# Rule-based Intelligent System for Dictating Mathematical Notation in Polish

Agnieszka Bier
Faculty of Applied Mathematics
Silesian University of Technology
Gliwice, Poland
email: agnieszka.bier@polsl.pl

Zdzisław Sroczyński
Faculty of Applied Mathematics
Silesian University of Technology
Gliwice, Poland
email: zdzislaw.sroczynski@polsl.pl

***Abstract*—The paper describes the design and implementation of the system for dictation of mathematical expressions in Polish. The details of the developed solution environment are presented, as well as the workflow scheme of the actual natural language parser. In order to test the dictation engine, we have extended the Equation wizard editor, developed initially as a classic desktop application for MS Windows operating system. Preliminary experiments prove the system works efficiently and accepts even complex expressions and can, therefore, be used as a basis for the novel approach to editing mathematical documents by people with disabilities. The proposed solution provides an efficient interface for input of mathematical content and may be easily adapted to human-computer interaction systems for educational purposes.**

***Keywords*—Natural language; Mathematical notation; Assistive technologies; Verbalization.**

## I. Introduction

People with different kinds of disabilities, such as mobility or visual impairment, often encounter difficulties with editing mathematical equations. As most popular tools use document description languages such as LaTeX and MathML, or editors with graphical user interface, they require visual control, as well as the effective and precise manipulation of the keyboard, mouse or touch screen.

In the following sections, we present details of design and implementation of a system translating the spoken version of mathematical notation into the structural and formal description language. The output from the translation system can be passed further to e-learning applications, desktop publishing or algebra manipulation mathematical software.

The rules of the translation engine are fine-tuned to Polish language and, therefore, the system fills the gap in the solutions available to people with sight or motion disabilities in Poland.

The structure of the translation system contains the ”Equation wizard” visual editor, enhanced with App Tethering technology by the extra ”Mobile assistant” application. The actual voice recognition is performed by a mobile device, which works as the terminal for the server – the visual editor. The results of recognition can be displayed on-the-fly by the mobile assistant app. This way, we have integrated the environment for testing translation rules from spoken to structural notation with minimal effort. Moreover, the mobile multi-platform clients can be the starting platform for various assisting educational projects.

The paper is organized into five sections. In Section II, we provide a short review of the state-of-the-art contributions for editing/verbalizing mathematical formulas, with particular interest in widely used standards for writing mathematical content and parsing to spoken language. Section III concerns the design and implementation details of the proposed solution. In Section IV, some experimental results and user experience are presented. We conclude the paper with a brief discussion on further development of the system in Section V.

## II. Related work

There are many alternative methods of presentation and editing mathematical content: automatic recognition of printed expressions [1], recognition of handwritten equations, editing with visual editors (WYSIWYG), editing with the use of document description languages (LaTeX, MathML), editing with the use of specialized notations and software (for example Braille dot language), verbalization (translation to/from the spoken, natural language version) [2]–[9]. Contemporary computer systems implement the voice input and natural language processing with increasing success rates and user satisfaction level [10]–[12]. It is worth noting that the understanding of commands and sentences of natural language in general tasks can be just statistically exact, while for the mathematical notation things get much more complicated – every single sign, letter or number matters a lot, and possibly can change the meaning of the whole expression [13].

### A. Description languages

There are two main description languages used to encode two-dimensional mathematical notation into linear form. The first one is LaTeX – de-facto standard for scientific and educational documents. The main advantages of LaTeX are popularity and consistent syntax, which allows editing of even very complex equations by hand.

The second one is MathML, compatible with XML specification, and therefore used in Internet applications. Some Internet browsers render MathML natively (Firefox, Chrome); for the rest (Internet Explorer) the end-user can install a plugin for that purpose. MathML uses two different layers of notation: the presentation layer, which describes the visual appearance of the equation, and the content layer, which describes the meaning of the expression. Although these two layers can be mixed in one document, the majority of applications utilizes only the presentation layer. This way, MathML suffers from ambiguities at the same level, as LaTeX does. The resolving of ambiguities in the encoded mathematical notation is the key issue for further processing and translating the given equation into another media.

