Tokenization of MIDI Sequences

Yosuke Amagasu

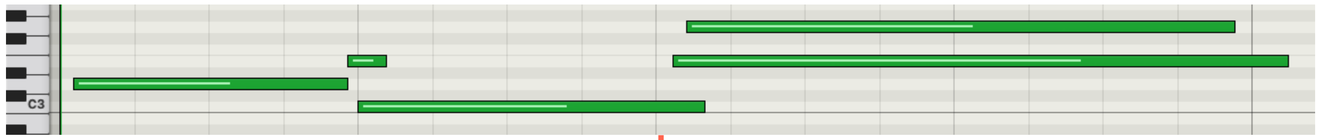
Florent Jacquemard

Masahiko Sakai

TENOR Zurich 2024

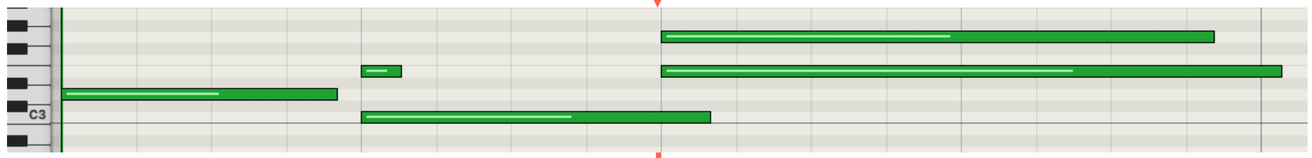
MIDI to score conversion

We consider the problem of converting a MIDI file into a music score.



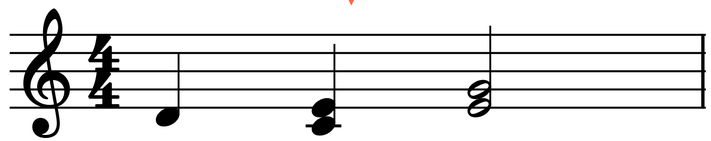
10 MIDI events
(note-on's and note-off's)

quantize (1/16 note)

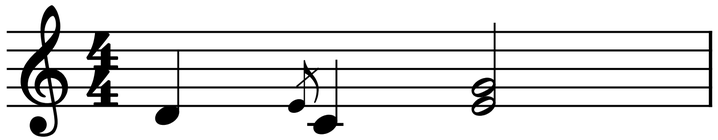


The note-on's are aligned to the grid (16th note)
The note-off's are left unchanged

open in MuseScore



Wouldn't we prefer the following?



- How to deal with ornaments and, distinguish a grace note + note from a chord?
- How to establish a correspondence between MIDI events and score elements (rest, note, chord, ornaments…) ?
  (many-to one or one-to-many or one-to-one)

Outline

1. concept of MIDI token = sequence of successive MIDI events
   correspondence: token ↔ music score element
   - Definition
   - Token type and validity

2. procedure of tokenization : optimal splitting of a MIDI file into tokens
   - Procedure
   - Application to Transcription

Focus : dealing with ornaments, assuming input is monophonic with chords

Examples of MIDI Tokens

token: sub-sequence of successive MIDI events in a larger MIDI sequence (MIDI file).

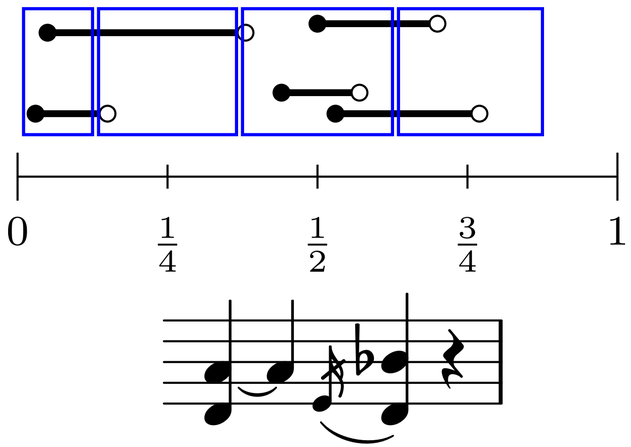
Every token corresponds to:

- simultaneous score elements: note w/wo ornament, chord w/wo ornament, rest,…
- one time position in score.

