On Sets, Types, Fixed Points, and Checkerboards

Peter B. Andrews and Matthew Bishop
Mathematics Department, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.
Andrews+@CS.CMU.EDU, mbishop+@CS.CMU.EDU[1]

Abstract

Most current research on automated theorem proving is concerned with proving theorems of first-order logic. We discuss two examples which illustrate the advantages of using higher-order logic in certain contexts. For some purposes type theory is a much more convenient vehicle for formalizing mathematics than axiomatic set theory. Even theorems of first-order logic can sometimes be proven more expeditiously in higher-order logic than in first-order logic. We also note the need to develop automatic theorem-proving methods which may produce proofs which do not have the subformula property.

1. Introduction

In some ways it appears that the field of automated deduction is ahead of its time. We have increasingly good methods for proving theorems, but the hardware available is still not adequate for many of the problems to which we would like to apply our techniques. However, this will change. Radically new computers based on exotic technologies such as DNA computing, quantum computing, and optical circuitry are on the horizon [7] [11].

As better hardware is combined with increasingly versatile, sophisticated, and friendly tools for using rigorous logic in practical ways, there will be a revolution in the foundations and developmental methodologies of many intellectual disciplines. Imagine what could happen in many fields if it were really easy to explore the logical consequences of rigorously formulated theories. We can look forward to a great flowering of logical literacy. The day will come when every scientist in every discipline should have at least some education in logic.

As we consider the variety of intellectual disciplines which may profitably use formal logic under such circumstances, it is natural to ask what logical languages will be used in the formalization of these disciplines. First-order logic is basic, because it's the logic of connectives and quantifiers, and these occur in most formal languages. However, as a practical matter, it may not be rich enough for certain purposes. For example, in certain contexts, type theory seems to have clear practical advantages over first-order logic. We shall illustrate this point with two examples.[2]

Proofs will be presented in the style of [1], and in wffs a dot stands for a left

---

[1]This material is based upon work supported by the National Science Foundation under grant CCR-9201893.

[2]Much of the work discussed below was done using the theorem proving system TPS [2], which has been developed in collaboration with Sunil Issar, Dan Nesmith, Frank Pfenning, and Hongwei Xi.

bracket whose mate is as far to the right as is consistent with the pairing of brackets already present. Quantifiers and negations have the smallest possible scope. Omitted brackets associated with the connectives $\wedge$, $\vee$, $\supset$, and $\equiv$ should be restored in that order, giving the connective the smallest possible scope in each case.

2. The Fixed Point Theorem

The best developed logical systems for formalizing mathematics are (1) various formulations of axiomatic set theory in first-order logic and (2) type theory (otherwise known as higher-order logic). The formulation of set theory in [4] seems particularly well suited to automated deduction, since it is finitely axiomatised, and has been enhanced with function constants to denote such operations as forming ordered pairs, intersections of sets, etc.

First-order logic is simpler than type theory in various ways, but one may pay a considerable price for this apparent simplicity. Let us illustrate this by considering the problem of proving THM15B of [2] in Church's type theory [6] and in set theory as formulated in [4]. THM15B asserts that if some iterate of a function f has a unique fixed point, then f has a fixed point. A type-theoretic proof of this theorem in natural deduction style which was found automatically in less than six minutes is presented in [2]. The proof is 35 lines long, but can easily be made a few lines shorter by (interactively) reorganizing lines 24-28.

An interactively generated outline of a proof of this theorem in the set theory of [4] is shown in Figures 2-1 and 2-2. Each line of the proof contains a number in parentheses which serves as a label for that line, a list of numbers of lines which are assumed as hypotheses for that line, the assertion sign $\vdash$, the wff asserted in that line, and a justification, which indicates by what rule and from what lines that line was inferred. The proof contains 47 lines, but we have numbered them to reflect our estimate of what the proof would look like if all the details were filled in. The notation is closely related to that used in [4]. Mx means that x is a set, [NOP x y] (written as {x,y} in [4]) is the nonordered pair formed from x and y, [OP x y] (written as <x,y> in [4]) is the ordered pair formed from x and y, and [APPLY f y] is the result of applying f to y. [ITERATE f g] is an abbreviation for the wff in line (7) of the proof.

Let us summarize the main ideas of the proof, with references to the lines where they occur. Initially, we assume that (6) g is an iterate of f and that (5) x is the unique fixed point of g. By using the definition of iterate we prove that (8529) $f \circ g = g \circ f$, and hence we conclude that (9036) g [f x] = f [g x]. Since x is a fixed point of g, (9037) g x = x, so (9038) g [f x] = f x. This shows that [f x] is also a fixed point of g, so by the uniqueness of x, (9041) f x = x.

The proof contains three gaps. Line (9030) is Lemma 26 of [4] and line (8518) is a lemma about the associativity of composition which must be proved. In type theory this is trivial, since functions are primitive, and $g_{\alpha\beta} \circ f_{\beta\gamma}$ is defined to be $\lambda x_{\gamma}\, g_{\alpha\beta}\, [f_{\beta\gamma}\, x]$. In set theory, functions are defined in terms of sets, and composition is defined by the more complex axiom

$z \,\varepsilon\, [g \circ f] \equiv M\, z \wedge \exists x\, \exists y\, \exists w . M\, x \wedge M\, y \wedge M\, w \wedge z = OP\, x\, y \wedge OP\, x\, w\, \varepsilon\, f \wedge OP\, w\, y\, \varepsilon\, g$

Actually, some additional hypotheses such as [FUNCTION f] may be needed to prove (8518), but this will not impede its applicability in this proof. We estimate that formal

