Evaluating Speech Synthesis on Mathematical Sentences

Alessandro Mazzei
Università degli Studi di Torino
alessandro.mazzei@unito.it

Michele Monticone
Università degli Studi di Torino
michele.monticone@edu.unito.it

Cristian Bernareggi
Università degli Studi di Torino
cristian.bernareggi@google.com

Abstract

English. In this paper we present the main features of a rule-based architecture to transform a LaTeX encoded mathematical expression into its equivalent *mathematical sentence* form, i.e. a natural language sentence expressing the semantics of the mathematical expression. Moreover, we describe the main results of a first human based evaluation of the system for Italian language focusing on speech synthesis engines.

Italiano. *In questo lavoro presentiamo le caratteristiche principali di un'architettura software a regole per trasformare un'espressione matematica, codificata in LaTeX, nella sua equivalente frase matematica, cioè una frase del linguaggio naturale che esprima la stessa semantica dell'espressione originale. Inoltre, descriviamo i primi risultati di una valutazione del sistema fatta da esseri umani per la lingua italiana riguardante principalmente i motori di sintesi del parlato.*

1 Introduction

Computational linguistics can help people in many ways, especially in the field of assistive technologies. In the case of mathematical domain, blind people can access to a mathematical expression by listening its LaTeX source. However, this process has several drawbacks. First of all, it assumes the knowledge of the LaTeX. Second, listening LaTeX is slow and error-prone, since LaTeX is a typographical language, that is a language designed for specifying the details of typographical visualization rather than for efficiently communicate the semantics of a mathematical expression. For instance, the simple LaTeX expression $f(x)$ is a typographical description and so it represents both the function application of f to x, and the multiplication of the variable f for the variable x surrounded by parenthesis.

There are many lines of research to enable people with sight impairments to access mathematical contents. It is possible to embed mathematical expressions in web pages not only as images but through MathML or MathJax (Cervone, 2012) and in PDF documents produced from LaTeX (Ahmetovic et al., 2018). Other research directions concern conversion into Braille (Soiffer, 2016) and speech reading (Raman, 1996; Waltraud Schweikhardt, 2006; Sorge et al., 2014).

In this paper we follow another direction: we consider the possibility to produce a *mathematical sentence*, i.e. a natural language sentence expressing the semantics of a mathematical expression. Indeed, the idea to use mathematical sentences for improving the accessibility of mathematical expressions has been previously presented and experimented for Spanish in (Ferres and Fuentes Sepúlveda, 2011; Fuentes Sepúlveda and Ferres, 2012). However, in contrast to previous work on mathematical sentences, in this work we use a natural language generation (NLG) architecture rather than a template-based one for generating sentences. By using NLG architecture we obtain (i) more portability, and (ii) a major and simple customization of the output.

We have two research goals in this paper. The first goal is to describe a system for transforming a mathematical expression natively encoded in LaTeX in its equivalent mathematical sentence (cf. Figure 1). The processing flow follows a well-known approach, called *interlingua* in the field of machine translation (Hutchins and Somer, 1992).

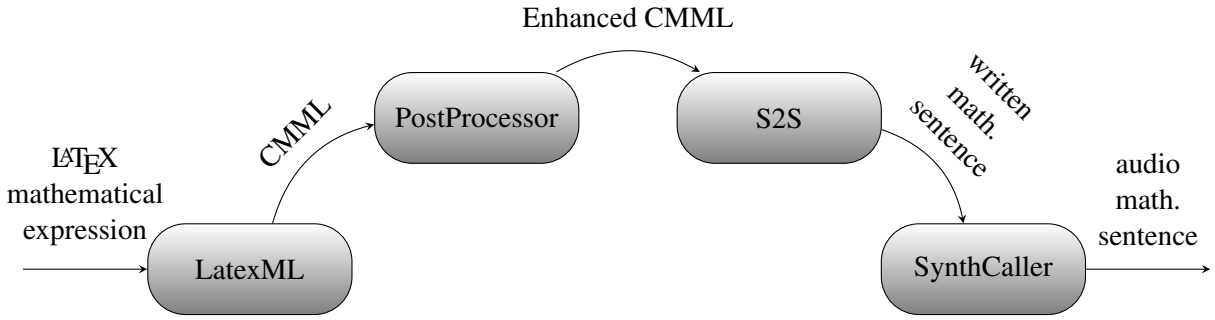

Figure 1: The software architecture for the generation of mathematical sentences. The process starts from (1) the LaTeX representation of the expression, (2) its translation in CMML, (3) enhancement of CMML, (4) generation of the written form of the mathematical sentence, (5) production of the audio form of the mathematical sentence.

