Automated Music Transcription based on Formal Language Models

Florent Jacquemard
Lydia Rodriguez-de la Nava
Inria

Philippe Rigaux
le cnam

Masahiko Sakai
Nagoya University

JSPS 採譜 Meeting, Maison Suger, Paris. Sept 16, 2022

Philippe Rigaux
le cnam Paris

Florent Jacquemard
Inria

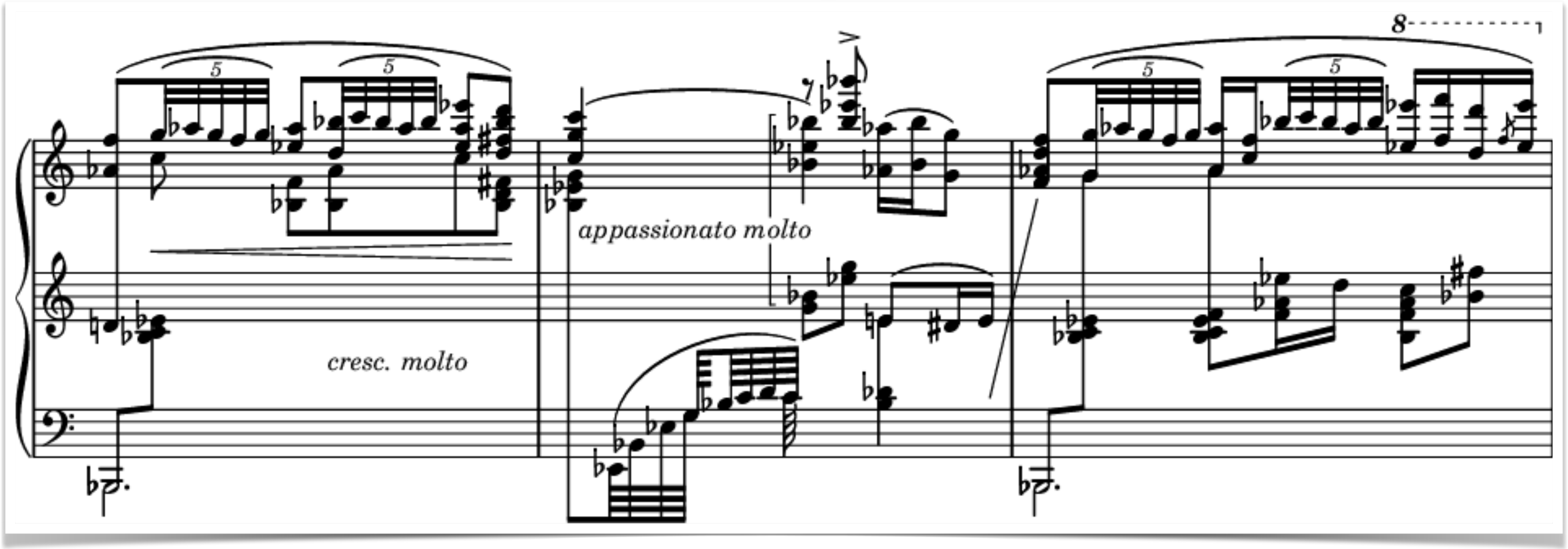
Raphaël Fournier-S'niehotta
le cnam

Lydia Rodriguez-de la Nava
PhD (Codex, Inria)

Tiange Zhu
PhD (Polifonia, H2020)

X
post-doc (Collabscore, ANR)

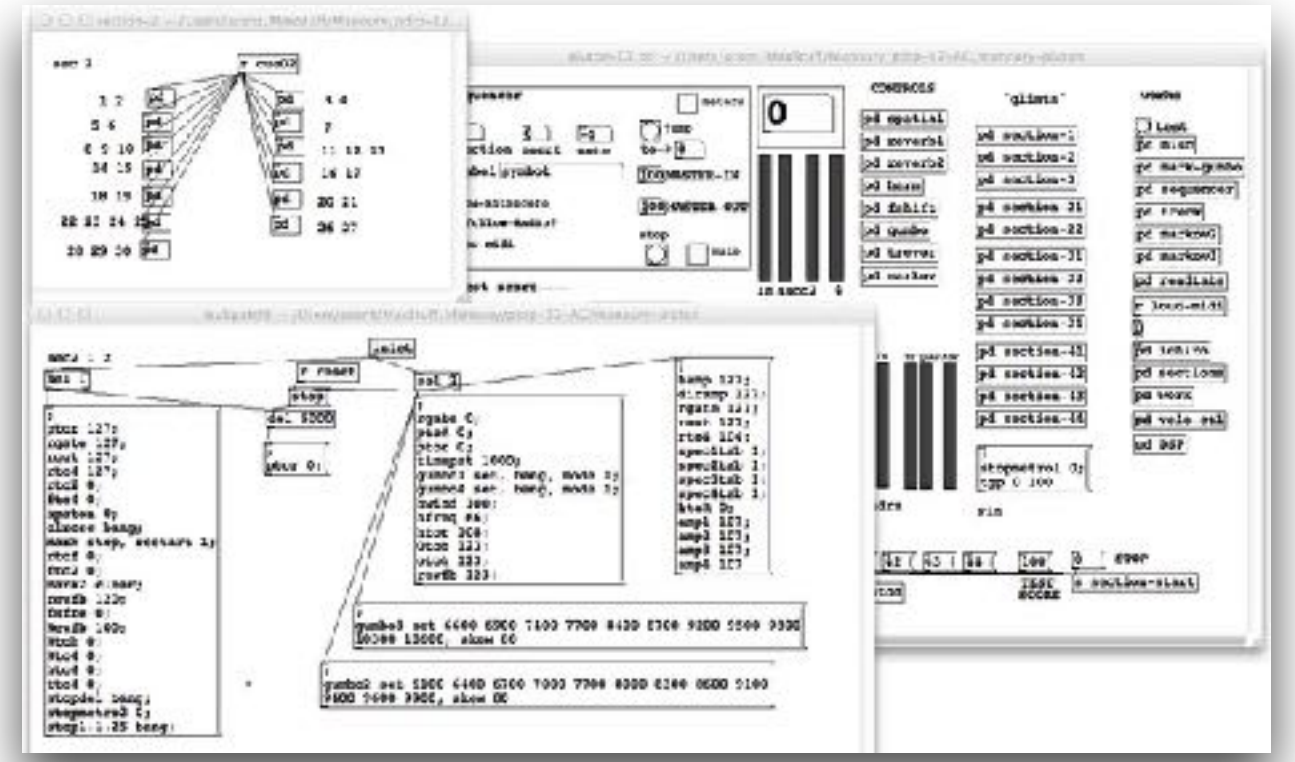
Music Notation Processing



E. Granados, Goyescas
typesetted with Lilypond

Why studying Music Notation Processing?

Western Music Notation = graphical format for music practice,
in use since ~1000 years (Guido d'Arezzo)



vs

Philippe Manoury
Tensio for string quartet and electronics

(digital) music scores, a natural language for

- performers
  performance : real-time reading or memoization
- composers
  authoring, exchange
- teachers & students
  transmission
- editors
  access digital score libraries *e.g.* nkoda.com
- librarians
  cultural heritage preservation: *e.g.* Gallica
- scholars (historians, musicologists…)
  research, analysis

Music Notation Processing by Vertigo & Codex

Structured Music Score models
<u>hierarchical</u> representation of music scores
finite representations of languages (*style*)

Search and Retrieval
indexing
exact and approximate
search and query

Similarity metrics
string and tree
edit-distances

H2020 Polifonia
U. Bologna
Open University
King's College London
Vrije U. Amsterdam

IReMus (Paris)

ANR Collabscore
IRISA (Renne)
French National Library
fond. Royaumont

JSPS 採譜
Nagoya U. (Sakai lab.)
JAIST (Tojo lab.),

AlgoMus (Lille)

Digital Music Score Collections
Cultural heritage preservation and study
neuma.huma-num.fr

Computational Musicology

Optical Music Recognition
Crowdsourced correction

Automated Music Transcription
qparse

Conversion of a recorded music performance into a music score ~ *speech-to-text* in NLP
a holy graal in Computer Music since 1970’s

646 *Nature Vol. 263 October 21 1976*

articles

Perception of melodies

H. C. Longuet-Higgins

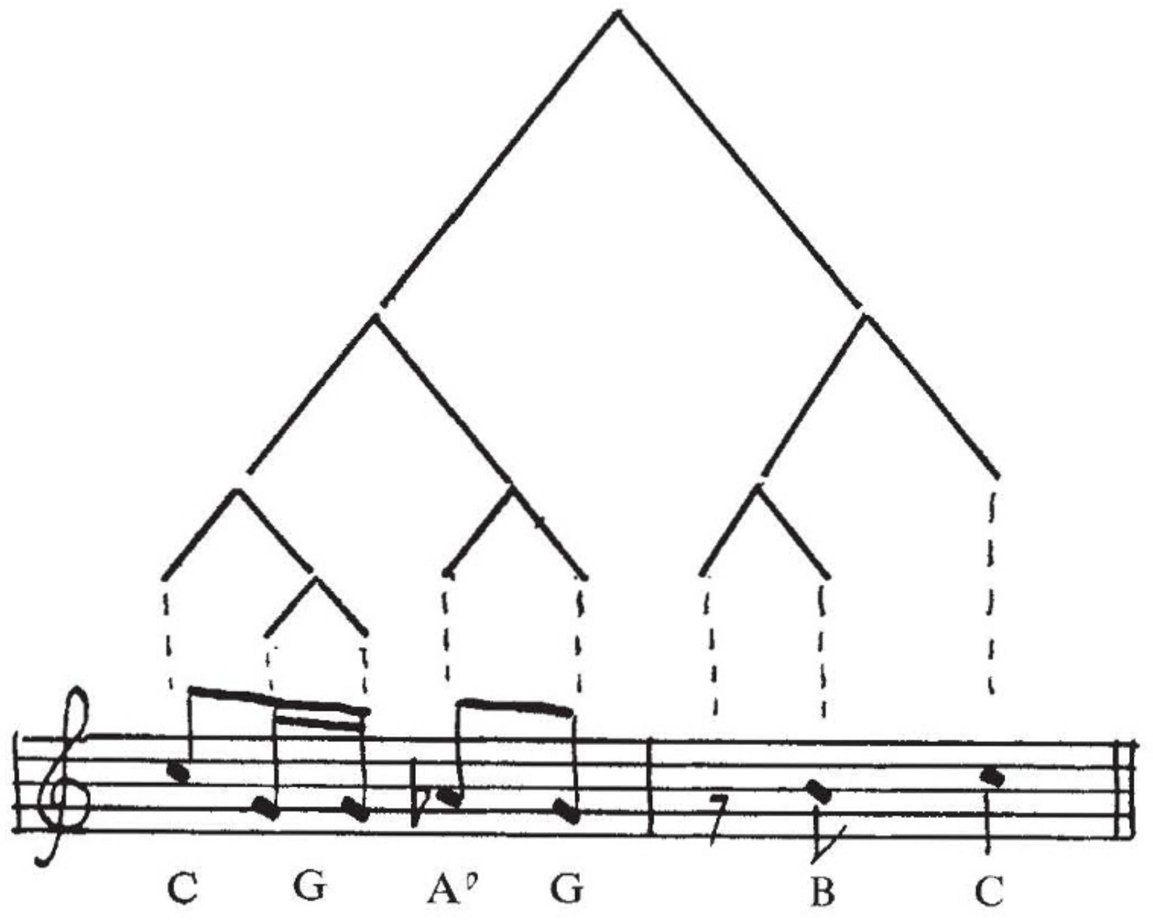
Centre for Research on Perception and Cognition, Laboratory of Experimental Psychology, University of Sussex, Brighton BN1 9QG, UK

A computer program has been written which will transcribe a live performance of a classical melody into the equivalent of standard musical notation. It is intended to embody, in computational form, a psychological theory of how Western musicians perceive the rhythmic and tonal relationships between the notes of such melodies.

