Rhythm Tree Rewriting

Jean Bresson (Ircam)
Pierre Donat-Bouillud (ENS Cachan/Rennes)
Florent Jacquemard (INRIA, Ircam)

IFIP WG1.6   July 13, 2014

general perspective

music symbolic representations
in particular traditional western music notation
processing of tree structured representations

general long term goal: symbolic MIR problems

http://music-ir.org/mirex

- querying bases of music scores
- transformations,
- melodic similarity detection,
- genre classification, recommendation,
- detection of repetitions, automatic segmentation, musicological analysis

one particular target application: rhythm transcription

target application

rhythm transcription: automatic generation of notation

plan of the talk

- context: computer assisted (Music) composition
- problem: rhythm transcription
- data structure: rhythm trees
- term rewriting approach
- weighted tree automata

1

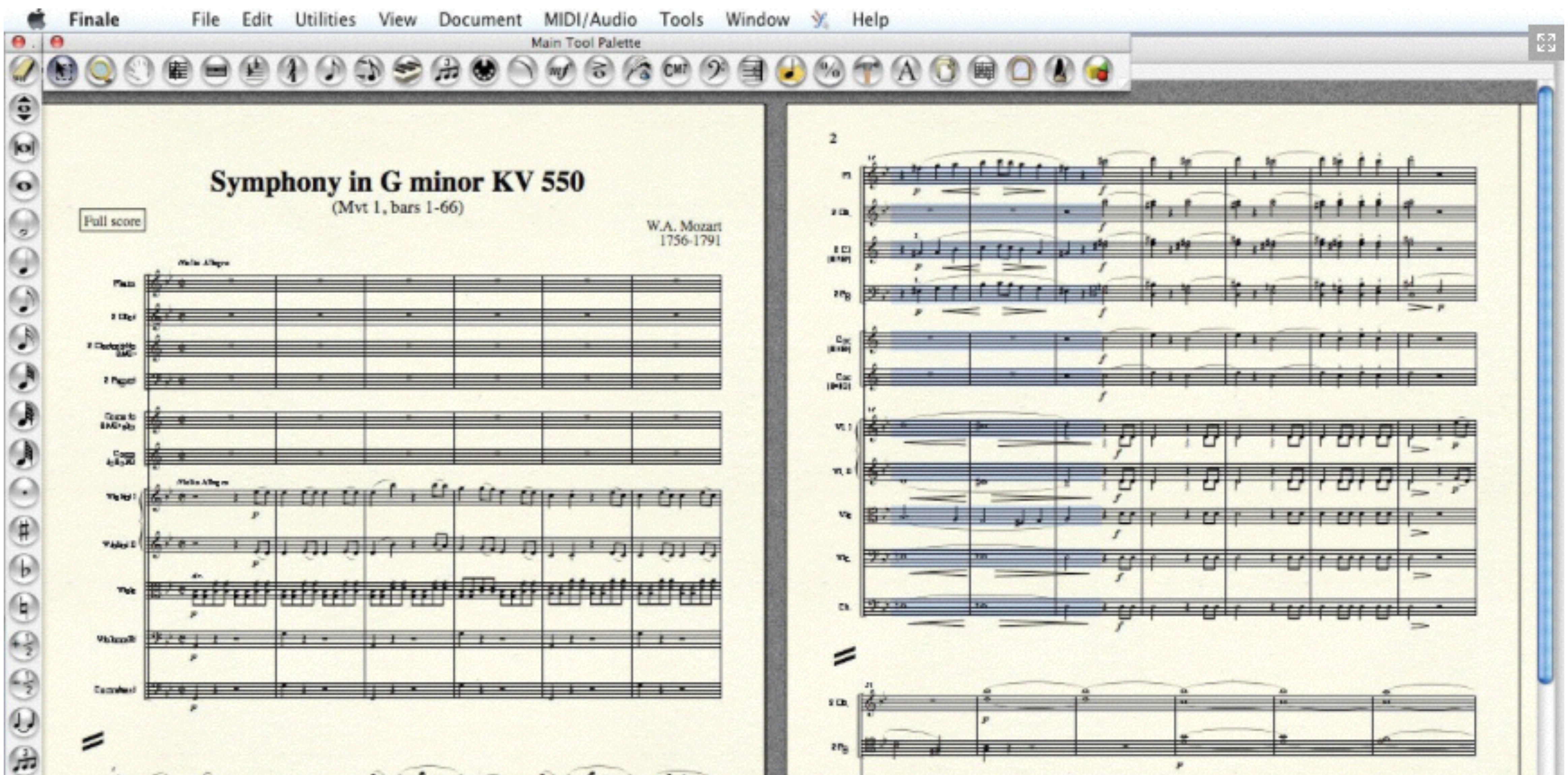
Assisted Composition

hardware and software tools for authoring music, production of music scores

music notation editors

score printing
import/export MIDI and MusicXML

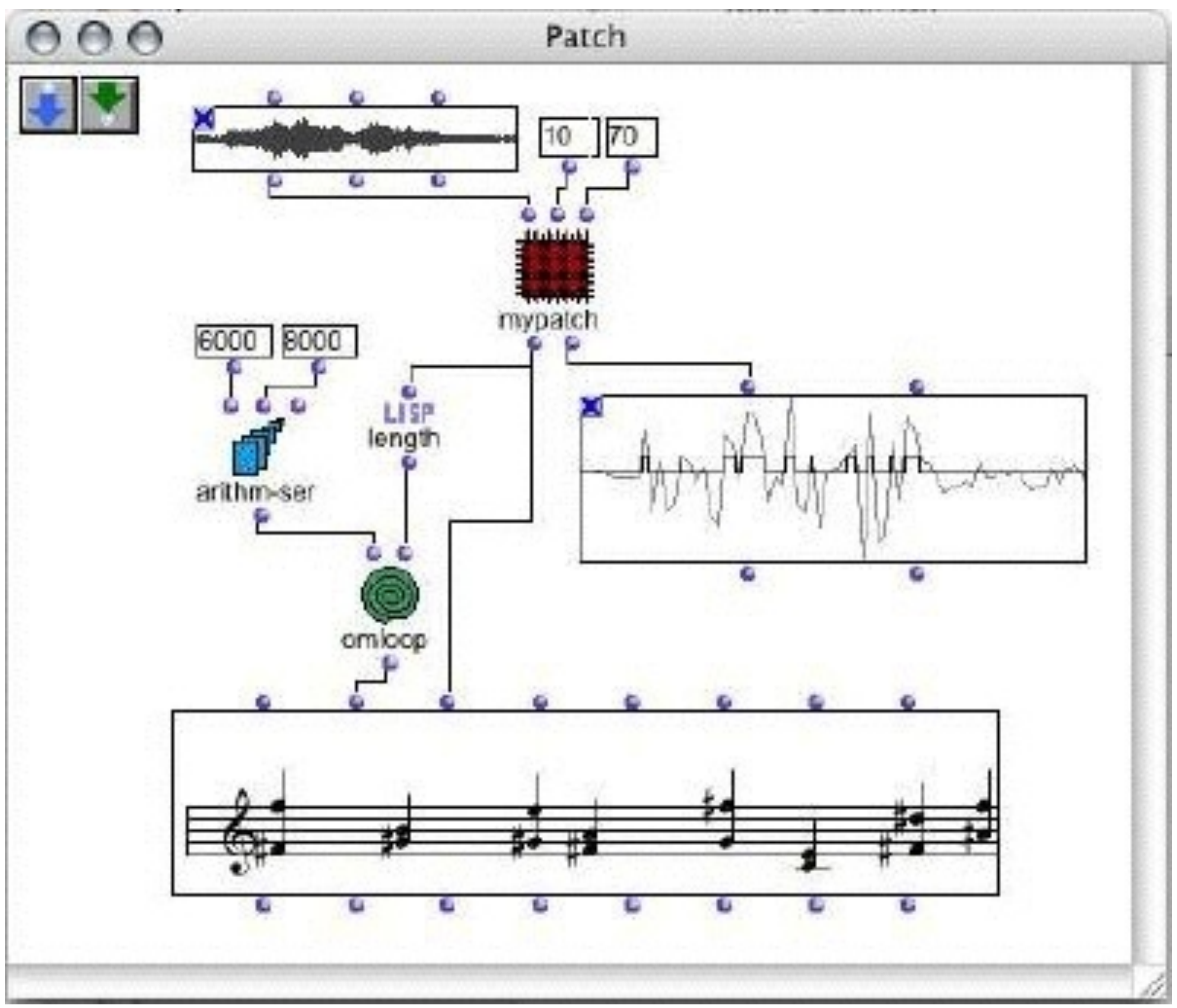
- Finale (MakeMusic)
- Sibelius (Avid)



algorithmic composition environments

solving musical problems
visual programming languages based on Lisp

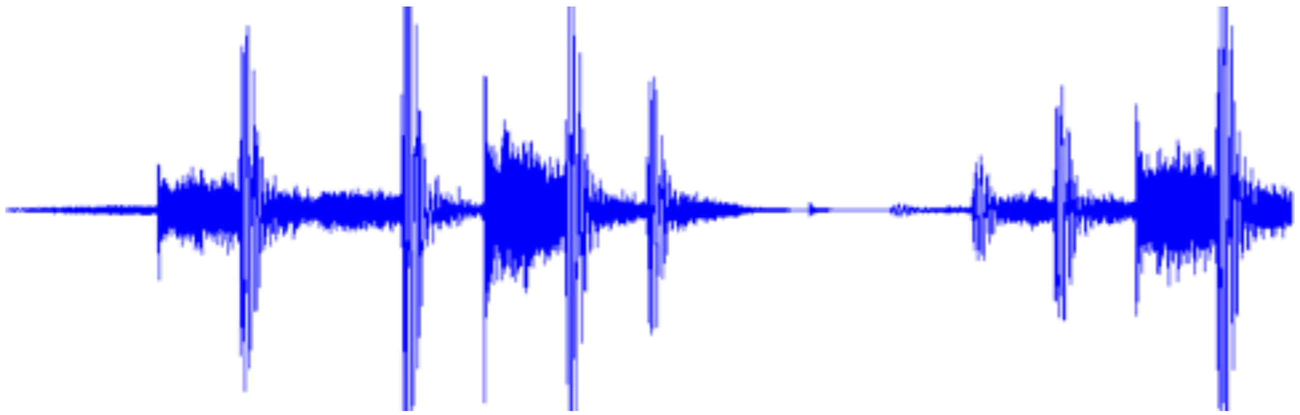
- PWGL (Sibelius Academy)
- OpenMusic (Ircam)

