STUDIES IN LOGIC, GRAMMAR AND RHETORIC 4 (17) 2001

Roman Matuszewski
University of Białystok

FORMAL MATHEMATICAL TEXTS. TOWARDS THEIR RENDERING INTO NATURAL LANGUAGE*

1. Introduction

Formal proof systems have long been studied as part of mathematical logic, especially proof checker systems were originally intended for the actual use in carrying out standard mathematical texts. From [SimSin97]:

> "With the advent of powerful and sophisticated implementations of logics, formal reasoning in general, and formal proofs in particular, are becoming relevant and accessible to other fields that use and rely on mathematical reasoning techniques".

Though the use of formalisms enables formal proofs to be written by a human and checked with the help of a computer, such articles have one serious deficiency: they are overburdened by large amounts of technical detail of the underlying formal systems. From [SimSin97]:

> "This formal view obscures the basic line of reasoning and hinders human comprehension. There is evidently a wide gap, then, between formal texts and conventional mathematical proofs, whose essential purpose, in addition to establishing the truth of propositions, is to provide insight and understanding. One may argue that it is not worthwhile trying to understand a formal proof at all, once it has been machine-checked for correctness. This is certainly the case where proofs are technical and tedious, and fail to offer any insights, and we would be happy to leave the verification of such arguments to the machine".

But in other cases, a formal mathematical text, just as its informal counterpart, carries important information that we would like to communicate.

* This research was partially supported by two European Community 5FP grants: CALCULEMUS (HPRN-CT-2000-00102) and TYPES (IST-1999-29001).

ISBN 83-89031-06-X ISSN 0860-150X

Traditionally, research in the area of computer aided formalization of mathematics focussed on how the computer can help in the process of constructing correct, mechanically checked mathematical articles. Consequently, the research concentrated on building a library of articles. The largest, developed since 1989, is the Mizar Mathematical Library [RudTry99]. In the last few years there has been a growing interest in reading and using proof checked articles, by authors of formalized articles and by the wider audience through the internet. Understanding and then practical application of articles depends strongly on their presentation. From [SimSin97]:

> Language and notation are used to explain the reasoning, intuition, association and stylistical paraphrasing may be used to help the reader. We therefore propose a planning approach to the presentation of Mizar proofs that integrates a formal proof and its explanation into a single document. The support of mathematical notation deserves special attention. Well designed notation, especially in TEX, plays an important role in the communication of mathematical understanding.

In the research we propose an approach to formal text presentation that attempts to combine formality with comprehensibility. This approach is guided by an analogy relating the activities of proving and programming. Although proving in Mizar is a declarative process, programming is a procedural process. From [SimSin97]:

> "Developing a program from a specification is very much like developing a proof for a theorem. By following this analogy, we apply techniques and principles known from program design to proof design and presentation. Most important, we apply the principle of refinement to proofs. Refinement has been used as both an informal and a formal abstraction principle to control complexity and to structure and guide the process of programming. By transferring the refinement paradigm to formal proof design, we will show how we arrive at *formal* and *hierarchically structured* texts, which are presented at different levels of abstraction."

The upper levels indicate how complete formal text can be constructed, and they carry the essential information that constitutes the basic line of reasoning. The lower levels fill in the necessary technical detail, a task that can be left entirely up to the machine. Thus, the choice of the most appropriate level of abstraction depends on the difficulty of the proof, and of those for whom the presentation is intended (education, report of the database, journal, etc.). Its depends also on the mechanical capabilities of the underlying reasoning system, which besides proof checking of the formal text must also maintain and verify the database of mathematical knowledge.

In this article we will show, how based on a proof plan, a structured presentation at the level of proof methods can be investigeted. From [Hua94]:

> "Mainly two kinds of knowledge are incorporated into the content planning in the form of presentation of proofs. The *hierarchical planning* split the task of presenting a particular proof into subtask of presenting subproofs. A *Local navigation* operators simulates the unplanned aspect, where the next conclusion to be presented is chosen under guidance of a local focus mechanism".

From the other side we have to investigate textbooks [BelSlo69, Gra79, HarWri79, KurMos76, Lan80] proofs consists the *structural* and *logical* information, which is contained in the proof. This includes the identification of:

- sequences of sentences that constitute subproofs,
- the contribution of each of the proof segments to the overall goal to decrease the proof obligations,
- the scope and quantification of the variables,
- the logical/structural relations between sentences/segments.