A SEARCHING test of practical musicianship is the ‘aural test’ in which the subject is required to write down, in standard, musical notation, a melody which he has never heard before. His transcription is not to be construed as a detailed record of the actual performance, which will inevitably be more or less out of time and out of tune, but as an indication of the rhythmic and tonal relations between the individual notes. How the musical listener perceives these relationships is a matter of some interest to the cognitive psychologist. In this paper I outline a theory of the perception of classical Western melodies, and describe a computer program, based on the theory, which displays, as best it can, the rhythmic and tonal relationships between the notes of a melody as played by a human performer on an organ console.

The basic premise of the theory is that in perceiving a melody the listener builds a conceptual structure representing the rhythmic groupings of the notes and the musical intervals between them. It is this structure which he commits to memory, and which subsequently enables him to recognise the tune, and to reproduce it in sound or in writing if he happens to be a skilled musician. A second premise is that much can be learned about the structural relationships in any ordinary piece of music from a study of its orthographic representation. Take, for example, the musical cliché notated in Fig. 1.

Fig. 1



Conversion of a recorded music performance into a music score

source(s)

Audio recording

MIDI device
(score edition)

Algorithmic composition
DAW

audio Music Information Retrieval

- fundamental freq. estimation
- onset detection
- beat tracking …

intermediate representation
piano roll (MIDI file)

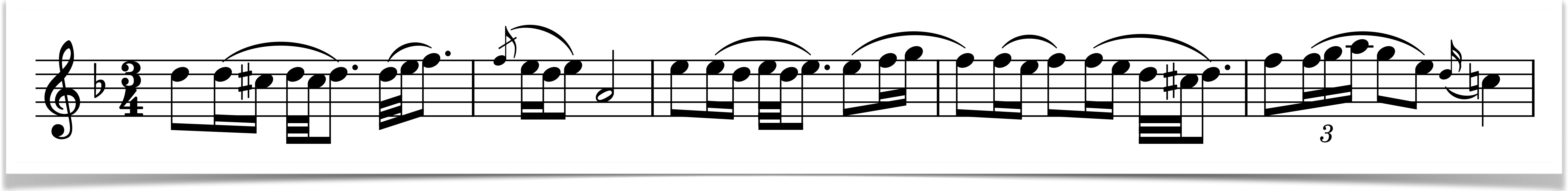
- reals-time durations (*seconds*), unquantized
- quantized pitches

symbolic Music Information Retrieval

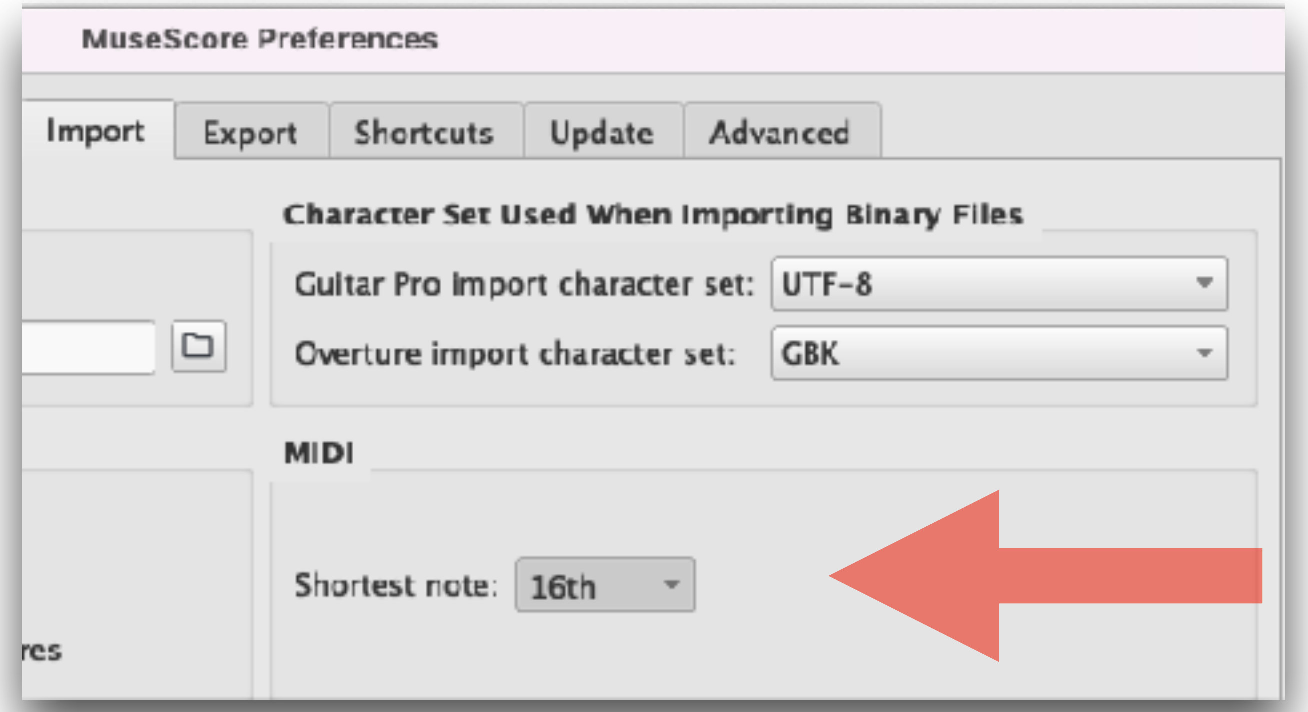
- rhythm quantization
- tempo tracking
- score engraving…

target music score
(e.g. XML file)

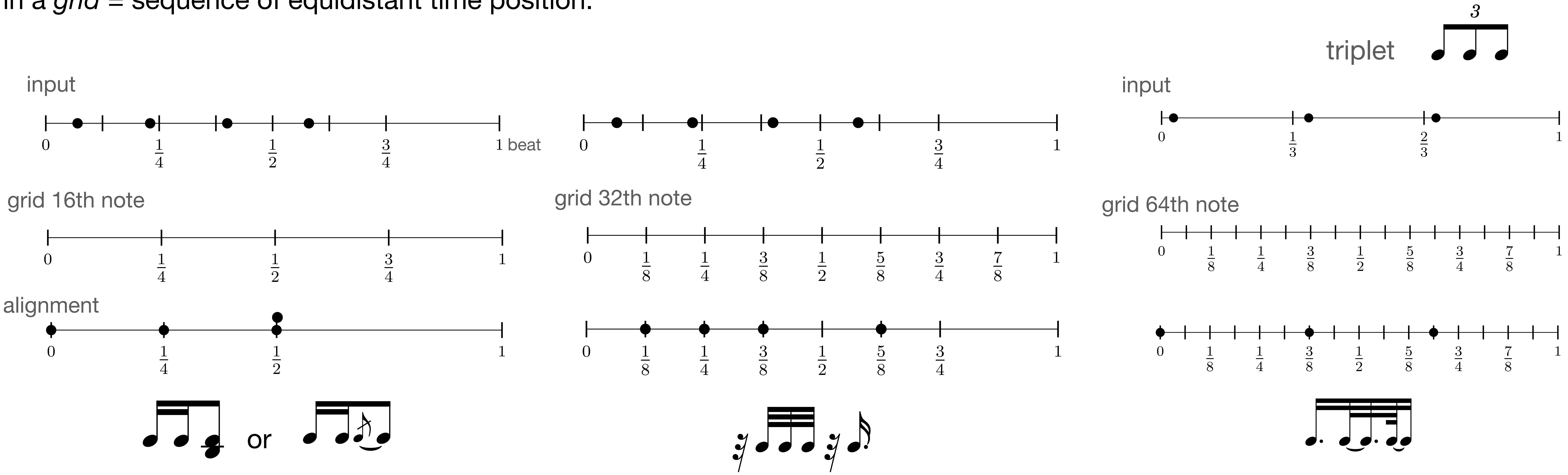
- musical time durations (*beats*) quantized



Rhythm quantization with grids, *e.g.* MIDI files import
- in score editors (Finale, Sibelius, Dorico, Musescore…),
- or in DAWs (Ableton Live, Logic…)



Alignment of every input time point (onset) to the closest position in a *grid* = sequence of equidistant time position.

