COMPUTATION AND VISUALIZATION OF DIFFERENCES BETWEEN TWO XML MUSIC SCORE FILES

Francesco Foscarin[1], **Florent Jacquemard**[2], **Raphaël Fournier-S'niehotta**[1]
[1]CNAM, Paris [2]INRIA, Paris
{francesco.foscarin,fournier}@cnam.fr, florent.jacquemard@inria.fr

EXTENDED ABSTRACT

The demonstration is about a tool for the computation of the differences between two XML scores and the intuitive visualization of those differences on the two scores side-to-side.

Our goal is to implement for music scores a technique similar to the Unix diff utility [5, 6], whose purpose is to identify differences between two text files relatively similar (typically two versions of the same file). The comparison is performed at the granularity level of lines with the *Longest Common Subsequence algorithm* (LCS), or at the granularity level of characters, with edit-distances. The output is a list of differences between the two files that corresponds to our intuitive notion of difference. With the large diffusion of digitized music scores in the last years, it would be extremely beneficial to evaluate and display the differences between two music scores in a similar way. Straightforward applications are version control systems and collaborative edition of scores, as well as speeding-up tasks that require human supervision, such as the visual inspection of the outcome of optical music recognition (OMR) systems.

A music score is inherently more complex than a text file and the computation of differences must be adapted accordingly. Two music scores can be compared at the *musical level* or the *music notation level*. Most of the literature so far focuses on the first level, *i.e.* on the *musical content*: the sequence of sound events described in the score, independently from any encoding or rendering concern. The comparison is made using some extended *edit-distances* [1, 3, 7–9], or employing tree models [2, 11]. It is particularly suited for tasks like automatic evaluation of music transcription systems and there is no effort to show the differences in a easy-to-read format for users. In this work, instead, we are interested in the *musical notation* level, *i.e.* how the music is intended to be displayed in the score. This involves aspects such as note figures, beams, tuplets, ties/dots, pitches spelling, *etc.* that can be found in XML scores along with the musical content.

To compute score difference at this level, we employ a representation of the music notation content of one measure as a tree, and we run an edit distance algorithm yielding a list of differences between two scores. The tree-based representation acts as a canonical model for disambiguation of the content for XML score formats and allows to decouple our approach from the exact file format (in general MusicXML or MEI); anyway for practical reason of implementation we chose to work mainly with scores in MEI format [12]. The computation of the difference list works on two steps, for performance reasons: first we compare lists of entire measures with a LCS algorithm, and second, we compare the content of measures identified as different. This approach is similar to a text diff tool that first finds the different lines and then explores the differences between the words of each different line. In order to comply with nested beamings and nested tuplets in music notation, our edit distance consider both string-edit-distance operations (insertion, deletion and substitution) and tree-edit-distance operations (node-insertion, node-deletion, node-descent). More details about the model and the algorithm are discussed in [4].

In Figure 1, we present a typical scenario of version control, with the second score being a new extended version of the first. We use three colors to represent different edit operations: green for insertions, red for deletions and yellow for substitutions. We can display differences between whole measures (*e.g.,* measure 4 in the second score is entirely inserted) and inside the measures (*e.g.,* the tie on measure 2 in the first

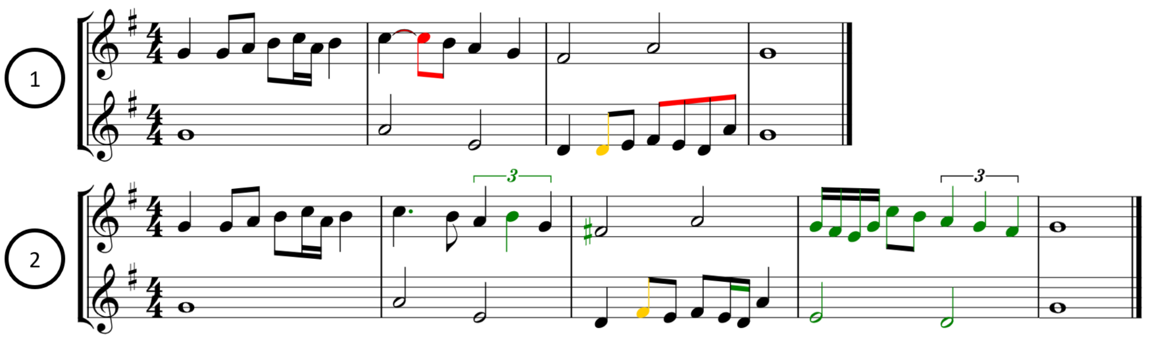

Figure 1. Visualization of the differences between two similar music scores.

score). Note that we are considering the differences at a music notation level, so we highlight things even if they would "sound" the same (*e.g.,* the accident in measure 3 on the second score or the tie in the first score vs the dot in the second score). The differences between tree-edit-operation and string-edit-operation highlights different elements in the scores: notes or note attributes (accidents, ties, note-heads, dots) for string-edit-distance and beams and irregular grouping for tree-edit-distance. The visualization is performed using Verovio [10] and works for polyphonic scores and for multiple instruments; it was made for our algorithm, but it may be extended to present other metrics and should also be easy to integrate into score editors.

ACKNOWLEDGMENTS

This work was partially funded by the ANR project MuNIR.

REFERENCES

[1] Julien Allali, Pascal Ferraro, Pierre Hanna, Costas Iliopoulos, and Matthias Robine. Toward a general framework for polyphonic comparison. *Fundamenta Informaticae*, 97(3):331–346, 2009.

[2] José F Bernabeu, Jorge Calera-Rubio, José M Iñesta, and David Rizo. Melodic identification using probabilistic tree automata. *Journal of New Music Research*, 40(2):93–103, 2011.

[3] Andrea Cogliati and Zhiyao Duan. A metric for music notation transcription accuracy. In *Proc. of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, pages 407–413, 2017.

[4] Francesco Foscarin, Raphaël Fournier-S'niehotta, and Florent Jacquemard. A diff procedure for XML music score files. In *Proc. of the 6th Int. Conf. on Digital Libraries for Musicology (DLfM)*. ACM, 2019.

[5] Paul Heckel. A technique for isolating differences between files. *Communications of the ACM*, 21(4):264–268, 1978.

[6] James Wayne Hunt and M Douglas MacIlroy. An algorithm for differential file comparison. Technical report, Bell Laboratories Murray Hill, 1976.

[7] Kjell Lemström. *String Matching Techniques for Music Retrieval*. PhD thesis, University of Helsinki, Department of Computer Science, 2000.

[8] Andrew Mcleod and Mark Steedman. Evaluating automatic polyphonic music transcription. In *Proc. of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[9] Marcel Mongeau and David Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24(3):161–175, 1990.

[10] Laurent Pugin, Rodolfo Zitellini, and Perry Roland. Verovio: A library for engraving MEI music notation into SVG. In *ISMIR*, pages 107–112, 2014.

[11] David Rizo. *Symbolic music comparison with tree data structures*. Universidad de Alicante, 2010.

[12] Perry Roland. The music encoding initiative (MEI). In *Proceedings of the First International Conference on Musical Applications Using XML*, volume 1060, 2002.