Searching for Circles of Pure Proofs*

Larry Wos

Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439

Abstract

When given a set of properties or conditions (say, three) that are claimed to be equivalent, the claim can be verified by supplying what we call a *circle of proofs*. In the case in point, one proves the second property or condition from the first, the third from the second, and the first from the third. If the proof that 1 implies 2 does not rely on 3, then we say that the proof is pure with respect to 3, or simply say the *proof is pure*. If one can renumber the three properties or conditions in such a way that one can find a circle of three pure proofs—technically, each proof pure with respect to the condition that is neither the hypothesis nor the conclusion—then we say that a *circle of pure proofs* has been found. Here we study the specific question of the existence of a circle of pure proofs for the thirteen shortest single axioms for equivalential calculus, subject to the requirement that condensed detachment be used as the rule of inference. For an indication of the difficulty of answering the question, we note that a single application of condensed detachment to the (shortest single) axiom known as $P4$ (also known as *UM*) with itself yields the (shortest single) axiom $P5$ (also known as *XGF*), and two applications of condensed detachment beginning with $P5$ as hypothesis yields $P4$. Therefore, except for $P5$, one cannot find a pure proof of any of the twelve shortest single axioms when using $P4$ as hypothesis or axiom, for the first application of condensed detachment must focus on two copies of $P4$, which results in the deduction of $P5$, forcing $P5$ to be present in all proofs that use $P4$ as the only axiom. Further, the close proximity in the proof sense of $P4$ when using as the only axiom $P5$ threatens to make impossible the discovery of a circle of pure proofs for the entire set of thirteen shortest single axioms. Perhaps more important than our study of pure proofs, and of a more general nature, we also present the methodology used to answer the cited specific question, a methodology that relies on various strategies and features offered by W. McCune's automated reasoning program OTTER. The strategies and features of OTTER we discuss here offer researchers the needed power to answer deep questions and solve difficult problems. We close this article (in the last two sections) with some challenges and some topics for research and (in the Appendix) with a sample input file and some proofs for study.

1. Motivation, Background, and the Specific Problem to Solve

To set the stage for the type of problem central to this article, we note that occasionally in mathematics and in logic researchers are interested in which axioms of a given field are independent and which dependent. For example, among the usual axioms for a group, those of right identity and right inverse are each dependent on the remaining set. Somewhat related, one sometimes wonders whether a chosen lemma (such as the inverse of the inverse of x equals x) is needed for a proof, or whether one can find a proof in which no terms of a specified form occur, such as $n(n(t))$ for any term t where the function n denotes negation. Given a set of formulas, equations, properties, conditions, or definitions each of which implies all of the others, the general problem in focus here asks whether one can find a sequence of proofs such that the sequence forms a *circle* and such that each proof is *pure* with respect to the remaining formulas.

* This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38.

Definition, circle of proofs. For a set of k equivalent elements—formulas, equations, properties, conditions, or definitions—a circle of proofs is a sequence of proofs such that the first proof shows that the first element implies the second, the second proof shows that the second element implies the third, ..., and the k-th proof shows that the k-th element implies the first.

Definition, pure proof with respect to a set of elements. For a set of elements—formulas, equations, properties, conditions, or definitions—a proof of element j from element i is pure with respect to the set of elements if and only if it does *not* rely on the use of any of the elements but the j-th and the i-th. The presence of a proper instance of an element other than the j-th or i-th does not render the proof impure. If such instances are in fact absent, we say the proof is *instance pure*. Further, if none of the deduced steps contains as a proper subterm an instance of an unwanted element, we say the proof is *subterm pure*.

To remove any ambiguity regarding what we mean by proof in general, we require that one or more specific inference rules be used and specified, for example, rules such as *paramodulation* [Wos87,Wos92] (which generalizes equality substitution) and *condensed detachment* [Kalman78,Kalman83] (which we discuss with examples in Section 1.2). The particular problem of interest here asks whether one can find a circle of thirteen pure proofs, relying solely on condensed detachment, given the thirteen (equivalent) shortest single axioms for *equivalential calculus* [Kalman78,Peterson76] (defined and briefly discussed in Section 1.2). In other words, regarding the circle property, can one order the thirteen shortest single axioms in such a way that one can find thirteen proofs, the first proof deducing (with condensed detachment) the second axiom from the first, ..., the thirteenth deducing the first axiom from the thirteenth? Regarding purity, the first proof must *not* rely on the use of the axioms numbered 3 through 13, ..., the thirteenth must *not* rely on the use of the axioms numbered 2 through 12.

>From the viewpoint of a graph, one is asked to arrange the nodes in such a manner that a path begins at one node, passes through a second node once, a third node once, ..., and completes with the original node. For a simple example that is closely related in the context of purity, imagine being asked to prove some theorem in group theory *without* proving and using the lemma that the inverse of the inverse of x equals x, or *without* using the axiom of, say, right identity.

To complete the stage setting for presenting our attack on the specific problem and for introducing the methodology that may prove of use in totally unrelated areas, we provide (in Section 1.1) a bit of history and (in Section 1.2) a brief treatment of equivalential calculus. In part to complement the material of this article, we offer in Sections 5 and 6 some challenges and some topics for research, and, to stimulate such research, we provide (in the Appendix) a sample input file and some proofs for study.

1.1. Genesis

Two factors led us to the study of the question of whether there exists a circle of pure proofs for the thirteen shortest single axioms for equivalential calculus. Perhaps most important, in the late 1970s, we were introduced to this area of logic by the logician J. Kalman, to whom we extend our thanks. That introduction (as various of our papers show) has led us to visit and revisit this area in various contexts, including seeking shorter proofs, testing new strategies, and developing methodology.

Also playing a role in our interest in the specific problem, in the mid-1960s (if we can trust our memory), we were told a small amount about Moufang loops. We were told that any one of three equations played the key role, and were told that the three are equivalent. We were told that the proof of the equivalence of the three identities was, in the following sense, flawed. Proofs existed that 1 implies 2, that 2 implies 3 and—rather than the preferred pure proof that 3 implies 1—the existing proof was two-stage, 3 implies 2, then 2 implies 1. In other words, in the obvious sense, four proofs were required, rather than three. Finally, we were told that every ordering of the three identities and corresponding proofs suffered a similar aesthetic flaw. Regarding the concern of this article, in effect we were told of a *circle* of proofs for the three Moufang identities—1 implies 2, 2 implies 3, and 3 implies 1—but the proofs are not *pure*. The lack of purity rests with the fact that the proof of 3 implies 1 relies on the use of the second identity, 2. With the given ordering of proofs—1, 2, 3, 1—purity requires the replacement of the proof of 3 implies 1, passing through the second identity, by a proof that 3 implies 1 *without* use of 2. As an alternative, to produce a pure circle of proofs requires a reordering of the three identities and corresponding proofs, each of which (three) does not rely on the intermediate use of the so-called third property.

1.2. Equivalential Calculus in Brief

Rather than a study of Moufang loops, the focus here, as noted, is on *equivalential calculus*. The elements to be studied are formulas that one can produce with the two-place function *e* (for equivalent) and the variables *x*, *y*, *z*, ... Among such formulas, we have

(1) e(x,x),
(2) e(e(x,y),e(y,x)),
(3) e(e(x,y),e(e(y,z),e(x,z))).

It is no coincidence that we selected these three formulas; they were chosen because they will remind one, respectively, of reflexivity, symmetry, and transitivity, properties naturally associated with ''equivalence''. In fact, these three formulas taken together provide an axiomatization for equivalential calculus. Not so obvious, and unlike equivalence in general, the formula (1) (that we identify with reflexivity) is provable from (2) and (3) [Wos90].

In place of (1), (2), and (3) as an axiomatization, the calculus is often studied in terms of one of thirteen formulas, each of which serves as an axiom system. Following are provably *all* of the shortest single axioms for equivalential calculus. Each is expressed as a *clause* [Wos87,Wos92] in notation acceptable to OTTER [McCune93], where the predicate *P* can be interpreted as ''provable'' and the function *e* as ''equivalent''; OTTER is offered on diskette in [Wos92]. (The numbering of the following is indeed strange, but we have repeated it as we were introduced to the notation; *P*6, if memory serves, was once thought to be a shortest single axiom.) When a line contains a ''%'', the characters from the first ''%'' to the end of the line are treated by the program as a comment.

```
P(e(e(x,y),e(e(z,y),e(x,z)))).  % P1_YQL
P(e(e(x,y),e(e(x,z),e(z,y)))).  % P2_YQF
P(e(e(x,y),e(e(z,x),e(y,z)))).  % P3_YQJ
P(e(e(e(x,y),z),e(y,e(z,x)))).  % P4_UM
P(e(x,e(e(y,e(x,z)),e(z,y)))).  % P5_XGF
P(e(e(x,e(y,z)),e(z,e(x,y)))).  % P7_WN
P(e(e(x,y),e(z,e(e(y,z),x)))).  % P8_YRM
P(e(e(x,y),e(z,e(e(z,y),x)))).  % P9_YRO
P(e(e(e(x,e(y,z)),z),e(y,x))).  % PYO
P(e(e(e(x,e(y,z)),y),e(z,x))).  % PYM
P(e(x,e(e(y,e(z,x)),e(z,y)))).  % XGK
P(e(x,e(e(y,z),e(e(x,z),y)))).  % XHK
P(e(x,e(e(y,z),e(e(z,x),y)))).  % XHN
```

The inference rule that is often used is *condensed detachment*. For this area of logic, condensed detachment considers two formulas, *e*(*A*,*B*) and *C*—respectively called the *major* and the *minor premiss*—and, if *C unifies* with *A*, yields the formula *D*, where *D* is obtained by applying to *B* the most general unifier of *C* and *A*. (For the curious, and consistent with earlier publications, the word ''premiss'' is spelled as Church recommends.) Unification [Wos87,Wos92] is a procedure that considers two expressions and seeks to find the most general substitution of terms for variables that makes the two identical. Unification is the basis of many of the procedures applied by OTTER and, more generally, is relied upon in automated reasoning.

Just for illustration, if one applies condensed detachment to

e(x,e(x,e(y,y)))

and

e(z,z)

with the second formula playing the role of (the minor premiss) *C*, one obtains

e(e(z,z),e(y,y)).

If one reverses the roles of the two formulas and applies condensed detachment, one obtains a copy of the first formula. To gain a fuller appreciation of the intricacy of using condensed detachment—ignoring the possible truth of the following expression (F)—one might by hand attempt to produce the conclusion obtainable from applying the inference rule to two copies of

(F) e(e(e(e(x,e(y,z)),e(y,x)),e(z,u)),u).

The conclusion one obtains with condensed detachment applied to two copies of the preceding formula is $e(x,x)$. To capture condensed detachment, we use hyperresolution [Wos87,Wos92] and the following clause, where lines beginning with ''%'' are treated as comments, where ''-'' denotes logical **not**, and where '' | '' denotes logical **or**.

```
% a clause for condensed detachment
-P(e(x,y)) | -P(x) | P(y).
```

>From the fact that any of the given thirteen shortest single axioms serves as a complete axiom system by itself, one sees that any can be used to deduce each of the other twelve. For example, to prove *P5* from *P*4 requires a single application of condensed detachment, applied to two copies of *P*4. Therefore, any proof of, say, *XHN* from *P*4 using condensed detachment must contain as its first deduced step *P*5. Regarding purity of proof, if one selects *P*4 as the hypothesis with the goal of proving one of the other twelve shortest single axioms, only one proof is possible, that of *P*5. Put another way, with *P*4 as the axiom and condensed detachment as the inference rule, all proofs of the other shortest single axioms (except that of *P*5) are *not* free of *P*5 and are, therefore, *not pure*.

1.3. Objectives

The following two questions are central to this article. First, from the viewpoint of mathematics and logic, does there exist an ordering of the thirteen shortest single axioms of equivalential calculus such that one can find a circle of pure proofs for that ordering? If such an ordering does exist, from the preceding remarks, *P*4 must immediately precede *P*5. Second, from the viewpoint of automated reasoning, what strategy or strategies can be used together with other mechanisms to enable a program such as OTTER to bear the brunt of the research?

Rather than giving a terse answer to either question, we shall detail our study with the additional objective of showing those unfamiliar with OTTER how it can be used as a research assistant. We suspect that even those with some experience with this program will find various techniques we present useful for quite unrelated research.

2. Attacking the Specific Problem

For those researchers who prefer a somewhat brisk treatment that focuses on the highlights rather than on the detailed development, we close (in Section 2.6) this section with a review. That review will also serve for those researchers who, after reading the details, wish access to a summary.

2.1. Beginning the Attack

With the knowledge that at least one constraint must be observed—*P*4 must immediately precede *P*5—the first move was to determine whether other constraints of the type existed. We took each of the thirteen formulas (shortest single axioms) *A* (given in Section 1.2) and, with the following command, instructed OTTER to (in effect) apply condensed detachment to *A* with itself and immediately cease drawing conclusions.

```
assign(max_given,1).
```

To verify the soundness of this aspect of the methodology, we began with *P*4, placing the negations (or denials) of all thirteen shortest single axioms in list(passive) [McCune93]. The clauses in list(passive) are used only for forward subsumption and for unit conflict, the latter being the most common test for establishing that a proof by contradiction has been completed. Clauses in that list are not subject to demodulation [Wos87,Wos92], and they are not used to draw conclusions (of course, other than the empty clause). If all is in order, *P*5 will be deduced, and it was, as confirmed by the corresponding proof by contradiction. When the cited methodology was applied to each of the other twelve shortest single axioms, no proofs were returned. In other words, no other constraints of the type under discussion exist.

With the knowledge that, other than the required *P*4-*P*5 immediate juxtaposition (in that order), any other immediate juxtaposition was possible, we set about to find out which (shortest single) axioms could be used by OTTER to deduce which axioms. We say ‘‘used by OTTER’’ because the theoretical deducibility of one axiom from another sheds no light on the practical deducibility by an automated reasoning program—or, for that matter, by a researcher unaided. Indeed, for two examples of the depth and difficulty offered by the study of the thirteen (shortest single axiom) formulas, we note that the questions of whether *XHK* and *XHN* are each strong enough to provide a complete axiomatization were open until the early 1980s. S. Winker settled each question in the affirmative [Wos83,Wos84].

Put another way, if one chooses either *XHK* or *XHN* and attempts to deduce one of the other twelve shortest single axioms—without excellent choices regarding strategy [Wos90]—one encounters a tough problem, for a person or for a program. The formula *XGK* presents similar problems. (Kalman [Kalman78] proved this formula to be a single axiom.) Being aware of the toughness presented by these three (axiom) formulas and being aware that *P*5 leads to a deduction of *P*4 in but two applications of condensed detachment—implying that, if one happens to exist, the escape route to completing a circle of pure proofs is indeed narrow—we expected that a variety of strategies would be needed to obtain deductions of various formulas from each of the twelve (including *P*5 and, of course, excluding *P*4).