Indeed, the process of generating a mathematical expression from its LaTeX source is a two-step algorithm. In the first step the LaTeX is analyzed and its semantics is represented in *Content MathML* (CMML henceforth), a W3C standard for expressing the syntax and the semantics of mathematical expressions[1]. In the second step, the CMML representation is used as input of the S2S (Semantics to Speech) module, that is a NLG module generating the mathematical sentence. Note that the S2S module inserts in the sentence parenthesis and pauses too. The sentence will finally be transformed in audio format encoding by an external synthesis engine.

The second goal of this paper is to give a first evaluation of the performance of two distinct synthesis engines in the domain of mathematical sentence. With a pilot experimentation conducted with four blind people, we will compare the perception of the mathematical sentences of a neural-network based speech engine and of a formant-based speech engine.

In Section 2 we will describe the main features of the developed system, in Section 3 we will describe the experimentation and finally in Section 4 we end the paper with some conclusions and introducing future work.

2 Building Mathematical Sentences

The first step of our algorithm is the generation of CMML associated to a LaTeX formula. We based this step on an external tool named *LatexML* (Miller, 2007). However, the CMML obtained from this tool needed to be enhanced by a post-processing procedure for (1) uniform them to CMML standard and (2) to remove ambiguity as for the case $y = f(x)$. In Figure 2 we report the CMML representation for the mathematical expression $x > b \Longrightarrow |f(x)| < M$.

Mathematical notation has been conceived with the aim of representing mathematical concepts using a specific written symbolic language. As working hypothesis, we decided to assume a "specialized" syntactic analysis for a number of mathematical objects. For instance, `x plus three` indicates the action of adding one quantity to another, so it can be represented as a declarative structure. As a consequence, `plus` can be analysed as verb and this assumption can be extended to all the mathematical sentences. In this paper we considered only the mathematical structures belonging to the subfield of the mathematical analysis. In particular, we considered all the expressions in an Italian analysis book (Pandolfi, 2013). By using this corpus of expressions and by assuming that all numbers and variables can be treated as nouns and that all arithmetic operators can be treated as verbs, we found eight additional categories for representing all complex mathematical expressions and we defined a specific syntactic construction for each category.

In Table 1, we reported some examples of syntactic constructions for mathematical expressions. We decided to analyse and represent the mathematical sentences of relational operators as copula sentences (`a è maggiore di b`, *a is greater than b*), algebraic operators as declarative sentences (`a prodotto cartesiano b`, *a cartesian product b*), logical operators as conjunctions (`a o b`, *a or b*), elementary operators (e.g. `radice`, *radical*), sequence (e.g. `limite`, *limit*), calculus (e.g. `integrale`, *inte-*

[1] `https://www.w3.org/TR/MathML3/chapter4.html`

```
<apply>
    <implies/>
    <apply>
        <gt/>
        <ci>x</ci>
        <ci>b</ci>
    </apply>
    <apply>
        <lt/>
        <apply>
            <abs/>
            <apply>
                <ci>f</ci>
                <ci>x</ci>
            </apply>
        </apply>
        <ci>M</ci>
    </apply>
</apply>
```

Figure 2: The CMML representation of the mathematical expression $x > b \implies |f(x)| < M$.

Mathematical Expression	Construction
$>, \geq, \gg, \ldots$	Copula
$+, -, *, \ldots$	Declarative
$\wedge, \vee, \neg, \ldots$	Coordination
$\sin, \cos, \tan, \ldots$	Noun Phrase
$\sum_{[x=a]}^{[b]} [f(x)], \ldots$	Noun Phrase
$\int_{[a]}^{[b]} f(x)\, \mathrm{d}x$	Noun Phrase
$\{[vars] \mid conditions\}$	Reduced Relative
$([x], [y])$	Reduced Relative

Table 1: Mathematical expressions and their linguistic constructions.

gral) as noun phrases (`La radice quadrata di x`, *the square root of x*), pairs and conditional sets as reduced relatives (`L'insieme delle x tali che x è minore di 3`, *the set of x such that x is less than 3*). Our syntactic representations for mathematical operators in the analysis domain could have alternative representations or could be specialized in a more refined classification (c.f. (Chang, 1983)), but we decided to use only eight category for sake of simplicity.