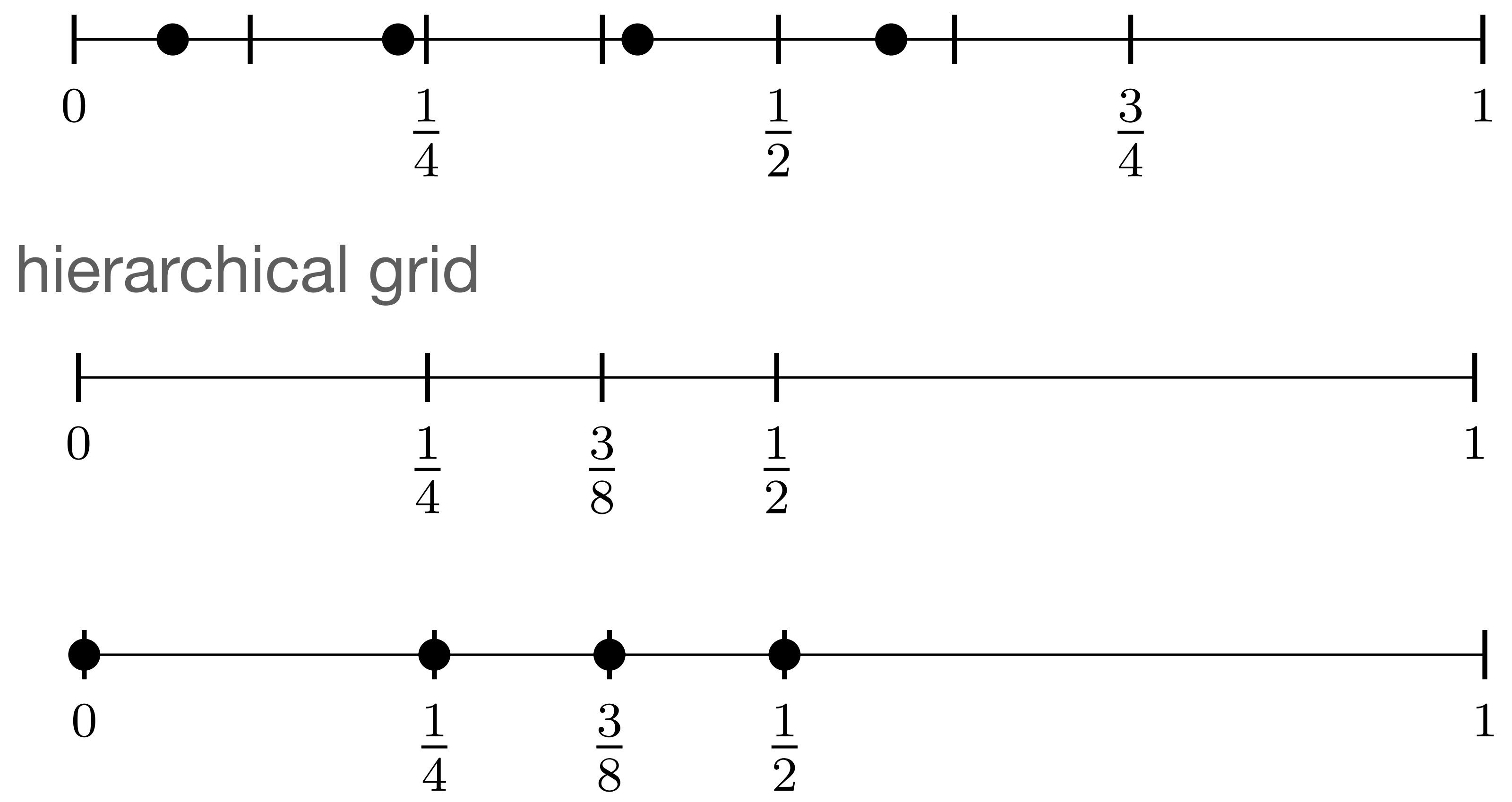


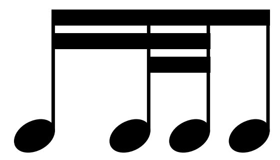
poor fit, good readability

good fit, bad readability

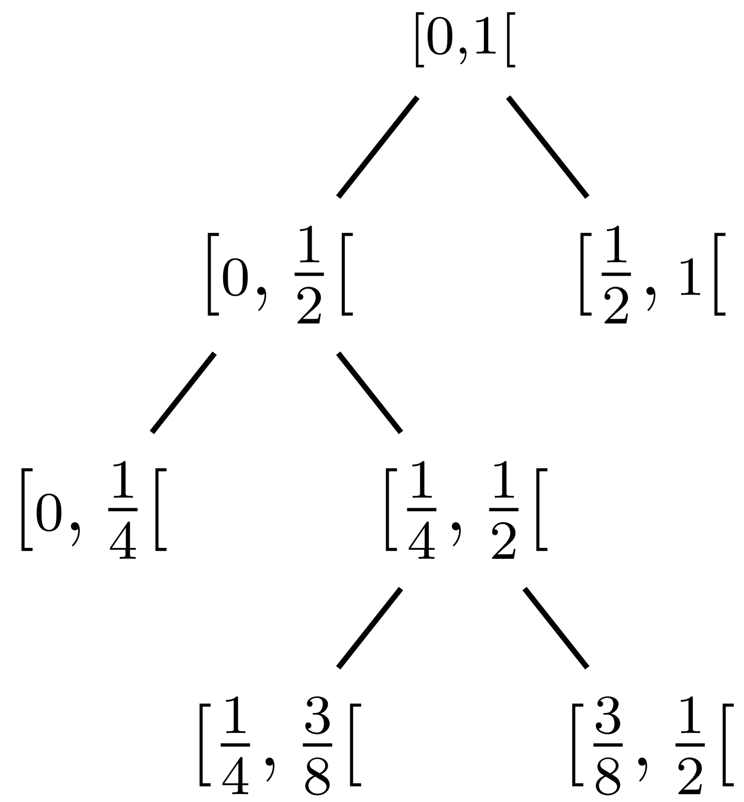
good fit, bad readability

Hierarchical (irregular) Grids





closer to intuition



regular grids

- search of a best quantization is possible by a brute-force enumeration: 8th note grid, 16th, 32th, 64th…
- result not always optimal
- problems with tuplets (so called "*irrationals*" 3, 5, 7…)

hierarchical grids

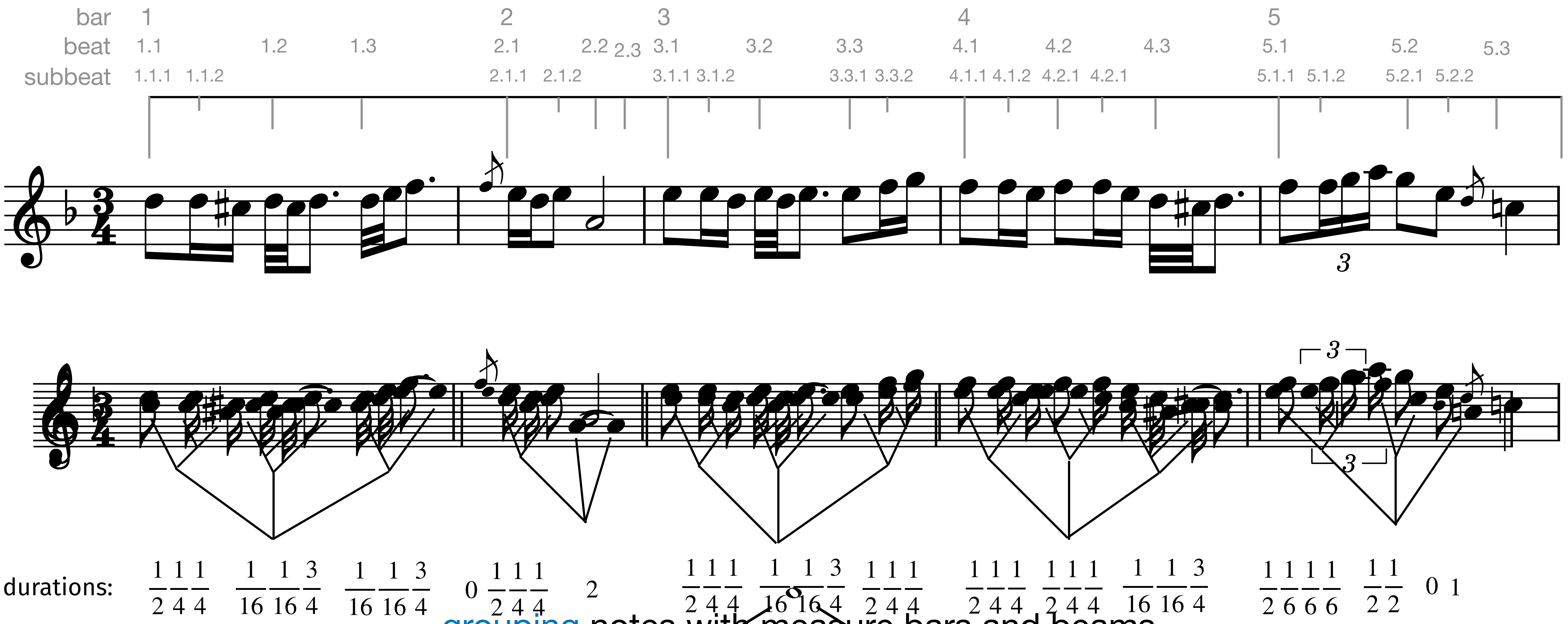
- more "natural" results
- brute force enumeration impossible
- how to specify the grids to try ?

Polonaise in D minor from Notebook for Anna Magdalena Bach BWV Anh II 128

metric structure

beamed

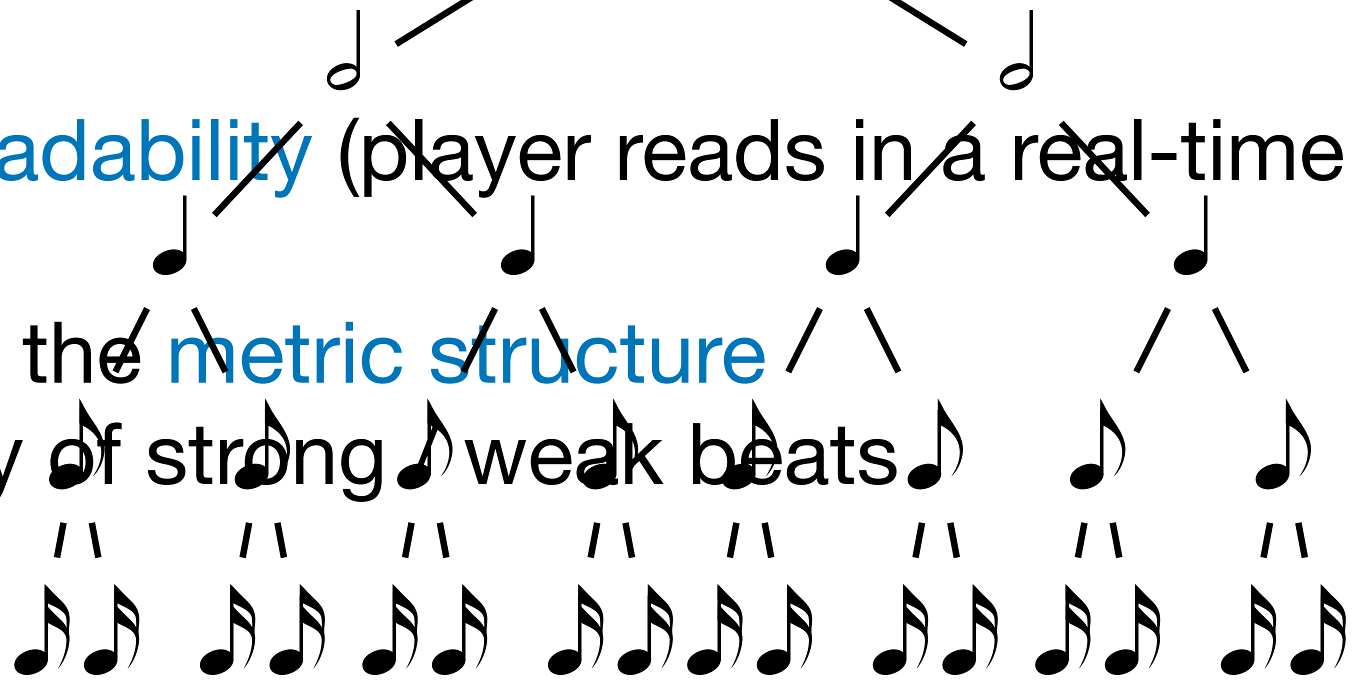
unbeamed



hierarchical note durations

grouping notes with measure bars and beams

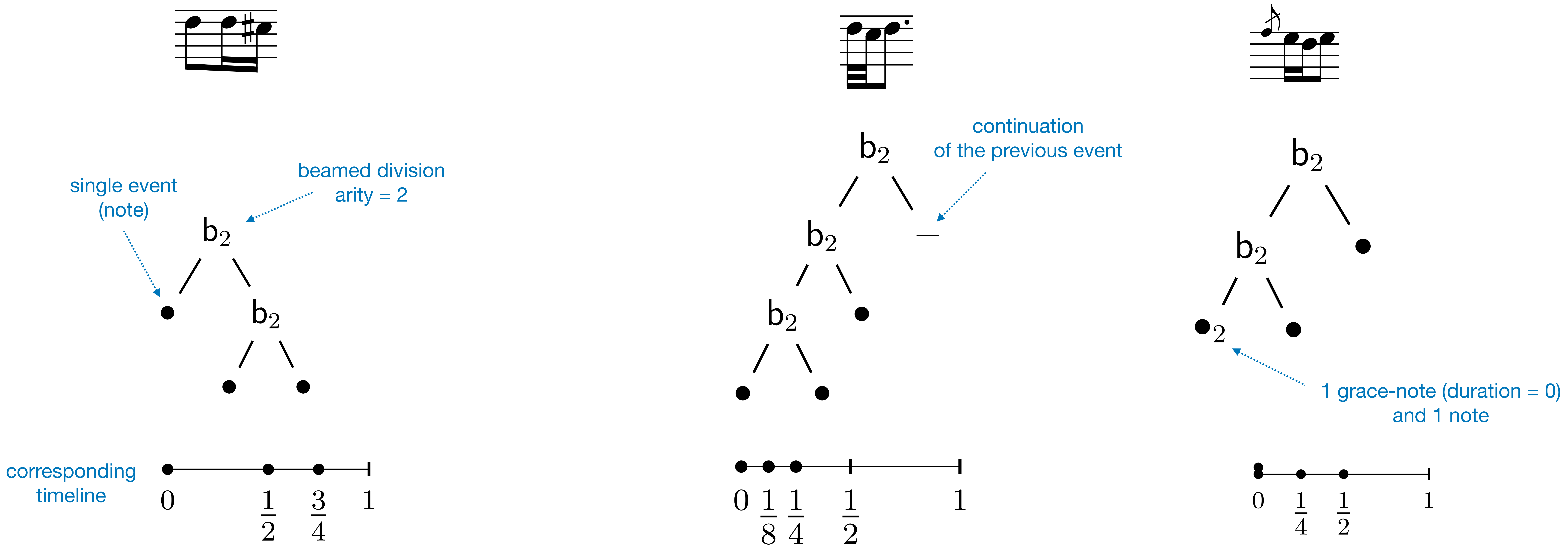
- eases readability (player reads in a real-time context)
- highlight the metric structure hierarchy of strong / weak beats



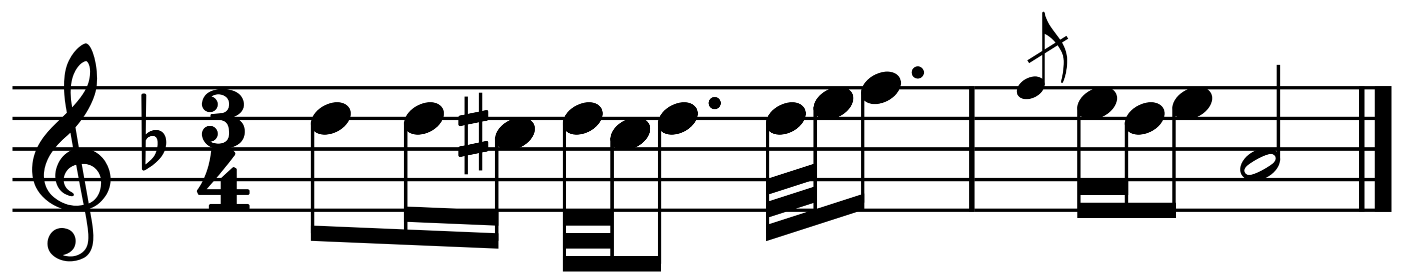
Tree-structured Representation of Music Notation