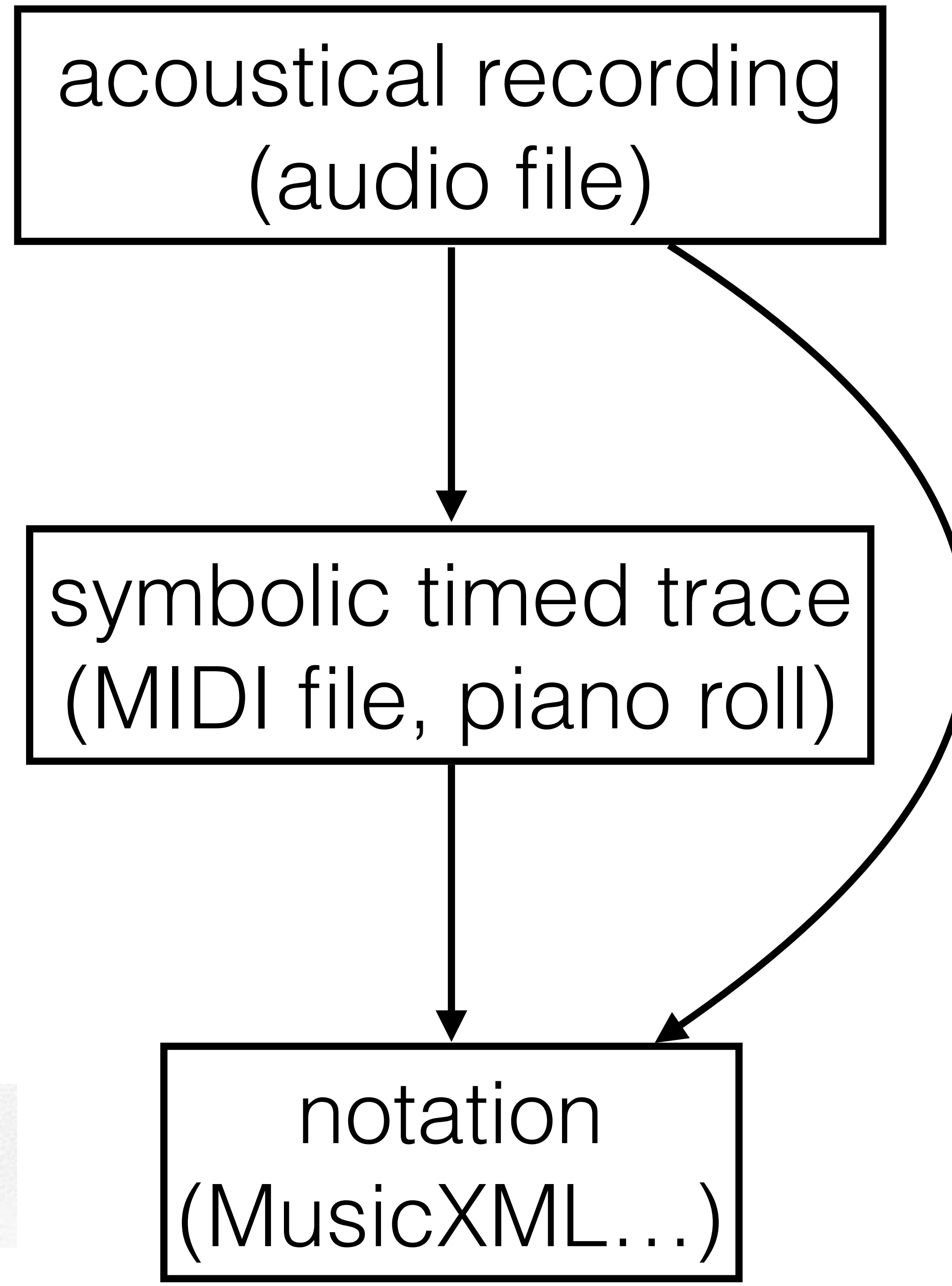


2

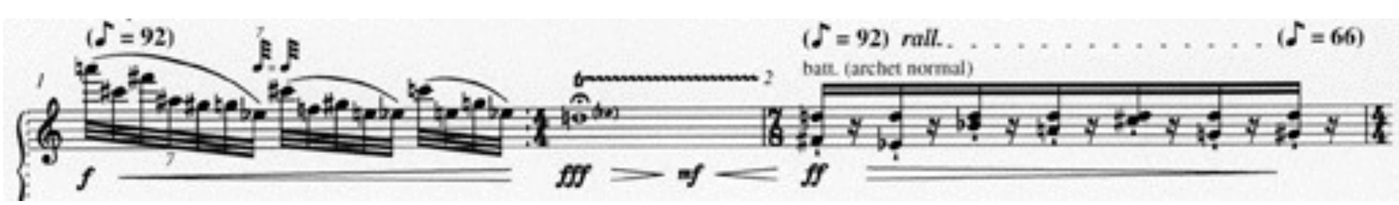
Automatic Music Transcription

automatic music transcription: goal

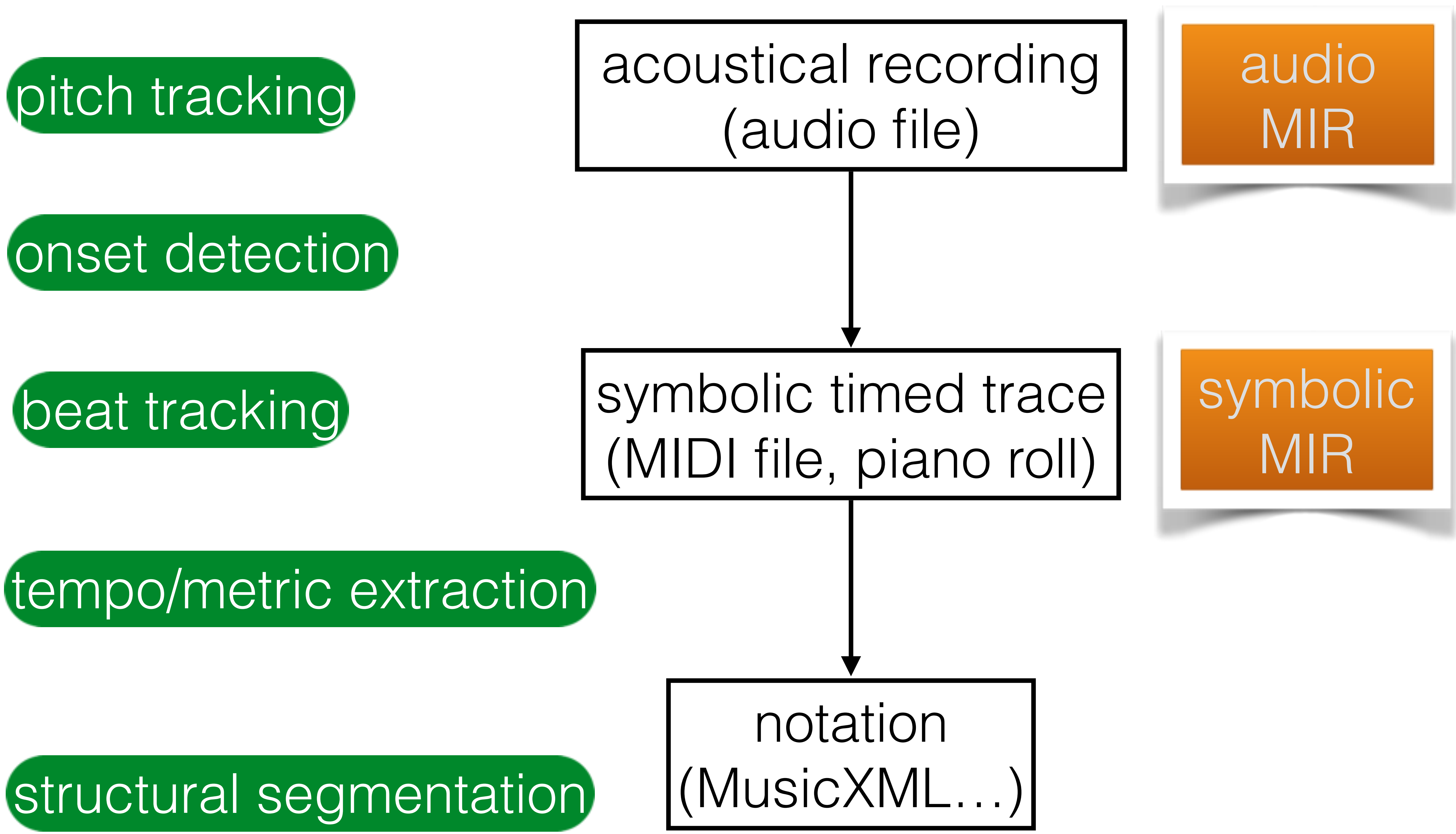




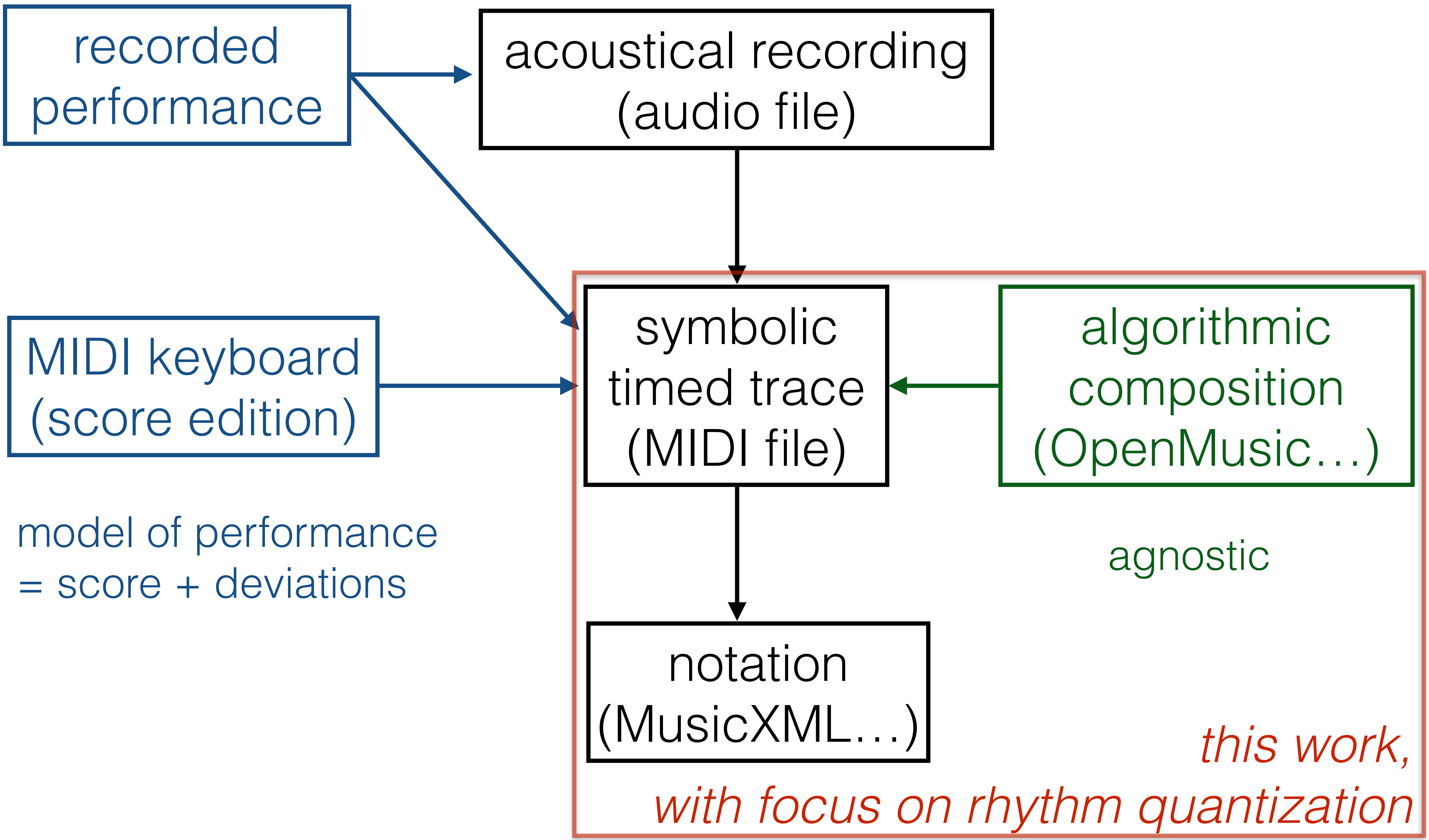
(71,0.0)(74,1.18)(46,1.73)(52,2.09)(5



automatic music transcription: tasks



automatic music transcription: applications

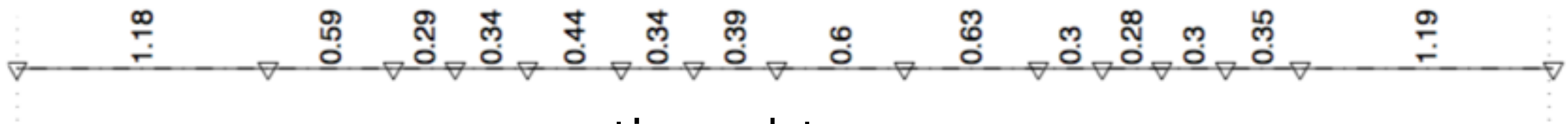


rhythm quantization (symbolic)