Traditional NLG architectures split the generation process into three distinct phases, that are *document planning*, *sentence planning* and *realization* (Reiter and Dale, 2000; Gatt and Krahmer, 2018). In particular document planning decides *what* to say and sentence planning and realization decides *how* to say it. In the system architecture depicted in Figure 1, the content of the communication is specified by the input mathematical expression, so the content selection phase is not necessary at all. In Section 2.1 we will give some details on the rule-based sentence planner designed for managing mathematical sentences and in Section 2.2 we will describe the use of the SimpleNLG-it realizer for the case of mathematical domain.

2.1 Building a Sentence Planner for Mathematical Sentences

The input of the sentence planner is a mathematical expression in the form of enhanced CMML. In order to associate a *sentence plan*, that is a a sort of under-specified tree-based syntactic structure, we devised a recursive algorithm that traverses top-down the CMML structure.

By considering the eight categories used to classify all mathematical expressions, for each category we designed a prototypical sentence plan that will be used in the recursive process. Each prototype builds a specific linguistic construction (e.g. *copula, reduced relative* etc.), that is designed for giving syntactic roles to the arguments of the specific mathematical construction. For instance, on the left of the Figure 3, we reported the prototypical sentence plan for the *conditional set* mathematical structure and on the right of we reported an example of its instantiation. In the final produced structures we have that, (1) the leaves of the sentence plan are lemmas rather than words, (2) the syntactic relations among the nodes are expressed using both dependency relations (e.g. subj, complement) as well as constituency nodes (e.g. Prepositional Phrase, PP). Note that this is the input format for sentence plan required by the SimpleNLG realizer (see Section 2.2).

In order to build a complete sentence plan for a mathematical sentence by using the eight categories for mathematical expressions, there are two important issues.

The first issue concerns the perception of precedence of the arithmetic operator. Listening mathematics has some peculiarities with respect to reading it. For instance, division is granted a higher precedence than addition, and during the reading process the expression $a + b/c$ is parsed as $a + \frac{b}{c}$ without ambiguities. A different result arises if one listens the equivalent mathematical sentence `a plus b divided by c` without reading the expression: we experimented that the most frequent perceived parse is $\frac{a+b}{c}$. After a limited num-

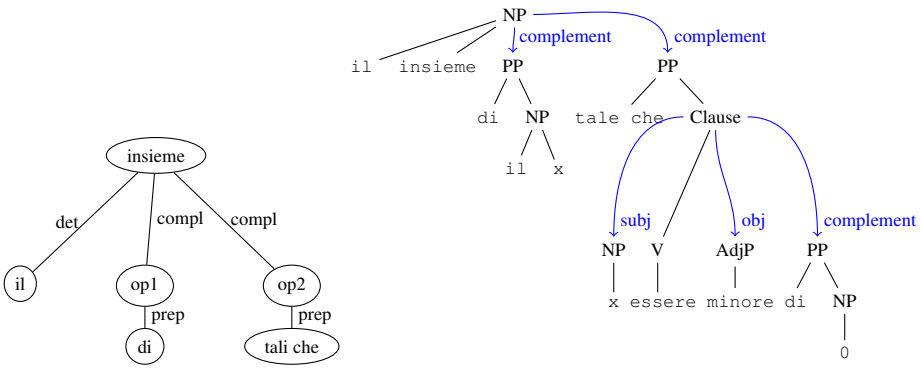

Figure 3: The prototypical sentence plan for the *conditional set* mathematical structure (left), and its fulfillment producing the sentence `L'insieme degli x tali che x è minore di 0` (rigth, *the set of all x such that x is lesser than 0*).

ber of experiments in listening arithmetic expressions with distinct (blind and not blind) people, we decided to state as working hypothesis that *the precedence of the arithmetic operators are perceived in the reverse order when one listens a mathematical expressions without reading it*[2].

A second issue is how to represent the correct structures of the operators. In other words, how we can build a mathematical sentence unambiguously equivalent to $\frac{a+b}{c}$? A trivial but effective solution is to use parenthesis, that is to produce the mathematical sentence `open parenthesis a plus b close parenthesis divided by c`. However, the drawback of this solution is the length of the sentence that, for very complex expressions, can augment substantially.

