Center for Research in Entertainment and Learning

Information Theoretic Creativity:

how to find optimal musical automata?

Shlomo Dubnov

UCSD

Talk plan

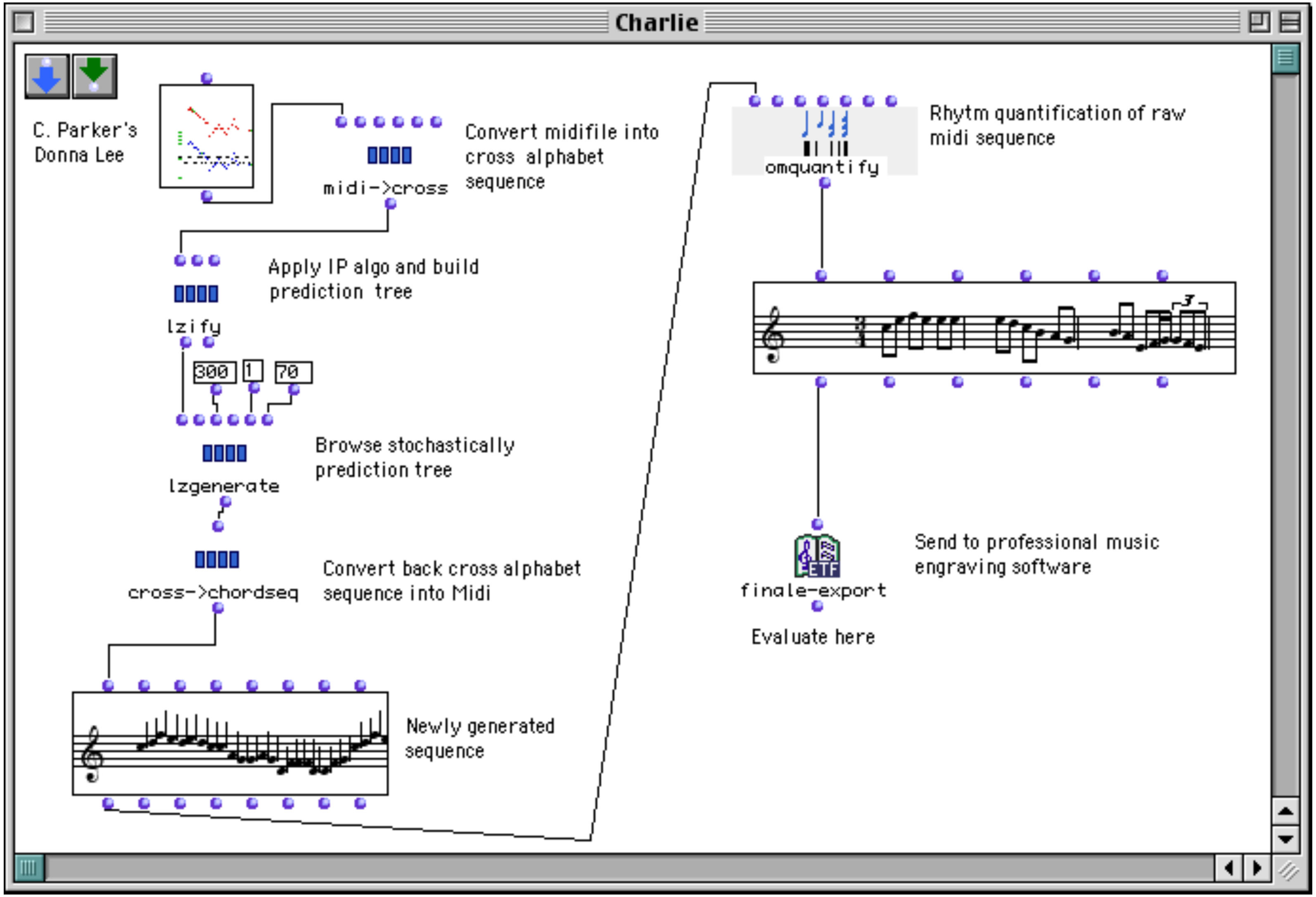
- Machine Improvisation
  - LZ, FO, AO
- Music Information Dynamics
  - PyOracle / VMO
- VMO Applications
- Dynamic Models of Creativity
- Questions / Discussion

Machine Improvisation

Existing Systems

- Mimi
- Continuator
- Omax
- PyOracle
- ImproteK

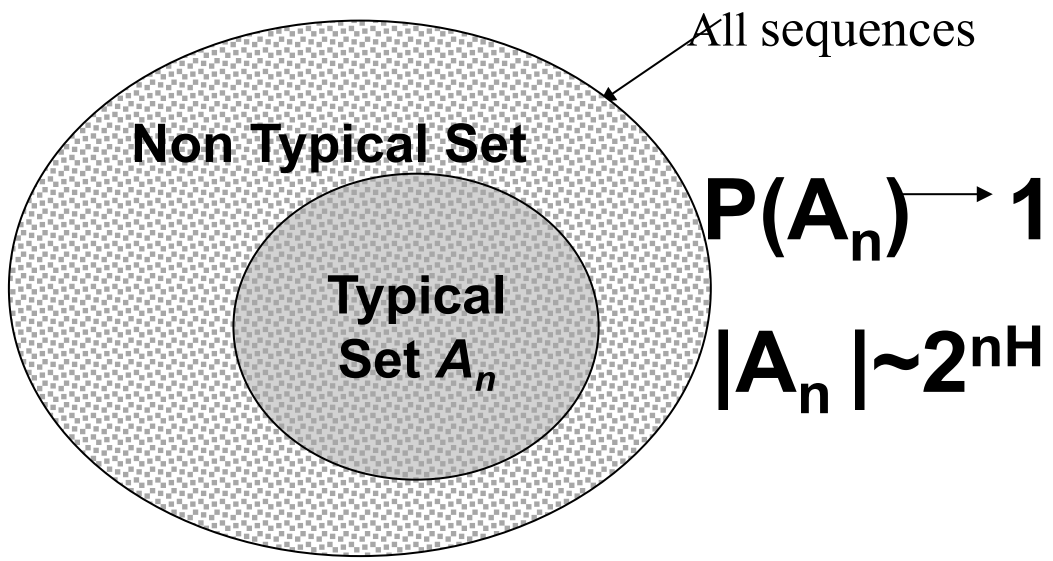
lzify



Charlie
C. Parker's
Donna Lee
midi->cross
Convert midifile into
cross alphabet
sequence
lzify
Apply IP algo and build
prediction tree
300
1
70
lzgenerate
Browse stochastically
prediction tree
cross->chordseq
Convert back cross alphabet
sequence into Midi
Newly generated
sequence
omquantify
Rhytm quantification of raw
midi sequence
finale-export
Send to professional music
engraving software
Evaluate here

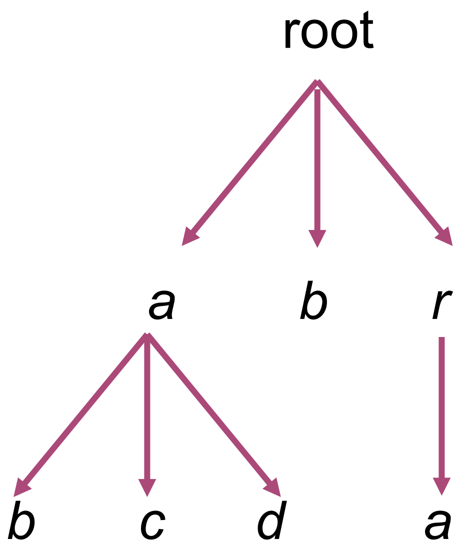
Style Learning Algorithms

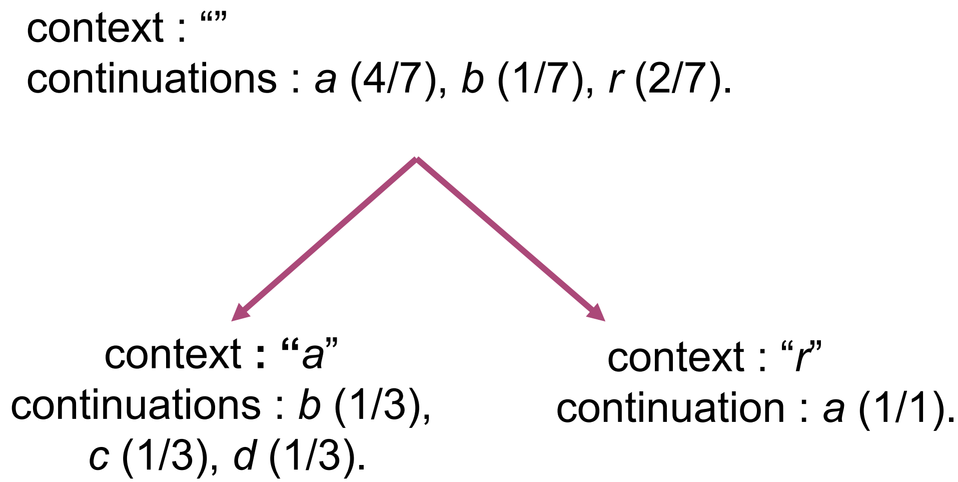
- Create Musical Generators from Examples that maintain *similarity* with the *style* of the original example



Suffix Automata using LZ

- Analysis of "*abracadabra*".





$$P(\text{generate "}abrac\text{"}) = P(a|\text{""})P(b|\underline{\mathbf{a}})P(r|ab)P(a|ab\underline{\mathbf{r}})P(c|abr\underline{\mathbf{a}}) = 4/7 \cdot 1/3 \cdot 2/7 \cdot 1 \cdot 1/3.$$

IPMotif

```python
def IPMotif(text):
    """Compute an associative dictionary (the motif dictionary)."""

    dictionary = {}
    motif = ""
    result = []
    for c in text:
        motif = motif + c
        if motif in dictionary:
            # Increase count for existing motif
#            print '%s in dictionary' % motif
            dictionary[motif] += 1
        else:
            # Add motif to the dictionary.
            dictionary[motif] = 1
            motif = ""
#            print 'add %s to dictionary' % motif

    return dictionary
```

{'a': 4, 'ac': 1, 'b': 1, 'ad': 1, 'r': 2, 'ra': 1, 'ab': 1}

IPContinuation

```
def IPContinuation(dict1):
    """Compute continuation dictionary from a motif dictionary"""

    dict2 = {}
    for Wk in dict1:
        counter = dict1[Wk]
        W = Wk[:-1]
        k = Wk[-1]
        if W in dict2:
            dict2[W].append((k,counter))
        else:
            dict2[W] = [(k,counter)]
    dict2 = Normalize(dict2)
    return dict2


def Normalize(dict2):
    """Turns the counters in every element of dict2 to probabilities

    for W in dict2:
        cnt = [tup[1] for tup in dict2[W]]
        ttl = sum(cnt)
        for k,tup in enumerate(dict2[W]):
            dict2[W][k] = (tup[0],float(tup[1])/ttl)
    return dict2
```

{'': [('a', 0.57), ('b', 0.14), ('r', 0.28)], 'a': [('c', 0.33), ('d', 0.33), ('b', 0.33)], 'r': [('a', 1.0)]}

Markov

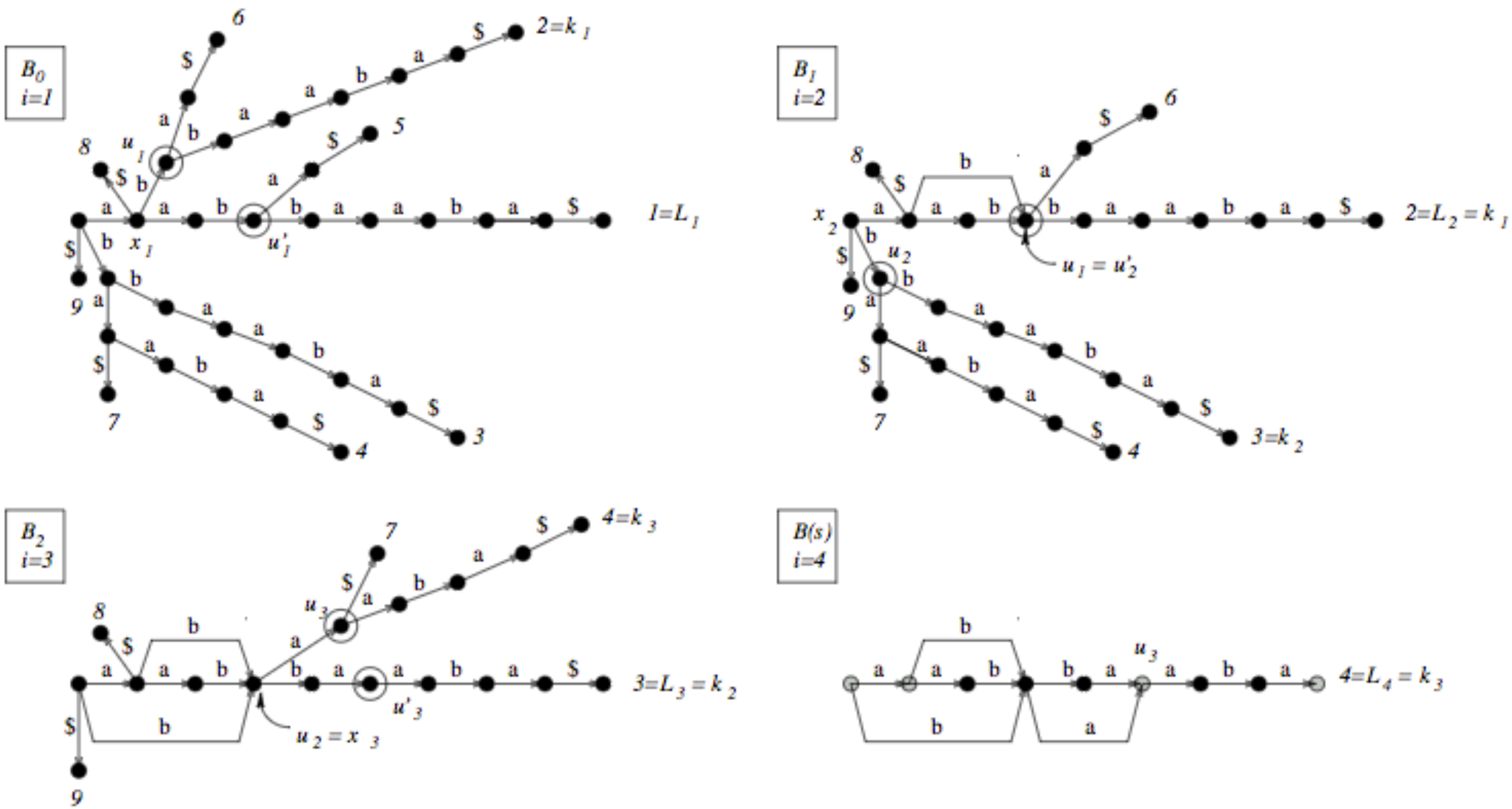
```
def Markov(text,order=0):
    """Compute a Markov models (fixed length motif dictionary)."""

    dict3 = {}
    for i in range(len(text)-order):
        W = text[i:i+order]
        k  = text[i+order]
        if W in dict3:
            if k in list(zip(*dict3[W])[0]):
                dict3[W][k] += 1
            else:
                dict3[W][k] = 1
        else:
            dict3[W] = {k:1}

    for x in dict3:
        dict3[x] = dict3[x].items()
    dict3 = Normalize(dict3)
    return dict3
```

