MIR IN ENP – RULE-BASED MUSIC INFORMATION RETRIEVAL FROM SYMBOLIC MUSIC NOTATION

Mika Kuuskankare
Sibelius Academy
Centre for Music and Technology
mkuuskan@siba.fi

Mikael Laurson
Sibelius Academy
Centre for Music and Technology
laurson@siba.fi

ABSTRACT

Symbolic music information retrieval is one of the most underrepresented areas in the field of MIR. Here, symbolic music means common practice music notation–the musician readable format. In this paper we introduce a novel rule-based symbolic music retrieval mechanism. The Scripting system–ENP-Script–is augmented with MIR functionality. It allows us to perform sophisticated retrieval operations on symbolic musical scores prepared with the help of the music notation system ENP.

We will also give a special attention to visualization of the query results. All the statistical queries, such as histograms, are visualized with the help of common music notation where appropriate. N-grams and more complex queries–the ones dealing with voice leading, for example–are visualized directly in the score.

Our aim is to demonstrate the power and expressivity of the combination of common music notation and a rule-based scripting language through several challenging examples.

1. BACKGROUND

Music (especially Classical music) is primarily a written tradition. Throughout the centuries musical compositions have been preserved in music notation. It is the most complete and widespread method that we know of for notating the complex and interrelated properties of a musical sound: pitch, intensity, time, timbre, and pace. [1] Common music notation is also an invaluable tool in the field of music information retrieval. In this paper we introduce a symbolic music retrieval mechanism based on a scripting language called ENP-Script [2] and music notation system ENP [3].

ENP-script is a rule-based object oriented scripting language that is here augmented with MIR functionality. The extensions allow the user to perform retrieval operations on scores prepared with the help of ENP and visualize the results in a meaningful way. ENP-Script allows us to define complex and musically relevant queries using it's pattern-matching language. It has a uniform and simple syntax. The structural elements of the score (e.g., notes, beats, measures, melodies, harmony, voice-leading, etc.) are accessed using a symbolic naming scheme where a collection of reserved keywords is used to denote the objects of interest.

On the score level the retrieval system is based on ENP's underlying music representation. ENP provides several interesting features in terms of the present application: (1) it can be used to store music using a wide range of notational styles (Western musical notation roughly from 17th century onward, including 20th century notation); (2) it can be used as a user-interface component allowing us to construct both eye-catching and functional visualizations of MIR data; (3) it provides access to its notational data structures, allowing the user to inspect the properties of the notational objects (e.g., time, pitch, duration); and (4) it provides a rich library of standard and user-definable expressions allowing us to annotate the score with analytical information [4]. One further detail of interest is that ENP scores can be written using both mensural (metric) and non-mensural (piano roll like) notation. The system described in this paper works without modifications on both types of notation. This makes it possible to use this system for applications dealing with contemporary music.

In this paper, we will also give a special attention to visualization. All the statistical queries, such as histograms, are visualized with the help of common music notation where appropriate. N-grams and more complex queries are visualized directly in the score. Both approaches allow us to associate the query results directly with the correct musical objects.

A few approaches have been introduced where the aim is to use some kind of symbolic notation as the basic for MIR queries (see, e.g., [5–13]. Most of these systems, however, primarily address large databases of music encoded in an array of formats, such as MIDI, Kern, MusicXML [14], etc. Most of the approaches can be seen as MIR tool chains comprising of several small or even larger utilities chained together. Humdrum is an example of such a system. In [13] Humdrum is even married with Perl [15] and LilyPond [16] to create kind of a meta tool chain. Other approaches use various music formats like GUIDO [17] in [6, 8, 9]; Kern in [5, 8]; and MusicXML in [18].

At the moment the presented retrieval system can be

seen as an Analytic/Production MIR System [19]. We concentrate predominantly on posing questions on a symbolic musical score rather than retrieving information from a large music database. Querying a database of all Bach Chorales, for example, is however not out of the scope of the present approach.