1. segmentation of the input timed trace
   each segment with constant tempo
   or known acceleration
2. identification of tempo / beat positions
3. local quantization on each segment
   (*on-the-beat quantization*)

- input: one segment = timed trace:
  sequence ***s*** of pairs (onset,duration) in $\mathbf{R}^2$ (ms)
- output: sequence ***t*** of pairs (onset, kind), with
  - onsets in $\mathbf{D}$, small discrete set
    (admissible subdivisions of beat)
  - kind = ‘note’ or ‘silence’
  - number of ‘notes’ = $|\boldsymbol{s}|$

rhythm quantization : measures of quality



timed trace

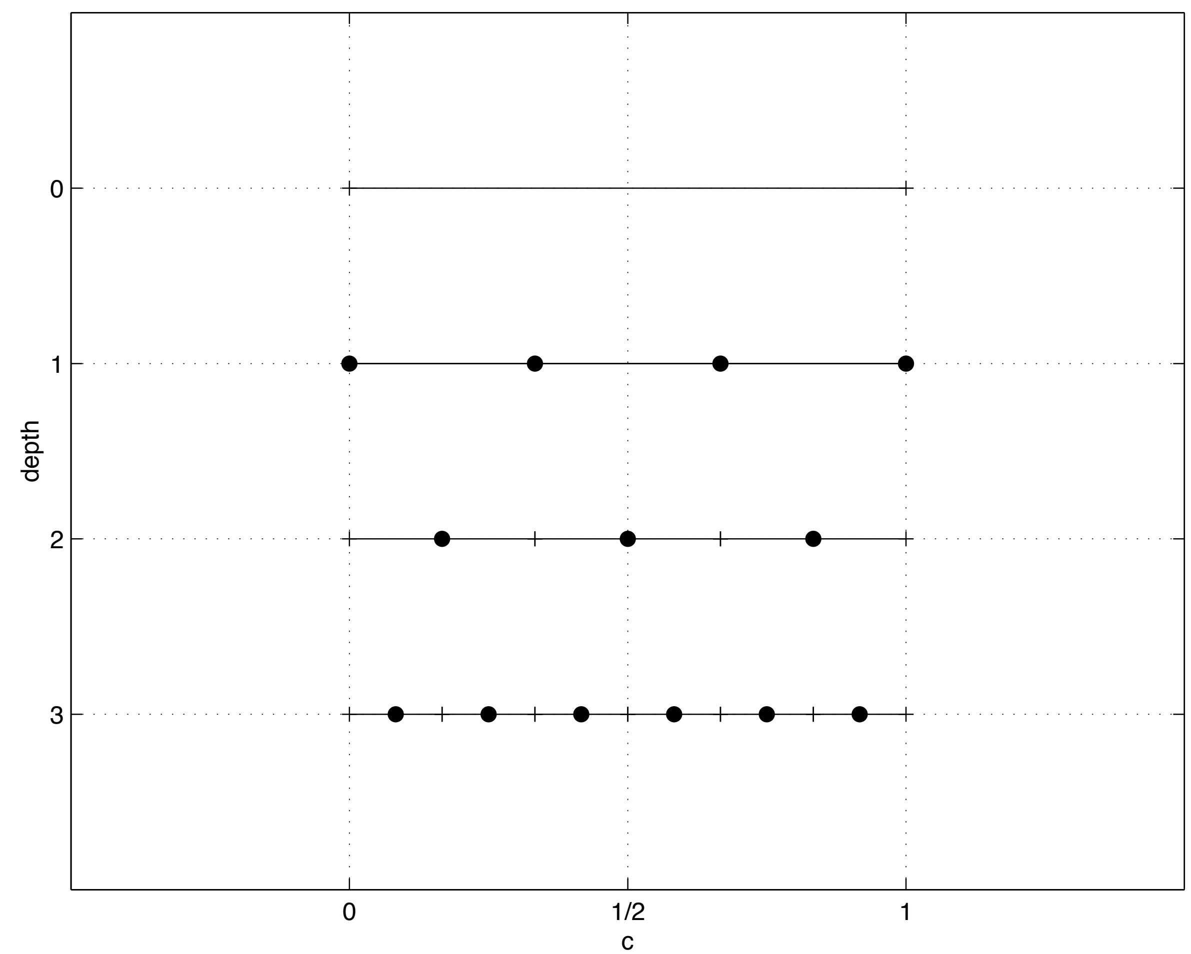
transcription 1: high precision, high complexity

transcription 2: lower precision, low complexity

alignment to grid

Longuet-Higgins
Mental Processes: Studies in Cognitive Science, 1987

Desain, Honing, de Rijk
Quantization of musical time
Music and Connectionnism, MIT Press 1991

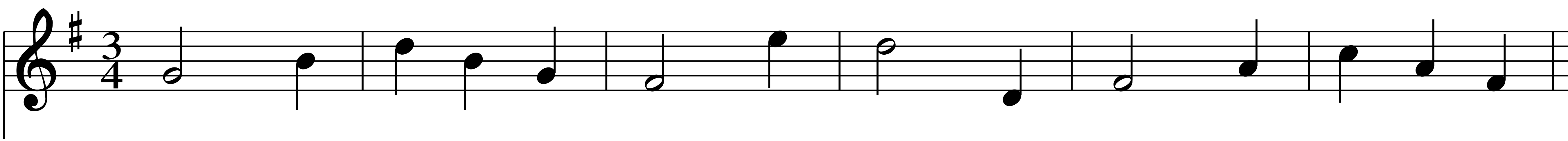


grid of depth 3 and subdivision schema (3, 2, 2)

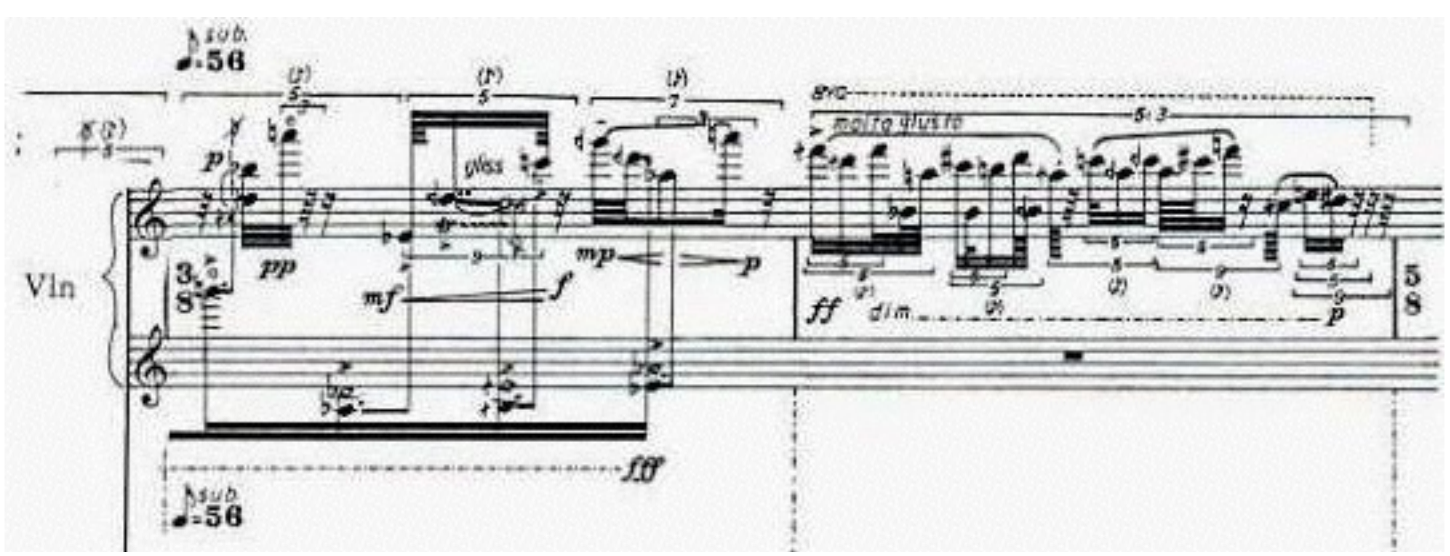
choice of the grid

according to depth, number of divisions…
with user parameters or heuristics

which grid for this?



which grid for that?



heuristics for grid selection

Pressing, Lawrence
Transcribe: A comprehensive autotranscription program
ICMC 1993

Agon, Assayag, Fineberg, Rueda
Kant: A critique of pure quantization
ICMC 1994

Meudic
OMquantify
PhD Ircam, 2005

- given a predefined set T of template grids (user parameters)
- align the input segment $\boldsymbol{s}$ to grids of T
- select the best grid $g \in T$ according to distance to $\boldsymbol{s}$
- return alignement of $\boldsymbol{s}$ to g (converted to a score)

local quantization workflow

OMquantify

- generation of several grids
- select best grid *g* (according to 3 distances)
- align input seq. ***s*** to *g*
- convert the alignement into OpenMusic Rhythm Tree return as score

RT based approaches

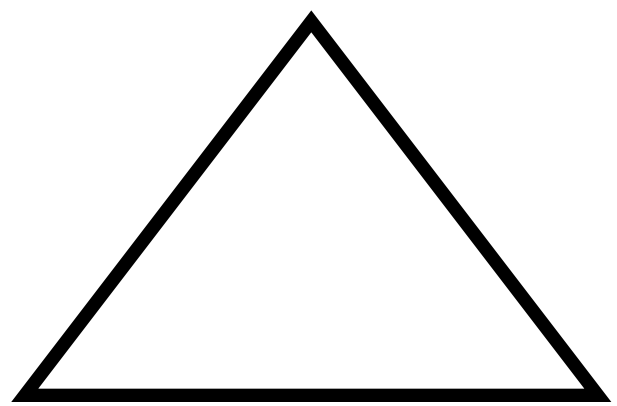
- convert input seq. ***s*** into RT
- computations on RT(s)
- return RT result as score