>From earlier studies in the context of seeking shorter and less complex proofs [Wos90], we had a proof that completes with the deduction of *P*4 starting with *XHN*, and that study yielded the deduction of none of the other eleven shortest single axioms. Therefore, the obvious move was to place *XHN* immediately before *P*4 in our search for a circle of pure proofs. (We again note that the definition of pure says nothing about proper instances of the disallowed items.) In contrast to the ease of extending the partial circle in what might be called the direction left—*XHN*, *P*4, *P*5—adding on the right presented a problem, for our first attempt to deduce shortest single axioms from *P*5 yielded *P*4 and *XHN*, and no others. Further, the proof of *XHN* (that was obtained) is not even pure, for it relies on the deduction of *P*4. Obviously, the insertion to the right of *P*5 of either *P*4 or *XHN* was out, and, almost as obvious, purity of a proof of a fourth axiom to be placed to the right of *P*5 was presenting a problem. In particular, *P*4 and *XHN* might be an intermediate step *in all proofs* starting with *P*5 as the only axiom, thus preventing one from ever producing a circle of pure proofs.

Closer examination of the run that deduces both *P*4 and *XHN* from *P*5 shows how narrow is the escape route, if one plans to escape to a pure proof by avoiding the deduction of either, but especially of *P*4. In particular, the condensed detachment of *P*5 with itself yields a clause numbered (18), and the condensed detachment of (18) as major premiss (unified with the first literal of the clause for condensed detachment) with *P*5 as the minor premiss yields *P*4. Two possible escape routes are left, if the corresponding application of condensed detachment succeeds rather than being blocked because of unification failure: *P*5 can be used as the major premiss and (18) as the minor, and (18) can be used as both the major and minor premisses. A further check of the run shows that both applications of condensed detachment succeed. Again one sees how the use of an automated reasoning program facilitates research, even at the more mundane level of ascertaining the results of specific applications of condensed detachment.

We therefore made two changes in our search for shortest single axioms deducible from *P*5. First, to give more latitude to the program, we increased the max_weight (permissible complexity of retained formulas) from 28 to 40. Second, because of our interest in pure proofs, with the inclusion of the following weight template (whose contained formula is in fact *P*4) in weight_list(purge_gen), we blocked the retention of the clause corresponding to *P*4.

```
weight(P(e(e(e(x,y),z),e(y,e(z,x)))),200).
```

(A clause technically has two weights, its pick_given weight that is used in the context of choosing clauses as the focus of attention to drive the program’s reasoning, and its purge_gen weight that is used in the context of clause discarding; often the two weights are the same.) The inclusion of the given weight template is not without risk, for other formulas that match it, ignoring variables (which is the way the weighting strategy [McCharen76,Wos87,Wos92] works), will also be purged because of our chosen max_weight, namely, 40. (The weight of a clause is computed solely in terms of symbol count if no included weight template applies.) (In Section 2.5, we give a means that relies on *demodulation* for having one's cake and eating it too, for purging an unwanted formula but not purging its relatives.)

Regarding the unchanged options, with the intention of decreasing the required CPU time, we used the *hot list strategy* [Wos94] with the heat parameter assigned to 1 and with *P*5 and the clause for condensed detachment in the hot list. (Briefly, the hot list strategy enables the researcher to include in the hot list clauses that are conjectured to merit repeated and immediate visiting to complete the application of an inference rule rather than initiate it, when the program has decided to retain a new clause.) We used a level saturation approach [Wos92], which instructs OTTER to drive its reasoning by focusing on the clauses in the order they are retained, first come first serve (breadth first). Because of the fecundity that would almost certainly occur with max_weight assigned to 40, after 50 clauses were chosen as the focus of attention to drive OTTER's reasoning, we reduced the max_weight to 24. The following two commands were, therefore, included.

```
assign(change_limit_after, 50).
assign(new_max_weight, 24).
```

We included the following two commands (which instruct OTTER to search for ever shorter proofs of each deduced conclusion) with the notion (perhaps, but not for certain) that shorter proofs might be correlated to pure proofs.

```
set(ancestor_subsume).
set(back_sub).
```

OTTER first deduced *XHN*, with a proof of length 27 and level 12 (in contrast to length 15 and level 8 when the program was allowed to retain the clause equivalent of *P*4) and then deduced *XHN* again, with a proof of length 24 and level 13. (With one exception, the length of a proof in this article as given by OTTER, is the number of applications of condensed detachment. The exception occurs when the dynamic hot list strategy [Wos94] is in use, for two copies of a deduced clause may appear, one from the modified hot list, and one from the usable list.) (The level of input clauses is 0; the level of a deduced clause is one greater than the maximum of the levels of its parents.) Then, in approximately 350 CPU-seconds (on a SPARCstation-10) with retention of clause (10266), OTTER deduced *PYM*. The proof has length 32 and level 14. Far more important for our purposes, the proof of *PYM* from *P*5 is pure. We were thus able to insert *PYM* to the right, obtaining for our partial circle (that might not be completable) the sequence *XHN*, *P*4, *P*5, *PYM*.

2.2. Our Original Approach

While we were gathering the evidence just presented, at the same time we also ran various experiments, each designed to see which axioms were deducible by OTTER from each of the remaining axioms taken one at a time. To gather our data, we used a level saturation approach, assigned a max_weight (on the complexity of retained conclusions) of 28, reassigned it to 24 after 50 clauses had been chosen as the focus of attention (to drive or direct the program's reasoning), used ancestor subsumption [McCune93,Wos92] (instructing OTTER to seek ever and ever shorter proofs of each deduced conclusion) coupled with back subsumption, and used the hot list strategy to promote ease of proof completion. (When the hot list strategy is combined with level saturation, it enables the program to, so to speak, look ahead, drawing conclusions that have a level greater than those being drawn without its use.) For the hot list strategy, we assigned the heat parameter to 1 and placed in the hot list only the clause for condensed detachment and the clause corresponding to the shortest single axiom under consideration. We shall refer to this as our *original approach*.

Motivated by the structure of the partially completed circle of pure proofs, from the ensemble of experiments, we immediately reviewed the experiment using *PYM* as the only axiom. The only difference between this experiment and the experiment focusing on *P*5 that deduced *P*4 and *XHN* and no other axioms was the axiom in use. Indeed, consistent with our approach, *PYM* was the only member in the initial set of support, and it and the clause for condensed detachment were the only members in the hot list. OTTER completed proofs of *P*5, *XGK*, *XHK*, *XHK* (a second time), and *P*4. Except for the first of the two proofs of *XHK*, all proofs are pure. Immediately we see the value of relying on ancestor subsumption (which, from a practical standpoint, requires the use of back subsumption), for its use enabled the program to find a pure proof of *XHK*, the second proof of that formula.

The structure of the partially completed circle of pure proofs made the deduction of both *P*5 and *P*4 worthless for extending it to the left or to the right. The choice between inserting (at the right) *XGK* and *XHK* was easy to make. First, the experiment with *XGK* as the only axiom produced one proof, that of *PYO*. Second, earlier studies of *XHK* as the only axiom in the pursuit of short proofs had produced various proofs of *P*1, *P*2, *P*7, *P*8, and *P*9; that of *P*2 is not pure—discovered only after this article was essentially complete—but the other proofs are. Clearly, the most appealing path to follow was to insert (on the right) *XGK*, and then immediately insert (on its right) *PYO* to yield the sequence *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO* (as a longer, partially completed circle of pure proofs).

Examination of other experiments immediately enabled us to find a shortest single axiom to insert on the left of our partially completed circle. Specifically, the search for a deduction of *XHN* revealed its proof from *P*8. Since, in that run, *XHN* was the first axiom deduced, the proof was guaranteed to be pure, for (as noted) list(passive) was provided with the negation of each of the thirteen shortest single axioms. We, therefore, inserted (on the left) *P*8. Next, a glance at the run using *P*1 as an axiom showed how fecund it indeed is; its use led OTTER to completing twenty proofs (including duplicates), and all of the other twelve shortest single axioms were deduced. All twenty proofs are pure, the explanation of which we leave to future research. The sixteenth and the twentieth are of *P*8. We immediately inserted (on the left of *P*8) *P*1. (Actually, one might have expected us to postpone the decision of where to place *P*1, in view of the fact that its use leads to the deduction of each of the other twelve shortest single axioms; we simply did not make this choice for reason or reasons we cannot give.)

We then inserted *P*2 immediately to the left of *P*1, for the deduction of *P*1 from *P*2 is the first proof (of thirteen) completed when using *P*2 as the only axiom. The proof is, therefore, pure. With almost the same justification, we next inserted (on the left of *P*2) *P*3. The deduction of *P*2 from *P*3 was the fifth proof completed, and we (as so often) used some Unix commands to check that the proof is indeed pure. We were nearing success—so we thought—for we now had the sequence *P*3, *P*2, *P*1, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO* (as a longer, partially completed circle of pure proofs).

2.3. Modifying Our Original Approach

Because none of the experiments conducted so far produced a proof that completed with the deduction of *P*3, except from formulas such as *P*1 and *P*2 (which we did not wish to displace), we decided to modify our study of the formula *PYO*. Modification of the options was required, for an assignment of 28 and of 32 to max_weight resulted in OTTER's running out of conclusions to draw (with the message sos empty) before any proofs were completed. Raising the max_weight to 36 produced a proof of *XGK* and none other; that proof was of no use given the partial circle. Raising the max_weight to 40 produced the same result.

Therefore, we replaced a level saturation approach (for choosing the clauses on which to focus to drive the program's reasoning) with the ratio strategy (formulated by McCune) [Wos92], with the pick_given_ratio assigned to 4. With this change, OTTER was instructed to choose (for the focus of attention) four clauses by weight (symbol complexity, in this case), one by first come first serve (or breadth first), then four, then one, and the like. Our motive was to sharply perturb the space of possible conclusions to be deduced. We replaced weight_list(purge_gen) by weight_list(pick_and_purge) and included as its sole member the clause equivalent of the formula *XGK* with an assignment of a weight of 200, strictly greater than the chosen max_weight of 36. In other words, similar to the actions discussed earlier (in Section 2.1) in the context of the formula *P*5, we prevented the program from retaining (if deduced) the clause equivalent of *XGK*, risking the loss of like formulas that differ only in terms of which variables are present. We took this action in deference to our goal of finding a pure proof of a shortest single axiom not yet a member of our partially completed circle. To perturb the search space further, we dropped the use of ancestor subsumption and back subsumption and instructed OTTER to change the max_weight to 24 after 70 clauses were chosen as the focus of attention, rather than after 50 were chosen.

In contrast to the sparsity of completed proofs *before* we made the cited changes, the program completed ten proofs of which eight are pure, (in order) of *P*4, *P*7, *P*9, *PYM*, *P*3, *XHK*, *P*1, and *P*2.

The impure proofs (the fourth and eighth) are of *P*8 and *XHN*, each relying on the deduction of *P*4. The last of the ten proofs was completed in approximately 6700 CPU-seconds (on a SPARCstation-10), with retention of clause (111681).

A glance at the partially completed circle shows that three of the eight pure proofs are of interest, those of *P*7, *P*9, and *XHK*. In particular, we rejected the proof of *P*3 for, although a circle would indeed be completed, its acceptance would leave no room for the as-yet unused axioms. The choice from among the three proofs of interest was easy to make. First, our experiment with *P*9 as the only axiom had deduced *XHK* and no other shortest single axiom. Second, in our earlier study of *XHK*, we had found a pure proof of *P*7, in approximately 78,273 CPU-seconds (on the equivalent of a SPARCstation-2), with retention of clause (98393); the proof has length 39 and level 25. (A SPARCstation-2 is roughly half as fast as a SPARCstation-10.) Therefore, we chose *P*9, inserting it on the right of our partially completed circle, which immediately allowed us to then insert to its right *XHK* followed by *P*7. We thus were near our goal of a complete circle of pure proofs, for we had the sequence *P*3, *P*2, *P*1, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *XHK*, *P*7. Indeed, with a pure proof of *P*3 from *P*7—if one existed—we would have answered the question (central to this article) in the affirmative.

2.4. Pursuing an Alternate Goal

But the desired proof to complete the (first) circle provided substantial resistance, so much so that we decided to attempt to complete a second circle of pure proofs. Because we had thought we had a pure proof of *P*2 using *XHK* as the only axiom, obtained in approximately 51,359 CPU-seconds on the equivalent of a SPARCstation-2 with retention of clause (77799), perhaps we could find pure proofs to enable us to move both *P*3 and *P*7. After this article was essentially completed, we discovered that the cited proof is not pure, for it relies on *P*9 as an intermediate step; on August 12, 1994, we found the needed proof, of length 76 and level 37, in approximately 767 CPU-seconds on a SPARCstation-10, with retention of clause (22558). In other words, our plan was to seek appropriate pure proofs so that *P*3 is not, so to speak, to the left of *P*2 and so that *P*7 is not, so to speak, to the right of *XHK* (of course meaning before the circle is closed).

By revisiting our original set of experiments designed to see which axiom could be used by OTTER to deduce which axioms, a place for *P*3 was suggested by finding a pure proof (using it as the only axiom) of *XHK*. What was therefore needed was a pure proof of *P*3 using *P*9 as the only axiom. A search of the early experiment with *P*9 as the only axiom revealed no proof, pure or impure, of *P*3. Noting that the original experiment completed three proofs of *XHK* and none for any other shortest single axiom, similar to what we did earlier in our focus on *P*5, we blocked the program from retaining the clause equivalent of *XHK*—at least, that was our intention. However, closer inspection of the input file reveals that, by mistake, we included in weight_list(purge_gen) a weight template corresponding to the formula *XHN*.

The experiment was saved by the fact that *XHK* and *XHN* have the same shape, if the variables are ignored (treated as indistinguishable), which is how weight templates work in OTTER, fortunately. We increased the max_weight to 36 (to give OTTER more room in which to work) and made no other changes, still directing the search with level saturation. The program completed the desired pure proof of *P*3 in approximately 1862 CPU-seconds (on a SPARCstation-10), with retention of clause (36379).

As for where to insert *P*7, we remarked earlier that from *P*1 OTTER deduces all of the other twelve shortest single axioms. The proof of *P*7 is pure. All that remained was to produce a pure proof from *P*7 of *P*8, and we would have the desired (second) circle of pure proofs, with the first circle still in doubt. A glance at the original experiment using *P*7 as the only axiom revealed that, if we were to succeed, some modification was required. For reason or reasons we cannot recall, we made but one change: We added the use of the dynamic hot list strategy, assigning 8 to the dynamic_heat_weight. With this addition, we instructed OTTER to adjoin to the hot list *during* the run any clause whose weight is less than or equal to 8; in this run, weight was determined solely in terms of symbol count. (Credit for extending the hot list strategy by formulating the dynamic hot list strategy belongs to W. McCune.) This small, but obviously significant, change was just what was needed. The program produced the desired deduction of *P*8, and we had answered the question of interest in the affirmative.