Our retrieval system is part of PWGL [20] which is freely available for Macintosh OS X (Intel and PPC versions) and for Windows at `www.siba.fi/PWGL`.

The rest of the paper is organized as follows. Section 2 illustrates how a collection of archetypal MIR assignments can be solved using our retrieval system. The examples are divided roughly into two categories: statistical and analytical. We end the paper with some concluding remarks and outlines for future work.

2. EXAMPLES

While mass queries can tell certain kinds of facts about the music in the database our approach emphasizes the musical meaningfulness of the query. On the one hand one might find out that Bach violated X times the rule Y in Z chorals. On the other hand one might want to reveal these cases in a musical score and study why this may have happened. To be able to do so requires a flexible searching mechanism and flexible notational and visualization tools.

In this section we present a collection of examples based on more or less archetypal MIR assignments. The section is divided roughly in two parts. In the first part we concentrate on statistical queries and more or less traditional visualizations. The second part, in turn, turns focus on more analytical queries and visualizes the results in the score. Altogether, we will address several subjects, e.g., histograms, counting, pitch-class set theory, rhythm, etc. Some of the case studies borrow shamelessly from the Humdrum example database presented at `http://music-cog.ohio-state.edu/Humdrum/`.

Each subsection is accompanied with a code example and potentially also a visualization of the result. It is not in the scope of this paper to give an exhaustive review of the Scripting syntax. The code examples require a little knowledge about Lisp programming language but they should be clear and simple enough to be followed by anyone with some background in programming.

Apart form the examples presented in this paper, many other types of queries could easily made with the current system including those about pitch-class set theory, voice-leading, word painting, harmonic analysis, etc.

The most important points of interest in the following examples are: (1) the terseness and expressivity of the query definitions, (2) the descriptiveness of the visualization, and (3) the overall versatility of the system.

2.1 Statistical Queries

2.1.1 Histograms

One of the prototypical MIR tasks is the histogram.

The musical score used as a starting point for Examples 1–4 is the guitar transcription by Andrés Segovia of the *Tango op. 156a* by Isaac Albéniz. The beginning of the score is shown in page 6 (Figure 6).

Example 1 shows the retrieval rule to generate a pitch histogram of the given score. The pattern given in line 1 means that the rule applies to every note object in the score, thus the traditional wild card `*`. `?1` is a variable to which every note in the score is bound one by one. `histogram` is a special MIR function that takes care of gathering the values and visualizing the result.

After the execution of the script the pertinent visualization method is called to generate the pitch histogram shown in Figure 1. Instead of the traditional horizontal bar graph we use here a vertical arrangement instead. The histogram values are also shown against a set of piano keys to give a better idea of the register (middle-C is highlighted using a shade of grey).

One special aspect of this particular histogram (including the ones shown in Figures 2 and 3) is that the result can be played back. Either as a whole or by selecting a subset of the result shown. Especially, in case of tonal music, an aural examination of the pitch histogram could among other things reveal potential problems in the integrity of the source material.

Furthermore, as the histogram is realized with the help of ENP, it itself can be scripted to select and highlight pitches above certain threshold, for example.

Example 1 An ENP-script collecting histogram values from a score.

```
1 (* ?1
2    (?if
3     (histogram :value (m ?1))))
```

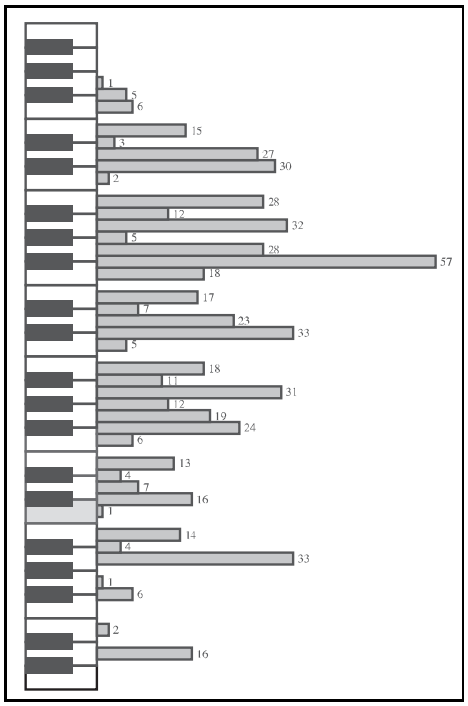


Figure 1. A traditional pitch histogram plotted using ENP as a visualization tool (Albéniz: Tango op. 156a).