3

Rhythm Trees

rhythms and durations

rhythmic values
(fractions of the period)
= symbolic notation

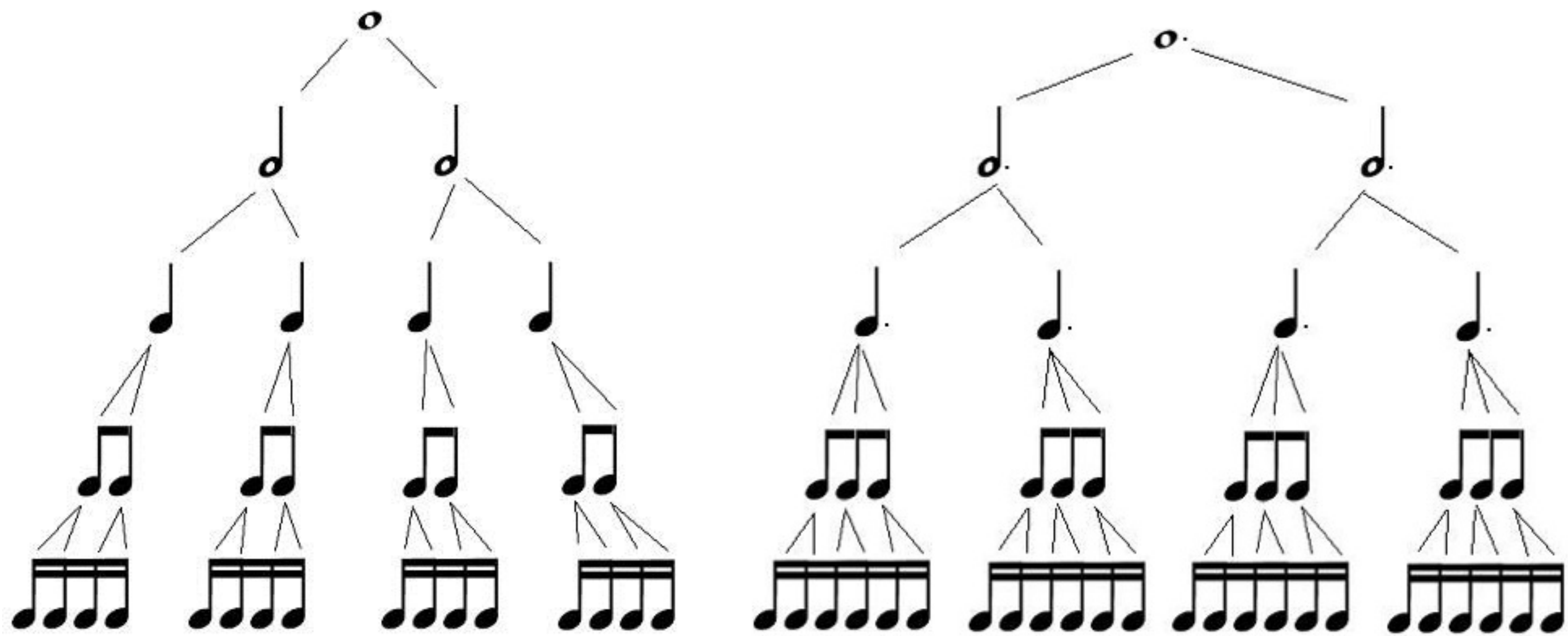


tempo (frequency)

durations (ms)

rhythm trees (RT)

in traditional western music notation:
durations are defined by recursive subdivisions of units



rhythm CF grammars

Lee
The rhythmic interpretation of simple musical sequences
Musical Structure and Cognition, 1985

$\frac{3}{4} \rightarrow$ 𝅗𝅥. | 𝄼. | $\frac{1}{4} + \frac{1}{4} + \frac{1}{4}$

$\frac{1}{4} \rightarrow$ ♩ | 𝄽 | $\frac{1}{8} + \frac{1}{8}$

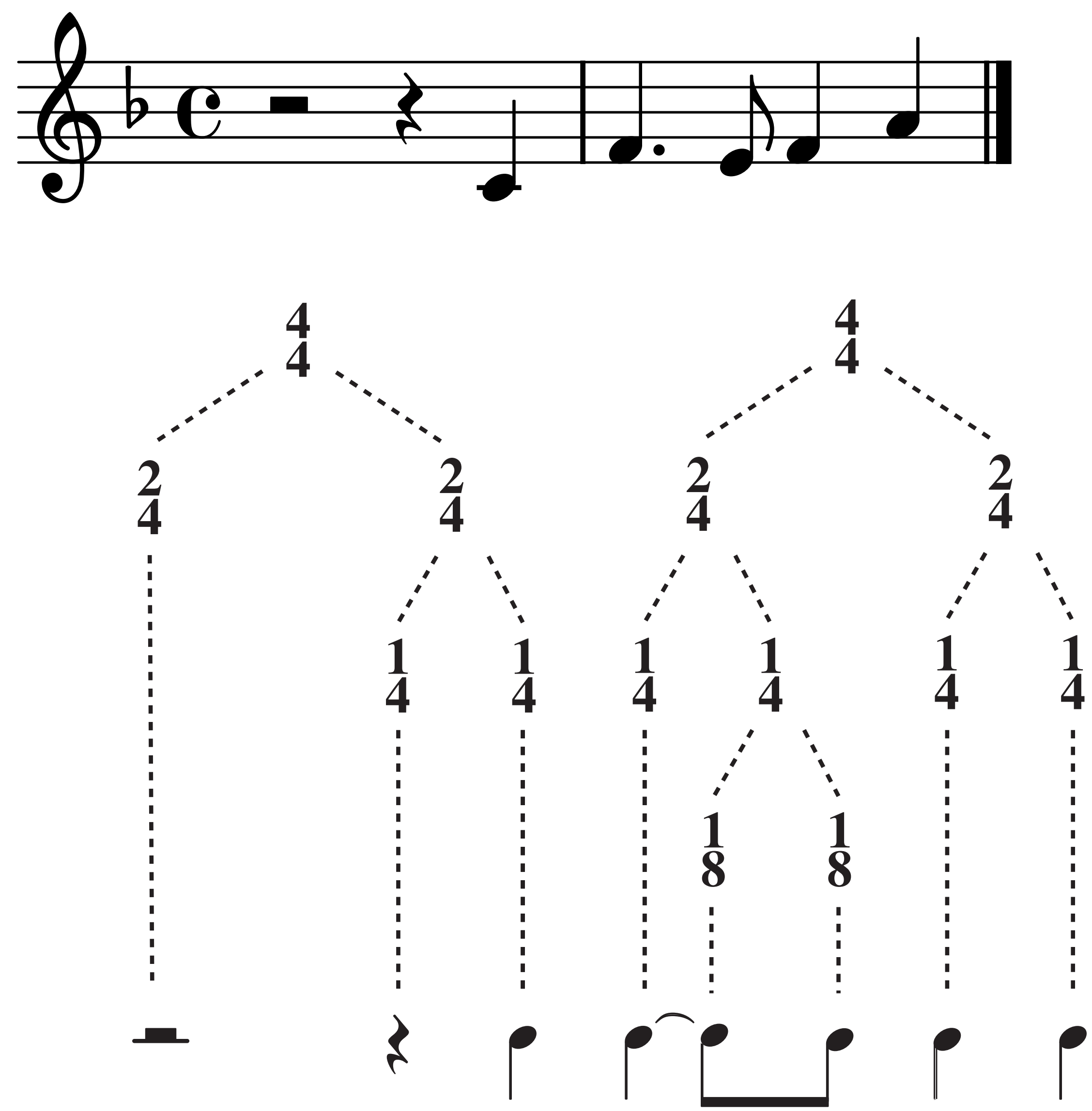
$\frac{1}{8} \rightarrow$ ♪ | 𝄾 | $\cdots$

𝄴 $\rightarrow$ 𝅝 | 𝄻 | $\frac{2}{4} + \frac{2}{4}$

$\frac{2}{4} \rightarrow$ 𝅗𝅥 | 𝄼 | $\frac{1}{4} + \frac{1}{4}$

$\frac{1}{4} \rightarrow$ ♩ | 𝄽 | $\frac{1}{8} + \frac{1}{8}$

$\frac{1}{8} \rightarrow$ ♪ | 𝄾 | $\cdots$

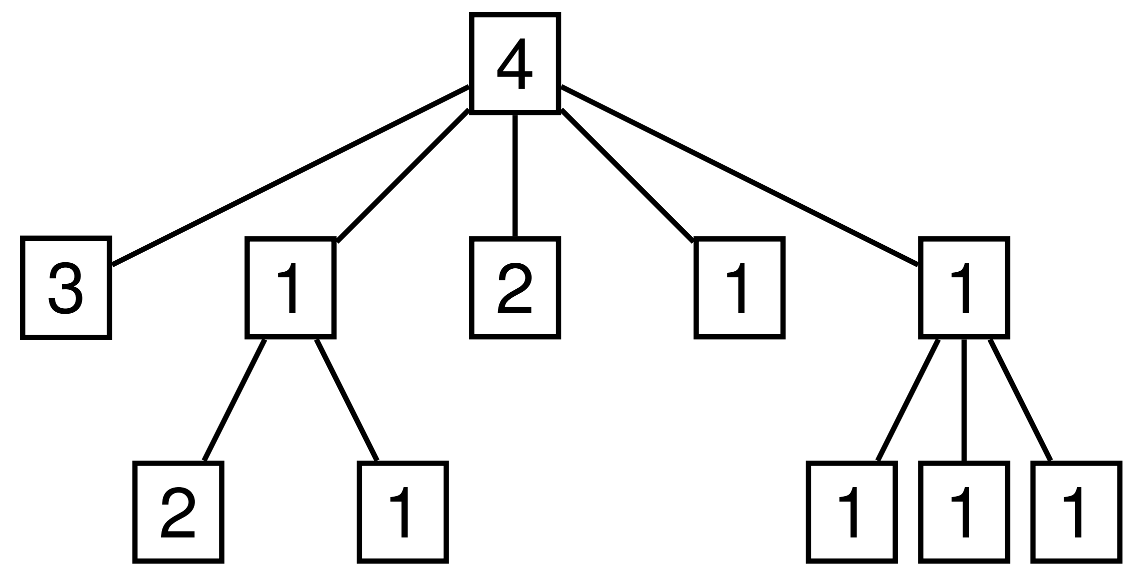


OM rhythm trees (OMRT)

trees are the standard data structure for OpenMusic

Laurson
Patchwork: A Visual Programming Language
Helsinki: Sibelius Academy, 1996

Agon, Haddad, Assayag
Representation et rendu de structures rythmiques
JIM, 2002



list `(4 (3 (1 (2 1)) 2 1 (1 (1 1 1))))`

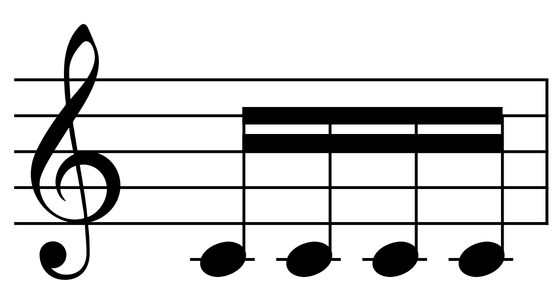
- labels in $\mathbf{N}$ = durations
- ∑ children = equal subdivisions of parent
- notes = leaves
- for numerical computations (not symbolic)

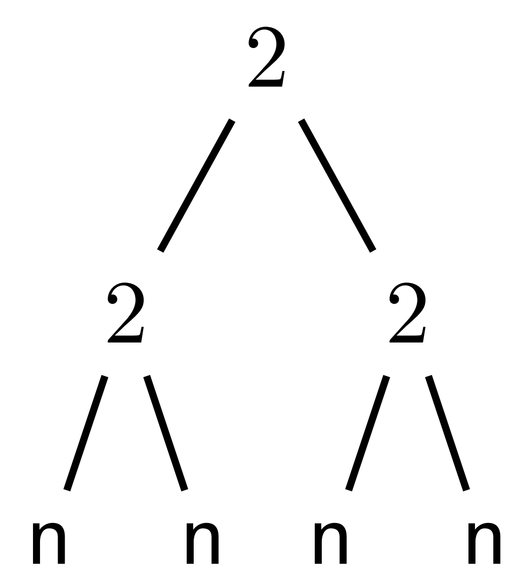
symbolic RT

- all sibling represent equal durations
- for symbolic computations (unary notation)