Figure 2-1: Proof of THM15B - part 1

```
(1)   1 ⊦ M f ∧ FUNCTION f ∧ ∃g.ITERATE f g ∧ M g ∧
  FUNCTION g ∧ ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
  ∀z.APPLY g z = z ⊃ z = x                              Hyp
(2)   1 ⊦ ∃g.ITERATE f g ∧ M g ∧ FUNCTION g ∧
   ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
   ∀z.APPLY g z = z ⊃ z = x                        RuleP: 1
(3)   1,3 ⊦ ITERATE f g ∧ M g ∧ FUNCTION g ∧
   ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
   ∀z.APPLY g z = z ⊃ z = x                     Choose: g 2
(4)   1,3 ⊦ ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
  ∀z.APPLY g z = z ⊃ z = x                         RuleP: 3
(5)   1,3,5 ⊦ x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
  ∀z.APPLY g z = z ⊃ z = x                      Choose: x 4
(6)   1,3 ⊦ ITERATE f g                            RuleP: 3
(7)   1,3 ⊦ ∀p.f ε p ∧ ∀j [j ε p ∧ M [j ∘ f] ⊃ j ∘ f ε p]
  ⊃ g ε p                                            Def: 6
              ...
(8008) ⊦ ∃P ∀g.M g ⊃.g ε P ≡ f ∘ g = g ∘ f          Lemma
(8009)8009 ⊦ ∀g.M g ⊃.g ε P ≡ f ∘ g = g ∘ f
                                              Choose: P 8008
(8010) ⊦ f ∘ f = f ∘ f                         Assert REFL=
(8011)8009 ⊦ M f ⊃.f ε P ≡ f ∘ f = f ∘ f       UI: f 8009
(8012)1,8009 ⊦ f ε P                   RuleP: 1 8010 8011
(8013)8013 ⊦ j ε P ∧ M.j ∘ f                            Hyp
(8014)8009 ⊦ M j ⊃.j ε P ≡ f ∘ j = j ∘ f       UI: j 8009
(8015) ⊦ ∀x ∀y.x ε y ⊃ M x                Axiom A-2 of [4]
(8016) ⊦ j ε P ⊃ M j                             UI : 8015
(8017)8009,8013 ⊦ f ∘ j = j ∘ f     RuleP: 8013 8014 8016
              ...
(8518) ⊦ ∀f ∀g ∀h.f ∘ [g ∘ h] = [f ∘ g] ∘ h          Lemma
(8519) ⊦ f ∘ [j ∘ f] = [f ∘ j] ∘ f                UI: 8518
(8520)8009,8013 ⊦ f ∘ [j ∘ f] = [j ∘ f] ∘ f
                                          Sub=: 8519 8017
(8521)8009 ⊦ M [j ∘ f] ⊃.j ∘ f ε P ≡
   f ∘ [j ∘ f] = [j ∘ f] ∘ f             UI: [j ∘ f] 8009
(8522)8009,8013 ⊦ j ∘ f ε P         RuleP: 8013 8520 8521
(8523)8009 ⊦ j ε P ∧ M [j ∘ f] ⊃ j ∘ f ε P   Deduct: 8522
(8524)8009 ⊦ ∀j.j ε P ∧ M [j ∘ f] ⊃ j ∘ f ε P
                                             UGen: j 8523
(8525)1,8009 ⊦ f ε P ∧ ∀j.j ε P ∧ M [j ∘ f] ⊃ j ∘ f ε P
                                          Conj: 8012 8524
(8526)3,1 ⊦ f ε P ∧ ∀j [j ε P ∧ M [j ∘ f] ⊃ j ∘ f ε P]
   ⊃ g ε P                                         UI: P 7
(8527)1,3,8009 ⊦ g ε P                       MP: 8525 8526
```

Figure 2-2: Proof of THM15B - part 2

```
(8528)8009 ⊦ M g ⊃.g ε P ≡ f ∘ g = g ∘ f        UI: g 8009
(8529)1,3,8009 ⊦ f ∘ g = g ∘ f          RuleP: 3 8527 8528
               ...
(9030) ⊦ ∀f ∀g ∀x.FUNCTION f ∧ x ε DOM f ⊃
   APPLY g [APPLY f x] = APPLY [g ∘ f] x   Lemma 26 of [4]
(9031) ⊦ FUNCTION f ∧ x ε DOM f ⊃
   APPLY g [APPLY f x] = APPLY [g ∘ f] x          UI: 9030
(9032)1,3,5 ⊦ APPLY g [APPLY f x] = APPLY [g ∘ f] x
                                          RuleP: 1 5 9031
(9033) ⊦ FUNCTION g ∧ x ε DOM g ⊃ APPLY f [APPLY g x] =
  APPLY [f ∘ g] x                                 UI: 9030
(9034)1,3,5 ⊦ APPLY f [APPLY g x] = APPLY [f ∘ g] x
                                          RuleP: 3 5 9033
(9035)8009,5,3,1 ⊦ APPLY g [APPLY f x] = APPLY [f ∘ g] x
                                          Sub=: 9032 8529
(9036)8009,5,3,1 ⊦ APPLY g [APPLY f x] =
   APPLY f.APPLY g x                    Subst=: 9035 9034
(9037)1,3,5 ⊦ APPLY g x = x                       RuleP: 5
(9038)1,3,5,8009 ⊦ APPLY g [APPLY f x] = APPLY f x
                                        Subst=: 9036 9037
(9039)1,3,5 ⊦ ∀z.APPLY g z = z ⊃ z = x           RuleP: 5
(9040)5,3,1 ⊦ APPLY g [APPLY f x] = APPLY f x ⊃
   APPLY f x = x                     UI: [APPLY f x] 9039
(9041)8009,1,3,5 ⊦ APPLY f x = x           MP: 9038 9040
(9042)1,3,5,8009 ⊦ ∃y.APPLY f y = y         EGen: x 9041
(9043)1,3,5 ⊦ ∃y.APPLY f y = y         RuleC: 8008 9042
(9044)1,3 ⊦ ∃y.APPLY f y = y              RuleC: 4 9043
(9045)1 ⊦ ∃y.APPLY f y = y                RuleC: 2 9044
(9046) ⊦ M f ∧ FUNCTION f ∧ ∃g [ITERATE f g ∧ M g ∧
  FUNCTION g ∧ ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
  ∀z.APPLY g z = z ⊃ z = x] ⊃ ∃y.APPLY f y = y
                                            Deduct: 9045
(9047) ⊦ ∀f.M f ∧ FUNCTION f ∧ ∃g [ITERATE f g ∧ M g ∧
  FUNCTION g ∧ ∃x.x ε DOM f ∧ x ε DOM g ∧.APPLY g x = x ∧
  ∀z.APPLY g z = z ⊃ z = x] ⊃ ∃y.APPLY f y = y
                                             UGen: f 904
```

---

proofs of these lemmas will each take about 500 lines.

The main gap is at line (8008), which is needed in order to instantiate the quantifier in line (7) with a term which provides the framework (in line (8526)) for an inductive proof of the crucial line (8529). The complexities involved in proving line (8008) are directly related to the fundamental differences between type theory and set theory. In type theory one can simply write notations for sets and functions, and it is trivial to prove their existence, so no axioms for the existence of sets are needed (except for axioms of infinity and choice, where appropriate). In set theory such a procedure would lead to contradictions, so there are carefully chosen axioms which

must be used to establish the existence of sets and functions. In order to prove a theorem of the form
$\exists P\ \forall x_1 \ldots \forall x_n.\ M\ x_1 \wedge \ldots \wedge M\ x_n \supset [<x_1 \ldots x_n> \varepsilon\ P \equiv \mathbf{A}]$,
one proceeds by structural induction on **A**, as in Gödel's proof (page 8 of [9]) of the General Existence Theorem. Let us illustrate this procedure.

