An Overview of the MIZAR Project

Piotr Rudnicki*
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada T6G 2H1

e-mail: piotr@cs.ualberta.ca

June 30, 1992

Abstract

The MIZAR project is a long-term effort aimed at developing software to support a working mathematician in preparing papers. A. Trybulec, the leader of the project, has designed a language for writing formal mathematics. The logical structure of the language is based on a natural deduction system developed by Jaśkowski. The texts written in the language are called MIZAR articles and are organized into a data base. The Tarski-Grothendieck set theory forms the basis of doing mathematics in MIZAR. The implemented processor of the language checks the articles for logical consistency and correctness of references to other articles.

1 Introduction

The idea that an automatic device should check our logical derivations is by no means new. It can be traced back not only to Pascal and Leibniz, but to Ramon Llull. In recent years, several projects have aimed at providing computer assistance for doing mathematics. Among the better known there are: AUTOMATH [4], EKL [10], QUIP [22], Nuprl [3], THEAX [14], Computational Logic [2], Ontic [11], and the more recent ones such as ALF, ELF, HOL, LEGO and many others (see [6, 7]). The specific goals of these projects vary. However, they have one common feature: the human writes mathematical texts and the machine verifies their correctness.

The project MIZAR started almost 20 years ago under the leadership of Andrzej Trybulec at the Płock Scientific Society, Poland. Its original goal was to design and implement a software environment to assist the process of preparing mathematical papers. For lack of a better alternative, the project was based upon the style of doing mathematics used by the mathematicians of the so-called Polish mathematical school. Therefore, the project can be seen as an attempt to develop software environment for writing traditional mathematical papers, where classical logic and set theory form the basis of all future developments.

The logical basis of the system is a "Polish" style of natural deduction. Only after years of using the logic has it been learned that it was the "composite system of logic" developed by Stanisław Jaśkowski, see [8] and [12] for the English translation. Katuzi Ono, [16] described a

*This work was supported in part by NSERC Grant OGP9207.

similar system. Various formalizations of set theory have been tried (Zermelo-Fraenkel, Morse-Kelley) but finally the Tarski-Grothendieck axiomatization has been adopted.

2 A bit of history

The name Mizar[1] was picked up in 1973 for a different project (a programming environment) that was discontinued, but its name has been recycled.

The first experiments in 1974-75 developed a modest proof-checker for propositional logic. The main concern of A. Trybulec was the input language to the checker, and it turned out to be the main concern for all future years. The proof checker was based on a fixed set of inference rules. The justification of an inference in the input text required the user to state only the premises and the conclusion; the checker searched for a rule or a sequence of rules to validate an inference step. This approach was abandoned, and all future MIZAR processors have used model checking.

In 1977 the language and the checker were extended with quantifiers to form MIZAR QC which had neither functional notation nor definitional facilities. These were added in subsequent years to form the MIZAR FC system, which was used to record a number of larger texts. Among these texts was the initial segment of the book on arithmetics by Grzegorczyk [5]. The book is so rigorous and detailed (the Chinese remainder theorem comes as late as page 67 and its proof takes 6 pages) that the blow-up factor in the translation to MIZAR FC was negligible.

At around 1978-1979 the MIZAR group started to grow substantially, being anchored at Warsaw University, Białystok[2] Branch[3].

In 1981, a language called MIZAR 2 had been designed by A. Trybulec and implemented on an ICL 1900 by Cz. Byliński, H. Oryszczyszyn, P. Rudnicki, and A. Trybulec. The system was written in Pascal and later ported to other computers (mainframe IBM and also to UNIX). Again, the stress was on the input language. The proof checker was rather weak which forced one to write very detailed proofs. The language included the following features: structured types, type hierarchy, comprehensive definitional facilities, built-in fragments of arithmetics, and a built-in variant of set theory. The translations into MIZAR 2 included: a number of recent papers in topology, projective geometry, and some text-book algebra. In one of the experiments, the pigeon-hole principle was proven from scratch and compared with the similar development by van Benthem Jutting [28] in AUT-QE (the AUTOMATH project). The MIZAR 2 text was about half as long as the AUT-QE text; however, MIZAR 2 and AUT-QE were substantially different environments for conducting proofs (logic based vs type based). Among other works with MIZAR 2, there were attempts to prove properties of programs [21] and software specifications [20].

In the following years, other MIZAR languages and their implementations have been developed but their character was experimental (MIZAR 3, MIZAR HPF); the systems were not distributed outside the MIZAR group in Białystok, with one exception.

[1]Mizar, ζ Ursae Majoris, is the second magnitude star set in the middle of the Big Dipper's handle; Mizar (Arabic: veil, cloak, burial grounds) makes a visual binary with the fainter Alcor (Arabic: faint one); each of the visual components is a spectroscopic binary; Mizar is a quadruple star.

Incidentally, Algol (Arabic: daemon, ghoul), β Persei, is an eclipsing visual binary star, spectroscopically a triple, probably a quadruple star as well.

[2]E. Post was born in Augustów, near Białystok. A. Lindenbaum was last seen in Białystok in 1941.

[3]Address: Mizar Group, Institute of Mathematics, Warsaw University—Białystok Branch, 15-276 Białystok, Poland.

A subset of MIZAR, named MIZAR MSE (short for Multi-Sorted with Equality) was implemented in 1982 by R. Matuszewski, P. Rudnicki, and A. Trybulec and has been widely used since then. The system is meant for teaching elementary logic with emphasis on the practical aspects of constructing proofs. The MIZAR MSE language encompasses 'raw' predicate calculus, multi-sorted with equality, but the language does not provide functional notation or special notation for definitions. There are numerous implementations of MIZAR MSE, see [26, 25, 13, 19, 18, 15].

In 1986, MIZAR 4 was implemented as a redesign of MIZAR 2 and distributed to several dozen users. Each MIZAR 4 article included a preliminaries part where the author could state some axioms that were not checked for validity.

In 1988 the design of the language was completed by A. Trybulec and the final language is named simply MIZAR. While articles in previous versions of the language must be self-contained, the final MIZAR allows for cross-references among articles. Moreover, an author of a MIZAR text is not allowed to introduce new axioms. Only the predefined axioms can be used, everything else must be proved. The two articles that are not checked for validity contain an axiomatics of the Tarski-Grothendieck set theory (see Appendix A) and definitional axioms of the built-in concepts and axioms of strong arithmetics of real numbers (see Appendix B).

Recently, the main effort in the MIZAR project has been in building the library of MIZAR articles which now numbers almost 300.

The development of MIZAR has been driven by "experience, not only doctrine (ENOD for short)"[4]. In this case it meant that any idea that made sense was first of all implemented and tried by its proponent. However, only after other users approved the idea by actually using its implementation, was the idea included into the state of the project. The ENOD approach has been applied in the development of the MIZAR processor, the input language (although it is almost exclusively of A. Trybulec's creation), and the MIZAR library.

Before giving more information about MIZAR it may be worthwhile to recall here the *Formulaire de mathématiques* project of Giuseppe Peano (quoted from Kennedy [9] p. 8):

> The end result of this project would be, he hoped, the publication of a collection of all known theorems in the various branches of mathematics. The notions of his mathematical logic were to be used and proofs of the theorems were to be given. There were five editions of the *Formulario*: the first appeared in 1895, and the last, completed in 1908, contained some 4200 theorems.