The very important principle in our approach is, that investigated formal mathematical text is logically and mathematically correct (proof checked). It is important to note, that this assumption give guaranty, that text will be coherent.

The readability of a proof depends on the effort required by the reader to understand it. Therefore, in order to be readable, a proof should contain the necessary information to be followed without undue effort. It should also omit irrelevant information, or any information, which can be easily deduced by the intended reader of the proof. Furthermore, in order to facilitate its readability, the information contained in a proof should be organized in a way, which highlights its structure.

Presenting a formal mathematical text may be seen as an attractive idea, particularly since formal arguments tend to be bogged down by technical details. These circumstantiality usually hides the basic line of reasoning underlying a proof, but is necessary to enable a computer to check the correctness of an argument. Strictly formal proofs contain too much technical detail, which is of no interest to the human reader, who only wishes to understand *the basic idea*. This results in a long, overly detailed proof in which the basic line of reasoning is obscured. The representation of a formal text is geared towards a form that is easy to parse for the computer, which differs from the form a human would choose in order to understand it. This results in a lack of structural information. Both above combined result in superfluous information on the one hand and the lack of helpful information on the other hand. But still, such a proof contains a representation of the basic proof idea that was on the mind of the person who

has written the *Mizar article*. Thus, by hiding the unnecessary information and by providing additional information it should be possible to recover the proof idea.

The intelligible presentation of formal proofs is usually not attempted because of their technical detail. From [SimSin97]:

> "We want to separate the discussion of the requirements into two areas: what features are necessary to present formal reasoning in a structured and natural language way, and how can the whole system be kept flexible, i.e., applicable for various instances of formal reasoning. We will keep the discussion at a rather abstract level so that the software architecture of the supports system becomes visible".

Another question is representation of the graphical text. The fact, that the formulas are displayed in the severely restricted ASCII character set doesn't add to the comprehensibility either. Thus we take this representation merely as a basis to derive step by step proof document that is independent from the syntax of the system, well structured, and oriented at common proving styles. Afterwards, a TEX, or HTML, document can be generated, where all the operators, constants, and so on are replaced by their appropriate mathematical symbols.

From [Hua94]::

> "In contrast to the belief that mathematical texts are only schematic and mechanical, *state of the art* techniques of natural language processing are necessary to produce coherent texts that resemble those found in typical mathematical textbooks".

The human proof presentation process is based on the natural language generation techniques. On one side we have a formal text as input, and as output we will have a text in natural language. Traditionally, the generation process has been divided into two stages:

- ***what to say***, and
- ***how to say it***.

The first stage comprises of processing from the concept and intermediate representation of the formal proof to the planning of contents. The second covers realization of the plans into text or output in other modalities.

The planning of the contents could be developed through the research of the Discourse Representation Theory [GroSid90]. While the realization of the plans – through the Rhetorical Structure Theory [ManTho87].

2. Current Situation in the Related Research

Over the past thirty years there have been significant achievements in the field of automated theorem proving. Although some effort has also been made to fields of artificial intelligence, such as natural language generation and user modeling. In particular, no model is available which accounts both for human deductive activities and for human proof presentation. In this thesis, a reconstructive approach is suggested which substantially abstracts, reorganizes and finally translates machine checked proofs into a natural language. Both the procedures and the intermediate representations of our architecture find their basis in the Discourse Representation Theory and in the Rhetorical Structure Theory for informal mathematical reasoning and for proof presentation. User modeling is not incorporated into the current theory, although we plan to do so later.

The need for better outputs of formal texts was recognized some years ago of the natural deduction calculus and otherwise readable proofs.

The system EXPOUND [Che76] was pioneering presentation of proofs in natural language, in 1976. It is an example of *direct translation*. Although a sophisticated linearization is applied on the input natural deduction proofs, the steps are then translated locally in a template driven way.

Proof presentation in natural language has recently been realized in ILF [Dah94] and PROVERB [Hua94], which slightly abstract proofs before the presentation:

- ILF provides a schematic verbalization, not intelligent natural language generation but merely the application of templates. Each logical rule, as well as each of the various manifestation of reasoning rules, has a template associated with it.
- PROVERB returns a more elaborate proof presentation at the so-called assertion level that employs linguistic knowledge in order to combine single assertion level steps. However a proof verbalization at the assertion level is not necessarily the most natural and best way to communicate a proof to mathematicians or to students. In particular, often the proof is not abstract enough and the user cannot go from an abstract level to a more detailed level because a hierarchical structure is missing.

