Two Computer-Supported Proofs in Metric Space Topology*

William M. Farmer and F. Javier Thayer

The MITRE Corporation

22 October 1992

1 Introduction

Every mathematician will agree that the discovery, analysis, and communication of theorems and proofs is at the heart of his or her discipline. A number of computer programs (such as Maple or Mathematica) assist mathematicians in testing conjectures and proving certain kinds of theorems, typically identities involving rational functions or trigonometric polynomials. These systems, however, were never intended for some of the most critical parts of the mathematical process: formulation of concepts and theories and rigorous proof of general theorems.

There are other programs, less well known in the mathematical community, that are designed to provide computer support for the actual theorem proving process. These programs, usually called theorem provers, tend to be very specialized tools, aimed at discovering or checking proofs in languages or logical systems not ordinarily used by mathematicians. Nevertheless, some theorem provers have been used to produce fully machine-checked proofs of mathematically significant results; for examples, see [1, 2, 3, 8, 9].

In this article we discuss two proofs that were created with the help of a computer theorem proving system called IMPS (Interactive Mathematical

*Supported by the MITRE-Sponsored Research Program. Appears in the Computers and Mathematics column of *Notices of the American Mathematical Society* **38** (1991) 1133–1138. Authors' address: The MITRE Corporation, 202 Burlington Rd, Bedford MA 01730-1420 USA; Telephone: 617-271-2749; Email: farmer@mitre.org, jt@mitre.org.

Proof System), which is currently being developed at The MITRE Corporation. The fact that these results were proven using a computer is not in itself noteworthy. However, it is significant that the proof itself can be organized in a way which is at the same time comprehensible to a mathematically trained person and recognizably valid to the computer. We want to conclude from this that computers can indeed support the standard techniques of mathematics and can provide strong organizational, as well as computational, assistance for theorem proving.

The two theorems proved are the following elementary results about the topology of metric spaces:

Theorem 1 *Let f be a continuous mapping from a metric space M to a metric space N. If $A \subseteq M$ is* **connected***, then $f(A)$ is likewise* **connected***.*

Theorem 2 *Let f be a continuous mapping from a metric space M to a metric space N. If $A \subseteq M$ is* **sequentially compact***, then $f(A)$ is likewise* **sequentially compact***.*

In Section 2, we shall discuss some of the facets of IMPS that make it particularly suitable for formulating and reasoning about mathematics. Then in Section 3 we will describe the formal theory in which the theorems are stated and proved. The proofs of the two theorems are given in Section 4. The final section, Section 5, contains a short conclusion.

2 IMPS

The IMPS system is intended to provide computational support for rigorous mathematical reasoning in a style that closely conforms to conventional practice. It can be used to formulate axiomatic theories and to prove theorems in them. The major goal of the system is to provide users with the means to develop machine-checked proofs that are convincing and intelligible to a wide audience.

For a detailed overview of the IMPS system see [6]. In the rest of this section we shall describe three aspects of IMPS that facilitate the construction of intelligible proofs: its logic, its support for the axiomatic method, and its style of proof.

Logic

In IMPS all concept formulation, calculation, and inference is performed with respect to a formal logic that is a version of classical higher-order predicate logic. This logic provides strong support for specifying and reasoning about functions. In particular, functions may be higher order (have arguments which are themselves functions) and partial (not defined on all arguments). In addition, formulas may contain arbitrary quantification (universal or existential) over functions. Partial functions are handled in a direct manner, without introducing special error elements. This means that some terms such as $2/0$ have no value assigned to them. A direct approach to partial functions is very convenient for formalizing mathematics in which partial functions play a prominent role. This is especially true for analysis where partial higher-order functions, such as the differentiation operator on functions, occur naturally.