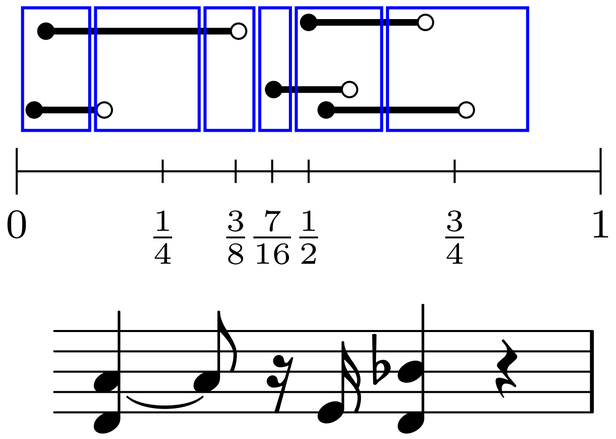
Intuitively, MIDI events within a token are aligned to the onset at the center of the token.

Various partitioning of a MIDI sequence into tokens will produce various notations

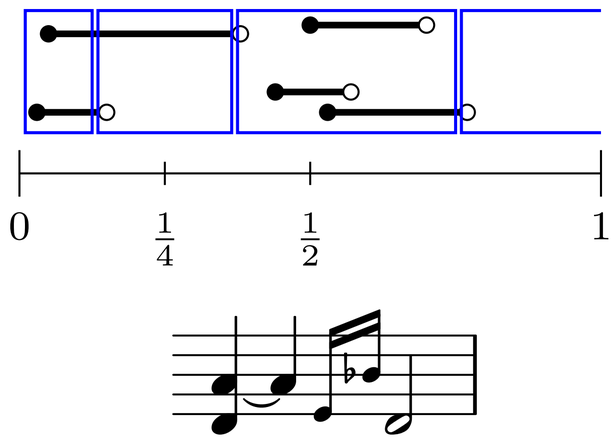
four tokens in one 4/4 measure:



six tokens in one 4/4 measure:



three tokens in one 4/4 measure,
the next continues in next measure



Role of MIDI event in Token

we consider only MIDI events (messages) of kind note-on or note-off

the matcher of a MIDI event e is denoted by e^{-1}

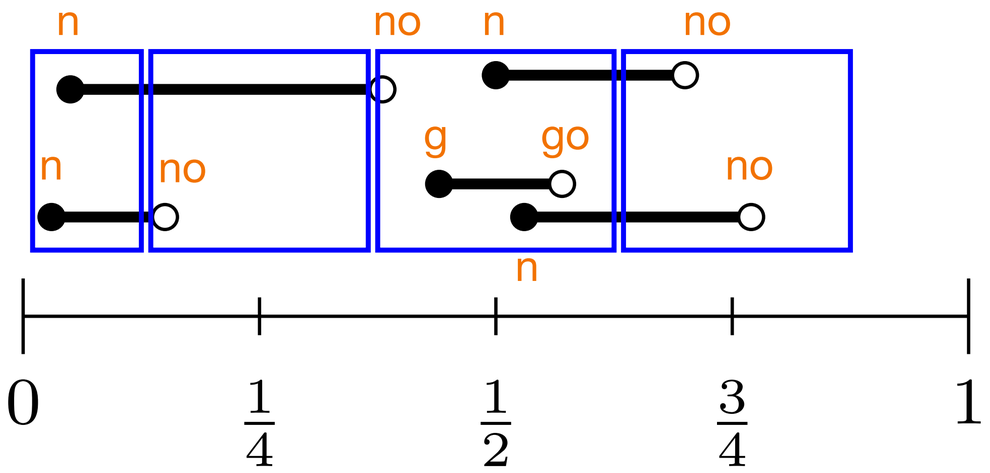
every event e in a token T is associated one role in T:

$T|_{\text{on}}$ = note-on events in token T

$T|_{\text{off}}$ = note-off events in token T

	$e^{-1} \in T$	$e^{-1} \notin T$
$e \in T|_{\text{on}}$	n (note)	g (grace note)
$e \in T|_{\text{off}}$	no	go