2.1.2 Harmony Histogram

In addition to horizontal events (i.e., melody, as in Example 1) with ENP-Script it is also possible to access the vertical (harmonic) dimension of the score.

Scores are partitioned by 'harmonic slices' (resembling 'moments' in [5]). A harmonic slice is a vertical entity defined as a point in time when any note event begins or ends in any part. This structural component allows us to perform queries involving simultaneity.

Example 2 shows a script accessing the harmonic slices to produce a 'moment' histogram. This can be accomplished simply by introducing the keyword `:harmony` in the pattern matching part (see line 1). This instructs the script to access all the vertical elements of the score instead of the horizontal ones. Compared to Example 1 the change is minimal but the effect is dramatic.

Another point of interest is the form `(m ?1 :complete? t)` given at the end of line 1. This is in fact a condition and it is used here due to the implementation of the scripting engine and cannot be explained in-depth in the score of this paper. Suffice to say that in addition to accessing the total harmony (i.e., all the notes sounding at a given point in time) we can also access partial harmonic formations (e.g., subsets of the sounding harmony). As we are here interested only in the total harmonies hence the condition.

Furthermore, line 2 introduces yet another additional condition to skip any grace notes that are abundant in our example score. In line 4 we simply record the pitch values of the harmony given by `(m ?1)` as a list of midi values, e.g., (60 64 67).

Example 2 A script collecting histogram values of type harmony.

```
1 (* ?1 :harmony (m ?1 :complete? t)
2    (?if (unless (some #'grace-note-p
3                       (m ?1 :object t))
4           (histogram :value (m ?1)))))
```

The result is shown as a histogram with the relevant parts visualized using common practice notation. In the analysis same kind of chords are grouped together irrespective of register, or pitch spelling. That means that, e.g., all major chords are identified as equal (i.e., pitch-class set 3-11B).[1] Furthermore, in the histogram the so called prime form is shown. This is why, for example, the second to last entry, the seventh chord, is displayed in first inversion.

2.1.3 Counting

Our next example demonstrates the ability of the scripting mechanism to access only parts of the given score. Example 3 shows a simple counting script where we count the number of events in the score. Here, instead of counting all the events we restrict the search to measures 8–16 (see line 1). Our analysis reveals that there are 96 events combined in the measures 8–16 in the Tango by Albéniz. `quantity` in line 3 simply increments a counter when presented with a new event. Note, that this script counts all *note* events. We could similarly count all *chords* by inserting the keyword `:chord` after the variable `?1`.

[1] The system used here is similar to that of Forte except that the letters A or B are added to distinguish between two inversionally related transpositional set-classes, e.g., 3-11A is the minor triad, and 3-11B is its inversion, the major triad.

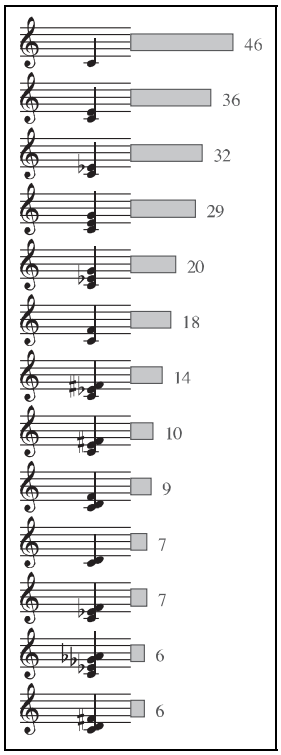


Figure 2. A 'moment' histogram (Albéniz: Tango op. 156a).