Last years research was done also by:

- Yann Coscoy [Cos94] – From [SimSin97]:

  "describe how a pseudo natural language proof description can be extracted auto-

matically from proof objects, which are in their case - terms of the Calculus of Construction of COQ, encoding a natural deduction proof. They use techniques that are similar to the ones employed to extract programs from proofs. Questions such as hierarchical structuring are not addressed, nor is the treatment of special proof styles. This technique would therefore appear - thus far - to be usable for small proofs only".

- Martin Simons [SimSin97] – literate and structured presentation of formal proofs of Deva language.
- Sh. Kobayashi, Y. Nakamura and Y. Fuwa [KobNakFuw97] – developed automatic translation from Japanese into English, using one intermediary language INE (Internet New Esperanto). This approach was based on translation into Function Format Language (predicate calculus), with help of specialized templates (devised by Writing Aid).

Concerning automatic translation of Mizar texts into natural language, besides research noted in [BanCar93], there was also one attempt to translation of Mizar-MSE texts into Chinese by Bin Qin [Qin84]. Similar approach as in [Mat89] was made by P. Rudnicki and A. Trybulec [RudTry89] – *A Collection of TEX-ed Mizar Abstracts*, but mostly on the typesetting level.

3. Mathematical Vernacular

The term *Mathematical Vernacular* has been used with varying meaning, e.g. in [deB87] – it is a mixture of words and formulas that mathematicians speak and write. It is a language, which is suitable for ordinary mathematical practice, and which can be implemented on the computer under the guidance of formal semantics. But more precisely, from [ZhaCall99]:

"we mean that a mathematical and natural language which is suitable for developing mathematics, has formal semantics, and is implementable for interactive mathematical development based on the technology of a computer assisted formal reasoning and natural language processing".

In this language logical content is a prominent part of meaning.

From [TrybBla85]:

"Mathematical vernacular is characterized in part by an open system of standardized notation. A writer of mathematics is not free to fill his text with an undisciplined growing of freely invented notation. If a standard notation is adequate for the purpose, the author is well advised to use it. Only the rarest circumstances permit a relaxation of this practice. Nevertheless, excessive formalism should be avoided since it invites a level of de-

tail simply too distracting, indeed boring, from the main point of the argument. The standardization of notation permits the possibility of a formal reconstruction of mathematical vernacular. The requirement that the mathematical presentation be not too formally detailed, but nevertheless clear in a step wise style, permits the use of some aspects of automated reasoning in a reasoning assistance system. A single human oriented step in a mathematical argument is viewed as a small, quickly solvable, automated reasoning task. The ultimate success and value of such system is determined by how useful a tool it proves to be in practice and not by how well it is alleged to embody a particular teaching style".

From [deB87]:

"The reason is, that mathematicians define the meaning of their words and sentences locally, without having to compare them to the existing habits in the outside world. On the one hand, the grammar of mathematical vernacular is much simpler than the one of natural languages. The rules of mathematical vernacular are expressed in terms of three grammatical categories:

- sentence,
- name,
- substantive.

On the other hand, it is more complex, since the correctness of mathematical statements depends on the context and on everything said before".

From [ZhaCall99]:

"The next question is the conceptual structure of mathematical language, which depends on the notion of conceptual category. An important issue in this analysis is that of well-formedness and meaningfulness of expressions in mathematical vernacular. Mathematicians attach importance to the criterion of semantic well-formedness, as well as to grammatical well-formedness. Conceptual categories play an important role not only in the correctness of checking (i.e. deciding whether an expression or a sentence is well-formed and meaningful) but also in capturing the generative nature of conceptual composition in mathematical vernacular. To mechanize any aspect of mathematics, we need a good formal understanding of the language".

As mentioned above, there is quite a lot of interesting and novel problems attached to mathematical vernacular. Given the prime need for correctness in implementing mathematical vernacular. From [ZhaCall99]:

We believe it is necessary to identify its successful parts, together with good practice suggested by our experience of formal mathematics, and fully formalize that, rather than attempt to formalize "all" of mathematics, a concept which we find hard to de-

> fine precisely. Naturally, the language should be as close as possible to good mathematical language, in the sense of exposing the richness of it, without having too many restrictions. Our view is, that mathematical vernacular should be developed by formalizing the essential *core* of the mathematical language, without which no useful mathematics may be practiced - even if the means of expression are cumbersome, then by extending this core to make the language more flexible without losing the formal properties.