- With the following ordering of the thirteen shortest single axioms, we found, with the invaluable assistance of OTTER, a circle of pure proofs. The (second) circle is *P*2, *P*1, *P*7, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *P*3, *XHK*, *P*2.

2.5. Returning to the Pursuit of the First Circle of Pure Proofs

Inspired by the cited success in hand, we conjectured that added effort might in fact enable us to complete the first circle, which (as noted) completes if one can find a pure proof from *P*7 of *P*3. Recognizing the possible difficulty (or, perhaps, impossibility) of the task, we began two almost simultaneous attacks, the first on the equivalent of a SPARCstation-2, and the second on a SPARCstation-10. In both attacks, because we wished to avoid *all* shortest single axioms but one—in contrast to blocking *one particular axiom*—we used demodulation (rather than weighting) in a significant way that we shall discuss shortly. (We thus fulfill our earlier promise made in Section 2.1 of giving a means that relies on demodulation for having one's cake and eating it too, for purging an unwanted formula but not purging its relatives.)

In the first attack, we assigned the max_weight to 48, reassigned it to 24 after 70 clauses were chosen as the focus of attention to drive the reasoning, assigned the pick_given_ratio to 4 (in place of level saturation), and dropped the use of ancestor subsumption and back subsumption. In the second and almost simultaneous attack, we assigned the max_weight to 32, reassigned it to 24 after 50 clauses were chosen as the focus of attention, used level saturation, and otherwise proceeded as in the first attempt. For a different comparison, we merely modified our original treatment of *P*7 as the only axiom by assigning the max_weight to 32 rather than to 28, dropped the use of ancestor subsumption and back subsumption, and used demodulation.

Regarding the use of demodulation, with the following technique (thanks to McCune), we instructed OTTER to block all proofs of shortest single axioms other than that of *P*3; of course, we did nothing to interfere with *P*7 itself. Put another way, of the thirteen shortest single axioms, we used demodulation to block the use of all but *P*7 and *P*3.

```
make_evaluable(_&_, $AND(_,_)).
list(demodulators).
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(z,y),e(x,z)))) = $T.  % P1_YQL
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(x,z),e(z,y)))) = $T.  % P2_YQF
% ($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(z,x),e(y,z)))) = $T.  % P3_YQJ
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,y),z),e(y,e(z,x)))) = $T.  % P4_UM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,e(x,z)),e(z,y)))) = $T.  % P5_XGF
% ($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,e(y,z)),e(z,e(x,y)))) = $T.  % P7_WN
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(z,e(e(y,z),x)))) = $T.  % P8_YRM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(z,e(e(z,y),x)))) = $T.  % P9_YRO
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,e(y,z)),z),e(y,x))) = $T.  % PYO
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,e(y,z)),y),e(z,x))) = $T.  % PYM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,e(z,x)),e(z,y)))) = $T.  % XGK
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,z),e(e(x,z),y)))) = $T.  % XHK
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,z),e(e(z,x),y)))) = $T.  % XHN
end_of_list.
```

Although the second attack began later than the first, it finished earlier, no doubt due in part to the almost double speed of a SPARCstation-10 over a SPARCstation-2.

The first attack proved successful, yielding a deduction of *P*3 in approximately 15,000 CPU-seconds, with retention of clause (176938); the proof is pure, which required no examination, for no other proof was completed because of our use of demodulation; it has length 28 and level 14. The second attack also proved successful, yielding a deduction of *P*3 in approximately 3320 CPU-seconds, with retention of clause (51777); the proof is pure, which required no examination, for no other proof was completed because of our use of demodulation; it has length 18 and level 9. In other words, the first circle does complete.

- With the following ordering of the thirteen shortest single axioms, we found, with the invaluable assistance of OTTER, a (first) circle of pure proofs. The (first) circle is *P*3, *P*2, *P*1, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *XHK*, *P*7, *P*3.

2.6. Review of the Attack on the Search for Pure Proofs

Here we provide a somewhat brisk review of our attack on the following question; the attack relies on a wide variety of strategies offered by McCune's automated reasoning program OTTER, strategies we touch on here for researchers who may wish to apply them elsewhere. Does there exist a *circle* of *pure* proofs for the thirteen shortest single axioms (given in Section 1.2) for equivalential calculus? For a circle of proofs to exist, one must find an ordering *A*1 through *A*13 of the axioms and a set of proofs such that the first proof deduces *A*2 from *A*1, the second deduces *A*3 from *A*2, ..., and the thirteenth deduces *A*1 from *A*13. For each of the thirteen proofs to be pure, each must rely on its hypothesis (shortest single) axiom and its conclusion (shortest single) axiom and on *none* of the other eleven (shortest single) axioms. In the context of equivalential calculus, we require the use of the inference rule *condensed detachment* (of Section 1.2) for deducing the steps of each proof.

To show that no circle of pure proofs exists (thus answering the question under attack in the negative), one must prove that (in effect) all possible paths to such an arrangement are blocked. For example, because a single application of condensed detachment to two copies of the formula *P*4 yields the formula *P*5, *P*4 must immediately precede *P*5. If a single application of condensed detachment to two copies of *P*5 yields *P*4, then no circle of pure proofs exists. On the other hand, to prove that a circle of pure proofs does exist (thus answering the question under attack in the affirmative), one must find an ordering of the thirteen shortest single axioms and a corresponding set of pure proofs. Of course, as observed, one of the proofs must deduce *P*5 from *P*4; any such proof will be pure, and all proofs of any of the other eleven shortest single axioms with *P*4 as hypothesis will *not* be pure.

The first phase of our attack had the objective of determining whether other *P*4-*P*5 constraints existed. For each of the thirteen shortest single axioms, we instructed OTTER to apply condensed detachment to two copies of it and cease. For this purpose, we used the following command.

```
assign(max_given,1).
```

For condensed detachment (in our entire study), we used hyperresolution and the following clause, where ''-'' denotes logical **not**, '' | '' denotes logical **or**, and the predicate *P* can be interpreted as ''provable'' and the function *e* as ''equivalent''.

```
-P(e(x,y)) | -P(x) | P(y).
```

We learned that no additional constraints of the type under discussion exist.

In the next phase, we instructed OTTER to take each of the thirteen axioms and deduce as many of the remaining axioms as reasonably possible with what we call our original approach (Section 2.2). Featured in the original approach was the use of level saturation (first come first serve or breadth first) to direct the search for conclusions. To give the program enough room to operate in regard to the weight or complexity of retained conclusions (measured solely in terms of symbol count) and yet avoid drowning the program, we included the following commands.

```
assign(max_weight,28).
assign(change_limit_after, 50).
assign(new_max_weight, 24).
```

With the perhaps naive notion that the fewer deduced steps in a proof, the greater the likelihood that the proof would be pure, we included the following two commands, the first designed specifically to cause the program to seek shorter proofs of each deduced conclusion.

```
set(ancestor_subsume).
set(back_sub).
```

The second of the two commands just given is, from a practical standpoint, required when including the first. (The default in OTTER is ''back subsumption is set''.) To instruct OTTER to seek as many proofs as the assigned CPU time and memory permit, we included the following command.

```
assign(max_proofs, -1).
```

Finally, because of earlier experiments in other contexts, we included one additional command (which will merit some explanation) to increase the reasoning power of the program, the following.

```
assign(heat,1).
```

(The default value for this parameter is 1; the hot list strategy is automatically invoked by including one or more members in the hot list.)

The preceding command, or a similar command with a value greater than 1, is included when the researcher instructs the program to use the *hot list strategy*. This strategy enables one to include in a hot list clauses that are conjectured to merit repeated immediate visiting and, when the value is greater than 1, even immediate revisiting. Such clauses are used to *complete* the application of an inference rule, and not to *initiate* the application. Regarding the phrase ''immediate visiting'', when OTTER decides to retain a conclusion (in the form of a clause), *before* the conclusion is placed in the database in list(sos), the new clause is considered with the appropriate number of members of the hot list in the attempt to draw additional conclusions. Use of the hot list strategy almost invariably dramatically rearranges the order in which clauses are deduced. If one wishes to add to the hot list *during* the run, then one includes a command of the following type to instruct the program to also rely on the *dynamic hot list strategy* (due to McCune).

```
assign(dynamic_heat_weight, 8).
```

With the command as given, OTTER will adjoin to the hot list each clause it decides to retain if the clause has weight less than or equal to 8, technically, pick_given weight, which is its weight in the context of being chosen as the focus of attention; its purge_gen weight is its weight in the context of being discarded. Very often the two weights of a clause are identical. Regarding the choice of members for the (initial) hot list, typically (in this study) we included the shortest single axiom under consideration and the clause for condensed detachment, in the following manner.

```
list(hot).
% Following clause is for condensed detachment.
-P(e(x,y)) | -P(x) | P(y).
% Following is hypothesis, P5
P(e(x,e(e(y,e(x,z)),e(z,y)))).  % P5_XGF
end_of_list.
```

When a line contains a ''%'', the characters from the first ''%'' to the end of the line are treated by the program as a comment.

The use of our original approach in the second phase of our attack yielded a substantial amount of information, not all of which was welcomed. Most welcome was the discovery that various formulas, such as *P*1, *P*2, and *P*3, are indeed fertile, yielding many proofs. Even better, most of the proofs obtained in the second phase are pure. Not so welcome, formulas such as *PYO* appear almost sterile; from *PYO*, the original approach yielded a deduction of *XGK* and no other. Also not welcome, use of the formula *P*5 yielded a deduction of *P*4 in but two applications of condensed detachment, correctly implying that the escape route (with it as hypothesis) to the deduction of another element to add to a possible circle of pure proofs is narrow.

Intending to use various proofs found in the second phase, supplemented by some proofs found in earlier studies in the context of seeking shorter proofs, we commenced the third phase of our attack. This phase had the objective of following the narrow (possible) escape route that would lead from *P*5 to another shortest single axiom as yet not used, via a pure proof. Our approach was to use the following weight template (whose contained formula is in fact *P*4) to prevent OTTER from retaining *P*4.

```
weight(P(e(e(e(x,y),z),e(y,e(z,x)))),200).
```

Because of being assigned such a high weight (200), *P*4 and clauses that are similar to it (where variables are treated as indistinguishable) will be purged for having a weight in excess of our chosen max_weight. To give the program added latitude, we increased the max_weight from 28 (which it was in our original approach) to 40. The modified approach succeeded: OTTER found a pure proof of the formula *PYM*, enabling us to place *PYM*, so to speak, to the right of *P*5. We completed this phase by relying on proofs found in phase 2, resulting in the following partially complete circle: *P*3, *P*2, *P*1, *P*8,

XHN, *P*4, *P*5, *PYM*, *XGK*, *PYO*.

The apparent lack of fecundity possessed by *PYO* took us to the fourth phase in which we replaced level saturation by the use of the ratio strategy (due to McCune), with the pick_given_ratio assigned to 4. The cited strategy blends (subject to the assignment) weighting and level saturation. With the assignment of 4, OTTER is instructed to choose for the focus of attention to drive its reasoning four clauses by weight, one by first come first serve (or breadth first), then four, then one, and the like. Our motive was to sharply perturb the search space. Therefore, we also dropped the use of ancestor subsumption and back subsumption. In addition, with an appropriate weight template, we prevented the program from retaining the clause equivalent of *XGK*. Our actions produced a sharp contrast to the apparent just-cited sparsity of proofs, for this phase yielded ten proofs of which eight are pure. >From these proofs, we chose a pure proof of *P*9. Borrowing from earlier experiments, we added to our partially completed circle, obtaining *P*3, *P*2, *P*1, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *XHK*, *P*7.

With a pure proof of *P*3 from *P*7, if one existed, we would have answered the specific question of interest in this article in the affirmative, by finding a circle of pure proofs for the thirteen shortest single axioms. Because our efforts were temporarily thwarted, (as we report in Section 2.4) we applied the methodology in an attempt to complete a second circle of pure proofs. Rather than a full account, here we note that success was the result. OTTER found the needed pure proofs for the following circle: *P*2, *P*1, *P*7, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *P*3, *XHK*, *P*2. We note that we were required to correct (on August 12, 1994) one flaw, the lack of purity in the proof of *P*2 with *XHK* as the only axiom; we found the needed proof, of reported length 80 (because of using the dynamic hot list strategy) and actual length 76 and level 37, in approximately 767 CPU-seconds on a SPARCstation-10, with retention of clause (22558).

Then, directly because of completing a (second) circle of pure proofs, (as reported in Section 2.5) we renewed our effort in the context of the first possible circle. With the objective of *deducing* precisely one shortest single axiom, *P*3, from *P*7—rather than *blocking* the retention of one particular shortest single axiom—we used demodulation (Section 2.5) in place of weighting. We initiated almost simultaneously two attacks (with demodulation). In the first attack, we assigned the max_weight to 48 and the pick_given_ratio to 4, and we dropped the use of both ancestor subsumption and back subsumption. In the second, we assigned the max_weight to 32 and used level saturation. Both attacks succeeded, and we had the desired circle of pure proofs, the following: *P*3, *P*2, *P*1, *P*8, *XHN*, *P*4, *P*5, *PYM*, *XGK*, *PYO*, *P*9, *XHK*, *P*7, *P*3.

3. Arbitrary Circles of Pure Proofs

When we conveyed our results to our colleague McCune, he made a suggestion that generalized one we had in mind. We had thought that, since we had developed an apparently powerful methodology (almost totally dependent on OTTER) for finding circles of pure proofs, perhaps we could produce a circle of pure proofs with the ordering of the shortest single axioms first given to us (see Section 1.2). More generally, McCune suggested that we choose an arbitrary ordering to test the methodology. Of course, he was ruling out placing *P*5, so to speak, to the left of *P*4, for (as noted) *P*5 must be immediately to the right of *P*4; *P*5 is yielded from *P*4 with the first application of condensed detachment that must be made. Despite the merit of McCune's suggestion, we were content to test the methodology on the order in which we were introduced to the thirteen shortest single axioms, the following: *P*1, *P*2, *P*3, *P*4, *P*5, *P*7, *P*8, *P*9, *PYM*, *PYO*, *XGK*, *XHK*, *XHN*, *P*1.

Immediately, we had a master plan: where possible, borrow from the earlier studies of the two completed circles of pure proofs, and otherwise run new experiments utilizing the methodology that produced the two circles. We searched for the proofs in the order (just cited) dictated by the intended circle of shortest single axioms. In the following, we give proof length and level and other such data to enable the researcher to extend our results in various ways and to provide information for evaluating new programs and new ideas.