In order to prove
(8008) $\exists P\ \forall g.M\ g \supset .g\ \varepsilon\ P \equiv [f \circ g = g \circ f]$
we use extensionality to eliminate = and reduce the problem to:
$\exists P\ \forall g.M\ g \supset .g\ \varepsilon\ P \equiv \forall u.M\ u \supset .u\ \varepsilon\ [f \circ g] \equiv u\ \varepsilon .g \circ f$
We then replace $\forall u$ by $\sim\exists u \sim$ and seek a proof of:
$\exists P\ \forall g.M\ g \supset .g\ \varepsilon\ P \equiv \sim\exists u.M\ u \wedge \sim .u\ \varepsilon\ [f \circ g] \equiv u\ \varepsilon .g \circ f$
Using Axiom B-3 of complements, in order to prove this it suffices to prove:
$\exists P^1\ \forall g.M\ g \supset .g\ \varepsilon\ P^1 \equiv \exists u.M\ u \wedge \sim .u\ \varepsilon\ [f \circ g] \equiv u\ \varepsilon .g \circ f$
With the aid of the axiom for range, which involves an existential quantifier, this can be proved if we can prove:
$\exists P^2\ \forall u\ \forall g.M\ g \supset .M\ u \supset .OP\ u\ g\ \varepsilon\ P^2 \equiv \sim .u\ \varepsilon\ [f \circ g] \equiv u\ \varepsilon .g \circ f$
Again we eliminate a negation to reduce the problem to:
$\exists P^3\ \forall u\ \forall g.M\ g \supset .M\ u \supset .OP\ u\ g\ \varepsilon\ P^3 \equiv .u\ \varepsilon\ [f \circ g] \equiv u\ \varepsilon .g \circ f$
Rewriting $\equiv$, we must now prove:
$\exists P^3\ \forall u\ \forall g.M\ g \supset .M\ u \supset .OP\ u\ g\ \varepsilon\ P^3 \equiv \sim[u\ \varepsilon\ [f \circ g] \wedge \sim u\ \varepsilon .g \circ f] \wedge \sim .u\ \varepsilon\ [g \circ f] \wedge \sim u\ \varepsilon .f \circ g$
This can be proved with the aid of Axiom B-2 for intersection if we can prove the two subgoals which correspond to the scopes of the conjunction. One of these is:
$\exists P^4\ \forall u\ \forall g.M\ g \supset .M\ u \supset .OP\ u\ g\ \varepsilon\ P^4 \equiv \sim .u\ \varepsilon\ [f \circ g] \wedge \sim u\ \varepsilon .g \circ f$

Following this branch of the proof tree, we eventually must prove:
$\exists P^5\ \forall u\ \forall g.M\ g \supset .M\ u \supset .OP\ u\ g\ \varepsilon\ P^5 \equiv u\ \varepsilon .f \circ g$
so we must deal with $\circ$, the definition of which was given above. We will work our way through a collection of lemmas about OP, which is defined in terms of NOP:
OP x y = [NOP [SINGLETON x].NOP x y], where [SINGLETON x] is [NOP x x], and so we eventually reach a lemma where the formula on the right is of the form [z ε NOP x y]. One such lemma is:
$\exists P^6\ \forall z\ \forall v\ \forall x\ \forall y\ \forall w\ \forall u\ \forall g\ .\ M\ z \wedge M\ v \wedge M\ x \wedge M\ w \wedge M\ y \wedge M\ g \wedge M\ u \supset .OP\ z\ [OP\ v\ .OP\ x\ .OP\ y\ .OP\ w\ .OP\ u\ g]\ \varepsilon\ P^6 \equiv z\ \varepsilon\ NOP\ x\ y$

This requires a substantial proof; first Axiom A-4 for nonordered pairs is used to eliminate the NOP, which will leave [z=x ∨ z=y]; then Axioms B-3 and B-2 (complement and intersection) are used to reduce this to two lemmas, one with z=x and the other with z=y. These are handled similarly, by extensionality to get (in the first case) $\forall t\ [\ M\ t \supset .t\ \varepsilon\ z \equiv t\ \varepsilon\ x]$. Then the quantifier $\forall t$ is removed by using the axioms for complement and range, which adds another quantifier to the start of the lemma and another component to the ordered n-tuple. Then the equivalence is broken down with the axioms for complement and intersection to get one case with [t ε z] and the other with [t ε x].

A typical example of such a lemma is:
$\exists P^7\ \forall z\ \forall v\ \forall x\ \forall y\ \forall w\ \forall u\ \forall g\ .\ M\ z \wedge M\ v \wedge M\ x \wedge M\ w \wedge M\ y \wedge M\ g \wedge M\ u \supset .OP\ z\ [OP\ v\ .OP\ x\ .OP\ y\ .OP\ w\ .OP\ u\ g]\ \varepsilon\ P^7 \equiv z\ \varepsilon\ v$
In order to prove this, we use Axiom B-7 (rotate-right) to rotate the variables in the

ordered n-tuple on the left (in general, we will also need the axioms for flip-range and converse) until the two variables which appear on the right have been isolated. This rearrangement of the variables near the end of each branch of the proof tree constitutes a very substantial part of the proof. In this case, this leaves the following formula to be proven:

$\exists P^8\ \forall z\ \forall v\ \forall x\ \forall y\ \forall w\ \forall u\ \forall g\ .\ M\ z \wedge M\ v \wedge M\ x \wedge M\ w \wedge M\ y \wedge M\ g \wedge M\ u \supset .OP\ [OP\ x\ .OP\ y\ .OP\ w\ .OP\ u\ g]\ [OP\ z\ v]\ \varepsilon\ P^8 \equiv z\ \varepsilon\ v$

Finally, we can assert that the required set exists; it is

$V \times [V \times .V \times .V \times V] \times ESTIN$, where ESTIN is the membership relation from Gödel's axiom B1:

$z\ \varepsilon\ ESTIN \equiv M\ z \wedge OPP\ z \wedge FST\ z\ \varepsilon\ SND\ z$,

and V is the universal set. Thus, we prove the wff above from the theorem

$\forall z\ \forall v\ \forall x\ \forall y\ \forall w\ \forall u\ \forall g\ .\ M\ z \wedge M\ v \wedge M\ x \wedge M\ w \wedge M\ y \wedge M\ g \wedge M\ u \supset .\ OP\ [OP\ x\ .OP\ y\ .OP\ w\ .OP\ u\ g]\ [OP\ z\ v]\ \varepsilon\ [V \times [V \times .V \times .V \times V] \times ESTIN] \equiv z\ \varepsilon\ v$

Substantial work was done on line (8008) to see what is involved in proving it, but the proof was not completed. Work was terminated when 3410 lines of the proof had been generated, and it seemed that little more would be learned by continuing. We estimate that the complete proof of line (8008) would occupy roughly 8000 lines. In the part of the proof which was completed, the axioms were used 222 times. It seems clear that any procedure which searched blindly for a proof of THM15B at a sufficient depth to generate this proof (in any format) would be searching in an extremely large search space.