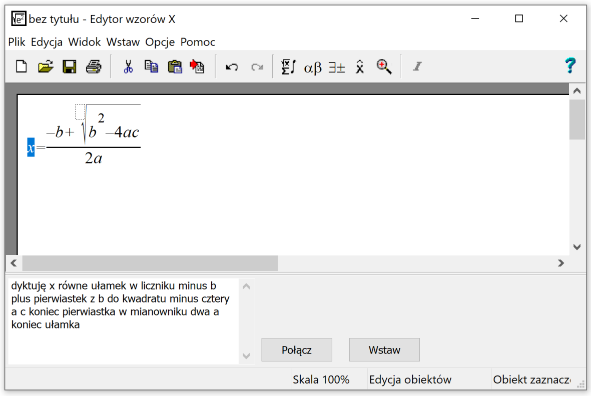
In our framework and ”Equation wizard” editor, we use dedicated internal language, which helps to preserve the context of the equation parts. We call it EQED (EQuation EDitor) format. It can be also easily translated into the internal object tree, representing the equation in the rendering engine of the editor. On the other hand, we provide filters for import and export of LaTeX and MathML notation.

### B. Math verbalization

There are no formal rules for the verbalization of mathematical content. Some common spelling suggestions are given in numerous mathematical schoolbooks and scientific works, as well as the tradition during teachers education [14][15]. Mathematical publications in Polish very seldom include natural language description of the equations, so the only source of unified de-facto standard in this area is the pronunciation used during maths lessons or university language centers' publications [16]. Moreover, spelling maths is mostly parallel with visual presentation on the blackboard, so any ambiguities are resolved immediately. On the other hand, people with sight disabilities have no such visual aid and the maths verbalization designed for them requires much more precise syntax [17][18], including the terminating signs for most of the subexpressions. The maths verbalization in English is certainly the most common and comprehensive [19][20], however other languages still need extended elaboration [21][22].

## III. DICTATING MATHEMATICAL CONTENT

Proper conversion from spoken language to formal, structured mathematical notation is an important issue, which could make the education of physically disabled people much easier. Moreover, common mathematical education could benefit as well, as speech based interfaces became popular nowadays.


bez tytułu - Edytor wzorów X
Plik Edycja Widok Wstaw Opcje Pomoc
Połącz
Wstaw
Skala 100%
Edycja obiektów
Obiekt zaznacz

$$x = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

dyktuję x równe ułamek w liczniku minus b plus pierwiastek z b do kwadratu minus cztery a c koniec pierwiastka w mianowniku dwa a koniec ułamka

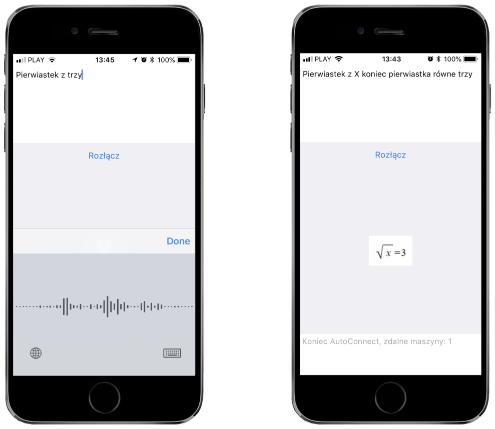
Figure 1. ”Equation wizard” editor: the user interface of dictation module.

In order to test the dictation engine, we have extended the ”Equation wizard” editor, developed initially as a classic desktop application for MS Windows operating system.

The user interface of the dictation module is shown in Figure 1. The natural language description of the mathematical notation appears in the multi-line text widget and the operator is able to paste it into the main window of the editor, invoking the translation process. This mode resembles more or less a debugging console and for the future production-ready versions of the software it should be altered to provide the complete voice driven human-machine interface. Eventually, this interface should include commands for the manipulation of the parts of the equation as well.