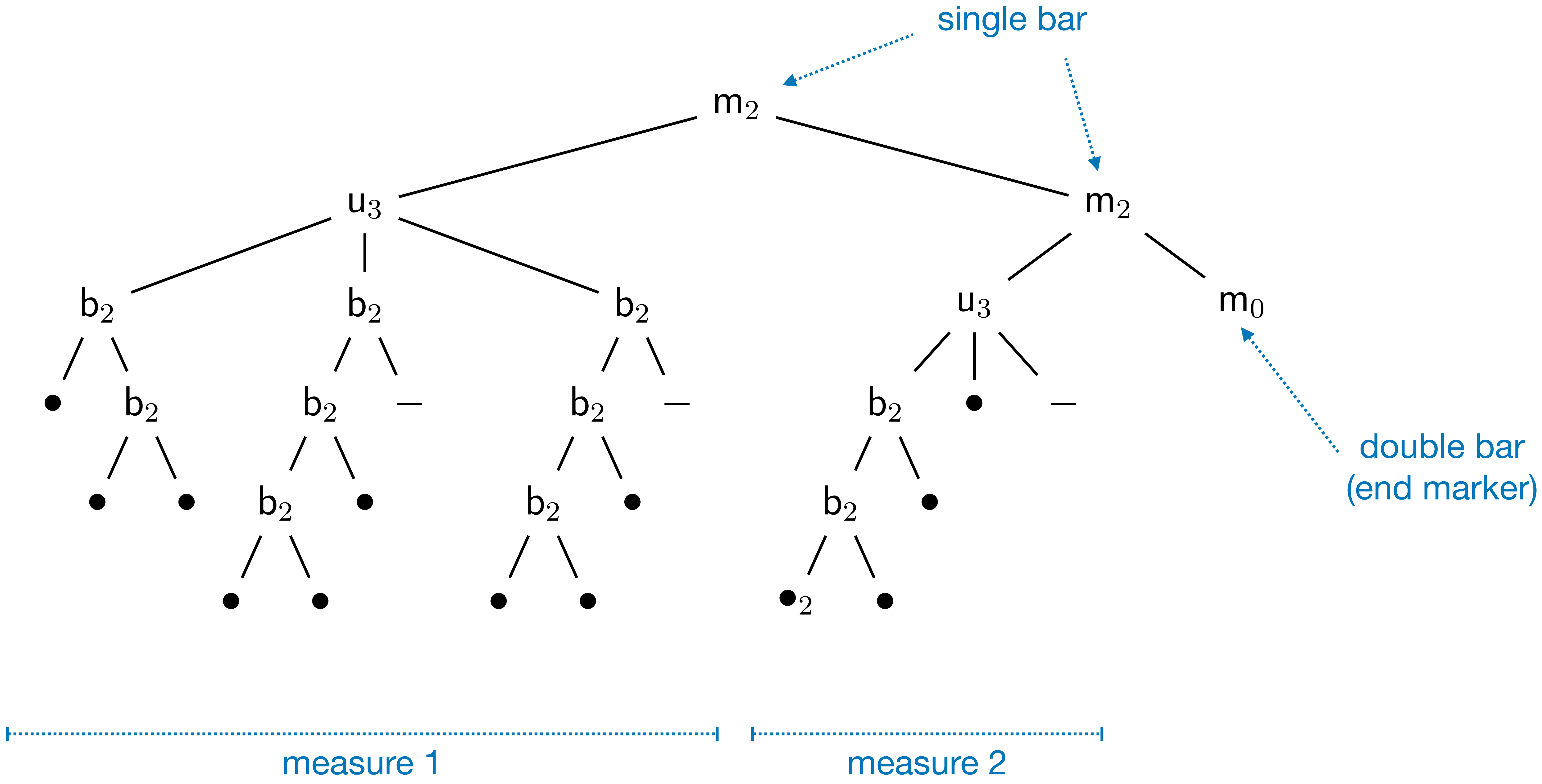
Tree representation of the proportional rhythmic notation
with hierarchical encoding of durations: "*the* (duration) *data is in the structure*"

- the tree leaves contain the events
- the branching define durations, by partitioning of time intervals



Tree-structured Representation of Music Notation





Regular Tree Language (of Music Notation)

defined by a Regular Tree Grammar:

- non-terminal symbols: $q, q_0, q_1, \ldots$
- terminal symbols (constants): • (1 note), $\bullet_2$ (1 grace-note + 1 note), — (continuation)
- production rules:

$$q \to \mathrm{m}_2(q_0, q) \mid \mathrm{m}_0$$

$$q_0 \to \mathrm{u}_3(q_1, q_1, q_1) \mid \bullet$$ measure

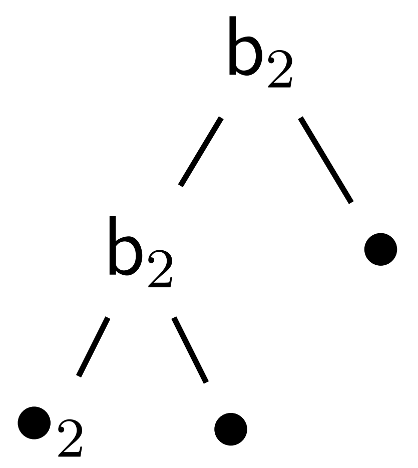
$$q_1 \to \mathrm{b}_2(q_2', q_2) \mid \bullet \mid \bullet_2 \mid -$$ beat = ♩

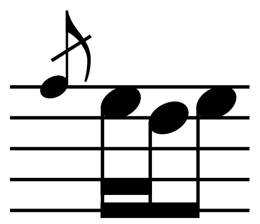
$$q_2' \to \mathrm{b}_2(q_3', q_3) \mid \bullet \mid \bullet_2 \mid - \qquad q_2 \to \mathrm{b}_2(q_3, q_3) \mid \bullet \mid -$$ sub-beat = 8th-note = ♪

$$q_3' \to \bullet \mid \bullet_2 \mid - \qquad q_3 \to \bullet \mid -$$ sub-sub-beat = 16nth note = 𝅘𝅥𝅯

derivations (lefmost)

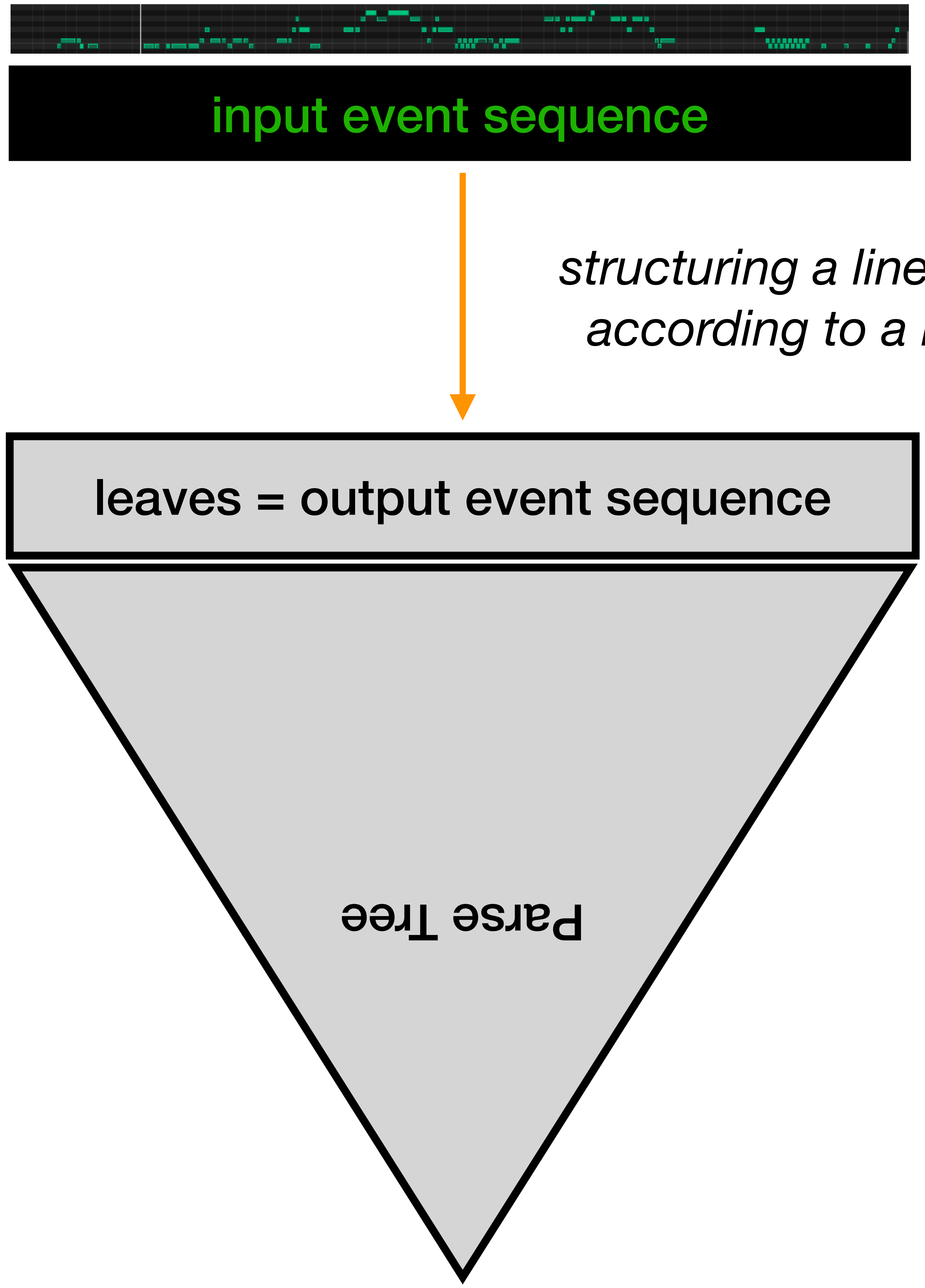
$$q_1 \to \mathrm{b}_2(q_2', q_2) \to \mathrm{b}_2(\mathrm{b}_2(q_3', q_3), q_2) \to \mathrm{b}_2(\mathrm{b}_2(\bullet_2, q_3), q_2) \to \mathrm{b}_2(\mathrm{b}_2(\bullet_2, \bullet), q_2) \to \mathrm{b}_2(\mathrm{b}_2(\bullet_2, \bullet), \bullet)$$





$$q \to \mathrm{m}_2(q_0, q) \to \mathrm{m}_2(\mathrm{u}_3(q_1, q_1, q_1), q) \to \mathrm{m}_2(\mathrm{u}_3(\mathrm{b}_2(q_2', q_2), q_1, q_1), q) \to \mathrm{m}_2(\mathrm{u}_3(\mathrm{b}_2(\bullet, q_2), q_1, q_1), q) \to \ldots$$