To achieve better `meta-variable' facilities, we have identified an aspect, which will allow the user to omit parts of his proofs temporarily -- such as details he considers trivial. This `feedback' of ideas can also occur in informal mathematical language: studying it in order to create mathematical vernacular will help us to identify good parts and bad parts, showing a way of improvements in mathematical language. Thus, we can regard development of mathematical vernacular as a constraint satisfaction problem – mathematical vernacular is between mathematics and its realization in today's mathematical language.

Another important question is, whether the current set of vernacular is sufficient to formalize all proofs in a mathematical style. Some existing vernaculars (Coq [Dow90] and LEGO [Luo89, CalLuo98]) have one disadvantage - the inability to express a proof in a traditionally mathematical style. Only Mizar and based on it declarative languages seems to be sufficiently close to today's mathematical language.

4. Formal Proof

In Webster's dictionary, a proof is "the process or an instance of establishing the validity of a statement esp. by derivation from other statements in accordance with principles of reasoning". To put it more succinctly, a proof yields evidence. An informal proof provides readers with sufficient intuitive evidence to convince them of the validity of the statement that is to be proven. Human understanding of why the theorem is true is achieved by explaining the line of reasoning underlying the proof. A formal proof, on the other hand, provides evidence by reducing every aspect of the preceding definition to the level of symbols and their precise formal syntactic manipulation within a well defined and sound logical framework. The sole purpose of a formal proof is to establish the validity of a statement by mechanical — not necessarily automatical — deduction from a given fixed set of axioms, and possibly a set of hypotheses. The process of actually proving a theorem, establishing the validity of a concrete

statement, is insignificant with respect to providing insight. The result of the process is what counts: the truth of the theorem is either established or not. Only at the meta level is the process relevant again: it is formally expressed, proven sound and possibly complete, and it is the object of further study, e.g. of proof theory or research investigating possible automation. By disregarding the specific process during which a proof is successfully produced, and by focusing exclusively on the mechanical establishment of truth, a formal proof ignores — or rather abstracts form — all other aspects. Moreover, formal proofs, which are intended to be checked by a machine, are by their very nature overburdened with so much technical detail that any line of reasoning that might have existed when the proof was originally conceived is completely hidden.

There are two main approaches towards the formalization of proofs to enable automatic verification. One of them are theorem provers – they interactively seek a proof of a certain theorem, at the same time guaranteeing that the constructed proof is correct. The internal result of such proof is not readable for a human. It is like an internal language of programming; very “close” for computer, “far” for humans.

Formal proofs in general, and formal proofs arising during formal system development in particular, tend to be of a very technical and shallow nature. With tedious theorems, we are really only interested in whether they are true or not. We are only too glad to leave their actual proof to a machine. It is clear that a formal proof as such cannot mirror the cognitive and intellectual aspects of a proof and its presentation. A formal proof is a game with symbols, by its very nature devoid of aids to human understanding. If, then, we are to make formal proofs, i.e. proofs that are machine checkable, not only intelligible and manageable, but also ultimately as useful to humans as informal proofs, we must provide mechanisms and facilities that address these human oriented issues.

In [Har97] Harrison describes several different uses of the word "proof" in the field of automated reasoning. Three of these are of interest here:

- “a proof as found in a mathematical text book, i.e. a sketch given in a mixture of natural, symbolic and formal languages, sufficient to convince the reader,
- a script to be presented to a machine for checking. This may be just a sketch, or a program, which describes the syntactic manipulations, needed to construct a formal proof,
- a formal "fully expansive" proof in a particular formal system, e.g. a derivation

tree of inference rules and axioms".

From the practical point of view we can say, that mathematics is the language of mathematicians, and a proof is a method of communicating a mathematical truth to another person, who also 'speaks' the language. Our general problem is, how to communicate a proof to a machine. We use the word "proof" in the second listed above sense, and "proof outline" to mean proofs (again in the second sense) that are merely sketches, and that require significant reasoning to fill in gaps. Proofs in Mizar are expressed as proof outlines, in a language that approximates written mathematics. Therefore, the user must write a rigorous, i.e., completely formalized proof, that he believes represents the intent of the author of the textbook proof, and use the computer to check this rigorous proof.

Rigorous and formal proofs

The difference between *rigorous* and *formal* proofs is not easy, because the English meaning for both terms refers to the same. We can say, that *rigorous* proof is written in a formal language above inference level, independent of some calculus. This kind of proof interest is not on the individual, step by step, inferences of the proof — the application of individual inference rules — but, rather, on the main ideas of the proof. A rigorous proof relies to some extent on the reader's ability to judge its correctness, and the reader is certain that steps are correct, with no gaps or omissions. Then, a Mizar proof seems to be a rigorous proof.