The authors of [4] presented their system of set theory as a vehicle for generating difficult problems for resolution-based theorem provers. In this they were surely successful. However, there are grounds for substantial doubts about whether this system is as well suited as type theory for serious exploration of mathematical problems involving sets and functions.

Of course, one may be inspired by examples such as this to design formulations of set theory which are increasingly expressive. One may thus be led to reinvent type theory.

There is a sense in which sets and functions are essentially higher-order concepts, and this example illustrates the price one may pay for forcing them into a first-order context. Next we turn to an example where higher-order concepts are useful in proving a genuinely first-order theorem.

3. McCarthy's Tough Nut

In 1964 John McCarthy wrote the memo [12], which is about a mutilated checkerboard, such as is shown in Figure 3-1, and by now various versions of the problem are well known. The problem is to prove that the mutilated checkerboard cannot be tiled by dominoes, each of which covers exactly two adjacent squares. The problem can be described formally in a variety of ways. McCarthy's memo gives two closely related formulations of the problem as sets of sentences to refute in first-order logic. Of course, a naive brute-force treatment of the problem will essentially consist of enumerating all possible tilings, and showing that none of them work. Such an approach will lead to proofs which are very lengthy, and get even longer if one increases the size of the checkerboard. On the other hand, proofs which use more sophisticated conceptual approaches tend to be more or less independent of the size of the checkerboard. In order to define the concepts involved in more sophisticated approaches, higher-order logic is often useful, and sometimes essential.

Some treatments of the problem, and references to additional treatments, may be

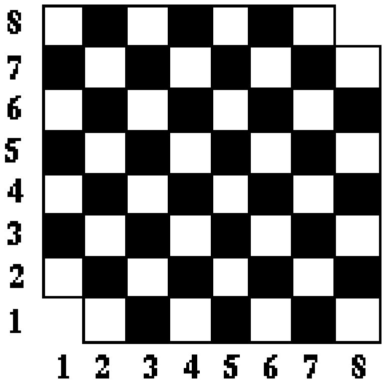

Figure 3-1: The mutilated checkerboard

found in [3], [13], [14], [17], [18], and [19] (section 5). It should be noted that most treatments of the problem in the literature solve it in the conceptual sense, but do not actually provide proofs of the theorem in either of McCarthy's original formulations. One exception to this is discussed in [15]. McCune reports that when he applied a program called MACE, which searches for finite models, to McCarthy's first formulation of the problem, MACE determined in a minute or two that there are no models of size 8. McCune notes that if there are any models, then there is a model of size 8, though MACE does not prove this. McCune has also remarked that the procedures used by MACE involve exhaustive search and are highly sensitive to the size of the checkerboard.

Let us use McCarthy's second formulation of the problem as a vehicle for illustrating how type theory can be applied to a problem of this sort, and discussing certain related issues. We start by considering the problem from a conceptual point of view. One can define certain squares of the checkerboard to be black and others to be white, and base the proof on the fact that each domino will cover one square of each color, but there are more white squares than black squares in the mutilated checkerboard. This is the approach taken by Subramanian [17], who uses certain built-in arithmetic functions and relations. A proof along these lines in pure higher-order logic would require defining these arithmetic concepts. One can certainly do this, as is shown in chapter 6 of [1], but simply borrowing the required theorems from [1] would require introducing an Axiom of Infinity, which does not seem warranted for this problem.

Note that while the theorem is true regardless of the size of the checkerboard, the finiteness of the checkerboard is essential. Indeed, the assumption that a tiling exists amounts to an axiom of infinity, and this can be combined with a description of a finite checkerboard to obtain a contradiction without using cardinalities. We shall discuss a proof (which was not obtained automatically) of the theorem based on this idea.

We present only the key lines of the proof below. We have numbered them in a way which suggests a natural structuring of the proof. These numbers are just being used as labels; the gaps between lines would have to be considerably larger than suggested by the line numbers to accommodate all the intermediate steps of the formal proof.

(0) Conjunction of lines (1)-(8) Hyp

Line 0 is the hypothesis which we wish to refute. All the lines of the proof below except the last line (900) are derived from it. Lines (1)-(8) are the sentences in the elegant and concise second formulation of the problem in [12].

(1) $s_{\iota\iota}\ [s\ .s\ .s\ .s\ .s\ .s\ .s\ 8_{\iota}] = 8$

(2) $\forall x_{\iota}\ .\sim .s_{\iota\iota}\ [s\ .s\ .s\ x] = x$

(3) $\forall x_{\iota}\ \forall y_{\iota}\ .g_{\iota\iota\iota}\ x\ y = 5_{\iota} \equiv .x = 8_{\iota} \wedge y = 8$
$\vee\ .x = 1_{\iota} \wedge y = 1$

(4) $\forall x_{\iota}\ \forall y_{\iota}\ .g_{\iota\iota\iota}\ x\ y = 1_{\iota} \equiv g\ [s_{\iota\iota}\ x]\ y = 3_{\iota}$

(5) $\forall x_{\iota}\ \forall y_{\iota}\ .g_{\iota\iota\iota}\ x\ y = 2_{\iota} \equiv g\ x\ [s_{\iota\iota}\ y] = 4_{\iota}$

(6) $\forall x_{\iota}\ \forall y_{\iota}\ .\ \sim[g_{\iota\iota\iota}\ 1_{\iota}\ y = 3_{\iota}] \wedge \sim[g\ 8_{\iota}\ y = 1]$
$\wedge \sim[g_{\iota\iota\iota}\ x\ 1 = 4_{\iota}] \wedge \sim .g\ x\ 8 = 2_{\iota}$

(7) $1_{\iota} = s_{\iota\iota}\ 8_{\iota} \wedge 2_{\iota} = s_{\iota\iota}\ 1_{\iota} \wedge 3_{\iota} = s_{\iota\iota}\ 2_{\iota} \wedge 4_{\iota} = s_{\iota\iota}\ 3_{\iota}$
$\wedge\ 5_{\iota} = s_{\iota\iota}\ 4_{\iota}$

(8) $\forall x_{\iota}\ \forall y_{\iota}\ .g_{\iota\iota\iota}\ x\ y = 1_{\iota} \vee g\ x\ y = 2_{\iota} \vee g\ x\ y = 3_{\iota}$
$\vee\ g\ x\ y = 4_{\iota} \vee g\ x\ y = 5_{\iota}$

g is a function which describes the tiling. g x y = 1 [2, 3, 4, respectively] means that the domino which covers square <x,y> also covers square <sx,y> [<x,sy>, <x-1,y>, <x,y-1>, respectively] (where x-1 means s .s .s .s .s .s .s x), and g x y = 5 means that the square <x,y> is not covered by any domino. (6) asserts that tiles don't hang off the edge of the board, but this condition is unnecessary and will not be used. Indeed, we may imagine that the checkerboard is wrapped around a torus, so that the upper and lower edges are adjacent, and the right and left edges are adjacent.

