Prosody: Thinking Outside the Box

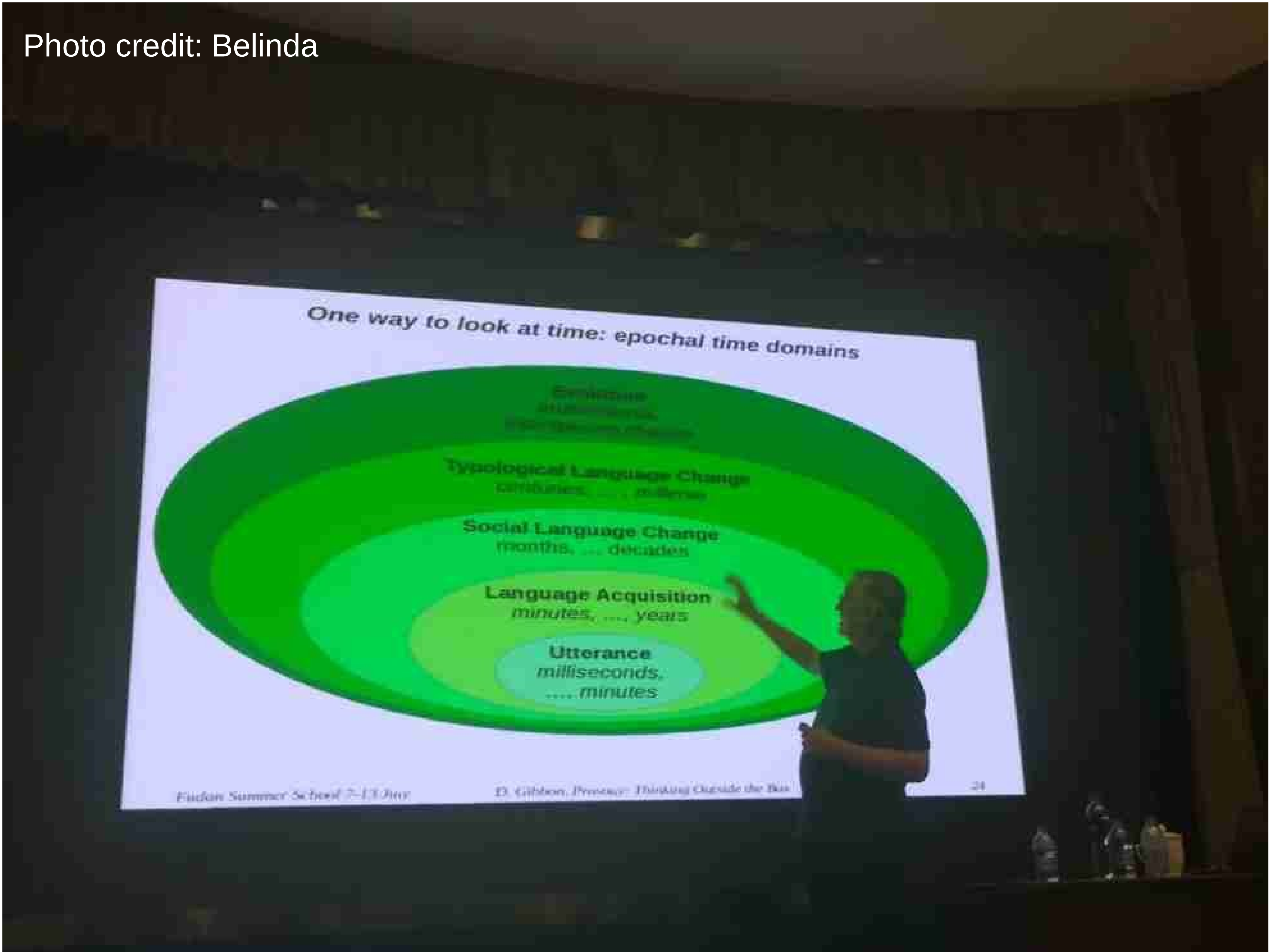
Lecture 2
The Phonetics of Prosody 1: Rhythm

Dafydd Gibbon

Bielefeld University

Fudan University Summer School: Contemporary Phonetics and Phonology
Shanghai, 7–13 July 2018

Photo credit: Belinda



Overview

1. What is rhythm?
2. Aspects of timing:
   - the TGA (Time Group Analysis) online software
   - TGA application: timing and tone in Tem (ISO 639-3 kfg, Togo)
3. Isochrony models of rhythm:
   - a one-dimensional approach
   - a two-dimensional approach
   - a three-dimensional approach
   - ***BUT MAYBE THERE IS MORE THAN ONE RHYTHM!***
4. The phonological basis of rhythm: ‘abstract oscillation’
   - finite transition networks with iteration
   - the concept of recursion
5. Towards an understanding of physical rhythm in speech
   - amplitude modulation
   - the envelope spectrum (next lecture!)

What is Rhythm?

What is rhythm?

1. One property of rhythm:
   - ‘isochrony’ (equal timing)
     - for example of morae, syllables, feet, …
     - or of larger units, in rhetorical speech or poetry
2. Another property of rhythm:
   - structural similarity of isochronous units
3. Yet another property of rhythm:
   - alternation (in structurally similar isochronous units)
4. A more general definition:

   RHYTHM IS OSCILLATION

Some rhythms are easy to identify physically.

Speech rhythm is not. It is an *emergent* property of many top-down and bottom-up factors.

Aspects of Timing - TGA

First Things First: Practical Prosody

Question: What can I do with my Praat annotations?

Answer: An annotation is a relation between labels and time-stamps. So:

- Extract and display labels.
- Extract and display time-stamps.
- Subtract neighbouring time-stamps to find durations.
- Calculate descriptive statistics over durations:
  - Average duration, average speech rate (for a particular tier)
  - Standard deviation, normalised Pairwise Variability
- Create visualisations:
  - Rhythm graphs
  - Scatter plots
  - Time trees

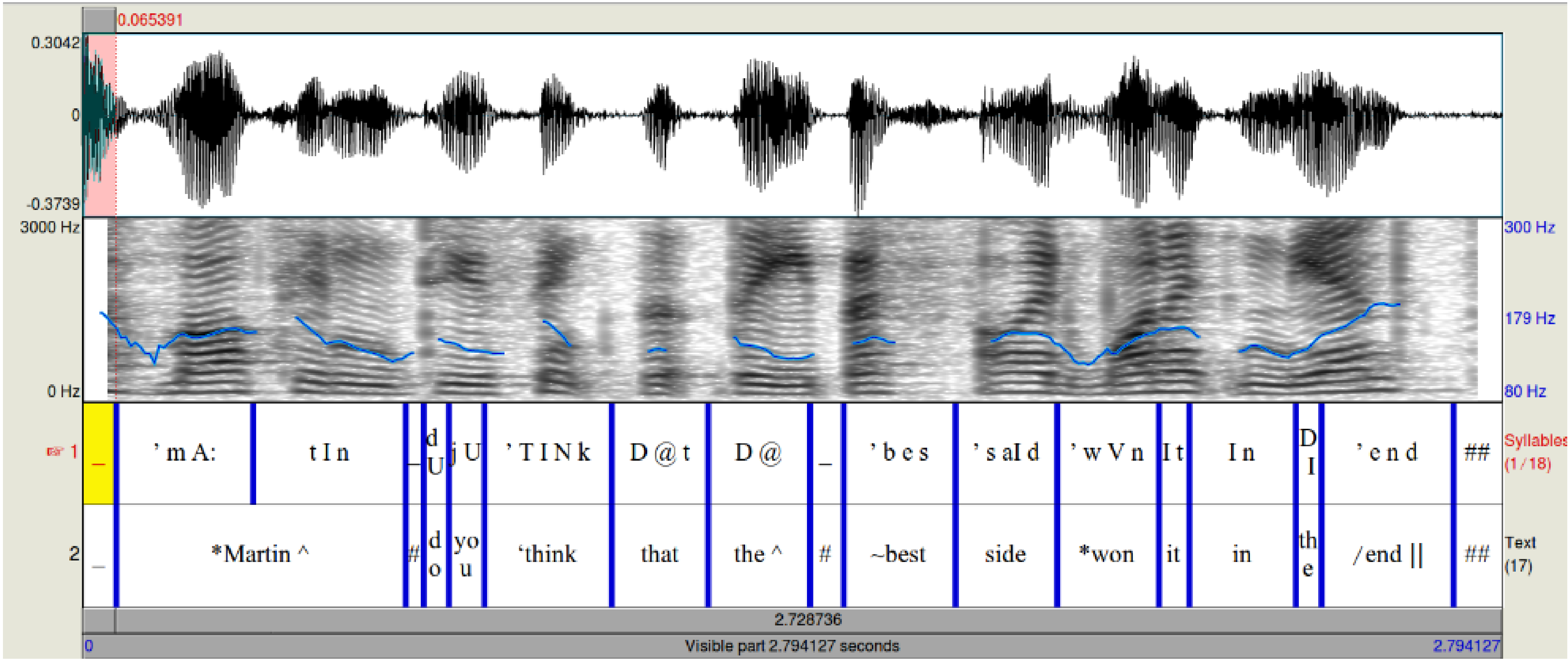
And use the Time Group Analyzer (TGA)

Time Group Analyzer (TGA)

First Things First: Practical Prosody

So here is a Praat visual model with

- waveform
- F0 trace
- 8 annotation tiers



First Things First: Practical Prosody

The Praat annotation file is just text.

It represents a small database of annotations for one recording.

This is what the Praat annotation file looks like:

1. <u>each interval tier</u> is a sequence of intervals
2. <u>each interval</u> represents an <u>event</u> consisting of
   - a <u>label</u>
   - a pair of <u>time-stamps</u>

```
File type = "ooTextFile"
Object class = "TextGrid"

xmin = 0
xmax = 2.7941273844617305
tiers? <exists>
size = 2
item []:
    item [1]:
        class = "IntervalTier"
        name = "Syllables"
        xmin = 0
        xmax = 2.7941273844617305
        intervals: size = 18
        intervals [1]:
            xmin = 0
            xmax = 0.0653912275449664
            text = "_"
        intervals [2]:
            xmin = 0.0653912275449664
            xmax = 0.3353912275449664
            text = """ m A:"
        intervals [3]:
            xmin = 0.3353912275449664
            xmax = 0.6353912275449667
            text = "t I n"
        intervals [4]:
            xmin = 0.6353912275449667
            xmax = 0.6703912275449664
            text = "_"
        intervals [5]:
            xmin = 0.6703912275449664
            xmax = 0.7203912275449667
            text = "d U"
        intervals [6]:
            xmin = 0.7203912275449667
            xmax = 0.7903912275449665
            text = "j U"
        intervals [7]:
            xmin = 0.7903912275449665
            xmax = 1.0403912275449665
            text = """ T I N k"
        intervals [8]:
            xmin = 1.0403912275449665
            xmax = 1.2303912275449664
            text = "D @ t"
        intervals [9]:
            xmin = 1.2303912275449664
            xmax = 1.4303912275449662
            text = "D @"

                (… etc.)
```

Inductive analysis: from pitch patterns to categories

Phonetic mode (signal analysis, ‘clock time’):

- Domains:
  - time functions (articulatory, acoustic, auditory)
- Analysis:
  - time domain
  - frequency domain (spectrum)

Tonal tokenisation (e.g. Tobi, ‘categorial time’, ‘rubber time’):

BoundaryTone PitchAccentTone PitchAccentTone* BoundaryTone

Boundary tone: { **H%**, **%L%** }

PitchAccentTone: { **H***, **L***, **L*H**, **LH***, **H*L**, **HL***, **H*H** }

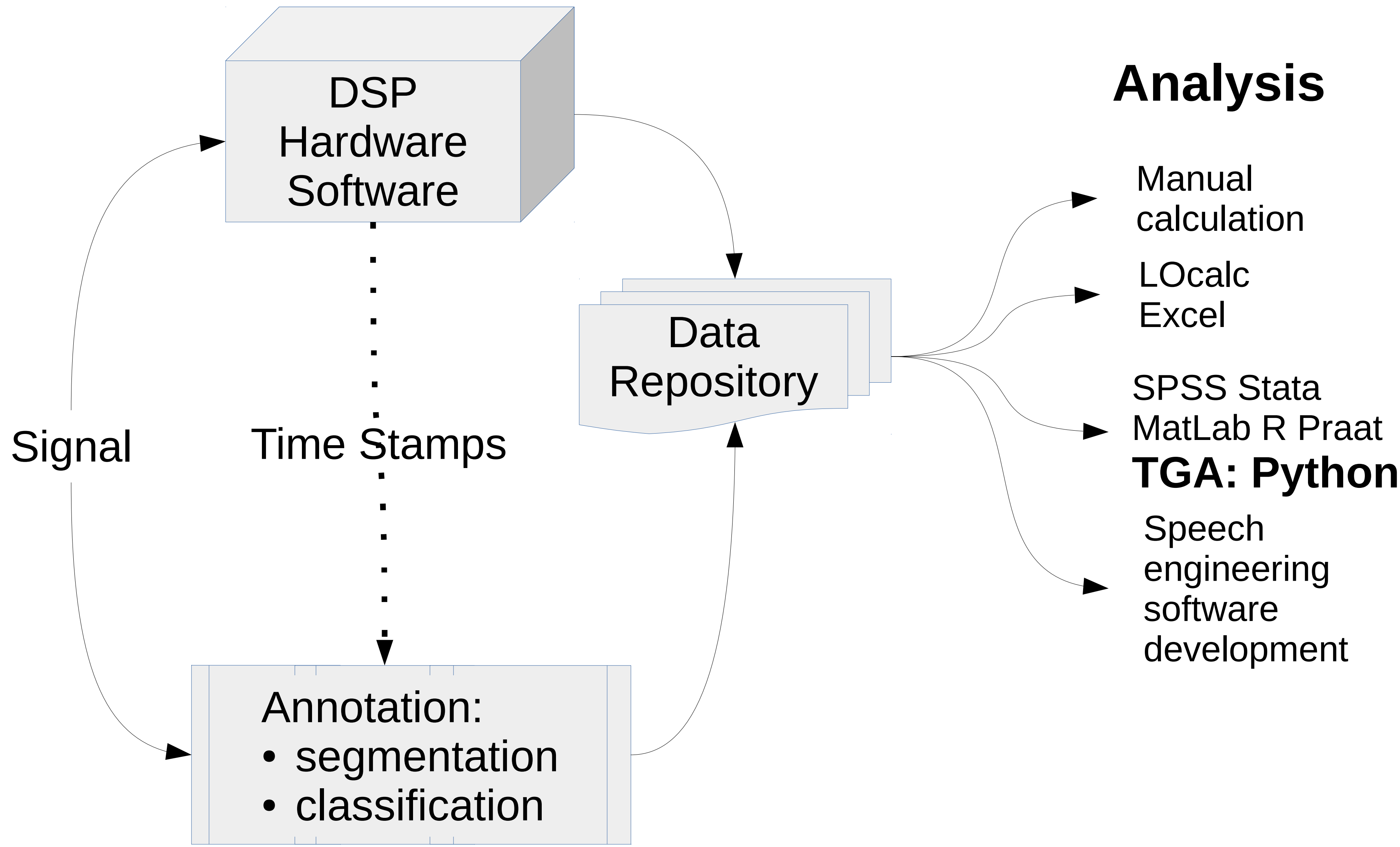
Contour parsing (Tonetics):

prehead head body nucleus tail

Categorial interpretation (prosodic phonologies):

