A Milestone Reached and a Secret Revealed*

Larry Wos
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439-4801
email: wos@mcs.anl.gov

Abstract

In this special issue of the *Journal of Automated Reasoning*, this article sets the stage for the succeeding articles, all of which focus on finding proofs of theorems of formal logic and on the various methodologies that were employed. The proofs that are offered mark an important milestone for automated reasoning and for logic, for each of them is indeed new. One of the key questions this article answers is why an automated reasoning program was able to find proofs that had eluded some of the finest mathematicians and logicians for many, many decades.

Keywords: automated reasoning, elegant proofs, logic, proof finding

1 Background

When the effort to automate logical reasoning seriously commenced in the early 1960s, few if any would have believed that so much would occur before the year 2000. (A far more complete story is told in the planned book *Automated Reasoning and the Finding of Missing and Elegant Proofs in Formal Logic.*) In the context of design and verification, among the astounding successes is the work of Moore and his colleagues in their monumental effort that culminated in the design of a manufactured and used chip (see [Moore1989] and the other articles on system verification in the corresponding special issue of the *Journal of Automated Reasoning*). In mathematics, William McCune's answering (with his program

*This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

EQP) of the Robbins algebra open question marked an impressive advance for automated reasoning and for mathematics [McCune1997].

This special issue of the *Journal of Automated Reasoning* marks an additional milestone, presenting numerous proofs that have been missing in formal logic since the late 1800s. These proofs are remarkable in many respects. First, some of them totally eluded such great minds as Frege, Hilbert and Ackermann, Tarski and Bernays, Church, Lukasiewicz, Meredith, and Rose and Rosser. Second, some of them offer markedly more satisfying properties than those that exist in the literature. Third, some of the proofs presented in this issue beautifully illustrate what can be accomplished by relying heavily on an automated reasoning program (such as William McCune's OTTER [McCune1989]), especially since no clue of any kind was available regarding a promising attack.

Of equal importance and in the context of automated reasoning, this special issue illustrates the intricate interplay of the various strategies and methodologies that have been formulated in recent years. Indeed, none of these powerful strategies and methodologies alone would have sufficed (from what it appears) to have enabled the completion of the proofs that are presented in this issue. Precisely how each strategy comes into play, the relative importance of each strategy, and the specific effects of each strategy at all points in a program's attack offer researchers in automated reasoning a deep and complex problem to solve. This article provides some clues in that regard and focuses on useful examples.

For logicians, this article presents a different—but equally challenging—problem. The proofs presented in this issue, in the main, share a startling property: All proof steps are free of double negation, contain no terms of the form $n(n(It))$ for any term t. In that regard, can the following conjecture be proved? Given a set of axioms A and a formula F deducible from A, a proof of F can always be found such that all of its deduced steps are free of double negation whenever F is free of double-negation terms.

2 A Secret Revealed

The reliance on various strategies is, of course, only part of the explanation for the recent successes of automated reasoning programs such as OTTER. What, then, is the secret ingredient to the successes reported in this issue? The answer lies in OTTER's ability to explore unappealing areas of the search space of conclusions, at least unappealing to unaided researchers. This ability enables the program to succeed where fine minds met an impasse.

OTTER, for example, experiences no reluctance at examining conclusions of level 70 and greater, where a formula or equation in the input has level 0 and has level n greater than 0 when one of its parents has level $n-1$. In contrast, the literature strongly suggests that researchers seldom if ever venture into that part of the search space, because of its lack

of appeal.

Many researchers—without the assistance of a program such as OTTER—also might avoid seeking proofs requiring a large number of applications of condensed detachment. One of the more charming results in that regard concerns the study of a single axiom of Lukasiewicz for two-valued sentential (or propositional) calculus for which *no* proof existed in the literature. The first proof of this deep theorem was found by OTTER, a proof of length 200 (applications of condensed detachment); its length alone suggests how difficult and how deep the theorem is. That proof was found with a methodology discussed in an article in this special issue. After numerous experiments, a 70-step proof was completed.