A glance at the original approach applied to *P*1 yielded a pure proof of *P*2; the proof has length 9 and level 6 and required approximately 36 CPU-seconds to obtain, completing with the retention of clause (2987); the computer was a SPARCstation-2 (used on July 18, 1994). The original approach also yielded the desired proof of *P*3, using as the only axiom *P*2; the proof has length 10 and level 7 and required approximately 40 CPU-seconds to obtain, completing with retention of clause (3301); the computer was a SPARCstation-2 (used on July 18, 1994). To obtain a pure proof of *P*4 from *P*3, after various failures, we succeeded by changing the max_weight from 28 to 32, dropping the use of ancestor subsumption and back subsumption, and using demodulation to prevent OTTER from retaining any of the twelve possible target axioms other than *P*4. The proof has length 13 and level 7 and required approximately 798 CPU-seconds to obtain, completing with retention of clause (25793); the computer was a SPARCstation-10 (used on July 20, 1994). The proof of *P*5 from *P*4 was obtained even before we applied our original approach, when we were simply having the program apply a single condensed detachment step (to each of the thirteen axioms in turn) to see, in the context of ordering the axioms, which moves were forced. The proof has length 1 and level 1 and required .3 CPU-seconds to obtain, completing with retention of clause (16); the computer was a SPARCstation-2 (used on July 18, 194).

Our search of various experiments for a proof of *P*7 using *P*5 as the only axiom yielded none. We also failed to find the desired proof when using the just-described approach that succeeded with *P*3 as the only axiom. Therefore, using level saturation, we assigned the max_weight to 36 rather than to 32, reassigned the max_weight to 20 rather than to 24 (because a max_weight of 36 produces many more conclusions), and, more important, instituted the use of the dynamic hot list strategy with an assignment of 8 to the dynamic_heat_weight. We also assigned the heat parameter to 2 rather than to 1. On a SPARCstation-10 (on July 20, 1994), OTTER obtained the desired proof in approximately 1275 CPU-seconds, completing with retention of clause (32512); the *reported* proof length is 46, and the level is 23. The actual proof length is 44; the extra two copies of deduced clauses are accounted for by being present in both the hot list, adjoined during the run, and in list(usable) after being chosen as the focus of attention.

To obtain a proof of *P*8 from *P*7, we made the single change to our original approach of adding the use of the dynamic hot list strategy, again with the dynamic_heat_weight assigned to 8 and with the heat parameter assigned to 1. After all, that strategy seemed to make the difference in the preceding sought-after proof. In approximately 65 CPU-seconds, the desired proof was obtained with (reported and actual) length 12 and level 8, completing with retention of clause (5759), and the computer used was a SPARCstation-10 (on July 19, 1994). Then, except for assigning heat to 1 (rather than to 2), with the same approach just described in the context of *P*5 as the only axiom, we turned to *P*8 with the objective of deducing *P*9. In approximately 436 CPU-seconds on a SPARCstation-10 (on July 20, 1994), the desired proof was obtained, with reported and actual length 58 and level 13, completing with retention of clause (17324). We next simply again applied the approach, but replacing the axiom of *P*8 with the axiom of *P*9 and replacing the target of *P*9 with the target of *PYO*. Still on a SPARCstation-10 (on July 20, 1994), OTTER obtained the desired proof in approximately 24,495 CPU-seconds with reported and actual length 53 and level 13, with retention of clause (145282).

Regarding our attack on finding a proof of *PYM* using *PYO* as the only axiom, we were indeed influenced by our early experiments that deduced *XGK* and no other axiom. Since purity was paramount, as we did in earlier-cited experiments, we used a weight template (as the only member of a pick_and_purge weight_list) to prevent OTTER from retaining the clause equivalent of *XGK*, risking the purging of similar formulas. (We did not rely on demodulation, for, at this point in our earliest experimentation, we had not yet begun to use that mechanism in the context of blocking the retention of various clauses.) We assigned max_weight to 36, reassigned it to 24 after 70 clauses were chosen as the focus of attention, assigned the pick_given_ratio to 4, and (for the hot list strategy) assigned the heat parameter to 1. On the equivalent of a SPARCstation-2 (on July 18, 1994), OTTER obtained the desired proof in approximately 4743 CPU-seconds with length 64 and level 25, with retention of clause (94387). On the same computer and same date, we had obtained (with our original approach) a proof of *XGK* from *PYM* in approximately 81 CPU-seconds; the proof has length 23 and level 11 and completes with retention of clause (3561).

We obtained the last three proofs with the following approach, using a SPARCstation-10 (on July 21, 1994). Rather than assigning a max_weight, we had OTTER (in effect) begin with none and periodically lower it through the use of set(control_memory), which computes lower max_weights based on the memory usage in relation to the assigned (by the researcher) max_mem. We assigned to max_memory a limit of 15 megabytes, deliberately squeezing the program despite the use of a computer offering more than 80 megabytes. We assigned max_proofs to 1, the pick_given_ratio to 3, the heat parameter to 1 (for the hot list strategy), and the dynamic_heat_weight to 8 (for the dynamic hot list strategy).

Two lists of demodulators were included, one to block the retention of all but the target (shortest single) axiom and the hypothesis (shortest single) axiom, and one to purge all conclusions that contain as a proper subterm a term of the form $e(x,x)$ for any term x. (In other words, with regard to the second list of demodulators, we used a strategy we call the *subtautology strategy* [Wos95].) In approximately 342 CPU-seconds, *XHK* was deduced from *XGK* with a proof of reported length 94 and actual length 91 and level 34, with retention of clause (15803). In approximately 688 CPU-seconds, *XHN* was deduced from *XHK* with a proof of reported length 76 and actual length 72 and level 40, with retention of clause (21506). In approximately 480 CPU-seconds, *P*1 was deduced from *XHN* with a proof of reported length 58 and actual length 55 and level 32, with retention of clause (15009).

- With the following ordering of the thirteen shortest single axioms—we found—with the invaluable assistance of OTTER—a (third) circle of pure proofs. The (third) circle is *P*1, *P*2, *P*3, *P*4, *P*5, *P*7, *P*8, *P*9, *PYM*, *PYO*, *XGK*, *XHK*, *XHN*, *P*1.

4. Pristine Proofs

We note that, although our original goal was that of finding, if one existed, a circle of pure proofs, far more was discovered through heavy use of McCune's automated reasoning program OTTER. Stronger than purity is the property of *instance purity*, which demands that, other than the hypothesis and the conclusion, no deduced step be even an *instance* of one of the other eleven shortest single axioms. An inspection of the proofs for each of the three circles shows that *all* of them are instance pure.

Also stronger than purity is the property of *subterm purity*, which requires that, other than the hypothesis and the conclusion, no deduced step *even* contain a proper subterm that is an instance of one of the other eleven shortest single axioms. Except for the proof of *PYM* with *P*5 as hypothesis and the proof of *P*7 with *P*5 as hypothesis, all of the proofs used in the three circles are subterm pure. We note that we did revisit the study of deducing *P*7 from *P*5 to obtain a proof that is subterm pure; in our original pure proof, proper subterms that are instances of *P*4 are present. On a SPARCstation-10, in approximately 5724 CPU-seconds, the desired proof was completed with actual length 50 and reported length 52 (because of using the dynamic hot list strategy) and level 22, with retention of clause (84762).

- In other words, the proofs of which the three circles consist are indeed pristine, for the proofs are *instance pure* and, with one exception (that of *PYM* from *P*5), *subterm pure* (after replacing the original proof of *P*7 from *P*5).

Were one to press forward (as we did after this article was virtually complete), one might ask about one additional property, still in the spirit of being pristine. The property in question concerns the number of deduced steps shared by more than one proof, where the focus is on each of the circles (taken one at a time) of thirteen pure proofs. In the following way, the cited property extends the notion of a circle of pure proofs. Indeed, for a given circle of thirteen pure proofs, each of the shortest single axioms appears twice, once as a hypothesis, and once as a conclusion. If the concern is confined to the thirteen axioms, then, because of purity, two proofs selected from one of the circles of pure proofs have no intersection, or they *touch* at a single (formula) point, the end of one and the beginning of the other. For example, from the first circle, the *P*3-*P*2 proof and the *P*7-*P*3 proof share *P*3 and touch at no other point (with respect to the thirteen shortest single axioms). In the same context, the *P*3-*P*2 proof does *not* touch the *P*5-*PYM* proof. Curiosity naturally led us to broaden the scope of formula sharing to include all formulas of equivalential calculus and to focus on the deduced steps of the thirteen proofs for a given circle.

If one uses various Unix features, one finds that, on the surface, the following seven deduced steps are shared by various proofs among the thirteen proofs of the first circle.

P(e(e(e(x,e(y,e(z,x))),z),y)).
P(e(e(e(x,y),x),y)).
P(e(e(e(x,y),y),x)).
P(e(e(x,y),e(y,e(e(z,x),z)))).
P(e(e(x,y),e(y,x))).
P(e(e(x,y),e(y,x))).
P(e(x,e(e(y,z),e(y,e(z,x))))).

Of these seven deduced steps (in the circle 1 proofs), 5 and 6 are identical. The first of the seven is present in the proof of *XHN* from *P*8, and in the proof of *PYM* from *P*5. The second is in the proof of *P*7 from *XHK*, and in the proof of *P*3 from *P*7. The third occurs twice in the proof of *PYM* from *P*5, because of the dynamic hot list strategy. The fourth occurs in the proof of *P*7 from *XHK*, and in the proof of *P*3 from *P*7. The fifth (and hence the sixth which is identical) occurs in three proofs: of *XHK* from *P*9, of *P*7 from *XHK*, and of *P*3 from *P*7. The seventh occurs in the proof of *P*9 from *PYO*, and in the proof of *P*3 from *P*7. Therefore, only 5 steps of the 265 deduced steps (of which the thirteen proofs of the first circle consist) are shared by more than one proof; OTTER (in effect) announces 266 because of a duplicate in one proof, resulting from the use of the dynamic hot list strategy.

With the aid of Unix, regarding the thirteen proofs of the second circle, one finds that, on the surface, twelve deduced steps are shared by more than one proof, the following.

P(e(e(e(x,e(y,e(z,x))),z),y)).
P(e(e(e(x,y),x),y)).
P(e(e(e(x,y),y),x)).
P(e(e(e(x,y),y),x)).
P(e(e(e(x,y),y),x)).
P(e(e(x,e(x,y)),y)).
P(e(e(x,y),e(y,x))).
P(e(e(x,y),e(y,x))).
P(e(x,e(e(y,z),e(y,e(z,x))))).
P(e(x,x)).
P(e(x,x)).
P(e(x,x)).

Of the twelve deduced steps, 3 and 4 and 5 are identical, as are 7 and 8, and are 10 and 11 and 12 identical. Therefore, only seven distinct steps occur. The first occurs in the proof of *XHN* from *P*8, and also in the proof of *PYM* from *P*5. The second occurs twice in the proof of *P*2 from *XHK*, because of the dynamic hot list strategy. The third (and fourth and fifth which are identical) occur twice in the proof of *PYM* from *P*5, and twice in the proof of *P*2 from *XHK*; duplicate deduced steps can be present in a proof when the dynamic hot list strategy is used. The sixth occurs in the proof of *P*9 from *PYO*, and occurs in the proof of *P*2 from *XHK*. The seventh (and hence eighth) occurs in the proof of *P*8 from *P*7, and occurs twice in the proof of *P*2 from *XHK* (because of the dynamic hot list strategy). The ninth occurs in the proof of *P*8 from *P*7, and occurs in the proof of *P*9 from *PYO*. The tenth (and hence eleventh and twelfth) occurs in three proofs: *P*9 from *PYO*, *XHK* from *P*3, and twice in *P*2 from *XHK* (because of the dynamic hot list strategy). Therefore, only 6 steps of the 284 deduced steps (of which the thirteen proofs of the second circle consist) are shared by more than one proof; OTTER (in effect) announces 289, because of the dynamic hot list strategy.

As for the third circle of pure proofs, they behave far less well in the sense that fifty-seven deduced steps are shared by more than one proof. Because of using the dynamic hot list strategy, which can cause a deduced step to appear more than once in a proof, OTTER announces a total of 517 deduced steps; actually, the number is 505. With respect to the property under discussion, which one might think of as a *proof intersection* property, the thirteen proofs of the third circle are far from pristine.

5. Persuasions and Challenges

Our persuasions (or beliefs) naturally lead us to offer challenges. We have always held that an attempt to meet a challenge is often met with intrigue and with the eventual advance of some type for the field. We are so attracted to this position that we sometimes offer to ourselves a challenge—in the case of this article—the challenge of finding a circle of pure proofs for the thirteen shortest single axioms of equivalential calculus.

The significance of the material that has resulted and that we have presented here rests with three factors. First, the pristine quality of the proofs that OTTER has produced for each of the three circles of pure proofs offers charm and even beauty, from the viewpoint of mathematics and logic (see the bulleted items in Sections 2.4, 2.5, 3, and 4). Second, the reported successes powerfully illustrate how research is facilitated when one's assistant is McCune's automated reasoning program OTTER, sharply reducing real time, CPU time, and the likelihood of error. Third, the methodology developed to meet the challenge offers techniques for attacking a wide variety of questions and problems from areas having no relation to equivalential calculus.

Regarding concrete challenges, one might attempt to find a subterm-pure proof of *PYM* from *P*5 to replace the corresponding proof that is *not* subterm pure and that is used in the first and second circle of proofs; our attempts have failed so far. One might attempt to find a circle of pure proofs such that no deduced step is common to two or more proofs; we have not studied this problem. Without losing purity, one might attempt to find replacement proofs for each of the given three circles of pure proofs, where the goal is to decrease their length, individually and collectively. Instead, one might seek a different circle of pure proofs whose collective proof length is ''small''.

6. Finale and Future Research

In this article, we have presented a methodology for attacking the question of finding a *circle of pure proofs* for the thirteen shortest single axioms for equivalential calculus. For an example of what the methodology has produced, we consider the following (circle) ordering of the thirteen shortest single axioms, an ordering that is the first we were given in the late 1970s. The ordering is *P*1, *P*2, *P*3, *P*4, *P*5, *P*7, *P*8, *P*9, *PYM*, *PYO*, *XGK*, *XHK*, *XHN*, *P*1. With indispensable aid from McCune's automated reasoning program OTTER, thirteen proofs were produced: the first deducing *P*2 from *P*1 (as the only axiom and with *condensed detachment* as the inference rule), the second deducing *P*3 from *P*2 (as the only axiom), ..., and the thirteenth deducing *P*1 from *XHN* (as the only axiom). In other words, OTTER produced the needed proofs for the *circle* under study. Regarding *purity*, each of the thirteen proofs relies on its hypothesis (shortest single) axiom and its conclusion (shortest single) axiom and on *none* of the other eleven shortest single axioms. (We note that, although purity places no restrictions on proper instances and on proper subterms of any of the other eleven axioms, the research reported here also addresses those aspects.) For one indication of the difficulty of obtaining the desired pure proofs, to obtain *any* proof with *XHN* as hypothesis can prove most challenging. For a second indication of the difficulty, many approaches to deducing one shortest single axiom from another result in the completion of a proof that is *not* pure, that relies on the use of a third shortest single axiom.