The IMPS logic is equipped with a hierarchy of objects called *sorts* which denote classes of elements. Sorts are used to help specify the value of an expression and to restrict quantification. They are especially useful for reasoning with respect to overlapping domains. For example, suppose $\mathbf{Z}$ and $\mathbf{R}$ are sorts denoting the integers and the real numbers, respectively. Then the Archimedean principle for the real numbers can be expressed quite naturally as

$$\text{for every } a : \mathbf{R} \quad \text{for some } n : \mathbf{Z} \quad a < n.$$

For more information on the IMPS logic, see [4, 5, 7].

2.1 Axiomatic Method

The axiomatic method comes in two basic styles. There is the "big theory" style in which all reasoning is carried out within one theory—usually some highly expressive theory, such as Zermelo-Fraenkel set theory. There is also the "little theories" style in which results are proven in small abstract theories and then used in more concrete theories. This latter style of the axiomatic method is employed extensively in standard practice. For example, if a mathematician needed a fact about multiplication over the nonzero elements in a field, he would usually not prove it in the context of fields if he could get away with proving it as a general result in group theory. That is, he would prove a general result in group theory, notice that multiplication over the nonzero elements has the structure of a group, and then appropriately instantiate the general result to obtain the desired fact. In other words, a mathematician will typically prove a result in a context free of unnecessary details, so that the result can be used freely in a variety of more specialized contexts.

Most theorem provers today support only the big theory style of the axiomatic method. In contrast, IMPS provides a number of facilities for using the axiomatic method in the little theories style. Users of IMPS can freely formulate multiple axiomatic theories. Each theory consists of a formal language—specified by a set of atomic sorts and constants—and a set of axioms expressed in the language. Theories are related to each other by theory interpretations. A theory interpretation is a syntactic device for translating the language of a theory T to the language of a theory T' with the property that each theorem of T is translated to a theorem of T'. Theory interpretations thus provide a mechanism for "transporting" theorems from abstract theories to more concrete theories.

The IMPS theory and theory interpretation mechanisms should be useful in much of mathematical analysis, where reasoning is typically done at various levels of abstraction. For example, the proof of the Picard-Lindelöf existence theorem for ordinary differential equations is often proved in textbooks by applying the fixed point principle for contractive mappings on a complete metric space. In the terminology of IMPS, that approach requires the construction of a theory interpretation in which a metric space is interpreted as a space of continuous functions on an interval. The real work here consists of showing that this instantiation is valid, which involves among other things, reasoning about integrals of real-valued functions on intervals.

2.2 Proofs

IMPS produces formal proofs, but they are very different from the formal proofs that are described in logic text books. Usually a formal proof is a tree or graph constructed in a purely syntactic way from axioms, previously proved theorems, and a small number of low-level rules of inference. Formal proofs of this kind tend to be composed of an enormous number of small logical steps and for this reason are usually exceedingly hard to understand. In contrast, the steps in an IMPS proof can be very large, and most low-level inference in the proof is performed by an expression simplification routine. Since inference is described at a high-level, proofs constructed in IMPS resemble informal proofs, but unlike an informal proof, all the details of an IMPS proof are machined checked.

In IMPS there are several devices for compressing complex deductions into single units. *Expression simplification* carries out myriad low-level inferences in one step using algebraic manipulation, term rewriting, and special algorithms for checking the definedness of terms. *Theorem assumption* allows one to assume intermediate assertions that have been proved independently, either in the home theory of the proof or in some appropriate outside theory. Collections of theorems can be automatically applied, in an organized manner, to a conjecture using *macetes*.[1] *Strategies* call rules of inference—including simplification, theorem assumption, and macete application—in useful patterns; they are akin to what are called tactics in other systems.

These devices for packaging inferences help the IMPS user to raise the essential ideas of a proof to the surface, while suppressing the details that would normally not appear in a written presentation of the proof. They also give the user the means to initiate and control machine deduction.

3 The Formal Theory

The formal proofs of Theorems 1 and 2 are carried out in a theory of metric space pairs. Before describing this theory, we need to describe two other basic theories that serve as building blocks for this theory.