Transcription as Parsing



piano roll
= sequence of timestamped input events

structuring a linear representation according to a language model = **parsing**

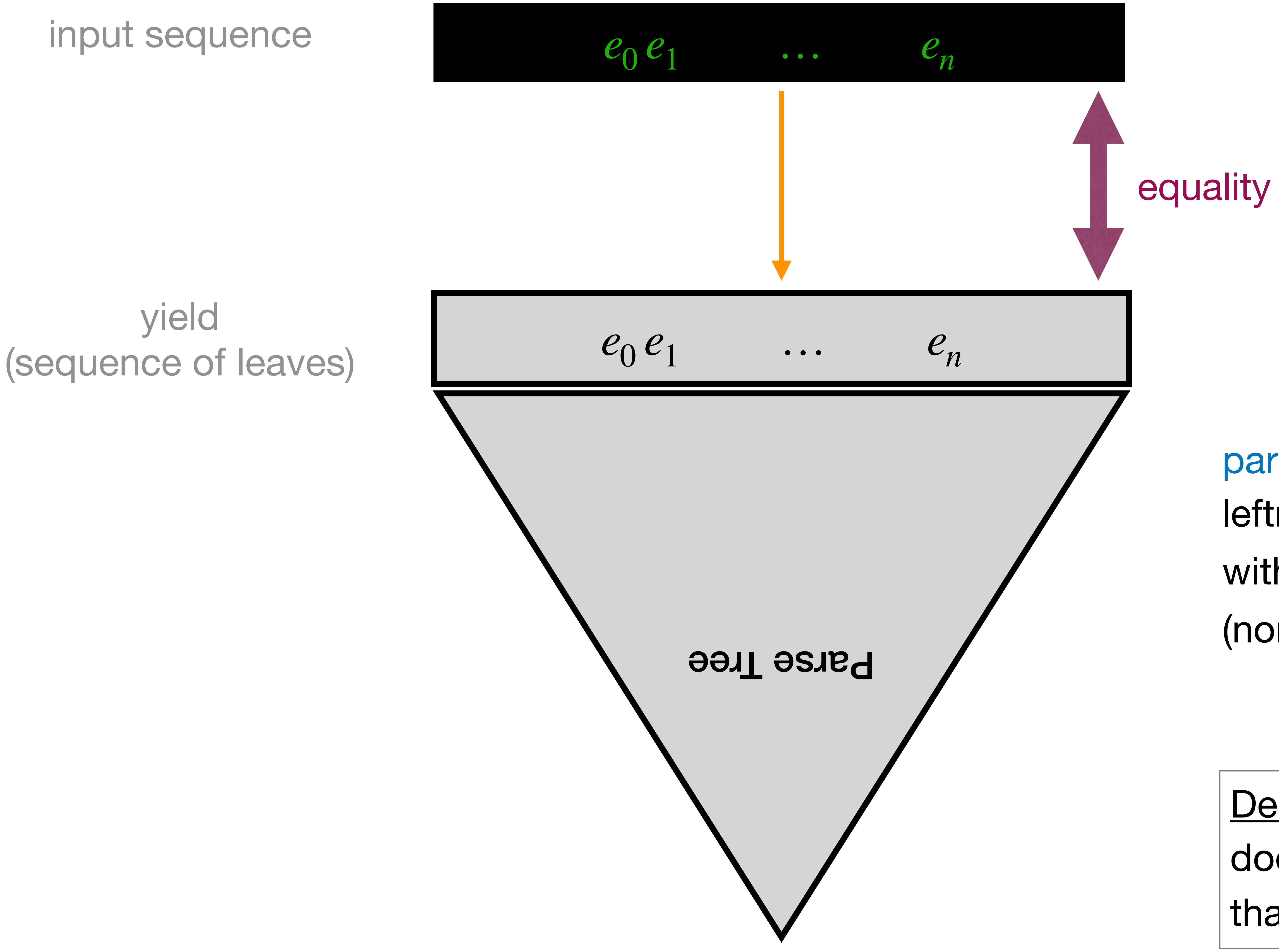
tree-structured representation of an output **music score**

conforming to a prior language (expected notation)

nested extensions of parsing are needed for the case music transcription:

- weighted extension
- symbolic weighted extension (*quantitative parsing*)

terminal symbols: $e_0, \ldots$ in a finite alphabet

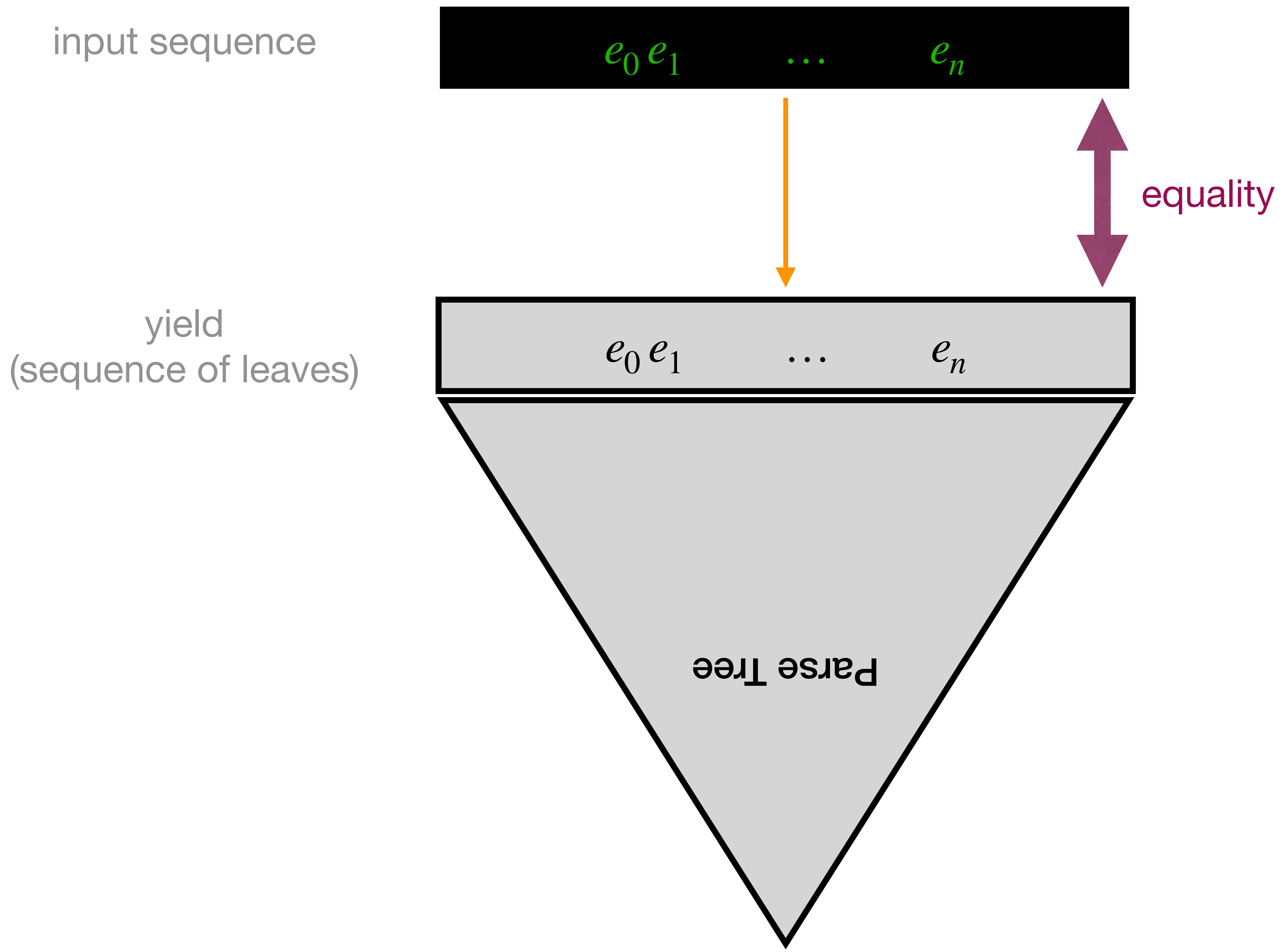


parse-tree = representation of a leftmost derivation of $e_0\, e_1 \ldots e_n$ by a prior CF-grammar $\mathscr{G}$ with production rules: $q_0 \to q_1\, q_2$ or $q_0 \to e$ (non-terminal symbols: $q_0, q_1, \ldots$)

<u>Decision problem</u>: (membership)
does there exists a parse tree (leftmost derivation) of $\mathscr{G}$ that yields $e_0\, e_1 \ldots e_n$?

Weighted Parsing (extension 1)

Returning a parse tree of $\mathcal{G}$ that yields $e_0\, e_1 \ldots e_n$



With an ambiguous prior CF-grammar $\mathcal{G}$ there might exists several parse trees (exponentially many).

in order to choose one (or some) parse trees, rank them according to their weight values, computed by Weighted Tree Grammar

Weighted Tree Language (*tree series*)

Weighted Regular Tree Grammar $\mathscr{G}$:

- non-terminal symbols: $q, q_0, q_1, \ldots$
- terminal symbols (constants): • (1 note), $\bullet_2$ (1 grace-note + 1 note), — (continuation)
- every production rule is assigned a weight value (*e.g.* cost to read):

$q \xrightarrow{0} \mathsf{m}_2(q_0, q)$ $\quad q \xrightarrow{0} \mathsf{m}_0$

$q_0 \xrightarrow{0.1} \mathsf{u}_3(q_1, q_1, q_1)$ $\quad q_0 \xrightarrow{1} \bullet$ — measure

$q_1 \xrightarrow{0.1} \mathsf{b}_2(q_2', q_2)$ $\quad q_1 \xrightarrow{1} \bullet$ $\quad q_1 \xrightarrow{1.9} \bullet_2$ $\quad q_1 \xrightarrow{1} -$ — beat = 𝅗𝅥

$q_2' \xrightarrow{0.1} \mathsf{b}_2(q_3', q_3)$ $\quad q_2' \xrightarrow{1} \bullet$ $\quad q_2' \xrightarrow{2.25} \bullet_2$ $\quad q_2' \xrightarrow{1} -$ — sub-beat = 8th-note = ♪

$q_2 \xrightarrow{0.1} \mathsf{b}_2(q_3, q_3)$ $\quad q_2 \xrightarrow{1} \bullet$ $\quad q_2 \xrightarrow{1} -$

$q_3' \xrightarrow{1} \bullet$ $\quad q_3' \xrightarrow{3.25} \bullet_2$ $\quad q_3' \xrightarrow{1} -$ $\quad q_3 \xrightarrow{1} \bullet$ $\quad q_3 \xrightarrow{1} -$ — sub-sub-beat = 16th note = 𝅘𝅥𝅯

derivation (lefmost): $d : q_1 \xrightarrow{0.1} \mathsf{b}_2(q_2', q_2) \xrightarrow{0.1} \mathsf{b}_2(\mathsf{b}_2(q_3', q_3), q_2) \xrightarrow{3.25} \mathsf{b}_2(\mathsf{b}_2(\bullet_2, q_3), q_2) \xrightarrow{1} \mathsf{b}_2(\mathsf{b}_2(\bullet_2, \bullet), q_2) \xrightarrow{1} \mathsf{b}_2(\mathsf{b}_2(\bullet_2, \bullet), \bullet)$

cost of derivation: $\mathsf{weight}(d) = 0.1 + 0.1 + 3.25 + 1 + 1$

learning weight values from corpus statistics
Francesco Foscarin

In general, the weight values are taken in a commutative Semiring $\langle \mathbb{S}, \oplus, \mathbb{0}, \otimes, \mathbb{1} \rangle$

- $\oplus$ and $\otimes$ are associative and commutative, with neutral elements $\mathbb{0}$ and $\mathbb{1}$
- $\otimes$ distributes over $\oplus$: $x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$
- $\mathbb{0}$ is absorbing for $\otimes$: $\mathbb{0} \otimes x = \mathbb{0}$

	domain	$\oplus$	$\otimes$	$\mathbb{0}$	$\mathbb{1}$
Boolean	$\{\bot, \top\}$	$\vee$	$\wedge$	$\bot$	$\top$
Viterbi	$[0,1] \subset \mathbb{R}$	max	$\times$	0	1
Tropical min-plus	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

Moreover, $\oplus$ is assumed to extend to infinite sums: there is an operation $\bigoplus_{i \in I} x_i$ for all $I \subseteq \mathbb{N}$ such that:

infinite sums extend finite sums: $\forall j, k \in \mathbb{N}, j \neq k, \bigoplus_{i \in \varnothing} x_i = \mathbb{0}, \bigoplus_{i \in \{j\}} x_i = x_j, \bigoplus_{i \in \{j,k\}} x_i = x_j \oplus x_k$

associativity and *commutativity*:
for all partition $(I_j)_{j \in J}$ of I, $\bigoplus_{j \in J} \bigoplus_{i \in I_j} x_i = \bigoplus_{i \in I} x_i$

distributivity of products over infinite sums: for all $I \subseteq \mathbb{N}$, $\forall x, y \in \mathbb{S}$