As but one indication of the power offered by OTTER and by some of the strategies we used, which we shall touch upon almost immediately, the entire study (with the exception of the smallest portion borrowed from earlier research and the added experiment on August 12, 1994, that we reported in Section 2.4) was completed in three days, starting on July 18, 1994, and concluding on July 21, 1994. Indeed, in contrast to so much of research, the preparation of this article required far more time and, more significant, far more effort than did the study itself. Moreover, we note that the CPU time required to complete the circle of proofs was dramatically less than we would have estimated, given the complexity of the problem. In particular, only 98,000 CPU-seconds were needed to find the (first) circle of pure proofs. Finally, we note that we found three circles of pure proofs for the thirteen shortest single axioms, and, though not our original goal, the proofs are also *instance pure*, free of proper instances of the axioms whose use is to be avoided. Further, if none of the deduced steps of a proof contains as a proper subterm an instance of an unwanted axiom, we say the proof is *subterm pure*. With one exception, with OTTER, we were able to find subterm pure proofs. Unaided, a researcher would find it at least troublesome and at worst most difficult to test proofs for instance purity and for subterm purity, again illustrating the value of reliance on an automated reasoning program.

Regarding strategy, both the *hot list strategy* and the *dynamic hot list strategy* (the latter due to McCune) played a key role. The hot list strategy enables the researcher to designate various input clauses as meriting repeated immediate visiting, and even immediate revisiting, to *complete* the applications of the chosen inference rules, rather than to *initiate* such applications. One might, for example, conjecture that an added axiom (such as the cube of x is x) should be immediately used with each new clause that is retained, even *before* the new clause is placed in, say, list(sos). If that is the conjecture, then one places the corresponding clause in list(hot). The *heat* parameter governs the *level* (or depth) of the use of the hot list strategy. The dynamic hot list strategy enables the researcher to instruct the program to adjoin *during* a run new members to the hot list. The parameter *dynamic_heat_weight* provides the threshold for deciding which clauses to adjoin during the run.

In addition, some experiments relied on the use of *level saturation* (breadth first or first come first serve) to direct the choice of where next to focus the program's attention, and some used the *ratio* strategy (the latter due to McCune). Regarding the ratio strategy, its actions are governed by the parameter *pick_given_ratio*. If, for example, this parameter is assigned the value 4, then OTTER will choose (as the focus of attention) four clauses by conclusion complexity, one by first come first serve, then four, then one, and the like. We also made occasional use of the *subtautology strategy*, which instructs the program to immediately purge on generation clauses containing terms of the form $e(x,x)$, where the function e (in this study) denotes equivalent and where x is a variable that, because demodulation is the mechanism that is used, captures terms.

Among the other techniques on which the methodology rests, two merit mention. First, to prevent OTTER from retaining the clause equivalent of some unwanted axiom, we used *weighting* (a strategy formulated by Overbeek) by including its pattern with an assignment of complexity that exceeds the chosen maximum weight allowed. This action has the risk of purging formulas that differ from the unwanted formula only in the choice of particular variables. This risk exists because, with OTTER, a weight template treats all of its variables as indistinguishable; nevertheless, this treatment is fortunate as shown in this article. Second, when we wished to block consideration of all but the hypothesis axiom and its target, we used *demodulation* (see Section 2.5) in a rather complex manner.

For future research, we suggest the study of other circles of pure proofs. We suggest that this avenue is of interest in contexts totally unrelated to equivalential calculus, as well as other circles focusing on the thirteen shortest single axioms. For example, perhaps the question of whether a circle of pure proofs exists for the three Moufang identities is still open. For related research, McCune suggests seeking an algorithm for transforming proofs that are not pure into proofs that are. For distantly related research, we suggest a study to determine why the methodology used to obtain the results we present here seems to promote purity of proof. Of a different nature entirely, we suggest for research a study of shortest proofs for various pairs of shortest single axioms.

For the researcher interested in practical applications, we suggest extracting from the methodology various techniques that are of general use. For example, one might apply the VAR option (with demodulation) of Section 2.5 to design an efficient circuit that avoids certain undesirable subcircuits. For another example, one might apply some of the techniques to algorithm synthesis.

For researchers interested in attacking problems whose solution might directly advance the field of automated reasoning, the following seem most appropriate. Of a global nature, for research we suggest a study of metarules for wisely choosing from among the options offered by OTTER, especially choosing from among the strategies this program offers. (Although each of the strategies is indeed independent of any reasoning program, OTTER is, from what we know, currently the only program that offers the entire menu.) Equally complicated is the research topic that focuses on what we call *global linearity*, in contrast to *local linearity*. The latter concerns the drawing of conclusions at a more or less constant rate per CPU-second. The former concerns choosing as the focus of attention (to drive the program's reasoning) conclusions at a more or less constant rate. McCune's use of discrimination trees enables OTTER to perform at nearly local linear speed. As yet, we only have a faint notion for addressing the problem of global linearity. Were a program to offer global linearity comparable to OTTER's offering of local linearity, (we conjecture that) a most significant increase in power would result.

We close by noting that the data presented in this article strongly suggests that obtaining the answers to deep questions and the solution to hard problems virtually require the use of strategy. Clearly (we believe), no single strategy suffices, nor will such an all-encompassing strategy ever exist.

However, the advances in automated reasoning witnessed in just the past five years are far beyond what we would have predicted. Indeed, we feel certain that the results presented here would have been out of reach but five years ago, and, further, these results would have been difficult or impossible to obtain without McCune’s program OTTER.

Remark. Although during the entire study reported here we had totally forgotten about some earlier joint research with our colleague McCune [McCune92], we note that in that paper we presented what might be called a near circle of proofs; purity had not yet been coined and was, almost certainly, lacking in the vast majority of the proofs.

Appendix

To aid and stimulate research, we present here a sample input file and the proofs of which the first circle consists. When a line contains a ‘‘%’’, the characters from the first ‘‘%’’ to the end of the line are treated by the program as a comment. In the following, ‘‘-’’ denotes logical **not** and ‘‘ | ’’ denotes logical **or**. In the proofs, two copies of an input clause denotes its presence in two input lists, one of which is the hot list. Also, as an example, [hyper,16,17,18] says that clause (16) is the nucleus, clause (17) is unified with the first literal of (16), and clause (18) is unified with the second literal.

A Sample Input File

```
set(hyper_res).
assign(max_weight,32).
% assign(max_given,1).
assign(max_proofs,-1).
clear(print_kept).
% set(ancestor_subsume).
% set(back_sub).
% assign(max_seconds,1200).
assign(max_mem,80000).
assign(report,1800).
% assign(max_distinct_vars,4).
% assign(pick_given_ratio,4).
assign(change_limit_after,50).
assign(new_max_weight,24).
assign(heat,1).
% assign(dynamic_heat_weight,8).
set(order_history).
set(input_sos_first).
set(sos_queue).

% weight_list(pick_and_purge).
% Following are to block all but P3 and P7.
% weight(P(e(e(x,y),e(e(z,y),e(x,z)))),200). % P1_YQL
% weight(P(e(e(x,y),e(e(x,z),e(z,y)))),200). % P2_YQF
% weight(P(e(e(e(x,y),z),e(y,e(z,x)))),200). % P4_UM
% weight(P(e(x,e(e(y,e(x,z)),e(z,y)))),200). % P5_XGF
% weight(P(e(e(x,y),e(z,e(e(y,z),x)))),200). % P8_YRM
% weight(P(e(e(x,y),e(z,e(e(z,y),x)))),200). % P9_YRO
% weight(P(e(e(e(x,e(y,z)),z),e(y,x))),200). % PYO
% weight(P(e(e(e(x,e(y,z)),y),e(z,x))),200). % PYM
% weight(P(e(x,e(e(y,e(z,x)),e(z,y)))),200). % XGK
% weight(P(e(x,e(e(y,z),e(e(x,z),y)))),200). % XHK
% weight(P(e(x,e(e(y,z),e(e(z,x),y)))),200). % XHN
% Following is for tail strategy.
% weight(e($(1),$(2)),1).
% end_of_list.

list(usable).
% Following clause is for condensed detachment.
-P(e(x,y)) | -P(x) | P(y).
end_of_list.

list(sos).
```

```
% Following are all of the shortest single axioms for equiv calc.
% P(e(e(x,y),e(e(z,y),e(x,z)))). % P1_YQL
% P(e(e(x,y),e(e(x,z),e(z,y)))). % P2_YQF
% P(e(e(x,y),e(e(z,x),e(y,z)))). % P3_YQJ
% P(e(e(e(x,y),z),e(y,e(z,x)))). % P4_UM
% P(e(x,e(e(y,e(x,z)),e(z,y)))). % P5_XGF
P(e(e(x,e(y,z)),e(z,e(x,y)))). % P7_WN
% P(e(e(x,y),e(z,e(e(y,z),x)))). % P8_YRM
% P(e(e(x,y),e(z,e(e(z,y),x)))). % P9_YRO
% P(e(e(e(x,e(y,z)),z),e(y,x))). % PYO
% P(e(e(e(x,e(y,z)),y),e(z,x))). % PYM
% P(e(x,e(e(y,e(z,x)),e(z,y)))). % XGK
% P(e(x,e(e(y,z),e(e(x,z),y)))). % XHK
% P(e(x,e(e(y,z),e(e(z,x),y)))). % XHN
end_of_list.

list(passive).
% Here are negations of the thirteen shortest single
% axioms for equivalential calculus.
-P(e(e(a,b),e(e(c,b),e(a,c)))) | $ANSWER(P1_YQL).
-P(e(e(a,b),e(e(a,c),e(c,b)))) | $ANSWER(P2_YQF).
-P(e(e(a,b),e(e(c,a),e(b,c)))) | $ANSWER(P3_YQJ).
-P(e(e(e(a,b),c),e(b,e(c,a)))) | $ANSWER(P4_UM).
-P(e(a,e(e(b,e(a,c)),e(c,b)))) | $ANSWER(P5_XGF).
-P(e(e(a,e(b,c)),e(c,e(a,b)))) | $ANSWER(P7_WN).
-P(e(e(a,b),e(c,e(e(b,c),a)))) | $ANSWER(P8_YRM).
-P(e(e(a,b),e(c,e(e(c,b),a)))) | $ANSWER(P9_YRO).
-P(e(e(e(a,e(b,c)),c),e(b,a))) | $ANSWER(PYO).
-P(e(e(e(a,e(b,c)),b),e(c,a))) | $ANSWER(PYM).
-P(e(a,e(e(b,e(c,a)),e(c,b)))) | $ANSWER(XGK).
-P(e(a,e(e(b,c),e(e(a,c),b)))) | $ANSWER(XHK).
-P(e(a,e(e(b,c),e(e(c,a),b)))) | $ANSWER(XHN).
end_of_list.

% Following blocks unwanted formulas of EC, excluding P7 and P3.
make_evaluable(_&_, $AND(_,_)).
list(demodulators).
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(z,y),e(x,z)))) = $T. % P1_YQL
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(x,z),e(z,y)))) = $T. % P2_YQF
% ($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(e(z,x),e(y,z)))) = $T. % P3_YQJ
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,y),z),e(y,e(z,x)))) = $T. % P4_UM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,e(x,z)),e(z,y)))) = $T. % P5_XGF
% ($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,e(y,z)),e(z,e(x,y)))) = $T. % P7_WN
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(z,e(e(y,z),x)))) = $T. % P8_YRM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(x,y),e(z,e(e(z,y),x)))) = $T. % P9_YRO
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,e(y,z)),z),e(y,x))) = $T. % PYO
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(e(e(x,e(y,z)),y),e(z,x))) = $T. % PYM
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,e(z,x)),e(z,y)))) = $T. % XGK
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,z),e(e(x,z),y)))) = $T. % XHK
($VAR(x) & $VAR(y) & $VAR(z)) -> P(e(x,e(e(y,z),e(e(z,x),y)))) = $T. % XHN
end_of_list.

list(hot).
% Following clause is for condensed detachment.
-P(e(x,y)) | -P(x) | P(y).
```

```
% Following is hypothesis, P7
P(e(e(x,e(y,z)),e(z,e(x,y)))). % P7_WN
end_of_list.
```

Thirteen Proofs for the First Circle, in Order

```
----> UNIT CONFLICT at 100.56 sec ----> 6583 [binary,6582.1,4.1] $ANSWER(P2_YQF).
Length of proof is 8. Level of proof is 6.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,y),e(e(z,x),e(y,z)))).
4 [] -P(e(e(a,b),e(e(a,c),e(c,b)))) | $ANSWER(P2_YQF).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(x,y),e(e(z,x),e(y,z)))).
----------------------------
18 [hyper,1,2,2] P(e(e(x,e(y,z)),e(e(e(u,y),e(z,u)),x))).
19 (heat=1) [hyper,16,17,18] P(e(e(x,e(y,e(z,u))),e(e(e(v,z),e(u,v)),y),x))).
24 [hyper,1,19,19] P(e(e(e(e(x,y),e(z,x)),e(e(e(u,v),e(w,u)),v6)),e(e(y,z),e(v6,e(v,w))))).
26 [hyper,1,2,19] P(e(e(x,e(y,e(z,e(u,v)))),e(e(e(e(w,u),e(v,w)),z),y),x))).
29 (heat=1) [hyper,16,24,17] P(e(e(x,y),e(e(e(y,z),e(x,u)),e(u,z)))).
152 [hyper,1,26,29] P(e(e(e(e(e(x,y),e(z,x)),u),e(e(v,e(y,z)),e(w,u))),e(w,v))).
159 [hyper,1,2,29] P(e(e(x,e(y,z)),e(e(e(e(z,u),e(y,v)),e(v,u)),x))).
6582 [hyper,1,152,159] P(e(e(x,y),e(e(x,z),e(z,y)))).
6583 [binary,6582.1,4.1] $ANSWER(P2_YQF).

----> UNIT CONFLICT at  13.17 sec ----> 1560 [binary,1559.1,3.1] $ANSWER(P1_YQL).
Length of proof is 7. Level of proof is 5.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,y),e(e(x,z),e(z,y)))).
3 [] -P(e(e(a,b),e(e(c,b),e(a,c)))) | $ANSWER(P1_YQL).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(x,y),e(e(x,z),e(z,y)))).
----------------------------
18 [hyper,1,2,2] P(e(e(e(x,y),z),e(z,e(e(x,u),e(u,y))))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(x,y),e(y,z)),e(e(x,u),e(u,z)))).
21 [hyper,1,18,18] P(e(e(x,e(e(y,z),e(z,u))),e(e(e(y,u),v),e(v,x)))).
52 [hyper,1,20,21] P(e(e(x,y),e(y,e(e(z,z),x)))).
56 [hyper,1,21,18] P(e(e(e(x,e(y,z)),u),e(u,e(e(v,z),e(x,e(v,y)))))).
1502 [hyper,1,56,52] P(e(e(e(x,y),e(e(z,z),u)),e(e(v,y),e(u,e(v,x))))).
1559 (heat=1) [hyper,16,1502,17] P(e(e(x,y),e(e(z,y),e(x,z)))).
1560 [binary,1559.1,3.1] $ANSWER(P1_YQL).

----> UNIT CONFLICT at 1208.41 sec ----> 27183 [binary,27182.1,9.1] $ANSWER(P8_YRM).
Length of proof is 10. Level of proof is 6.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,y),e(e(z,y),e(x,z)))).
9 [] -P(e(e(a,b),e(c,e(e(b,c),a)))) | $ANSWER(P8_YRM).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(x,y),e(e(z,y),e(x,z)))).
----------------------------
18 [hyper,1,2,2] P(e(e(x,e(e(y,z),e(u,y))),e(e(u,z),x))).
19 (heat=1) [hyper,16,17,18] P(e(e(x,e(e(y,z),u)),e(e(u,e(e(v,z),e(y,v))),x))).
21 [hyper,1,18,18] P(e(e(x,y),e(e(x,z),e(e(u,y),e(z,u))))).
```