From (2) it is trivial to derive (100)-(117):

(100) $\forall x_{\iota}\ .\sim .s_{\iota\iota}\ [s\ x] = x$

(101) $\forall x_{\iota}\ .\sim .s_{\iota\iota}\ x = x$

(110) $\sim .4_{\iota} = 5_{\iota}$

(111) $\sim .3_{\iota} = 5_{\iota}$

(112) $\sim .1_{\iota} = 5_{\iota}$

(113) $\sim .3_{\iota} = 4_{\iota}$

(114) $\sim .2_{\iota} = 4_{\iota}$

(115) $\sim .2_{\iota} = 3_{\iota}$

(116) $\sim .1_{\iota} = 3_{\iota}$

(117) $\sim .1_{\iota} = 2_{\iota}$

(118) $\sim .2_{\iota} = 5_{\iota}$ 1, 100

(119) $\sim . 1_{\iota} = 4_{\iota}$ 118

A model of (0) might contain many objects in addition to those denoted by the terms 1, 2, 3, 4, 5, s5, ss5, and 8, so we define the set NUM which contains just these objects. We define it inductively for the sake of generality (i.e., so that our proof will be easier to modify to accommodate larger checkerboards) and to facilitate proofs of certain lemmas.

In making definitions, we shall write "**A** $=_D$ **B**" to mean that "**A** is defined to be the wff **B**".

$$\mathrm{NUM} =_D \lambda x_{\iota}\ \forall p_{o\iota} . p\ 1_{\iota} \wedge \forall w_{\iota}\ [p\ w \supset p . s_{\iota\iota}\ w] \supset p\ x$$

The definition says that x_{ι} is in NUM iff it is in every set which contains 1_{ι} and is closed under $s_{\iota\iota}$. With this definition, it is fairly straightforward to prove (130)-(143):

(130) $\forall x_{\iota}\ . \mathrm{NUM}\ x \supset \mathrm{NUM}\ . s_{\iota\iota}\ x$

(131) $\forall x_{\iota}\ . \mathrm{NUM}\ x \supset \mathrm{NUM}\ . s_{\iota\iota}\ . s\ x$

(140) $\forall x_{\iota}\ . \mathrm{NUM}\ x \supset s_{\iota\iota}\ [s\ . s\ . s\ . s\ . s\ . s\ . s\ x] = x$

(141) $\forall x_{\iota}\ \forall y_{\iota}\ . \mathrm{NUM}\ x \wedge \mathrm{NUM}\ y \wedge s_{\iota\iota}\ x = s\ y \supset x = y$

(142) $\forall x_{\iota}\ \forall y_{\iota}\ . \mathrm{NUM}\ x \wedge \mathrm{NUM}\ y \wedge s_{\iota\iota}\ [s\ x] = s\ [s\ y] \supset x = y$

(143) $\forall x_{\iota}\ \forall y_{\iota}\ . \mathrm{NUM}\ x \wedge \mathrm{NUM}\ y \wedge s_{\iota\iota}\ [s\ . s\ x] = s\ [s\ . s\ y] \supset x = y$

In a similar way, we define the set of black squares. In spite of the impression one might gain from Figure 3-1, we regard <1,1> and <8,8> as black squares. Because we can regard the board as being wrapped around a torus, the following simple inductive definition suffices:

$$\mathrm{BLACK} =_D \lambda u_{\iota}\ \lambda v_{\iota}\ \forall w_{o\iota\iota} . w\ 1_{\iota}\ 1 \wedge \forall j_{\iota}\ \forall k_{\iota}\ [w\ j\ k \supset . w\ [s_{\iota\iota} . s\ j]\ k \wedge w\ [s\ j] . s\ k] \supset w\ u\ v$$

(150) $\forall j_{\iota}\ \forall k_{\iota}\ . \mathrm{BLACK}\ j\ k \supset \mathrm{BLACK}\ [s_{\iota\iota}\ . s\ j]\ k$

(151) $\forall j_{\iota}\ \forall k_{\iota}\ . \mathrm{BLACK}\ j\ k \supset \mathrm{BLACK}\ [s_{\iota\iota}\ j]\ . s\ k$

(152) $\forall j_{\iota}\ \forall k_{\iota}\ . \mathrm{BLACK}\ j\ k \supset \mathrm{BLACK}\ [s_{\iota\iota}\ . s\ . s\ j]\ . s\ k$

(153) $\forall j_{\iota}\ \forall k_{\iota}\ . \mathrm{BLACK}\ j\ k \supset \mathrm{BLACK}\ [s_{\iota\iota}\ . s\ j]\ . s\ . s\ k$

(160) $\forall j_{\iota}\ \forall k_{\iota}\ . \mathrm{BLACK}\ j\ k \supset . \mathrm{NUM}\ j \wedge \mathrm{NUM}\ k$

(161) $\mathrm{BLACK} \subseteq \mathrm{NUM}\ \mathrm{X}\ \mathrm{NUM}$ 160

Now that we have some basic lemmas about the checkerboard, we introduce some definitions which will enable us to express the idea that a set of ordered pairs is infinite. Of course, the notions of finiteness and infinity can be defined in various ways. (See section 60 of [1] for a brief discussion of some possibilities.) One way to assert that a set k is infinite is to assert that it contains something which looks like the set of natural numbers as described by Peano’s Postulates: there is an initial object in the set, and there is a function h such that for any object p in the set, h(p) is a member of k different from all those previously considered. In this case, we may say that h "eXPLores" k.

$$\mathrm{INJ} =_D\ \lambda h_{o\iota\iota\iota\iota}\ \forall x1_\iota\ \forall y1_\iota\ \forall x2_\iota\ \forall y2_\iota\ \forall u_\iota\ \forall v_\iota\ .h\ x1\ y1\ u\ v \wedge$$
$$h\ x2\ y2\ u\ v \supset .x1 = x2 \wedge y1 = y2$$
$$\mathrm{XPL} =_D \lambda h_{o\iota\iota\iota\iota}\ \lambda k_{o\iota\iota}\ \lambda m_\iota\ \lambda n_\iota\ .k\ m\ n \wedge \forall x_\iota\ \forall y_\iota\ .k\ x\ y \supset$$
$$\exists u_\iota\ \exists v_\iota .h\ x\ y\ u\ v \wedge k\ u\ v \wedge \sim .u = m \wedge v = n$$
$$\mathrm{XINF} =_D \lambda k_{o\iota\iota}\ \exists h_{o\iota\iota\iota\iota}\ \exists m_\iota\ \exists n_\iota\ .\mathrm{INJ}\ h \wedge [h\ \mathrm{XPL}\ k]\ m\ n$$

[INJ h] means that h is injective when it is regarded as (the relation corresponding to) a function mapping ordered pairs to ordered pairs. Of course, we could use functions rather than the corresponding relations in our proof, but this would require using the Axiom of Descriptions (see section 53 of [1]), and we have chosen not to do this here. It would also be possible to formulate these definitions using ordered pairs as explicitly defined entities.
[h XPL k m n] means that h (regarded as a function mapping ordered pairs to ordered pairs) eXPLores the set k of ordered pairs starting from <m,n>, in a manner similar to the way the successor function explores the set of natural numbers starting from the number 0.
[XINF k] expresses the fact that the set k of ordered pairs is infinite.