3.1 Higher-Order Real Arithmetic

The IMPS theory of higher-order real arithmetic, called *h-o-real-arithmetic*, axiomatizes the real number system as a complete ordered field and characterizes the integers and rationals as imbedded structures. This is a fairly extensive theory, so we only describe it informally here. The language constants of this theory are of two kinds:

- The function constants *+*, ***, */*, *^*, *sub*, *-*, $<$, $\leq$ that denote the arithmetic operations of addition, multiplication, division, exponentiation, subtraction, negation, and the binary predicates less than and less than or equal to, respectively.
- An infinite set of individual constants, one for each rational number.

The atomic sorts of the language are **Z**, **Q**, **R** denoting the integers, rationals, and reals. Other constants and sorts can be added by definitions.

[1] In Portuguese, a macete is a clever trick.

The axioms of this theory are the usual field and order axioms as well as the completeness axiom which states that any predicate which is non-vacuous and bounded above has a least upper bound. This theory also contains the full second-order induction principle for the integers as an axiom. One can prove in this theory the basic facts about real numbers such as the archimedean principle stated above.

3.2 Metric Spaces

The theory *metric-spaces* is a formal theory of a single metric space. It is sufficiently expressive to formulate the basic concepts of a metric spaces, such as open and closed sets, connectedness, sequential compactness (equivalent to compactness for separable metric spaces), and continuity of real-valued functions. Results proven in this theory can be transported, for example, to the theory of higher-order real arithmetic. The theory is defined in IMPS by the two forms (or s-expressions) given below. The first of these forms defines the language (i.e., the atomic sorts and constants which constitute the vocabulary of the theory), and the second form essentially just lists the axioms of the theory.

```
(language-from-definition
 '(metric-spaces-language
   (embedded-languages h-o-real-arithmetic)
   (base-types points)
   (constants
    (dist (points points rr)))))
```

```
(theory-from-definition
 '(metric-spaces
   (component-theories h-o-real-arithmetic)
   (language metric-spaces-language)
   (axioms
    (positivity-of-distance
     "forall(x,y:points, 0<=dist(x,y))")
    (point-separation-for-distance
     "forall(x,y:points, x=y iff dist(x,y)=0)")
    (symmetry-of-distance
     "forall(x,y:points, dist(x,y) = dist(y,x))")
    (triangle-inequality-for-distance
     "forall(x,y,z:points, dist(x,z)<=dist(x,y)+dist(y,z))"))))
```

These forms say the following:

- The theory *metric-spaces* includes *h-o-real-arithmetic* as a subtheory (which, in particular, means completeness arguments can be freely used).

- The underlying set of the metric space is denoted by the sort *points*. The function constant *dist* denotes the distance between two points in the metric space. (Note: the sort $\mathbf{R}$ is entered at the keyboard by *rr*.)

3.3 Metric Space Pairs

The relevant theory for stating and proving Theorems 1 and 2 is called *metric-space-pairs*. The theory of a single metric space is insufficient to formulate a completely general theory of continuous functions between metric spaces, one that will include for instance continuous mappings between $\mathbf{R}^3$ to $\mathbf{R}^2$ as a special case of the general theory. IMPS has a theory replication mechanism which allows users to automatically create a new theory which contains a fixed number of imbedded copies of a given theory. By an imbedding we mean a theory interpretation as explained earlier in the paper.

Theory replication is essential for doing interesting mathematics because most often one considers several instances of the same structure and mappings between these structures. Thus ordinarily mathematicians think of "ring theory" not as a *formal* theory of a single ring (which does not provide much material for mathematical development) but at the very least as a theory of rings and morphisms between them.

The theory replication is specified in IMPS by the following form:

```
(poly-replicate-theory-with-definitions
 metric-spaces
 (list 'first 'second)
  'metric-space-pairs)
```

The theory *metric-space-pairs* includes *h-o-real-arithmetic* plus the following additional components:

- The additional atomic sorts (i.e., the atomic sorts of the theory's language other than those in *h-o-real-arithmetic*) are *first_points* and *second_points*.

- The additional constants are *first_dist* and *second_dist*.
- The additional axioms of the theory are the metric distance axioms (triangle inequality, symmetry, etc.) for the functions *first_dist* and *second_dist*.