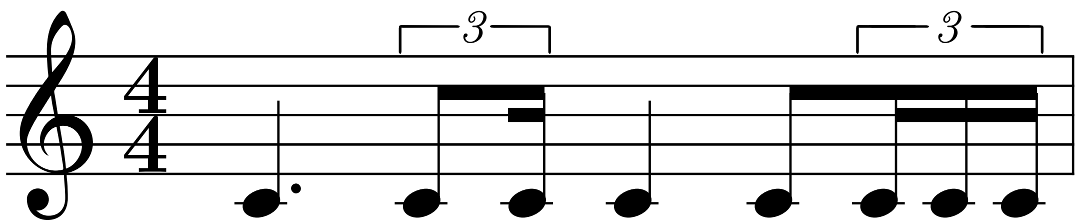
terms over the following signature:

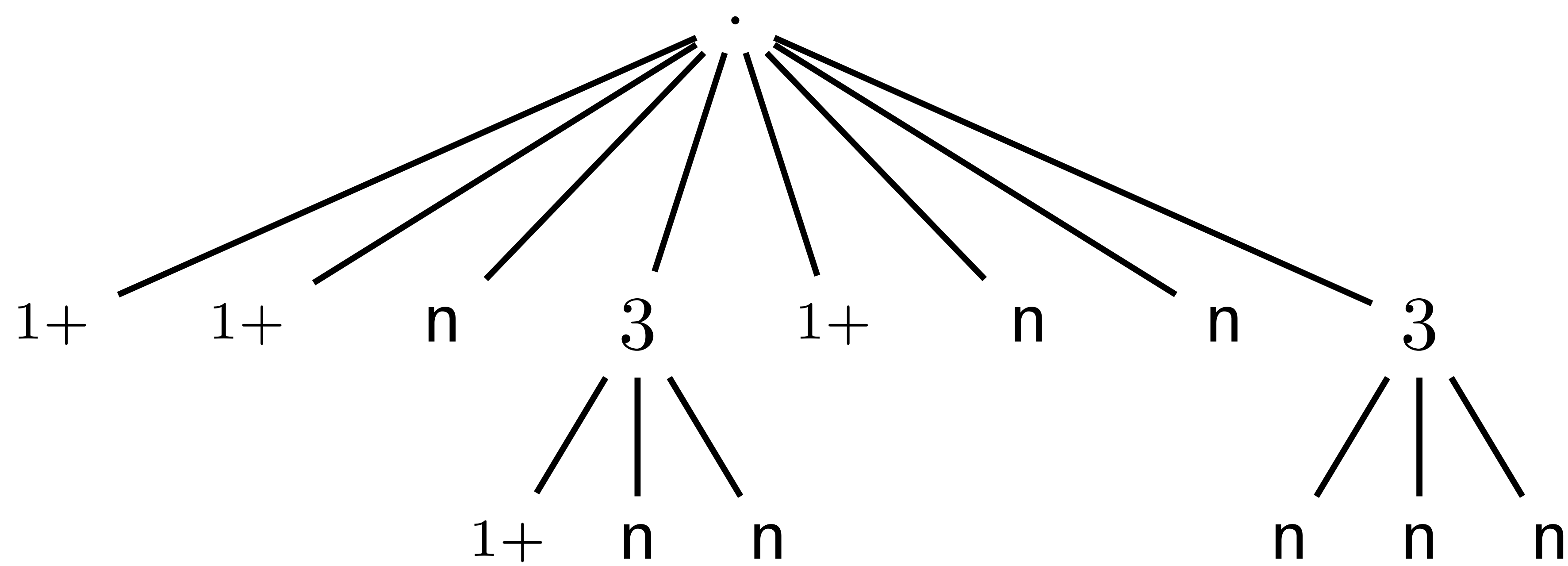
- inner nodes: labelled with arity = prime numbers 2-13
- leaves:
  - n : note
  - r : rest
  - s : slur. Sum with previous leaf in dfs ordering
  - 1+ : add to next sibling



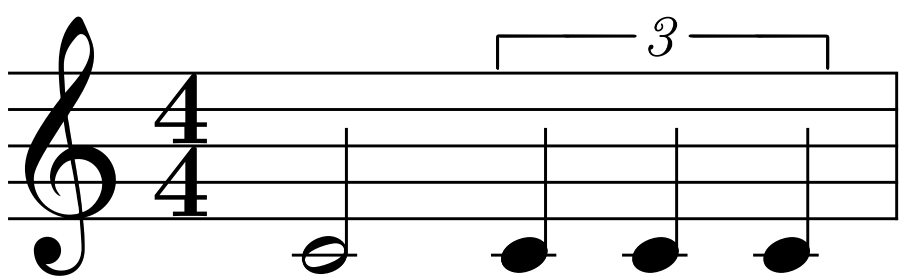


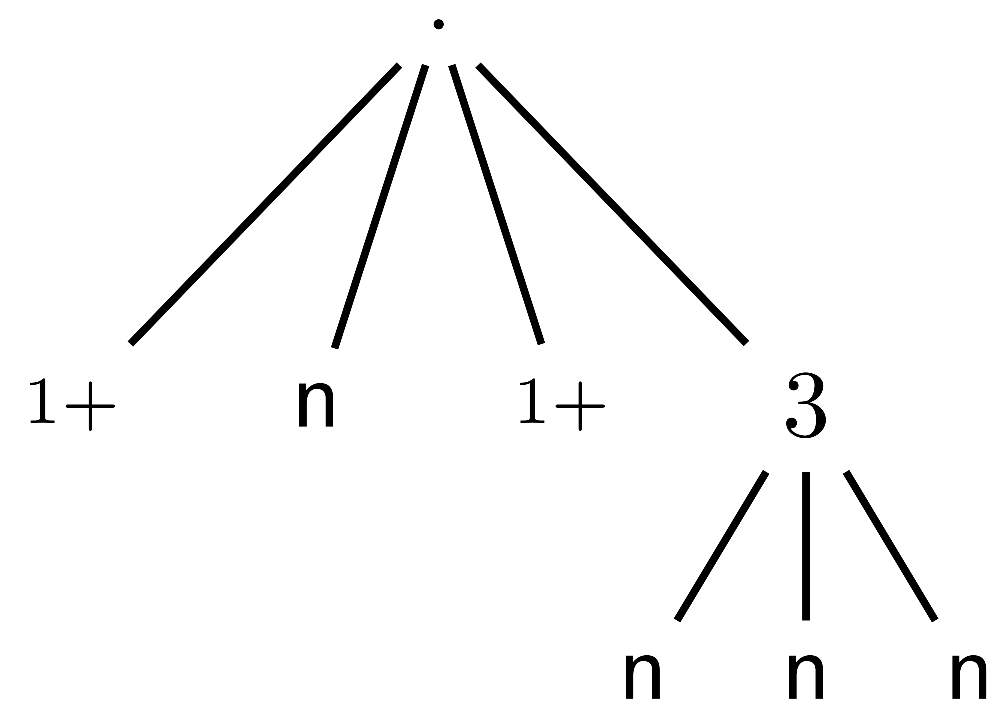
symbolic RT (example 2)





symbolic RT (example 3)





advantage of RT representation

over string representation

- close to traditional music notation
- keep the integrity constraint
  sum of durations = 1
- groups of correlated events
  reflected in the tree structure
  = sequences of siblings
  → preserved in local transformations

4

Local Quantization by RT Rewriting

principle

1. generate an initial tree $\boldsymbol{t_0}$ from input $\boldsymbol{s}$
   with maximum precision, maximum complexity
   - $\boldsymbol{t_0}$ complete
   - closest to $\boldsymbol{s}$ given maximum depth, signature
   - alignment to a complete grid

2. simplify $\boldsymbol{t_0}$ into $\boldsymbol{t}$ using a set of rewrite rules
   of 2 kinds:
   - conservative rules (preserve durations)
   - simplifying rules (do not preserve durations)

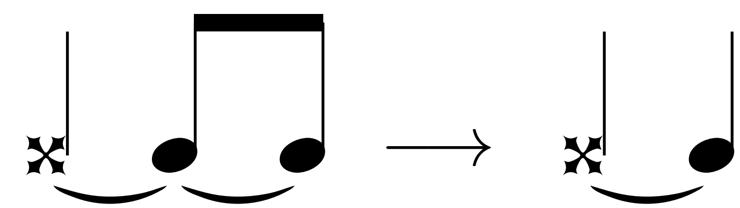
3. return score corresponding to $\boldsymbol{t}$

conservative rules

merge-ns $p(\mathsf{n}, \mathsf{s}, \ldots, \mathsf{s}) \to \mathsf{n}$

merge-r $p(\mathsf{r}, \ldots, \mathsf{r}) \to \mathsf{r}$

merge-s $p(\mathsf{s}, \ldots, \mathsf{s}) \to \mathsf{s}$



replace-s

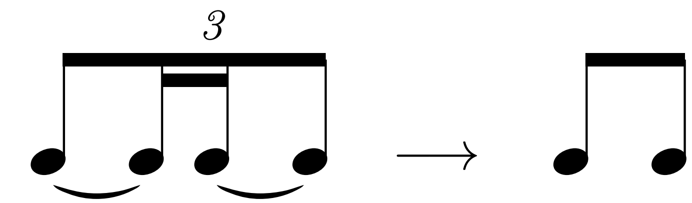
$$p(x_1, \ldots, x_m, t, \underbrace{1+, \ldots, 1+}_{k}, \mathsf{n}, y_1, \ldots, y_n)$$

$$\to p(x_1, \ldots, x_m, t, \mathsf{n}, \underbrace{\mathsf{s}, \ldots, \mathsf{s}}_{k}, y_1, \ldots, y_n)$$

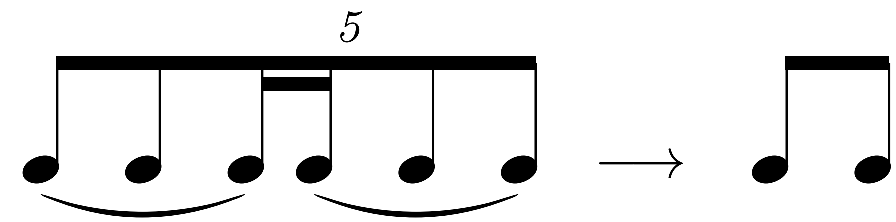
$$t = \mathsf{n} \text{ or } t = \mathsf{r} \text{ or } t = \mathsf{s},\ t = p(z_1, \ldots, z_p)$$

conservative rules (2)

3/2 $\quad 3(\mathsf{n}, 2(\mathsf{s}, \mathsf{n}), \mathsf{s}) \to 2(\mathsf{n}, \mathsf{n})$

