~(theo(**R**)), is in ZF* equivalent to its reflection formula (theo(**R**) -> R)).

(ii) ZF* I -(~(R)) IMPLIES ZF* I -((~(theo(**R**)) <-> (theo(**R**) -> R)))

Proof

```
  1.   ZF* I -(~(R))                                Premise
 *2.   I ~(theo(R))                                 Assumption
       I ------------
 *3.   I ! theo(R)                                  Assumption
       I !------------
 *4.   I ! I ~(R)                                   Assumption
       I ! I ------------
 *5.   I ! R                                        NotE 2, 3
 *6.   I (theo(R) -> R)                             ImpI 5
 *7.   (~(theo(R)) -> (theo(R) -> R))               ImpI 6
 *8.   I (theo(R) -> R)                             Assumption
       I ------------
 *9.   I ! theo(R)                                  Assumption
       I !------------
*10.   I ! ~(R)                                     ProvE 1
*11.   I !R                                         ImpE 8, 9
*12.   I ~(theo(R))                                 NotI 10, 11
*13.   ((theo(R) -> R) -> ~(theo(R)))               ImpI 12
*14.   (~(theo(R)) <-> (theo(R) -> R))              IffI 7, 13
 15.   ZF* I -((~(theo(R)) <-> (theo(R) -> R)))     ProvI 14
```

q.e.d.