- Configurative: Initial/final boundary; ip, IP boundary
- Contrastive: accents
- Culminative: accent placement

1D, 2D and 3D Annotation Mining (Labels + Time-stamps)



Online Application: TGA (Time Group Analyser)

Time Group Analyzer (TGA) Online TextGrid Processor: Overview

1. TGA specifications
   - Requirements, design, implementation
2. Design and Implementation
3. TGA Input, screenshot
4. TGA Output (CGI response)
   - text extraction
   - syllable duration statistics reports
   - Duration Bars & Duration Difference Tokens
   - DDTs, DBs and Time Tree bracketing, DDT n-gram count
   - induced Time Tree
   - Wagner Quadrant Plot
5. Published applications: example
6. Planned: NLP applications, box plots

Time Group Analyzer (TGA)

Time Group Analyzer (TGA) specifications

1. Requirements specification
2. Design and implementation
3. Input parameters
4. Outputs
5. Applications

Time Group Analyzer (TGA)

Requirements specification (1)

1. Annotation mining: the extraction of information from annotations, e.g. Praat TextGrids.
2. In speech technology, annotated data are generally mined (semi-)automatically and efficiently.
3. In phonetics, manual or semi-manual mining is common but inefficient:
   - copying Praat information into a spreadsheet
   - defining functions sich as nPVI in the spreadsheet
   - calculating and generating graphics
4. In phonetics and linguistics there is a need for faster and more consistent mining of larger numbers of annotated (e.g. TextGrid) files, without necessarily working with programming experts

Time Group Analyzer (TGA)

Requirements specification (1)

The Time Group Analyzer (TGA) is designed to support phoneticians by automatizing a wide range of relevant computational tasks:

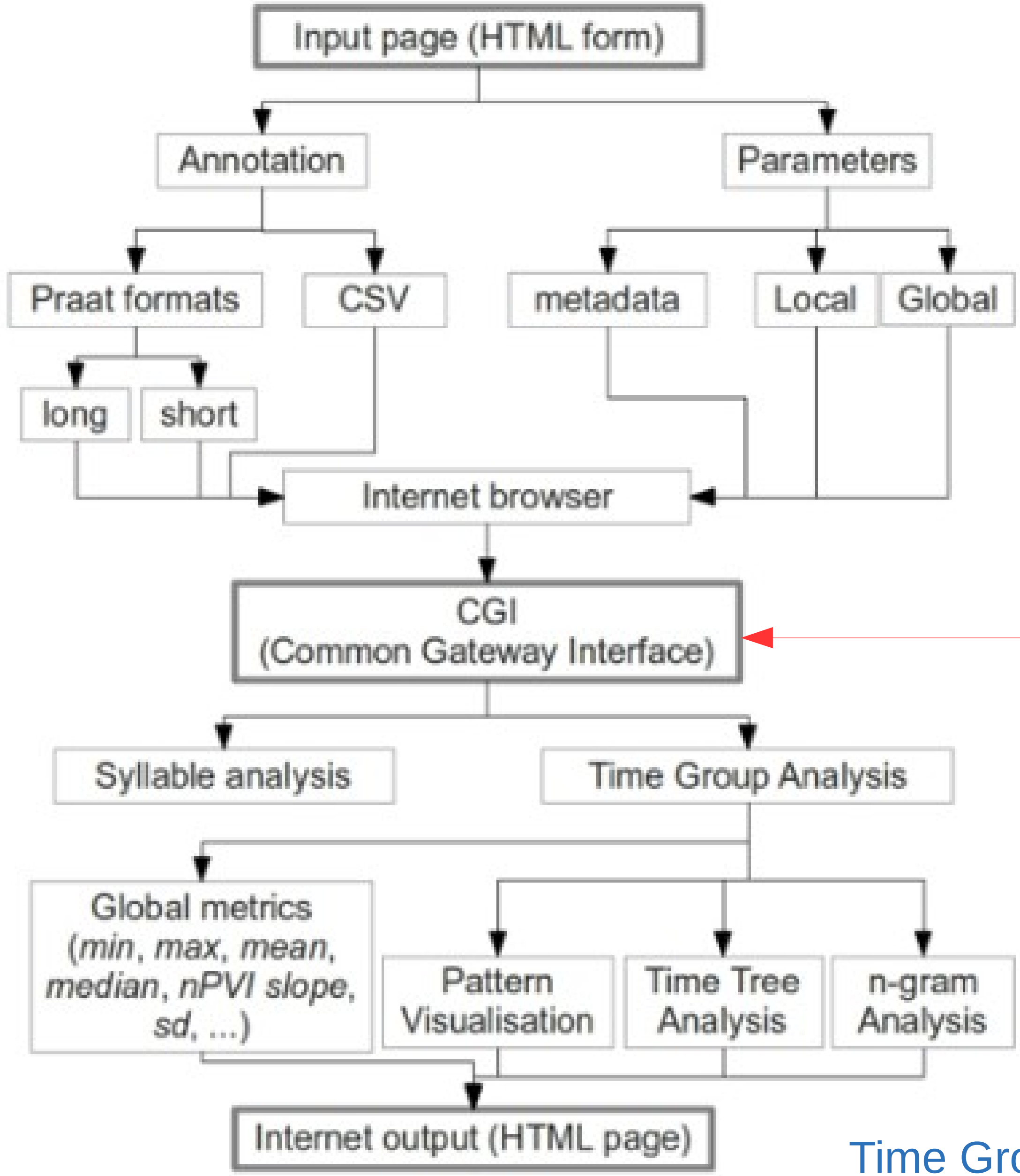
- duration extraction from TextGrids to table format,
- basic descriptive statistics, slope, nPVI …,
- novel visualisations of timing structure:
  - global acceleration/deceleration patterns
    - local acceleration/deceleration (trochaic/iambic, shorter/longer) Duration Difference Tokens (DDTs) and DDT sequences, for study of rhythm
  - Time Trees, for comparison of timing with grammatical structure
  - Wagner Quadrant plots
  - Box plots of unit durations

Time Group Analyzer (TGA)

Design and Implementation (1)

1. Software Development Environment:
   - HTML, CGI, Python 2.7
2. Input:
   - Praat TextGrid (long or short),
   - CSV (Character Separated Values, with various separator chars).
3. Output:
   - HTML with text, syllable propertues, interpausal group statistics, Difference Tokens, Time Trees
   - CSV for further processing.

Design and Implementation (2)

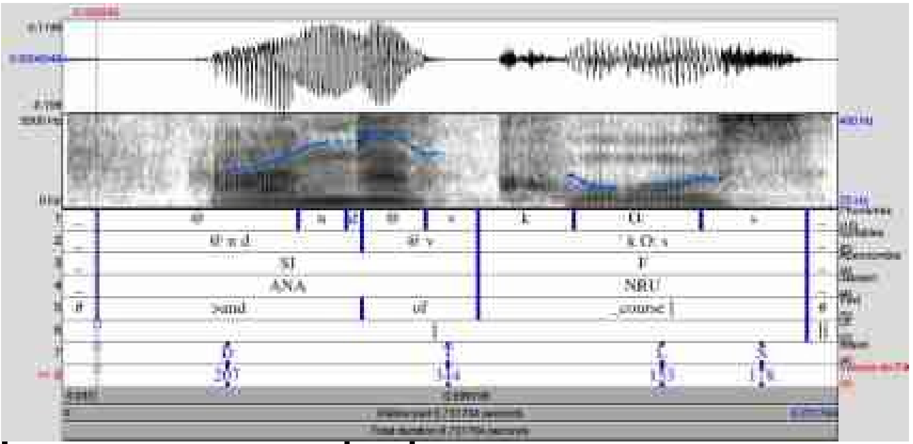


TGA dataflow

INTERNET

Time Group Analyzer (TGA)

TGA Input Parameters



1. Input form
   - Input control parameter choices
   - Time Group duration difference parameters
   - TextGrid (long or short) or CSV file
   - Output parameter choices
     - Statistics
       - Global (for entire file)
       - Local (for each time group)
     - Visualisations
       - Local (Duration Bars, Duration Difference Tokens)
       - Global (Wagner Quadrant Plots; sequence plots)

Time Group Analyzer (TGA)

TGA Input Form: screenshot

TextGrid input control parameters (long or short TextGrid format accepted; only Interval Tiers, obviously)

Tier name: Syllables (max length 20; not needed for CSV formats)

Pause symbol: _ (max length 20; also needed for CSV formats)

More than one pause symbol permitted; separate with spaces. Delete any of the examples which might occur as an annotation label. If your pause symbol is not in the examples given, enter it

Time Group duration difference parameters:

TG criterion: ◉ *pausegroup* ○ *deceleration* (increasing) ○ *acceleration* (decreasing)

Local threshold: 10 ms (try values less than common syllable lengths, e.g. 0 ... 300 ms)

Used for local pattern extraction and TimeTree parsing.

Local pattern symbols: **Longer:** \ (1 char) **Shorter:** / (1 char) **Same:** = (1 char)

Time Tree criterion: ○ *(quasi-)iambic TTgt* ○ *(quasi-)trochaic TTlt* ◉ *show all TT*
○ *(quasi-)iambic TTgte* ○ *(quasi-)trochaic TTlte* ○ *do not show TT*

Global TG threshold range: 90 ··· 120 ms (minimal duration difference)

Ranges > 30 are not permitted because of possible server overload.

Global threshold is ignored with the 'pausegroup' criterion.

Experiment with values from 0 to 500 (negative values are permitted).

Equal range boundaries are adjusted to have range of 1, not null; if necessary values are switched to ensure 'low before high'.

Min TG length: > 2 (generally >2, as 'minimal rhythm')

Time Group output control parameters:

Print text? ◉ *no* ○ *yes* **n-grams?** ◉ *no* ○ *yes* **All outputs:** ○ *no* ◉ *yes*

TG element info? ◉ *no* ○ *yes* **Time Trees?** ◉ *no* ○ *yes*

TG detail? ◉ *no* ○ *yes* **CSV output?** ◉ *no* ○ *yes*

Time Group Analyzer (TGA)

TGA Input Form: parameter choices

1. Input control parameter choices
   - Textgrid tier name selection (e.g. 'Syllables', 'syllable', 'syll' - the tier can also be other items than syllables)
   - Pause symbol selection (e.g. '_', 'p', 'sil') for segmenting into interpausal groups
2. Time Group duration difference parameters:
   - Local TG threshold: sets the minimal difference (in ms) which counts as a difference; any difference below this threshold counts as equal duration
   - Local TG pattern symbols: select the symbols used for longer, shorter and equal duration difference relations ('duration difference n-grams')
   - Global threshold range: for time group induction
   - Minimum TG length in syllables (e.g. 2, 3)

Time Group Analyzer (TGA)

TGA Input Form: parameter choices

1. Output control parameter choices
   - Text extracted from labels
   - General information about TG elements
     - descriptive statistics, nPVI, regression slope and intercept
   - Details about individual interpausal groups:
     - descriptive statistics
     - visualisation:
       - Duration Difference Token (DDT) sequences
       - Time Trees (TT) types
   - DDT n-grams
   - TT types
   - Conversion of input TextGrid to Character Separated Value (CSV) format

Time Group Analyzer (TGA)

TGA Output (CGI response)

1. Text extraction
2. Descriptive statistics
   - tables
   - graphs
     - box plots
     - time plots of durations and duration differences
3. Time Group visualisations
   - DDT n-grams (local threshold dependent)
   - Time Trees (four types; local threshold dependent)
4. TextGrid input format reformatted as tables in Character Separated Value (CSV) format

Time Group Analyzer (TGA)

TGA Output: text extraction (English)

_
'mO: 'nju:z @ 'baUt D@ 're vr@n 'sVn 'mjVN 'mu:n _
'faUn d@ r@v D@ ,ju: nI fI 'keI Sn 'tS3:tS _
'hu:z 'kV r@nt lI In 'dZeIl _
f@ 't{ks I 'veI Zn _

TGA extract from first annotation file in Aix-MARSEC corpus
of BBC radio English
(SAMPA keyboard friendly encoding of the IPA)

Time Group Analyzer (TGA)

TGA Output: syllable duration properties (English)

Duration properties (syllables)

Attributes	Values	Attributes	Values
n:	31	intercept:	192.177
min:	50	slope:	0.242
max:	500	std:	102.258
mean:	195.81	nPVI:	54
median:	160.0	rPVI:	97
total:	6070	100*rPVI/med:	61
range:	450	nPVI*med/100:	86

Time Group Analyzer (TGA)

TGA Output: four dispersion measures

$PIM(I_{1,\ldots n})$	$=$	$\sum_{i \neq j} \lvert \log \frac{I_i}{I_j} \rvert$
$PFD(foot_{1\ldots n})$	$=$	$\frac{100 \times \sum \lvert MFL - len(foot_i) \rvert}{len(foot_{1\ldots n})}$ where $MFL = \frac{\sum_{i=1}^{n} len(foot_i)}{n}$
$rPVI(d_{1\ldots m})$	$=$	$\sum_{k=1}^{m-1} \lvert d_k - d_{k+1} \rvert / (m-1)$
$nPVI(d_{1\ldots m})$	$=$	$100 \times \sum_{k=1}^{m-1} \lvert \frac{d_k - d_{k+1}}{(d_k + d_{k+1})/2} \rvert / (m-1)$