Another aspect of the secret to OTTER's success—where researchers had been thwarted—rests with moves that are unnatural. Such moves are designed to have the program go where no person has gone before (at least voluntarily), to explore parts of the search space of conclusions that are indeed forbidding. For one example, by instructing the program to treat certain very long formulas as extremely simple (short), OTTER will explore that part of the conclusion space found at level 65 and beyond. The first proof of the Lukasiewicz 23-letter single axiom for two-valued sentential (or propositional) calculus has level 68. For a second example, a program can be directed to explore the conclusions derivable from hypotheses that are on-the-surface ugly and lengthy expressions—expressions that instinct and perhaps experience would suggest be avoided.

An even more striking example of an unnatural move that led to OTTER's success in breaking new ground concerns the avoidance of double negation. Before this practice was introduced, many proofs were harder to complete and, in too many cases, never completed without substantial guidance from a researcher. A case in point focuses on Meredith's single axiom for two-valued sentential calculus.

```
%  Following is the Meredith single axiom.
P(i(i(i(i(i(x,y),i(n(z),n(u))),z),v),i(i(v,x),i(u,x)))).
```

For years, on and off, attempts to produce (from the Meredith axiom) an unguided derivation of a three-axiom system of Lukasiewicz met with failure.

```
%  Following are Lukasiewicz 1 2 3.
P(i(i(x,y),i(i(y,z),i(x,z)))).
P(i(i(n(x),x),x)).
P(i(x,i(n(x),y))).
```

Then, when a methodology was formulated that emphasizes the avoidance of double negation (as detailed in this special issue), the first fully automated proof was obtained.

The secret to the recent accomplishments of automated reasoning programs, and in

particular the secret to the many successes reported in this special issue, can now be cast in a more general light.

> When a question resists answer or a problem resists solution, (at least when a reasoning program is part of the team) to sharply increase the likelihood of success, replace the question or problem with one that is far tougher.

Intuition might naturally suggest to the contrary, but the evidence suggests such is the case.

3 Interplay and Intricacy

To me, the most fascinating aspect of automated reasoning concerns strategy. With a few examples, this article illustrates not only the intricate interplay of various strategies themselves, but also their interplay with other procedures offered by OTTER, showing how complex is the program's attack.

For the first example, consider the case in which the hot list strategy is used to permit the program to visit, revisit, and re-revisit some chosen set of formulas or equations (depending on the value assigned the input heat parameter). Also assume that the resonance strategy is used to instruct the program to prefer new items that pattern-match (where all variables are considered indistinguishable) any of the included resonators. In the case so far described, little or no interplay between the strategies occurs. However, if the (static) variation of the hot list strategy is replaced with the dynamic variation and the value assigned to the dynamic_heat_weight input parameter as well as the values assigned to the resonators is appropriately chosen, then much interplay can result.

If, say, the dynamic_heat_weight is assigned the value 1 and the resonators corresponding to key lemmas are each assigned the value 1, if and when the lemmas are proved, their clause equivalents will be adjoined to the hot list permitting the program to emphasize their role by visiting, revisiting, and re-revisiting them. Much interplay between the dynamic hot list strategy and the resonance strategy may occur, and the interplay may be crucial to reaching the objective. In contrast, in the given situation, unwise assignments of values may result in too much interplay and cause the size of the hot list to grow rapidly and produce the side effect of destroying the program's effectiveness.

For a second example, assume that some formula or equation is conjectured to be unwanted, whether the goal is to find a first proof or to find a shorter proof than that in hand. Assume further (as occurred in some of the research reported in this issue) that closely related items are also conjectured to provide interference with reaching the objective. A reasonable beginning would suggest that appropriate demodulators be included to block the use of unwanted items and, because of the nature of demodulation, also block (children

by subsumption) items subsumed by the unwanted items. However, so-called cousins of the undesirables might create havoc, items that pattern-match an undesirable (where all variables are treated as indistinguishable). To protect against such cousins, appropriate resonators with assigned values greater than the assigned max_weight could be included. Finally, assume that a list of hints is included to instruct the program to prefer any formula or equation that was identical to one of the hints. If an undesirable item is a member of such a list of hints and the options are chosen to also give preference to any (subsuming parent) item that subsumed one of the hints, the mere presence of the corresponding resonator with a very high assigned value will not suffice. Instead, the hint or hints in question must be removed. Thus, a quite different type of interplay is encountered, interplay among demodulation, resonance, and hints. The interplay has the effect of avoiding not only the immediate undesirable items, but also a type of child, of cousin, and of parent. Sometimes experimentation shows that one of the three suffices, and sometimes the triad is needed.