Example 3 A script counting the number of events in a score from measure 8 through 16.

```
1 (* ?1 :measures (8_16)
2    (?if
3     (quantity :value ?1)))
```

For comparison we give the following Unix script in Humdrum performing the equivalent operation:

```
yank -n = -r 8-16 Tango | census -k
```

2.1.4 Rhythmic Patterns

Next, we define a search based on the rhythmic dimension of the score. We aim to determine the most common rhythmic pattern spanning a measure. Example 4 shows the retrieval script. Once again our script has changed relatively little. The keyword `:measure` shown in the first line denotes that we want to access measure objects this time. Thus, the variable `?1` is here bound in succession to every measure object found in the score. As we want to access the whole rhythmic entity inside the measure we add once again the condition `(m ?1 :complete? t)`. This ensures that the body of the script is executed only when the rhythm for the entire measure is known. In line 4 we read the rhythmic definition cached by the system in a list form.

Example 4 A script for determining the most common rhythmic pattern spanning a measure.

```
1 (* ?1 :measure (m ?1 :complete? t)
2    (?if
3     (histogram :value
4          (read-key ?1 :rtm-pattern))))
```

We define here the rhythm histogram again using ENP and common music notation as it allows us to visualize the data in a straightforward manner. Figure 3 shows a part of the rhythm histogram displaying the actual rhythms in rhythm notation and the corresponding values as a bar graph.

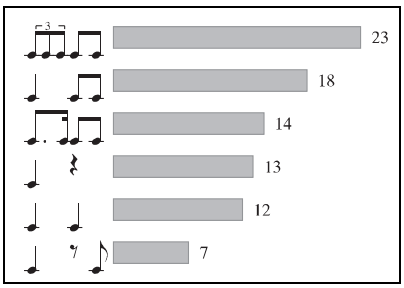


Figure 3. A per measure rhythm histogram (Tango op. 156a by Albéniz).

The above is certainly more descriptive than, let's say, producing print-outs like this:

```
((23 ((1 (1 1 1)) (1 (1 1))))
 (18 ((1 (1)) (1 (1 1)))) ...)
```

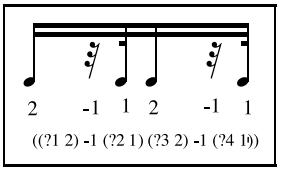
2.2 Analytical Queries

2.2.1 Rhythmic Patterns

Another kind of rhythmic query is presented in Example 5. Here, instead of counting all the possible rhythms we restrict our search to certain kind of rhythmic pattern. Furthermore, we have chosen to visualize the result of the query directly in the score. Figure 4 shows all the occurrences of our rhythmic pattern enclosed inside rectangles (drawn in red color in the original score). To save space we show here only the right-hand melody of the original composition (measures 1–8 of Humoresque in G♭ major by Antonin Dvořák).

The retrieval script is now a bit more complex than in the previous cases. The pattern matching part reads `(* ?1 ?2 ?3 ?4` as in this script we are now interested in the rhythmic formation between four consecutive notes. `match-rtm?` in line 3 is a special function that is used match the score rhythms against a given pattern.

Deciphering the rhythm matching syntax can be at first quite challenging and it requires some knowledge about the internal representation of ENP (see, e.g., [21]). We cannot give a comprehensive review of the format here. However, the line 5 defines a pattern where an event having a duration of 2 units is followed by a rest (a negative number) with a duration of 1 unit and another event with the duration of 1 unit. The durations are relative instead of absolute. The aforementioned pattern is repeated twice in row 5. This description is translated to the following rhythmic pattern:



Example 5 A script searching for a given four-note rhythmical pattern in a score.

```
1 (* ?1 ?2 ?3 ?4
2    (?if
3     (when (match-rtm?
4            (1
5             ((?1 2) -1 (?2 1) (?3 2) -1 (?4 1))))
6       (add-expression 'score-expression
7                       ?1 ?2 ?3 ?4
8                       :color :red))))
```