$\bigoplus_{i \in I} (x \otimes y_i) = x \otimes \bigoplus_{i \in I} y_i$ and $\bigoplus_{i \in I} (x_i \otimes y) = (\bigoplus_{i \in I} x_i) \otimes y$

Weight of derivations and trees

	domain	$\oplus$	$\otimes$	$\mathbb{0}$	$\mathbb{1}$
Boolean	$\{\bot, \top\}$	$\vee$	$\wedge$	$\bot$	$\top$
Viterbi	$[0,1] \subset \mathbb{R}$	max	$\times$	0	1
Tropical min-plus	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

$\otimes$ is for composition of rule's weights in derivations and $\oplus$ is for optimal choice:

For a Weighted Regular Tree Grammar $\mathscr{G}$

$$\text{weight}_{\mathscr{G}}(d : q \xrightarrow{w_1} \ldots \xrightarrow{w_n} t) = \bigotimes_{i=1}^{n} w_i \quad \text{and} \quad \text{weight}_{\mathscr{G}}(q, t) = \bigoplus_{d : q \xrightarrow{+} t} \text{weight}_{\mathscr{G}}(d)$$

or recursively:

$$\text{weight}_{\mathscr{G}}(q, a(t_1, \ldots, t_n)) = \bigoplus_{q \xrightarrow{w} a(q_1, \ldots, q_n) \in \mathscr{G}} \left(w \otimes \bigotimes_{i=1}^{n} \text{weight}_{\mathscr{G}}(q_i, t_i) \right)$$

	domain	$\oplus$	$\otimes$	$\mathbb{0}$	$\mathbb{1}$
Boolean	$\{ \bot, \top \}$	$\vee$	$\wedge$	$\bot$	$\top$
Viterbi	$[0,1] \subset \mathbb{R}$	$\max$	$\times$	0	1
Tropical min-plus	$\mathbb{R}_+ \cup \{+\infty\}$	$\min$	$+$	$+\infty$	0

$\mathbb{S}$ is assumed :

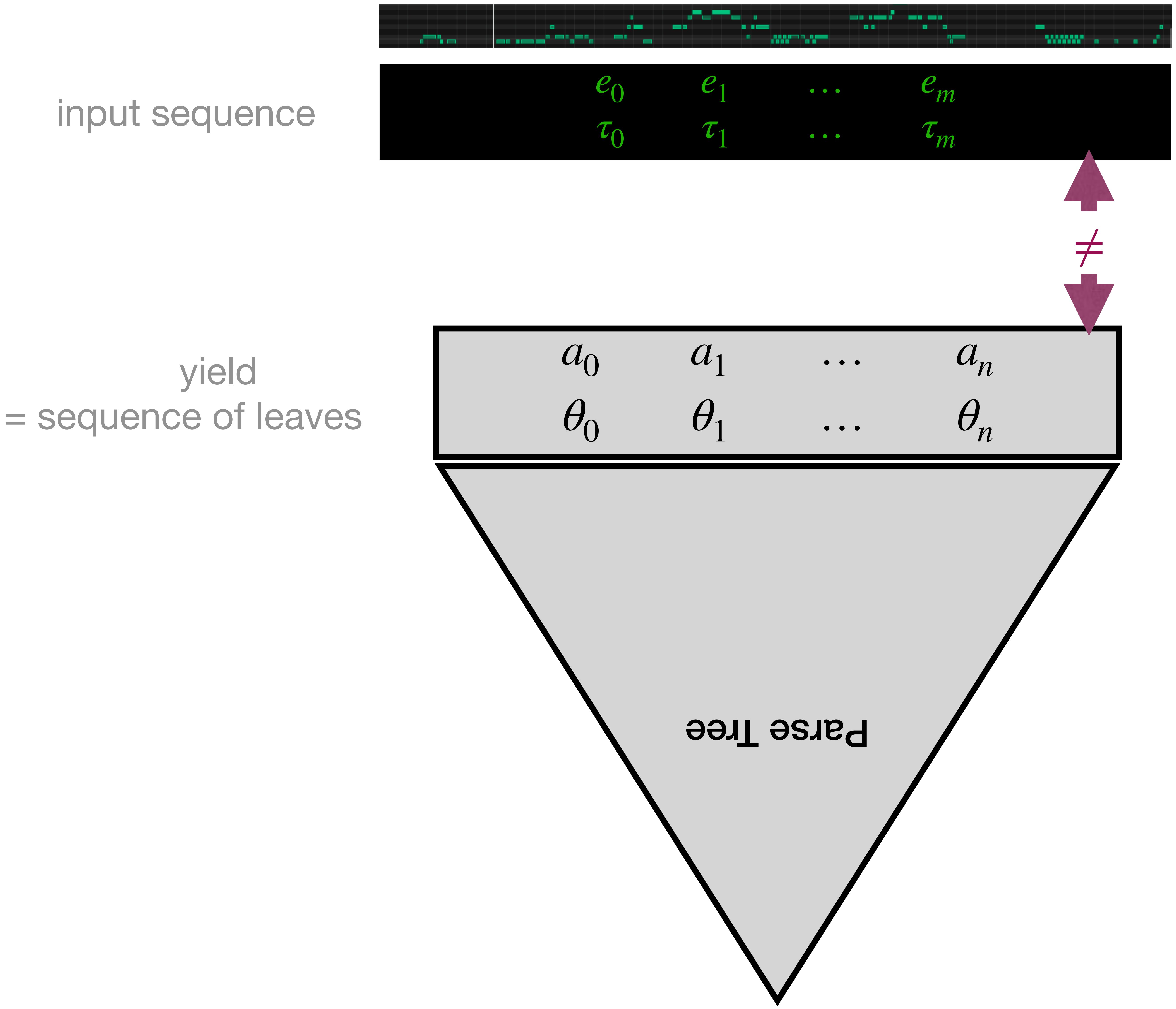
- idempotent $x \oplus x = x$
  that induces a partial ordering: $x \leq_{\oplus} y$ iff $x \oplus y = x$
- total : $\forall x, y \in \mathbb{S}$, either $x \oplus y = x$ or $x \oplus y = y$ *i.e.* $\leq_{\oplus}$ is total
- bounded : $\mathbb{1} \oplus x = \mathbb{1}$, or equivalently: $\forall x, y \in \mathbb{S}, x \leq_{\oplus} x \otimes y$
  i.e. combining elements with $\otimes$ always increases their weight,
  see the *non-negative weights* condition for Dijkstra's shortest path algorithm

k-best parsing : enumeration of the k best weighted trees *wrt* $\leq_{\oplus}$ for $\mathscr{G}$ and a non-terminal q, in PTIME, user the above assumptions.

Similar to best path search in hyper-graphs (Dynamic Programming)

- Viterbi algorithm in acyclic case
- Knuth generalization of Dijkstra's algorithm in the general case

there is no 1-1 correspondance between input sequence and output leave sequence



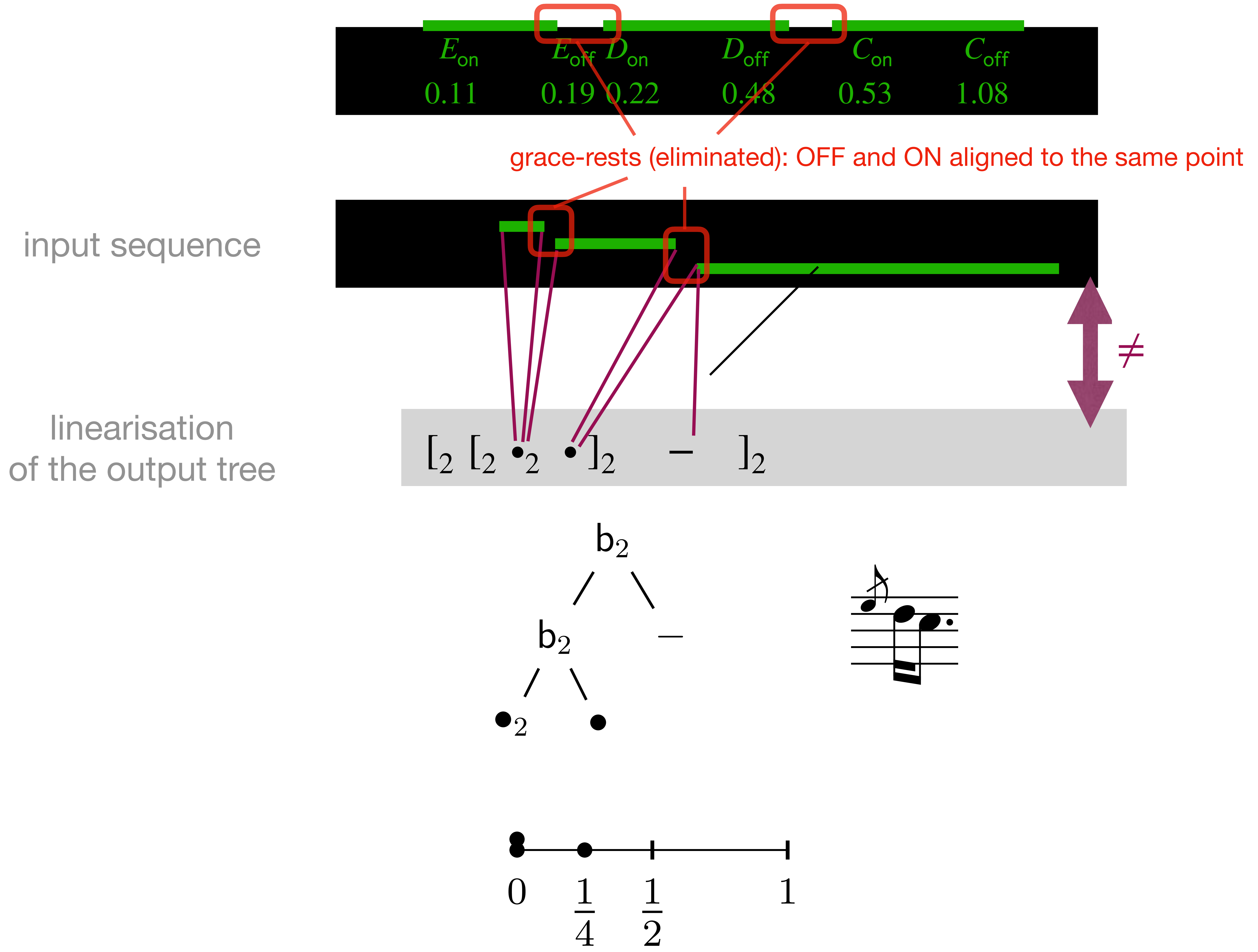
we extend weighted parsing by ranking solutions with:

a measure of input / output fitness
= cost of IO alignement

$\otimes$

measure of cost-to-read
weight value
computed by the Weighted Tree Grammar

measure of input/output fitness



cost of IO alignement
computed by a
Weighted word-to-word Transducer
(stateful definition of an edit-distance)