{'a': [('c', 0.25), ('b', 0.5), ('d', 0.25)], 'r': [('a', 1.0)], 'b': [('r', 1.0)], 'c': [('a', 1.0)], 'd': [('a', 1.0)]}

Suffix Automata

- Suffix Trees (ST) are used for string searching
- Probabilistic Extensions (PST, PPM) are useful for inference
- LZ uses a ST to find the (start,len) pair
  - S = ababababababababababababababababab
  - ab**(1, 2)(1, 4)(1, 8)(1, 16).**
- ST can be “bent” into Factor Oracle

Bending Suffix Tree



IPContinuation continued

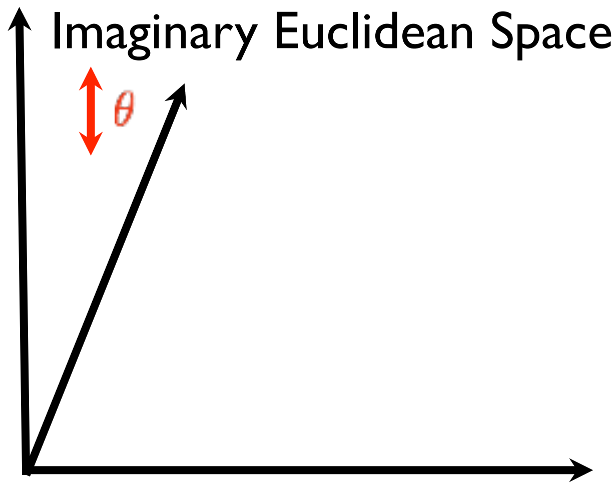
- In LZify we are traversing the LZ tree over and over with longest suffix
- FO algorithm during construction creates suffix links to longest repeated suffix (LRS)
- Suffix links and reverse suffix links constitute all points in a sequence that share a common suffix
- We can use these suffixes to “remix” the signal, i.e. “improvise”

FO versus Audio Oracle Learning

REALTIME Scheduler

Time = 0

No event!

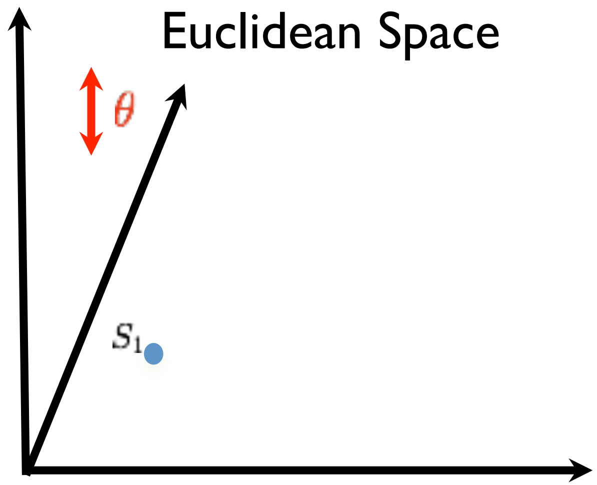


Sequential AO Learning:

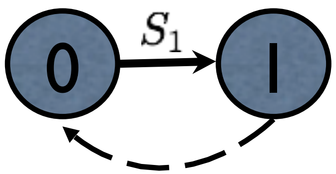
FO versus Audio Oracle Learning

Create suffix link 1

REALTIME Scheduler
Time = 1
Arrival of S_1



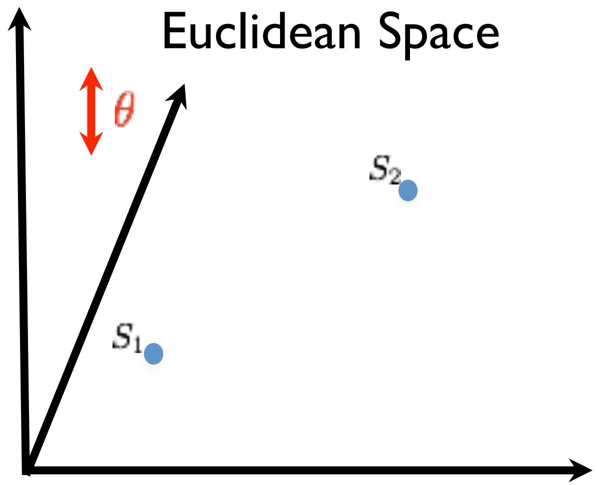
Sequential AO Learning:



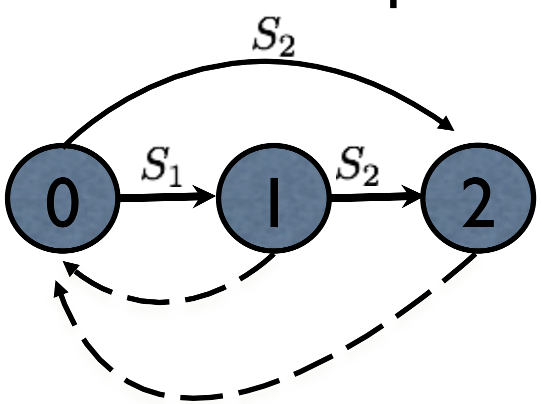
FO versus Audio Oracle Learning

Follow suffix 1
Create suffix 2
Create forward link 2

REALTIME Scheduler
Time = 2
Arrival of S_2



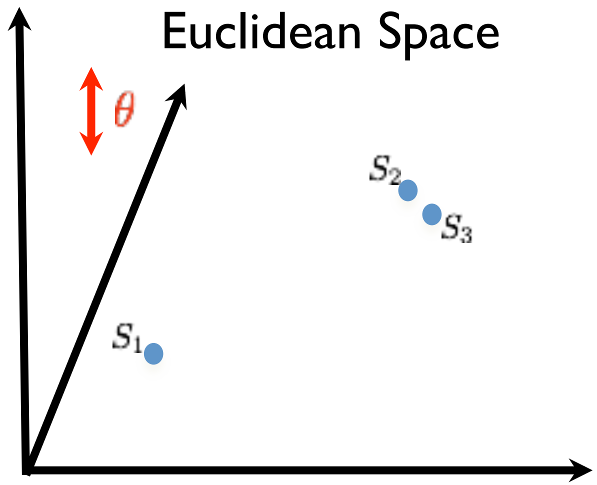
Sequential AO Learning:



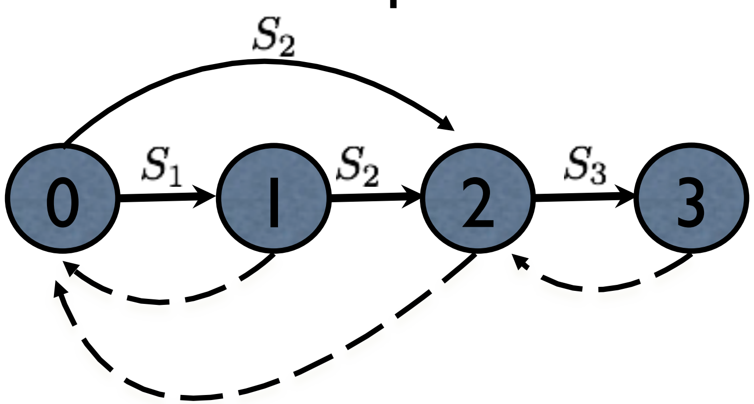
FO versus Audio Oracle Learning

Follow suffix 2
Follow forward link 2
Create suffix 3

REALTIME Scheduler
Time = 3
Arrival of S_3



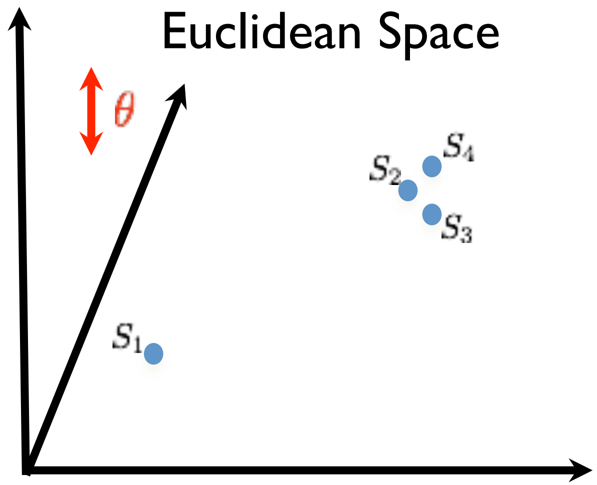
Sequential AO Learning:



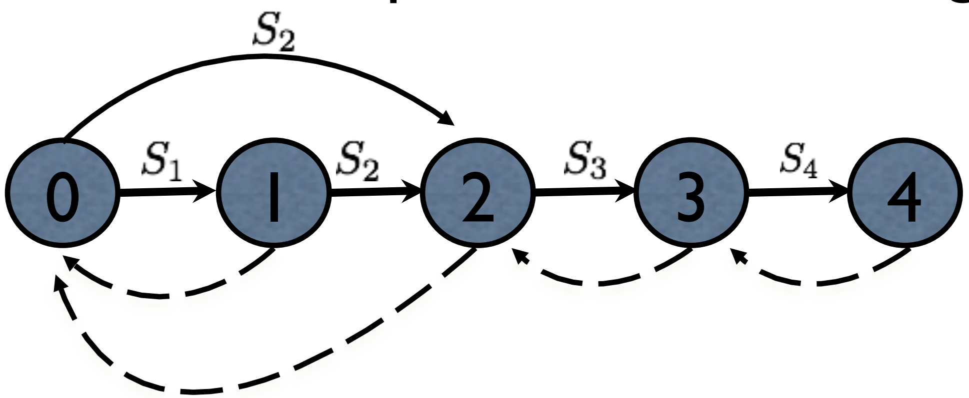
FO versus Audio Oracle Learning

Follow suffix 3
Follow forward link 3
Create suffix 4

REALTIME Scheduler
Time = 4
Arrival of S_4

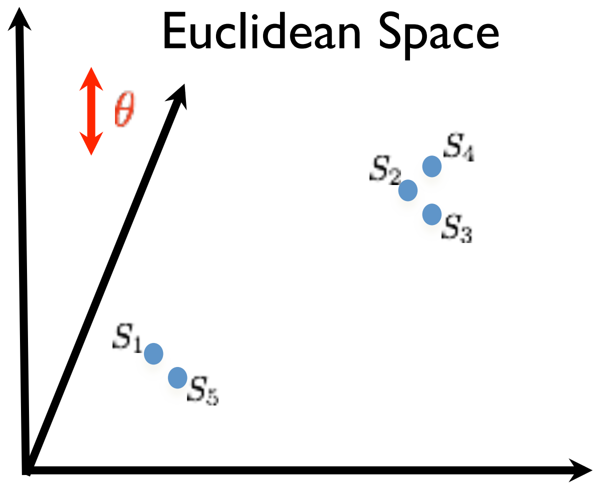


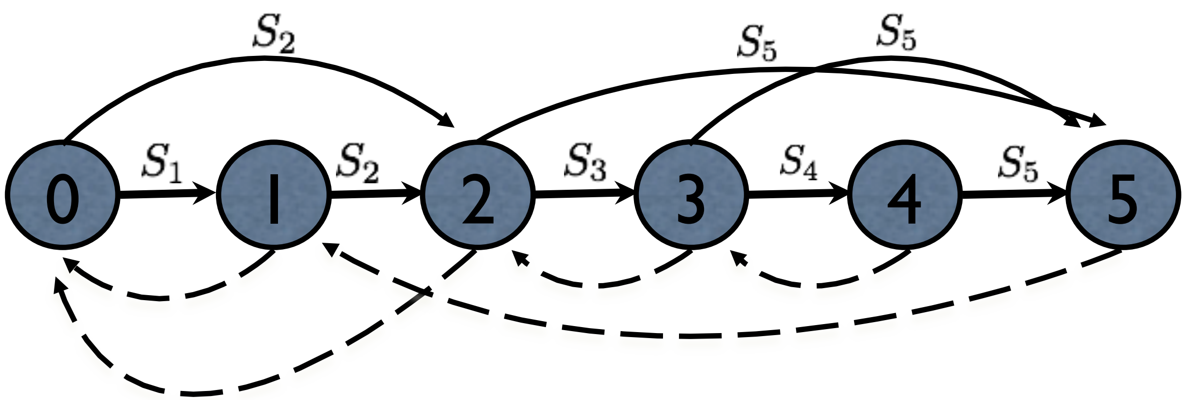
Sequential AO Learning:



FO versus Audio Oracle Learning

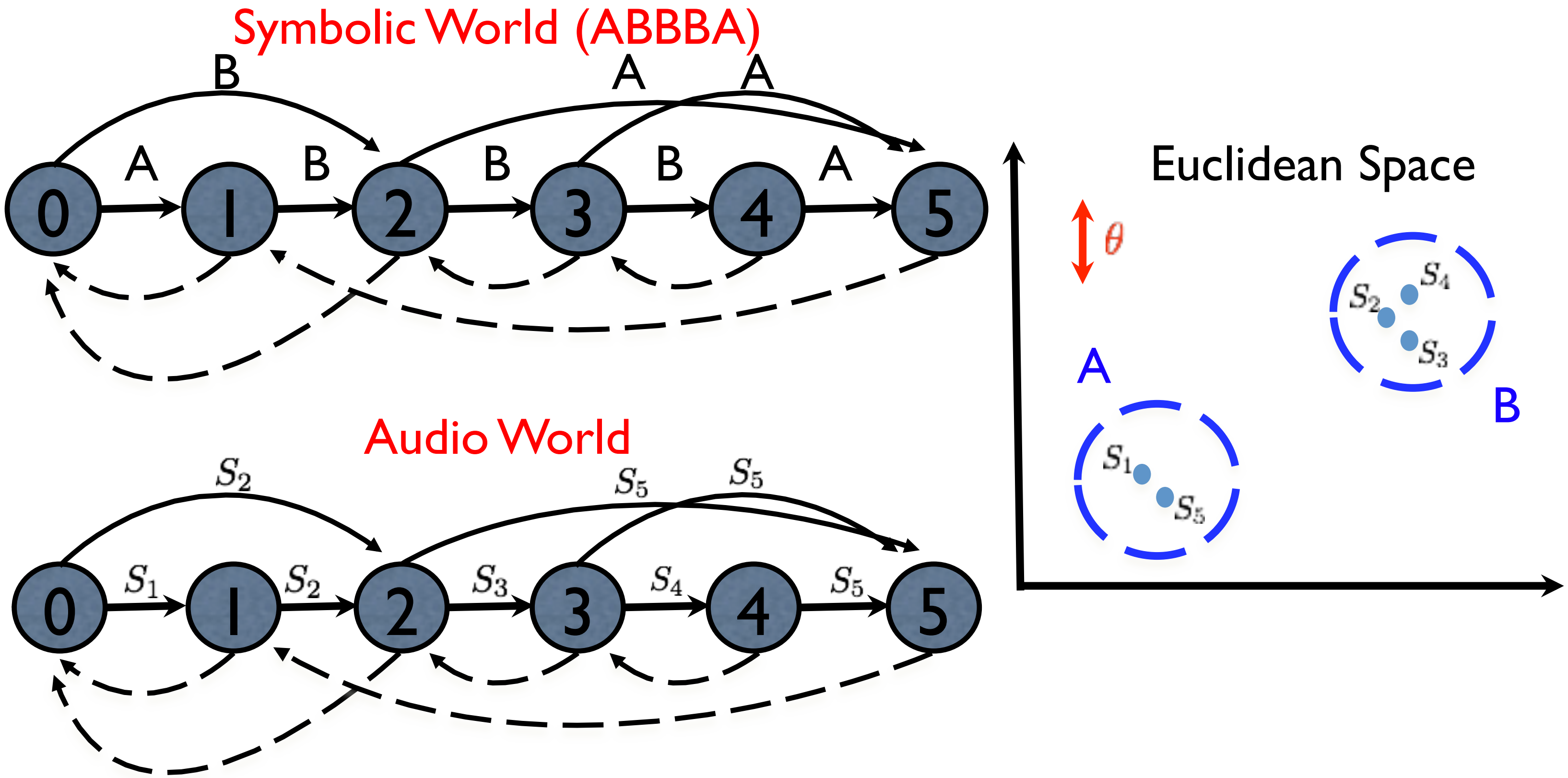
Follow suffix 4
Create forward link 5
Follow suffix 3
Create forward link 5
Follow suffix 2
Follow forward link 1
Create suffix 5





