# Computer Processing of Nemeth Braille Math Notation[1]

*(Extended Abstract)*

**H. Guo, G. Gupta, J. Mendez, C. Weaver, A. Karshmer, S. Geiger**

*The MAVIS Group*
*New Mexico State University*
*Las Cruces, NM 88003*

## 1 The Problem: Converting Nemeth Math to LaTeX

Nemeth Braille Code is a Braille-based notation to enable visually impaired scholars and students to read and write Mathematics. It was designed by Dr. Abraham Nemeth in the late 1940s to allow visually impaired individuals to convey mathematical and technical information. While Nemeth Braille code is well suited for humans, it is not easy to process with a computer. To make Nemeth Math code computer processable, we should be able to *parse* Nemeth code text for grammatical correctness (i.e., check that it conforms to the rules of syntax laid out in the Nemeth code specification [5]), and once parsed, transform it into a form so that it can be processed further. To the best of our knowledge there are no parsers available that will parse Nemeth code to check its grammatical correctness. The principal reason for this is that the Nemeth code notation falls in the class of languages known as *context sensitive languages* [7]. Context sensitivity means that the grammatical correctness of a subexpression written in Nemeth code depends on the context in the expression it occurs in. This is similar to the case of English, where, for example, to check the grammatical correctness of the sentence one has to make sure that a singular subject noun must have a singular verb. So `The man eats an apple` is grammatically correct, while *The man eat an apple* is not. Another negative point about the Nemeth code is that it is not an *LALR(k)* [7] language. This means that if we were to write grammar rules for Nemeth code, then given a Nemeth code expression that we are trying to parse, we cannot tell by looking ahead $k$ symbols in the expression, which grammar rule should we apply to parse the expression. Thus, traditional parsing and compiling technology [7], that is normally used to process computer languages, cannot be used. It should be noted that these properties of the Nemeth code—especially context sensitivity—are precisely the reason why Nemeth code is so suitable for a visually impaired person. Since a blind person reads by touching the Braille code, it is important that he/she be aware of the context of the current symbol or expression he is currently looking at, at any given time. Thus, building-in the context into the language is quite helpful for the blind reader, but it make computer processing quite difficult as the language becomes context sensitive and non-LALR(k). Also, at the time the Nemeth notation was designed (late 40s and early 50s), the computer was just being invented. Thus, computer readability or computer processability was not one of the design criteria for Nemeth Math code. Additionally, very little was known about structure of languages then (the seminal work of Chomsky on structure of languages appeared only in the early 50s).

In this paper we present a technique based on *denotational semantics* and *logic programming* that allows us to build a parser and transform Nemeth code into an intermediate form that can then be used for generating equivalent LaTeX code or MathML code, or whatever one wishes. Our system has been developed as part of the MAVIS project (Mathematics Accessible to Visually Impaired Students) at New Mexico State University. The goal that lead to this research was to design two systems, one to automatically translate LaTeX to Nemeth Braille code and the other to automatically translate Nemeth code to LaTeX in order to facilitate the communication between a blind student and a sighted instructor. Thus, we envisage the following scenario: an instructor prepares homework using LaTeX , which is automatically converted to Nemeth code, using the LaTeX to Nemeth code translator, so a blind student can read it. The blind student prepares his/her answer in Nemeth code, which is then automatically translated to LaTeX so the instructor can grade it.

Building the LaTeX to Nemeth code translator was a relatively easy task, since LaTeX's mathematical expressions are written using a simple, context-free notation. However, building a Nemeth code to LaTeX translator is difficult due to the problems of context sensitivity and non-LALR(k) properties of Nemeth Math code. However, by using a combination of innovative techniques we have been able to develop such a

[1]This work has been supported by NSF HRD 9800209 and INT 9904063.

translator for translating Nemeth code to LaTeX . As part of the MAVIS project a complete translator has been developed. The translator we have developed is the first of its kind.

## 2 A Semantics based Approach to Translation

*Denotational Semantics* [3] is a well-established methodology for design and analysis of computer languages. In the denotational approach, the semantics or meaning of a computer language/notation is specified in terms of mathematical objects (such as sets and functions). A denotational specification has three components: *syntax specification* which maps sentences of the language to parse trees; *semantic algebras* that specify these mathematical objects, and *valuation functions* that map parse trees to semantic algebras. It turns out [4] that the syntax specification, the semantic algebra, and the valuation functions can be specified using logic programming [2]. This logic programming specification is also executable. To obtain a translator from a language $\mathcal{L}_1$ to $\mathcal{L}_2$ we choose the semantic algebra consisting of parse trees of $\mathcal{L}_2$. The logic programming specification then automatically turns into a translator which can be executed to translate sentences of language $\mathcal{L}_2$ to sentences of language $\mathcal{L}_2$.

In the MAVIS project the language $\mathcal{L}_1$ is Nemeth Math code, and the language $\mathcal{L}_2$ is LaTeX . Syntax and Semantics of Nemeth Math code is given in terms of parse trees of LaTeX using the denotational approach. These specifications are written using logic programming, and thus when executed, produce a translator from $\mathcal{L}_1$ to $\mathcal{L}_2$.

The syntax and semantic specification covers the whole Nemeth Math code, except for a few symbols that are found in Nemeth Math code, but do not have a counterpart in LaTeX . There is error recovery built in for some of the constructs, if the user types Nemeth code that is not well-formed. Examples of such error recovery includes missing grouping indicator (e.g., missing end-fraction symbol, missing parenthesis, etc.).

The translator we have developed as part of the MAVIS project is, to the best of our knowledge, the first of its kind ever developed. The final paper, if accepted, will include: (i) complete details of our novel denotational approach, (ii) a description of the implementation, and (iii) examples of translation from Nemeth to LaTeX and MathML.

It should be noted that the translator from Nemeth to Braille is part of a larger system, which also includes an audio equation browser for the LaTeX output [1]. Essentially, the LaTeX expression generated is graphically displayed, and the audio equation browser also reads it out using sound. The user has the ability to press keys and go deeper into the subexpressions, and to re-hear subexpressions and expressions. The audio browser is still under development.

Our research makes several contributions: It has produced a system that solves a problem that was considered to be unsolvable [6]. It also develops a general methodology that can be used for processing other similar codes for marking up mathematics (e.g., those used in Europe).

## References


[1] A. Karshmer, G. Gupta, S. Geiger, C. Weaver. A Framework for Translation of Braille Nemeth Math to LaTeX: The MAVIS Project. In *Proc. ACM Conference on Assistive Technologies*, ACM Press, pp. 136-143, Mar. 1998.

[2] L. Sterling & S. Shapiro. The Art of Prolog. MIT Press, '94.

[3] D. Schmidt. *Denotational Semantics: a Methodology for Language Development*. W.C. Brown Publishers, 1986.

[4] G. Gupta. Horn Logic Denotations and Their Applications. In *The Logic Programming Paradigm: A 25 year perspective*. Springer Verlag. pp. 127-160. May 1999.

[5] A. Nemeth. The Nemeth Braille Code for Mathematics and Science Notation 1972 Revision (Frankfort KY: American Printing House for the Blind, 1972)

[6] K. Miesenberger, B. Stöger. Personal Communication.

[7] A. Aho, J.D. Ullman, R. Sethi. Compilers: Principles, Techniques, and Tools. Addison Wesley. 1986.