The project was based on a formal language. In the introduction to the second volume of *Formulaire de mathématiques*, Peano writes (quoted from [17], p. 197, vol. II):

> Dans le petit livre ≪Arithmetices principia, nova methodo exposita, a. 1889≫, nous avons pour la première fois exposé toute une théorie, théorèmes, définitions et démonstrations, en symboles qui remplacent tout-à-fait le langage ordinaire.
>
> Nous avons donc la solution du problème proposé par Leibniz.

3 Anatomy of a Mizar article

Each MIZAR article is written as a text file. The general structure of such an article is as follows:

[4]The expression has been coined by G. Kreisel, see his contribution in *Logic and Computer Science*, P. Odifreddi (ed.), Academic Press, pp. 205–278.

```
environ
                    Environment directives
begin
                    Text-Proper Section
                    . . .
                    . . .
begin
                    Text-Proper Section
```

The *Text-Proper* contains statements of facts with their proofs and definitions of new concepts with justification of their correctness. The *Environment directives* declare which items of the MIZAR library can be referenced from the *Text-Proper*. The directive

`vocabulary` *Vocabulary-File-Name*;

adds the symbols introduced in the *Vocabulary-File-Name* to the article's lexicon. Vocabulary files introduce new symbols of MIZAR expression constructors. The vocabularies also indicate the binding strength of the introduced symbols for parsing purposes. The authors can use existing vocabularies (there are hundreds of symbols there) and also are free to create new ones. The lexical make-up of new symbols is governed by a small set of quite liberal rules. The authors can freely enhance the zoo of mathematical symbols with new symbols of their own design.

One vocabulary is automatically attached to every MIZAR article. It introduces the following symbols: `Element`, `Subset`, `DOMAIN`, `Real`, `Nat`, `+`, `<>`, $\leq$, $\in$, and the like. The information pertaining to the usage of these symbols is built into the MIZAR processor, e.g. $\in$ can be used as a binary, infix predicate symbol with both arguments expandable to type `set`. Similarly, $\leq$ requires two arguments of type expandable to `Element of Real`.

Besides vocabularies, there are four kinds of data base directives:

`signature` *Signature-File-Name*;
`definitions` *Definitions-File-Name*;
`theorems` *Theorems-File-Name*;
`schemes` *Schemes-File-Name*;

The directive `signature` informs the MIZAR processor that the article is permitted to use the notation (definienda) introduced in article *Signature-File-Name*.

Any article can define ways in which the symbols contained in vocabularies can be used to form MIZAR expressions. Each of the MIZAR expression constructors (functor, predicate, mode) can be syntactically defined in a number of formats. These constructors may take various numbers of arguments of various types, and in the case of functors they may return results of various types. This creates a complicated system of overloaded constructors. The information needed for parsing is kept in auxiliary files associated with every article and commonly referred to as the signature of the article.

The remaining three directives allow us to use definitions, theorems, and schemes that are defined or proved in another article.

The *Text-Proper* is a sequence of sections, each being a sequence of *Text-Item*. There are the the following kinds of items:

- *Reservation* is used to reserve identifiers for a type. If a variable has an identifier reserved for a type, and no explicit type is stated for the variable, then the variable type defaults to the type for which its identifier was reserved.
- *Definition-Block* is used to define (or redefine) constructors of MIZAR phrases: term constructors (functions), formula constructors (predicates), and type constructors (modes).
- *Structure-Definition* introduces new structures. A structure is an entity that consists of a number of fields that are accessed by selectors.
- *Theorem* announces a proposition that can be referenced from other articles.
- *Scheme* also announces a proposition, visible from outside. Second order terms can occur in *scheme*.
- *Auxiliary-Item* introduces objects that are local to the article in which they occur and are not exported to the library files (e.g. lemmas, definitions of local predicates).

The goal of writing an article is to prove some theorems and schemes or to define some new concepts such that they can be referenced by other authors. Before the theorems and definitions are included into the library they must be proved valid and correct.

4 The PC Mizar

PC MIZAR is a MIZAR processor implemented on IBM PCs under DOS by Cz. Byliński, A. Trybulec, and S. Żukowski, and now further developed by the first two authors.

The central concept of MIZAR is a MIZAR *article*. Such an article can be viewed as an extremely detailed mathematical text written in a fixed formal notation, originally as a text file. There are rather few interesting things that one can prove in a short MIZAR article without making references to other articles. Usually, we base our work on the achievements of others.

The power of the MIZAR system is in its automatic processing of cross-references among articles contained in the MIZAR library. In order to speed up the process of cross reference checking, some internal files, derived from the submitted articles, are maintained. These files (they are not meant to be read by humans) are created in the process of including an article into the MIZAR library.

- *signature files* that for each newly defined constructor of MIZAR phrases in the article give information necessary for parsing the constructor occurrences.
- *definitions file* stores the definiens of every definition in the article, the definiendum is stored in the signature file.
- *theorems file* stores the theorems proved in the article (without proofs).
- *schemes file* stores the schemes proved in the article (without proofs).

The MIZAR software is a collection of about 50 programs that process MIZAR articles.

- The verifier must run in the appropriate environment with access to all the vocabulary and library files referenced in the given article. For efficiency reasons, each checked article obtains a dedicated environment (by a program called *accommodator*) in order to avoid too many references to different library files.

- Parsing of MIZAR texts is relatively complicated mainly because of the rich MIZAR syntax, multi-way overloading of names, and new definitions or redefinitions of MIZAR phrase constructors and their priorities.
- Checking whether proofs are correctly structured requires some processing as MIZAR permits a multitude of proof structures in the spirit of natural deduction proposed by Jaśkowski.
- An inference of the form

  $$premise_0,\ premise_1,\ \ldots,\ premise_k\ \vdash\ conclusion$$

  is transformed into the conjunction

  $$premise_0\ \&\ premise_1\ \&\ \ldots\ \&\ premise_k\ \&\ \texttt{not}\ conclusion.$$

  If the checker finds the conjunction contradictory then the original inference is accepted. Unfortunately but inevitably, the checker sometimes does not accept an inference that is logically correct; to get the inference accepted one has to split it into a sequence of 'smaller' ones, or possibly use a proof structure. The stress in the inference checker is on the processing speed, not power.

5 The Input Language

Experience has shown that people with even minimal mathematical training develop a good idea about the nature of the MIZAR language just by browsing through a sample article. This is not a big surprise as one of the original goals of the project was to build an environment that mimics the traditional ways that mathematicians work. A sample MIZAR article is presented in Appendix C.

Because of the richness of the MIZAR grammar, even a sketchy presentation of it is far beyond the scope of this text.

6 Mizar abstracts

The source texts of MIZAR articles tend to be lengthy as they contain complete proofs in a rather demanding formalism. New articles strongly depend on already existing ones. Therefore, there was a need to provide authors with a quick reference to the already collected articles. The solution was to automatically create an *abstract* for each MIZAR article. Such an abstract includes a presentation of all the items that can be referenced from other articles. The abstract of the article presented in Appendix C is contained in Appendix D. Therefore, there is no need to examine the entire article to make a reference to a single theorem.

To make the abstracts resemble a mathematical paper at least at the lexical level, they are automatically typeset using TEX. The TEXed MIZAR abstract from Appendix D is in Appendix E.

The typeset MIZAR abstracts are periodically published[5] by Université Catholique de Louvain as *Formalized Mathematics (a computer assisted approach)* with R. Matuszewski as editor.

[5] For more information write to: Fondation Philippe le Hodey, MIZAR Users Group, Av. F. Roosevelt 134 (Bte7), 1050 Brussels, Belgium, fax +32 (2) 640 89 68

7 Main Mizar Library

At the beginning of 1989, the MIZAR group in Białystok started collecting MIZAR articles and organizing them into a library that is distributed to other MIZAR users.