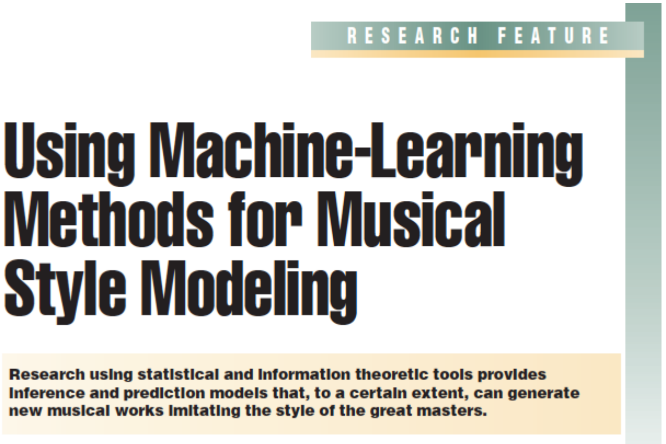
Implicit Quantization

AO requires a threshold for symbolization / quantization



INTERNATIONAL COMPUTER MUSIC ASSOCIATION

Guessing the Composer's Mind: Applying Universal Prediction to Musical Style
Assayag, Gérard; Dubnov, Shlomo; Delerue, Olivier, ICMC 1999


RESEARCH FEATURE

Using Machine-Learning Methods for Musical Style Modeling

Research using statistical and information theoretic tools provides inference and prediction models that, to a certain extent, can generate new musical works imitating the style of the great masters.

Shlomo Dubnov, Gerard Assayag, Olivier Lartillot, Gill Bejerano, IEEE Computer, 2003, Issue 10

Using Factor Oracle for Machine Improvisation, G. Assayag and S. Dubnov, Soft Computing 8 (9), 2004

Problems

- What if the radius of the balls is unknown?
  - Using FO on an Audio Signal requires symbolization / quantization prior to FO
- Audio Signal Can be analyzed in terms of multiple features.
  - Which feature to choose?

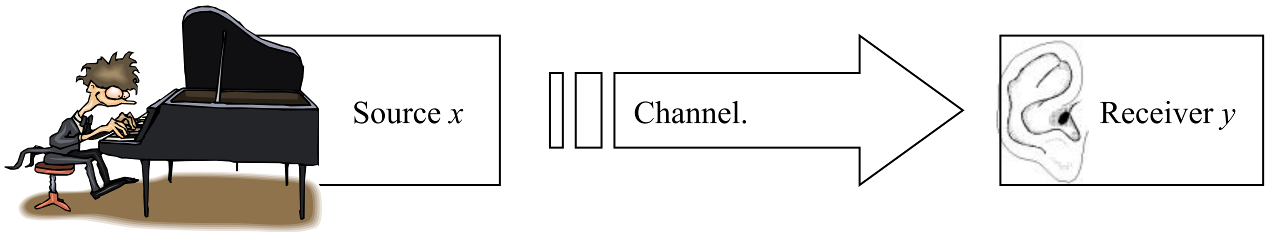
Information Dynamics criteria for quality of FO model of a quantized signal:

– one that compresses the best

Musical Information Dynamics

Information Dynamics

- Listener anticipates the continuations of music
- Such predictions reduce his uncertainty about the future

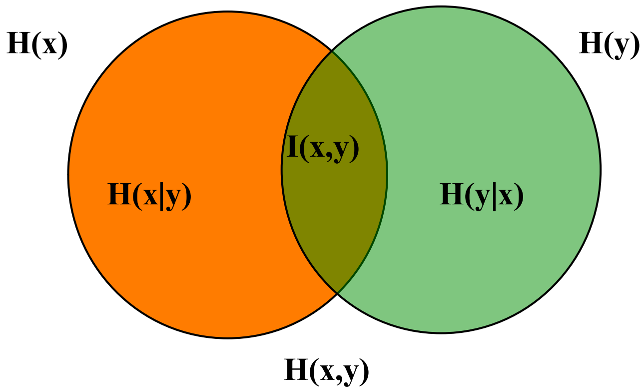


Validation and violation of expectations is a technique commonly used by composers

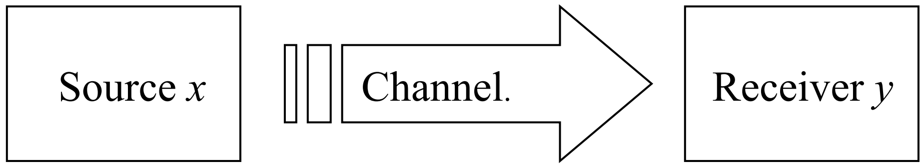
The information paradox:
Discover more by listening more….
(you gain by learning, not get bored!)

Information Rate

- Entropy & Information



- Communication Channel



$$I(x,y) = H(x) - H(x \mid y)$$

y – past experience, what you heard so far

x – new material

$H(x)$ – uncertainty about x

$H(x|y)$ – uncertainty about x when we know already y

If y is long enough, H(x|y) becomes ***Entropy Rate***

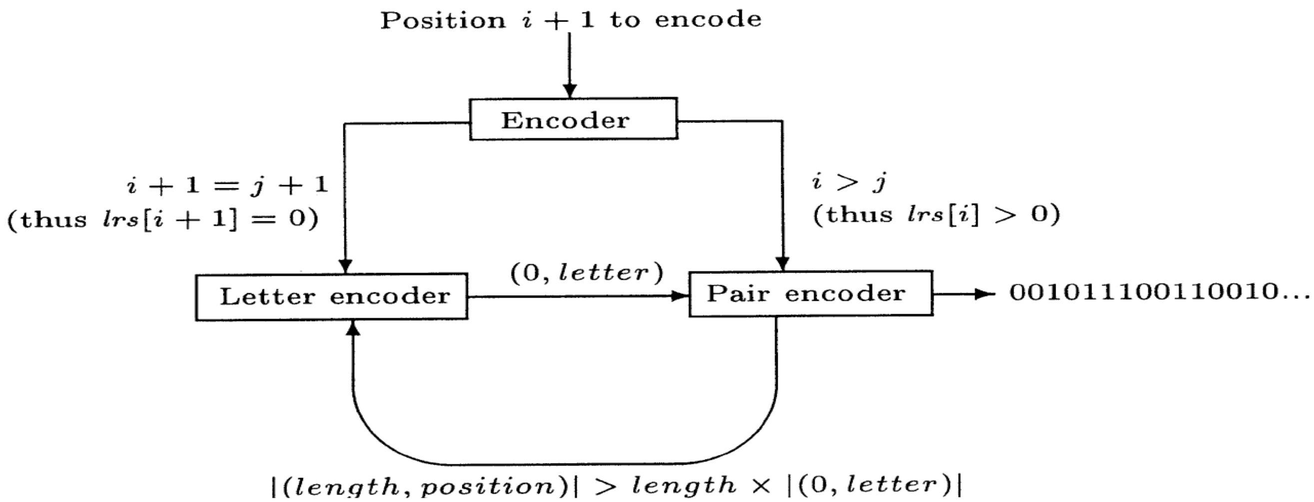
$I(x,y)$ – how much the past tells us about the future

Use of IR in VMO

- Replace Entropy H with Coding Length C

IR(letter) = C(letter) – C(letter | past sequence)

= | Single letter encoder | - | Block encoder | / Block size



Compror encoding scheme

Compror Explained:

- Source: *aabbabbabbab*
- Encoding: *a(1,1)b(1,3)(8,2)*
- Decode recursively

Recopy position

Recopy length

Variable Markov Oracle

Multimedia Research and Applications

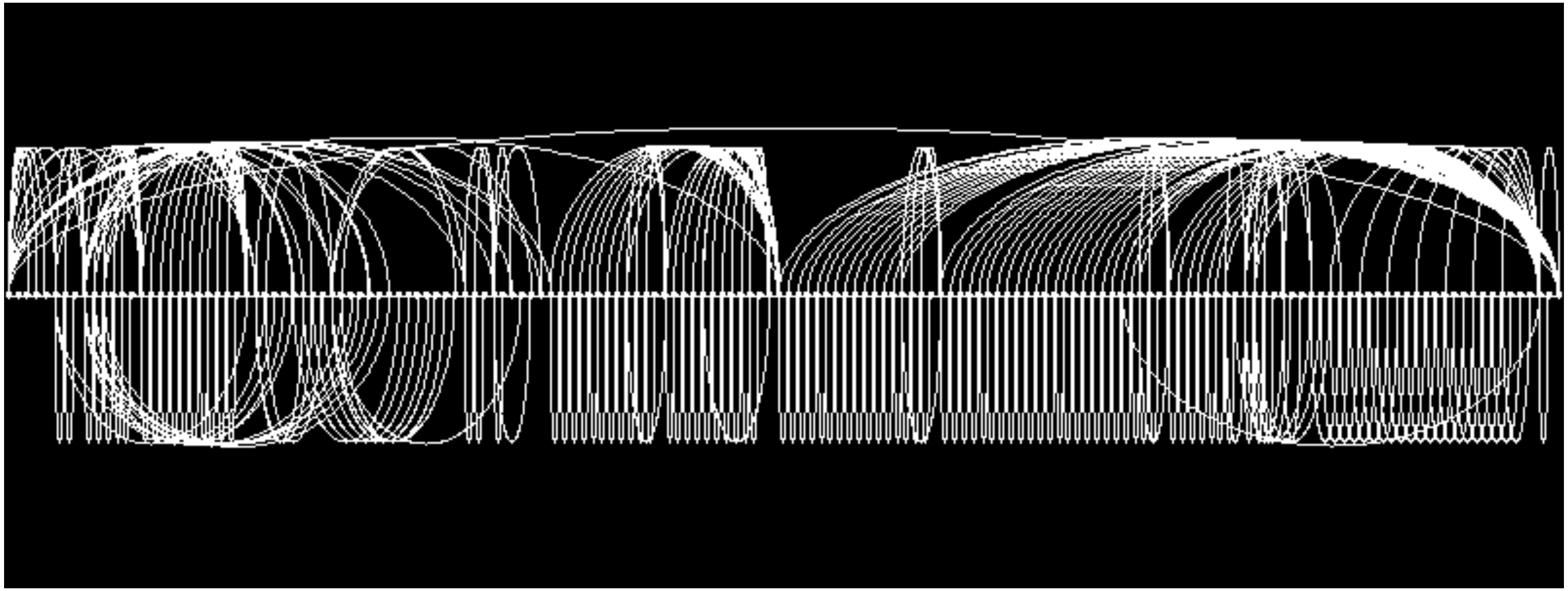
The Variable Markov Oracle: Algorithms for Human Gesture Applications