For a third example of interplay and intricacy, assume that the goal is to find a shorter proof than that which the literature offers. Briefly, ancestor subsumption compares two derivation paths to the same conclusion and (in effect) gives preference to the shorter. One might immediately conjecture that this procedure suffices and that no strategy is needed. Easily overlooked is the fact that the use of a shorter proof to an intermediate step may force the entire proof to be longer, an occurrence that has frequently been encountered in my research, for example, when studying proofs of the Meredith single axiom.

To experience a taste of this phenomenon, consider the following. Assume that the program or the researcher has found two proofs of a key lemma L, the first of length 10, and the second of length 7, and that the lemma and its proof are to be used in the proof of a theorem T. Next, assume that all of the 10 steps of the first proof of L can be put to frequent use in the proof of T, but 5 of the 7 steps of the second proof of L are useless for the rest of a proof of T. If the situation is worsened by the fact that inaccessibility to the steps that are present in the first proof of L but absent from the second proof of L forces the inclusion of, say, 9 steps that could have been avoided by relying on the first proof, a nice illustration is given of how a shorter subproof can get in the way of finding a shorter total proof.

The subtle intricacies of reasoning and the interplay that can occur become even more apparent with the following example. Let the task be that of finding a first proof that P (perhaps consisting of one or more formulas or equations) implies Q (perhaps consisting of one or more formulas or equations). Or, let the task be that of finding a shorter proof that P implies Q. Assume that a proof that Q implies R is in hand. As experimentation has shown, the goal can be reached by including resonators that correspond to the proof that Q implies R. Indeed, success can occur by acting as if the goal were to directly prove that P implies R, rather than merely juxtaposing two proofs (one for P implies Q followed by one for Q implies R).

At least odd, and perhaps quite counterintuitive, is the fact that a proof of R from Q should provide any assistance in deducing Q from P. To emphasize the point, clearly if a proof of Q from P is found or is in hand, a proof of R from P is readily producible (transitivity suffices), but the Q implies R segment should be irrelevant to the P implies Q segment. Nevertheless, as the earlier-cited planned book details, a far shorter double-negation-free proof that the Meredith single axiom implies the Lukasiewicz three-axiom system was found by relying, in part, on a proof that the Lukasiewicz system implies that of Church. (For a fine treatment of the need for strategy and related topics, see [Fitting1996], pages 104–107.)

4 The Value of Elegance

Some of the proofs presented in this special issue are more *elegant* than proofs ever before found. One might naturally question the value of access to elegant proofs. Three answers immediately come to mind. First, the completion of a proof whose elegance rests with its brevity can lead directly to the discovery of a dependency among axioms, a dependency that was not previously known. Second, such a proof might instead show that a lemma thought indispensable is in fact dispensable, not key to the proof of one or more deep theorems. Third, as illustrated in one of the articles in this issue, sometimes the elegance rests with the avoidance of certain types of term. Such a proof can refute a myth, showing that a type of argument is totally unneeded and relied upon merely because of tradition.

Indeed, a glance at or an intense study of the literature might readily and understandably lead to the conclusion that the equivalence of t and **not** of **not** of t is indispensable to the proof of the deep theorems in various areas of formal logic. The nature of mathematics and logic is a most fascinating topic, and part of that nature rests with what is necessary and what is sufficient.

Everything being equal, more elegant (when the focus is on length) proofs are easier to follow, yield their treasure more readily, and are more tractable in various respects, not the least of which is that concerning soundness. More generally, contrary to the views expressed in a *New York Times* article in 1991 [Kolata1991], much can often be learned from the proofs yielded by a program such as OTTER. Indeed, a study of the output files can lead to important advances, as occurred with the formulation of demodulation.