Because of the lack of Polish language native support in MS Windows voice recognition engine, there was the need for an extra software solution. Therefore, we have integrated the supplementary multi-platform application for mobile operating systems (iOS and Android). Both of the mentioned operating systems support dictation in Polish, although there are some differences, especially when spelling numbers. The quality of recognition for particular words also differs and is below standard results for the sentences, because in the natural language description of mathematics every sign and single letter has significant meaning and placement. Nevertheless, the results of the dictation were satisfactory enough to perform some tests on rather complicated equations.

The ”Mobile assistant” application provides the result of the recognition in the graphical form, which can be considered as a substitute of the editor module. Some screen shots of the user interface of this assistant application are shown in Figure 2.



Pierwiastek z trzy

Pierwiastek z X koniec pierwiastka równe trzy

$\sqrt{x} = 3$

Koniec AutoConnect, zdalne maszyny: 1

Figure 2. ”Mobile assistant” application: the user interface during dictation in iOS (left) and after the recognition and visualization of the results (right).

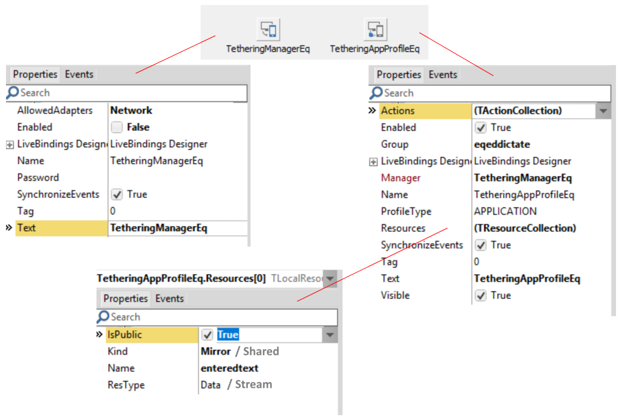
### A. App Tethering

App Tethering is the technology which allows to easily extend the existing applications developed in RAD Studio [23] with the network connectivity, regardless of the operating system used. The ”Equation wizard” editor has been enhanced this way with the use of the supplementary mobile apps mentioned above.

The main aim of App Tethering is to provide the novel mobile interface for classic desktop application, which is also our case. The classic MS Windows desktop application could work as a server, responding to the information changes in the mobile terminal and possibly returning the result data.

App Tethering technology:

- operates at Windows, MacOS, Android and iOS platforms,
- allows to exchange the data (resources) between platforms,
- can be introduced into any application based on RAD Studio Run-Time Library (RTL) with the use of:
  - IP connections in the same subnet,
  - classic Bluetooth connections,
- provides an automatic search for connected applications,
- allows to run remote procedures,
- ensures easy exchange of the data,
- introduces simple network protocol and is easy to implement with ready to use high level software components.



| | |
|---|---|
| AllowedAdapters | Network |
| Enabled | False |
| LiveBindings Design | LiveBindings Designer |
| Name | TetheringManagerEq |
| Password | |
| SynchronizeEvents | ✓ True |
| Tag | 0 |
| Text | TetheringManagerEq |

| | |
|---|---|
| Actions | (TActionCollection) |
| Enabled | ✓ True |
| Group | eqeddictate |
| LiveBindings Design | LiveBindings Designer |
| Manager | TetheringManagerEq |
| Name | TetheringAppProfileEq |
| ProfileType | APPLICATION |
| Resources | (TResourceCollection) |
| SynchronizeEvents | ✓ True |
| Tag | 0 |
| Text | TetheringAppProfileEq |
| Visible | ✓ True |

TetheringAppProfileEq.Resources[0] TLocalReso

| | |
|---|---|
| IsPublic | ✓ True |
| Kind | Mirror / Shared |
| Name | enteredtext |
| ResType | Data / Stream |

Figure 3. App Tethering: software components flowchart.

The exemplary setup of the App Tethering components and source code snippets are given in Figure 3 and Figure 4, respectively.

```
procedure TMainForm.ResourceReceived(
  const Sender: TObject;
  const AResource: TRemoteResource);
begin
 MemoEqDictate.Text:='dyktuje␣' +
        AResource.Value.AsString.ToLower;
end;
...
TetheringAppProfile.Resources.
         FindByName('eqpicture').Value :=
                         EqPictureStream;
```

Figure 4. App Tethering: source code snippets for resource exchange.