The person responsible for the library (E. Woronowicz) requires that authors of contributed articles supply an additional file that describes the bibliographic data such as title, authors' names and affiliations, and a summary (in English). The bibliographic information is included at the beginning of each typeset abstract.

As of June 19, 1992, the library consisted of 279 MIZAR articles authored by some 60 people. 214 theorem files that are referenced from other articles contained 5810 theorems; there were 109903 cross-references among articles. Totally there were about 15 MB of source text files with articles. Although the majority of articles have been authored by people from Białystok, many papers have been written by mathematicians from other universities in Poland, and there are articles written by foreign authors (Canada, Japan, Spain, USA).

The nature of the articles varies. Most of them are MIZAR translations of basic mathematics. Few of them contain new results. To get an idea about the contents of the library, look at Table 1 for a sample of article titles.

No.	Name	Inclusion date	Title	Author(s)
1	`BOOLE`	6.I.1989	*Boolean Properties of Sets*	Z. Trybulec and H. Święczkowska
17	`FUNCT_2`	6.IV.1989	*Functions from a Set to a Set*	Cz. Byliński
33	`WELLORD2`	26.IV.1989	*Zermelo Theorem and Axiom of Choice*	G. Bancerek
67	`FRAENKEL`	7.II.1990	*Function Domains and Fraenkel Operator*	A. Trybulec
106	`TRANSLAC`	12.VI.1990	*Translations in Affine Planes*	H. Oryszczyszyn and K. Prażmowski
185	`MEASURE1`	15.X.1990	*The σ-additive Measure Theory*	J. Białas
231	`ALI2`	17.VII.1991	*Fix Point Theorem for Compact Spaces*	A. de la Cruz
248	`HEINE`	21.XI.1991	*Heine-Borel's Covering Theorem*	A. Darmochwał and Y. Nakamura
274	`MIDSP_3`	28.V.1992	*Reper Algebras*	M. Muzalewski

Table 1: Sample of titles from MIZAR library.

The development of the MIZAR library may be perceived as an experiment in the sociology of mathematics. The acceptance criteria are very liberal: every submitted paper that is accepted by the MIZAR processor is included into the library. There are some efforts towards having automated reviewers (that looks, for example, for repeated theorems or trivial ones). There is a collection of programs called enhancers and improvers that try to automatically meliorate the submitted articles. The melioration makes changes in the submitted articles, that is, replaces a proof by a one step inference referencing a number of propositions, or replaces a sequence of inferences by a single one, or removes references to the superfluous premises in an inference step.

As the Mizar system evolves, and this includes the input language, there is a need to rewrite pieces of the library articles to mirror the changes. To a large extent, this is done automatically. Sometimes, however, a manual intervention is required.

The people maintaining the library collect statistics about the references across articles. Consideration is given to whether or not the frequently quoted theorems or definition should be built into the Mizar verifier. This was the fate of proposition `TARSKI:1` which stated that everything was a set, see Appendix A.

Table 2 contains the list of the 10 most frequently quoted theorems.

No. of ref.	% of all ref.	Name	Statement of the theorem
2347	2.1355%	`BOOLE:11`	$x \in X$ & $X \subseteq Y$ **implies** $x \in Y$
1642	1.4940%	`BOOLE:9`	$x \in X \cap Y$ **iff** $x \in X$ & $x \in Y$
1558	1.4176%	`TARSKI:3`	$X = \{y\}$ **iff for** x **holds** $x \in X$ **iff** $x = y$
1349	1.2274%	`BOOLE:8`	$x \in X \cup Y$ **iff** $x \in X$ **or** $x \in Y$
1252	1.1392%	`BOOLE:def 1`	$Z = \emptyset$ **iff not ex** x **st** $x \in Z$
1242	1.1301%	`BOOLE:29`	$X \subseteq Y$ & $Y \subseteq Z$ **implies** $X \subseteq Z$
964	0.8771%	`FINSEQ_1:13`	$k = \mathsf{len}\ p$ **iff** $\mathsf{Seg}\ k = \mathsf{dom}\ p$
922	0.8389%	`BOOLE:64`	$(X \cup Y) \cup Z = X \cup (Y \cup Z)$
848	0.7716%	`BOOLE:5`	$X \subseteq Y$ **iff for** x **holds** $x \in X$ **implies** $x \in Y$
792	0.7206%	`AXIOMS:2`	X **is** Subset **of** Y **iff** $X \subseteq Y$

Table 2: Top 10 theorems of Mizar library.

8 The future

The Mizar language: The logical level of the language has long been fixed. It is the type hierarchy that fuels all the changes. Recently, the development has focused on introducing a mechanism for deriving Mizar structures in the spirit of the object-oriented approach, and on deriving new Mizar modes by adding attributes to existing ones. Both proposed derivation techniques result in Boolean algebras of structures and sets of attributes, respectively. According to A. Trybulec both these changes will have a dramatic impact on the style of doing mathematics in Mizar.

The language still lacks some polymorphic or generic facilities such that one has to prove analogous facts twice, for example, about lower and upper semilattices.

Translations: It is planned to implement the mechanical translation of Mizar texts into other existing systems for doing mathematics, and vice versa. However, H. Barendregt's optimism on the time frame required for such a work is not commonly shared.

Large data base: A large data base would require a major effort from numerous parties and the administrative problems of such an enterprise should not be neglected. It is estimated that maintenance of a data base 10 times bigger than the current state (i.e. with about 3000 articles by several hundred authors) could stabilize a number of issues whose current solutions tend to be unstable.

Presentation: The usefulness of *Formalized Mathematics* containing typeset abstracts instigated some thoughts on typesetting entire MIZAR articles. Similarly, a need arises to develop some automated techniques for extracting topical monographs from the MIZAR library.

Accommodator: The process of preparing a local environment for checking a single article awaits a better solution. The problem here is similar to linking a newly written program with library modules, known to be a challenge in software engineering. An accommodator is expected to speed up the process of checking articles by cutting off the complex data base interaction at a certain level.

Neglected issues: There are some aspects of the MIZAR system that draw immediate criticism. To name a few: restriction to IBM PC and compatibles; poor user interface restricted to the text editor level; only textual searches of the data base; weak inference checker. All of them have been recognized as problems to work on but were perceived as second priority issues. Eventually, they have to be addressed.

Hibernation: Freezing the changes in the input language and in the MIZAR processor has been a goal for quite a while, yet it seems to move away like the horizon when you try to approach it.

9 How to learn Mizar?

The MIZAR language, its processor, and the organization of the MIZAR library evolve, and therefore there is not much in the way of written documentation, see [1].

In the face of documentation shortages the best way to learn MIZAR is to spend approximately four weeks in Białstok[6] and co-author a MIZAR article with a native user of the system. However, numerous cases are known of MIZAR users who that advanced their knowledge of the system by studying the existing texts (and there are 15MB of these).

Acknowledgements

I am indebted to all members of the MIZAR development group (which I was a member of years ago) and to all the authors of articles in MIZAR Library. Special thanks are to Andrzej Trybulec, Roman Matuszewski, Czesław Byliński, Edmund Woronowicz, Grzegorz Bancerek, and Zbigniew Karno.

This "commercial" has been written with the hope that their work meets with the recognition it deserves.

[6]Other places include: Lódź and Rzeszów in Poland, Madrid in Spain, and Nagano in Japan.

References

[1] Ewa Bonarska. *An Introduction to PC Mizar*. Mizar Users Group. Fondation Philippe le Hodey, Brussels, 1990.

[2] R. S. Boyer and J. S. Moore. *A Computational Logic Handbook*. Academic Press, 1988.