26 [hyper,1,2,19] P(e(e(x,e(e(y,e(e(z,u),e(v,z))),w)),e(e(w,e(e(v,u),y)),x))).
37 [hyper,1,19,21] P(e(e(e(e(x,y),e(z,x)),e(e(u,z),e(v,u))),e(v,y))).
38 [hyper,1,18,21] P(e(e(e(x,y),z),e(e(z,x),y))).
422 [hyper,1,26,38] P(e(e(x,e(e(y,z),u)),e(e(e(e(v,z),e(y,v)),x),u))).
430 [hyper,1,38,37] P(e(e(e(x,y),e(e(z,y),e(u,z))),e(e(v,u),e(x,v)))).
453 (heat=1) [hyper,16,422,17] P(e(e(e(e(x,y),e(z,x)),e(u,y)),e(u,z))).
27182 [hyper,1,453,430] P(e(e(x,y),e(z,e(e(y,z),x)))).
27183 [binary,27182.1,9.1] $ANSWER(P8_YRM).

----> UNIT CONFLICT at 32.77 sec ----> 1930 [binary,1929.1,15.1] $ANSWER(XHN).
Length of proof is 19. Level of proof is 10.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,y),e(z,e(e(y,z),x)))).
15 [] -P(e(a,e(e(b,c),e(e(c,a),b)))) | $ANSWER(XHN).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(x,y),e(z,e(e(y,z),x)))).
----
18 [hyper,1,2,2] P(e(x,e(e(e(y,e(e(z,y),u)),x),e(u,z)))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(x,e(e(y,x),z)),e(e(u,v),e(w,e(e(v,w),u)))),e(z,y))).
21 [hyper,1,18,18] P(e(e(e(x,e(e(y,x),z)),e(u,e(e(e(v,e(e(w,v),v6)),u),e(v6,w)))),e(z,y))).
22 (heat=1) [hyper,16,21,17] P(e(e(e(x,e(y,e(z,x))),z),y)).
26 [hyper,1,20,20] P(e(e(x,e(e(e(y,e(e(e(z,e(e(u,z),v)),y),e(v,u))),x),w)),w)).
29 (heat=1) [hyper,16,26,17] P(e(e(x,e(e(y,e(e(e(z,e(e(u,z),v)),y),e(v,u))),e(w,x))),w)).
34 [hyper,1,2,22] P(e(x,e(e(y,x),e(e(z,e(y,e(u,z))),u)))).
37 (heat=1) [hyper,16,34,17] P(e(e(x,e(e(y,z),e(u,e(e(z,u),y)))),e(e(v,e(x,e(w,v))),w))).
62 [hyper,1,34,34] P(e(e(x,e(y,e(e(z,y),e(e(u,e(z,e(v,u))),v))))),e(e(w,e(x,e(v6,w))),v6))).
64 [hyper,1,29,34] P(e(x,e(y,e(e(e(e(z,e(e(u,z),v)),y),e(v,u)),x)))).
66 [hyper,1,34,22] P(e(e(x,e(e(e(y,e(z,e(u,y))),u),z)),e(e(v,e(x,e(w,v))),w))).
68 (heat=1) [hyper,16,62,17] P(e(e(x,e(e(e(e(y,e(z,e(u,y))),u),z),e(v,x))),v)).
70 (heat=1) [hyper,16,17,64] P(e(x,e(e(e(y,e(e(e(e(z,e(e(u,z),v)),y),e(v,u)),w)),x),w))).
135 [hyper,1,68,34] P(e(x,e(e(e(y,e(z,e(u,y))),u),e(z,x)))).
143 [hyper,1,66,70] P(e(e(x,e(e(e(y,e(e(z,y),u)),e(u,z)),e(v,x))),v)).
301 [hyper,1,37,135] P(e(e(x,e(e(e(y,z),e(u,e(z,e(y,u)))),e(v,x))),v)).
330 [hyper,1,143,64] P(e(e(e(x,e(e(y,x),z)),e(e(u,e(e(v,u),w)),e(w,v))),e(z,y))).
1905 [hyper,1,330,301] P(e(e(e(e(x,e(e(y,x),z)),e(z,y)),e(u,e(v,e(w,u)))),e(w,v))).
1929 (heat=1) [hyper,16,1905,17] P(e(x,e(e(y,z),e(e(z,x),y)))).
1930 [binary,1929.1,15.1] $ANSWER(XHN).

----> UNIT CONFLICT at 770.69 sec ----> 9778 [binary,9777.1,6.1] $ANSWER(P4_UM).
Length of proof is 20. Level of proof is 14.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(x,e(e(y,z),e(e(z,x),y)))).
6 [] -P(e(e(e(a,b),c),e(b,e(c,a)))) | $ANSWER(P4_UM).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(x,e(e(y,z),e(e(z,x),y)))).
----
18 [hyper,1,2,2] P(e(e(x,y),e(e(y,e(z,e(e(u,v),e(e(v,z),u)))),x))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(e(x,y),e(e(y,z),x)),e(u,e(e(v,w),e(e(w,u),v)))),z)).
21 [hyper,1,18,18] P(e(e(e(e(x,e(y,e(e(z,u),e(e(u,y),z)))),v),e(w,e(e(v6,v7),e(e(v7,w),v6)))),e(v,x))).
24 [hyper,1,18,20] P(e(e(x,e(y,e(e(z,u),e(e(u,y),z)))),e(e(e(v,w),e(e(w,x),v)),e(v6,e(e(v7,v8),e(e(v8,v6),v7)))))).
27 [hyper,1,21,20] P(e(e(x,e(e(y,z),e(e(z,x),y))),e(e(e(u,v),e(e(v,e(w,e(v6,e(e(v7,v8),e(e(v8,v6),v7))))),u)w
30 (heat=1) [hyper,16,27,17] P(e(e(e(x,y),e(e(y,e(z,e(u,e(e(v,w),e(e(w,u),v))))),x)),z)).
37 [hyper,1,2,30] P(e(e(x,y),e(e(y,e(e(e(z,u),e(e(u,e(v,e(w,e(e(v6,v7),e(e(v7,w),v6))))),z)),v)),x))).

```
42 (heat=1) [hyper,16,37,17] P(e(e(e(e(x,y),e(e(y,z),x)),e(e(e(u,v),e(e(v,e(w,e(v6,e(e(v7,v8),
      e(e(v8,v6),v7))))),u)),w)),z)).
58 [hyper,1,20,42] P(e(e(x,e(e(x,y),e(z,e(e(u,v),e(e(v,z),u))))),e(y,e(w,e(e(v6,v7),e(e(v7,w),v6)))))).
69 [hyper,1,58,24] P(e(e(e(e(x,e(e(y,z),e(e(z,x),y))),u),u),e(v,e(e(w,v6),e(e(v6,v),w))))).
70 (heat=1) [hyper,16,17,69] P(e(e(x,y),e(e(y,e(e(e(e(z,e(e(u,v),e(e(v,z),u))),w),w),
      e(v6,e(e(v7,v8),e(e(v8,v6),v7))))),x))).
77 [hyper,1,24,69] P(e(e(e(x,y),e(e(y,e(e(e(z,e(e(u,v),e(e(v,z),u))),w),w)),x)),e(v6,e(e(v7,v8),e(e(v8,v6),7
78 [hyper,1,30,70] P(e(e(e(x,e(e(y,z),e(e(z,x),y))),u),u)).
102 [hyper,1,21,77] P(e(e(e(e(x,e(e(y,z),e(e(z,x),y))),e(e(e(u,e(e(v,w),e(e(w,u),v))),v6),v6)),v7),v7)).
103 (heat=1) [hyper,16,102,17] P(e(e(x,y),e(e(y,e(e(z,e(e(u,v),e(e(v,z),u))),e(e(e(w,e(e(v6,v7),
      e(e(v7,w),v6))),v8),v8))),x))).
105 [hyper,1,42,78] P(e(e(e(e(x,y),z),e(y,e(u,e(e(v,w),e(e(w,u),v))))),e(z,x))).
202 [hyper,1,20,105] P(e(e(x,e(e(y,z),e(e(z,e(u,x)),y))),u)).
204 [hyper,1,105,103] P(e(e(e(e(e(x,e(e(y,z),e(e(z,x),y))),u),u),v),e(e(w,v6),e(e(v6,v),w)))).
871 [hyper,1,105,204] P(e(x,e(e(y,e(e(z,u),e(e(u,y),z))),e(e(e(v,w),e(e(w,e(v6,x)),v)),v6)))).
9777 [hyper,1,202,871] P(e(e(e(x,y),z),e(y,e(z,x)))).
9778 [binary,9777.1,6.1] $ANSWER(P4_UM).

----> UNIT CONFLICT at    0.31 sec ----> 19 [binary,18.1,7.1] $ANSWER(P5_XGF).
Length of proof is 1.  Level of proof is 1.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(e(x,y),z),e(y,e(z,x)))).
7 [] -P(e(a,e(e(b,e(a,c)),e(c,b)))) | $ANSWER(P5_XGF).
----------------------------
18 [hyper,1,2,2] P(e(x,e(e(y,e(x,z)),e(z,y)))).
19 [binary,18.1,7.1] $ANSWER(P5_XGF).

----> UNIT CONFLICT at 350.31 sec ----> 10267 [binary,10266.1,12.1] $ANSWER(PYM).
Length of proof is 32.  Level of proof is 14.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(x,e(e(y,e(x,z)),e(z,y)))).
12 [] -P(e(e(e(a,e(b,c)),b),e(c,a))) | $ANSWER(PYM).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(x,e(e(y,e(x,z)),e(z,y)))).
----------------------------
18 [hyper,1,2,2] P(e(e(x,e(e(y,e(e(z,e(y,u)),e(u,z))),v)),e(v,x))).
19 (heat=1) [hyper,16,17,18] P(e(e(x,e(e(e(y,e(e(z,e(e(u,e(z,v)),e(v,u))),w)),e(w,y)),v6)),e(v6,x))).
20 [hyper,1,18,18] P(e(x,e(x,e(e(y,e(e(z,e(y,u)),e(u,z))),e(v,e(e(w,e(v,v6)),e(v6,w))))))).
21 (heat=1) [hyper,16,17,20] P(e(e(x,e(e(y,e(y,e(e(z,e(e(u,e(z,v)),e(v,u))),e(w,e(e(v6,e(w,v7)),
      e(v7,v6)))))),v8)),e(v8,x))).
26 [hyper,1,19,2] P(e(e(x,e(x,e(e(y,e(e(z,e(y,u)),e(u,z))),v))),v)).
29 [hyper,1,18,20] P(e(e(e(x,e(e(y,e(x,z)),e(z,y))),e(u,e(e(v,e(u,w)),e(w,v)))),e(v6,e(e(v7,e(v6,v8)),e(v8,v7))))).
30 [hyper,1,21,2] P(e(e(e(e(x,e(e(y,e(x,z)),e(z,y))),e(u,e(e(v,e(u,w)),e(w,v)))),v6),v6)).
32 (heat=1) [hyper,16,30,17] P(e(e(x,e(e(e(y,e(e(z,e(y,u)),e(u,z))),e(v,e(e(w,e(v,v6)),e(v6,w)))),v7)),e(v7,x))).
35 [hyper,1,18,26] P(e(x,e(y,e(y,e(e(z,e(e(u,e(z,v)),e(v,u))),e(e(w,e(e(v6,e(w,v7)),e(v7,v6))),x)))))).
36 (heat=1) [hyper,16,17,35] P(e(e(x,e(e(y,e(z,e(z,e(e(u,e(e(v,e(u,w)),e(w,v))),
      e(e(v6,e(e(v7,e(v6,v8)),e(v8,v7))),y))))),v9)),e(v9,x))).
48 [hyper,1,26,29] P(e(e(e(e(x,e(y,z)),e(z,x)),e(u,e(e(v,e(u,w)),e(w,v)))),y)).
50 [hyper,1,18,30] P(e(x,e(e(e(y,e(e(z,e(y,u)),e(u,z))),e(v,e(e(w,e(v,v6)),e(v6,w)))),
      e(e(v7,e(e(v8,e(v7,v9)),e(v9,v8))),x)))).
60 [hyper,1,18,35] P(e(e(e(x,e(e(y,e(x,z)),e(z,y))),e(e(u,e(e(v,e(u,w)),e(w,v))),
      e(e(v6,e(e(v7,e(v6,v8)),e(v8,v7))),v9))),v9)).
62 [hyper,1,36,2] P(e(e(e(x,e(e(y,e(e(z,e(y,u)),e(u,z))),e(e(v,e(e(w,e(v,v6)),e(v6,w))),v7))),v7),x)).
```

63 (heat=1) [hyper,16,17,62] P(e(e(x,e(e(e(e(y,e(e(z,e(e(u,e(z,v)),e(v,u))),e(e(w,
e(e(v6,e(w,v7)),e(v7,v6))),v8))),v8),y),v9)),e(v9,x))).
95 [hyper,1,32,50] P(e(e(e(x,e(e(y,e(x,z)),e(z,y))),u),u)).
104 [hyper,1,60,18] P(e(e(e(x,e(e(y,e(z,e(e(u,e(z,v)),e(v,u)))),w)),e(w,x)),y)).
124 [hyper,1,62,48] P(e(e(x,e(e(e(e(y,z),u),e(z,e(u,y))),v)),e(v,x))).
126 (heat=1) [hyper,16,124,17] P(e(e(e(x,y),e(e(y,z),x)),z)).
127 [hyper,1,63,2] P(e(e(x,e(e(e(y,x),e(e(z,e(e(u,e(z,v)),e(v,u))),e(e(w,e(e(v6,e(w,v7)),e(v7,v6))),v8))),v8)),y)).
143 [hyper,1,95,126] P(e(e(e(x,y),y),x)).
148 [hyper,1,2,126] P(e(e(x,e(e(e(e(y,z),e(e(z,u),y)),u),v)),e(v,x))).
154 [hyper,1,104,127] P(e(e(e(e(e(x,y),z),e(y,e(z,x))),u),u)).
177 [hyper,1,124,143] P(e(x,e(e(e(e(e(e(y,z),u),e(z,e(u,y))),x),v),v))).
222 [hyper,1,60,154] P(e(e(e(x,y),y),x)).
230 (heat=1) [hyper,16,17,222] P(e(e(x,e(e(e(e(y,z),z),y),u)),e(u,x))).
338 [hyper,1,148,177] P(e(x,e(e(e(y,e(z,u)),x),e(e(u,y),z)))).
345 (heat=1) [hyper,16,17,338] P(e(e(x,e(e(y,e(e(e(z,e(u,v)),y),e(e(v,z),u))),w)),e(w,x))).
1206 [hyper,1,230,338] P(e(e(e(x,e(y,e(z,x))),z),y)).
1227 (heat=1) [hyper,16,17,1206] P(e(e(x,e(e(e(e(y,e(z,e(u,y))),u),z),v)),e(v,x))).
1280 [hyper,1,345,222] P(e(x,e(e(e(e(y,e(e(e(z,e(u,v)),y),e(e(v,z),u))),x),w),w))).
10266 [hyper,1,1227,1280] P(e(e(e(x,e(y,z)),y),e(z,x))).
10267 [binary,10266.1,12.1] $ANSWER(PYM).