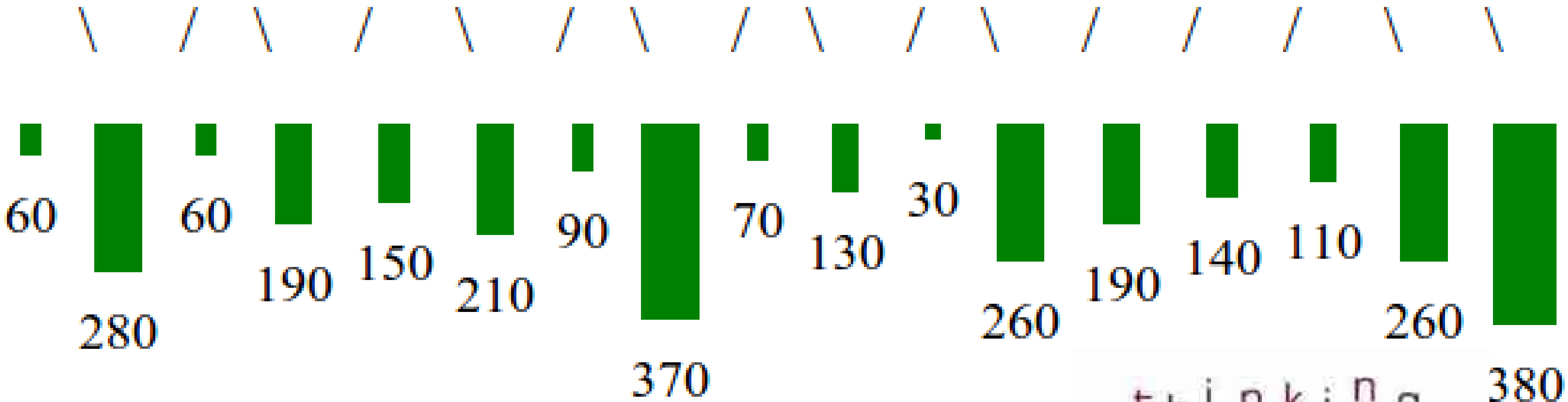
Time Group Analyzer (TGA)

TGA Output: overall statistics summary (English)

Summary table of global and accumulated TG duration functions (some do make sense...)
Time Group criterion: pausegroup, local threshold: 10, Min valid TG length: 2
Only inter-pause intervals measured; pauses not included

Overall duration:	6070	Overall raw longer, ms:	1510	Overall raw shorter, ms:	1410
Overall min:	50.00	Overall max:	500.00	Overall range:	450.00
Valid Time Groups:	4	Overall rate/sec:	5.11		
Components: global tendencies					
Overall mean:	195.81	Overall median:	160.00	Overall SD:	102.26
Overall npvi:	54.00	Overall intercept:	192.18	Overall slope:	0.24
Mean of means:	196.00	Median of means:	194.50	SD of means:	23.89
Mean of medians:	187.50	Median of medians:	170.00	SD of medians:	43.95
Mean of SDs:	93.25	Median of SDs:	89.12	SD of SDs:	18.97
Mean of nPVIs:	58.00	Median of mnPVIs:	52.00	SD of nPVIs:	5.59
Mean of intercepts:	154.94	Median of intercepts:	137.78	SD of intercepts:	56.84
Mean of slopes:	7.52	Median of slopes:	9.90	SD of slopes:	14.97
Components: correlations					
mean::TGdur:	0.384	median::TGdur:	-0.296	SD::TGdur:	0.935
nPVI::TGdur:	-0.623	slope::TGdur:	0.875	intercept::TGdur:	-0.762
nPVI::mean:	0.408	slope::mean:	-0.020	intercept::mean:	0.288
nPVI::median:	0.931	slope::median:	-0.710	intercept::median:	0.832
nPVI::SD:	-0.317	slope::SD:	0.666	intercept::SD:	-0.483

TGA Output:
Duration Difference Tokens and Duration Bars (English)



Duration Difference Tokens:

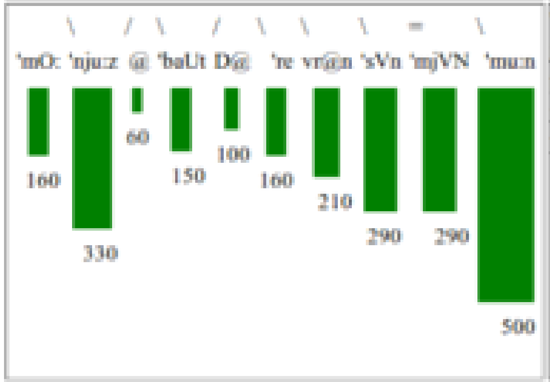
/ long-short
\ short-long
= equal
Identification depends on local duration difference threshold.

Duration Bars:

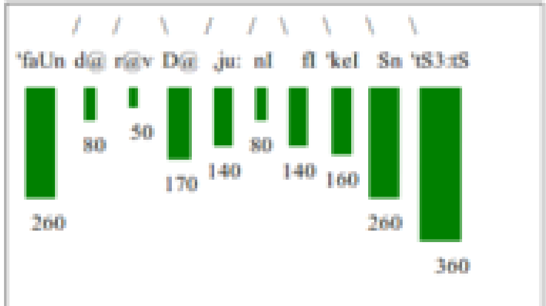
Linear relations to durations for both width and length.
Eyeball impression of rhythm, rate change, final lengthening...

Inspect the relation between DDTs and DBs directly. Time Group Analyzer (TGA)

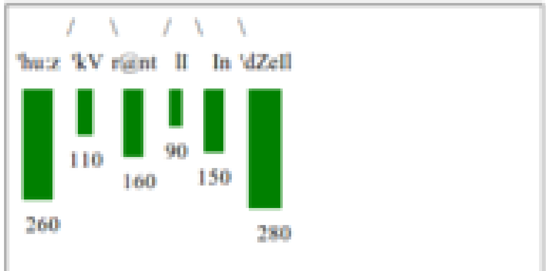
TGA Output: DDTs, DBs, Time Tree bracketing (English)



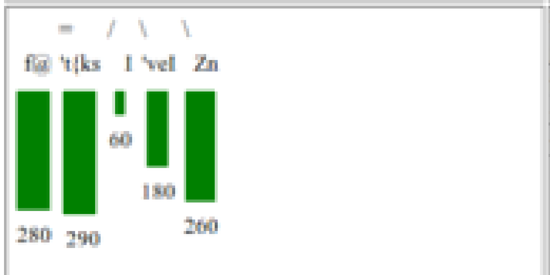
'mO::160 'nju:z:330 @:60 'baUt:150 D@:100 're:160 vr@n:210 'sVn:290 'mjVN:290 'mu:n:500 **PAUSE**:117 #
IambicTTgt: (('mO: 'nju:z) ((((@ 'baUt) ((D@ 're) vr@n)) 'sVn) ('mjVN 'mu:n)))
IambicTTgte: ('mO: 'nju:z @ 'baUt D@ 're vr@n ('sVn 'mjVN) 'mu:n **PAUSE**)
trochaicTTlt: (('mO: ('nju:z @)) ('baUt D@) ('re (vr@n ('sVn ('mjVN ('mu:n **PAUSE**))))))
trochaicTTlte: ('mO: 'nju:z @ 'baUt D@ 're vr@n 'sVn 'mjVN 'mu:n **PAUSE**)



'faUn:260 d@:80 r@v:50 D@:170 ,ju::140 nl:80 fl:140 'kel:160 Sn:260 'tS3:tS:360 **PAUSE**:184 #
IambicTTgt: ('faUn (((d@ (r@v D@)) ((ju: ((nl fl) 'kel)) Sn)) 'tS3:tS))
IambicTTgte: ('faUn d@ r@v D@ ju: nl fl 'kel Sn 'tS3:tS **PAUSE**)
trochaicTTlt: ((('faUn d@) r@v) ((D@ ju:) nl) fl 'kel (Sn ('tS3:tS **PAUSE**)))
trochaicTTlte: ('faUn d@ r@v D@ ju: nl fl 'kel Sn 'tS3:tS **PAUSE**)



'hu:z:260 'kV:110 r@nt:160 II:90 In:150 'dZeII:280 **PAUSE**:30 #
IambicTTgt: ('hu:z (('kV r@nt) ((II In) 'dZeII)))
IambicTTgte: ('hu:z 'kV r@nt II In 'dZeII **PAUSE**)
trochaicTTlt: (((('hu:z 'kV) (r@nt II)) (In ('dZeII **PAUSE**))))
trochaicTTlte: ('hu:z 'kV r@nt II In 'dZeII **PAUSE**)



f@:280 't{ks:290 I:60 'vel:180 Zn:260 **PAUSE**:674 #
IambicTTgt: (f@ ('t{ks (((I 'vel) Zn) **PAUSE**)))
IambicTTgte: (f@ 't{ks I 'vel Zn **PAUSE**)
trochaicTTlt: ((f@ ('t{ks I)) 'vel Zn **PAUSE**)
trochaicTTlte: (f@ 't{ks I 'vel Zn **PAUSE**)

Time Group Analyzer (TGA)

TGA Output: DDT n-gram count (English)

Difference digram ranks and counts (n=270):

1.[22%(60):/\] **2.**[20%(55):\/] **3.**[11%(31):\\] **4.**[9%(24):\}] **5.**[6%(17):{\] **6.**[6%(15)://] **7.**[5%(14):{/] **8.**[4%(11):=\] **9.**[4%(11):/=] **10.**[3%(9):\=] **11.**[3%(8):=/] **12.**[2%(6):/}] **13.**[1%(4):=}] **14.**[1%(3):{=] **15.**[1%(2):==]

Summary:

42% alternations in the top 2 places

Next step:

Check DDT trigrams etc. for /\/, \/\, /\\, \// etc.

Note:

DDT *n*-gram identification is determined by the *local threshold*

Time Group Analyzer (TGA)

TGA Output: induced Time Tree (English)

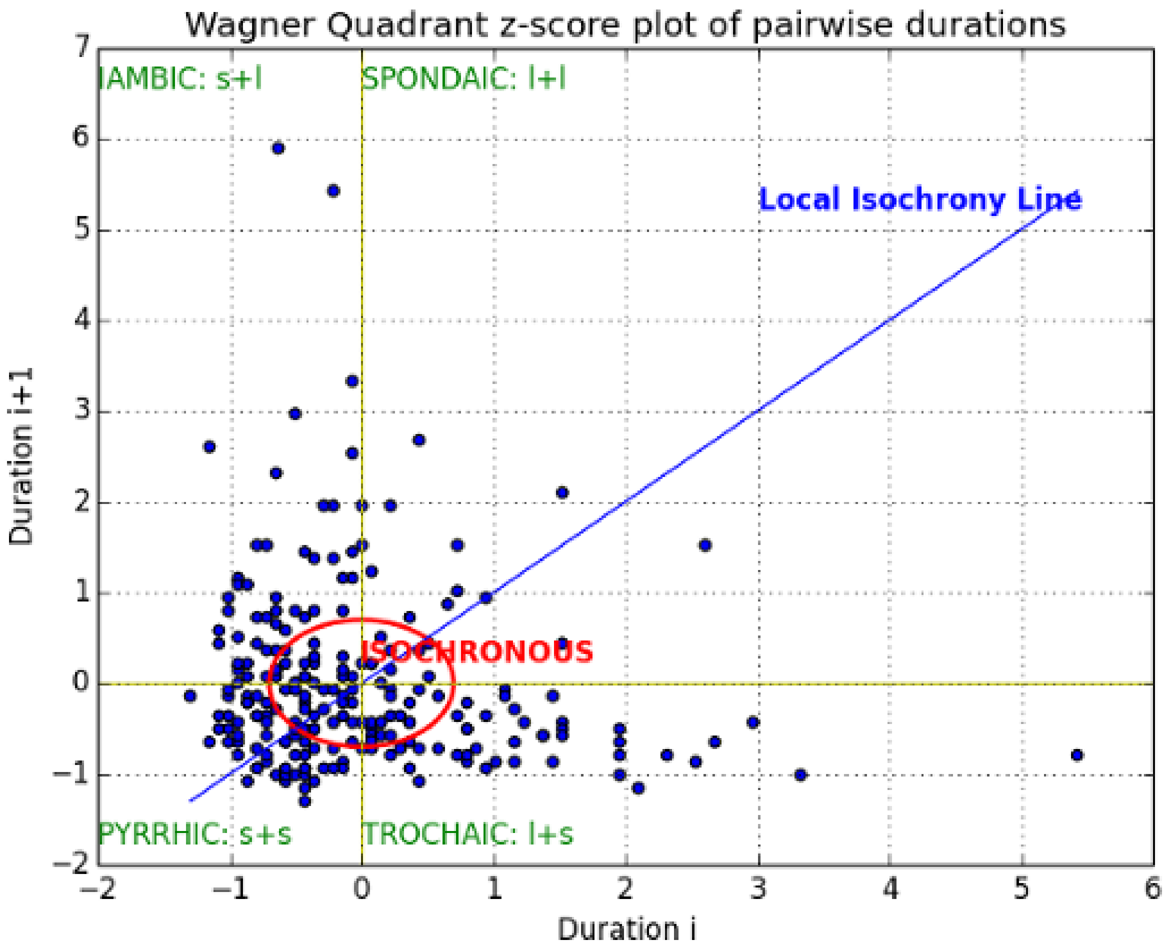
```
(((@  'baUt)
    ((('{N  glI)
          (kn  {m))
        ('bI  vl@ns)))
    ((((t@  D@)
          ('brI  tIS))
        (('kaUn
            (sl
               (@v  'tS3:)))
        tSIz))
    PAUSE))
```

Time tree:

Induced from digram duration relations
Larger groupings inherit longest duration from constituent
Parenthesis notation
Python automatic prettyprint

Time Group Analyzer (TGA)

TGA Output: Wagner Quadrant Plot (English)



Scatter plot:

z-scores of durations

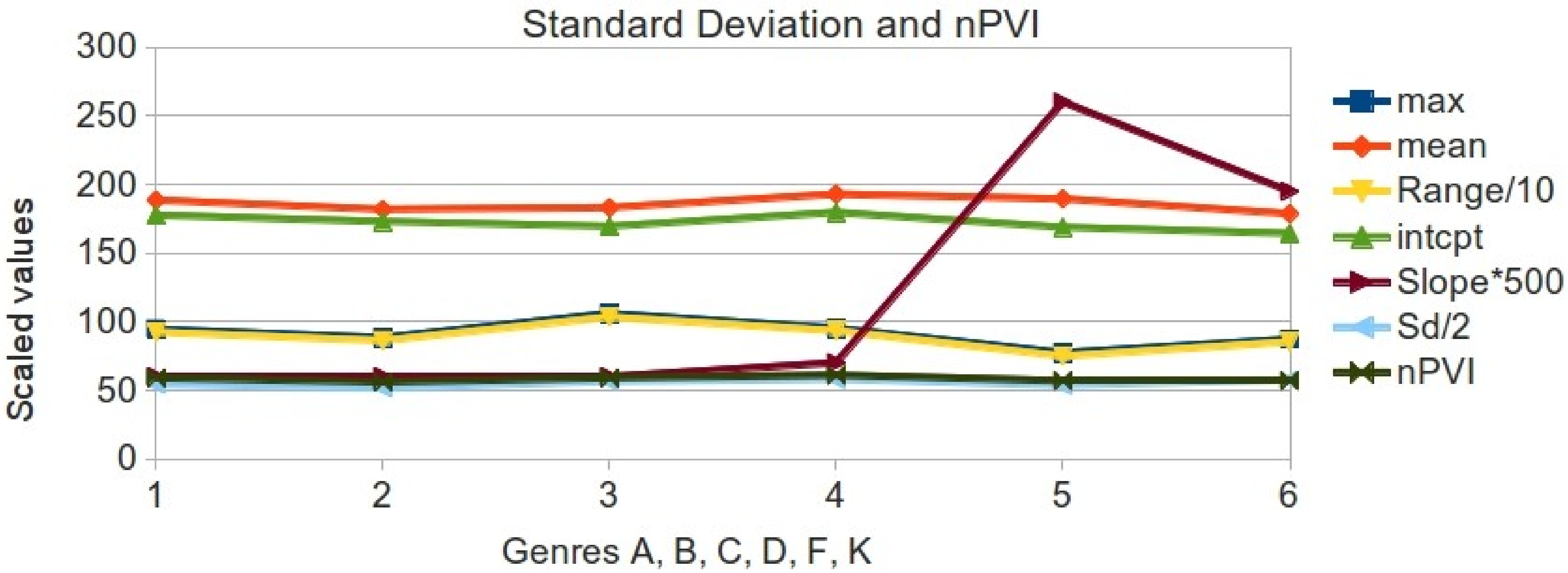
duration relations d_i and d_{i-1} on X and Y axes

syllable timing: typically random distribution

toot/stress timing: typically ‘L-shaped’, as in this example (Aix-MARSEC genre G)

Time Group Analyzer (TGA)

Published further analyses: example



Comparison of different timing measures:
nPVI, SD, etc.