In order to account for both the issues, we modified the sentence planner in two ways. First, we decided to model parenthesis as lexical items, that is we considered *open-parenthesis* and *closed-parenthesis* as two new lexical items of the SimpleNLG lexicon which can be used as pre-modifier and post-modifier of a mathematical sentence respectively. Second, similar to (Fuentes Sepúlveda and Ferres, 2012), we allowed to use a *speech pause* as a synonymous of open/closed-parenthesis items. Moreover, in order to experiment both with parentheses and pauses in the understanding of a mathematical sentence, we decided to implement three distinct parenthesization strategies, called *parenthesis*, *pause*, and *smart*. In the *parenthesis strategy*, all the necessary parentheses are inserted in the sentence plan.

[2]We have not been able to find any scientific reference on this point.

Note that a parenthesis has to be considered necessary with respect to the inverted precedence order hypothesis stated above. In the *pause strategy*, all the necessary pauses are inserted in the sentence plan. In the *smart strategy*, all the necessary parentheses are inserted in the higher nodes of the sentence plan, and the necessary pauses are inserted close to the leaves of the sentence plan. This is a hybrid strategy that combines parentheses and pauses in order to have a less verbose mathematical sentence.

2.2 NLG for spoken mathematics

In order to produce a spoken mathematical sentences in Italian with the SimpleNLG-it realizer (Mazzei et al., 2016), we needed to account for the construction of a domain specific lexicon for the field of the mathematical analysis. SimpleNLG-it is the Italian porting of the SimpleNLG realizer, that was originally designed only for English (Gatt and Reiter, 2009). As default Italian lexicon, SimpleNLG-it uses a basic vocabulary of around 7000 words, that is a *simple* lexicon studied to be perfectly understood by most Italian people (Mazzei, 2016; Conte et al., 2017; Ghezzi et al., 2018). However, for this specific project we needed to augment the basic lexicon with both (i) a mathematical specialized lexicon, that contains both new lexical entries (as `arcotangente`, *arctangent*), and (ii) new values for lexical entries which are yet in the basic lexicon (as the value *noun* for the part of speech of the lemma `integrale`, *integral*). This specialized lexicon contains 113 entries which are mostly categorized as nouns (e.g. `logaritmo`, *logarithm*), verbs (e.g. `intersecare`, *intersect*), adjective (e.g. `iperbolico`, *hyperbolic*). In the lexicon, there

are only two new instances of adverbs (that are `relativamente` and `propriamente`, *relative, properly*), and only one instance of "prepositional locution" (that is `tale che`, *such that*). Finally, we added specific lexical items to realize both parenthesis (that are `parentesi aperta` and `parentesi chiusa`, *open/closed parenthesis*) and speech pause. This latter item will be finally realized by using the SSML (Speech Synthesis Markup Language) tag `<break/>`, that can be processed by many speech synthesis engines[3].

The actual version of the mathematical sentence generator has been interfaced with two speech synthesis engines, that are the web service provided by the IBM-Watson framework[4] (*W-engine* henceforth), and the Espeak API[5] (*E-engine* henceforth). W-engine is a commercial, closed software based on deep learning, while E-engine is a free, open-source software based on formant synthesis algorithms. Note that for not visual impaired people W-engine sounds more fluent but, in contrast, for visual impaired people E-engine sounds more familiar since it is used by a widespread free screen reader.

3 Evaluation

In order to have a first evaluation of the generation system, we built a web-based test explicitly designed for visually impaired people. We designed a questionnaire composed by a 6 multiple choices questions concerning personal data, a core of 25 open questions each one concerning the listening of a mathematical sentence and its comprehensibility, 1 Likert-scale question globally comparing LaTeX and system comprehensibility, 1 open question for free comments.

The 25 core questions have a all the same schema: there is a audio file encoding a mathematical sentence and there is a open form for transcribing it. In the compilation instructions, we asked the users to fill this section by using "LaTeX or with other non ambiguous formal representation". The mathematical expressions obtained have been manually translated to CMML for evaluation. We implemented the questionnaire by using the Google Form framework, that was preliminarily judged accessible by a blind person.