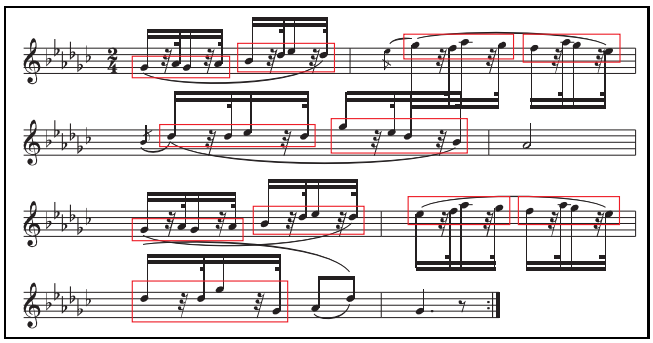


Figure 4. A specific rhythmic pattern visualized in the score (Humoresque in G♭ major by Antonin Dvořák).

2.2.2 N-grams

N-grams have been used extensively in MIR in both monophonic and polyphonic contexts (see, e.g., [19, 22]). In our next example we will show how to represent n-grams with the scripting language and how to mark them in a score. Naturally, instead of marking the n-grams in the score we could have recorded the statistical distribution of n-grams and display them in the same manner as displayed in Figure 2.

Example 6 shows the script definition. This particular script is demonstrating yet another interesting ability of the scripting system. Here, instead of defining a fixed pattern, as in the previous examples, we write the script in a dynamic fashion. The pattern matching part becomes now quite minimalistic again. The complexity of the search lies inside the script definition. In order to represent any n-gram the user is require to change only the list of context lengths enumerated in line 3. Here, we use a list `(2 3)` denoting di- and tri-grams respectively. The body of the script from line 4 onwards is written so that any sized n-grams can be found and visualized. In line 6 we add an ENP expression in the score displaying the extent of the n-gram and also the sizes of the consecutive intervals as can be seen in Figure 5.

3. DISCUSSION

Currently this retrieval system is not suitable for very large corpus of data. The mechanism used here requires that the whole score is read in the memory and that all the musical objects are instantiated. One could possibly provide an alternative loading mechanism that creates only the necessary data structures needed for the scripting language to operate. This would most likely guarantee much more shorter loading times and in turn facilitate larger searches.

Our system needs to support more input formats. Not only MIDI and the native ENP file formats but also at least MusicXML and perhaps even Kern.

However, at its present state the system is capable of performing very sophisticated and complex queries. Also the visualization capabilities are second to none. The ability to be able to mix common music notation with statistical graphics is also beneficial and allows us to represent the query result in a musician readable way.

The presented notational front-end combined with our scripting engine allows us to query, annotate and analyze musical scores and visualize the results in a manner probably not matched by any other software package.

4. ACKNOWLEDGMENTS

The work of Mika Kuuskankare and Mikael Laurson has been supported by the Academy of Finland (SA 114116 and SA 105557).

5. REFERENCES

[1] Curtis Roads. *The Computer Music Tutorial*. The MIT Press, Cambridge, Massachusetts, London, England, 1996.

[2] Mika Kuuskankare and Mikael Laurson. Intelligent Scripting in ENP using PWConstraints. In *Proceedings of International Computer Music Conference*, pages 684–687, Miami, USA, 2004.

[3] Mika Kuuskankare and Mikael Laurson. Expressive Notation Package. *Computer Music Journal*, 30(4):67–79, 2006.

[4] Mika Kuuskankare and Mikael Laurson. Annotating Musical Scores in ENP. In *International Symposium on Music Information Retrieval*, London, UK, 2005.

[5] Michael Droettboom. Expressive and efficient retrieval of symbolic musical data. In *International Symposium on Music Information Retrieval*, 2001.

[6] Andreas Kornstädt. The jring system for computer-assisted musicological analysis. In *ISMIR*, 2001.

[7] Shyamala Doraisamy. An approach towards a polyphonic music retrieval system. In *International Symposium on Music Information Retrieval*, 2001.