The procedure *poly-replicate-theory-with-definitions* also creates a pair of translations from *metric-spaces* to *metric-space-pairs*. Moreover, defined constants in the theory *metric-spaces* are automatically translated in two different ways to similarly defined constants in the theory *metric-space-pairs*. For example, the predicate *open* which is defined in the theory *metric-spaces* is translated as two predicates *first_open* and *second_open*, corresponding to the property of being an open subset of *first_points* and *second_points*, respectively.

3.4 Indicators

So far we have not discussed how to quantify over sets of points in a way which would allow us to define connectedness or sequential compactness. One possibility is to imbed the theory of metric space pairs in a larger theory such as Zermelo-Fraenkel set theory. This approach is quite feasible but has the disadvantage that it requires a certain amount of preparatory work in set theory. Moreover, results in this larger theory can only be transported to other theories which are similarly imbedded in set theory. This considerably restricts the flexibility with which results can moved around from theory to theory.

In our development of metric spaces, we have adopted the more direct approach of conceptually identifying a set S with a function f which takes on a fixed value (say the number 1) on S and is undefined everywhere else. We call such functions *indicators*. We have developed several "generic" theories involving indicators which allow us to prove theorems about sets, covers and inverse images in a very abstract setting. Since these theories have no axioms, theorems proved in them can quite easily be transported to other theories.

The following is an example of a useful fact (named *direct-image-subset-conversion*) which can be proved in one of these generic theories. It allows us to replace, in certain cases, statements involving direct images with statements involving inverse images.

for every $f : ind_1 \to ind_2, a : sets[ind_1], b : sets[ind_2]$ implication

- $total(f, [ind_1 \to ind_2])$
- $f(a) \subseteq b \iff a \subseteq f^{-1}(b)$.

This is a useful result, since inverse images have nicer properties than direct images. We make use of this result and similar results in our computer-assisted proofs of Theorems 1 and 2.

3.5 Definitions

Theories can be enriched by sort and constant definitions. The definitions we need to formulate Theorems 1 and 2 all define predicate constants which denote boolean-valued functions. Our definitions of the requisite topological predicates are very close to the conventional ones adopted in most textbooks. For example, the following expression is the condition for x to be a connected subset of $points$.

for every $a, b : sets[points]$ implication

- conjunction
  - $open(a)$
  - $open(b)$
  - $empty?\{a \cap b \cap x\}$
  - $x \subseteq a \cup b$
- disjunction
  - $x \subseteq a$
  - $x \subseteq b$.

The condition for a function $f : first\_points \to second\_points$ to be continuous can be given in several equivalent ways, but for our purposes, the most convenient one is

for every $v : sets[second\_points]$ implication

- $second\_open(v)$
- $first\_open(f^{-1}(v))$.

Of course one can show in IMPS (with a certain amount of user assistance) that this condition is equivalent to the usual ϵ, δ condition for continuity.

4 The Proofs

We describe in this section formal proofs of Theorems 1 and 2 produced with IMPS. A proof in IMPS is represented by a data structure called a *deduction graph*. A deduction graph is a directed graph with nodes of two kinds, representing formulas and inferences respectively. The formulas appearing in a deduction graph are actually *sequents* consisting of a single formula called the *assertion* together with a set of assumptions called the *context*. A sequent is considered to be true if its context implies its assertion.

There are two basic routines in IMPS for presenting the information contained in a deduction graph in a TEX format. One routine describes each logical inference recorded in the deduction graph. The other routine is prescriptive: it presents the deduction graph in terms of the user commands (i.e., rules of inference, macetes, and strategies) that were used to construct the deduction graph. This is analogous to how proofs are given in a lecture or in a textbook. Few details are provided by the lecturer, who limits him or herself to giving the information on how to reconstruct the proof. The proofs in the section are presented using this latter routine.