ID	Formula
$E1$	$A \times B = \{(x, y) \mid x \in A, y \in B\}$
$E4$	$x > b \Longrightarrow \lvert f(x)\rvert < M$
$E6$	$\lim_{x \to x_0} \left\{ \frac{f(x) - f(x_0)}{x - x_0} - f'(x_0) \right\} = 0$
$E8$	$\int \frac{1}{\sqrt{m^2 - x^2}} \mathrm{d}x = \arcsin \frac{x}{m} + c$
$E10$	$\lim \left(1 + \frac{1}{n}\right)^n = e$

Table 2: The five mathematical expressions used for experimentation.

In this paper we discuss the results of 10 core questions of the questionnaire that have been created by using the 5 mathematical expressions belonging to the Table 2. We use the W-engine to build 5 mathematical sentences and the E-engine to build other 5 mathematical sentences. Note that we change the names of the variables in the two set of sentences.

In order to score the comprehension of the user we decided to use the *SPICE* (Anderson et al., 2016) metric. SPICE is obtained by computing the F-score of the overlap between two trees: the overlap is measured by decomposing trees in typed elementary substructures, that are operands, operators and their relations. For instance, the expression $x - 1$ is decomposed as $\{1, x, \textit{minus}, (\text{op: } \textit{minus}, \text{first: } x), (\text{op: } \textit{minus}, \text{second: } 1)\}$ (cf. (Anderson et al., 2016) for more details). For the experimentation, we recruited 4 visually impaired people with personal invitation without any rewards. All users are Italian mother tongue, have a good knowledge of mathematical analysis and have a bachelor degree (only one related to mathematics).

In Table 3 we reported the averaged values of SPICE for W-engine and E-engine. A first view of data seems suggest a preference for the E-engine, but there is not a significant effect on the performance of the system: by applying the t-test we obtained for 0.08 (two-tailed p-value), indicating no statistical significance. So, new experiments with more trials and users are necessary to statistically confirms the preference of for the E-engine.

In Table 4, we report the The distribution of the answers in Likert scale for the question of the web form concerning comprehensibility, that is "Quanto sei d'accordo con la frase: - La frase pronunciata è facile da capire -" (*How much do*

[3] `https://www.w3.org/TR/speech-synthesis11/`

[4] `https://www.ibm.com/watson/services/text-to-speech/`

[5] `http://espeak.sourceforge.net`

Engine	U1	U2	U3	U4
W-engine	0.96 (0.06)	0.95 (0.12)	0.97 (0.06)	0.97 (0.06)
E-engine	0.99 (0.03)	0.99 (0.03)	0.97 (0.04)	0.97 (0.04)

Table 3: The averaged SPICE measures and standard deviations for the speech synthesis W-engine and E-engine.

Engine	U1	U2	U3	U4
W-engine	4.60 (0.55)	5.20 (1.10)	4.00 (0.71)	4.00 (1.22)
E-engine	4.60 (0.89)	4.00 (1.73)	5.60 (1.14)	4.40 (1.52)

Table 4: The distribution of the answers in Likert scale $(1-7)$ for the question concerning comprehensibility.

you agree with the sentence: - The pronounced sentence is easy to understand -"). The value 1 corresponds to "per nulla" (*nothing*), the value 7 corresponds to "completamente" (*completely*). It seems from data that there is not notable difference between the perceived comprehensibility of the W-engine with respect to the E-engine and the t-test we obtained for the Likert score is 0.67 (two-tailed p-value).

4 Conclusion

In this paper we have presented a study on the generation of mathematical sentences, i.e. natural language sentences encoding mathematical expressions[6]. In particular, we have described the main features of the system and the a first experimentation centred on the evaluation of two distinct speech engine. The results of the experimentation suggests a good performance of the formant-based synthesis engine with respect to the neural-network base synthesis engine. However, more data is necessary to achieve statistical significance.

In future work we intend to repeat the evaluation of the system for Italian with a larger number of users and to repeat the experiment by using English lanaguage too.

References

[Ahmetovic et al.2018] Dragan Ahmetovic, Tiziana Armano, Cristian Bernareggi, Michele Berra, Anna Capietto, Sandro Coriasco, Nadir Murru, Alice Ruighi, and Eugenia Taranto. 2018. Axessibility: A latex package for mathematical formulae accessibility in pdf documents. In Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS '18, pages 352–354, New York, NY, USA. ACM.

[Anderson et al.2016] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: semantic propositional image caption evaluation. CoRR, abs/1607.08822.