Cheng-i Wang and Shlomo Dubnov
University of California, San Diego

October–December 2015

PyOracle finds the most informative structure of time series data

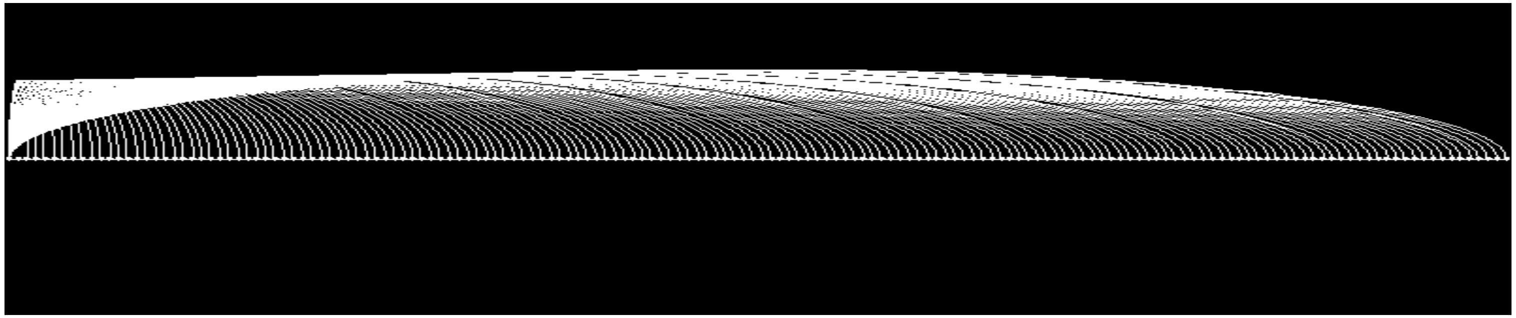
Optimal Threshold



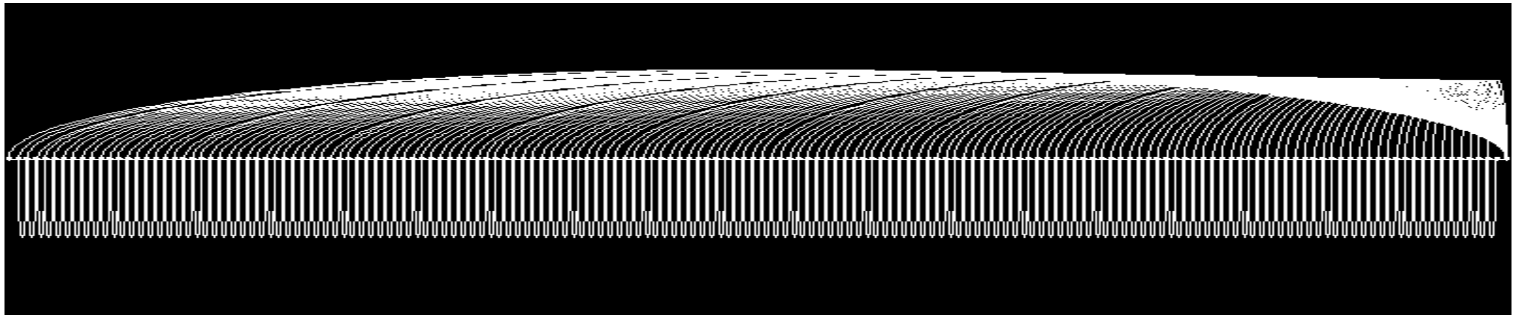
S. Prokofiev, Visions Fugitives No.4

Structure depends on similarity sensitivity

Extremely Low Threshold



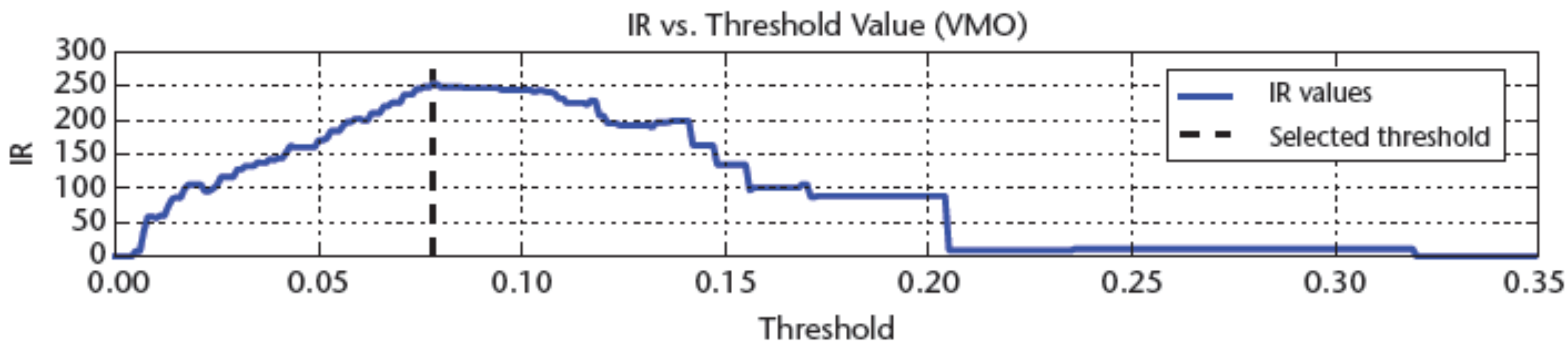
Extremely High Threshold



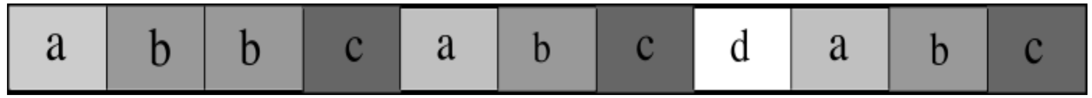
Use of IR in VMO

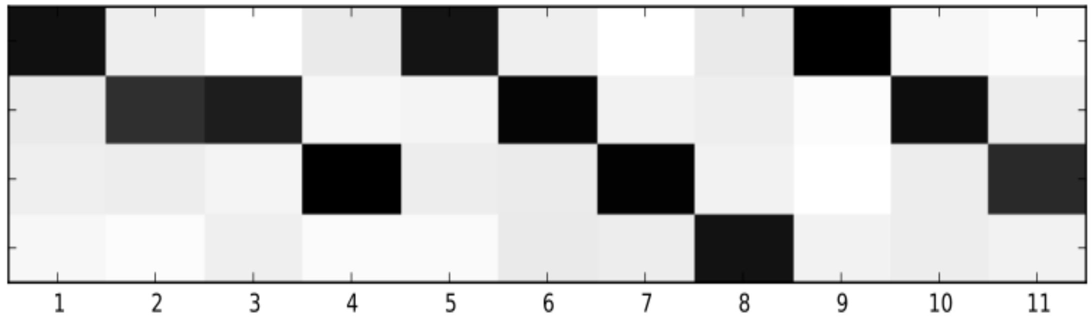
- Instead of Entropy use Compression and count number of bits using Compror

  IR = log (size of alphabet) – number of bits used in Compror encoding / size of the encoding block
- Search over all thresholds to choose a model that has highest IR

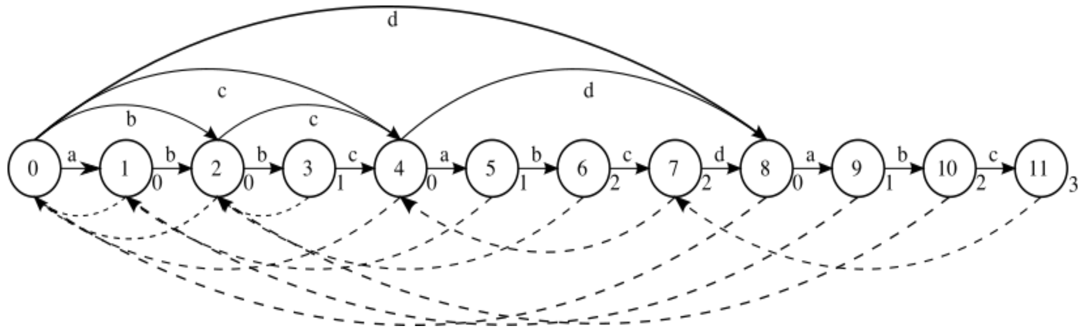


VMO vs Markov

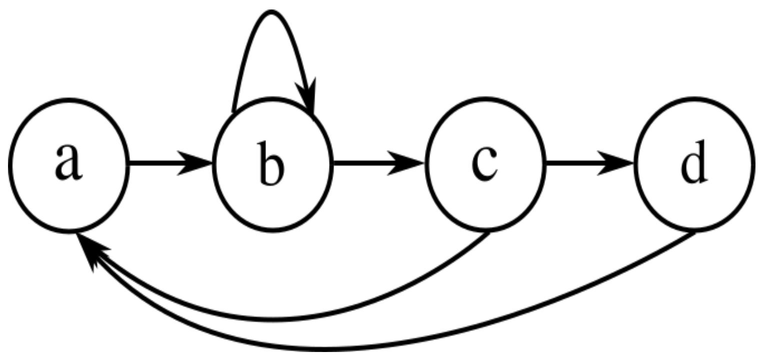




{c: abc->d, bc->{d,a}}
{b: ab->c, ab->c, b->c}
{a: a->b, a->b}
{d: d->a}



[1,5,9],[2,3,6,10],[4,7,11],[8]



Memory >= 4: abcd
Memory >= 3: abcd abc
Memory >= 2: abcd abc
Memory >= 1: abcd abc ab b
No memory: b a c d

Statistical Interpretation

HMM

- Observed **R**
- Estimated P, $a_{m,m'}$, $V_{n,m}$

$$V_{1,m} = P(\mathbf{R}[1]|m) \cdot \pi_m, \quad \text{and}$$
$$V_{n,m} = \max_{m'} \left(P(\mathbf{R}[1]|m) \cdot a_{m,m'} \cdot V_{n-1,m'} \right)$$

$P(\mathbf{R}[n]|\mathrm{m})$ – *emission*

$a_{m.m'}$ - *transition*

VMO

- VMO state transitions counted by forward-links

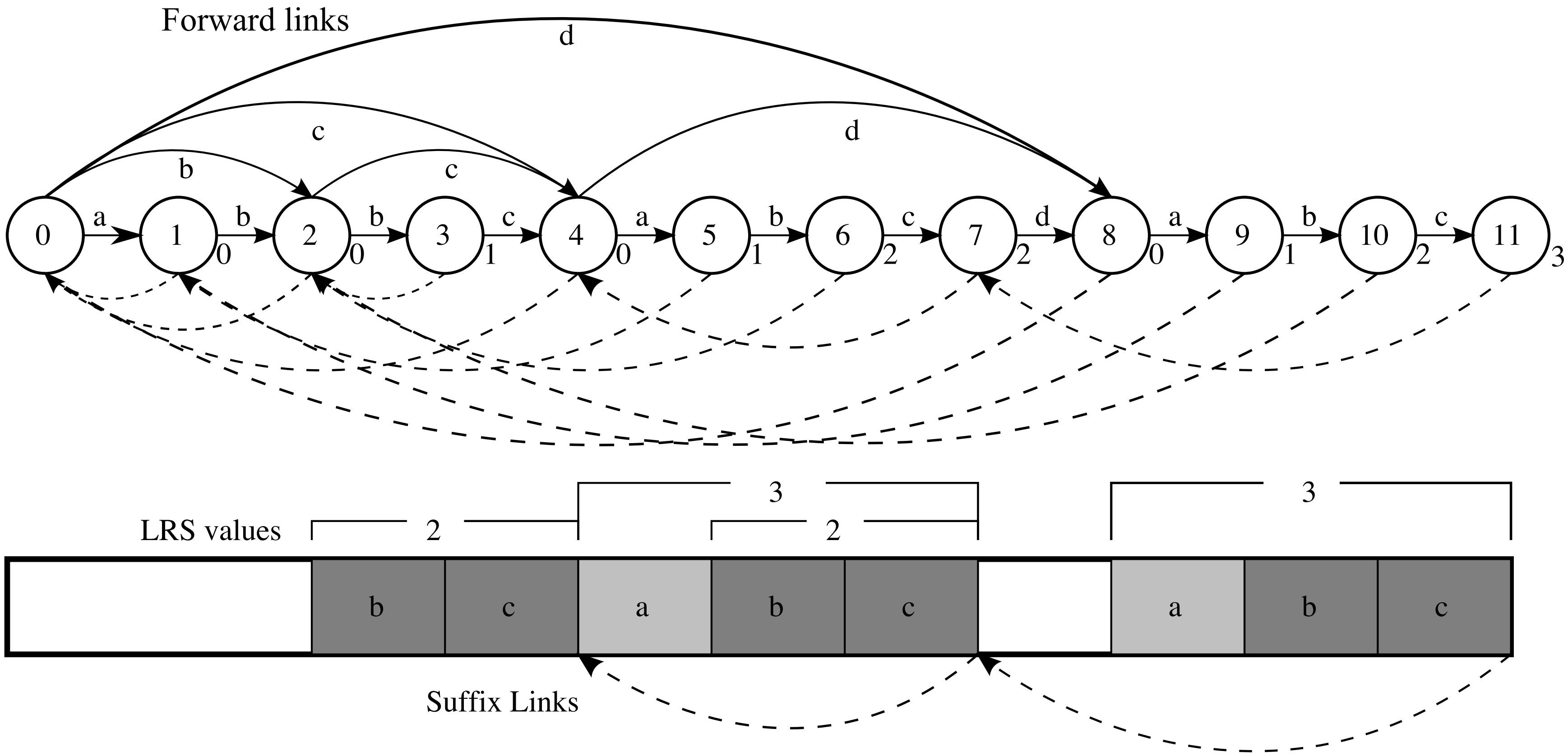
$$a_{ij} = \frac{1\left(\exists \delta(t,j), t \in \sigma_i\right)|\sigma_j|}{\sum_{j'=1}^{M} 1\left(\exists \delta(t,j'), t \in \sigma_i\right)|\sigma_{j'}|},$$

- Emission according to observed distance

$$P(\mathbf{R}[n]|m) \propto \exp\left(\frac{-d(\mathbf{R}[n]|m)}{\alpha}\right),$$

$$d(\mathbf{R}[n], m) \triangleq \min_{t' \in \sigma_m} \|\mathbf{R}[n] - \mathbf{O}[t']\|.$$

Pattern Discovery with VMO

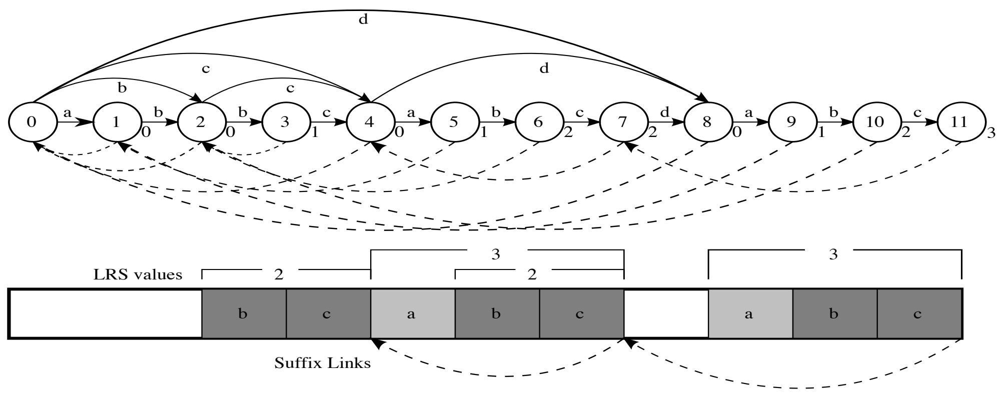


Pattern Discovery with VMO

MIREX Challenge: Discovery of Repeated Themes & Sections

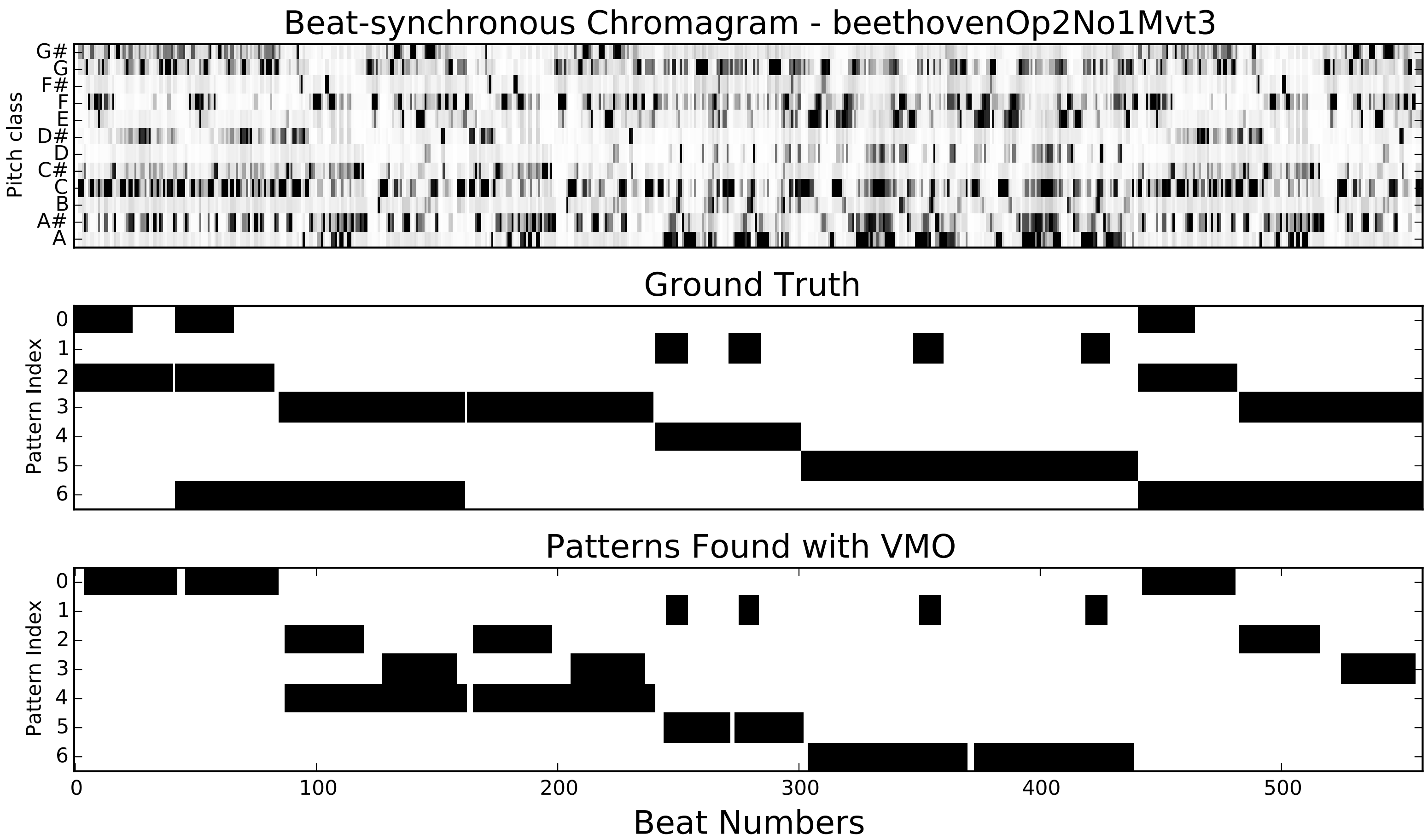
Develop algorithms that take a single piece of music as input, and output a list of patterns repeated within that piece. Also known as *intra-opus discovery* [Conklin & Anagnostopoulou, 2001].