[8] Matthew J. Dovey. A technique for regular expression style searching in polyphonic music. In *International Symposium on Music Information Retrieval*, 2001.

[9] Holger H. Hoos, Kai Renz, and Marko Görg. GUIDO/MIR — an experimental musical information retrieval system based on GUIDO music notation. In *International Symposium on Music Information Retrieval*, 2001.

[10] Donncha Ó Maidín and Margaret Cahill. Score Processing for MIR. *International Symposium on Music Information Retrieval*, pages 59–64, 2001.

[11] Goffredo Haus, Maurizio Longari, and Emanuele Pollastri. A score-driven approach to music information retrieval. *J. Am. Soc. Inf. Sci. Technol.*, 55(12):1045–1052, 2004.

[12] David Huron. Music information processing using the humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):15–30, 2002.

[13] Ian Knopke. The perlhumdrum and perllilypond toolkits for symbolic music information retrieval. In *International Symposium on Music Information Retrieval*, pages 147–152, 2008.

[14] M. Good and G. Actor. Using MusicXML for File Interchange. In *Third International Conference on WEB Delivering of Music*, page 153, Los Alamitos, CA, 2003. IEEE Press.

[15] Wikipedia. Perl — wikipedia, the free encyclopedia, 2009. [Online; accessed 8-May-2009].

[16] Han-Wen Nienhuys and Jan Nieuwenhuizen. LilyPond, a system for automated music engraving. In *XIV Colloquium on Musical Informatics (XIV CIM 2003)*, Firenze, Italy, 2003.

[17] H. H. Hoos, K. A. Hamel, K. Renz, and J. Kilian. The GUIDO Music Notation Format - A Novel Approach for Adequately Representing Score-level Music. In *Proceedings of International Computer Music Conference*, pages 451–454, San Francisco, 1998.

[18] Goffredo Haus and Alberto Pinto. Mx structural metadata as mir tools. In *Sound and Music Computing '05*, 2005.

[19] J. Stephen Downie. *Evaluating a Simple Approach to Music Information Retrieval: Conceiving Melodic N-grams as Text*. PhD thesis, University of Western Ontario, 1999.

[20] Mikael Laurson, Mika Kuuskankare, and Vesa Norilo. An Overview of PWGL, a Visual Programming Environment for Music. *Computer Music Journal*, 33(1), 2009.

[21] Mika Kuuskankare and Mikael Laurson. Recent Developments in ENP-score-notation. In *Sound and Music Computing '04*, Paris, France, 2004.

[22] Shyamala Doraisamy. *Polyphonic Music Retrieval: The N-gram Approach*. PhD thesis, University of London, 2004.

Example 6 A script to visualize n-grams directly in an ENP score. Here, di- and tri-grams are shown. Simply by editing the parameter list shown in line 3 (n-grams) it is possible to visualize n-grams of any size. No other changes are necessary.

```
1 (* ?1
2    (?if
3     (let ((n-grams '(2 3)))
4       (iter (for n-gram in n-grams)
5             (?incase-let (intervals (m ?1 :L (1+ n-gram) :data-access :int :complete? t))
6               (add-expression 'group (m ?1 :L (1+ n-gram) :object t)
7                               :kind :bracket-at-end
8                               :info (format () "~{~3,@d ~^|~}"
9                                             intervals)))))))
```

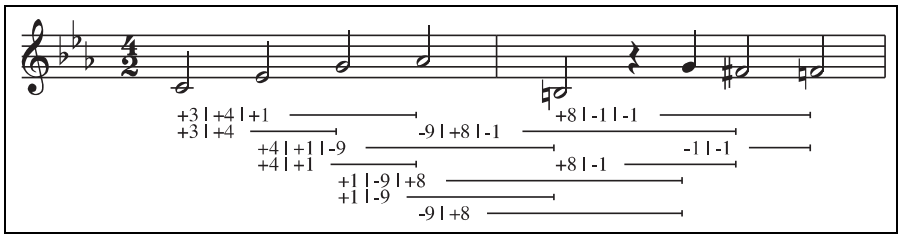


Figure 5. N-grams visualized directly in the score using a dynamically adapting script.