We now come to the heart of the proof. We define a function H which maps a black square <x,y> to the black square <u,v> which is covered by the same domino as the white square <ssx,sy>. Note that H never maps a square to itself or to <1,1>.

$$H_{o\iota\iota\iota\iota} =_D\ \lambda x_\iota\ \lambda y_\iota\ \lambda u_\iota\ \lambda v_\iota .\mathrm{BLACK}\ x\ y \wedge$$
$$.[g_{\iota\iota\iota}\ [s_{\iota\iota} .s\ x]\ [s\ y] = 1_\iota \wedge u = s\ [s.s\ x] \wedge v = s\ y]$$
$$\vee [g_{\iota\iota\iota}\ [s_{\iota\iota} .s\ x]\ [s\ y] = 2_\iota \wedge u = s\ [s\ x] \wedge v = s\ [s\ y]]$$
$$\vee [g_{\iota\iota\iota}\ [s_{\iota\iota} .s\ x]\ [s\ y] = 3_\iota \wedge u = s\ x \wedge v = s\ y]$$
$$\vee [g_{\iota\iota\iota}\ [s_{\iota\iota} .s\ x]\ [s\ y] = 4_\iota \wedge u = s\ [s\ x] \wedge v = y]$$

Let us now consider how to prove (250) that H eXPLores BLACK. The main problem is to establish (241) that for each black square <x,y>, the square <ssx,sy> is indeed covered by some domino. For this we use a parity argument.
$*\!\equiv\!* =_D \lambda x_\iota\ \lambda y_\iota\ \forall p_{o\iota}\ .p\ x \wedge \forall u_\iota\ [p\ u \supset p\ .s_{\iota\iota}\ .s\ u] \supset p\ y$
[x *≡* y] means that x and y have the same parity; i.e., they are both even or both odd.

(201) $\forall x_\iota\ .x *\!\equiv\!* x$

(202) $\forall x_\iota\ .x *\!\equiv\!* s_{\iota\iota}\ .s\ x$

(205) $[\mathrm{TRANSITIVE} *\!\equiv\!*]$

(207) $\forall x_\iota\ \forall y_\iota\ .x *\!\equiv\!* y \supset s_{\iota\iota}\ x *\!\equiv\!* s\ y$

(209) $\forall x_\iota\ .\mathrm{NUM}\ x \supset s_{\iota\iota}\ [s\ x] *\!\equiv\!* x$

(220) $\forall x_\iota\ \forall y_\iota\ .\mathrm{BLACK}\ x\ y \supset x *\!\equiv\!* y$

The proof of (220) is an elegant structural induction based on the definition of BLACK.

(230) $\sim .1_\iota *\!\equiv\!* 2_\iota$

To prove (230), one assumes that $1_\iota *\!\equiv\!* 2_\iota$, and instantiates the quantifier in the definition of $*{\equiv}*$ with $[\lambda t_\iota . t = 1_\iota \vee t = 3_\iota \vee t = 5_\iota \vee t = s\ .s\ 5]$. Hence one derives $[2 = 1_\iota \vee 2 = 3_\iota \vee 2 = 5_\iota \vee 2 = s\ .s\ 5]$, which leads to a contradiction.

$$(240)\ \forall x_\iota\ \forall y_\iota\ .\ \ \mathrm{BLACK}\ x\ y\ \supset$$
$$\sim.[s_{\iota\iota}\ [s\ x] = 1_\iota \wedge s\ y = 1]\ \vee\ [s\ [s\ x] = 8_\iota \wedge s\ y = 8]$$

To prove (240), assume it is false, and use (220) to conclude that x $*{\equiv}*$ y. With the aid of (230) one can then obtain a contradiction.

$$(241)\ \forall x_\iota\ \forall y_\iota\ .\mathrm{BLACK}\ x\ y \supset \sim.g_{\iota\iota\iota}\ [s_{\iota\iota}\ .s\ x]\ [s\ y] = 5_\iota \qquad 3,\ 240$$

$$(250)\ [H_{o\iota\iota\iota\iota}\ \mathrm{XPL}\ \mathrm{BLACK}]\ 1_\iota\ 1$$

The next objective is to prove (270) that H is injective. Line (255) below, which is easily derived using the definition of H, shows that we can consider four cases, which are determined by the value of guv. The cases are very similar, and for the case where guv = 3 we obtain (266) and its corollary (267). From (267) and the corresponding results for the other cases it is easy to derive (270).

$$(255)\ \forall x_\iota\ \forall y_\iota\ \forall u_\iota\ \forall v_\iota\ .H_{o\iota\iota\iota\iota}\ x\ y\ u\ v\ \supset$$
$$.g_{\iota\iota\iota}\ u\ v = 1_\iota\ \vee\ g\ u\ v = 2_\iota\ \vee\ g\ u\ v = 3_\iota\ \vee\ g\ u\ v = 4_\iota$$

$$(266)\ \forall x_\iota\ \forall y_\iota\ \forall u_\iota\ \forall v_\iota\ .\ \ H_{o\iota\iota\iota\iota}\ x\ y\ u\ v \wedge g_{\iota\iota\iota}\ u\ v = 3_\iota\ \supset$$
$$u = s_{\iota\iota}\ [s\ .s\ x] \wedge v = s\ y$$

$$(267)\ \forall x_\iota\ \forall y_\iota\ \forall u_\iota\ \forall v_\iota\ .\ \ H_{o\iota\iota\iota\iota}\ x\ y\ u\ v \wedge g_{\iota\iota\iota}\ u\ v = 3_\iota\ \supset$$
$$s_{\iota\iota}\ [s\ .s\ .s\ .s\ u] = x\ \wedge\ s\ [s\ .s\ .s\ .s\ .s\ .s\ v] = y$$

$$(270)\ \mathrm{INJ}\ H_{o\iota\iota\iota\iota}$$

$$(299)\ \exists h_{o\iota\iota\iota\iota}\ \exists m_\iota\ \exists n_\iota\ .\mathrm{INJ}\ h \wedge [h\ \mathrm{XPL}\ \mathrm{BLACK}]\ m\ n \qquad 250,\ 270$$

$$(300)\ \mathrm{XINF}\ \mathrm{BLACK} \qquad \mathrm{Defn:}\ 299$$

This completes the first half of the proof. We now need to devote some attention to the finiteness of the checkerboard. We first introduce a very natural inductive definition of finiteness (which is not the negation of XINF).

