Ziffers - Numbered Notation for Algorithmic Composition

Miika Alonen
Department of Computer Science. Aalto University, Finland
miika.alonen@aalto.fi

Raphaël Maurice Forment
ECLLA. Université Jean Monnet, France
a@b.c

ABSTRACT

???

1 INTRODUCTION: ZIFFERS, QUICKLY EDITABLE NUMBER BASED NOTATION

Ziffers – named and inspired by the older Ziffersystem (Warkentin 2022) — proposes a system focused on the conciseness and expressiveness of musical notation. Ziffers is designed to enable generation and transformation of musical patterns (McCormack et al. 1996; McLean 2020) in minimal amounts[2] of time and typing. To do so, Ziffers uses a text-based syntax for representing various short-hand symbolic notations used in music theory and concepts in programming languages. The need for concise and succinct notation in the context of live coding performance is a well-known constraint (Roberts and Wakefield, 2018) that gave rise to many practices and techniques aiming to reduce frictions in the conversational feedback loop between the musician and its programming interface (REFERENCE). Domain specific languages (DSLs) and the use of terse mini notations have become an important design pattern in the conception of live coding interfaces (Hoogland 2019; ADD OTHER), and can also be noted to be a key concept in the larger realm of exploratory or conversational open-ended programming where reactivity and fast decision-making is of prime importance to achieve a creative state of flow (Nash 2015). MISSING (Beth Kery & Myers 2018 REFERENCE?)

Ziffers is designed as a terse and platform independent syntax capable of embedding algorithmic and generative processes at the notational level. Basic notation tokens denoting pitch, rhythm or expression marks form the base notation, enriched by a set of generative operators and tokens denoting algorithmic transformations of that first layer of notation. Implementation of the Ziffers system is specific to each targeted platform, in conformity with the notation described in this article and the documentation for the prototype implementation. The first prototype of Ziffers has been created in 2018 (Alonen 2018) as an inline parser for Sonic Pi. It was thought as an attempt to speed up the process of writing melodic lines on-the-fly, taking advantage of the large number of predefined methods and compositional helpers offered by Sonic Pi's rich internal library. Its design aimed at resolving the dissociation between pitch and rhythm imposed by Sonic Pi data structure and imperative-oriented programming model. After a few years of evolutionary prototyping, Ziffers 2.0 was released in 2022 (Alonen 2022), including a new parser aiming at extending Ziffers capabilities with new operators, nested structures and better support for stochastic and aleatoric composition.

1.1 Running pandoc

Pandoc is software which turns text written in markdown into a beautiful looking document, complete with references. You will need to run it to create PDF documents of your paper for checking and uploading for peer review.

You may download pandoc for all major operating systems (including MS Windows, Apple Mac OS and GNU/Linux) from the following website: http://pandoc.org

As an alternative to the above downloads, on OS X only, the homebrew package manager can be used to install pandoc: http://brew.sh/

If you use homebrew to install on OS X you will need to install the pandoc package as follows:

```
brew update
brew install pandoc
```

To produce PDF files you will need to have LaTeX installed, as well as pandoc. See the pandoc website for installation instructions: http://pandoc.org/installing.html. LaTeX is used internally, you will not have to edit any LaTeX documents.

To render your markdown source as HTML, open a terminal window, change into the folder where the template is and run the following command:

```
pandoc --template=pandoc/iclc.html --citeproc --number-sections iclc2023.md -o iclc2023.html
```

To produce a PDF document, make sure you have LaTeX installed (see above), and run the following:

```
pandoc --template=pandoc/iclc.latex --citeproc --number-sections iclc2023.md -o iclc2023.pdf
```

For a higher quality output, add the option `--latex-engine=xelatex` to the above. You will need the Inconsolata and Linux Libertine opentype fonts installed.

An example Makefile is also provided to run these commands for you. If you have *make* installed, you can use it to build the pdf files.

1.2 Bibliographic references

Pandoc accepts bibliographic databases in a range of formats, so make sure you have the right extension on your file.

Table 1: Supported bibliography formats with file extension.

Format	File extension
MODS	.mods
BibLaTeX	.bib
BibTeX	.bibtex
RIS	.ris
EndNote	.enl
EndNote XML	.xml
ISI	.wos
MEDLINE	.medline
Copac	.copac
JSON citeproc	.json

Authors may be referenced in two ways; inline, e.g. Schwitters (1932) wrote the Ursonate sound poem, or in parenthesis, e.g. Ursonate is a sound poem (Schwitters 1932). Multiple references should be grouped together like so (Schwitters 1932; Miller 1956; Greenewalt 1946).

The pandoc command given in the above section will automatically render your references according to Chicago author-date style.

At the head of the markdown source file for this template, you will see an entry for "bibliography" that points to the file references.bib. Here you'll find examples of bibliography entries in BibLaTex format, including examples for articles, books, book chapters and items from conference proceedings.

1.3 Code

We have chosen a single column layout to better support code examples without having to break lines. The following shows how to include a code example with syntax highlighting:

```
d1 $ every 3 (iter 4) $ brak $ "bd [sn [[sn bd] sn]]*1/3"
```

For more information please visit this page: [https://hackage.haskell.org/package/skylighting]

Figure 1: *A descriptive caption should be given for all figures, understandable without reference to the rest of the article.*

1.4 Figures

Images should be included as figures, with captions provided and formatted as shown in Figure 1. Be prepared for the page layout and image size to be changed during the editing and layout process, and consider this when referring to the figures in the text.

2 Conclusion

We look forward to receiving your completed papers. Submission is through the online peer review system at https://iclc2023.creativecodingutrecht.nl/openconf.php only. Do not send papers directly by e-mail. In all cases, please submit a PDF version of your paper for peer review. At a later stage in preparing the proceedings, we may ask for the markdown or Word versions of your paper.

2.1 Acknowledgments

At the end of the Conclusions, acknowledgements to people, projects, funding agencies, etc. can be included after the second-level heading “Acknowledgments”.

References

10 Alonen, Miika. 2018. *Ziffers: Numbered notation for algorithmic composition* (version 0.1). https://github.com/amiika/ziffers/commit/c12516d2b1604836925cbe829e488e033578405c.

———. 2022. *Ziffers: Numbered notation for algorithmic composition* (version 2.0). https://github.com/amiika/ziffers.

Greenewalt, Mary H. 1946. *Nourathar, the Fine Art of Light Color Playing*. Philadelphia. Pa. Westbrook.

McCormack, Jon et al. 1996. “Grammar Based Music Composition.” *Complex Systems* 96: 321–36.

McLean, Alex. 2020. “Algorithmic Pattern.” In *Proceedings of the 20th Conference on New Interfaces for Musical Expression. Birmingham, UK*.

Miller, G. A. 1956. “The Magical Number Seven Plus or Minus Two: Some Limits on Our Capacity for Processing Information.” *Psychological Review* 63 (2): 81–97.

Nash, Chris. 2015. “The Cognitive Dimensions of Music Notations.”

Schwitters, Kurt. 1932. “Ursonate.” *Merz* 24.

Warkentin, Emily Grace. 2022. “The Story of the Ziffersystem and the Russian Mennonites: Counting Blessings.”