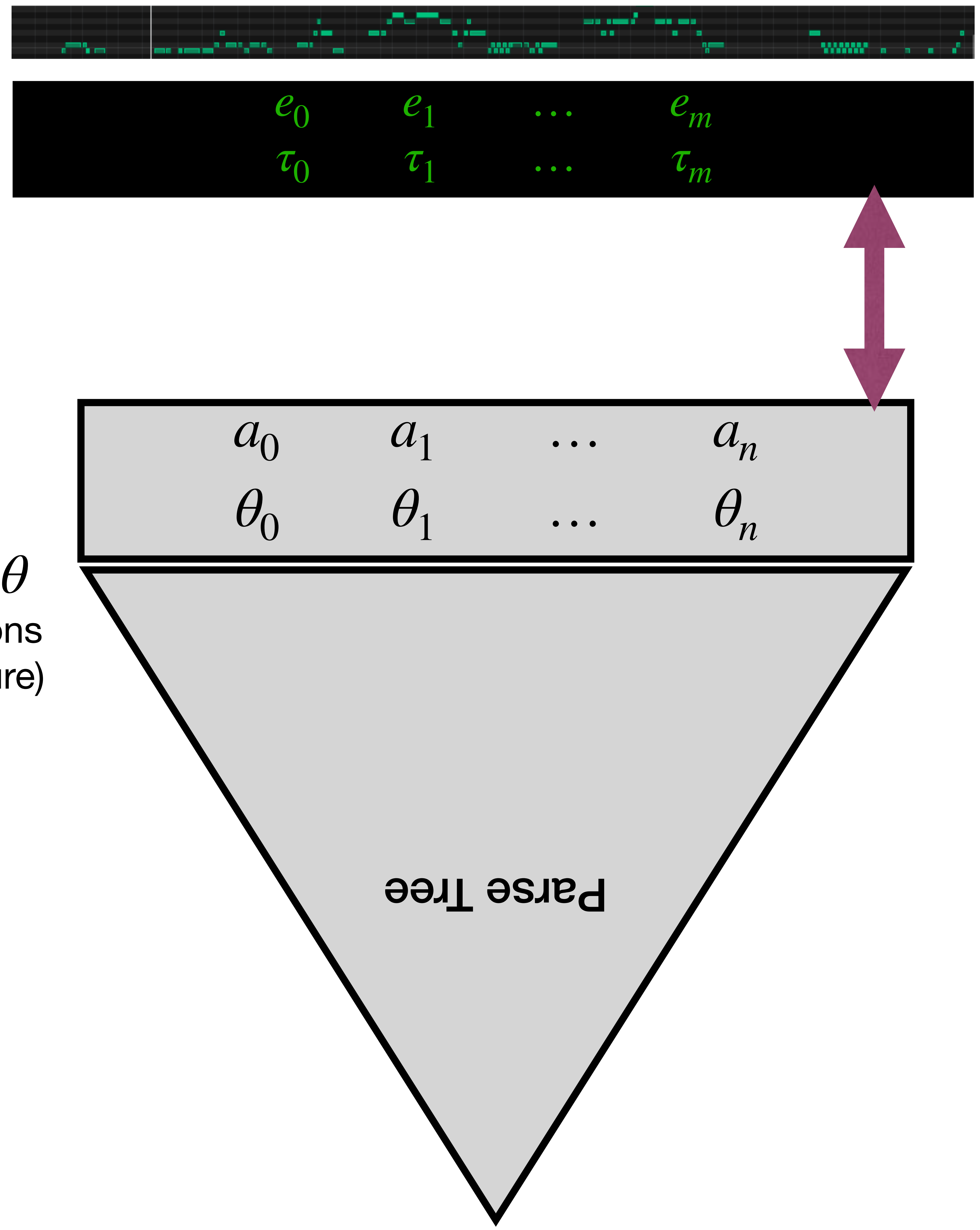
input symbol, output symbol, desynchronise

$$q_0 \xrightarrow{\langle E_{on}, \bullet_2\rangle} q_1 \xrightarrow{\langle E_{off}, \varepsilon\rangle} q_2 \xrightarrow{\langle D_{on}, \varepsilon\rangle} q_3$$

$$\xrightarrow{\langle D_{off}, \bullet\rangle} q_4 \xrightarrow{\langle C_{on}, \varepsilon\rangle} q_5 \xrightarrow{\langle \varepsilon, -\rangle} q_5$$

in the context of music transcription, the symbols are timestamped → infinite alphabet Σ_{inf}
the weighted formalisms below must be able to read such symbols → symbolic extension

input sequence

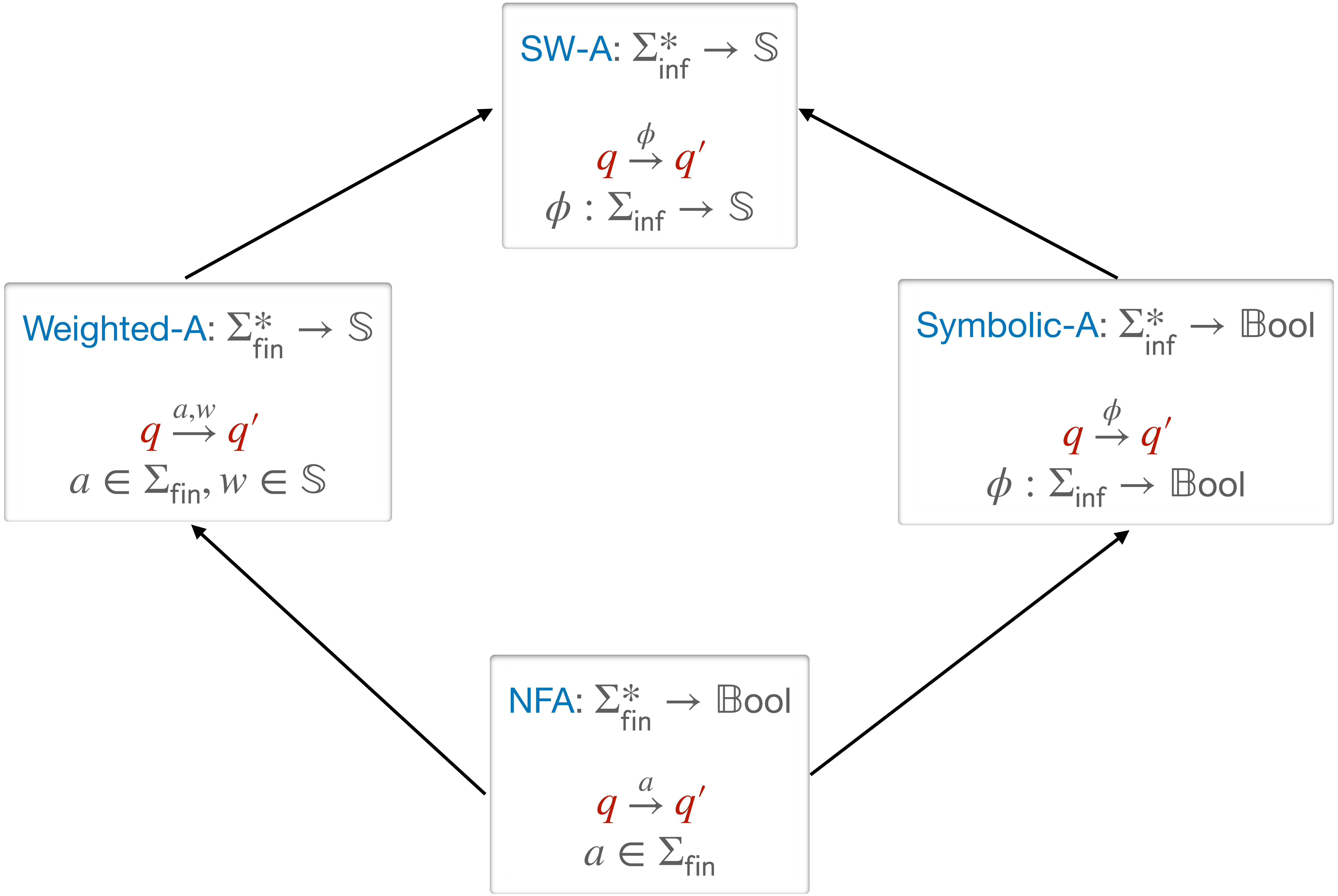


measure of input / output fitness
= cost of IO alignement
computed by a
Weighted word-to-word Transducer

$\otimes$

yield
= sequence of leaves
decorated with dates θ
(computed with the durations encoded in the tree structure)

measure of cost-to-read
computed by the Weighted Tree Grammar



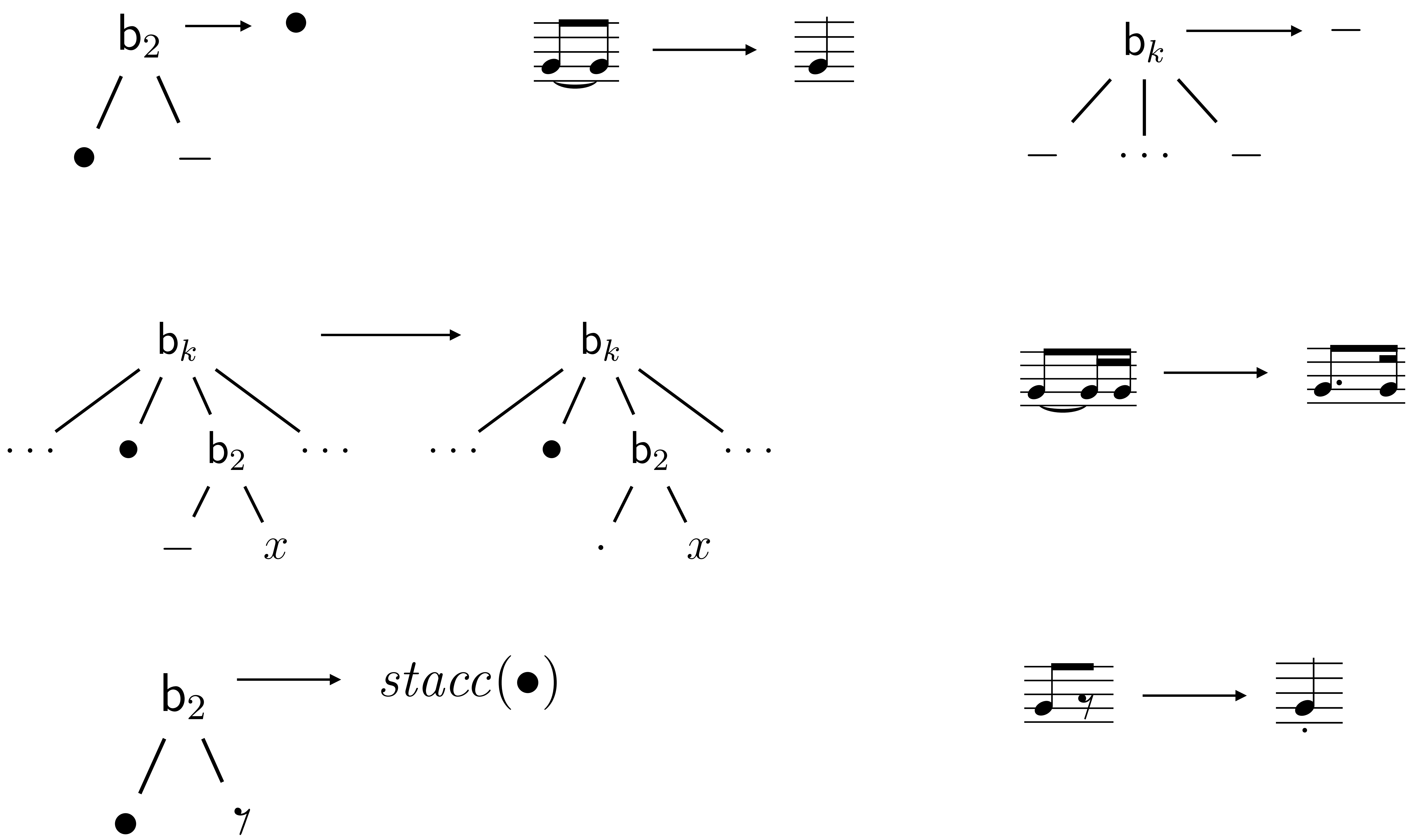
Droste, M., Kuich, W., Vogler
Handbook of WA, 2009

Veanes et al.
CAV 2017, CACM 2021

Automated Music Transcription: qparse framework

Term Rewriting Rules

for the transformation of the intermediate score representation



questions: rewrite strategies (*e.g.* IO or OI), conflicts…

Implementation of

- the above transcription by parsing framework
- the intermediate score model
- other subtasks: pitch-spelling, key estimation, beat tracking…

https://gitlab.inria.fr/qparse/qparselib
https://qparse.gitlabpages.inria.fr

qparse: 75 Kloc C++

- command lines tools:
  `monoparse`, `drumparse`, grammar-learning, engraving (from quantified input)
- Python binding - Lydia Rodrigez-de la Nava
  scripts for automatic evaluation
- online port, real-time - Leyla Villaroel

Related work

- Piano transcription system (Kyoto U.)