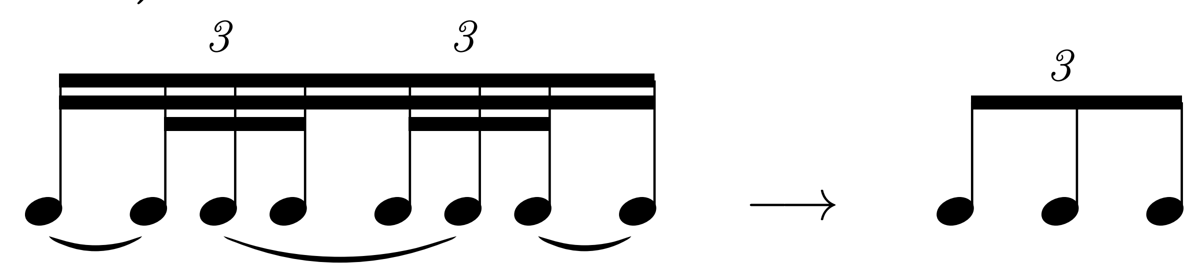


5/2 $\quad 5(\mathsf{n}, \mathsf{s}, 2(\mathsf{s}, \mathsf{n}), \mathsf{s}, \mathsf{s}) \to 2(\mathsf{n}, \mathsf{n})$

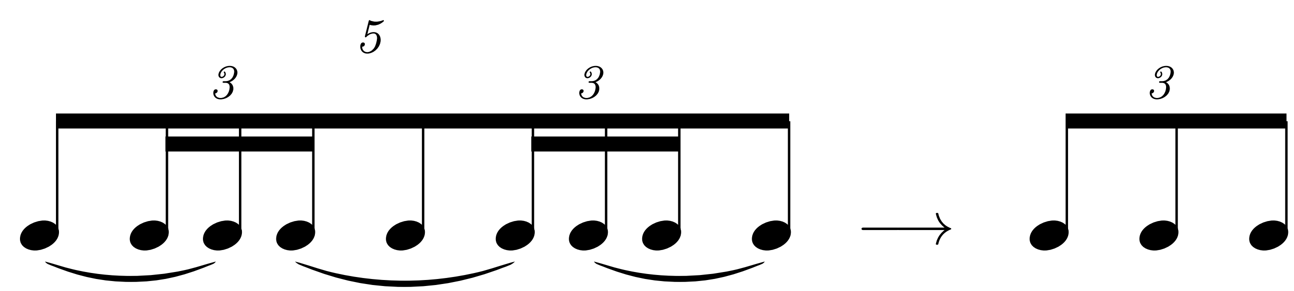


7/2, 11/2,…

4/3 $\quad 2\big(2(\mathsf{n}, 3(\mathsf{s}, \mathsf{n}, \mathsf{s})), 2(3(\mathsf{s}, \mathsf{s}, \mathsf{n}), \mathsf{s})\big) \to 3(\mathsf{n}, \mathsf{n}, \mathsf{n})$



5/3 $\quad 5(\mathsf{n}, 3(\mathsf{s}, \mathsf{s}, \mathsf{n}), \mathsf{s}, 3(\mathsf{s}, \mathsf{n}, \mathsf{s}), \mathsf{s}) \to 3(\mathsf{n}, \mathsf{n}, \mathsf{n})$



…

conservative rules (application)

equational theory of rhythm notation

the set of conservative rules is
- confluent and terminating

it can be used as a tool for
- simplifying rhythm notations
- identify equivalent rhythm notations

simplifying rules

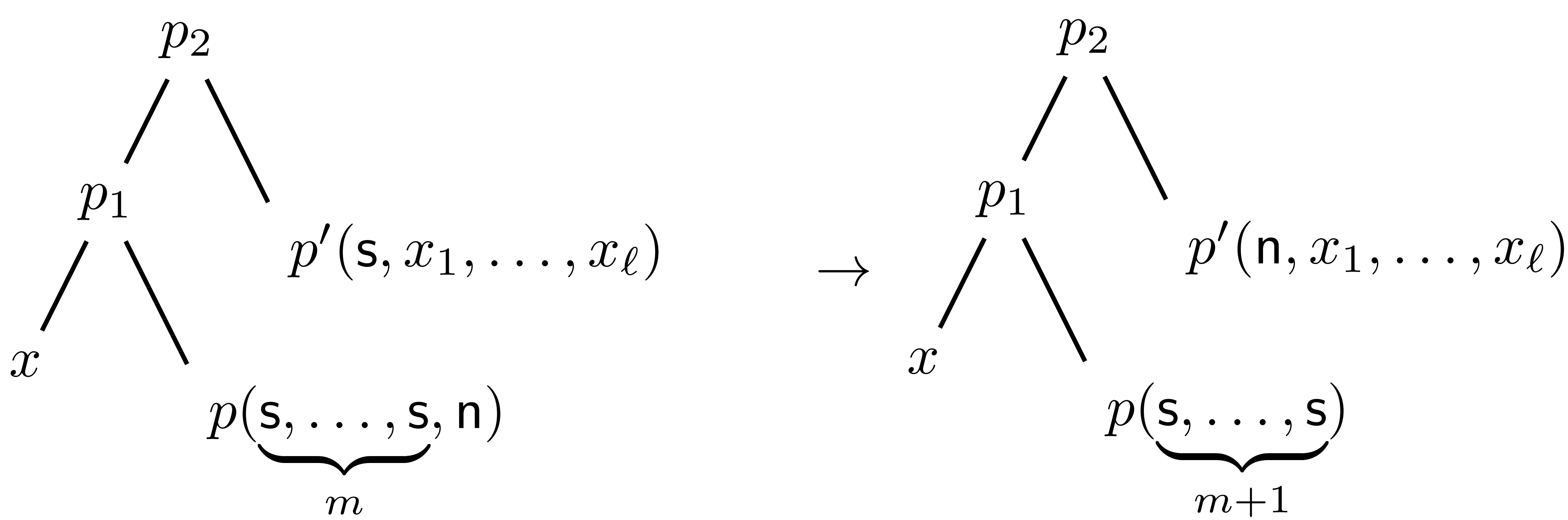
reduce

$$p(\mathsf{s},\mathsf{n},x_1,\ldots,x_m) \quad\to\quad p(\mathsf{n},\mathsf{s},x_1,\ldots,x_m)$$
$$p(\mathsf{s},\mathsf{r},x_1,\ldots,x_m) \quad\to\quad p(\mathsf{r},\mathsf{r},x_1,\ldots,x_m)$$

inflate

$$p(\underbrace{\mathsf{s},\ldots,\mathsf{s}}_{m},\mathsf{r}) \to p(\underbrace{\mathsf{s},\ldots,\mathsf{s}}_{m+1})$$

$$p_1\big(p(\underbrace{\mathsf{s},\ldots,\mathsf{s}}_{m},\mathsf{n}),p'(\mathsf{s},x_1\ldots,x_k)\big) \to p_1\big(p(\underbrace{\mathsf{s},\ldots,\mathsf{s}}_{m+1}),p'(\mathsf{n},x_1\ldots,x_k)\big)$$



TRS with regexp constraints

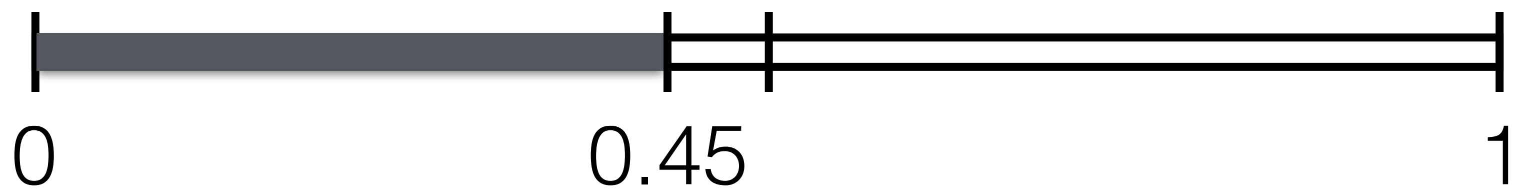
- there is an exponential number of simplifying rules
- currently implemented as LISP functions
  (1 function represents a family of rules)
- studying other compact rule-based representation, with
  - tree, context and function variables
  - regular constraints (variable in regular tree language)

Kutsia, Marin
Matching with Regular Constraints, 2005

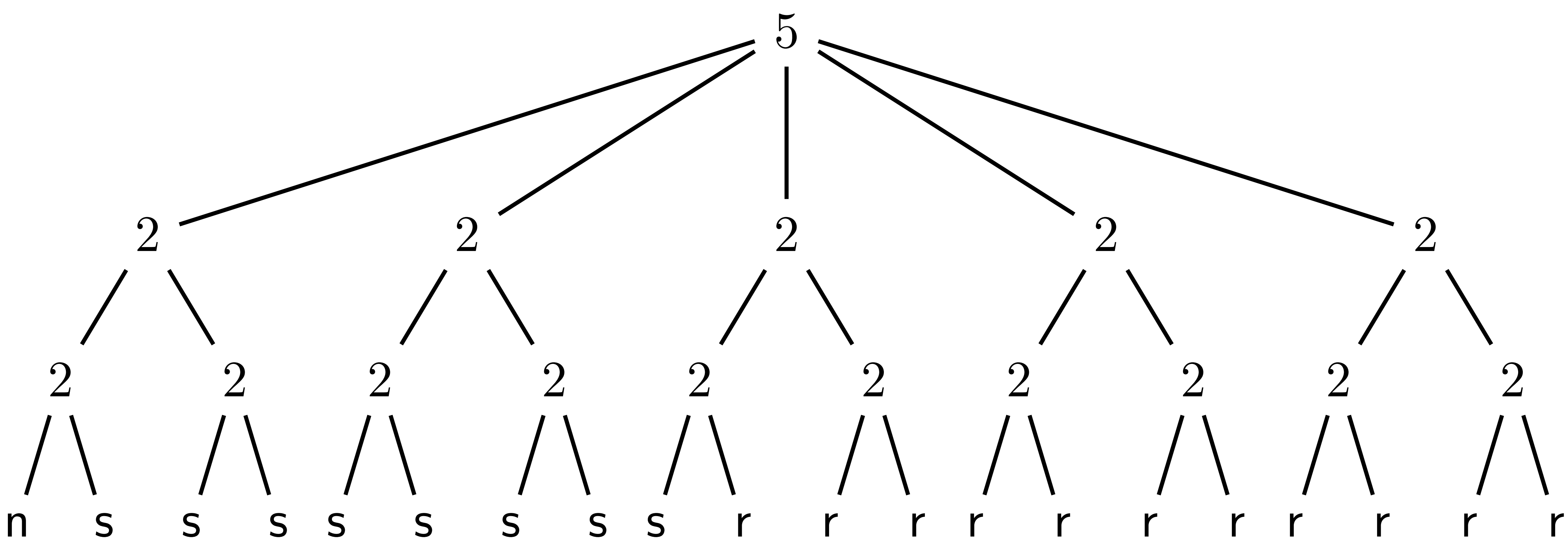
problems:

- matching
- rewriting strategies (bottom-up)
- characterization of set of descendants