----> UNIT CONFLICT at 81.43 sec ----> 3562 [binary,3561.1,13.1] $ANSWER(XGK).
Length of proof is 23. Level of proof is 11.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(e(x,e(y,z)),y),e(z,x))).
13 [] -P(e(a,e(e(b,e(c,a)),e(c,b)))) | $ANSWER(XGK).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(e(x,e(y,z)),y),e(z,x))).
----
18 [hyper,1,2,2] P(e(x,e(y,e(e(e(z,y),x),z)))).
19 (heat=1) [hyper,16,18,17] P(e(x,e(e(e(y,x),e(e(e(z,e(u,v)),u),e(v,z))),y))).
23 [hyper,1,19,18] P(e(e(e(x,e(y,e(z,e(e(e(u,z),y),u)))),e(e(e(v,e(w,v6)),w),e(v6,v))),x)).
24 [hyper,1,19,2] P(e(e(e(x,e(e(e(y,e(z,u)),z),e(u,y))),e(e(e(v,e(w,v6)),w),e(v6,v))),x)).
26 (heat=1) [hyper,16,23,17] P(e(e(x,y),e(e(z,e(u,e(e(e(v,u),z),v))),e(e(y,e(w,x)),w)))).
27 (heat=1) [hyper,16,17,24] P(e(e(x,y),e(e(e(y,e(z,x)),z),e(e(e(u,e(v,w)),v),e(w,u))))).
41 [hyper,1,24,23] P(e(e(e(e(x,e(y,z)),y),e(z,x)),e(u,e(v,e(e(e(w,v),u),w))))).
49 [hyper,1,26,18] P(e(e(x,e(y,e(e(e(z,y),x),z))),e(e(e(u,e(e(e(v,u),w),v)),e(v6,w)),v6))).
51 (heat=1) [hyper,16,17,49] P(e(e(e(e(x,e(e(e(y,e(e(e(z,y),u),z)),e(v,u)),v)),w),x),w)).
53 [hyper,1,24,27] P(e(e(e(x,y),e(z,e(e(y,e(u,x)),u))),z)).
54 [hyper,1,23,27] P(e(e(e(x,e(e(e(y,x),z),y)),e(u,z)),u)).
55 [hyper,1,27,18] P(e(e(e(e(x,e(e(e(y,x),z),y)),e(u,z)),u),e(e(e(v,e(w,v6)),w),e(v6,v)))).
56 [hyper,1,27,2] P(e(e(e(e(x,y),e(z,e(e(y,e(u,x)),u))),z),e(e(e(v,e(w,v6)),w),e(v6,v)))).
84 [hyper,1,51,54] P(e(e(e(e(x,y),z),x),e(y,z))).
89 [hyper,1,53,55] P(e(e(x,e(y,e(z,e(x,e(e(u,e(e(e(v,u),w),v)),e(z,w)))))),y)).
90 (heat=1) [hyper,16,17,89] P(e(e(x,e(y,e(e(z,e(e(e(u,z),v),u)),e(x,v))))),y)).
91 [hyper,1,53,56] P(e(e(x,e(y,e(z,e(x,e(e(u,v),e(z,e(e(v,e(w,u)),w))))))),y)).
118 [hyper,1,84,41] P(e(x,e(e(e(e(y,x),z),y),z))).
307 [hyper,1,91,118] P(e(e(e(x,y),e(z,e(y,e(e(u,v),e(z,e(e(v,e(w,u)),w)))))),x)).
308 [hyper,1,90,118] P(e(e(e(x,y),e(e(z,e(e(e(u,z),v),u)),e(y,v))),x)).
322 (heat=1) [hyper,16,17,307] P(e(e(x,e(e(y,z),e(u,e(e(z,e(v,y)),v)))),e(u,x))).
325 (heat=1) [hyper,16,308,17] P(e(e(x,y),e(x,e(z,e(e(e(u,z),y),u))))).
3561 [hyper,1,322,325] P(e(x,e(e(y,e(z,x)),e(z,y)))).
3562 [binary,3561.1,13.1] $ANSWER(XGK).

```
----> UNIT CONFLICT at 205.36 sec ----> 3496 [binary,3495.1,11.1] $ANSWER(PYO).
Length of proof is 10.  Level of proof is 8.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(x,e(e(y,e(z,x)),e(z,y)))).
11 [] -P(e(e(e(a,e(b,c)),c),e(b,a))) | $ANSWER(PYO).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(x,e(e(y,e(z,x)),e(z,y)))).
----------------------------
18 [hyper,1,2,2] P(e(e(x,e(y,e(z,e(e(u,e(v,z)),e(v,u))))),e(y,x))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(e(x,e(y,z)),e(y,x)),e(z,u)),u)).
24 [hyper,1,18,20] P(e(x,e(e(e(y,e(z,u)),e(z,y)),e(u,e(x,e(v,e(e(w,e(v6,v)),e(v6,w)))))))).
27 [hyper,1,20,18] P(e(e(x,e(e(y,e(z,x)),e(z,y))),e(u,u))).
30 (heat=1) [hyper,16,17,27] P(e(e(x,e(y,e(e(z,e(e(u,e(v,z)),e(v,u))),e(w,w)))),e(y,x))).
38 [hyper,1,18,24] P(e(e(e(x,e(y,z)),e(y,x)),e(e(e(u,e(v,w)),e(v,u)),e(w,z)))).
86 [hyper,1,30,2] P(e(e(e(x,x),e(e(y,e(e(z,e(u,y)),e(u,z))),v)),v)).
89 (heat=1) [hyper,16,17,86] P(e(e(x,e(y,e(e(e(z,z),e(e(u,e(e(v,e(w,u)),e(w,v))),v6)),v6))),e(y,x))).
542 [hyper,1,89,2] P(e(e(x,e(e(e(y,y),e(e(z,e(e(u,e(v,z)),e(v,u))),x)),w)),w)).
3495 [hyper,1,542,38] P(e(e(e(x,e(y,z)),z),e(y,x))).
3496 [binary,3495.1,11.1] $ANSWER(PYO).

----> UNIT CONFLICT at 3442.39 sec ----> 79194 [binary,79193.1,10.1] $ANSWER(P9_YRO).
Length of proof is 50.  Level of proof is 23.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(e(x,e(y,z)),z),e(y,x))).
10 [] -P(e(e(a,b),e(c,e(e(c,b),a)))) | $ANSWER(P9_YRO).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(e(x,e(y,z)),z),e(y,x))).
----------------------------
18 [hyper,1,2,2] P(e(x,e(y,e(z,e(x,e(z,y)))))).
19 (heat=1) [hyper,16,18,17] P(e(x,e(y,e(e(e(e(z,e(u,v)),v),e(u,z)),e(y,x))))).
20 [hyper,1,18,18] P(e(x,e(y,e(e(z,e(u,e(v,e(z,e(v,u))))),e(y,x))))).
21 (heat=1) [hyper,16,20,17] P(e(x,e(e(y,e(z,e(u,e(y,e(u,z))))),e(x,e(e(e(v,e(w,v6)),v6),e(w,v)))))).
25 [hyper,1,19,2] P(e(x,e(e(e(e(y,e(z,u)),u),e(z,y)),e(x,e(e(e(v,e(w,v6)),v6),e(w,v)))))).
37 [hyper,1,21,18] P(e(e(x,e(y,e(z,e(x,e(z,y))))),e(e(u,e(v,e(w,e(u,e(w,v))))),e(e(v6,e(v7,v8)),v8),e(v7,v6))))).
39 (heat=1) [hyper,16,17,37] P(e(x,e(e(x,e(e(y,e(z,e(u,e(y,e(u,z))))),v)),v))).
47 [hyper,1,39,18] P(e(e(e(x,e(y,e(z,e(x,e(z,y))))),e(e(u,e(v,e(w,e(u,e(w,v))))),v6)),v6)).
53 (heat=1) [hyper,16,17,47] P(e(e(x,e(y,e(z,e(x,e(z,y))))),e(u,e(v,e(w,e(u,e(w,v))))))).
74 [hyper,1,2,53] P(e(e(x,e(x,y)),y)).
75 (heat=1) [hyper,16,17,74] P(e(x,x)).
103 [hyper,1,74,18] P(e(x,e(y,e(x,y)))).
111 [hyper,1,103,103] P(e(x,e(e(y,e(z,e(y,z))),x))).
119 [hyper,1,103,75] P(e(x,e(e(y,y),x))).
128 [hyper,1,103,18] P(e(x,e(e(y,e(z,e(u,e(y,e(u,z))))),x))).
152 [hyper,1,119,103] P(e(e(x,x),e(y,e(z,e(y,z))))).
153 [hyper,1,119,75] P(e(e(x,x),e(y,y))).
154 [hyper,1,119,74] P(e(e(x,x),e(e(y,e(y,z)),z))).
170 (heat=1) [hyper,16,17,152] P(e(x,e(x,e(y,e(z,e(y,z)))))).
171 (heat=1) [hyper,16,17,153] P(e(x,e(x,e(y,y)))).
172 (heat=1) [hyper,16,17,154] P(e(x,e(x,e(e(y,e(y,z)),z)))).
357 [hyper,1,111,171] P(e(e(x,e(y,e(x,y))),e(z,e(z,e(u,u))))).
363 [hyper,1,111,74] P(e(e(x,e(y,e(x,y))),e(e(z,e(z,u)),u))).
384 (heat=1) [hyper,16,17,357] P(e(e(x,e(y,y)),x)).
389 (heat=1) [hyper,16,17,363] P(e(x,e(y,e(y,x)))).
```

441 [hyper,1,384,25] P(e(e(e(x,y),e(e(e(y,e(x,z)),z),u)),u)).
511 [hyper,1,389,389] P(e(x,e(x,e(y,e(z,e(z,y)))))).
824 [hyper,1,111,170] P(e(e(x,e(y,e(x,y))),e(z,e(z,e(u,e(v,e(u,v))))))).
863 (heat=1) [hyper,16,17,824] P(e(e(x,e(y,e(z,e(y,z)))),x)).
887 [hyper,1,111,172] P(e(e(x,e(y,e(x,y))),e(z,e(z,e(e(u,e(u,v)),v))))).
927 (heat=1) [hyper,16,17,887] P(e(e(x,e(e(y,e(y,z)),z)),x)).
3064 [hyper,1,111,511] P(e(e(x,e(y,e(x,y))),e(z,e(z,e(u,e(v,e(v,u))))))).
3116 (heat=1) [hyper,16,17,3064] P(e(e(x,e(y,e(z,e(z,y)))),x)).
3420 [hyper,1,863,2] P(e(e(e(x,e(y,x)),e(y,z)),z)).
3464 [hyper,1,927,2] P(e(e(x,e(e(y,e(y,x)),z)),z)).
4424 [hyper,1,3116,2] P(e(e(e(x,e(x,y)),e(y,z)),z)).
4641 [hyper,1,111,3464] P(e(e(x,e(y,e(x,y))),e(e(z,e(e(u,e(u,z)),v)),v))).
4649 [hyper,1,3464,18] P(e(x,e(y,e(x,e(z,e(z,y)))))).
4674 (heat=1) [hyper,16,17,4641] P(e(x,e(y,e(e(z,e(z,y)),x)))).
5812 [hyper,1,4424,4649] P(e(e(x,e(x,y)),e(z,e(z,y)))).
5973 [hyper,1,3420,4674] P(e(e(x,e(x,y)),e(z,e(y,z)))).
16888 [hyper,1,441,5973] P(e(x,e(e(e(y,x),e(y,z)),z))).
16893 [hyper,1,441,5812] P(e(e(e(e(x,y),e(x,z)),z),y)).
16926 (heat=1) [hyper,16,17,16893] P(e(x,e(e(y,z),e(y,e(x,z))))).
17658 [hyper,1,128,16888] P(e(e(x,e(y,e(z,e(x,e(z,y))))),e(u,e(e(e(v,u),e(v,w)),w)))).
17874 (heat=1) [hyper,16,17,17658] P(e(x,e(e(y,z),e(y,e(z,x))))).
19456 [hyper,1,441,16926] P(e(e(x,e(y,z)),e(e(y,x),z))).
24676 [hyper,1,441,17874] P(e(e(x,e(y,z)),e(z,e(y,x)))).
36056 [hyper,1,19456,16926] P(e(e(e(x,y),z),e(x,e(z,y)))).
79193 [hyper,1,24676,36056] P(e(e(x,y),e(z,e(e(z,y),x)))).
79194 [binary,79193.1,10.1] $ANSWER(P9_YRO).