Non-local musical statistics as guides for audio-to-score piano transcription
Kentaro Shibata, Eita Nakamura, Kazuyoshi Yoshii

  - deep-neural-network-based multipitch detection
    audio to unquantized MIDI
  - statistical-model-based (HMM) rhythm quantization
    unquantized MIDI to quantized MIDI
  - delegate to Muse Score + Voice separation algorithm for
    quantized MIDI to score
  - study of use of non-local statistics (pitch and rhythm)
    for the inference of global characteristics (metre, bar line positions…)

- Score Transformer (Yamaha) - piano transcription

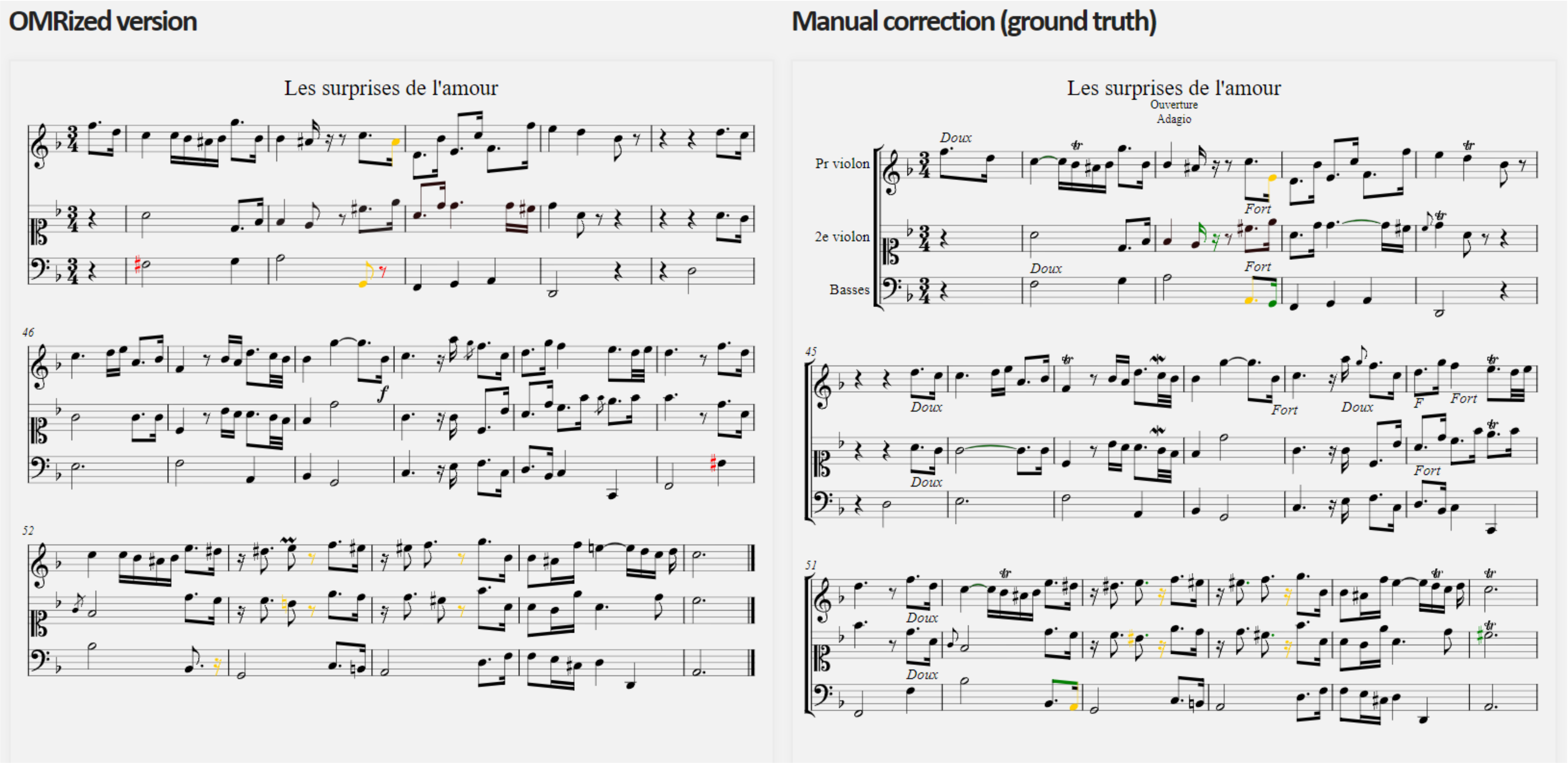
Score Transformer: Generating Musical Score from Note-level Representation
Masahiro Suzuki

Transformer model trained with popular songs (piano arrangements), KernScores (piano Sonata)
MIDI to score (tokenization)

Score diff

by Francesco Foscarin

- identify the diff. between 2 XML music scores
- string/tree edit distance applied to a intermediate score representation



Evaluation data for Monophonic transcription

Generation of artificial performances
Madoka Goto, Masahiko Sakai (Nagoya U.), Satoshi Tojo (JAIST)

- construction of a GTTM tree
- segmentation accordingly
- performance generation by Director Musices (Anders Friberg)

Lamarque-Goudard dataset (w. Francesco Foscarin, Teysir Baoueb)

- 283 monophonic extracts
  of classical repertoire
  inspired by a rhythm learning method
- ~ 20 measures per extract
- progressive difficulty
  cover a very large spectrum of rhythmic features
- score files (XML) and MIDI performances
  for evaluation and calibration of transcription tools

monophonic : one note at a time
Good results for complex cases (ornaments, mixed tuplets, mixed note durations, silences…)
~ 100ms for the transcription of 1 score

Polonaise in D minor from Notebook for Anna Magdalena Bach BWV Anh II 128

original score

transcription of MIDI recording by qparse

Monophonic transcription

Polonaise in D minor from Notebook for Anna Magdalena Bach   BWV Anh II 128

original score

transcription of MIDI recording by Finale

Monophonic transcription

Beethoven, Trio for violin, cello and piano, op.70 n.2 (2d mov)

original score

transcription of MIDI recording by qparse

Beethoven, Trio for violin, cello and piano, op.70 n.2 (2d mov)

original score

transcription
of MIDI recording
by Finale

options:
- mixed rhythms,
- tuplets
- smallest note = 32nd
The time signature
and the tempo are given.

FiloBass by John-Xavier Riley (QMUL, C4DM)
project "*Dig That Lick*"

- jazz bass lines, acc. of saxophone
- 48 tracks,
  24 recorded hours of melodies and improvisations
- qparse as backend of an audio-to-MIDI transcription procedure
- prior beat (measure) tracking

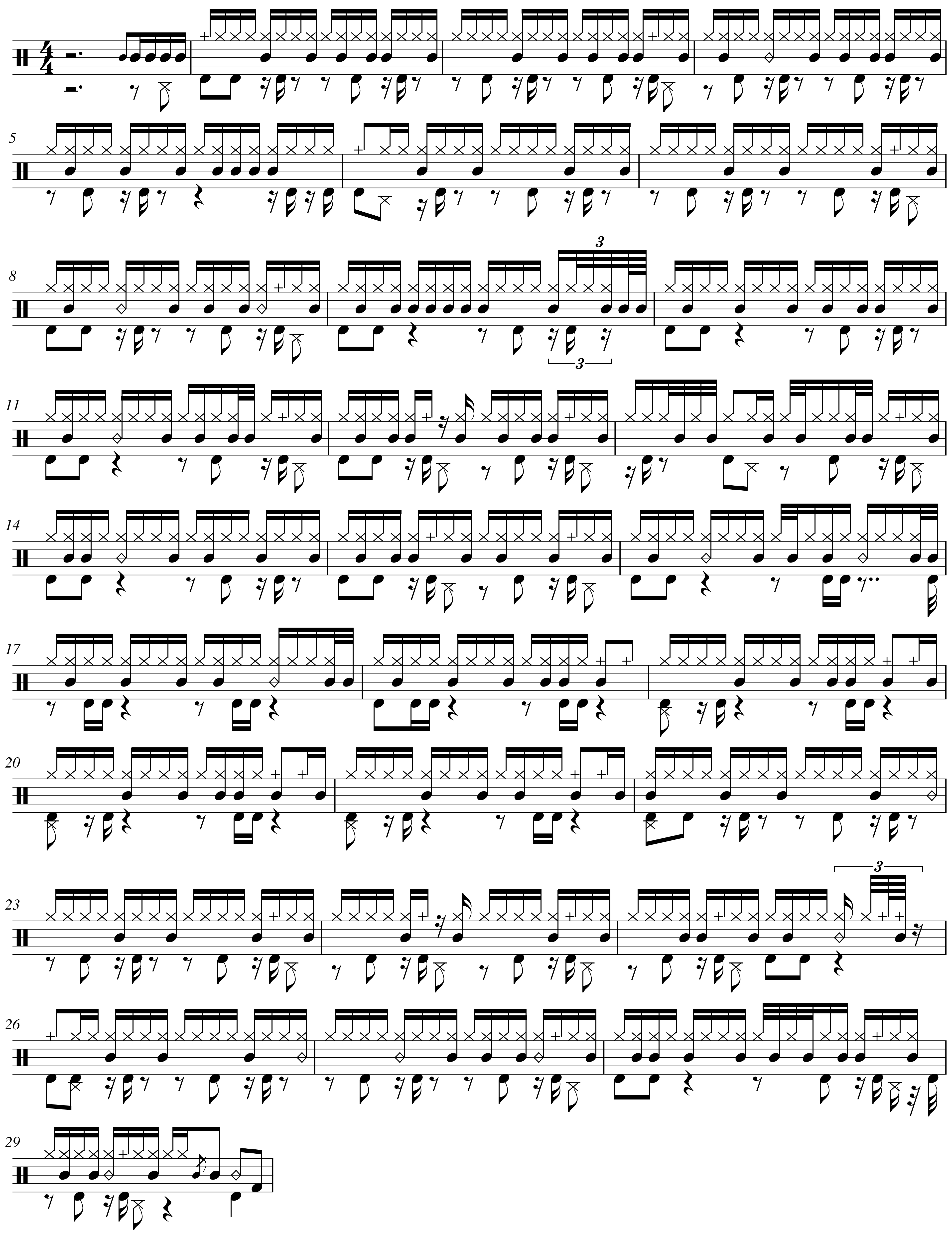
Groove MIDI Dataset

- by Google Magenta
- 13.6 hours, 1150 MIDI files, ~ 22000 measures recorded by professional drummers on a electronic drum kit
- audio (wav) files synthesized from (and aligned to) MIDI files for evaluation of audio-to-MIDI drum transcription
- no score files!

Scoring the GMD with qparse

Martin Digard (INALCO)

- all score files (XML) produced from the MIDI files with the same generic tree grammar (4/4 measure)
- polyphonic case-study, simpler than piano
- specific drumming constraints (hands ≤ 2, feet ≤ 2)
- processing errors from MIDI sensors



From Monophonic to Polyphonic Transcription, stepwise:

- From Monophonic to **Homophonic Transcription** (chords)
  Yusuke (Nagoya U.)

- **Drum Transcription** Martin Digard, Lydia Rodrigez-de la Nava
  Google GMD

- **Voice separation** - Lydia Rodrigez-de la Nava, evaluation Augustin Bouquillard
  integration for piano guitar transcription:
  - before parsing, or
  - after parsing (on intermediate model), or
  - joint with parsing.

- **Dataset ASAP** - Francesco Foscarin, Andrew Mc Leod
  MIDI and audio recording from Yamaha piano competition
  + XML scores
  + alignments
  + beat tracking annotations

MIDI-to-Score Automated Music Transcription approach

- quantitative parsing technique
  based on Symbolic Weighted formal language formalisms
  Tree Automata and word-to-word Transducers
- with prior quantitative language of notation *style*
  and prior IO measure
- (abstract) hierarchical score model
  as intermediate representation for score generation

- can handle complex notation cases:
  ornaments, mixed tuplets, mixed note durations, silences…
- efficient

- case studies: Monophonic, Drums
- ongoing work on Polyphonic case studies: guitar, piano

MERCI!
THANK YOU!