ADD1 $=_D \lambda p_{o\iota}\ \lambda x_\iota\ \lambda t_\iota . p\ t \vee t = x$.

[ADD1 $r_{o\iota}$ x_ι] is printed as $[r_{o\iota} + x_\iota]$, and denotes the result of adding the element x_ι to the set $r_{o\iota}$; in more conventional notation it would be printed as $[r_{o\iota} \cup \{x_\iota\}]$.

FINITE $=_D \lambda p_{o\iota}\ \forall w_{o(o\iota)} . w\ [\lambda x_\iota\ \bot] \wedge \forall r_{o\iota}\ \forall x\ [w\ r \supset w.r + x] \supset w\ p$

$\bot$ denotes falsehood, so $[\lambda x_\iota\ \bot]$ denotes the empty set. We thus define a set $p_{o\iota}$ to be finite iff it is in every collection $w_{o(o\iota)}$ of sets which contains the empty set and is closed under any application of ADD1.

$$(401)\ \mathrm{FINITE}.\lambda x_\iota\ \bot$$

$$(402)\ \forall r_{o\iota}\ \forall x_\iota . \mathrm{FINITE}\ r \supset \mathrm{FINITE}.r + x$$

NUM8 $=_D [\lambda x_\iota\ \bot] + 1_\iota + 2_\iota + 3_\iota + 4_\iota + 5_\iota + s_{\iota\iota}\ 5 + s\ [s\ 5] + 8_\iota$

By expanding the definitions it is easy to see that

NUM8 $= [\lambda t_\iota . t = 1_\iota \vee t = 2_\iota \vee t = 3_\iota \vee t = 4_\iota \vee t = 5_\iota \vee t = s_{\iota\iota}\ 5 \vee t = s\ [s\ 5] \vee t = 8_\iota]$

(410) FINITE NUM8 401 402

(450) NUM = NUM8

(500) FINITE NUM 410 450

We must next find a way to make a useful connection between (300) and (500).
XFIN $=_D \lambda k_{o\iota\iota}.\sim$XINF k
HXFIN $=_D \lambda s_{o\iota\iota}\ \forall k_{o\iota\iota}.k \subseteq s \supset$ XFIN k
We may informally speak of a set k of ordered pairs as being xfinite iff [XFIN k]. Then [HXFIN s] means that s is hereditarily xfinite, i.e., all subrelations of s are xfinite.
ADDPAIR $=_D \lambda k_{o\iota\iota}\ \lambda x_\iota\ \lambda y_\iota\ \lambda u_\iota\ \lambda v_\iota$.k u v $\vee$ .u = x $\wedge$ v = y
[ADDPAIR $s_{o\iota\iota}\ x_\iota\ y_\iota$] is printed as [+ $s_{o\iota\iota}\ x_\iota\ y_\iota$], and denotes the result of adding the ordered pair <x,y> to the set s of ordered pairs.

(600) $\forall p_{o\iota}$.FINITE p $\supset$ $\forall q_{o\iota}$.FINITE q $\supset$
$\forall w_{o(o\iota\iota)}$.w [$\lambda u_\iota\ \lambda v_\iota\ \bot$] $\wedge$ $\forall x_\iota\ \forall y_\iota\ \forall s_{o\iota\iota}$ [w s $\supset$ w.+ s x y]
$\supset$ w.p X q

The proof of (600) is a straightforward proof by double induction based on the definition of FINITE.

(601) $\forall w_{o(o\iota\iota)}$ .w [$\lambda u_\iota\ \lambda v_\iota\ \bot$] $\wedge$ $\forall x_\iota\ \forall y_\iota\ \forall s_{o\iota\iota}$ [w s $\supset$ w .+ s x y]
$\supset$ w .NUM X NUM 500, 600

(602) HXFIN [$\lambda u_\iota\ \lambda v_\iota\ \bot$]
$\wedge$ $\forall x_\iota\ \forall y_\iota\ \forall s_{o\iota\iota}$ [HXFIN s $\supset$ HXFIN .+ s x y]
$\supset$ HXFIN .NUM X NUM 601

(701) HXFIN .$\lambda u_\iota\ \lambda v_\iota\ \bot$

(750) $\forall x_\iota\ \forall y_\iota\ \forall r_{o\iota\iota}$ .XINF [+ r x y] $\supset$ XINF r

This theorem serves as a lemma for proving (760). The formal proof contains 266 lines. Let us provide an informal description of the basic ideas in it. Given that r $\cup$ {<i,j>} is XINFinite, we must show that r is XINFinite. If <i,j> is already in r, we are done, so we may assume that ~rij. We are given an injective h which eXPLores r $\cup$ {<i,j>} from <m,n>. If <i,j> = <m,n>, we can choose <m′,n′> distinct from <m,n> such that hmnm′n′, and show that h eXPLores r from <m′,n′>.

The remaining case is that <i,j> ≠ <m,n>. The problem is that there may be a pair <x,y> ε r such that hxyij, and h does not map <x,y> to anything in r. In this case we must modify h. Let $h'_{o\iota\iota\iota\iota}$ = [$\lambda x_\iota\ \lambda y_\iota\ \lambda u_\iota\ \lambda v_\iota$.$r_{o\iota\iota}$ x y $\wedge$.$h_{o\iota\iota\iota\iota}$ x y u v $\wedge$ r u v $\vee$ .h x y $i_\iota\ j_\iota$ $\wedge$ h i j u v]. h′ generally does what h does, but if h maps <x,y> to <i,j>, and maps <i,j> to <u,v>, then h′ maps <x,y> directly to <u,v>. It can be seen that h′ is injective, and that h′ eXPLores r from <m,n>.

(760) $\forall x_\iota\ \forall y_\iota\ \forall s_{o\iota\iota}$ .HXFIN s $\supset$ HXFIN .+ s x y 750

(800) HXFIN .NUM X NUM 602, 701, 760

(801) $\forall k_{o\iota\iota}$ .k $\subseteq$ NUM X NUM $\supset$ XFIN k Defn: 800