[Cervone2012] Davide Cervone. 2012. Mathjax: A platform for mathematics on the web. Notices of the American Mathematical Society, 59, 02.

[Chang1983] Lawrence A. Chang. 1983. Handbook for spoken mathematics (larry's speakeasy). Lawrence Livermore Laboratory, The Regents of the University of California., 1.

[Conte et al.2017] Giorgia Conte, Cristina Bosco, and Alessandro Mazzei. 2017. Dealing with italian adjectives in noun phrase: a study oriented to natural language generation. In Proceedings of the Fourth Italian Conference on Computational Linguistics (CLiC-it 2017), Rome, Italy, December 11-13, 2017., December.

[Ferres and Fuentes Sepúlveda2011] Leo Ferres and José Fuentes Sepúlveda. 2011. Improving accessibility to mathematical formulas: the wikipedia math accessor. In Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, W4A 2011, Hyderabad, Andhra Pradesh, India, March 28-29, 2011, page 25.

[Fuentes Sepúlveda and Ferres2012] José Fuentes Sepúlveda and Leo Ferres. 2012. Improving accessibility to mathematical formulas: The wikipedia math accessor. New Rev. Hypermedia Multimedia, 18(3):183–204, September.

[Gatt and Krahmer2018] Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. J. Artif. Intell. Res., 61:65–170.

[Gatt and Reiter2009] Albert Gatt and Ehud Reiter. 2009. SimpleNLG: A Realisation Engine for Practical Applications. In Proceedings of the 12th European Workshop on Natural Language Generation, ENLG '09, pages 90–93, Stroudsburg,

[6]The described system can be freely downloaded at `https://bitbucket.org/tesimagistralemonticone/formula-to-speech/`

PA, USA. Association for Computational Linguistics.

[Ghezzi et al.2018] Ilaria Ghezzi, Cristina Bosco, and Alessandro Mazzei. 2018. Auxiliary selection in italian intransitive verbs: A computational investigation based on annotated corpora. In Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), pages 1–6, Berlin. CEUR.

[Hutchins and Somer1992] W. John Hutchins and Harold L. Somer. 1992. An Introduction to Machine Translation. London: Academic Press.

[Mazzei et al.2016] Alessandro Mazzei, Cristina Battaglino, and Cristina Bosco. 2016. SimpleNLG-IT: adapting SimpleNLG to Italian. In Proceedings of the 9th International Natural Language Generation conference, pages 184–192, Edinburgh, UK, September 5-8. Association for Computational Linguistics.

[Mazzei2016] Alessandro Mazzei. 2016. Building a computational lexicon by using SQL. In Pierpaolo Basile, Anna Corazza, Francesco Cutugno, Simonetta Montemagni, Malvina Nissim, Viviana Patti, Giovanni Semeraro, and Rachele Sprugnoli, editors, Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016), Napoli, Italy, December 5-7, 2016., volume 1749, pages 1–5. CEUR-WS.org, December.

[Miller2007] Bruce Miller. 2007. LaTeXML: A LaTeX to XML converter.

[Pandolfi2013] Luciano Pandolfi. 2013. ANALISI MATEMATICA 1. Dipartimento di Scienze Matematiche “Giuseppe Luigi Lagrange”, Politecnico di Torino.

[Raman1996] T. V. Raman. 1996. Emacs-peak—direct speech access. In Proceedings of the Second Annual ACM Conference on Assistive Technologies, Assets ’96, pages 32–36, New York, NY, USA. ACM.

[Reiter and Dale2000] Ehud Reiter and Robert Dale. 2000. Building Natural Language Generation Systems. Studies in Natural Language Processing. Cambridge University Press.

[Soiffer2016] Neil Soiffer. 2016. A study of speech versus braille and large print of mathematical expresisons. In Lecture Notes in Computer Science, volume 9758, Berlin. Springer.

[Sorge et al.2014] Volker Sorge, Charles Chen, T. V. Raman, and David Tseng. 2014. Towards making mathematics a first class citizen in general screen readers. In Proceedings of the 11th Web for All Conference, W4A ’14, pages 40:1–40:10, New York, NY, USA. ACM.

[Waltraud Schweikhardt2006] Nadine Jessel Benoit Encelle Margaret Gut Waltraud Schweikhardt, Cristian Bernareggi. 2006. Lambda: A european system to access mathematics with braille and audio synthesis. In Lecture Notes in Computer Science, volume 4061, Berlin. Springer.