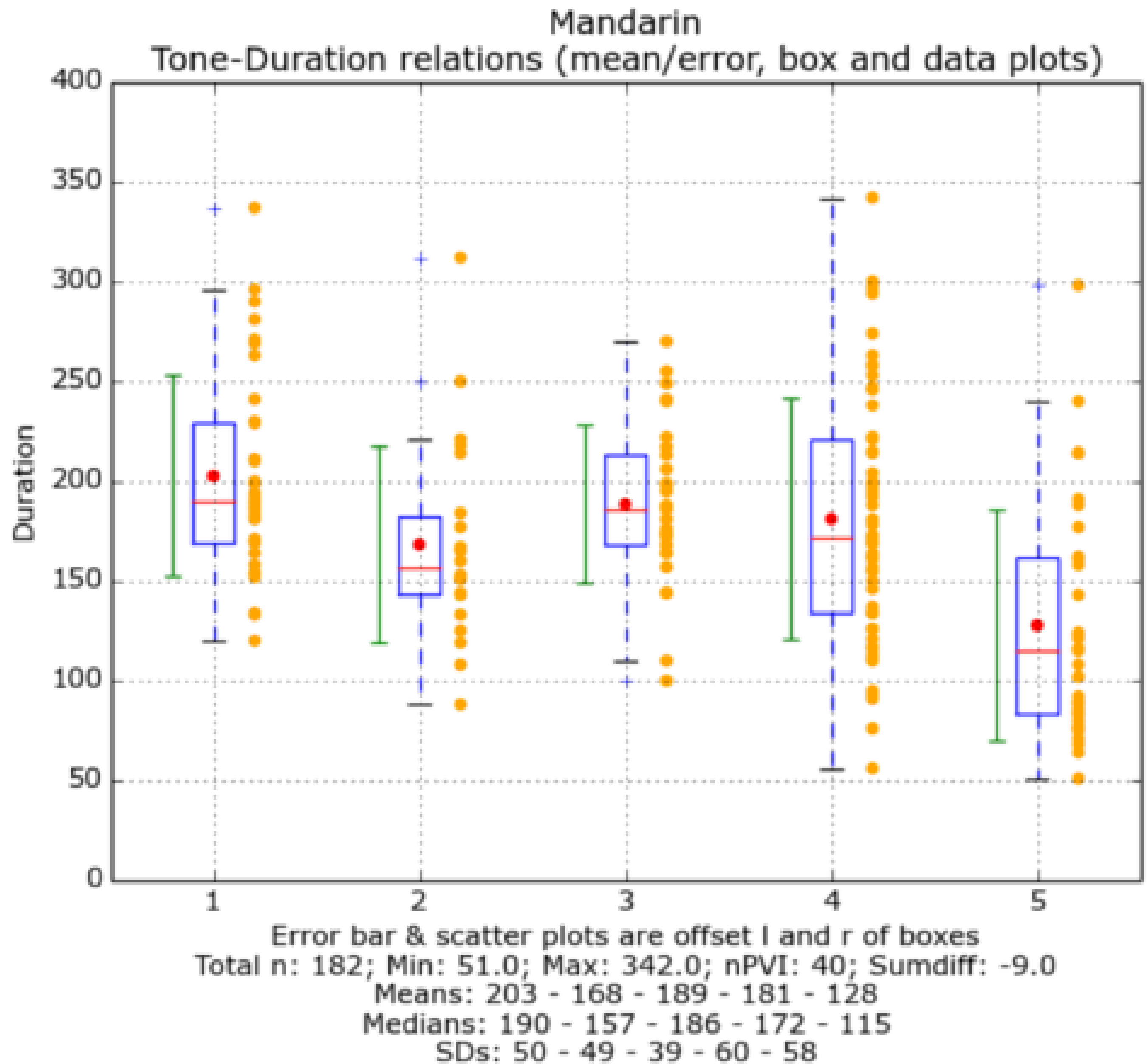
NLP applications, box plots

Corpus linguistic applications

Word frequency lists
Concordance

Visualisations

For example, automatic generation of syllabic time-tone relations in Mandarin:



Time Group Analyzer (TGA)

Time Group Analyzer: Summary

1. TGA specifications
   - Requirements, design, implementation
2. Design and Implementation
3. TGA Input, screenshot
4. TGA Output (CGI response)
   - text extraction
   - syllable duration statistics reports
   - Duration Bars & Duration Difference Tokens
   - DDTs, DBs and Time Tree bracketing, DDT n-gram count
   - induced Time Tree
   - Wagner Quadrant Plot
5. Pubished applications: example
6. Planned: NLP applications, box plots

Time Group Analyzer: Bibliography

Yu, Jue and Gibbon, Dafydd, Criteria for database and tool design for speech timing analysis with special reference to Mandarin, Oriental COCOSDA 2012 (cf. IEEEexplore Conf ID 21048)

Gibbon, Dafydd, TGA: a web tool for Time Group Analysis, TRASP 2013 (poster)

Yu, Jue, Timing analysis with the help of SPPAS and TGA tools, TRASP 2013 (poster)

Klessa, Katarzyna and Dafydd Gibbon, Annotation Pro+TGA: automation of speech timing analysis, LREC 2013.

Yu, Jue, Dafydd Gibbon and Katarzyna Klessa, Computational annotation-mining of syllable durations in speech varieties, Speech Prosody 7, 2014.

Yu, Jue and Dafydd Gibbon, How natural is Chinese L2 English? ICPhS, Glasgow, 2015.

Yu, Jue and Dafydd Gibbon, Time Group Types in Mandarin Syllable Annotations, O-COCOSDA, Shanghai, 2015.

Gibbon, Dafydd and Jue Yu. Time Group Analyzer: Methodology And Implementation. The Phonetician 111/112:9-34, 2015.

Isochrony Models of Rhythm: 1D, 2D and 3D

Annotation Mining:

Exploiting Labels and their Time-stamps

1D, 2D and 3D Annotation Mining (Labels + Time-stamps)

Annotation with labels and time stamps: overview

1. Heuristic annotation based approaches
   - rhythm: the truth – but not the whole truth
2. Annotation: event property + time stamps
3. Annotation mining: information extraction from annotations
4. Rhythm definition:

   similarity + isochrony + alternation
5. **1D dispersion measures: duration variability**
6. **2D area measures: duration quadrant**
7. **3D hierarchical analysis:**
   - Time Tree Analysis – induction of duration graphs

One-dimensional Annotation Mining

One-dimensional Annotation Mining (Labels + Time-stamps)

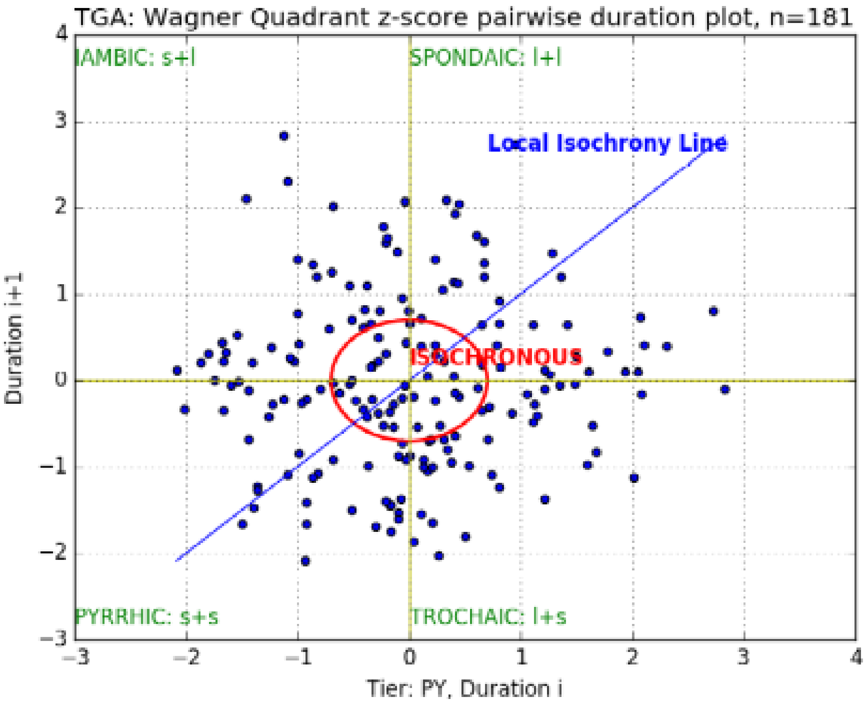
$Variance(x_{1\ldots n}) = \frac{\sum_1^n (x_i - \bar{x})^2}{n-1}$	
$PIM(x_{1\ldots n}) = \sum_{i \neq j} \left| \log \frac{I_i}{I_j} \right|$	where $I_{i,j}$ are intervals in a given sequence
$PFD(d_{1\ldots n}) = \frac{\sum_{i=1}^n \lvert \bar{d} - d_i \rvert}{\sum_{j=1}^n d_j} \times 100$	where d is typically the duration of a *foot*
$nPVI(d_{1\ldots n}) = \frac{\sum_{k=1}^{k-1} \frac{\lvert d_k - d_{k+1} \rvert}{(d_k + d_{k+1})/2}}{n-1} \times 100$	d refers to duration of vocalic segment, syllable or foot, typically

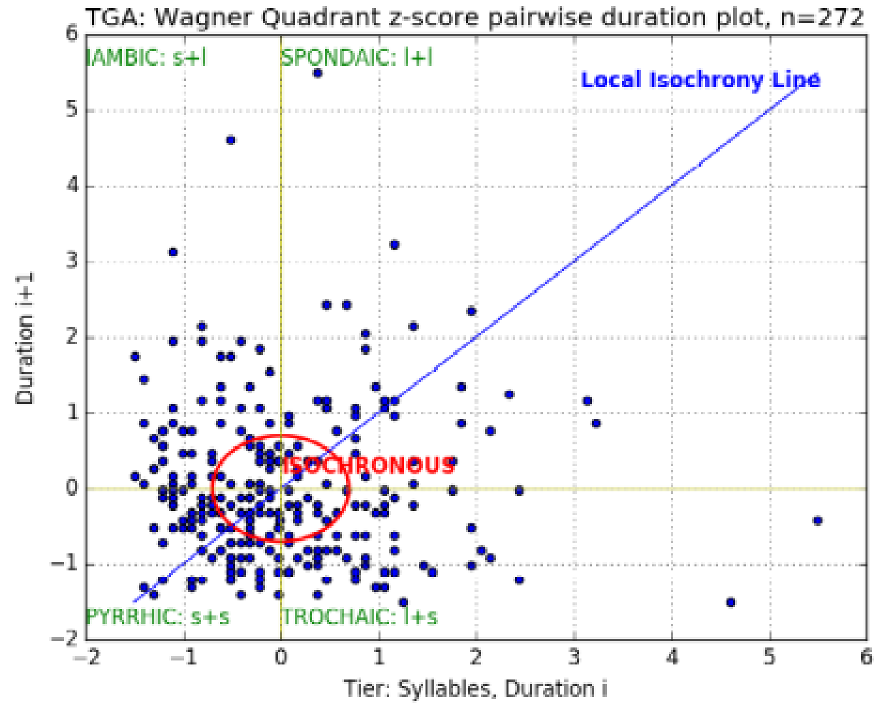
1-dimensional time-stamp duration analysis:

- scales of averages of
    - sequences (Var, PIM, PFD) – no compensation from tempo change
    - pairs (PVI) – abstracts away from tempo change
- no account of rhythm as an alternation relation
- only binary relations

Two-dimensional Annotation Mining

Two-dimensional Annotation Mining (Labels + Time-stamps)





Wagner, Petra (2007). “Visualizing levels of rhythmic organisation.” *Proc. International Congress of Phonetic Sciences, Saarbrücken 2007*, pp. 1113-1116, 2007

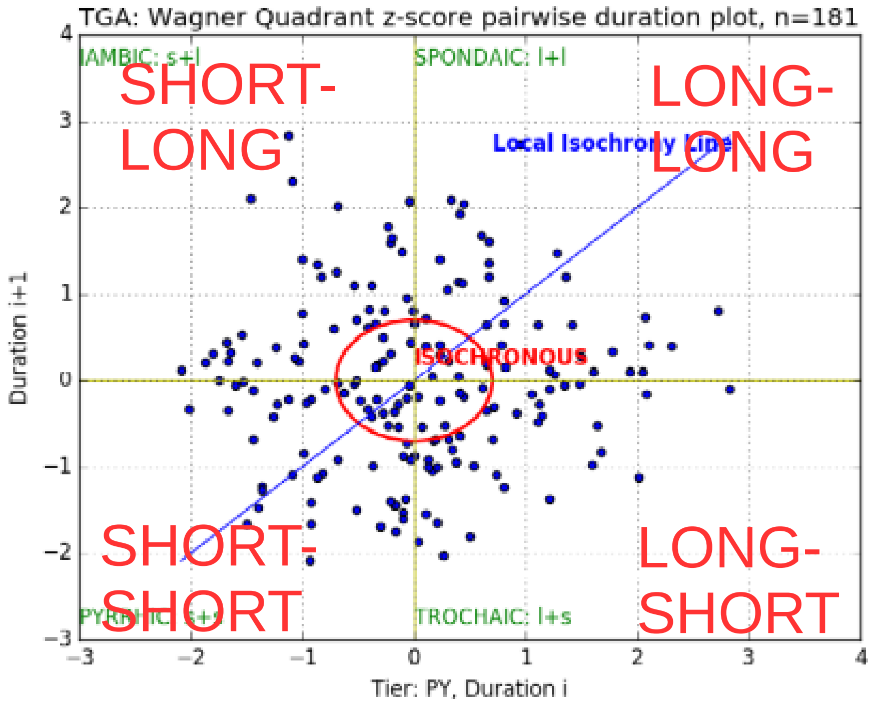
2-dimensional time-stamp duration analysis:

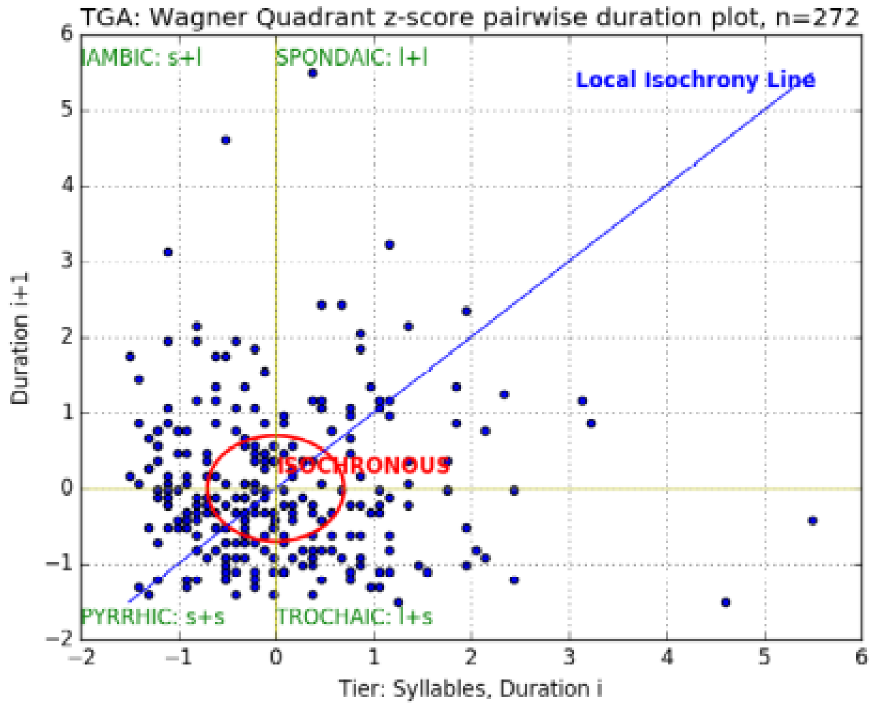
- classification of alternation relations in z-scored scatter plot
  - means: zero
  - x-axis: durations; y-axis: duration of next neighbour
  - long: positive, longer than average; short: negative, shorter than average

Mandarin: means scattered relatively evenly around the centre

English: highly skewed: |short+short| >> |long+long|
majority or relations: <u>non-binary</u>

Two-dimensional Annotation Mining (Labels + Time-stamps)





Wagner, Petra (2007). “Visualizing levels of rhythmic organisation.” *Proc. International Congress of Phonetic Sciences, Saarbrücken 2007*, pp. 1113-1116, 2007

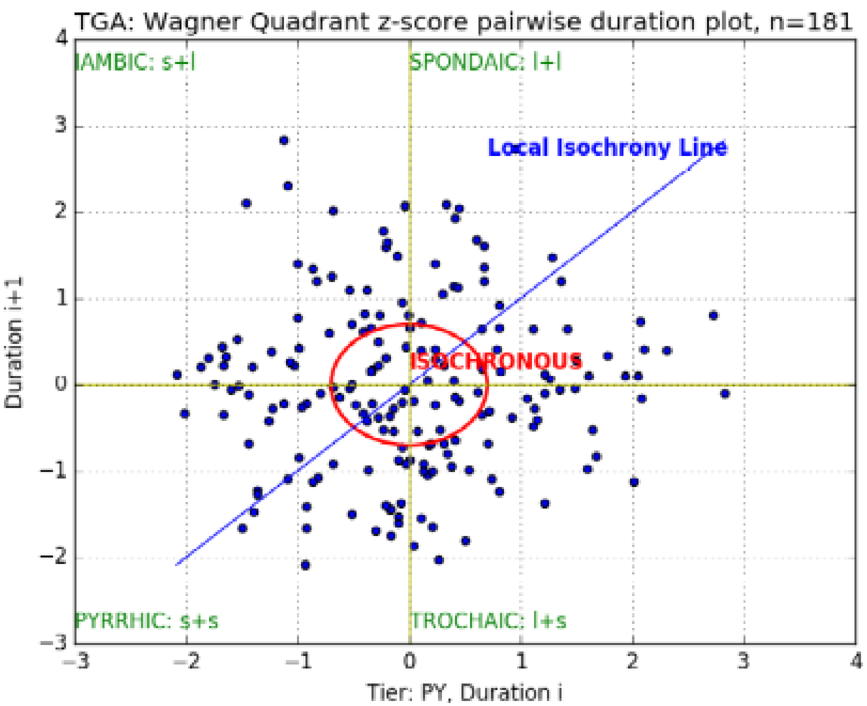
2-dimensional time-stamp duration analysis:

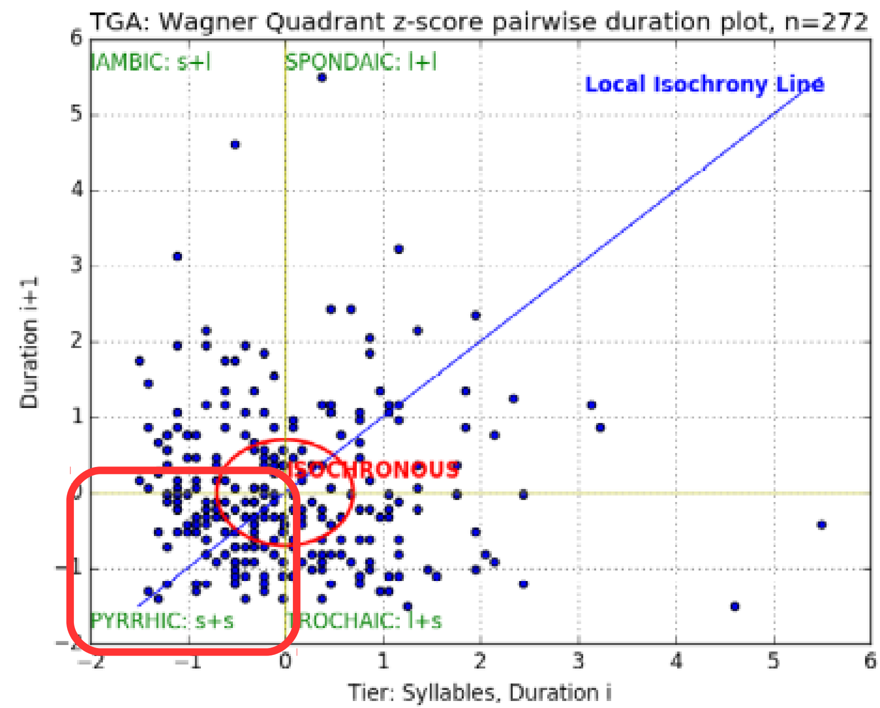
- classification of alternation relations in z-scored scatter plot
  - means: zero
  - x-axis: durations; y-axis: duration of next neighbour
  - long: positive, longer than average; short: negative, shorter than average

Mandarin: means scattered relatively evenly around the centre

English: highly skewed: |short+short| >> |long+long|
majority or relations: non-binary

One-dimensional Annotation Mining (Labels + Time-stamps)





Wagner, Petra (2007). "Visualizing levels of rhythmic organisation." *Proc. International Congress of Phonetic Sciences, Saarbrücken 2007*, pp. 1113-1116, 2007

2-dimensional time-stamp duration analysis:

- classification of alternation relations in z-scored scatter plot
  - means: zero
  - x-axis: durations; y-axis: duration of next neighbour
  - long: positive, longer than average; short: negative, shorter than average

Mandarin: means scattered relatively evenly around the centre
English: highly skewed: |short+short| >> |long+long|
majority or relations: non-binary

Three-dimensional Annotation Mining

(more like 2.5 dimensional)

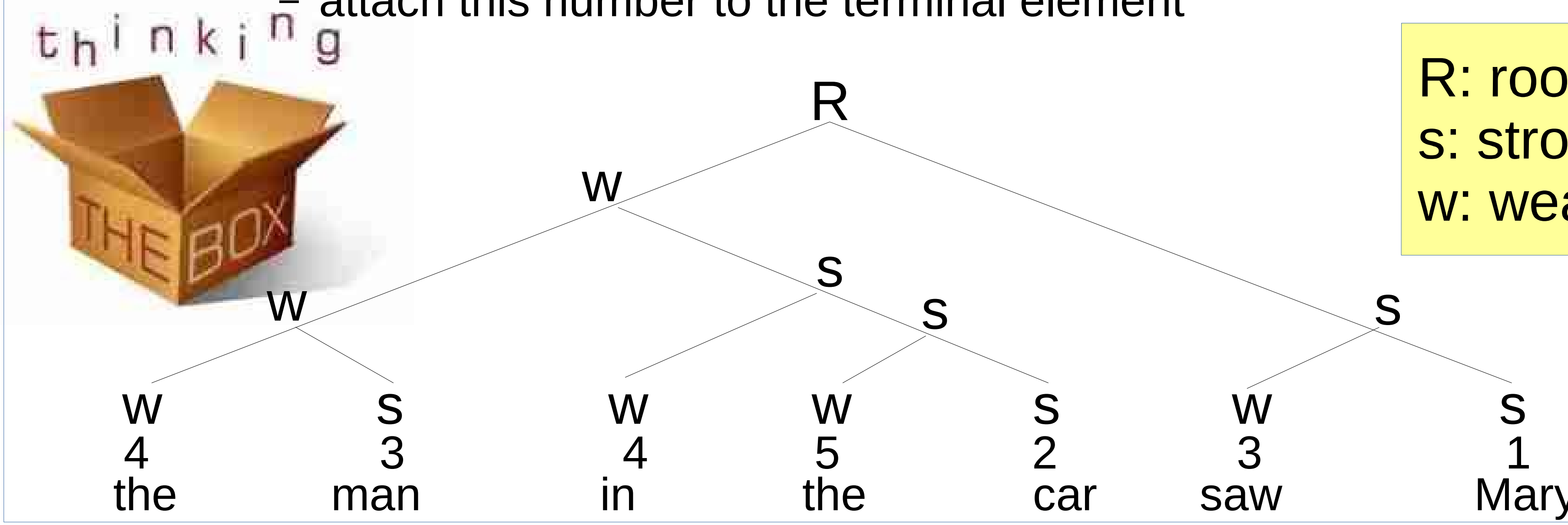
3-Dimensional Models of Timing Relations: Gibbon Time Trees

1. Hypothesis in Generative and Metrical Phonologies:
   - Prominence follows the stress hierarchy
2. Liberman's version of the Nuclear Stress Rule (1976):

   label a sentence tree with "w" and "s" nodes ("weak", "strong")

   for each terminal element of the tree:

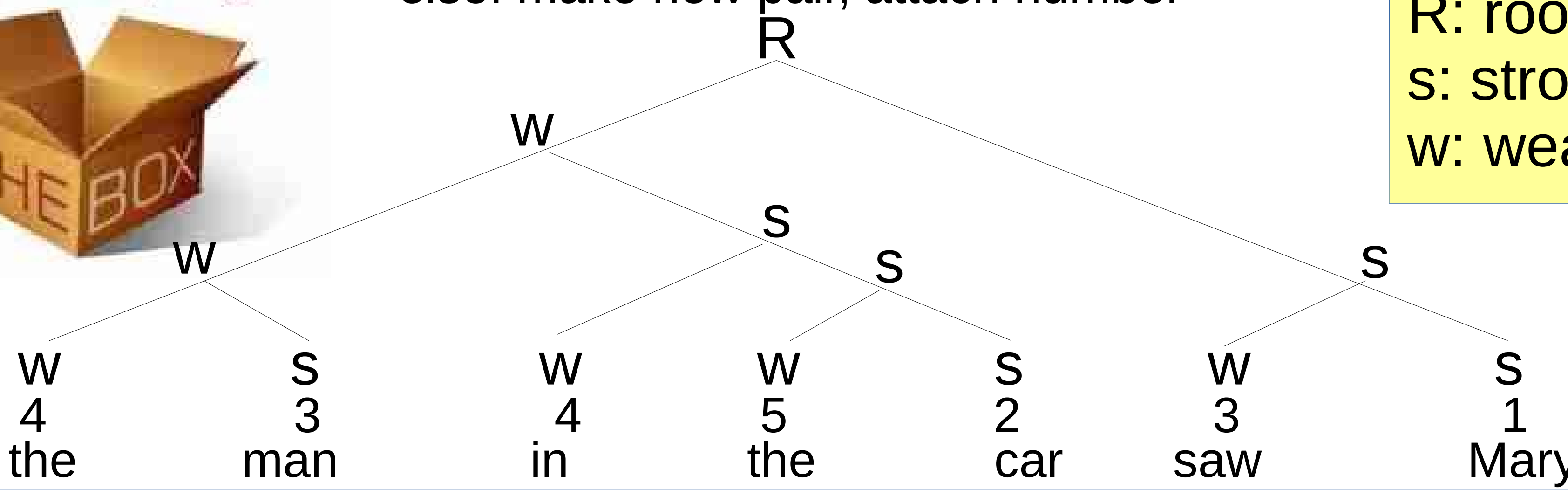
   move up the branch from this element
   - look for the first "w" node
   - count the number of nodes from the first "w" through "R"
   - attach this number to the terminal element



R: root
s: strong
w: weak

3-Dimensional Models of Timing Relations: Gibbon Time Trees

1. Inverse hypothesis: Stress hierarchy follows prominence
2. Gibbon (2003), Time Trees:
   - label a sequence of items with numbers (e.g. durations)
   - create an empty store (stack)
   - for number-word pair in sequence:
     - if left pair < right pair: store left pair and continue with sequence
     - else: join into new larger pair, attach right number
       - if store is not empty, for item in store:
         - if store item < current item: break, continue sequence
         - else: make new pair, attach number