- **Our Algorithm**:
  - Choose Oracle with highest Information Rate
  - look through *lrs* from t = 1, 2, …., T to find discontinuity in *lrs* values as an indication of an "ending" of a pattern
  - Follow suffix links to both directions to find all related patterns



$$lrs[t] - lrs[t-1] \neq 1,$$

Pattern Discovery with VMO

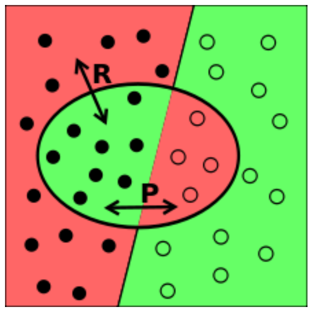


Pattern Discovery with VMO

- Results

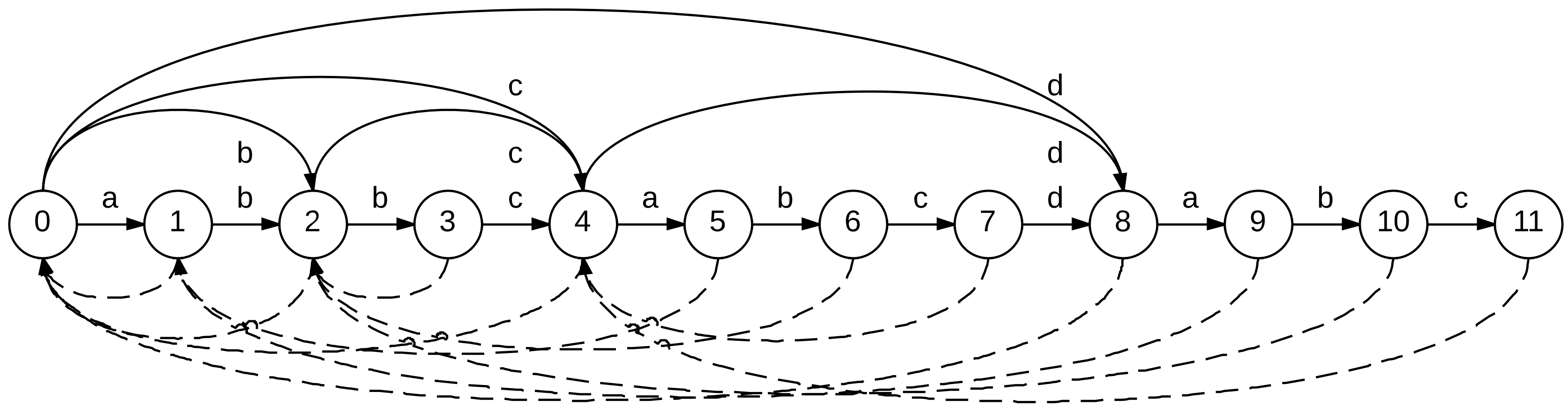
Algorithm	F_{est}	$F_{o(.5)}$	$F_{o(.75)}$	F_3	Time (s)
Proposed	**53.75**	**68.84**	**70.47**	**48.36**	**96**
[Nieto and Farbood, 2014]	49.8	38.73	31.79	32.01	454
[Collins et al., 2014]	23.94	56.87	—	—	—
[Nieto and Farbood, 2013]	41.43	23.18	24.87	28.23	196

Recall = TP/(TP + FN)
Precision = TP/(TP + FP)
1/F = (1/R + 1/P)/2
F = 2TP/(2TP + FP + FN)



Query matching

- Query: “abcd”
- Target: “abbcabcdabc
- Result: location “8” at the end of Query

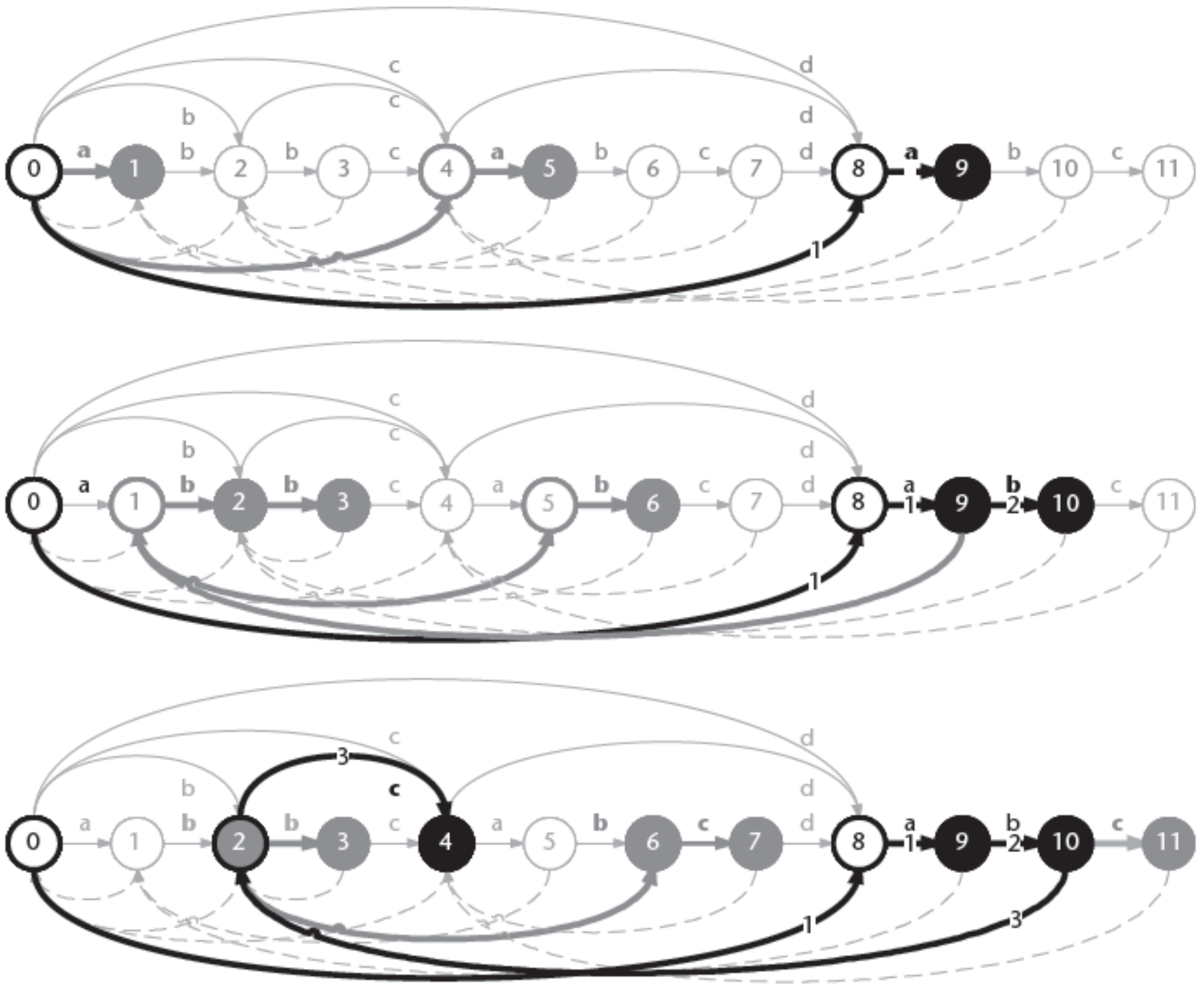


Query Matching Algorithm

Require: Target signal in *VMO*, $Oracle(\boldsymbol{Q} = q_1, q_2, \ldots, q_T, \boldsymbol{O} = \boldsymbol{O}[1], \boldsymbol{O}[2], \ldots, \boldsymbol{O}[T])$ and query time series $\boldsymbol{R} = \boldsymbol{R}[1], \boldsymbol{R}[2], \ldots, \boldsymbol{R}[N]$

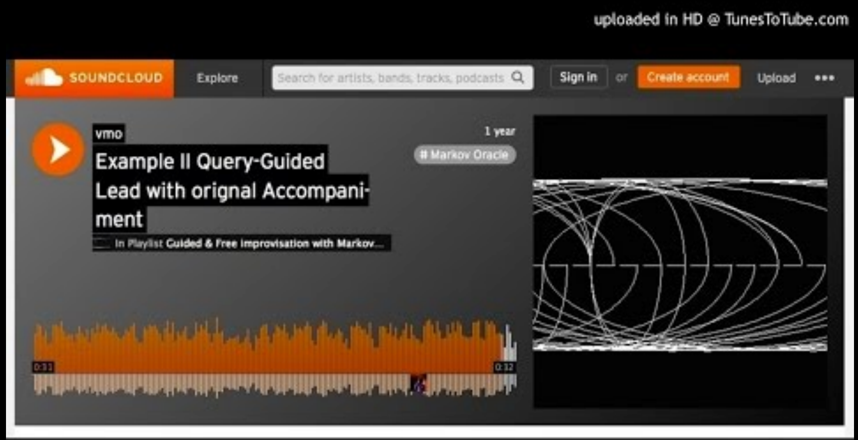
1. Get the number of clusters, $M \leftarrow |\boldsymbol{\Sigma}|$
2. Initialize cost vector $C \in \mathbb{R}^M$ and path matrix $P \in \mathbb{R}^{M \times N}$.
3. **for** $m = 1 : M$ **do**
4. $\quad P_{m,1} \leftarrow$ Find the state, t, in the mth list from $\boldsymbol{\Sigma}$ with the least distance, $d_{m,1}$, to $\boldsymbol{R}[1]$
5. $\quad C_m \leftarrow d_{m,1}$
6. **end for**
7. **for** $n = 2 : N$ **do**
8. $\quad$ **for** $m = 1 : M$ **do**
9. $\quad\quad P_{m,n} \leftarrow$ Find the state, t, in lists with labels corresponding to forward links from state $P_{m,n-1}$ with the least distance, $d_{m,n}$ to $\boldsymbol{R}[n]$
10. $\quad\quad C_m \mathrel{+}= d_{m,n}$
11. $\quad$ **end for**
12. **end for**
13. **return** $P[\mathrm{argmin}(C)], \min(C)$

Approximate Query matching using VMO

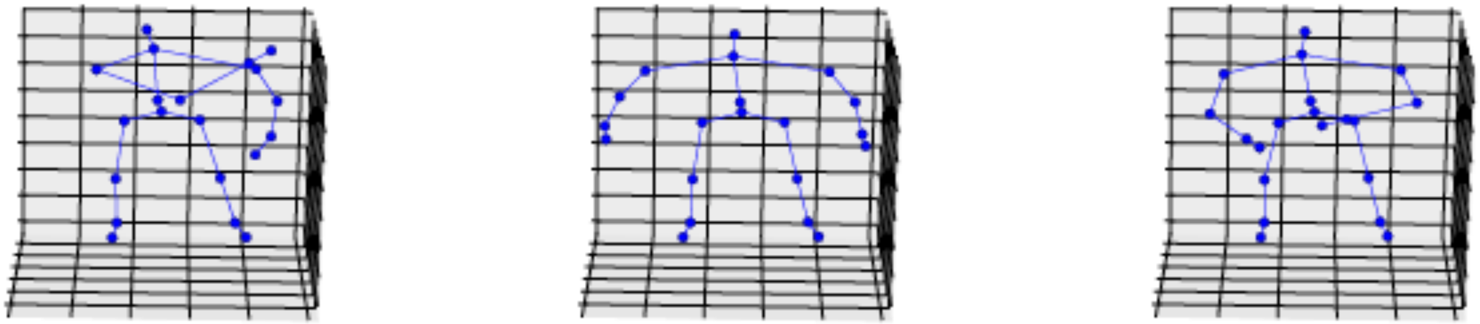


Guided Improvisation

- Automatic Accompaniment
- Automatic Solo Generation



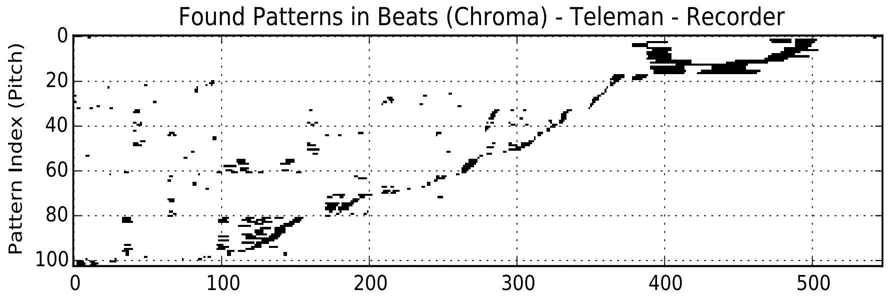
Gesture Recognition

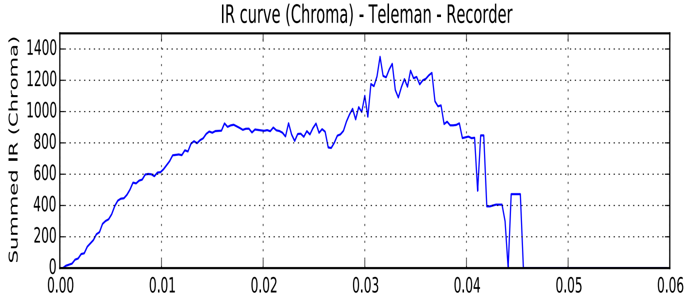


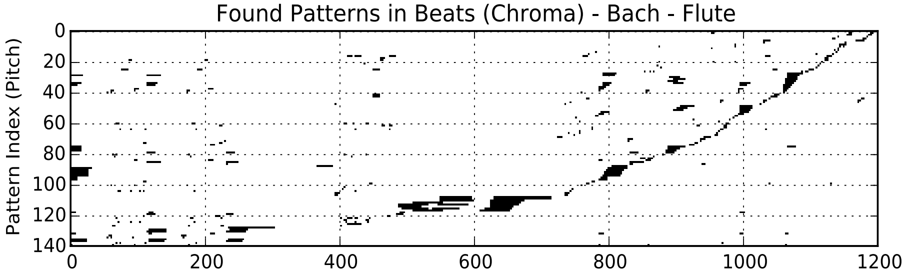
Gesture	Precision(%)
Crouch or hide	99.9
Shoot with a pistol	90.6
Throw an object	84.6
Change weapon	98.4
Kick to attack	92.7
Put on a goggle	92.2
Collection	**93.7**
Avg.±std.	**93.1±5**
State of the Art[16]	**93.6**