[3] R.L. Constable et al. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, 1986.

[4] N. G. de Bruijn. A survey of the project AUTOMATH. In J. P. Seldin and Hindley J. R., editors, *Essays in Combinatory Logic, Lambda Calculus, and Formalism*, pages 589–606. Academic Press, 1980.

[5] Andrzej Grzegorczyk. *Zarys arytmetyki teoretycznej*. PWN Warszawa, 1971.

[6] G. Huet and G. Plotkin, editors. *Proceedings of the 1st Workshop on Logical Frameworks*. ESPRIT BRA 3245, 1990. Anonymous ftp: `nuri.inria.fr`.

[7] G. Huet and G. Plotkin, editors. *Proceedings of the 2nd Workshop on Logical Frameworks*. ESPRIT BRA 3245, 1991. Anonymous ftp: `colonsay.dcs.ed.ac.uk`.

[8] S. Jaśkowski. On the rules of supposition in formal logic. *Studia Logica*, 1, 1934.

[9] H. C. Kennedy, editor. *Selected works of Giuseppe Peano*. University of Toronto Press, 1973.

[10] J. Ketonen. EKL—a mathematically oriented proof checker. In *Proceedings of 7th Int. Conf. on Automated Deduction*, pages 65–79, Napa, CA, May 1984.

[11] D. A. McAllester. *Ontic*. The MIT Press, 1989.

[12] S. McCall, editor. *Polish Logic in 1920–1939*. Clarendon Press, Oxford, 1967.

[13] M. Mostowski and Z. Trybulec. A certain experimental computer aided course of logic in Poland. In *Proceedings of World Conference on Computers in Education*, Norfolk, VA, 1985.

[14] Y. Nakamura. A language for description of mathematics—THEAX. Technical report, Shinshu University FIE, Nagano City, Japan, 1985. In Japanese.

[15] S. Nieva Soto. The Reasoner of MIZAR/LOG. *Computerized Logic Teaching Bulletin*, 2(1):22–35, March 1989.

[16] K. Ono. On a practical way of describing formal deductions. *Nagoya Mathematical Journal*, 21, 1962.

[17] Giuseppe Peano. *Opere Scelte*. Edizioni Cremonese, Roma, 1958.

[18] K. Prażmowski, P. Rudnicki, et al. Mizar-MSE Primer and User Guide. TR 88-9, The University of Alberta, Department of Computing Science, Edmonton, 1988.

[19] P. Rudnicki. Obvious inferences. *Journal of Automated reasoning*, 3:383–393, 1987.

[20] P. Rudnicki. What should be proved and tested symbolically in formal specifications? In *4th IEEE International Workshop on Software Specification and Design*, pages 190–195, Monterey, Ca., 1987.

[21] P. Rudnicki and W. Drabent. Proving properties of Pascal programs in MIZAR 2. *Acta Informatica*, 22:311–331, 1985. Erratum pp. 699–707.

[22] R.L. Smith et al. Computer-assisted axiomatic mathematics: Informal rigor. In O. Lecarme and R. Lewis, editors, *Computers in Education*, pages 803–809. North Holland, 1975.

[23] Alfred Tarski. Über unerreichbare Kardinalzahlen. *Fundamenta Mathematicae*, 30:68–89, 1938.

[24] Alfred Tarski. On well-ordered subsets of any set. *Fundamenta Mathematicae*, 32:176–183, 1939.

[25] A. Trybulec and H. Blair. Computer aided reasoning. In R. Parikh, editor, *Logic of Programs, LNCS 193*. Springer Verlag, 1985.

[26] A. Trybulec and H. Blair. Computer assisted reasoning with Mizar. In *Proceedings of the 9th IJCAI*, pages 26–28, Los Angeles, Ca., 1985.

[27] Andrzej Trybulec. Tarski Grothendieck Set Theory. *Formalized Mathematics*, 1:9–11, 1990.

[28] L. S. van Benthem Jutting. The development of a text in AUT-QE. In *Proceedings of APLASM'73, Symposium d'Orsay sur la Manipulation des Symboles at d'Utilisation d'APL*, Universite Paris XI, 1973.

A Tarski Grothendieck Set Theory

The following is the abstract and the actual article on which the MIZAR library is built. For obvious reasons the article has not been checked for validity. Unfortunately for the international audience the comments in text are mainly in Polish. First, the English summary provided by A. Trybulec (author).

> This is the first part of the axiomatics of the Mizar system. It includes the axioms of the Tarski Grothendieck set theory. They are: the axiom stating that everything is a set, the extensionality axiom, the definitional axiom of the singleton, the definitional axiom of the pair, the definitional axiom of the union of a family of sets, the definitional axiom of the boolean (the power set) of a set, the regularity axiom, the definitional axiom of the ordered pair, the Tarski's axiom A introduced in [23] (see also [24]), and the Frænkel scheme. Also, the definition of equinumerosity is introduced.

```
environ vocabulary EQUI_REL, BOOLE, FAM_OP;
  :: Andrzej Trybulec
  :: Teoria mnogosci Tarskiego Grothendiecka
begin
 reserve x,y,z,u for Any,
        N,M, X,Y,Z for set;
:: axiom  Tarski:1        ---- wszystko jest zbiorem
                    canceled; :: as obvious:  x is set;
axiom  :: Tarski:2        ---- ekstensjonalnosc zbiorow
                    (for x holds x ∈ X iff x ∈ Y) implies X = Y;
 :: Singletony i pary
                    definition
                      let y; func { y } -> set means
:: TARSKI: def 1
                       x ∈ it iff x = y;
                      let z; func { y, z } -> set means
:: TARSKI: def 2
                       x ∈ it iff x = y or x = z;
                    end;
axiom  :: Tarski:3        ---- definicja singletonu
                    X = { y } iff for x holds x ∈ X iff x = y;
axiom  :: Tarski:4        ---- definicja pary nieuporzadkowanej
                    X = { y,z } iff for x holds x ∈ X iff x = y or x = z;
                    definition let X,Y;
                     pred X c= Y means
:: TARSKI: def 3
                      x ∈ X implies x ∈ Y;
                      reflexivity;
                    end;
                    definition let X;
                       func union X -> set means
:: TARSKI: def 4
                        x ∈ it iff ex Y st x ∈ Y & Y ∈ X;
                    end;
axiom  :: Tarski:5        ---- definicja unii rodziny zbiorow
                    X = union Y iff for x holds x ∈ X iff ex Z st x ∈ Z & Z ∈ Y;
axiom  :: Tarski:6        ---- definicja zbioru potegowego
                    X = bool Y iff for Z holds Z ∈ X iff Z c= Y;
```

```
axiom  :: Tarski:7        ---- aksjomat regularnosci
                    x ∈ X implies ex Y st Y ∈ X & not ex x st x ∈ X & x ∈ Y;

 scheme Fraenkel { A()-> set, P[Any, Any] }:
  ex X st for x holds x ∈ X iff ex y st y ∈ A() & P[y,x]
  provided for x,y,z st P[x,y] & P[x,z] holds y = z;

                    definition let x,y;
                     func [x,y] means
:: TARSKI: def 5
                     it = { { x,y }, { x } };
                    end;
axiom  :: Tarski:8        ---- definicja pary uporzadkowanej
                    [ x,y ] = { { x,y }, { x } };
                    definition let X,Y;
                     pred X ≈ Y means
:: TARSKI: def 6
                      ex Z st
                            (for x st x ∈ X ex y st y ∈ Y & [x,y] ∈ Z) &
                            (for y st y ∈ Y ex x st x ∈ X & [x,y] ∈ Z) &
                            for x,y,z,u st [x,y] ∈ Z & [z,u] ∈ Z holds x = z iff y = u;
                    end;
:: Alfred Tarski
:: Ueber unerreichbare Kardinalzahlen,
:: Fundamenta Mathematicae, vol.30 (1938), pp.68-69
:: Axiom A. (Axiom der unerreichbaren Mengen). Zu jeder Menge N gibt es
:: eine Menge M mit folgenden Eigenschaften :
:: A1. N ∈ M;
:: A2. ist X ∈ M und Y c= X, so ist Y ∈ M;
:: A3. ist X ∈ M und ist Z die Menge, die alle Mengen Y c= X und keine
::     andere Dinge als Element enthaelt, so,ist z ∈ M;
:: A4. ist X c= M und sind dabei die Menge X und M nicht gleichmaechtig,
::     so ist X ∈ M.
:: takze
:: Alfred Tarski
:: On Well-ordered Subsets of any Set,
:: Fundamenta Mathematicae, vol.32 (1939), pp.176-183
:: A. For every set N there exists a system M of sets which satisfies
::    the following conditions :
:: (i)    N ∈ M
:: (ii)   if X ∈ M and Y c= X, then Y ∈ M
:: (iii)  if X ∈ M and Z is the system of all subsets of X, then Z ∈ M
:: (iv)   if X c= M and X and M do not have the same potency, then X ∈ M.

axiom :: Tarski:9
                      ex M st N ∈ M &
                        (for X,Y holds X ∈ M & Y c= X implies Y ∈ M) &
                        (for X holds X ∈ M implies bool X ∈ M) &
                        (for X holds X c= M implies X ≈ M or X ∈ M);
```