----> UNIT CONFLICT at 1469.51 sec ----> 24535 [binary,24534.1,14.1] $ANSWER(XHK).
Length of proof is 19. Level of proof is 10.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,y),e(z,e(e(z,y),x)))).
14 [] -P(e(a,e(e(b,c),e(e(a,c),b)))) | $ANSWER(XHK).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(e(x,y),e(z,e(e(z,y),x)))).
----
18 [hyper,1,2,2] P(e(x,e(e(x,e(y,e(e(y,z),u))),e(u,z)))).
19 (heat=1) [hyper,16,17,18] P(e(x,e(e(x,e(e(y,e(z,e(e(z,u),v))),e(v,u))),y))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(e(x,y),e(z,e(e(z,y),x))),e(u,e(e(u,v),w))),e(w,v))).
23 [hyper,1,19,2] P(e(e(e(e(x,y),e(z,e(e(z,y),x))),e(e(u,e(v,e(e(v,w),v6))),e(v6,w))),u)).
25 [hyper,1,20,20] P(e(e(e(e(e(x,y),z),y),x),z)).
27 (heat=1) [hyper,16,17,25] P(e(x,e(e(x,y),e(e(e(e(z,u),y),u),z)))).
29 [hyper,1,18,25] P(e(e(e(e(e(e(e(x,y),z),y),x),z),e(u,e(e(u,v),w))),e(w,v))).
30 (heat=1) [hyper,16,17,29] P(e(x,e(e(x,e(y,z)),e(e(e(e(e(e(u,v),w),v),u),w),e(v6,e(e(v6,z),y)))))).
33 [hyper,1,27,27] P(e(e(e(x,e(e(x,y),e(e(e(e(z,u),y),u),z))),v),e(e(e(e(w,v6),v),v6),w))).
47 [hyper,1,23,29] P(e(e(e(e(x,e(e(x,y),z)),u),e(z,y)),u)).
51 [hyper,1,29,30] P(e(e(x,e(e(x,y),e(z,y))),z)).
52 (heat=1) [hyper,16,17,51] P(e(x,e(e(x,y),e(z,e(e(z,u),e(y,u)))))).
57 [hyper,1,33,29] P(e(e(e(e(x,y),e(e(e(e(e(z,u),v),u),z),v)),y),x)).
155 [hyper,1,52,2] P(e(e(e(e(x,y),e(z,e(e(z,y),x))),u),e(v,e(e(v,w),e(u,w))))).
195 [hyper,1,57,51] P(e(e(x,y),e(y,x))).
204 (heat=1) [hyper,16,17,195] P(e(x,e(e(x,e(y,z)),e(z,y)))).
792 [hyper,1,51,155] P(e(e(e(e(e(x,y),e(z,e(e(z,y),x))),u),e(u,v)),v)).
1424 [hyper,1,204,47] P(e(e(e(e(e(e(x,e(e(x,y),z)),u),e(z,y)),u),e(v,w)),e(w,v))).
24534 [hyper,1,1424,792] P(e(x,e(e(y,z),e(e(x,z),y)))).

24535 [binary,24534.1,14.1] $ANSWER(XHK).

----> UNIT CONFLICT at 78273.10 sec ----> 98394 [binary,98393.1,8.1] $ANSWER(P7_WN).
Length of proof is 39. Level of proof is 25.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(x,e(e(y,z),e(e(x,z),y)))).
8 [] -P(e(e(a,e(b,c)),e(c,e(a,b)))) | $ANSWER(P7_WN).
16 [] -P(e(x,y)) | -P(x) | P(y).
17 [] P(e(x,e(e(y,z),e(e(x,z),y)))).
----
18 [hyper,1,2,2] P(e(e(x,y),e(e(e(z,e(e(u,v),e(e(z,v),u))),y),x))).
19 (heat=1) [hyper,16,17,18] P(e(e(x,y),e(e(e(e(z,u),e(e(v,e(e(w,v6),e(e(v,v6),w))),u),z)),y),x))).
20 (heat=1) [hyper,16,18,17] P(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(u,v),e(e(w,v),u))),w)).
21 [hyper,1,18,18] P(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(e(u,e(e(v,w),e(e(u,w),v))),v6),v7)),e(v7,v6))).
27 [hyper,1,19,18] P(e(e(e(e(x,y),e(e(e(z,e(e(u,v),e(e(z,v),u))),y),x)),e(e(e(w,e(e(v6,v7),
e(e(w,v7),v6))),v8),v9)),e(v9,v8))).
35 (heat=1) [hyper,16,27,17] P(e(e(e(e(e(x,y),e(e(e(z,e(e(u,v),e(e(z,v),u))),y),x)),w),
e(v6,e(e(v7,v8),e(e(v6,v8),v7)))),w)).
47 [hyper,1,21,20] P(e(e(e(x,y),e(e(e(z,e(e(u,v),e(e(z,v),u))),w),y),x)),w)).
56 [hyper,1,35,47] P(e(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(u,e(e(v,w),e(e(u,w),v)))),
e(e(e(v6,e(e(v7,v8),e(e(v6,v8),v7))),v9),v10)),e(v10,v9))).
67 [hyper,1,56,21] P(e(x,e(e(e(e(y,e(e(z,u),e(e(y,u),z))),x),e(e(v,w),e(e(v6,w),v))),v6))).
74 [hyper,1,67,47] P(e(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(e(u,v),e(e(e(e(w,e(e(v6,v7),
e(e(w,v7),v6))),v8),v),u)),v8)),e(e(v9,v10),e(e(v11,v10),v9))),v11)).
81 [hyper,1,20,74] P(e(e(e(x,y),e(e(z,e(e(u,v),e(e(z,v),u))),x)),y)).
93 [hyper,1,67,81] P(e(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(e(u,v),e(e(w,e(e(v6,v7),
e(e(w,v7),v6))),u)),v)),e(e(v8,v9),e(e(v10,v9),v8))),v10)).
97 [hyper,1,81,47] P(e(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(u,e(e(v,w),e(e(u,w),v))),e(v6,v7))),v7),v6)).
107 [hyper,1,20,93] P(e(e(e(x,e(e(y,z),e(e(u,z),y))),u),x)).
117 [hyper,1,107,107] P(e(x,e(e(y,z),e(e(e(e(u,v),e(e(x,v),u)),z),y)))).
136 [hyper,1,18,117] P(e(e(e(x,e(e(y,z),e(e(x,z),y))),e(e(u,v),e(e(e(e(w,v6),e(e(v7,v6),w)),v),u))),v7)).
156 [hyper,1,97,136] P(e(e(e(x,y),e(e(z,y),x)),e(e(u,v),e(e(z,v),u)))).
175 [hyper,1,107,156] P(e(e(e(e(e(x,y),e(e(z,y),x)),u),z),u)).
176 [hyper,1,81,156] P(e(e(x,y),e(x,e(e(z,u),e(e(y,u),z))))).
188 [hyper,1,107,175] P(e(e(e(x,y),e(e(e(e(z,u),e(e(v,u),z)),y),x)),v)).
279 [hyper,1,176,188] P(e(e(e(x,y),e(e(e(e(z,u),e(e(v,u),z)),y),x)),e(e(w,v6),e(e(v,v6),w)))).
280 [hyper,1,175,188] P(e(e(e(e(x,y),e(e(z,y),x)),e(e(z,u),v)),e(v,u))).
295 [hyper,1,280,280] P(e(e(e(x,y),x),y)).
304 [hyper,1,280,176] P(e(e(e(x,y),e(e(e(e(z,u),z),y),x)),u)).
328 [hyper,1,295,280] P(e(e(x,e(e(x,y),z)),e(z,y))).
380 [hyper,1,107,304] P(e(e(e(x,y),e(e(z,x),z)),y)).
385 [hyper,1,81,328] P(e(e(x,x),e(y,e(e(z,u),e(e(y,u),z))))).
388 [hyper,1,176,380] P(e(e(e(x,y),e(e(z,x),z)),e(e(u,v),e(e(y,v),u)))).
389 [hyper,1,380,328] P(e(e(x,y),e(y,x))).
396 [hyper,1,328,385] P(e(e(e(x,y),e(e(e(z,z),u),y),x)),u)).
402 [hyper,1,389,388] P(e(e(e(x,y),e(e(z,y),x)),e(e(u,z),e(e(v,u),v)))).
426 [hyper,1,107,396] P(e(e(e(x,y),e(e(z,z),x)),y)).
433 [hyper,1,295,402] P(e(e(x,e(e(y,z),y)),e(z,x))).
500 [hyper,1,389,433] P(e(e(x,y),e(y,e(e(z,x),z)))).
516 [hyper,1,433,280] P(e(x,e(e(e(y,z),e(e(u,z),y)),e(e(u,v),e(v,x))))).
686 [hyper,1,500,500] P(e(e(x,e(e(y,z),y)),e(e(u,e(z,x)),u))).
5430 [hyper,1,426,279] P(e(e(e(e(x,y),e(e(z,y),x)),u),e(z,u))).
12201 [hyper,1,686,516] P(e(e(x,e(e(e(y,e(z,u)),e(y,z)),u)),x)).
98393 [hyper,1,5430,12201] P(e(e(x,e(y,z)),e(z,e(x,y)))).

98394 [binary,98393.1,8.1] $ANSWER(P7_WN).

----> UNIT CONFLICT at 14999.29 sec ----> 176939 [binary,176938.1,5.1] $ANSWER(P3_YQJ).
Length of proof is 28. Level of proof is 14.
---------------- PROOF ----------------
1 [] -P(e(x,y)) | -P(x) | P(y).
2 [] P(e(e(x,e(y,z)),e(z,e(x,y)))).
5 [] -P(e(e(a,b),e(e(c,a),e(b,c)))) | $ANSWER(P3_YQJ).
27 [] -P(e(x,y)) | -P(x) | P(y).
28 [] P(e(e(x,e(y,z)),e(z,e(x,y)))).
----
29 [hyper,1,2,2] P(e(e(x,y),e(e(x,e(y,z)),z))).
30 (heat=1) [hyper,27,28,29] P(e(x,e(e(y,z),e(y,e(z,x))))).
32 [hyper,1,29,29] P(e(e(e(x,y),e(e(e(x,e(y,z)),z),u)),u)).
35 [hyper,1,29,30] P(e(e(x,e(e(e(y,z),e(y,e(z,x))),u)),u)).
36 [hyper,1,30,29] P(e(e(x,y),e(x,e(y,e(e(z,u),e(e(z,e(u,v)),v)))))).
41 (heat=1) [hyper,27,28,36] P(e(e(x,e(e(y,z),e(e(y,e(z,u)),u))),e(e(v,x),v))).
80 [hyper,1,35,29] P(e(x,e(e(y,x),y))).
87 [hyper,1,80,80] P(e(e(x,e(y,e(e(z,y),z))),x)).
201 [hyper,1,29,87] P(e(e(e(x,e(y,e(e(z,y),z))),e(x,u)),u)).
216 (heat=1) [hyper,27,201,28] P(e(e(e(x,y),x),y)).
238 [hyper,1,29,216] P(e(e(e(e(x,y),x),e(y,z)),z)).
241 [hyper,1,216,32] P(e(e(e(x,e(y,z)),z),e(x,y))).
255 (heat=1) [hyper,27,28,241] P(e(x,e(e(e(y,e(x,z)),z),y))).
1764 [hyper,1,29,41] P(e(e(e(x,e(e(y,z),e(e(y,e(z,u)),u))),e(e(e(v,x),v),w)),w)).
1799 (heat=1) [hyper,27,1764,28] P(e(e(x,y),e(y,x))).
1842 [hyper,1,29,1799] P(e(e(e(x,y),e(e(y,x),z)),z)).
1843 [hyper,1,2,1799] P(e(x,e(e(x,y),y))).
1923 [hyper,1,1843,1799] P(e(e(e(e(x,y),e(y,x)),z),z)).
2068 [hyper,1,1799,238] P(e(x,e(e(e(y,z),y),e(z,x)))).
2096 (heat=1) [hyper,27,28,2068] P(e(e(x,y),e(y,e(e(z,x),z)))).
2935 [hyper,1,1799,255] P(e(e(e(e(x,e(y,z)),z),x),y)).
17269 [hyper,1,1799,1842] P(e(x,e(e(y,z),e(e(z,y),x)))).
17345 (heat=1) [hyper,27,28,17269] P(e(e(e(x,y),z),e(z,e(y,x)))).
25963 [hyper,1,1923,2096] P(e(e(x,y),e(e(z,e(y,x)),z))).
28685 [hyper,1,241,2935] P(e(e(e(e(x,y),e(y,z)),z),x)).
88894 [hyper,1,25963,28685] P(e(e(x,e(y,e(e(e(y,z),e(z,u)),u))),x)).
89072 (heat=1) [hyper,27,88894,28] P(e(e(e(x,y),e(y,z)),e(z,x))).
176938 [hyper,1,17345,89072] P(e(e(x,y),e(e(z,x),e(y,z)))).
176939 [binary,176938.1,5.1] $ANSWER(P3_YQJ).

References

[Kalman78] Kalman, J., ‘‘A shortest single axiom for the classical equivalential calculus’’, *Notre Dame J. Formal Logic* **19,** 141-144 (1978).

[Kalman83] Kalman, J., ‘‘Condensed Detachment as a Rule of Inference’’, *Studia Logica* **42,** 443-451 (1983).

[McCharen76] McCharen, J., Overbeek, R., and Wos, L., ‘‘Complexity and related enhancements for automated theorem-proving programs’’, *Computers and Mathematics with Applications* **2,** 1-16 (1976).

[McCune92] McCune, W., and Wos, L., ‘‘Experiments in automated deduction with condensed detachment’’, in *Proceedings of the Eleventh International Conference on Automated Deduction* (CADE-11), *Lecture Notes in Artificial Intelligence*, Vol. 607, ed. D. Kapur, Springer-Verlag, New York, 1992, pp. 209-223.

[McCune93] McCune, W., ''Otter 3.0'', Preprint MCS-P399-1193, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, November 1993.

[Peterson76] Peterson, J., ''Shortest single axioms for the equivalential calculus'', *Notre Dame J. Formal Logic* **17,** 267-271 (1976).

[Wos83] Wos, L., Winker, S., Veroff, R., Smith, B., and Henschen, L., ''Questions concerning possible shortest single axioms in equivalential calculus: An application of automated theorem proving to infinite domains'', *Notre Dame J. Formal Logic* **24,** 205-223 (1983).

[Wos84] Wos, L., Winker, S., Veroff, R., Smith, B., and Henschen, L., ''A new use of an automated reasoning assistant: Open questions in equivalential calculus and the study of infinite domains'', *Artificial Intelligence* **22,** 303-356 (1984).

[Wos87] Wos, L., *Automated Reasoning: 33 Basic Research Problems,* Prentice-Hall: Englewood Cliffs, N.J., 1987.

[Wos90] Wos, L., ''Meeting the challenge of fifty years of logic'', *J. Automated Reasoning* **6**, no. 2, 213-222 (1990).

[Wos92] Wos, L., Overbeek, R., Lusk, E., and Boyle, J., *Automated Reasoning: Introduction and Applications*, 2nd ed., McGraw-Hill: New York, 1992.

[Wos94] Wos, L., Personal information, Argonne National Laboratory (1994).

[Wos95] Wos, L., *Automating the Search for Elegant Proofs: An Experimenter's Notebook*, submitted for publication (1994).