The formal statement of the first theorem is:

Theorem 1

for every $f : first\_points \to second\_points, o : sets[first\_points]$ implication

- conjunction
  - $continuous(f)$
  - $total(f, [first\_points \to second\_points])$
  - $first\_connected(o)$
- $second\_connected(f(o))$.

We present the proof of Theorem 1 as it is actually formatted by IMPS using the prescriptive proof presentation routine. In this particular proof the full deduction graph consists of 18 sequent nodes.

Before presenting the proof, we give the intuitive idea behind it. The proof begins by expanding all definitions at the top level of the expression. In this case the defined constants are the predicates *first_connected*, *second_connected* and *continuous*. Since these constants are defined as λ-expressions (a λ-expression is a fancy name for a term of the form "the function which carries $x, y, \ldots$ to the expression *blah*"), expanding the definitions merely replaces these constants with their defining λ-expressions. We next have to apply the rule of β-reduction, which essentially plugs in values to the expressions which define the functions. This leaves us with an assertion which contains several subexpressions of the form $f(a) \subseteq b$. We now apply a user-defined *macete* which applies repeatedly a number of generic (and very easy to prove) results on indicators, such as the *direct-image-subset-conversion* lemma mentioned above. Application of this macete turns these subexpressions into subexpressions of the form $a \subseteq f^{-1}(b)$ and also uses the preservation properties of f^{-1}. To complete the proof, we use an *ending strategy*. An ending strategy attempts to find a proof of a goal sequent by successively applying rules of inference from a fixed list (depending on the strategy) and backtracking when a particular branch fails.

PROOF. Apply the strategy DEFINITION-EXPANSION to the claim of the theorem. This yields the following new subgoal:

Sequent 1.

for every $f : first\_points \to second\_points, o : sets[first\_points]$ implication

- conjunction
  - $\lambda\{f : first\_points \to second\_points \mid \forall v : sets[second\_points] \quad second\_open(v) \supset first\_open(f^{-1}(v))\} \quad (f)$
  - $total(f, [first\_points \to second\_points])$
  - $\lambda\{x : sets[first\_points] \mid \forall a, b : sets[first\_points] \quad (first\_open(a) \wedge first\_open(b) \wedge empty?\{a \cap b \cap x\} \wedge x \subseteq a \cup b) \supset (x \subseteq a \vee x \subseteq b)\} \quad (o)$
- $\lambda\{x : sets[second\_points] \mid$ for every $a, b : sets[second\_points]$ implication
  - $second\_open(a) \wedge second\_open(b) \wedge empty?\{a \cap b \cap x\} \wedge x \subseteq a \cup b$
  - $x \subseteq a \vee x \subseteq b\} \quad (f(o))$.

Apply the inference rule BETA-REDUCTION to the previous sequent. This yields the following new subgoal:

Sequent 2.

for every $f : first\_points \to second\_points, o : sets[first\_points]$ implication

- conjunction
  - $\forall v : sets[second\_points] \quad second\_open(v) \supset first\_open(f^{-1}(v))$
  - $total(f, [first\_points \to second\_points])$
  - $\forall a, b : sets[first\_points] \quad (first\_open(a) \wedge first\_open(b) \wedge empty?\{a \cap b \cap o\} \wedge o \subseteq a \cup b) \supset (o \subseteq a \vee o \subseteq b)$
- for every $a_0, b_0 : sets[second\_points]$ implication
  - $second\_open(a_0) \wedge second\_open(b_0) \wedge empty?\{a_0 \cap b_0 \cap f(o)\} \wedge f(o) \subseteq a_0 \cup b_0$
  - $f(o) \subseteq a_0 \vee f(o) \subseteq b_0$.