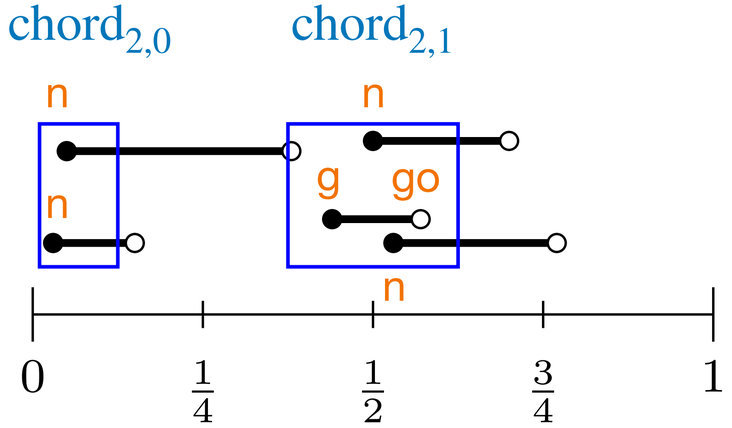
grace note = appogiatura, accaciatura, or part of ornament (mordent *etc*)



Token Type: Chord

The type of a token T is defined according to the role of its events:

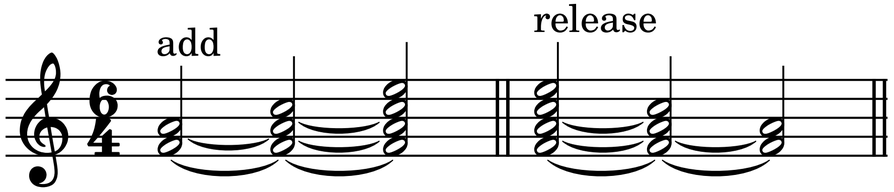
- a chord of size k, with with ornament of size p, denoted $\text{chord}_{k,p}$ contains exactly:
  - k note (n) events,
  - $p \geq 0$ grace notes (g) events and matching go events,
  - every g occurs before any n in T,
  - there are exactly k notes sounding after the token T.
- a note is a chord of size $k = 1$.



(*) note sounding after the token T:

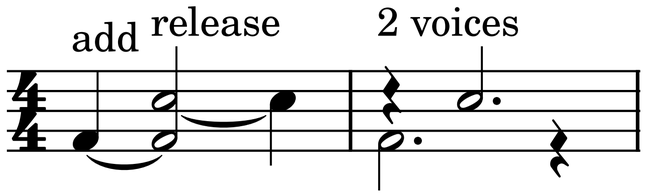
- there exists one event e in token T or in a previous token
- $e^{-1} \notin T$