R: root
s: strong
w: weak

Phonological Tree Induction

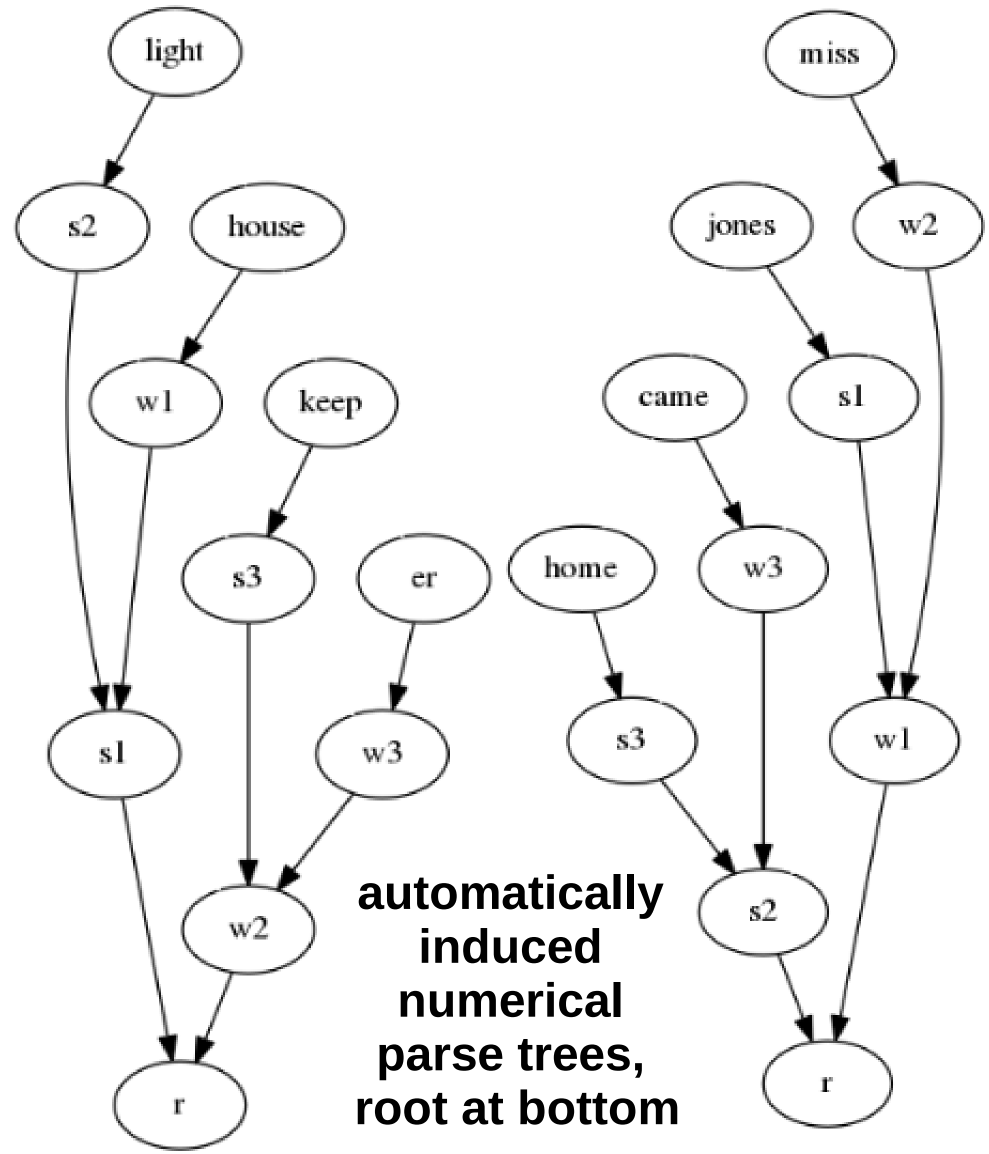
Inverses of Chomsky & Halle’s or Liberman’s metrical generation algorithms (Compound and Nuclear Stress Rules)

Inductive input-output relation (example stereotypes), number-word pair sequence to strong-weak node pair hierarchy:

Iambic (*weak-strong*) *directionality, iNSR*:
((miss . **3**) (jones . **2**) (came . **3**) (home . **1**))
→ (r (w (w miss) (s jones)) (s (w came) (s home)))

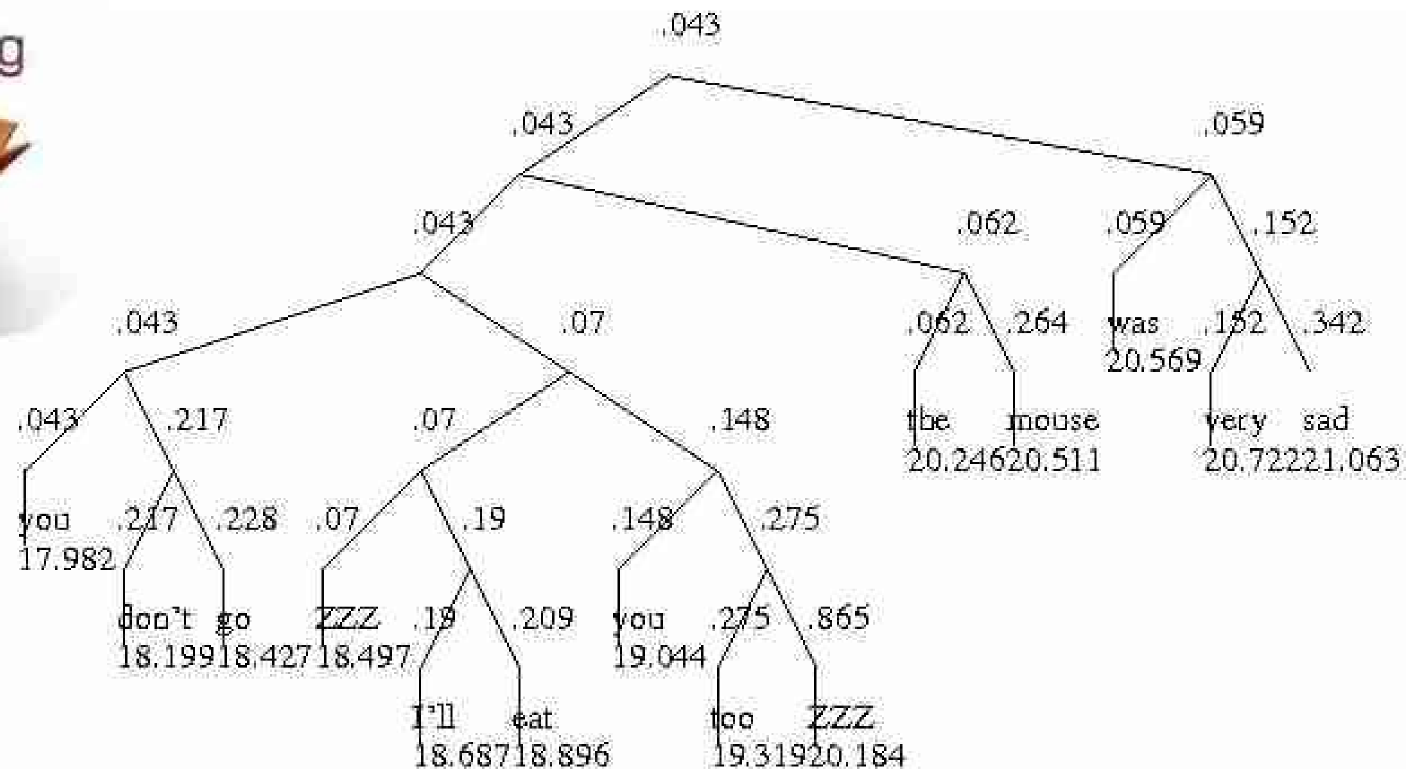
Trochaic (*strong-weak*) *directionality, iCSR*:
((light . **1**) (house . **3**) (keep . **2**) (er . **3**))
→ ((r (s (s light) (w house)) (w (s keep) (w er))))

Implemented in Scheme



Gibbon, Dafydd. 2006. “Time types and time trees: Prosodic mining and alignment of temporally annotated data”. In: Stefan Sudhoff et al., eds. *Methods in Empirical Prosody Research*. Walter de Gruyter, pp. 281–209, 2006.

Two-dimensional Annotation Mining (Labels + Time-stamps)



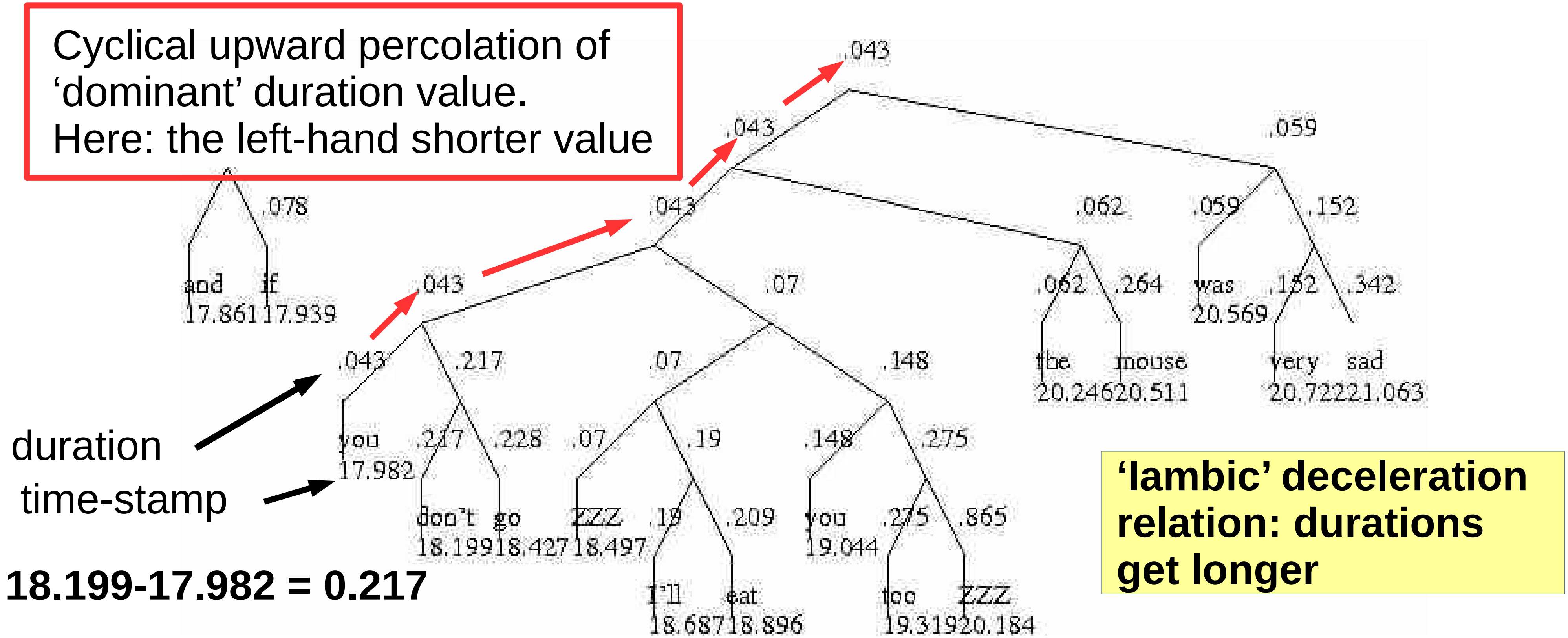
Gibbon, Dafydd. 2006. "Time types and time trees: Prosodic mining and alignment of temporally annotated data". In: Stefan Sudhoff, et al., eds. *Methods in Empirical Prosody Research*. Berlin: Walter de Gruyter, pp. 281–209, 2006.

3-dimensional time-stamp duration analysis:

Time-Tree induction:

- *length* × *depth* with 1-place lookahead (so actually 2D+1):
- hierarchical classification of alternation relations
- several processing options: binary/nonbinary, lower/higher percolated
- related to phrasal and discourse patterns

Three-dimensional Annotation Mining (Labels + Time-stamps)



Gibbon, Dafydd. 2006. “Time types and time trees: Prosodic mining and alignment of temporally annotated data”. In: Stefan Sudhoff, et al., eds. *Methods in Empirical Prosody Research*. Berlin: Walter de Gruyter, pp. 281–209, 2006.

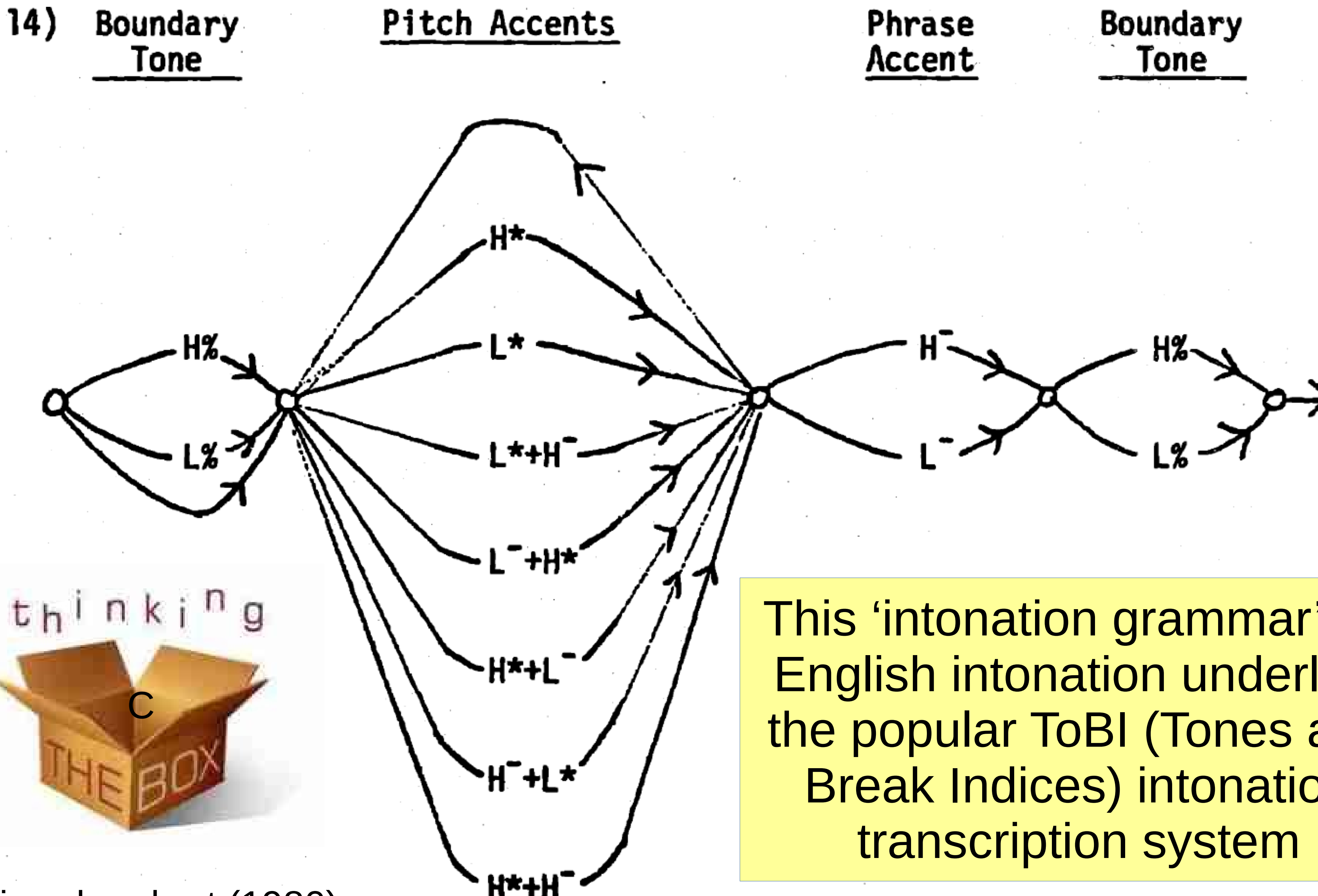
3-dimensional time-stamp duration analysis:

Time-Tree induction:

- *length* × *depth* with 1-place lookahead (so actually 2D+1):
- hierarchical classification of alternation relations
- several processing options: binary/nonbinary, lower/higher percolated
- related to phrasal and discourse patterns

The phonological basis of rhythm: ‘abstract oscillation’

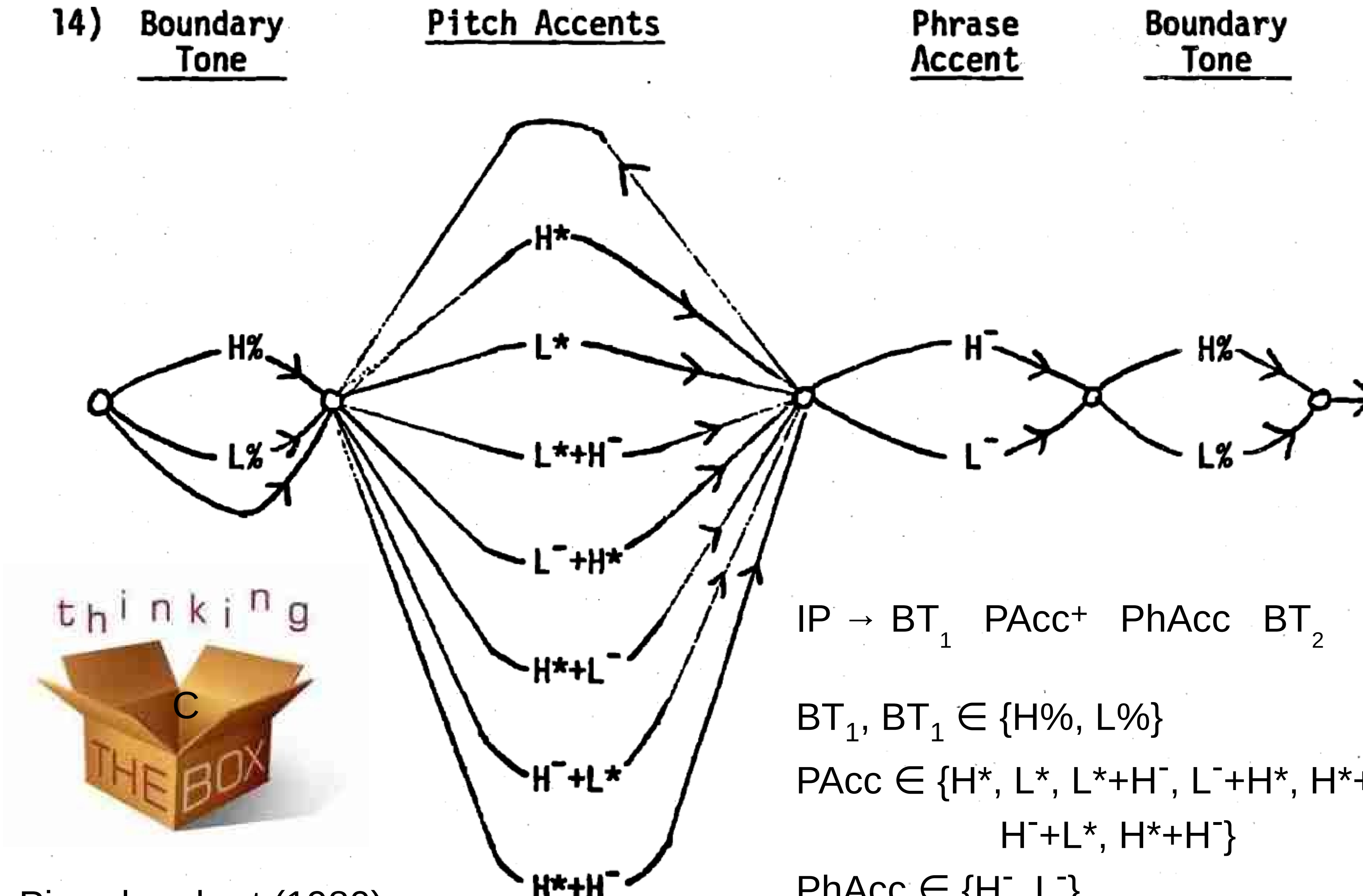
Pierrehumbert's Finite Machine Model: an 'abstract oscillator'



Pierrehumbert (1980)

This 'intonation grammar' for English intonation underlies the popular ToBI (Tones and Break Indices) intonation transcription system

Pierrehumbert's Finite Machine Model: an 'abstract oscillator'



Pierrehumbert (1980)

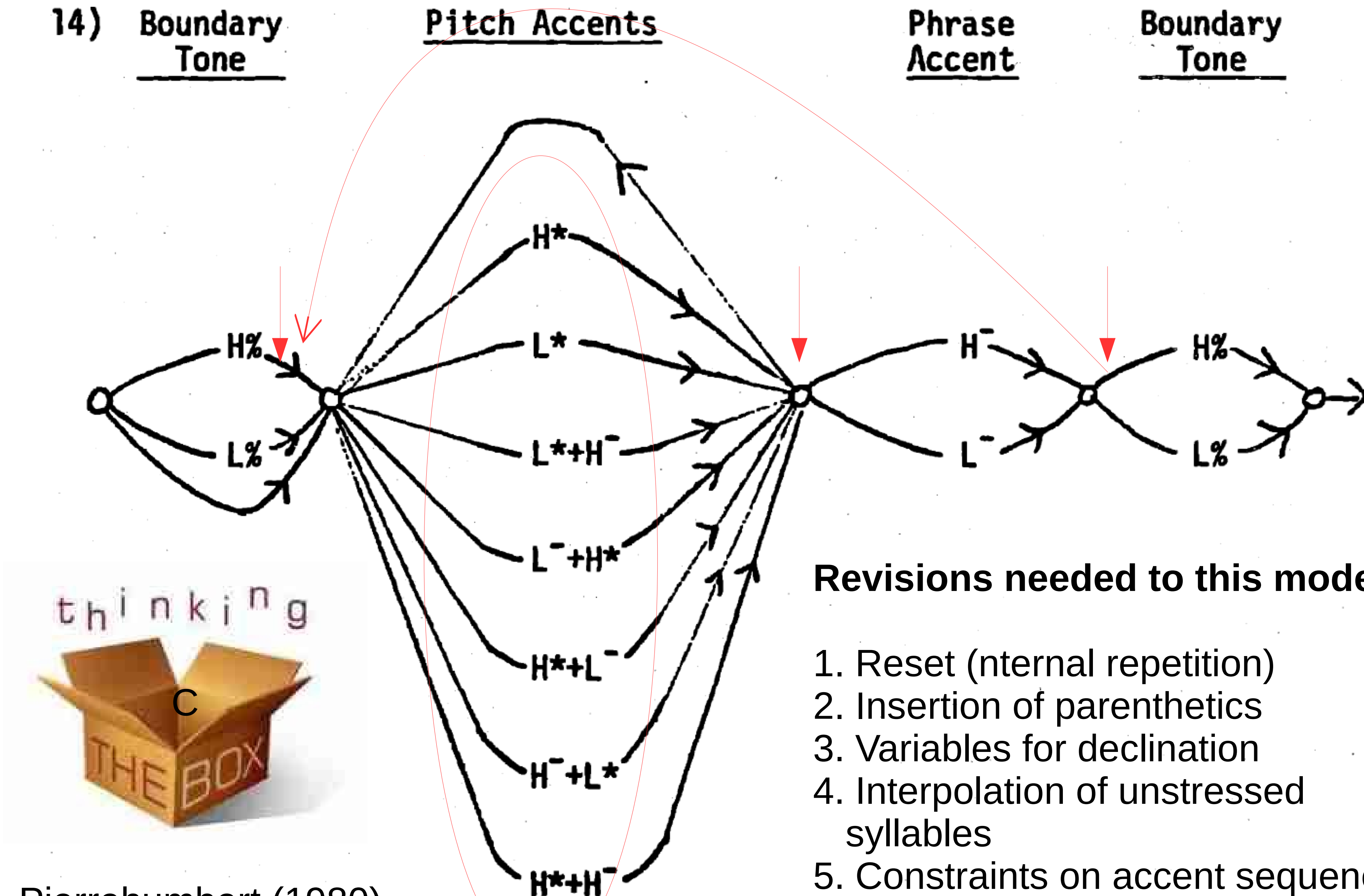
$IP \rightarrow BT_1 \quad PAcc^{+} \quad PhAcc \quad BT_2$

$BT_1, BT_1 \in \{H\%, L\%\}$

$PAcc \in \{H^*, L^*, L^*{+}H^{-}, L^{-}{+}H^*, H^*{+}L^{-}, H^{-}{+}L^*, H^*{+}H^{-}\}$

$PhAcc \in \{H^{-}, L^{-}\}$

Pierrehumbert’s Finite Machine Model: an ‘abstract oscillator’

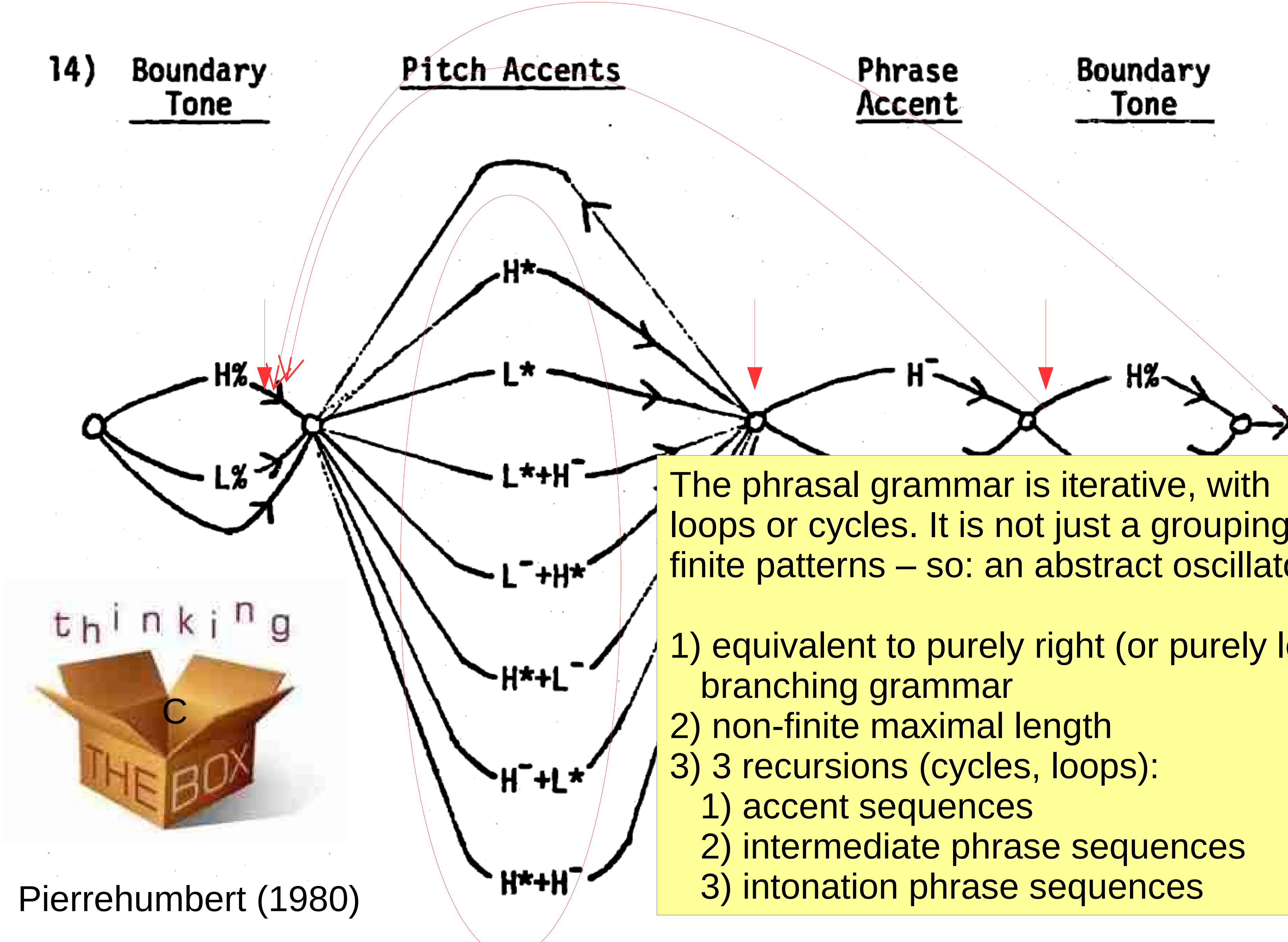


Pierrehumbert (1980)

Revisions needed to this model:

1. Reset (nternal repetition)
2. Insertion of parenthetics
3. Variables for declination
4. Interpolation of unstressed syllables
5. Constraints on accent sequences

Pierrehumbert's Finite Machine Model: an 'abstract oscillator'