B Built-in Concepts

The English summary provided by A. Trybulec (author):

> This abstract contains the second part of the axiomatics of the Mizar system (the first part is in abstract [27]). The axioms listed here characterize the Mizar built-in concepts that are automatically attached to every Mizar article. We give definitional axioms of the following concepts: element, subset, Cartesian product, domain (non-empty subset), subdomain (non empty-subset of a domain), set domain (domain consisting of sets). Axioms of strong arithmetics of real numbers are also included.

Numerous axioms that were needed some time ago have been cancelled; they have been built into the processor and they are now obvious to the verifier. The trace of them is required to properly process older articles that made references to the axioms.

Polish comments have been deleted from the text.

```
 :: Andrzej Trybulec

environ
 vocabulary Boole;    signature Tarski;
begin
  reserve x,y,z for Any,
          X,X1,X2,X3,X4,Y for set;
:: axiom AXIOMS:1  (ex x st x ∈ X) implies (x is Element of X iff x ∈ X);
                   canceled; :: as obvious
axiom :: AXIOMS:2
                   X is Subset of Y iff X c= Y;
axiom :: AXIOMS:3
                   z ∈ [:X,Y:] iff ex x,y st x ∈ X & y ∈ Y & z = [x,y];
axiom :: AXIOMS:4
                   X is non-empty implies ex x st x ∈ X;
axiom :: AXIOMS:5
                   [: X1,X2,X3 :] = [:[:X1,X2:],X3:];
axiom :: AXIOMS:6
                   [: X1,X2,X3,X4 :] = [:[:X1,X2,X3:],X4:];
 reserve D1,D2,D3,D4 for non-empty set;
:: axiom AXIOMS:7  for X being Element of [: D1,D2 :] holds X is TUPLE of D1, D2;
                   canceled; :: as obvious
:: axiom AXIOMS:8  for X being Element of [: D1,D2,D3 :] holds X is TUPLE of D1, D2, D3;
                   canceled; :: as obvious
:: axiom AXIOMS:9  for X being Element of [: D1,D2,D3,D4 :] holds X is TUPLE of D1, D2, D3, D4;
                   canceled; :: as obvious
 reserve D for non-empty set;
axiom :: AXIOMS:10
                   D1 is non-empty Subset of D2 iff D1 c= D2;
:: axiom AXIOMS:11  D is SET DOMAIN;
                   canceled; :: as obvious
 reserve x,y,z for Element of REAL;
axiom :: AXIOMS:12
                   x + y = y + x;
axiom :: AXIOMS:13
                   x + (y + z) = (x + y) + z;
axiom :: AXIOMS:14
                   x + 0 = x;
axiom :: AXIOMS:15
```

```
                  x · y = y · x;
axiom :: AXIOMS:16
                  x · (y · z) = (x · y) · z;
axiom :: AXIOMS:17
                  x · 1 = x;
axiom :: AXIOMS:18
                  x · (y + z) = x · y + x · z;
axiom :: AXIOMS:19
                  ex y st x + y = 0;
axiom :: AXIOMS:20
                  x <> 0 implies ex y st x · y = 1;
axiom :: AXIOMS:21
                  x ≤ y & y ≤ x implies x = y;
axiom :: AXIOMS:22
                  x ≤ y & y ≤ z implies x ≤ z;
axiom :: AXIOMS:23
                  x ≤ y or y ≤ x;
axiom :: AXIOMS:24
                  x ≤ y implies x + z ≤ y + z;
axiom :: AXIOMS:25
                  x ≤ y & 0 ≤ z implies x · z ≤ y · z;
axiom :: AXIOMS:26
                  for X,Y being Subset of REAL st
                   (ex x st x ∈ X) & (ex x st x ∈ Y) &
                    for x,y st x ∈ X & y ∈ Y holds x ≤ y
                   ex z st
                    for x,y st x ∈ X & y ∈ Y holds x ≤ z & z ≤ y;
:: axiom AXIOMS:27  x is Real;
                  canceled; :: as obvious
axiom :: AXIOMS:28
                  x ∈ NAT implies x + 1 ∈ NAT;
axiom :: AXIOMS:29
                  for A being set of Real
                    st 0 ∈ A & for x st x ∈ A holds x + 1 ∈ A holds NAT c= A;
 reserve i,j,k for Nat;
axiom :: AXIOMS:30
                 k = { i: i<k };
```