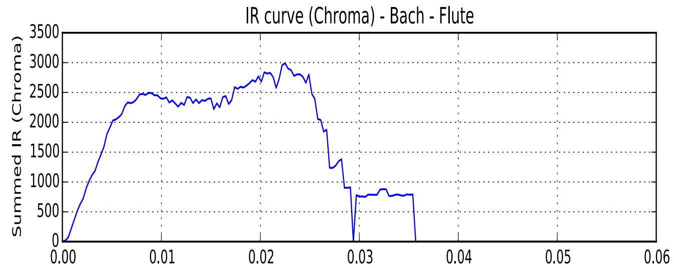
Accepted for IEEE-ISM2015

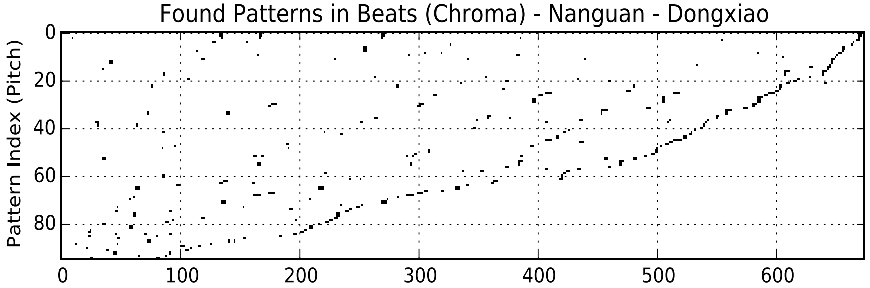
Cultural Acoustic Sensibility

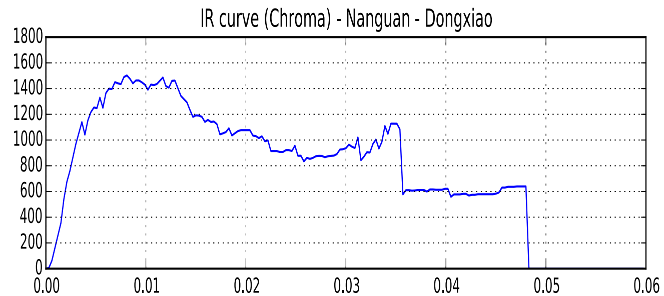


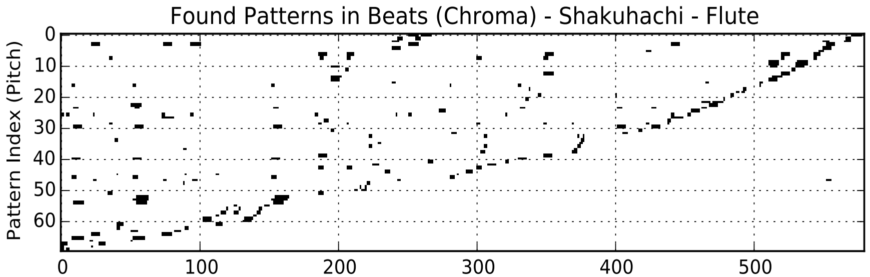


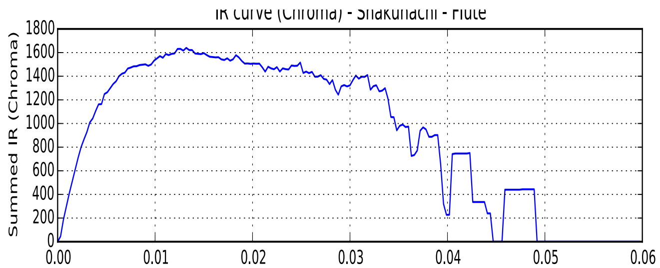




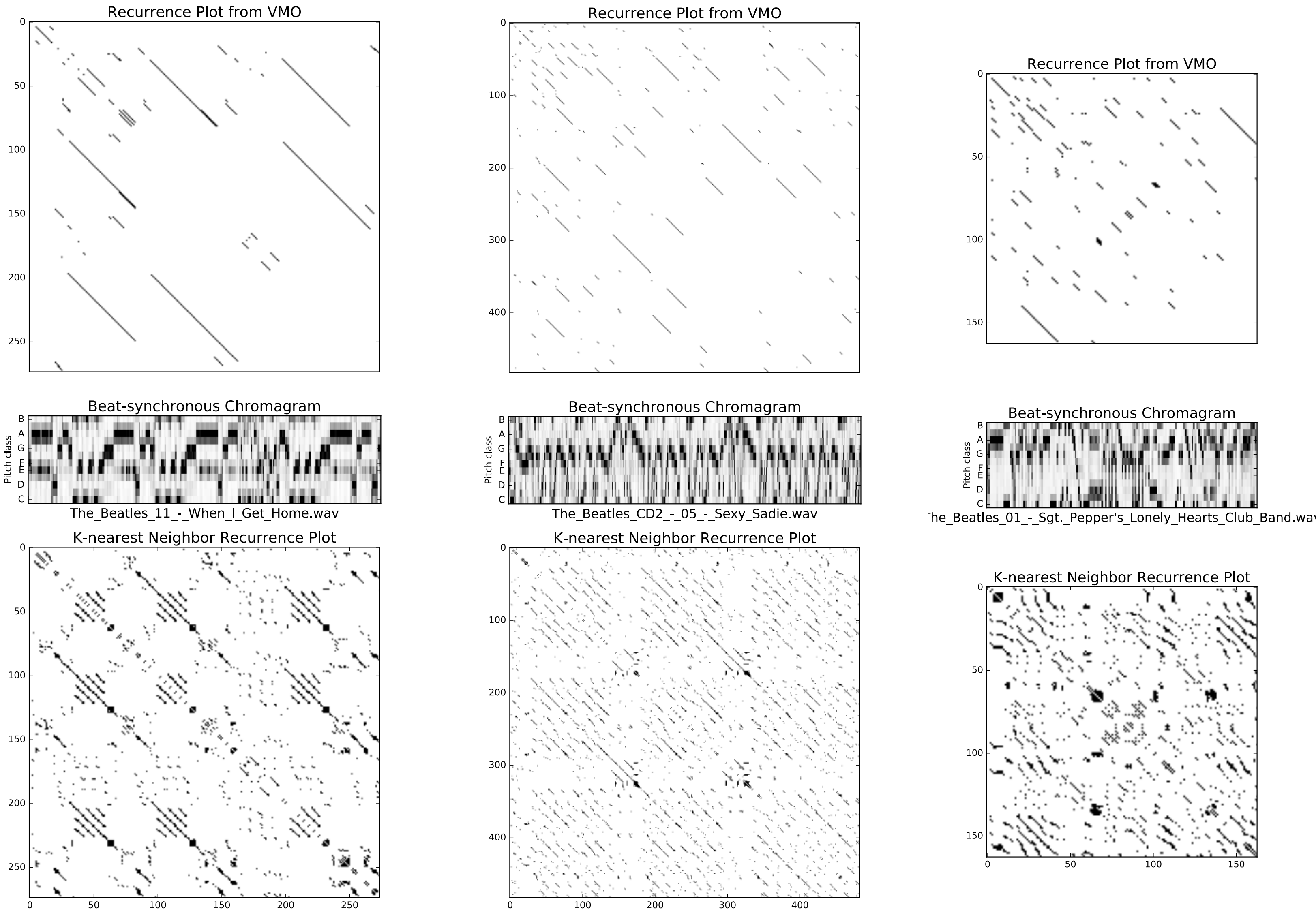




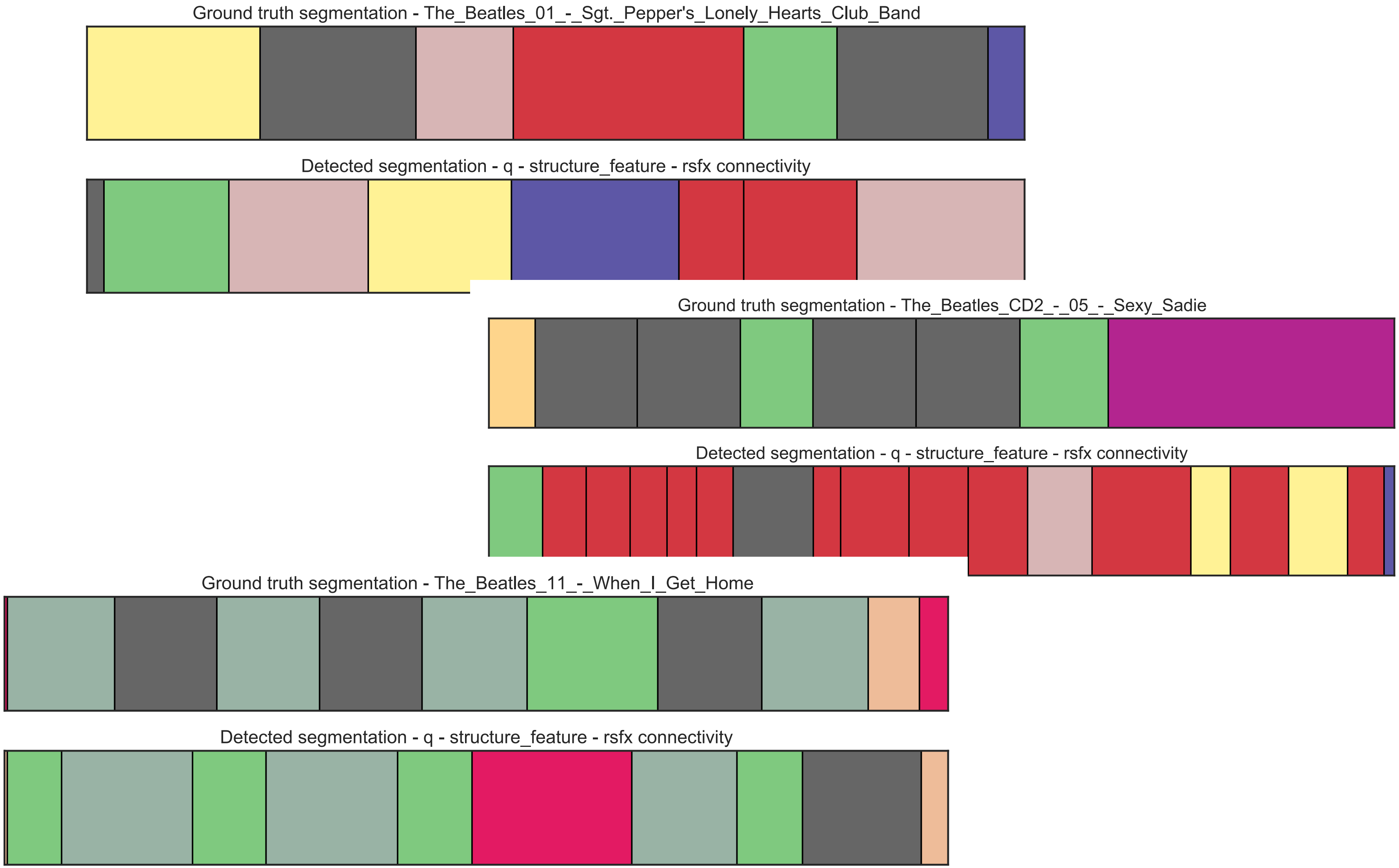




Self Similarity



Segmentation



Structured Improvisation

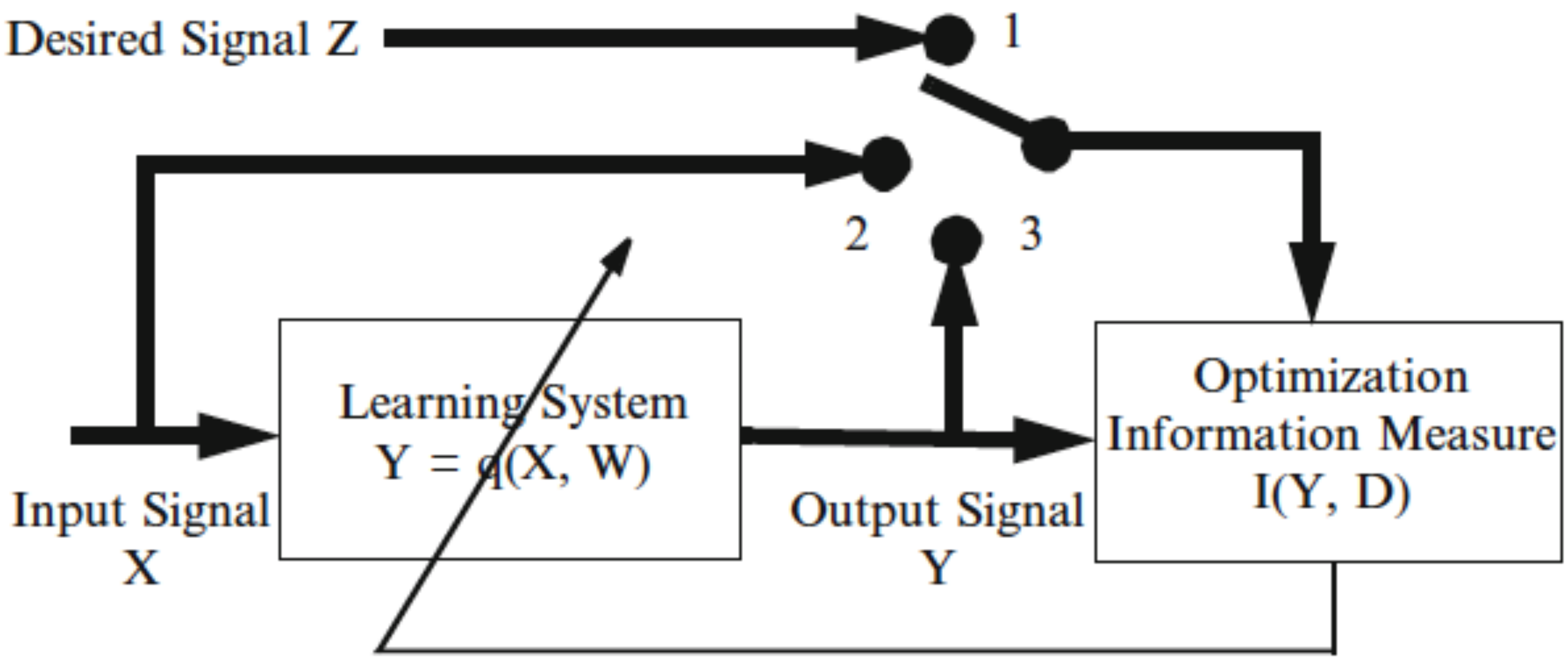
- One of the challenges in using Style Learning algorithms is a problem of controlling the output
- FO is a finite state machine and can be "controlled" using tools of DES (Donze)
- Path corresponding to input can be found using greedy search (Wang)
- If future is known for given scenario, prefix search can be used to optimize Value to Go (Nikka)
- VMO can be translated to Kripke structure and checked for desired path (Bazin)
- CP used on LZ trees by Pachet