Apply the macete DIRECT-IMAGE-TO-INVERSE-IMAGE-CONVERSION-MACETE to the previous sequent. This yields the following new subgoal:

Sequent 3.

for every $f : first\_points \to second\_points, o : sets[first\_points]$ implication

- conjunction
  - $\forall v : sets[second\_points] \quad second\_open(v) \supset first\_open(f^{-1}(v))$
  - $total(f, [first\_points \to second\_points])$
  - $\forall a, b : sets[first\_points] \quad (first\_open(a) \wedge first\_open(b) \wedge empty?\{a \cap b \cap o\} \wedge o \subseteq a \cup b) \supset (o \subseteq a \vee o \subseteq b)$
- for every $a_0, b_0 : sets[second\_points]$ implication
  - $second\_open(a_0) \wedge second\_open(b_0) \wedge empty?\{f^{-1}(a_0) \cap f^{-1}(b_0) \cap o\} \wedge o \subseteq f^{-1}(a_0) \cup f^{-1}(b_0)$
  - $o \subseteq f^{-1}(a_0) \vee o \subseteq f^{-1}(b_0)$.

Apply the strategy PROVE-BY-LOGIC-AND-SIMPLIFICATION to the previous sequent. This completes the proof.

The formal statement of the Theorem 2 is almost identical to that of the previous theorem except that "connected" is replaced with "compact." Moreover, the sequence of user commands required to prove Theorem 2 is identical to that of Theorem 1. Of course, the actual TEX form of the proof is different because the goal formula and all intermediate formulas are different. We omit the details.

5 Conclusion

In this article we presented computer-supported proofs of two theorems in metric space topology. The theorems were stated within an axiomatic theory of metric space pairs using familiar topological concepts such as open set and continuous function which were defined in a very direct and natural way. The proofs were constructed within a version of predicate logic with the help of the IMPS theorem proving system. Each proof was fully machine checked and required only four commands from the user.

The theorems and proofs were developed in the little theories style of the axiomatic method. This approach benefited our proofs in two ways. First, both proofs utilized results proved outside of the theory of the proof, specifically results about converting statements involving direct images into statements involving inverse images. These results were proved in a theory about abstract mappings, and the system determined on its own how and where to apply them. Second, the two theorems can be transported to any theory that contains structures which are metric spaces. For example, they can be transported to the theory *h-o-real-arithmetic* by interpreting the two metric spaces of *metric-space-pairs* as $\mathbf{R}^3$ and $\mathbf{R}^2$.

The everyday practice of mathematics involves proving numerous elementary, but not entirely trivial results, which are similar in complexity to Theorems 1 and 2. In fact, many substantial theorems are proved by just skillfully combining elementary facts. Systems such as IMPS, which can be effectively used to formulate, prove, and apply elementary theorems, thus have the potential to play a significant role in mathematics research. Moreover, they offer mathematicians a new technology for organizing, checking, and reusing their work.

References

[1] W. W. Bledsoe. Some automatic proofs in analysis. In *Automated Theorem Proving: After 25 Years*, pages 89–118. American Mathematical Society, 1984.

[2] R. S. Boyer and J S. Moore. A theorem prover for a computational logic. In M. E. Stickel, editor, *10th International Conference on Automated Deduction*, volume 449 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 1990.

[3] S.-C. Chou. An introduction to Wu's method for mechanical theorem proving in geometry. *Journal of Automated Reasoning*, 4:237–267, 1988.

[4] W. M. Farmer. A partial functions version of Church's simple theory of types. *Journal of Symbolic Logic*, 55:1269–91, 1990.

[5] W. M. Farmer. A simple type theory with partial functions and subtypes. Technical report, The MITRE Corporation, 1991. Submitted to *Annals of Pure and Applied Logic*.

[6] W. M. Farmer, J. D. Guttman, and F. J. Thayer. IMPS: an Interactive Mathematical Proof System. Technical Report M90-19, The MITRE Corporation, 1991. Submitted to *Journal of Automated Reasoning*.

[7] J. D. Guttman. A proposed interface logic for verification environments. Technical Report M91-19, The MITRE Corporation, 1991.

[8] N. Shankar. *Proof Checking Metamathematics*. PhD thesis, University of Texas at Austin, 1986.

[9] C. A. Wick and W. W. McCune. Automated reasoning about elementary point-set topology. *Journal of Automated Reasoning*, 5:239–255, 1989.