## B. Parsing the verbal form of mathematical expressions

A parser designed for automatic recognition of the structure of the mathematical expression given in spoken form is incorporated into the main engine of the ”Equation wizard” editor. The core of the equation editor uses a tree-like data structure to represent the expression being manipulated in the main window. The leaves of the tree contain specialized objects representing particular parts of the equation i.e. simple symbols (as letters or numbers) or two-dimensional templates (as for example fractions, roots and integrals). Recursive traversal through the entire tree gives the possibility to translate, edit, move or alter parts of the expression. Moreover, the core engine offers an interface for expression import and export using the internal domain-specific language (EQED, based on LaTeX), MathML or plain LaTeX. In previous experiments, this engine was extended with the verbalization and search modules, ensuring the export of the equation with natural language, with possible multi-language support [2][24].

The import of the natural language form of an equation is the next level achievement for the ”Equation wizard” environment. The parser environment processes the input from the natural language into the graphical, visual representation in the following steps:

- the acquisition of the input description string in the natural language (there is the requirement of the compatibility with the rules for Polish verbalization engine at the moment, which is convenient for testing) with the use of an assistant mobile application and speech recognition engine from the mobile operating system (iOS or Android),
- transfer of the input description string into the recognition module in the ”Equation wizard” application, working as server in this mode,
- the actual translation with the set of replacement rules described as pairs of natural language phrase and the corresponding EQED internal format command,
- backward scanning for the corrections of the keywords without explicit termination mark,
- the transfer of the result string encoded with EQED commands to the standard import module of the equation editor in order to visualize it and edit,
- the final visual appearance of the dictated equation is returned to the assistant mobile application and shown to the user.

The translation rules have the following format:

`natural_language_phrase~EQED_internal_format`,

where ˜ (tilde sign) is a separator. The left side is the part to search for and the right side is the corresponding notation in EQED format. Some exemplary rules are presented in Figure 5.

Symbols `#`, `@` and `$` were introduced to maintain subexpressions which do not have an explicit termination command or require pairing of possibly nested symbols as brackets. These symbols are processed at the final stage of parsing.

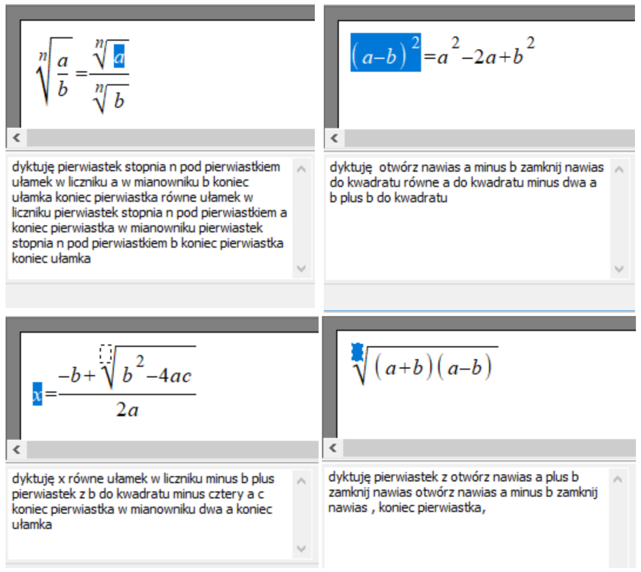
The current version of the parser supports different variants of the spoken notation, making the dictation as natural as possible. The main inconvenience, however, is the requirement to provide the termination command for the majority of subexpressions.

```
pierwiastek stopnia~\EQEDroot{
pierwiastek z~\EQEDroot{\EQEDplain{}}{
pod pierwiastkiem~}{
koniec pierwiastka~}
do kwadratu~#}{\EQEDnplain{2}}
do potegi~#}{\EQEDnplain{
otworz nawias~$EQEDbrackets{\left(}{
zamknij nawias~}{\right)@
```

Figure 5. Example rules for translation Polish verbalized mathematical notation into structural form.