E. Gould Behind Bars page 70



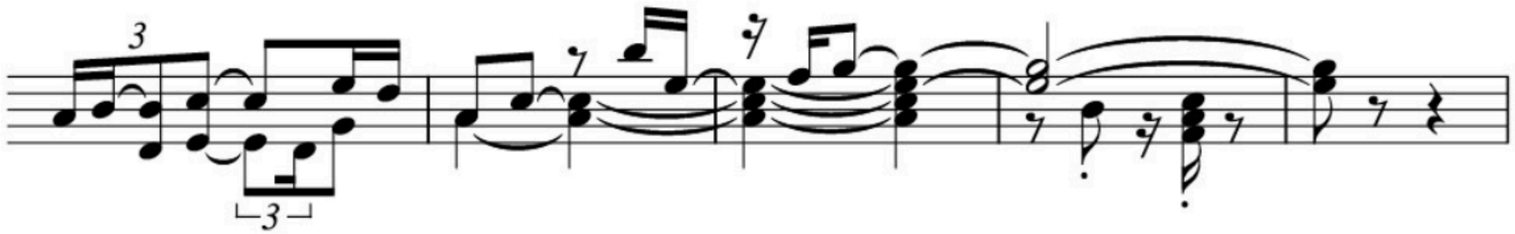
yet unsupported partial continuation

ambiguity



E. Gould Behind Bars page 310:

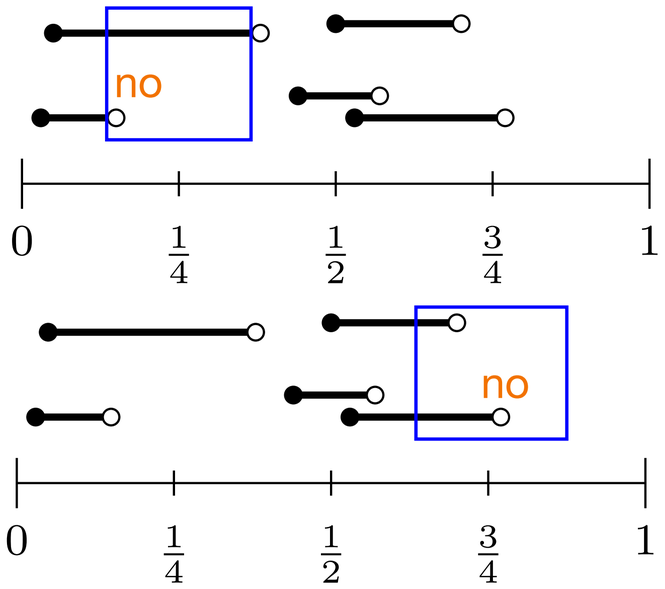
1. Separate stems are required only when an additional voice has an individual rhythm.
2. It is acceptable to combine additional voices onto a single stem
   so that a second stem direction can then be used for an entry of independent rhythm.
3. Reverse ties away from the entry of another voice.



Token Type: Rest

The type of a token T is defined according to the role of its events:

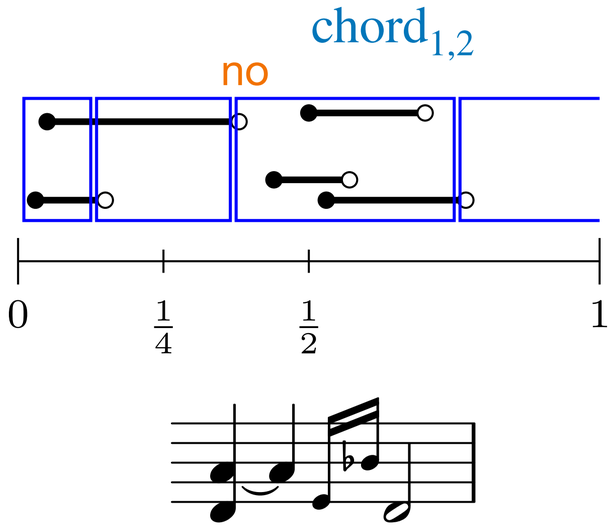
- T is a partial continuation if
  - it contains only events of role no,
  - there is at least one note sounding after T.
- T is a rest if
  - it contains only events of role no,
  - there is no note sounding after T.
- the empty token (continuation of note) is not considered as a token



Remarks

- an event of role no, in a token containing a note-on (n or g)
  is simply ignored.

That corresponds to a micro-rest, when midi recording
lacks of legato.
Micro rests are generally not displayed in a score.



- a token containing only
  - note-on events e of role g,
  - and the matching note-off events e^{-1} of role go

  has no valid type.
  It would be interpreted (in a score) as an ornament decorating a rest.

Outline

1. concept of MIDI token = sequence of successive MIDI events
   and correspondence: token ↔ music score element
   - Definition
   - Token type and validity

2. procedure of tokenization : optimal splitting of a MIDI file into tokens
   - Procedure
   - Application to Score Engraving

Focus : dealing with ornaments, assuming input is monophonic with chords

Tokenisation as a Combinatorial Optimisation Problem

Problem input

- sequence S of MIDI events (note-on and -off, timestamped)

search space

- sequences of tokens of S = partition $T_1, \ldots, T_n$ of S
- time points $\tau_1, \ldots, \tau_n$ for the alignment of $T_1, \ldots, T_n$

objective function (vaguely)

- minimise alignment cost
- *nice score* (readable) → *how to evaluate* ?