Problems / Challenges

- Specification of control / improvisation guides might have different alphabet and different time scales
- Need to find intermediate representation that link between two music materials, such as human (h) and machine (m)
- Score (s) can be such intermediate representation – a categorization scheme with optimal tradeoff between accuracy (quantization) and complexity (information dynamics)

Musical Bottleneck

- Find score *s* so that $I(m{:}s) - \alpha I(h{:}s)$ is optimized



Two criteria for categorizing

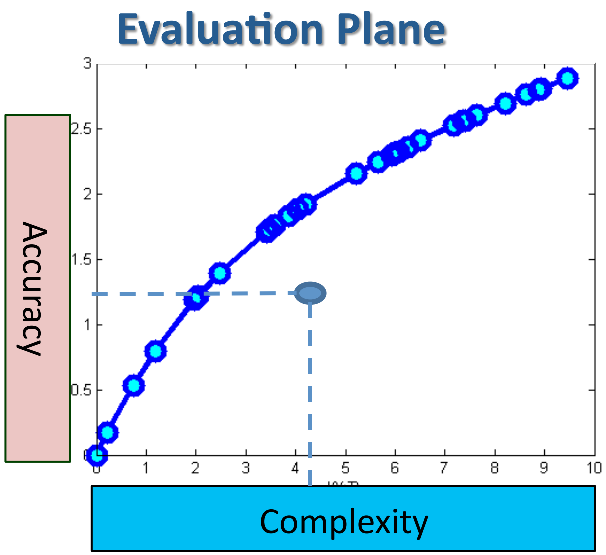
$$I(F;Y) = \sum_{f,y} p(F=f, Y=y) \log_2 \left(\frac{p(F=f, Y=y)}{p(F=f)p(Y=y)} \right)$$

- Two criteria for a categorization scheme:

Accuracy (How well it describes the musical surface)

Complexity (Its efficiency according to Occam's razor).

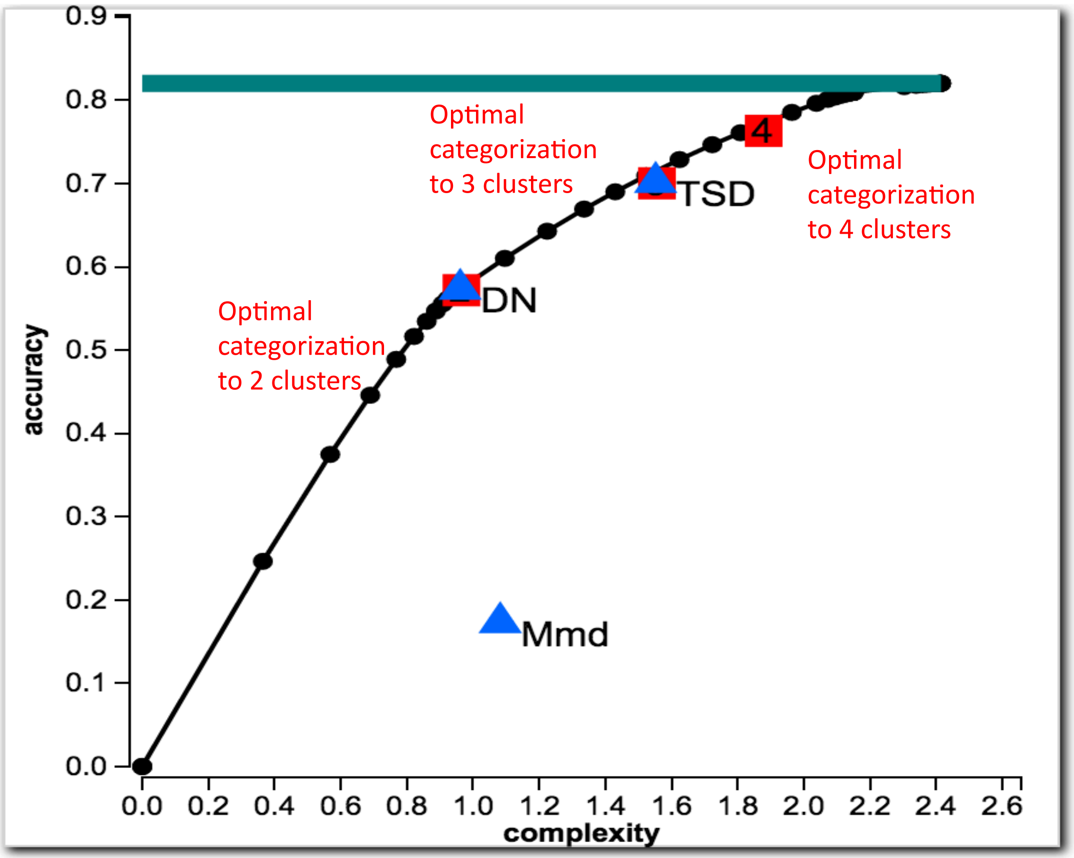
$$I(F;X) = \sum_{f,x} p(F=f, X=x) \log_2 \left(\frac{p(F=f, X=x)}{p(F=f)p(X=x)} \right)$$

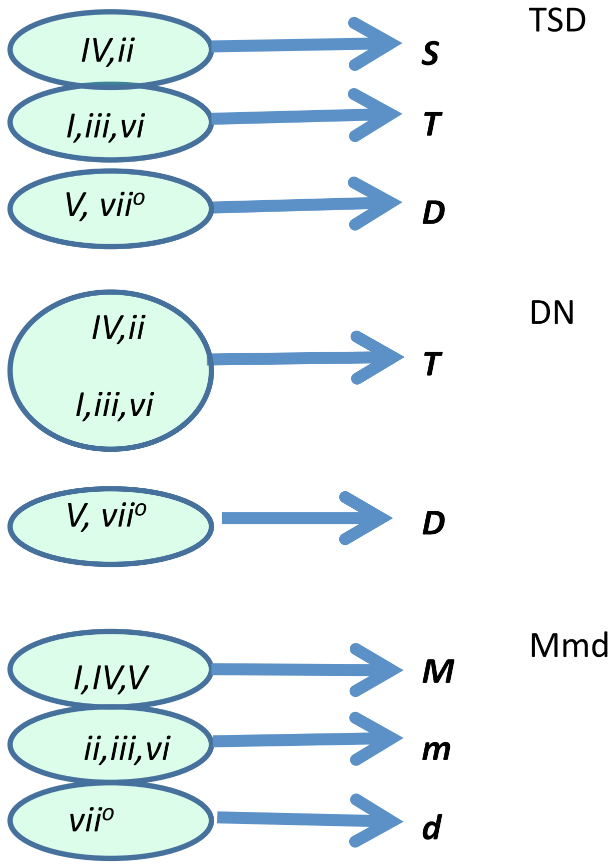


Tishby, N., F. Pereira, and W. Bialek. (1999). The information bottleneck method. Paper presented at the the 37th Annual Allerton Conference on Communication, Control and Computing

Jacoby, Tishby, and Tymoczko (2016)

"An Information Theoretic Approach to Chord Categorization and Functional Harmony."






```
4 categories: scroll to see the full list
category 1 : ii
category 2 : I, iii
category 3 : V, viio
category 4 : IV, vi
```

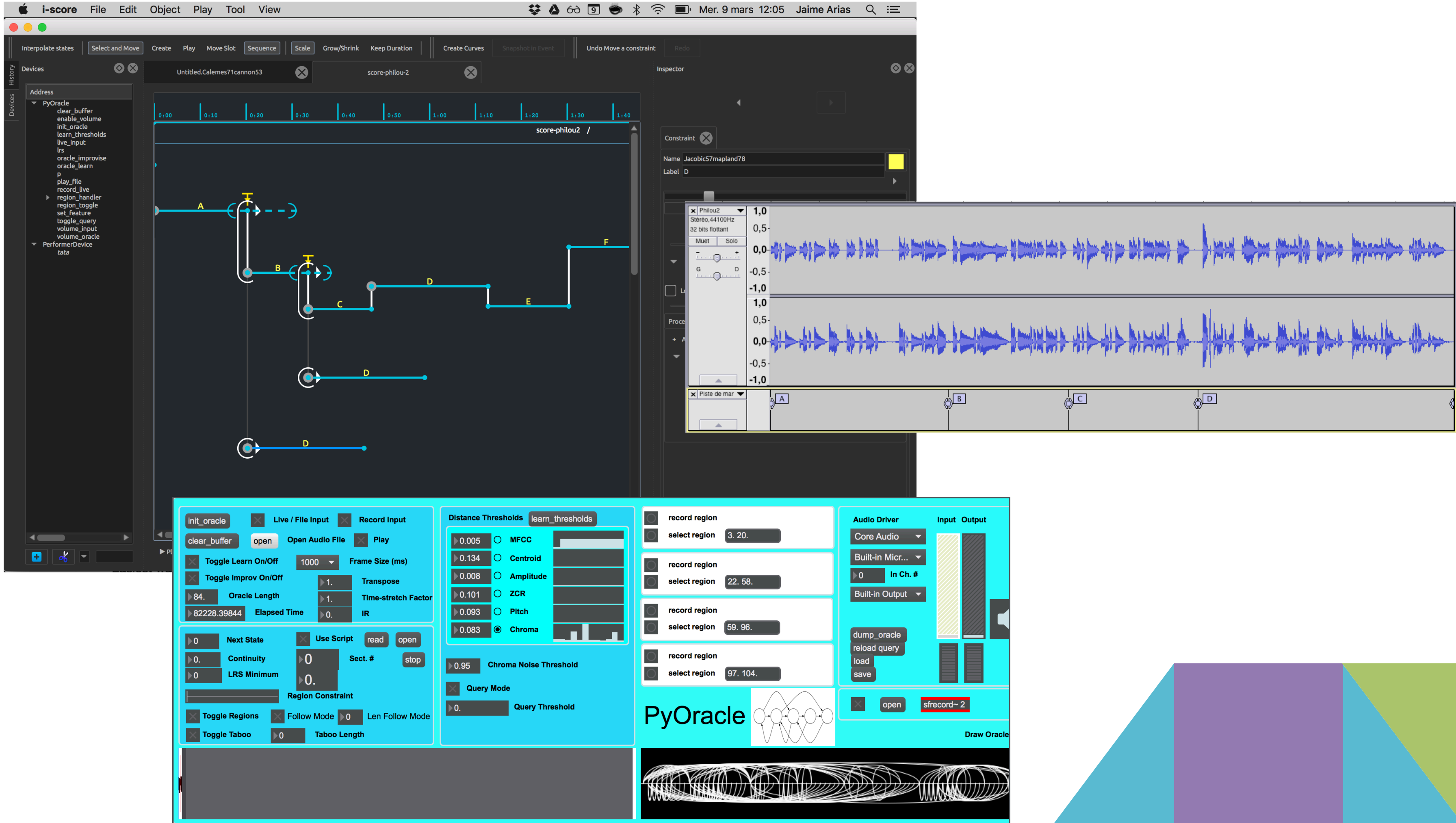
Optimal (deterministic) clusters from IB are identical to pre-existing ***TSD*** categorization scheme known from music theory. However an alternative existing scheme (according to the mode of the chord, ***Mmd***) is suboptimal.

Towards Automatic I-Score

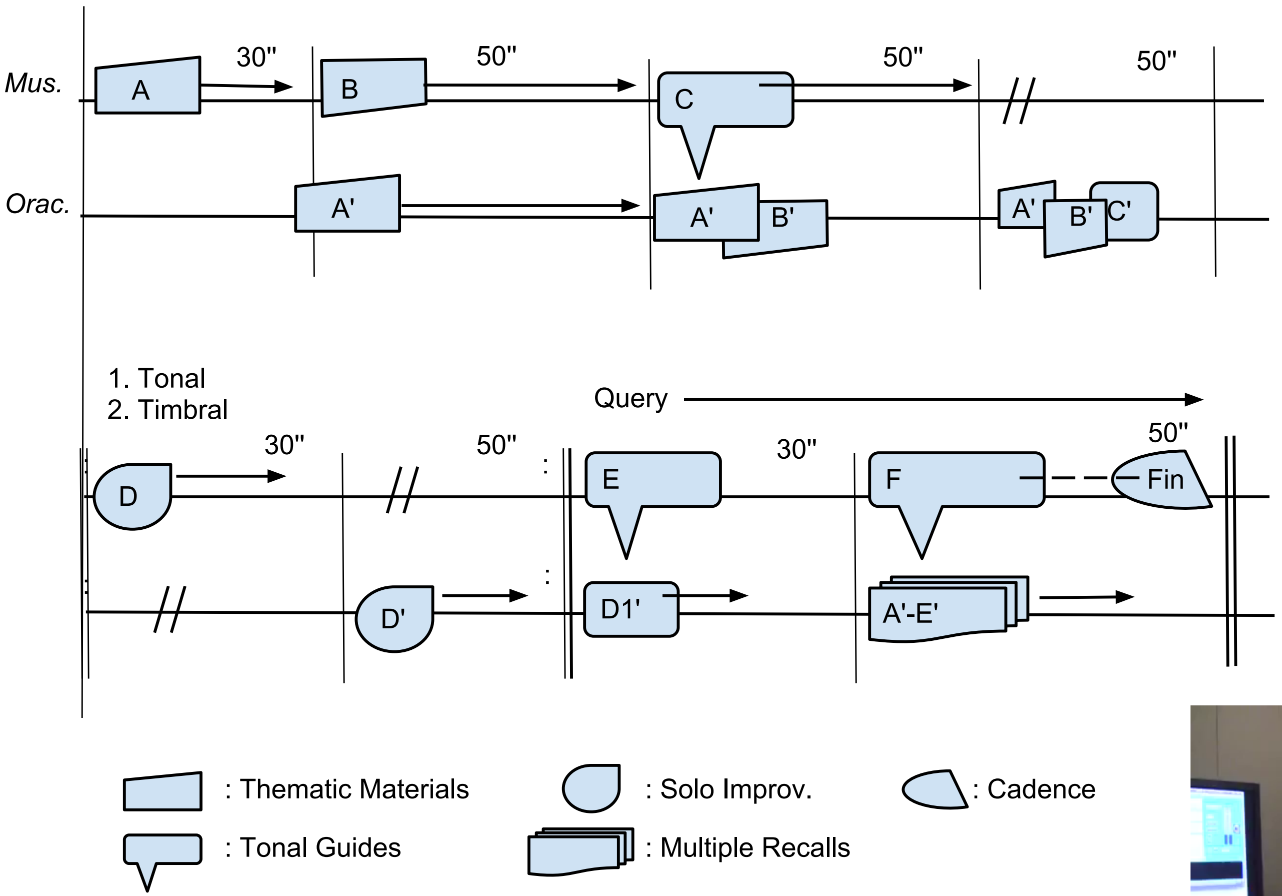
Using analysis of a recording produce:

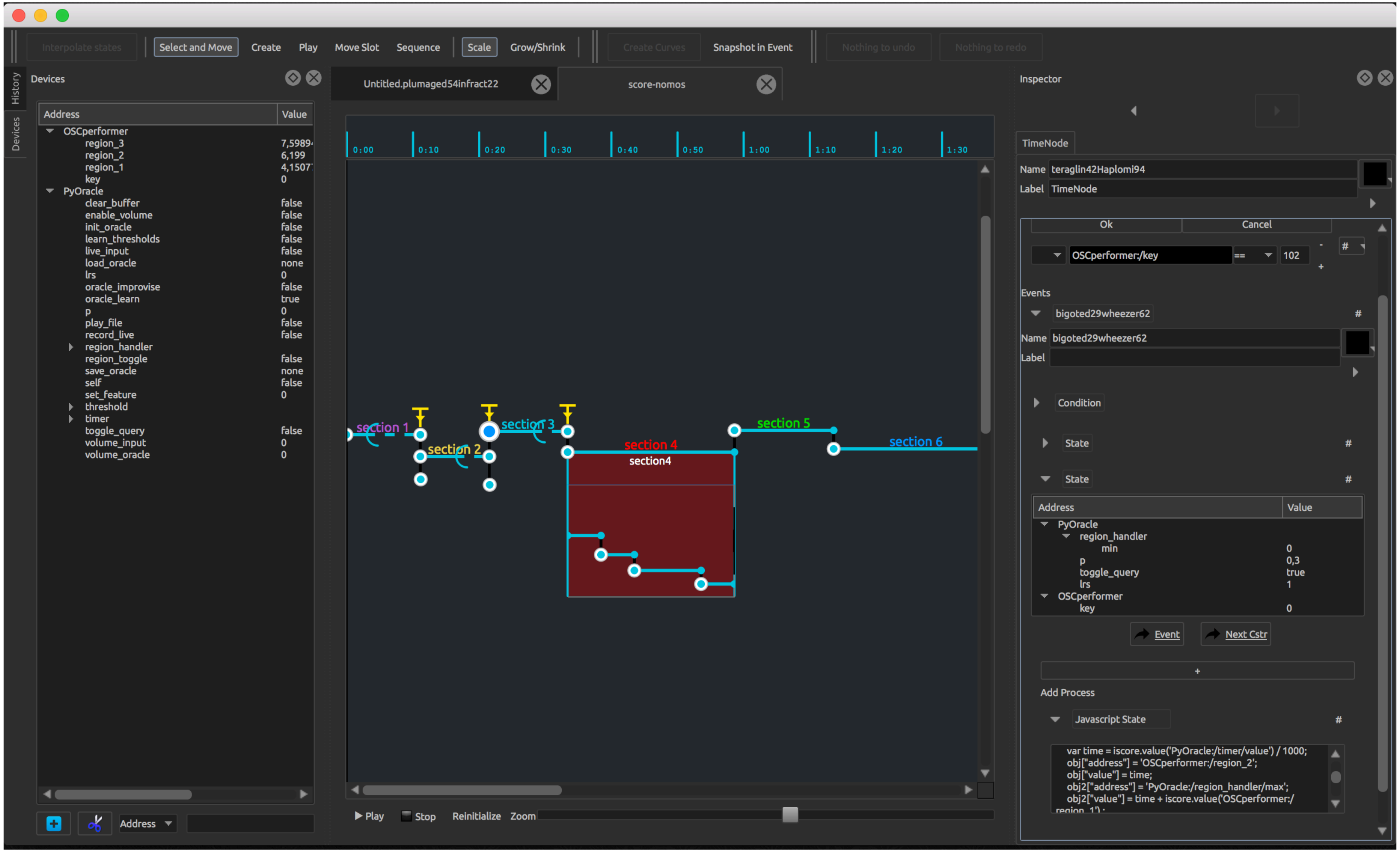
1. An automatic set of generative models
2. Segmentation
3. Identification of repeated sections
4. Rules for transition

Sound and Interaction



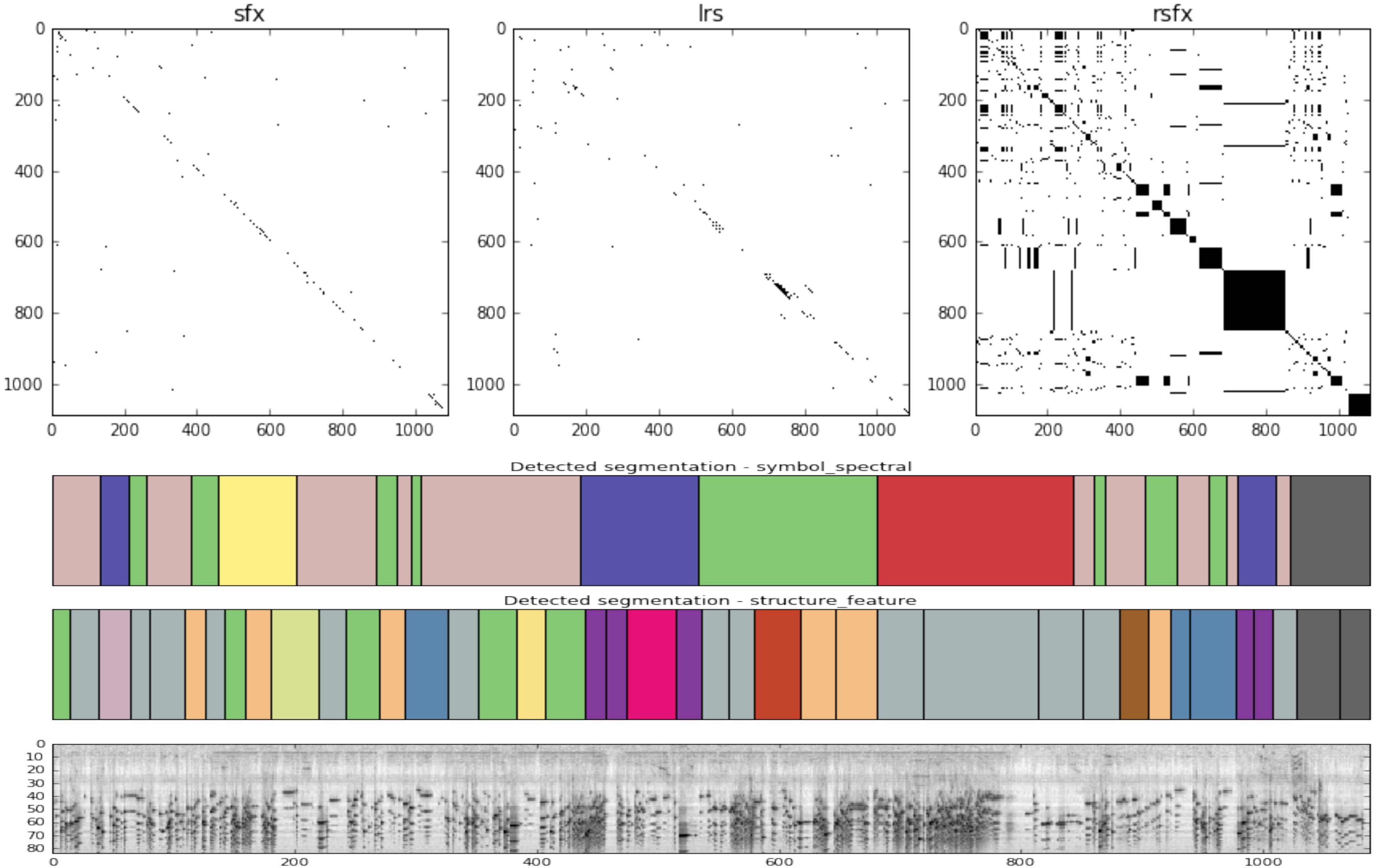
Nomos Ex Machina (No. 1)





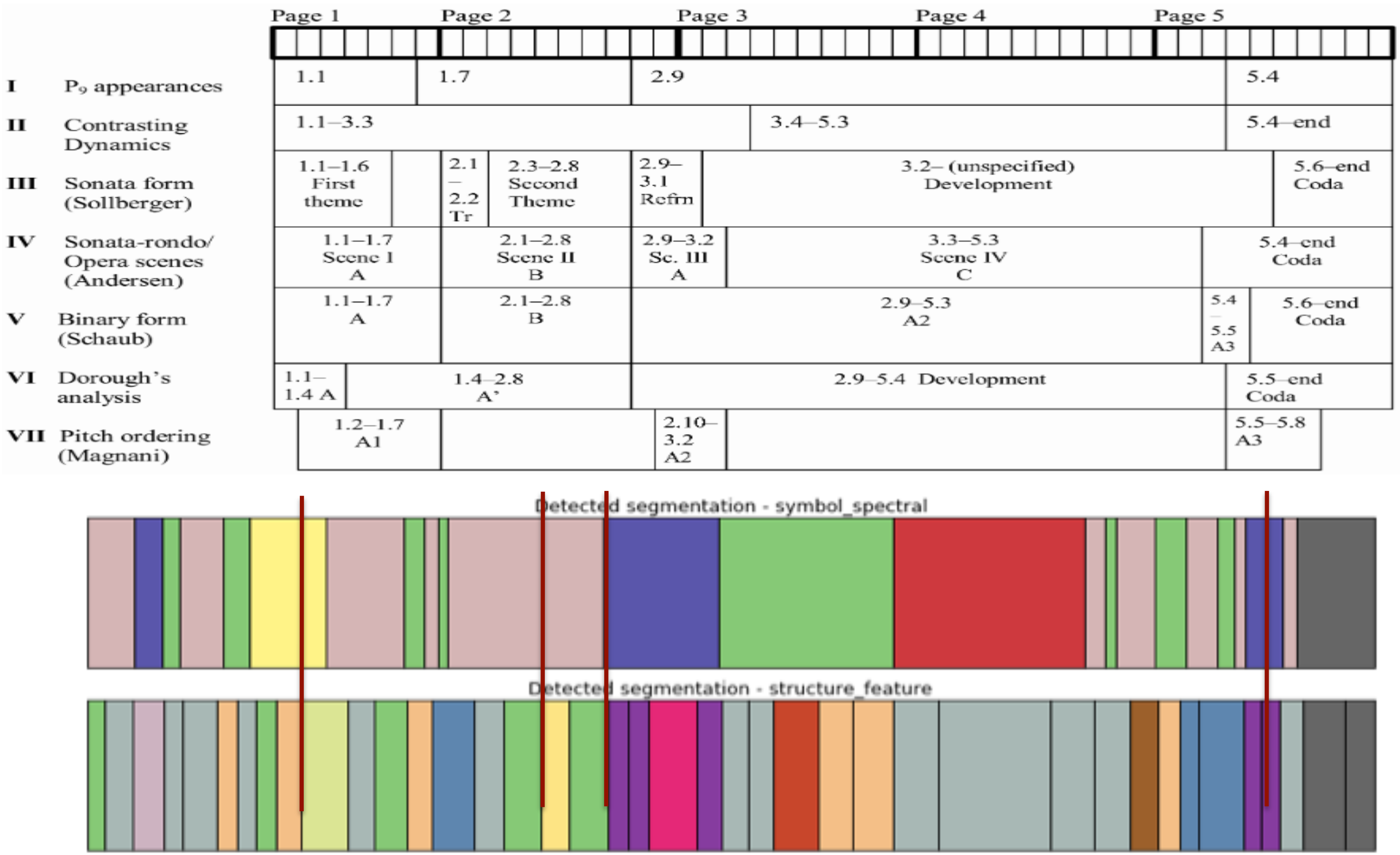
Interpolate states
Select and Move
Create
Play
Move Slot
Sequence
Scale
Grow/Shrink
Create Curves
Snapshot in Event
Nothing to undo
Nothing to redo
History
Devices
Devices
Address
Value
OSCperformer
region_3 7,5989
region_2 6,199
region_1 4,1507
key 0
PyOracle
clear_buffer false
enable_volume false
init_oracle false
learn_thresholds false
live_input false
load_oracle none
lrs 0
oracle_improvise false
oracle_learn true
p 0
play_file false
record_live false
region_handler
region_toggle false
save_oracle none
self false
set_feature 0
threshold
timer
toggle_query false
volume_input 0
volume_oracle 0
Address
Untitled.plumaged54infract22
score-nomos
0:00 0:10 0:20 0:30 0:40 0:50 1:00 1:10 1:20 1:30
section 1
section 2
section 3
section 4
section4
section 5
section 6
Play
Stop
Reinitialize
Zoom
Inspector
TimeNode
Name teraglin42Haplomi94
Label TimeNode
Ok
Cancel
OSCperformer:/key
==
102
#
Events
bigoted29wheezer62
Name bigoted29wheezer62
Label
Condition
State
State
Address
Value
PyOracle
region_handler
min 0
p 0,3
toggle_query true
lrs 1
OSCperformer
key 0
Event
Next Cstr
+
Add Process
Javascript State
var time = iscore.value('PyOracle:/timer/value') / 1000;
obj["address"] = 'OSCperformer:/region_2';
obj["value"] = time;
obj2["address"] = 'PyOracle:/region_handler/max';
obj2["value"] = time + iscore.value('OSCperformer:/
region_1');

Berio Sequenza I



Berio Sequenza I

Fig. 11.1 Seven interpretations of the formal structure of *Sequenza I*



Open Work

- Eco's exemplary open musical works consist of rigorously composed parts that may be assembled in many different orders (as in Stockhausen's Klavierstück XI [1957]), or of parts whose relation is capable of change even if their order is fixed (as in the durations and tempos of Berio's original Sequenza for flute [1958]);
- An open work is not improvisatory like jazz or Indian raga, nor is it a complete refusal of intention and control, as in Cage's Zen-influenced works.
- Open works are not indeterminate, not totally without pre-existing structure, but rather suspended between many different but fully determinate structures.

Thus they ***enable a composer****, in principle at least,* ***to reconcile the apparently contradictory imperatives of complete control,*** which reached its apotheosis in the total serialism of the earlier Boulez and Stockhausen, ***and the freedom in performance*** that was the hallmark of Cage's aleatory works.

Timothy Murphy, 99

Model of Creativity

Technique – Sensibility – Intent

- Machine Improvisation *(ML / AI)*
- Music Information Dynamics *(IT / Dynamic Sys.*)
- Scenario and Narrative (*Formal Methods*, *HCI*)

Summary

- Creativity can be seen as a dynamic process of information transfer between different levels (resolutions) existing in the message structure
- It operates by finding a tradeoff between complexity of individual elements (variation) and the ability to establish a coherent structure on a higher level (abstraction and selective retention)
- Controlling information production can be done by maximizing information (minimizing an error) between the information source and an external specification such as a query sequence

end