## IV. Experimental results

We have performed a series of experiments involving four experts, with the use of different operating systems for the assistant mobile application (Android and iOS). The experts were familiar with the ”Equation wizard” editor user interface and editing rules, knew the rules of verbalization of the mathematical notation and could check it live before dictation experiments. There were over a dozen dictation attempts of different equations by every expert. All these circumstances were undoubtedly helpful to build the dictation commands in the proper form (see some exemplary results in Figure 6). Especially, the overall high level of experience in editing mathematical content was decisive for rather high success rate (about 70%). On the other hand, the requirement to spell the whole equation at once (temporary for current beta solution), could certainly make the dictation extremely difficult for not prepared user.



$\sqrt[n]{\frac{a}{b}}=\frac{\sqrt[n]{a}}{\sqrt[n]{b}}$

dyktuję pierwiastek stopnia n pod pierwiastkiem ułamek w liczniku a w mianowniku b koniec ułamka koniec pierwiastka równe ułamek w liczniku pierwiastek stopnia n pod pierwiastkiem a koniec pierwiastka w mianowniku pierwiastek stopnia n pod pierwiastkiem b koniec pierwiastka koniec ułamka

$(a-b)^2=a^2-2a+b^2$

dyktuję otwórz nawias a minus b zamknij nawias do kwadratu równe a do kwadratu minus dwa a b plus b do kwadratu

$x=\frac{-b+\sqrt{b^2-4ac}}{2a}$

dyktuję x równe ułamek w liczniku minus b plus pierwiastek z b do kwadratu minus cztery a c koniec pierwiastka w mianowniku dwa a koniec ułamka

$\sqrt{(a+b)(a-b)}$

dyktuję pierwiastek z otwórz nawias a plus b zamknij nawias otwórz nawias a minus b zamknij nawias , koniec pierwiastka,

Figure 6. Examples of successfully dictated equations with the visualization in ”Formula wizard” editor.

## V. Conclusion

In this paper, we have described the design of the natural language to structural mathematical notation translation system. The main language of the tests was Slavic language – Polish, which significantly differs from the western languages, especially when it comes to numerals and details of mathematical notation.

The usage of the existing engine and graphical equation editor (”Equation wizard”) for rendering and processing of the mathematical notation increased the effectiveness of the translation. Several preliminary experiments proved that the system works smoothly for the current, limited list of translation rules. Nevertheless, some of the mathematical expressions in the test suite were relatively complex and careful dictation gave very promising results for them.

It is worth to note that translation rules could be profiled according to specific features of speech recognition modules from the mobile operating systems. The introduction of some editing voice commands would certainly make the system more user friendly in real life examples for the people with motion disabilities. The last improvement could be a closer integration with mobile operating system, i.e., usage of smart glasses interface, sharing the results with other applications, e-mail or messaging clients.

## References

[1] Z. Sroczyński, “Priority levels and heuristic rules in the structural recognition of mathematical formulae,” *Theoretical and Applied Informatics*, vol. 22, no. 4, p. 273, 2010.

[2] A. Bier and Z. Sroczynski, “Adaptive math-to-speech interface,” in *Proceedings of the Multimedia, Interaction, Design and Innnovation*, ser. MIDI ’15. New York, NY, USA: ACM, 2015, pp. 7:1–7:9. [Online]. Available: http://doi.acm.org/10.1145/2814464.2814471

[3] A. Bier and Z. Sroczyński, “Rule based intelligent system verbalizing mathematical notation,” *Multimedia Tools and Applications*, vol. 78, no. 19, pp. 28 089–28 110, 2019.

[4] J. Cuartero-Olivera, G. Hunter, and A. Pérez-Navarro, “Reading and writing mathematical notation in e-learning environments,” *eLearn Center Research Paper Series*, no. 4, pp. 11–20, 2012.

[5] D. Attanayake, J. Denholm-Price, G. Hunter, E. Pfluegel, and A. Wigmore, “Speech interfaces for mathematics: Opportunities and limitations for visually impaired learners,” in *IMA International Conference on Barriers and Enablers to Learning Maths: Enhancing Learning and Teaching for All Learners*, 2015, pp. 1–8.