In metamathematics, rigorous arguments showing how various pieces of rigorous mathematics can be codified in the predicate calculus. Textbooks of axiomatic set theory, which state the axioms of ZFC and then sketch how to introduce various mathematical concepts such as the real number system in (definitional extensions of) ZFC and prove standard theorems about such cocepts, all on the basis of the axioms.

The *formal* proof, in this comparison, is written in a language at inference level and the language depends only on the calculi used. The reader is able to expand a proof into primitive rules and it is constructed in conformance with a set of precisely defined and mechanically checkable rules.

The advantage of a rigorous proof is that a human can concentrate on formulating the important steps of a proof and not be diverted by technical details of how

these steps must actually be performed. The notion of rigorous proof is related to the human activity of ***planning*** when proving a theorem. We will apply this activity in our presentation of the proof, see [Mat99].

Formal proofs vs. informal proofs

Our work in analyzing chosen mathematical textbooks: [BelSlo80], [Gra79], [HarWri79], [KurMos76], [Lan80] and Mizar texts shows, that the major difference between formal and informal proofs is the level of detail between the two. From [Zam99]:

> "Informal proofs contain gaps in their reasoning that the reader is required to fill in order to understand the proof. The author of an informal proof usually has a specific type of reader in mind, one who has a certain amount of knowledge in a number of mathematical fields, and one who has read and understood the preceding sections of the literature containing the proof. The author can therefore rely on his, usually justified, assumptions about what the intended reader is able to understand when deciding what to include in an informal proof and what can be easily inferred by the reader, and can (or must) therefore be unjustified. For example, if one assumes that some set A is a subset of B, and that some element a is a member of A, then the inference which derives the membership of a in B can usually be omitted if the reader is assumed to be familiar with the notions of set membership and containment".

This is dependings, of course, on the power of the proof checker. V. Zammit in [Zam99] proposes to reinforce the proof checker by facts of trivial knowledge (in Mizar we call them – requirements), which increase the speed of checking and which have been derived much earlier in the mechanization. Case studies have shown, that the length of the proofs can be substantially reduced through the use of a much more powerful proof checker.

Besides our approach, there are also results reported in [Zin98]. It is interesting to compare informal proofs taken from the above mentioned textbooks, with formal proofs:

- A formal proof is written in a formal language. An informal proof is written in an informal language, say English, enriched with terms and formulae. The syntactic constructions one encounters in proofs are relatively easy and stylized.
- Finding proofs (informal and formal) is not trivial.
- Machines are good at verifying formal proofs, but (currently) cannot check informal proofs. Humans are good at verifying informal proofs, and (will hopefully forever remain) bad in verifying formal proofs.

- In a formal proof, for each proof step, it is explicitly given which conclusion is derivable by which set of premises and by which inference rule. In an informal proof, many proof steps are omitted or incomplete (incomplete set of premises, and lack of reference to inference rules used).
- In formal proofs there is no ambiguity. In informal proofs there should be no ambiguity, but there is.
- In a formal proof, the theory in which the proof is stated is explicitly stated. The theory, and nothing else, defines the context. In an informal proof, there is no full and explicit theory that one can refer to. The context is to be completed by the proof reader.
- In a formal proof, form matters. In an informal proof, meaning matters.
- Formal proofs are structured (e.g., resolution graphs, natural deduction trees, semantics tableaux). Informal proofs are structured, too (e.g., a proof per induction consists of induction base case, induction hypothesis and induction step, the first and the latter contain subproofs themselves).

Bibliography

[BanCar93] Bancerek G., Carlson P., *Semi-automatic translation for mathematics*, Sprache-Kommunikation-Informatik, Band 1, Max Niemeyer Verlag, 1993.

[BelSlo69] Bell J., Slomson A., *Models and Ultraproducts: an Introduction*, North Holland, 1969.

[CalLuo98] Callaghan P., Luo Z., *Mathematical Vernacular in Type Theory Based Proof Assistants*, Proceedings of UITP'98, 1998.

[Che76] Chester D., *The Translation of Formal Proofs into English*, AI 7, 1976.

[Cos94] Coscoy Y., *Traductions de preuves an langage naturel pour le systeme Coq*, Ecole Polytechnique, Paris, 1994.