(802) BLACK $\subseteq$ NUM X NUM $\supset$ XFIN BLACK 801

(803) XFIN BLACK 161, 802

(804) ~XINF BLACK Defn: 803

(899) $\bot$ 300, 804

(900)
$$\begin{array}{l}
\vdash\ \sim.\ s_{\iota\iota}\ [s\ .s\ .s\ .s\ .s\ .s\ .s\ 8_{\iota}] = 8 \\
\quad \wedge\ \forall x_{\iota}\ [\sim.s\ [s\ .s\ .s\ x] = x] \\
\quad \wedge\ \forall x\ \forall y_{\iota}\ [g_{\iota\iota\iota}\ x\ y = 5_{\iota} \equiv \\
\qquad .x = 8 \wedge y = 8 \vee .x = 1_{\iota} \wedge y = 1] \\
\quad \wedge\ \forall x\ \forall y\ [g\ x\ y = 1 \equiv g\ [s\ x]\ y = 3_{\iota}] \\
\quad \wedge\ \forall x\ \forall y\ [g\ x\ y = 2_{\iota} \equiv g\ x\ [s\ y] = 4_{\iota}] \\
\quad \wedge\ \forall x\ \forall y\ [\ \sim[g\ 1\ y = 3]\ \wedge\ \sim[g\ 8\ y = 1] \\
\qquad\qquad \wedge\ \sim[g\ x\ 1 = 4]\ \wedge\ \sim.g\ x\ 8 = 2] \\
\quad \wedge\ 1 = s\ 8\ \wedge\ 2 = s\ 1\ \wedge\ 3 = s\ 2\ \wedge\ 4 = s\ 3 \\
\qquad \wedge\ 5 = s\ 4 \\
\quad \wedge\ \forall x\ \forall y\ .g\ x\ y = 1\ \vee g\ x\ y = 2\ \vee g\ x\ y = 3 \\
\qquad\qquad \vee\ g\ x\ y = 4\ \vee\ g\ x\ y = 5
\end{array}$$

Indirect proof: 899

4. Conclusion

One can easily modify McCarthy’s statement of the checkerboard problem to accommodate larger checkerboards by adding an even number of occurrences of $s_{\iota\iota}$ to the formula in line (1), and (if necessary) adding (118) as an axiom. As a matter of good notational taste, one should also replace the constant 8_{ι} by a different constant everywhere; 8_{ι} simply denotes the largest number needed to describe the checkerboard, whatever it may be. By making this number sufficiently large, one can cause brute-force proofs of the toughnut theorem to be arbitrarily long. On the other hand, the proof given above need be extended only slightly to accommodate such a change in the theorem. (The main adjustments are in the proofs of (230) and (500), where we essentially list all the odd numbers, or all the numbers. If one adds an odd number of occurrences of $s_{\iota\iota}$, the theorem is still provable, since the number of squares is odd, but our proof of (230) will fail.)

We are reminded of Gödel’s assertion in [8] that "passing to the logic of the next higher order has the effect, not only of making provable certain propositions that were not provable before, but also of making it possible to shorten, by an extraordinary amount, infinitely many of the proofs already available." A complete proof of this may be found in [5]. Gödel was actually referring to n-th order arithmetic when he wrote n-th order logic, but a related result by Statman (Proposition 6.3.5 of [16]) establishes that the minimal length of a proof in first-order logic of a wff of first-order logic may be extraordinarily longer than that the minimal length of a proof of the same wff in second-order logic.

Most methods for automatically proving theorems of first- or higher-order logic which have been developed so far trace their origins back to the methods of Herbrand, Gentzen, or Beth. In general, the proofs found automatically by these methods have the subformula property: all the expressions which occur in the proof of a theorem are constructed from expressions which are in an appropriate sense subformulas of the theorem. Thus, such a method can find only a first-order proof for a first-order theorem. Our proof of the toughnut theorem is an example of a proof which does not have the subformula property.

Much remains to be done in developing improved methods for finding proofs with the subformula property of theorems of first-order logic. However, what we can learn from the examples discussed above is that we must also direct some research toward broader horizons. We need to develop good search procedures for logical languages such as type theory which are richer than first-order logic, and we need to develop quite different search procedures which may produce proofs without the subformula property. To ignore these research directions would severely limit what we can hope to achieve.

5. References

1. Peter B. Andrews, *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof,* Academic Press, 1986.

2. Peter B. Andrews, Matthew Bishop, Sunil Issar, Dan Nesmith, Frank Pfenning, Hongwei Xi, *TPS: A Theorem Proving System for Classical Type Theory*, Journal of Automated Reasoning (to appear).

3. Grzegorz Bancerek, "The Mutilated Chessboard Problem - checked by Mizar," in *The QED Workshop II,* edited by Roman Matuszewski, 1995, 43-45. Available from http://www.mcs.anl.gov/qed/index.html

4. Robert Boyer, Ewing Lusk, William McCune, Ross Overbeek, Mark Stickel, and Lawrence Wos, *Set Theory in First-Order Logic: Clauses for Gödel's Axioms*, Journal of Automated Reasoning **2** (1986), 287-327.

5. Samuel R. Buss, *On Gödel's Theorem on Lengths of Proofs I: Number of Lines and Speedup for Arithmetic*, Journal of Symbolic Logic **59** (1994), 737-756.

6. Alonzo Church, *A Formulation of the Simple Theory of Types*, Journal of Symbolic Logic **5** (1940), 56-68.

7. James Glanz, *Computer Scientists Rethink Their Discipline's Foundations*, Science **269** (September 1995), 1363-1364.

8. Kurt Gödel, *Über die Länge von Beweisen*, Ergebnisse eines Mathematischen Kolloquiums (1936), 23-24. Translated in [10], pp. 396-399.

9. Kurt Gödel, *The Consistency of the Axiom of Choice and of the Generalized Continuum Hypothesis with the Axioms of Set Theory,* Princeton University Press, Princeton, 1940.

10. Kurt Gödel, *Collected Works, Volume I,* Oxford University Press, 1986.

11. Seth Lloyd, *Quantum-Mechanical Computers*, Scientific American **273** (October 1995), 140-145.

12. John McCarthy. A Tough Nut for Proof Procedures, Stanford Artificial Intelligence Project Memo No. 16, 1964.
Available from http://www-formal.stanford.edu/jmc/.

13. John McCarthy. http://www-formal.stanford.edu/jmc/nut.html.,

14. John McCarthy, "The Mutilated Checkerboard in Set Theory," in *The QED Workshop II*, edited by Roman Matuszewski, 1995, 25-26. Available from http://www.mcs.anl.gov/qed/index.html.

15. William McCune, *Another Crack in a Tough Nut*, Association for Automated Reasoning Newsletter **31** (1995), 1-3.

16. Richard Statman, *Bounds for Proof Search and Speed-up in the Predicate Calculus*, Annals of Mathematical Logic **15** (1978), 225-287.

17. Sakthi Subramanian, *An Interactive Solution to the n x n Mutilated Checkerboard Problem*, Journal of Logic and Computation (to appear).

18. Thomas E. Uribe and Mark E. Stickel, "Ordered Binary Decision Diagrams and the Davis-Putnam Procedure," in *Proceedings of the First International Conference on Constraints in Computational Logics*, edited by J. P. Jouannaud, Lecture Notes in Computer Science 845, Springer-Verlag, 1994, 34-49.

19. Larry Wos, Ross Overbeek, Ewing Lusk, Jim Boyle, *Automated Reasoning. Introduction and Applications,* Prentice Hall, 1984.

Table of Contents

List of Figures