[6] M. Isaac *et al.*, “Improving automatic speech recognition for mobile learning of mathematics through incremental parsing.” in *Intelligent Environments (Workshops)*, 2016, pp. 217–226.

[7] A. Mazzei, M. Monticone, and C. Bernareggi, “Using nlg for speech synthesis of mathematical sentences,” in *Proceedings of The 12th International Conference on Natural Language Generation*, 2019, pp. 463–472.

[8] A. Szesz Junior, L. Ribeiro Mendes, and S. de Carvalho Rutz da Silva, “Math2text: Software para gerao e converso de equaes matemticas em texto - limitaes e possibilidades de incluso,” *RISTI*, no. 37, pp. 99–115, 2020.

[9] P. Sojka, V. Novotn, E. F. Ayetiran, D. Luptk, and M. tefnik, “Quo vadis, math information retrieval,” in *Proceedings of Recent Advances in Slavonic Natural Language Processing*, 2019, pp. 117–128.

[10] D. Polap, “Voice control in mixed reality,” in *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2018, pp. 497–500.

[11] J. Kowalski *et al.*, “Older adults and voice interaction: A pilot study with google home,” in *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–6.

[12] I. Tautkute, T. Trzciński, A. P. Skorupa, Ł. Brocki, and K. Marasek, “Deepstyle: Multimodal search engine for fashion and interior design,” *IEEE Access*, vol. 7, pp. 84 613–84 628, 2019.

[13] R. Fateman, “Handwriting + speech for computer entry of mathematics,” *Style, Benjamin L. Kovitz, Manning Publications Company*, 2004.

[14] ——, “How can we speak math?” Computer Science Division, EECS Department, University of California at Berkeley, Tech. Rep., 2013.

[15] T. Sancho-Vinuesa *et al.*, “Automatic verbalization of mathematical formulae for web-based learning resources in an on-line environment,” *INTED2009 Proceedings*, pp. 4312–4321, 2009.

[16] R. Szopa and D. Zuziak, *Selected Texts on Higher Mathematics*, ser. Academic Textbooks, No 1837. SUT Press, 1994.

[17] S. Bocconi, S. Dini, L. Ferlino, C. Martinoli, and M. Ott, *ICT Educational Tools and Visually Impaired Students: Different Answers to Different Accessibility Needs*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 491–500. [Online]. Available: https://doi.org/10.1007/978-3-540-73283-9_55

[18] A. Salamonczyk and J. Brzostek-Pawlowska, “Translation of MathML formulas to Polish text, example applications in teaching the blind,” in *Cybernetics (CYBCONF), 2015 IEEE 2nd International Conference on*. IEEE, 2015, pp. 240–244.

[19] D. Attanayake, J. Denholm-Price, G. Hunter, E. Pfluegel, and A. Wigmore, “Intelligent assistive interfaces for editing mathematics.” in *Intelligent Environments (Workshops)*, 2012, pp. 286–297.

[20] D. Attanayake, G. Hunter, J. Denholm-Price, and E. Pfluegel, “Novel multi-modal tools to enhance disabled and distance learners’ experience of mathematics,” *ICTer*, vol. 6, no. 1, pp. 1–10, 2013.

[21] M. Maćkowski, P. Brzoza, M. Żabka, and D. Spinczyk, “Multimedia platform for mathematics’ interactive learning accessible to blind people,” *Multimedia Tools and Applications*, pp. 1–18, 2017.

[22] P. Riga, G. Kouroupetroglou, and P.-P. Ioannidou, “An evaluation methodology of math-to-speech in non-English DAISY digital talking books,” in *International Conference on Computers Helping People with Special Needs*. Springer, 2016, pp. 27–34.

[23] Z. Sroczyński, “Internet of things location services with multi-platform mobile applications,” in *Proceedings of the Computational Methods in Systems and Software*. Springer, 2017, pp. 347–357.

[24] A. Bier and Z. Sroczynski, “Towards semantic search for mathematical notation,” in *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2018, pp. 465–469.