[Dah94] Dahn I., Gehne J., Honigmann Th., Walther L., Wolf A., *Integrating Logical Functions with ILF*, Humboldt University, Berlin, 1994.

[deB87] De Bruijn N.G., *The mathematical vernacular, a language for mathematics with typed sets*, Marstrand, Sweden, 1987.

[Dow90] Dowek G., *Naming and scoping in a mathematical vernacular*, Inria-Rocquencourt, 1990.

[Gra79] Graetzer G., *Universal Algebra*, Springer Verlag, 1979.

[GroSid90] Grosz B. and Sidner C., *Plans for discourse*. In P. Cohen, J. Morgan editors, Intentions and Communication, MIT Press, Cambridge, MA, 1990.

[HarWri79] Hardy G., Wright E., *An Introduction to the Theory of Numbers*,

Oxford Press, 1979.

[Har97] Harrison J., *Proof style*, Technical Report, University of Cambridge, 1997.

[Hua94] Huang X., *Planning Argumentative Texts*. In Proceedings of the 15th International Conference on Computational Linguistics, Kyoto, 1994.

[KobNakFuw97] Kobayashi Sh., Nakamura Y., Fuwa Y., *A Function Format Language and its Writing/Representation Tool*, Shinshu University, Nagano, 1997.

[KurMos76] Kuratowski K., Mostowski A., *Set Theory*, North Holland, 1976.

[Lan80] Lang S., *Algebra*, Addison-Wesley, 1980.

[Luo89] Luo Z., Pollack R., Taylor P., *How to use LEGO*, LFCS-TN-27, Edinburgh, 1989.

[ManTho87] Mann W. and Thompson S., *Rhetorical Structure Theory: a Theory of Text Organization*, In L. Polanyi**,** editor, The Structure of Discourse. Ablex, Norwood, NJ, 1987.

[Mat84] Matuszewski R. - *On Automatic Translation of Texts from Mizar-QC language into English*, Studies in Logic, Grammar and Rhetoric, 1984.

[Mat89] Matuszewski R., *An Example of the Miz to TeX Production*, Fondation Ph. le Hodey, Brussels, 1989.

[Mat94] Matuszewski R. - *Mizar-MSE - a computer aided course for elementary logic of quantifiers*, Proceedings of International Congress of Mathematcians, ICM 94, Zurich, 1994.

[Mat95] Matuszewski R. - *Mathematical Proof-Checked Journal Automatically Translated into English*, Proceedings of Conference "Language and Technology", European Union, Poznań, 1995.

[Mat98] Matuszewski R. - *Journal of Formalized Mathematics - a collection of mechanically checked articles*, Proceedings of the International Congress of Mathematicians, ICM 98, Birkhauser, Berlin, 1998.

[Mat99] Matuszewski R., *Mizar Texts Automatically Translated into English*, Proceedings of Calculemus Workshop, CADE 99, Trento,1999 (position paper).

[Mat99a] Matuszewski R. - *Automatic Translation of Machine-Checked Mathematical Texts into English*, Proceedings of the 11th International Congress of Logic, Cracow, 1999.

[Qin84] Qin Bin, *Automatic Translation of Mizar-MSE Text into Chinese*, University of Connecticut, internal report, 1984.

[RudTry89] Rudnicki P., Trybulec A., *A Collection of TeX-ed Mizar Abstracts*, University of Alberta, internal report, 1989.

[RudTry99] Rudnicki P., Trybulec A., *On Equivalents of Well-foundedness –*

An Experiment in Mizar, Journal of Automated Reasoning, Vol 23, Issue 3, 1999,

[SimSin97] Simons M., Sintzoff M., *Algebraic composition and refinement of proofs*, TU Berlin, 1997.

[TrybBla85] Trybulec A., Blair H., *Computer Assisted Reasoning with Mizar*, IJCAI 1985, Vol.1.

[Zam99] Zammit V., *On the implementation of an Extensible Declarative Proof Language*, LNCS 1690, Springer Verlag 1999.

[Zin98] Zinn C., *Verifying textbook proofs*, Technical Report, Technical University of Vienna, 1998.

[ZhaCall99] Zhaohui Luo, Paul Callaghan, *Mathematical Vernacular and Conceptual Well-Formedness in Mathematical Language*, Lecture Notes in Computer Science Volume 1582, 1999, pp 231-250.

Roman Matuszewski
Department of Logic, Informatics and Philosophy of Science
Philological Faculty
University of Białystok, Poland
e-mail: romat@mizar.org