C A Mizar article

The following is the text of an article from the Main Mizar Library. This article is unusual—it is shortest in the library. In order to decrease the number of text lines and with hope of improving readability I have manipulated the white space of the original submission. The characters of extended ASCII have been replaced by some symbols available in LaTeX. The MIZAR processor restricts the length of input lines to 80 characters. The text below violates this restriction for presentation purposes.

```
:: Alicia de la Cruz
:: Fix Point Theorem for Compact Spaces

environ
 vocabulary METRYKA, SFAMILY, POWER1, FINITE, SEQ1, SEQ2, SEQM, SUB_OP, REAL_1,
            TOPCON, PCOMPS, FUNC, TOP2, FAM_OP, BOOLE, FINITER2, ALI2, FUNC_REL;
 signature FINSET_1, METRIC_1, FUNCT_1, FUNCT_2, PRE_TOPC, POWER, BOOLE, FUNCOP_1,
           TARSKI, COMPTS_1, PCOMPS_1, SETFAM_1, TOPS_1, TOPS_2, SEQ_1, SEQ_2, SEQM_3,
           SUBSET_1, REAL_1, NAT_1;
 definitions COMPTS_1, TARSKI, TOPS_2, FUNCT_2;
 theorems METRIC_1, SUBSET_1, REAL_1, PCOMPS_1, REAL_2, COMPTS_1, POWER, BOOLE, SEQ_2,
          SEQ_4, SERIES_1, SEQM_3, AXIOMS, SETFAM_1, TARSKI, SEQ_1, PRE_TOPC, TOPS_1,
          SQUARE_1, DOMAIN_1, FUNCOP_1, ZFMISC_1;
 schemes SETFAM_1, SEQ_1, GROUP_4, FINSET_1, NAT_1;

begin
      reserve M for MetrSpace, x, y for Element of the carrier of M;
theorem VIT:
       for F being set st F is_finite & F <> ∅ &
                          for B, C being set st B ∈ F & C ∈ F holds B c= C or C c= B
            ex m being set st m ∈ F & for C being set st C ∈ F holds m c= C
 proof    pred P[set] means
          $1 <> ∅ implies ex m being set st m ∈ $1 & for C being set st C ∈ $1 holds m c= C;
  let F be set such that
   Z: F is_finite   and   Y: F <> ∅ and
   X: for B, C being set st B ∈ F & C ∈ F holds B c= C or C c= B;
   A: P[∅];
   B: now let x, B be set such that j0: x ∈ F & B c= F & P[B];
         now per cases;                          :: we have to prove P[B U{x}]
          case j: not ex y being set st y ∈ B & y c=x;
           assume B U {x} <> ∅;   take m = x;   x ∈ {x} by TARSKI:def 1;
           hence m ∈ B U {x} by BOOLE:8;   let C be set;   assume C ∈ B U {x};   then
            _1_: C∈B or C∈{x} by BOOLE:8;   then j1:C∈B or C=x by TARSKI:def 1;
            j2: not C c=x or C=x by j, TARSKI:def 1, _1_;   C ∈ F by j0, BOOLE:11, j1;
           hence m c= C by j0, X, j2;
          case ex y being set st y ∈ B & y c=x;   then consider y being set such that
            j5: y ∈ B & y c=x;   assume B U {x} <> ∅;   consider m being set such that
            j3:m ∈ B   and
            j4: for C being set st C ∈ B holds m c= C by j0, j5, BOOLE:def 1;
                m c= y by j4, j5;   then j6:m c= x by j5, BOOLE:29;
           take m;   thus m ∈ B U {x} by j3, BOOLE:8;
           let C be set;   assume C ∈ B U {x};   then C ∈ B or C ∈ {x} by BOOLE:8;
           hence m c= C by j4, j6, TARSKI:def 1;            end;
         hence P[B U {x}];        end;
     P[F] from Finite(Z, A, B);
  hence thesis by Y;  end;
```

```
definition   let M be MetrSpace;
 mode contraction of M -> Function of  the carrier of M, the carrier of M
          means :DD: ex L being Real st 0<L & L<1 &
                          for x, y being Point of M holds dist(it.x, it.y)≤L·dist(x, y);
 existence   proof   consider x being Point of M;
    (the carrier of M) --> x is Function of the carrier of M, the carrier of M   proof
     thus dom ((the carrier of M) --> x) = the carrier of M by FUNCOP_1:19;
        cc:rng ((the carrier of M) --> x) c= {x} by FUNCOP_1:19;
        {x} c= the carrier of M by ZFMISC_1:37;
     hence rng ((the carrier of M) --> x) c= the carrier of M by BOOLE:29, cc;     end;
    then reconsider f = (the carrier of M) --> x
                    as  Function of  the carrier of M, the carrier of M;
  take f,  1/2; 0<1;   hence aa: 0<1/2 & 1/2<1 by SEQ_2:3, SQUARE_1:3;
  let z, y being Point of M;   f.z=x & f.y=x by FUNCOP_1:13;
     then bb: dist(f.z, f.y) = 0 by METRIC_1:def 3;   dist(z, y)≥0 by METRIC_1:7;
  hence dist(f.z, f.y)≤(1/2)·dist(z, y) by bb, aa, REAL_2:121;      end;
end;

theorem   for f being contraction of M st TopSpaceMetr(M) is_compact
              ex c being Point of M st f.c =c &            :: exists a fix point
              for x being Point of M st f.x=x holds x=c    :: exactly 1 fix point
proof
 let f be contraction of M;   consider L being Real such that
  a1: 0<L & L<1 and
  a2: for x, y being Point of M holds dist(f.x, f.y)≤L·dist(x, y) by DD;
 assume a7: TopSpaceMetr(M) is_compact;   consider x0 being Point of M;   set a=dist(x0, f.x0);
  now assume a <> 0;
         consider F being Subset-Family of the carrier of TopSpaceMetr(M) such that
    kkk: for B being Subset of the carrier of TopSpaceMetr(M) holds B∈F iff
               ex n being Nat st B = { x where x is Point of M : dist(x, f.x) ≤ a·L to_power n }
                                          from SubFamEx;
    TopSpaceMetr(M) = TopStruct≪the carrier of M, Family_open_set(M)≫ by PCOMPS_1:def 6; then
    d5: the carrier of M = the carrier of TopSpaceMetr(M);
        set B = { x where x is Point of M : dist(x, f.x) ≤ a·L to_power (0+1) };
        B is Subset of the carrier of M from SubsetD;   then B ∈ F by kkk, d5;   then
    d1: F<>∅ by BOOLE:def 1;
    a8: F is_centered   proof   thus F <> ∅ by d1;
     let G be Subset-Family of the carrier of TopSpaceMetr(M) such that
      b1: G <> ∅   and   b2: G c= F   and   b3: G is_finite;
      for B, C being set st B ∈ G & C ∈ G holds B c= C or C c= B   proof
       let B, C be set ;   assume h0:B ∈ G & C ∈ G;   then h3: B ∈ F & C ∈ F by b2, BOOLE:11;
            B is Subset of the carrier of TopSpaceMetr(M) by SETFAM_1:43, h0;
            then consider n being Nat such that
        h4: B = { x where x is Point of M : dist(x, f.x) ≤ a·L to_power n } by kkk, h3;
            C is Subset of the carrier of TopSpaceMetr(M) by SETFAM_1:43, h0;
            then consider m being Nat such that
        h5: C = { x where x is Point of M : dist(x, f.x) ≤ a·L to_power m } by kkk, h3;
        lemma1: for n, m being Nat st n≤m holds L to_power m ≤ L to_power n   proof
         let n, m being Nat such that iii: n≤m;
          now per cases by iii, REAL_1:def 5;
           case n<m;   then L to_power n > L to_power m by POWER:45, a1;
            hence L to_power n ≥ L to_power m by REAL_1:def 5;
           case n=m;   hence L to_power n ≥ L to_power m;     end;
         hence thesis;         end;
        lemma2: for n, m being Nat st n≤m holds a·L to_power m ≤ a·L to_power n   proof
         let n, m being Nat such that iii: n≤m;
          now per cases;
           case c:  a=0; then a·L to_power m = 0 by REAL_1:20 .= a·L to_power n by REAL_1:20, c;
```

```
        hence a·L to_power m ≤ a·L to_power n;
       case a<>0;  cc:  a≥0 by METRIC_1:7;  L to_power m ≤ L to_power n by iii, lemma1;
        hence a·L to_power m ≤ a·L to_power n by cc, REAL_1:51;     end;
     hence thesis;         end;
    now per cases by AXIOMS:23;