Grid based Tokenisation

Problem input

- sequence S of MIDI events (note-on and -off, timestamped)

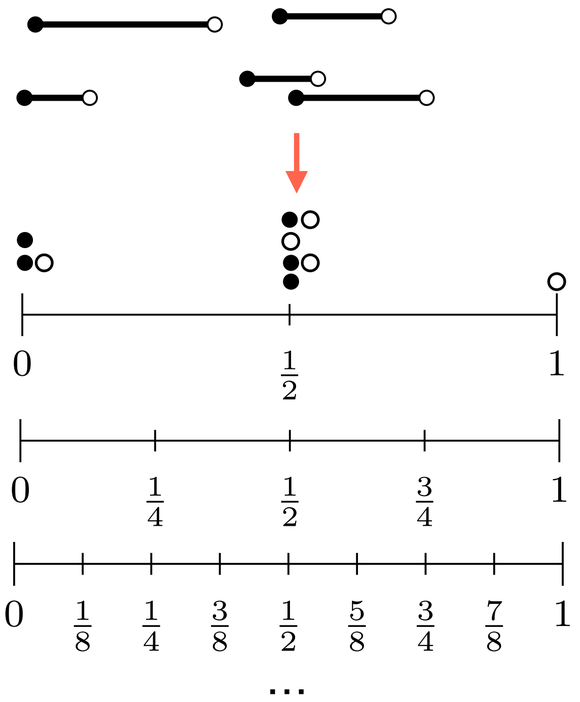
search space

- several grids and combinations (triplets etc)
  = parameters of score editor or DAW *etc*

→ it defines token $T_1, \ldots, T_n$ and alignment points $\tau_1, \ldots, \tau_n$

objective function (vaguely)

- (valid tokens)
- minimise alignment cost
- *nice score* (readable) → *how to evaluate* ?



alignment at the closest point in grid

- we can compute a (cumulated) distance of alignment of T_1 to τ_1 *etc*
- how to assess score *readability* ?