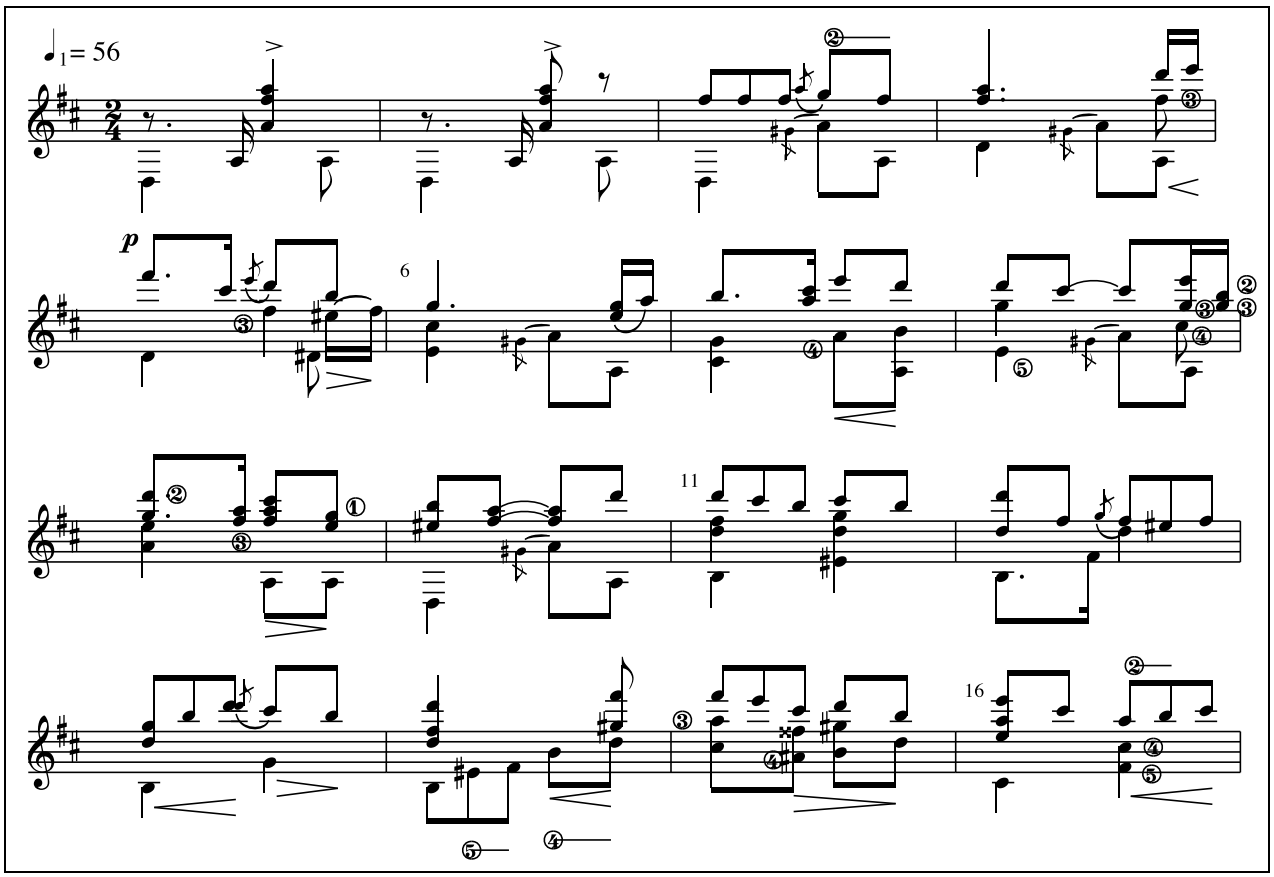


Figure 6. Tango op. 156a by Isaac Albéniz notated with ENP (transcribed for Guitar by Andrés Segovia).