     case i:n≤m;   thus C c= B   proof   let y be Any;   assume y ∈ C;
           then consider x being Point of M  such that
       C1: y = x   and   C2: dist(x, f.x) ≤ a·L to_power m by h5;
           a·L to_power m ≤ a·L to_power n by i, lemma2;    then
           dist(x, f.x) ≤ a·L to_power n by AXIOMS:22, C2;
       hence y ∈ B by C1, h4;   end;
     case ii:m≤n;   thus B c= C   proof   let y be Any;   assume y ∈ B;
           then consider x being Point of M  such that
       C1: y = x   and   C2: dist(x, f.x) ≤ a·L to_power n by h4;
           a·L to_power n ≤ a·L to_power m by ii, lemma2;   then
           dist(x, f.x) ≤ a·L to_power m by AXIOMS:22, C2;
       hence y ∈ C by C1, h5;   end;         end;
   hence B c= C or C c= B;        end;     then consider  m being set such that
  n1: m ∈ G   and   n2: for C being set st C ∈ G holds m c= C by VIT, b1, b3;
  n3: m c= meet G by SETFAM_1:6, b1, n2;   n4: m ∈ F by n1, BOOLE:11, b2;
  h5: m is Subset of the carrier of TopSpaceMetr(M) by SETFAM_1:43, n1;
  CC: L <> 0 by a1;   dist(x0,f.x0) = a·1 by REAL_1:7 .= a·L to_power 0 by POWER:29, CC;
      then x0 ∈ { x where x is Point of M : dist(x, f.x) ≤ a·L to_power 0 };   then
  P: { x where x is Point of M : dist(x,f.x) ≤ a·L to_power 0 } <> ∅ by BOOLE:def 1;
  P': for k being Nat st {x where x is Point of M : dist(x, f.x) ≤ a·L to_power k} <> ∅
                  holds {x where x is Point of M : dist(x,f.x) ≤ a·L to_power (k+1)} <> ∅
  proof let k be Nat; assume {x where x is Point of M : dist(x,f.x) ≤ a·L to_power k} <> ∅;
        then consider z being Any such that
    s1: z∈{ x where x is Point of M : dist(x, f.x) ≤ a·L to_power k } by BOOLE:def 1;
        consider y being Point of M such that y=z and s2: dist(y,f.y)≤ a·L to_power k by s1;
    L ≥ 0 by a1, REAL_1:def 5; then s4: L·dist(y,f.y) ≤ L·(a·L to_power k) by REAL_1:51, s2;
    s3: L·(a·L to_power k)  = L·a·L to_power k by AXIOMS:16  .= a·L·L to_power k by AXIOMS:15
         .= a·(L·L to_power k) by AXIOMS:16  .= a·(L to_power 1·L to_power k) by POWER:30
         .= a·(L to_power k·L to_power 1) by AXIOMS:15  .= a·L to_power (k+1) by POWER:32, a1;
        dist(f.y, f.(f.y)) ≤ L·dist(y, f.y) by a2;   then
        dist(f.y, f.(f.y)) ≤ a·L to_power (k+1) by s3, s4, AXIOMS:22;   then
        f.y ∈ { x where x is Point of M : dist(x, f.x) ≤ a·L to_power (k+1) };
   hence { x where x is Point of M : dist(x,f.x)≤a·L to_power (k+1) } <> ∅ by BOOLE:def 1;
  end;
  for k being Nat holds { x where x is Point of M : dist(x,f.x) ≤ a·L to_power k} <> ∅
                               from Ind(P, P');   then  m <> ∅ by h5, kkk, n4;
 hence  meet G <> ∅ by BOOLE:30, n3;         end;
F is_closed   proof   let B being Subset of the carrier of TopSpaceMetr(M);
 assume B ∈ F;   then consider n being Nat such that
  mm:  B= { x where x is Point of M : dist(x, f.x) ≤ a·L to_power n } by kkk;
  A1: TopSpaceMetr(M)=TopStruct≪the carrier of M, Family_open_set(M)≫ by PCOMPS_1: def 6;
      then reconsider V = B` as Subset of the carrier of M;    set B' = B`;
  for x being Point of M st x ∈ V ex r being Real st r>0 & Ball(x, r) c= V   proof
   let x be Point of M such that m1: x ∈ V;   take r = (dist(x, f.x)-a·L to_power n)/2;
        2<>0;   then 2·r = dist(x, f.x)-a·L to_power n by REAL_2:73;   then
    m20: dist(x,f.x)-2·r = a·L to_power n by REAL_2:12; not x∈B by m1, SUBSET_1:53; then
        dist(x,f.x)>a·L to_power n by mm; then dist(x,f.x)-a·L to_power n>0 by REAL_2:108;
   hence r>0 by SEQ_2:3;
   let z be Any;   assume m: z ∈ Ball(x, r);   Ball(x, r) c= the carrier of M by AXIOMS:2;
    then reconsider y=z as Point of M by BOOLE:11, m;   m8: dist(x, y)<r by METRIC_1:19, m;
         dist(x, y) + dist(y, f.y) ≥ dist(x, f.y) by METRIC_1:6;   then
    m13: (dist(x,y)+dist(y,f.y))+dist(f.y,f.x) ≥ dist(x,f.y)+dist(f.y,f.x) by REAL_1:53;
         dist(x, f.y) + dist(f.y, f.x) ≥ dist(x, f.x) by METRIC_1:6;   then
```

```
          dist(x, y)+dist(y, f.y)+dist(f.y, f.x)≥dist(x, f.x) by m13, AXIOMS:22;   then
          dist(y, f.y)+dist(x, y)+dist(f.y, f.x)≥dist(x, f.x) by AXIOMS:12;   then
     m6:  dist(y, f.y)+(dist(x, y)+dist(f.y, f.x))≥dist(x, f.x) by AXIOMS:13;
     m10: dist(x, y) = dist(y, x) by METRIC_1:5;   m3: dist(f.y, f.x)≤L·dist(y, x) by a2;
          dist(y, x)≥0 by METRIC_1:7;   then L·dist(y, x)≤dist(y, x) by REAL_2:147, a1;
          then dist(f.y, f.x)≤dist(y, x) by m3, AXIOMS:22;   then
     m4:  dist(f.y, f.x)+dist(y, x) ≤ dist(y, x)+dist(y, x) by REAL_1:49;
          2>0;   then 2·dist(x, y)<2·r by m8, REAL_1:70;   then
     m9:  dist(y, f.y) + 2·dist(x, y)< dist(y, f.y) + 2·r by REAL_1:59;
          dist(f.y, f.x)+dist(y, x) ≤ 2·dist(y, x) by m4, SQUARE_1:5;   then
          dist(y, x) + dist(f.y, f.x) ≤ 2·dist(y, x) by AXIOMS:12;   then
          dist(y,f.y)+(dist(y,x)+dist(f.y,f.x))≤dist(y,f.y)+2·dist(y,x) by REAL_2:103;
          then dist(y, f.y)+2·dist(x, y)≥dist(x, f.x) by m6, AXIOMS:22, m10;
          then dist(y, f.y)+2·r>dist(x, f.x) by REAL_1:58, m9;   then
          not ex x being Point of M st y=x & dist(x,f.x)≤a·L to_power n by m20, REAL_1:91;
         then m22: not y ∈ B by mm;   the carrier of M <> ∅ by DOMAIN_1:1;
    hence z ∈ V by SUBSET_1:50, m22, A1;       end;   then
   B‘ ∈ Family_open_set(M) by PCOMPS_1:def 5;   then B’ is_open by A1,  PRE_TOPC:def 4;
  hence B is_closed by TOPS_1:29;
 end;   then
 meet F<>∅ by COMPTS_1:13,a8,a7; then consider c’ being Point of TopSpaceMetr(M) such that
 d2: c’∈meet F by SUBSET_1:10;
     reconsider c = c’ as Point of M by d5;   consider s’ being Real_Sequence such that
 b3: for n being Nat holds s’.n=L to_power (n+1) from ExRealSeq;    set s = a □ s’;
 b6’’: s’ is_convergent & lim s’=0 by a1, SERIES_1:1, b3;   then
 b6: s is_convergent by SEQ_2:21;
 b6’: lim s = a·0  by b6’’, SEQ_2:22 .= 0 by REAL_1:20;
     consider r being Real_Sequence such that
 b4: for n being Nat holds r.n=dist(c, f.c) from ExRealSeq;
 b5: r is_constant by SEQM_3: def 5, b4; then b7: r is_convergent by SEQ_4:39;
 now let n be Nat;   set B = { x where x is Point of M : dist(x, f.x) ≤ a·L to_power (n+1)};
   B is Subset of the carrier of M from SubsetD;   then B ∈ F by kkk, d5;   then
   c∈ B by d1, SETFAM_1:1, d2;   then
   d3: ex x being Point of M st c = x & dist(x, f.x) ≤ a·L to_power (n+1);
   d3’: r.n = dist(c, f.c) by b4;   s.n = a·s’.n by SEQ_1:def 5 .= a·L to_power (n+1) by b3;
  hence r.n ≤ s.n by d3, d3’;       end;   then
 c1: lim r ≤ lim s by SEQ_2:32, b6, b7;   r.0=dist(c, f.c) by b4;   then
     dist(c, f.c)≤0 & dist(c, f.c)≥0 by c1, b6’, METRIC_1:7, SEQ_4:40, b5;   then
     dist(c, f.c)=0 by AXIOMS:21;
hence ex c being Point of M st dist(c, f.c) = 0;
end;   then consider c being Point of M such that
XX: dist(c, f.c) = 0;
take c;
thus a4: f.c =c by METRIC_1: def 3, XX;
let x be Point of M ; assume a3: f.x=x; assume x<>c; then a6: dist(x,c)<>0 by METRIC_1:def 3;
 dist(x, c)≥0 by METRIC_1:7;   then dist(x, c)>0 by REAL_1:def 5, a6;   then
 L·dist(x, c)<dist(x, c) by a1, REAL_2:145;
hence contradiction by a3, a4, a2;
end;
```