→ *the user choses manually the best combination (trial and error)*

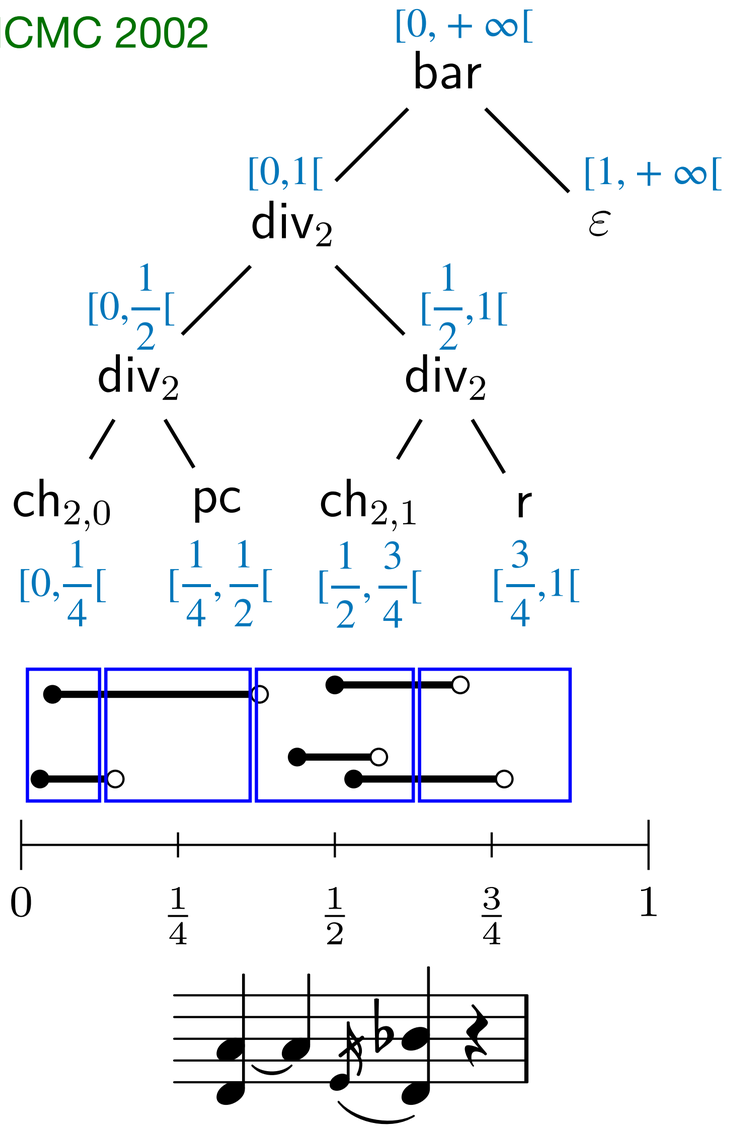
Rhythm Trees based Tokenisation

Rhythm Trees (RT) have been introduced for CAC, e.g.

- C. Agon, K. Haddad, G. Assayag, Representation and Rendering of Rhythm Structures, ICMC 2002
- OpenMusic

here we consider a particular kind of RT with

- leaves labelled with token types
- inner nodes are labelled with
  - bar : division of a time interval into 1 measure and the rest,
  - div_p : : division of a time interval into p sub-interval of equal duration



a time interval is associated to every node of the RT

alignment points = left bounds of intervals associated to leaves

Hence tokens can be derived from a RT

We can then infer the type of each token. and reject the RT in case of type failure.

Rhythm Trees based Tokenisation (2)

Problem input

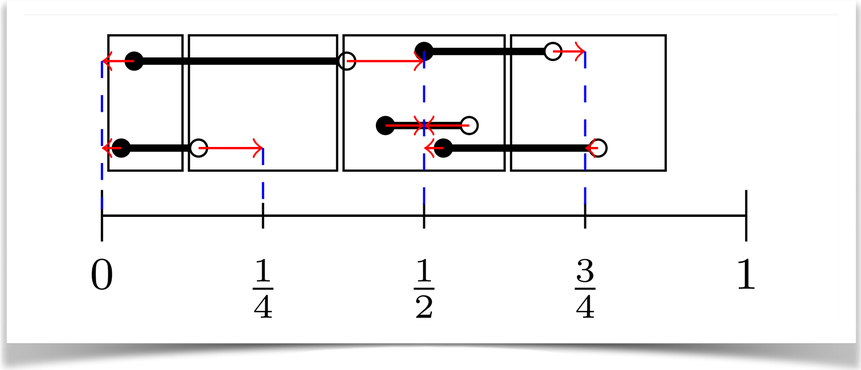
- sequence S of MIDI events (note-on and -off, timestamped)

search space

- RT

objective function

- minimal alignment cost
- minimal readability cost :
  cost associated to the RT by a grammar



alignment cost =
cumulated sum of alignment dist.
of every point

Optimisation: how to choose best tree?

→ quantitative parsing using a weighted grammar

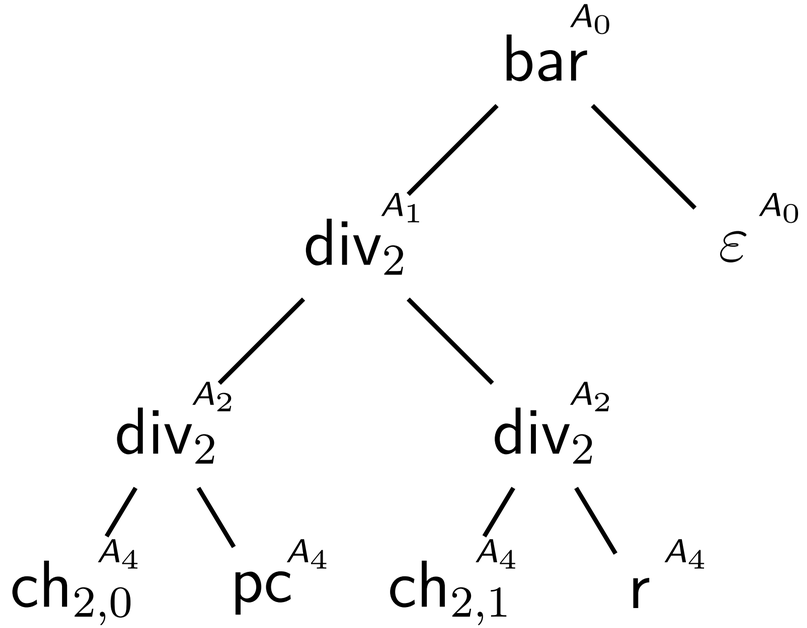
see

F. Foscarin, F. Jacquemard, P. Rigaux, and M. Sakai.

A Parse-based Framework for Coupled Rhythm Quantization and Score Structuring. MCM 2019