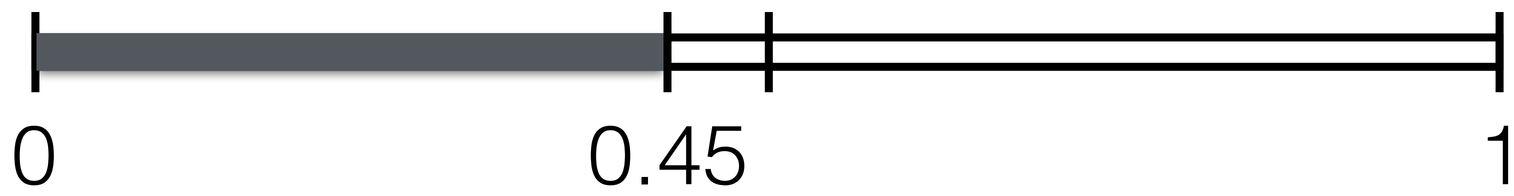
example

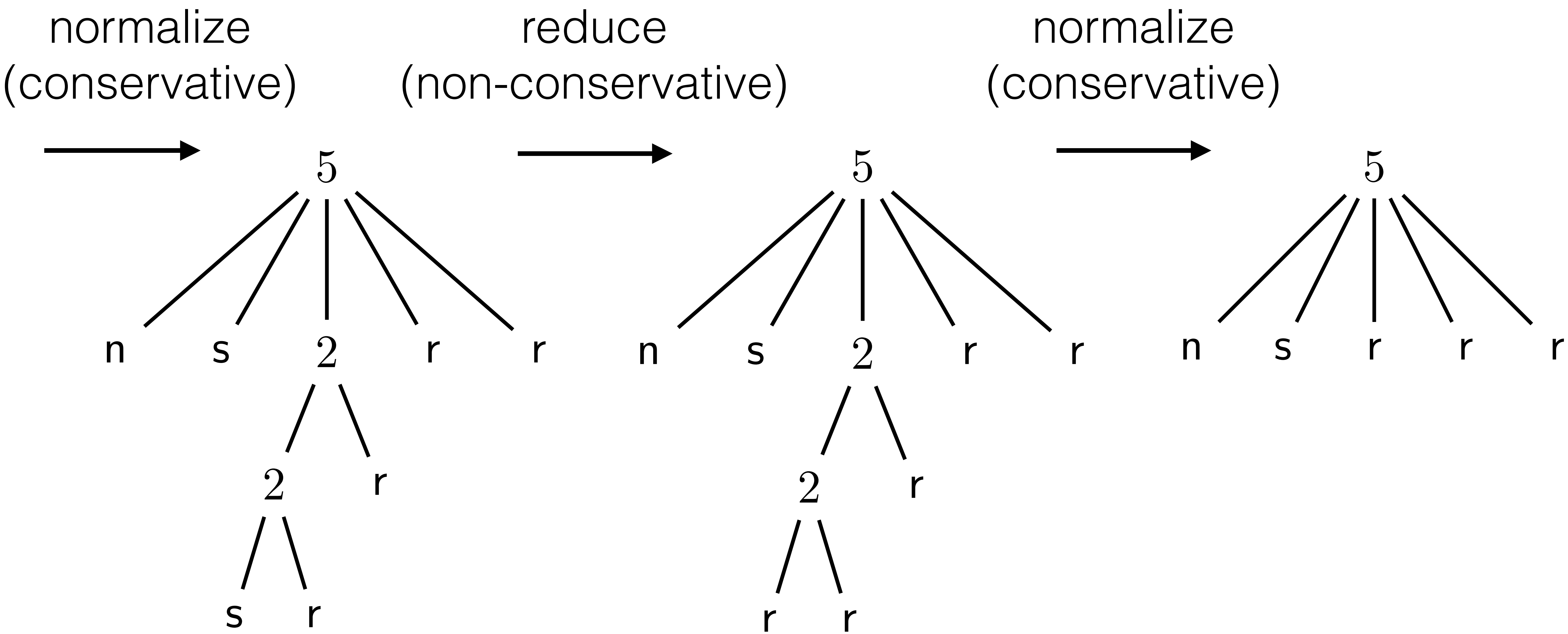


initial tree (complete)

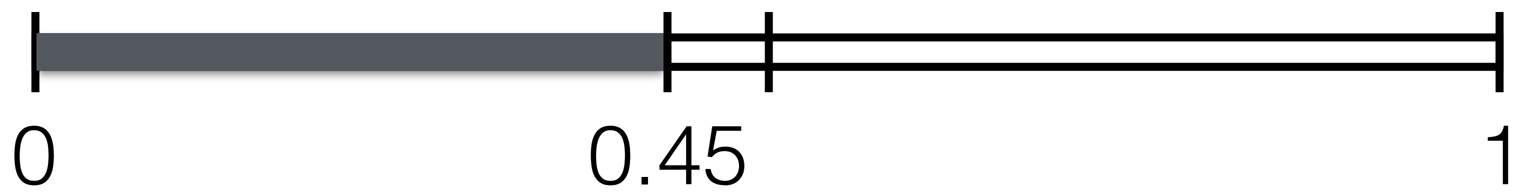


example

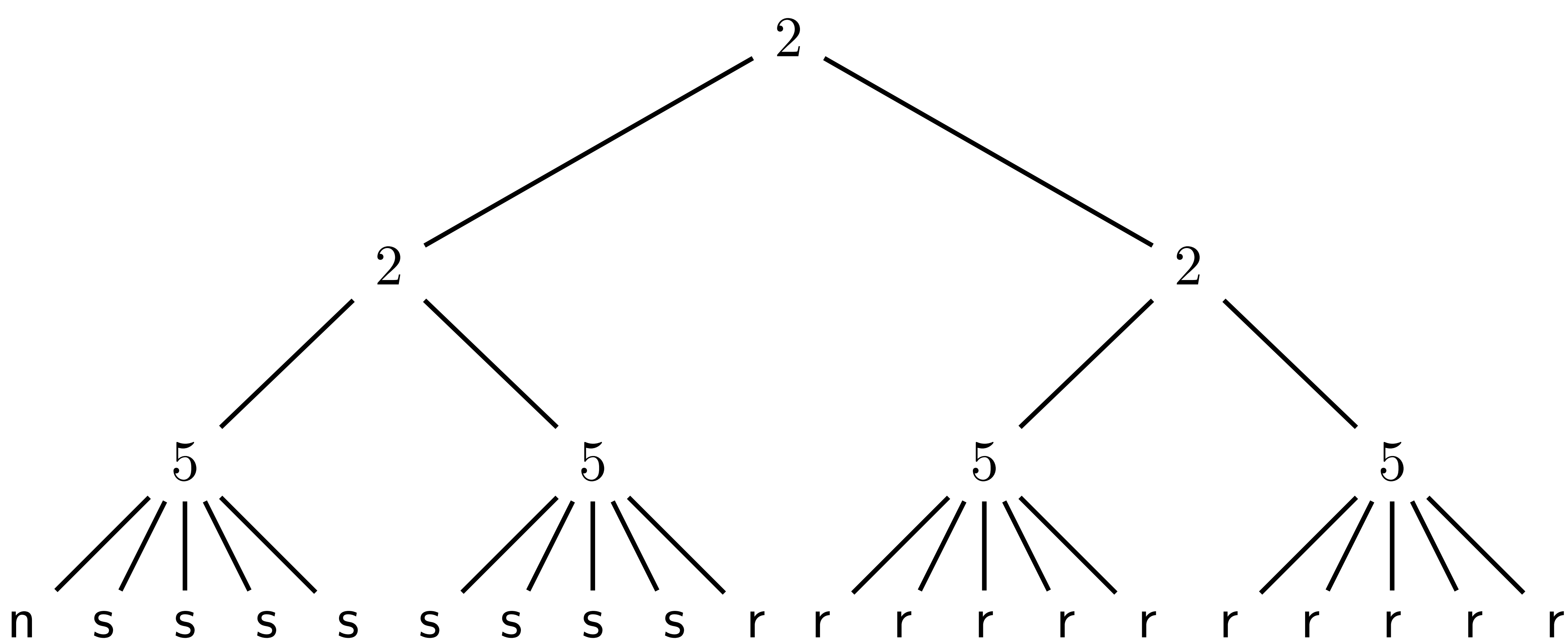

0
0.45
1


normalize
(conservative)
reduce
(non-conservative)
normalize
(conservative)
5
n s 2 r r
2 r
s r
5
n s 2 r r
2 r
r r
5
n s r r r

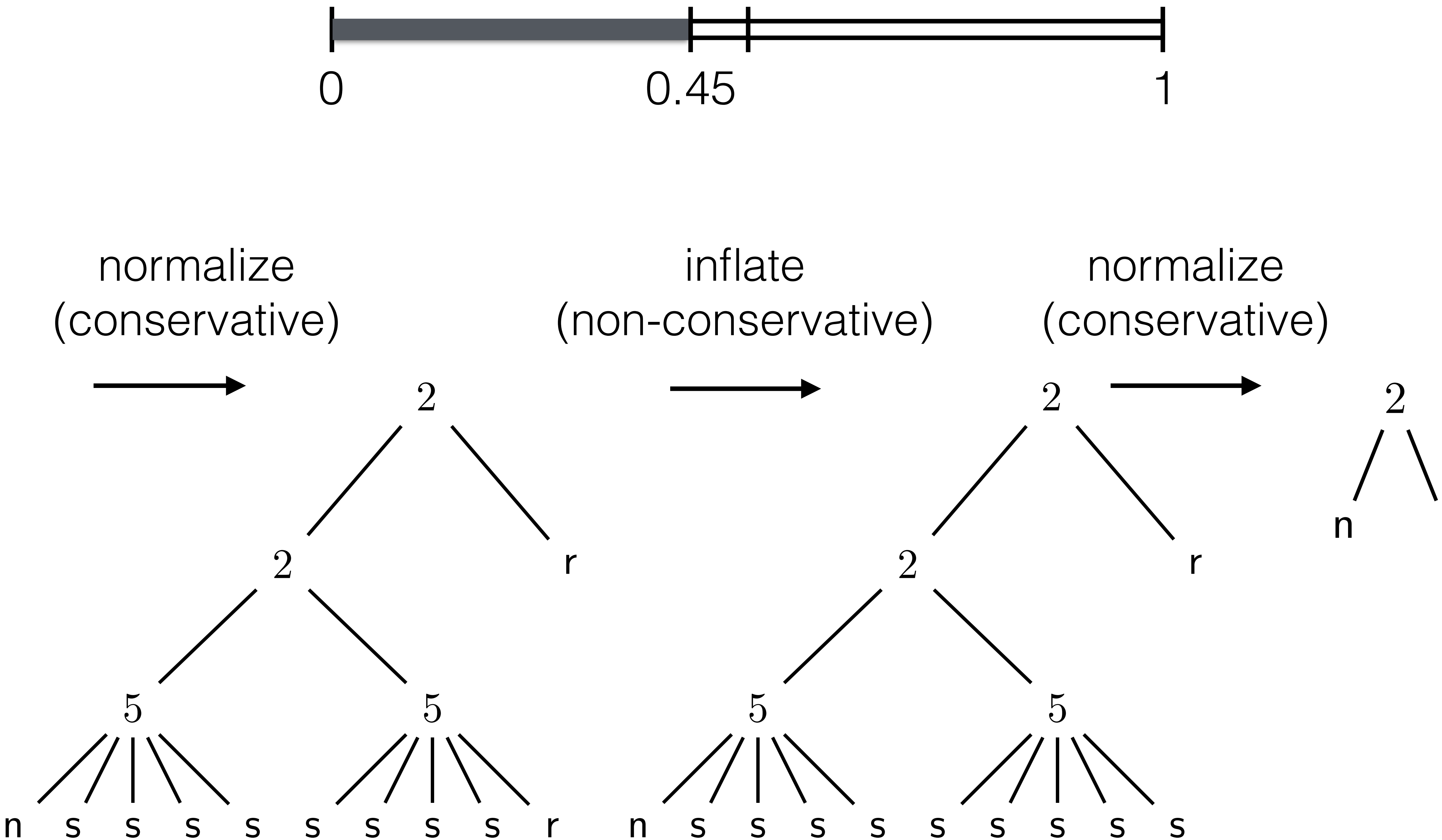
example (2)



other initial tree (complete)



example



0
0.45
1
normalize
(conservative)
inflate
(non-conservative)
normalize
(conservative)
2
2
r
5
5
n s s s s s s s s r
2
2
r
5
5
n s s s s s s s s s
2
n r

summary and questions

1. generate an initial tree $\boldsymbol{t_0}$ from input $\boldsymbol{s}$
   → several possibilities of $\boldsymbol{t_0}$
2. simplify $\boldsymbol{t_0}$ into $\boldsymbol{t}$
   - simplifying rules diverge

   → (symbolic) exploration of space of reachable $\boldsymbol{t}$
   → choice of best $\boldsymbol{t}$ according to
     - complexity and
     - distance to $\boldsymbol{s}$
3. return score corresponding to $\boldsymbol{t}$
   → conversion of RT into notation (through OMRT)

5

Weighted Tree Automata

→ the set of initial trees, given an input sequence $\boldsymbol{s}$
is a regular tree language $\boldsymbol{L_s}$,
recognized by a tree automaton $\boldsymbol{A_s}$.