D A Mizar abstract

Here is the abstract of the article presented in Appendix C. The abstracts are extracted mechanically from the submitted articles and the reference identification of MIZAR items is automatically inserted. If one wants to make a reference to another article, they have to use reference names as presented in the article's abstract.

```
:: Alicia de la Cruz
:: Fix Point Theorem for Compact Spaces

environ

 vocabulary METRYKA,SFAMILY,POWER1,FINITE,SEQ1,SEQ2,SEQM,SUB_OP,REAL_1,TOPCON,
     PCOMPS,FUNC,TOP2,FAM_OP,BOOLE,FINITER2,ALI2,FUNC_REL;
 signature FINSET_1,METRIC_1,FUNCT_1,FUNCT_2,PRE_TOPC,POWER,BOOLE,FUNCOP_1,
     TARSKI,COMPTS_1,PCOMPS_1,SETFAM_1,TOPS_1,TOPS_2,SEQ_1,SEQ_2,SEQM_3,
     SUBSET_1,REAL_1,NAT_1;

begin
  reserve M for MetrSpace,x,y for Element of the carrier of M;

theorem :: ALI2:1
for F being set st F is_finite & F <> ∅ &
 for B,C being set st B ∈ F & C ∈ F holds B c= C or C c= B
 ex m being set st m ∈ F & for C being set st C ∈ F holds m c= C;

definition  let M be MetrSpace;
mode contraction of M -> Function of  the carrier of M , the carrier of M
means :: ALI2:def 1
ex L being Real st 0<L & L<1& for x,y being Point of M holds
dist(it.x,it.y)≤L·dist(x,y);
end;

theorem :: ALI2:2
for f being contraction of M st TopSpaceMetr(M) is_compact
ex c being Point of M st f.c =c &           :: exists a fix point
for x being Point of M st f.x=x holds x=c;
```

E A TEXed Mizar abstract

MIZAR abstracts are further mechanically processed, typeset using TEX, and published (see footnote on page 6). Each author of an article accepted to the library must provide a title, the author's name and address, and a summary (in English) that are included into the typeset abstract.

The next two pages contain a copy of the typeset version of the abstract from Appendix D One can consider the English of the text rather poor, but it should be remembered that this text has been generated mechanically. The process of mechanical translation from MIZAR into English is still being worked on.

FORMALIZED MATHEMATICS
Vol.2, No.4, September–October 1991
Université Catholique de Louvain

Fix Point Theorem for Compact Spaces

Alicia de la Cruz
Universidad Politecnica de Madrid

Summary. The Banach theorem in a compact metric spaces is proved.

MML Identifier: `ALI2`.

The terminology and notation used in this paper have been introduced in the following papers: [9], [15], [3], [4], [8], [11], [13], [9], [11], [5], [7], [18], [6], [17], [1], [2], [6], [4], and [5]. In the sequel M will be a metric space. Next we state the proposition

(1) For every set F such that F is finite and $F \neq \emptyset$ and for all sets B, C such that $B \in F$ and $C \in F$ holds $B \subseteq C$ or $C \subseteq B$ there exists a set m such that $m \in F$ and for every set C such that $C \in F$ holds $m \subseteq C$.

Let M be a metric space. A function from the carrier of M into the carrier of M is said to be a contraction of M if:

(Def.1) there exists a real number L such that $0 < L$ and $L < 1$ and for all points x, y of M holds $\rho(\mathrm{it}(x), \mathrm{it}(y)) \leq L \cdot \rho(x, y)$.

Next we state the proposition

(2) For every contraction f of M such that M_{top} is compact there exists a point c of M such that $f(c) = c$ and for every point x of M such that $f(x) = x$ holds $x = c$.

References

[1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.
[2] Leszek Borys. Paracompact and metrizable spaces. *Formalized Mathematics*, 2(**4**):481–485, 1991.
[3] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[4] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.

ISSN 0777–4028

[5] Agata Darmochwał. Compact spaces. *Formalized Mathematics*, 1(**2**):383–386, 1990.

[6] Agata Darmochwał. Families of subsets, subspaces and mappings in topological spaces. *Formalized Mathematics*, 1(**2**):257–261, 1990.

[7] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.

[8] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.

[9] Stanisława Kanas, Adam Lecko, and Mariusz Startek. Metric spaces. *Formalized Mathematics*, 1(**3**):607–610, 1990.

[10] Jarosław Kotowicz. Convergent sequences and the limit of sequences. *Formalized Mathematics*, 1(**2**):273–275, 1990.

[11] Jarosław Kotowicz. Monotone real sequences. Subsequences. *Formalized Mathematics*, 1(**3**):471–475, 1990.

[12] Jarosław Kotowicz. Real sequences and basic operations on them. *Formalized Mathematics*, 1(**2**):269–272, 1990.

[13] Beata Padlewska. Families of sets. *Formalized Mathematics*, 1(**1**):147–152, 1990.

[14] Beata Padlewska and Agata Darmochwał. Topological spaces and continuous functions. *Formalized Mathematics*, 1(**1**):223–230, 1990.

[15] Konrad Raczkowski and Andrzej Nędzusiak. Real exponents and logarithms. *Formalized Mathematics*, 2(**2**):213–216, 1991.

[16] Andrzej Trybulec. Binary operations applied to functions. *Formalized Mathematics*, 1(**2**):329–334, 1990.

[17] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.

[18] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.

[19] Mirosław Wysocki and Agata Darmochwał. Subsets of topological spaces. *Formalized Mathematics*, 1(**1**):231–237, 1990.

Received July 17, 1990