Tree grammar whose rules weighted in the cost domain

$$
\begin{aligned}
A_0 &\xrightarrow{w_0} \mathsf{bar}(A_1, A_0)\\
A_0 &\xrightarrow{w_1} \varepsilon\\
A_1 &\xrightarrow{w_2} \mathsf{div}_2(A_2, A_2)\\
A_2 &\xrightarrow{w_3} \mathsf{div}_2(A_4, A_4)\\
A_4 &\xrightarrow{w_4} \mathsf{ch}_{n,p}\\
A_4 &\xrightarrow{w_5} \mathsf{r}\\
A_4 &\xrightarrow{w_6} \mathsf{pc}\\
&\vdots
\end{aligned}
$$



$$weight_{tree} = w_0 + w_2 + w_0 + w_3 + w_3 + w_4 + w_5 + w_4 + w_6$$

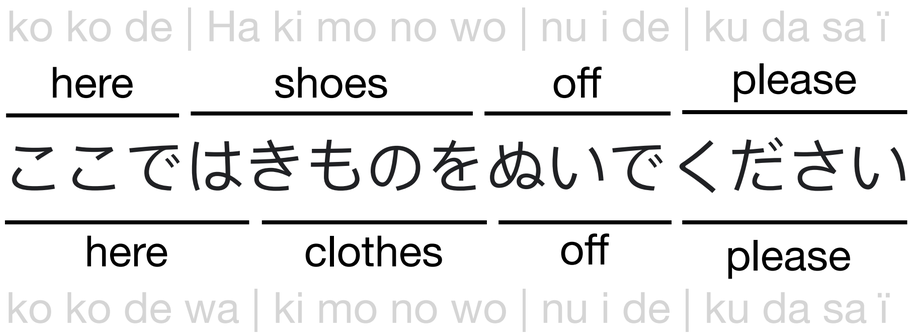
from the derivation of a tree by the grammar, we deduce a readability cost

Tokenisation (related)

other use of the term *tokenization* in literature

NLP (training of statistical language models)

tokenization = building sequential encodings of character sequences into integer vectors,
using a dictionary of tokens



MIDI processing (for training generative models)

⚠️ in this context, token = elementary component of a MIDI event (position, duration, pitch value, *etc*).

survey:

N. Fradet, J.-P. Briot, F. Chhel, A. E. F. Seghrouchni, and N. Gutowski

Byte pair encoding for symbolic music (lib. MIDItok)

arXiv 2023.

Tokenisation (related or not)

other use of the term *tokenization* in literature

NLP (training of statistical language models)

tokenization = building sequential encodings of character sequences into integer vectors,
using a dictionary of tokens

MIDI processing (for training generative models)

⚠️ in this context, token = elementary component of a MIDI event (position, duration, pitch value, *etc*).

survey:

N. Fradet, J.-P. Briot, F. Chhel, A. E. F. Seghrouchni, and N. Gutowski

Byte pair encoding for symbolic music (lib. MIDItok)

arXiv 2023.

Conclusion

Summary

- Correspondence: score elements ↔ well typed token made of MIDI events
- Parse-based tokenization procedure
- Implementation: command line tool https://gitlab.inria.fr/qparse/qparselib

Focus : dealing with ornaments and rests.

Restriction: for input that is monophonic with chords.

This procedure has been applied to Automatic Drum Transcription:

M. Digard, F. Jacquemard, and L. Rodriguez de la Nava. *Automated transcription of electronic drumkits*. WoRMS 2022.

Further work

- evaluation on a dataset of (monophonic + chords) extracts
  e.g. voices in piano scores
- polyphonic (piano) case:
  voice separation in pre-processing or coupled with tokenisation