The preceding observations in part explain the interest in elegant proofs and the interest in proofs that pursue a direct path from point P to point R, rather than merely being satisfied with proof juxtaposition. They also help set the stage for many of the articles found in this special issue.

5 Experimentation and an Arsenal of Weapons

When attacking a deep theorem or a hard problem with the aid of a powerful reasoning program, the researcher is presented with tough decisions to make. Indeed, if one considers that the number of options offered by OTTER and the choices of values for those options, one gets a taste of the impracticality of an exhaustive set of experiments. For example, the value to assign to max_weight may be as small as 2 and as large as 48 (or larger), for heat as small as 0 and as large as 10, and for the pick_given_ratio as small as 1 and as large as 12. In other words, because some experiments require much CPU time, a full treatment of the possibly fruitful paths to a proof might require many thousands of CPU-hours. No shortcut to the automation of logical reasoning exists when the goal is that of proving a deep theorem.

Such is also the nature of mathematics and logic, as evidenced by the many decades that sometimes elapse before a question is finally answered. Indeed, some of the proofs found in the McCune and Padmanabhan monograph [McCune1996] would not have been found (so experimentation strongly suggests) without just the appropriate combination of values for the parameters offered by OTTER. For one example, an assignment of the value 11 to the pick_given_ratio provided what was needed, where both smaller and larger values produced essentially nothing of use. For a quite different example, in contrast to the reliance on the hot list strategy with an assignment of the value 8 to the heat parameter when studying a single axiom being generally a good move, total avoidance of the hot list strategy proved to be the right move, resulting in the discovery of a 51-step proof (rather than a 53-step) proof for the Meredith single axiom in which no formula contains more than 6 distinct variables.

In the articles found in this special issue, and reviewed briefly in the following section, the need for a large arsenal of weapons becomes apparent, and the complexity of mathematics and logic is exemplified.

What is common to the articles is reliance on experimentation. Indeed, experimentation at many levels is required. However, when a result is presented as the culmination of a sequence of experiments, only the highlights are typically presented, with the excursions omitted.

6 The Articles in Brief

The first of the articles that follow, written by Dolph Ulrich, provides an overview of the importance of the areas of logic on which the research focuses. It also discusses the significance of the research itself.

Next is an article by Robert Veroff showing how, in some cases, linked inference can be

used to prove that no shorter proof than that in hand exists. This use of linked inference is indeed unique and contrasts nicely with the original objective of the formulation of such inference rules.

The next article, written by Ken Harris and Branden Fitelson, presents remarkable results focusing on distributivity in many-valued sentential calculus. Indeed, before their research, no proofs of the featured theorems existed.

Next are two articles, respectively by Robert Veroff and Larry Wos, both concerned with the Meredith single axiom and its fully automated proof. Although the two approaches differ widely, each is successfully used to prove the cited theorem and, most important, each of the approaches is of general interest and useful in unrelated contexts.

The closing article for this special issue, by Branden Fitelson and Larry Wos, is concerned with the successful finding of various missing proofs. Some of the proofs are included, as well as the key elements of the various methodologies that were used.

References

[Fitting1996] Fitting, M., *First-Order Logic and Automated Theorem Proving,* Graduate Texts in Computer Science, Springer-Verlag, Berlin, 1996.

[Kolata1991] Kolata, G., "Computers still can't do beautiful mathematics", *New York Times*, Review Section E, 14 July 1991, p. 4.

[McCune1989] McCune, W., *OTTER 1.0 users' guide*, Tech. Report ANL-88/44, Argonne National Laboratory, Argonne, IL, January 1989.

[McCune1996] McCune, M., and Padmanabhan, R., *Automated Deduction in Equational Logic and Cubic Curves, Lectures Notes in Computer Science 1095,* Springer-Verlag, New York, 1996.

[McCune1997] McCune, W., "Solution of the Robbins problem", *J. Automated Reasoning* **19**, no. 3 (1997) 263–276.

[Moore1989] Moore, J Strother, "System Verification", *J. Automated Reasoning* **5**, no. 4 (1989) 409–410.