→ exploration: the closure of $\boldsymbol{L_s}$ under rewriting
is characterized by a tree automaton $\boldsymbol{A_s}^*$
constructed incrementally from $\boldsymbol{A_s}$
and the rewrite rules (tree automata completion)

→ choice of the best $\boldsymbol{t}$:
minimizing complexity and distance to $\boldsymbol{s}$
both computed by weighted tree automata

space exploration, selection best choice

Bayesian approach to quantization

Cemgil, Desain, Kappen
Rhythm Quantization for Transcription
Computer Music Journal 24(2), 2000.

Cemgil
Bayesian Music Transcription
PhD Radboud University of Nijmegen, 2004

$$p(\boldsymbol{t} \mid \boldsymbol{s}) = p(\boldsymbol{s} \mid \boldsymbol{t}) \, . \, p(\boldsymbol{t})$$

dist($\boldsymbol{s}$, $\boldsymbol{t}$)
square of a weighted
Euclidian distance

$$\frac{1}{\text{score complexity}}$$

(length of Shannon code in bits)

ad hoc performance model $\boldsymbol{s} = \boldsymbol{t} / v + \varepsilon$
v = tempo (constant)
ε = expressive timing deviation

space exploration, selection of best choice

weighted tree automaton (WTA) $\boldsymbol{A}$:
finite bottom-up tree automaton
with a weight for each transition.
associates a weight $\boldsymbol{A}(\boldsymbol{t})$ to each term $\boldsymbol{t}$.

choice of $\boldsymbol{t}$ for $\boldsymbol{s}$ minimizing $\text{dist}(\boldsymbol{s}, \boldsymbol{t}) \,.\, \mathit{cpty}(\boldsymbol{t})$

$$= \boldsymbol{A}_{\boldsymbol{ws}}^{*}(\boldsymbol{t}) \quad . \ \boldsymbol{A}_{\mathbf{c}}(\boldsymbol{t})$$

where

- $\boldsymbol{A}_{\boldsymbol{ws}}^{*}(\boldsymbol{t}) = \text{dist}(\boldsymbol{s}, \boldsymbol{t})$ if $\boldsymbol{t}$ recognized by $\boldsymbol{A}_{\boldsymbol{s}}^{*}$

  $= \infty$ otherwise

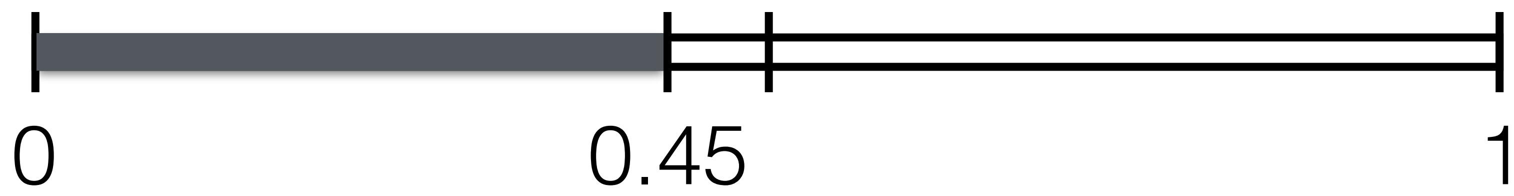
  (*ongoing*) construction follows the construction of $\boldsymbol{A}_{\boldsymbol{s}}^{*}$ from $\boldsymbol{A}_{\boldsymbol{s}}$
- $\boldsymbol{A}_{\mathbf{c}}$ is a weighted tree automaton characterizing the user preferred notations (independently of $\boldsymbol{s}$).
  It is learned online by quantization of previous segments.

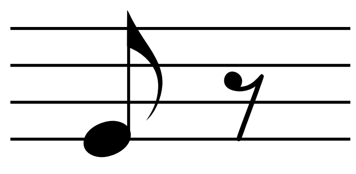
construction of $\boldsymbol{A}_{\mathbf{c}}$ (user's style)

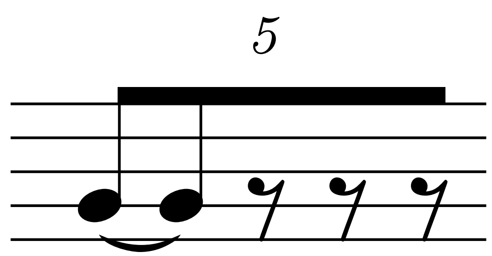
for each segment ***s***

1. enumerate the *k* first ***t*** candidates according to $\boldsymbol{A}_{\boldsymbol{ws}}^{*}(\boldsymbol{t}).\boldsymbol{A}_{\mathbf{c}}(\boldsymbol{t})$ (*k* is a user parameter)
2. print these ***t*** in traditional music notation
3. the user chooses his preferences,
   $\boldsymbol{A}_{\mathbf{c}}$ is updated accordingly
   using extension of RPNI algorithm to WTA
   (Adrien Maire)

example



proposition 1:  d = 0.05, low complexity

proposition 2: 

d = 0.05, higher complexity

- if the user chooses proposition 1 for this segment then *A* is left unchanged
- if the user prefers proposition 2 then *A* is updated: 5-uplets will get higher weights for next segments

enumeration

- there exists a WTA $\boldsymbol{A}$
  s.t. for all $\boldsymbol{t}$, $\boldsymbol{A}(\boldsymbol{t}) = \boldsymbol{A}_{\boldsymbol{ws}}{}^{*}(\boldsymbol{t}).\boldsymbol{A}_{\mathbf{c}}(\boldsymbol{t})$

- apply *k*-best algorithm to the
  weighted hypergraph presentation of $\boldsymbol{A}$

Huang, Chiang
Better k-best parsing
Parsing 2005

improves Knuth extension to hyper graphs
of Dijkstra's shortest path algorithm

conclusion

- approach for local rhythm quantization based on RT, involving
  1. term rewriting
  2. weighted tree automata

perspectives

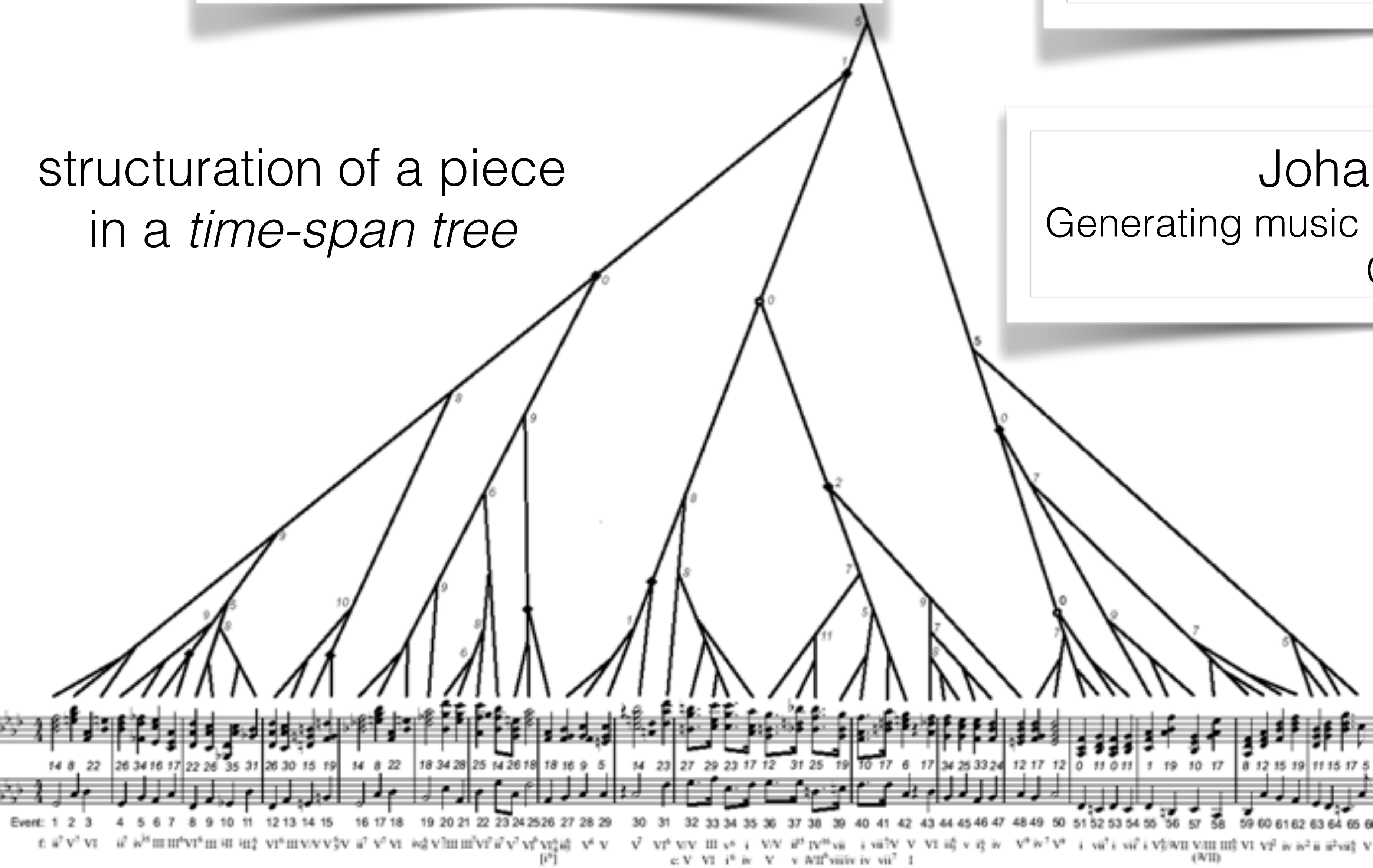
- joined tempo inference and local quantization
- application of rhythm trees techniques (automata, transformations) to other symbolic MIR problems

trees representations in music

Lerdahl and Jackendoff
A Generative theory of tonal music
MIT Press, 1983

Gilbert, Conklin
A probabilistic context- free grammar
for melodic reduction
Artificial Intelligence and Music, IJCAI, 2007

structuration of a piece
in a *time-span tree*



Johanna Högberg
Generating music by means of tree transducers
CIAA, 2005