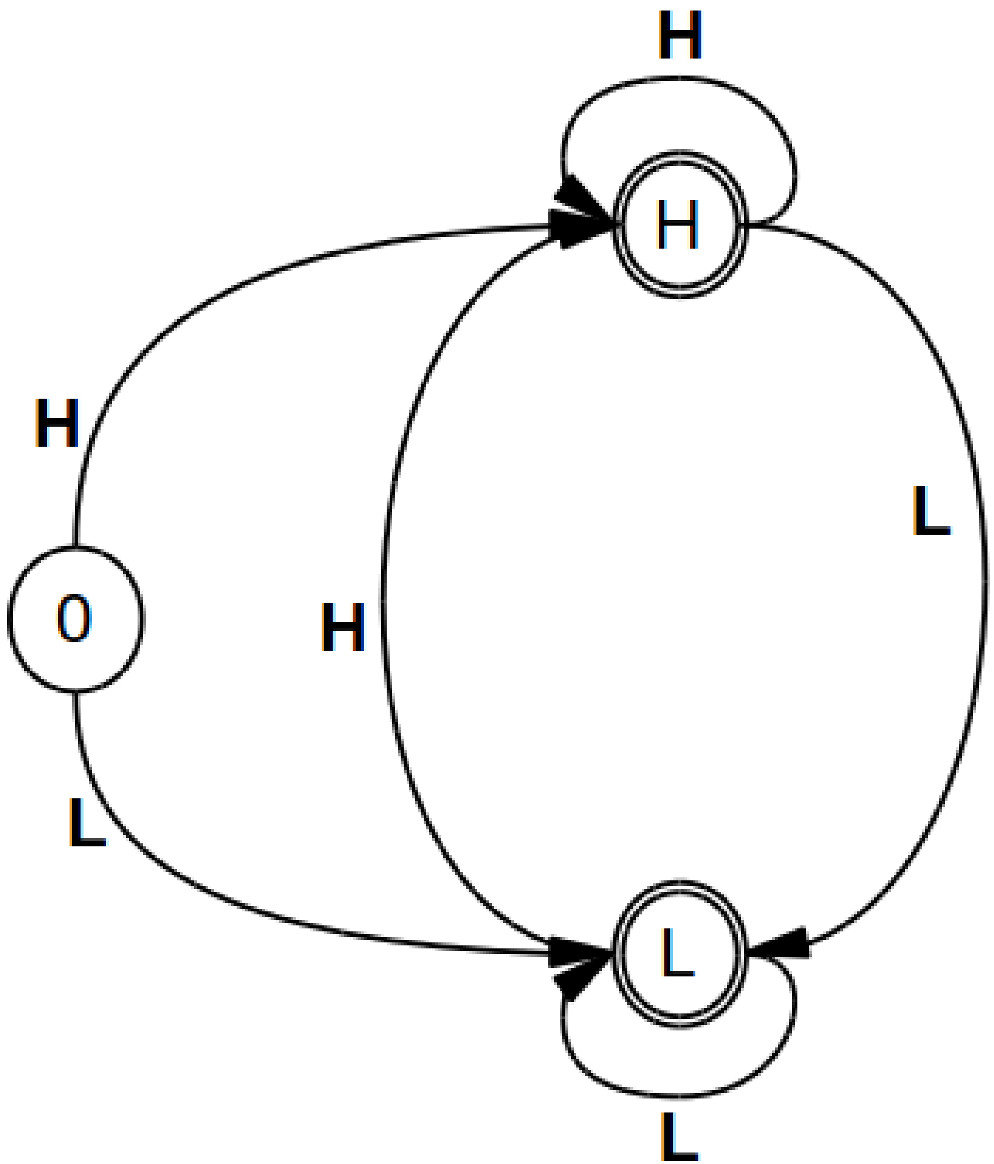
Pierrehumbert (1980)

The phrasal grammar is iterative, with loops or cycles. It is not just a grouping of finite patterns – so: an abstract oscillator

1) equivalent to purely right (or purely left) branching grammar
2) non-finite maximal length
3) 3 recursions (cycles, loops):
   1) accent sequences
   2) intermediate phrase sequences
   3) intonation phrase sequences

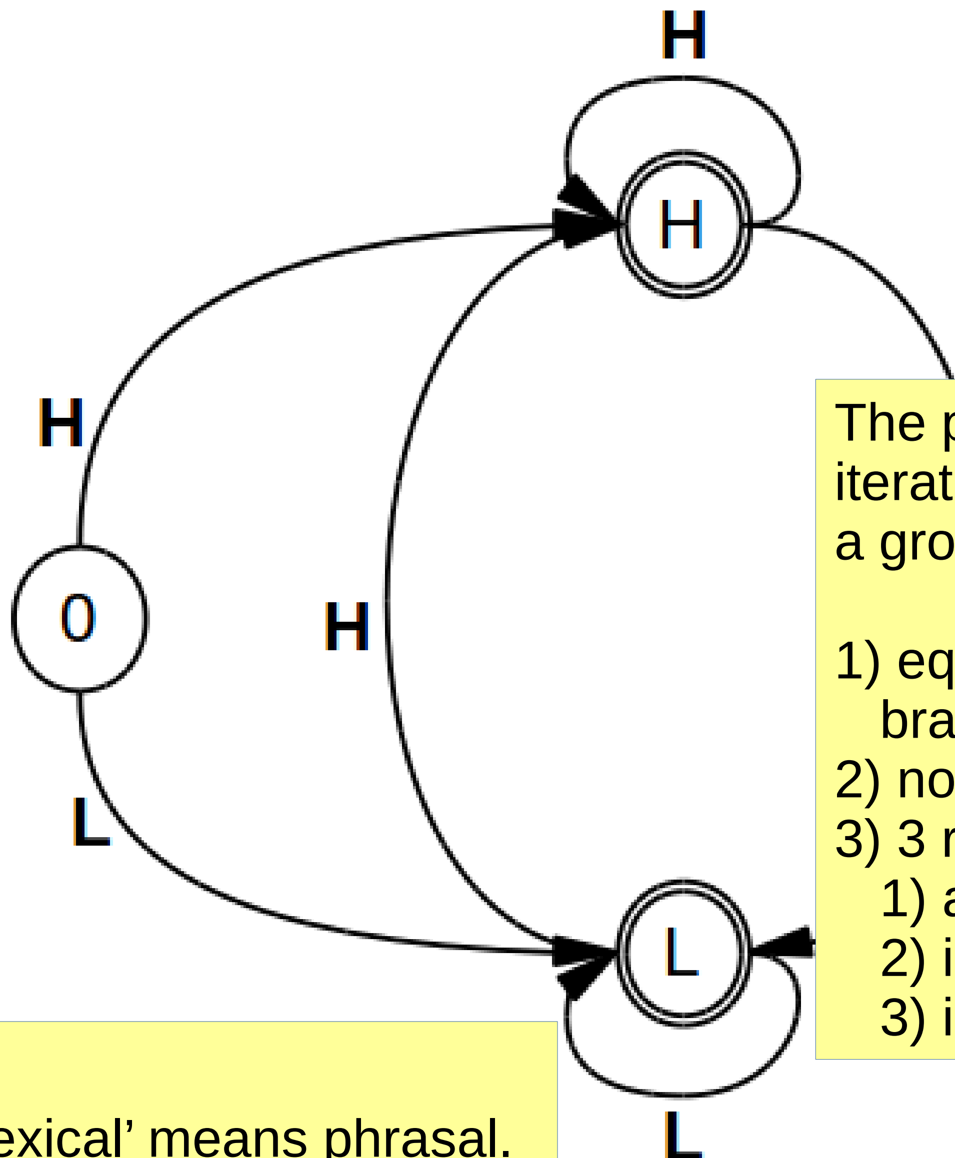
Abstract Oscillator: Niger-Congo Languages with 2 Lexical Tones

1-tape (1-level) transition network



Abstract Oscillator: Niger-Congo Languages with 2 Lexical Tones

1-tape (1-level) transition network



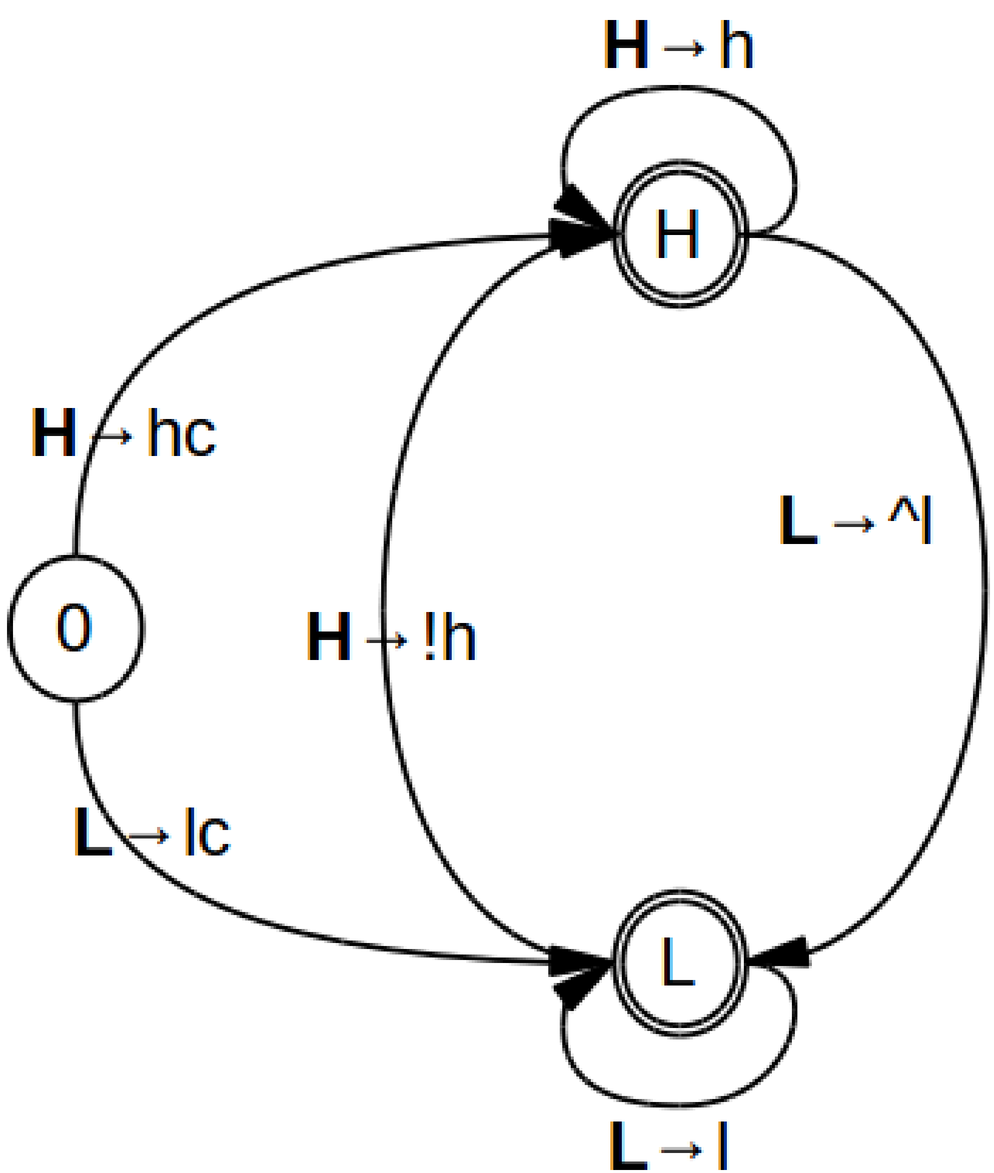
The phrasal tone-sandhi grammar is iterative, with loops or cycles. It is not just a grouping of finite patterns:

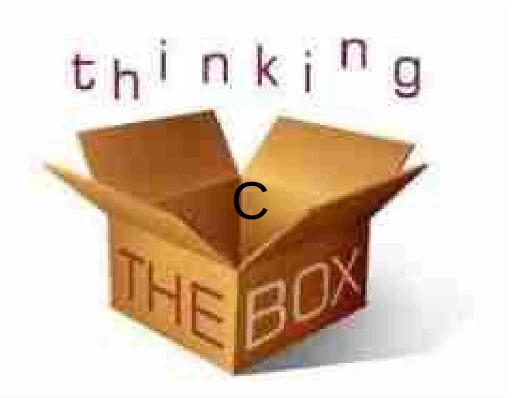
1) equivalent to purely right (or purely left) branching grammar
2) non-finite maximal length
3) 3 recursions (cycles, loops):
   1) accent sequences
   2) intermediate phrase sequences
   3) intonation phrase sequences

Note:
'post-lexical' means phrasal.

Abstract Oscillator: Niger-Congo Languages with 2 Lexical Tones

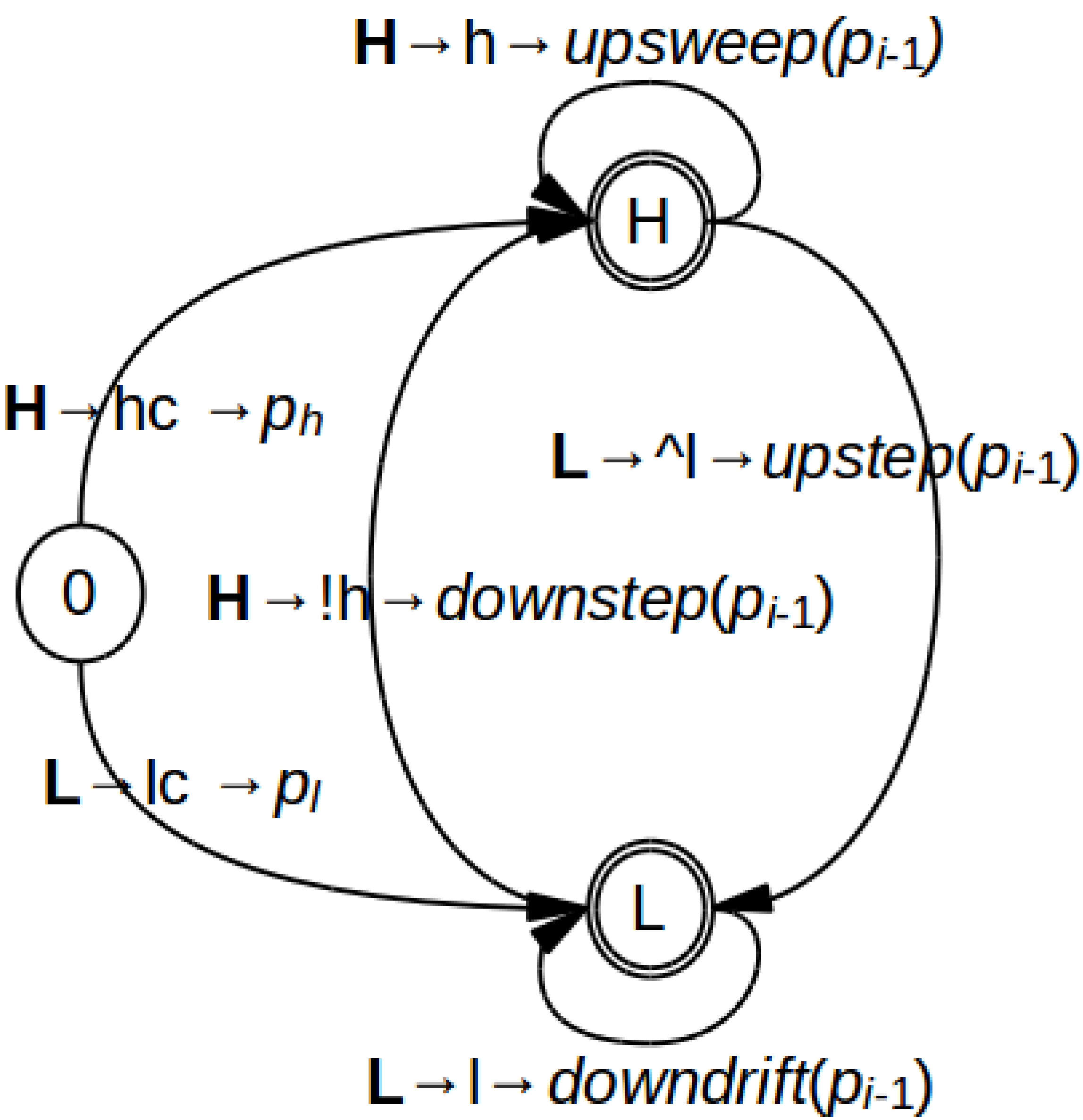
2-tape (2-level) transition network





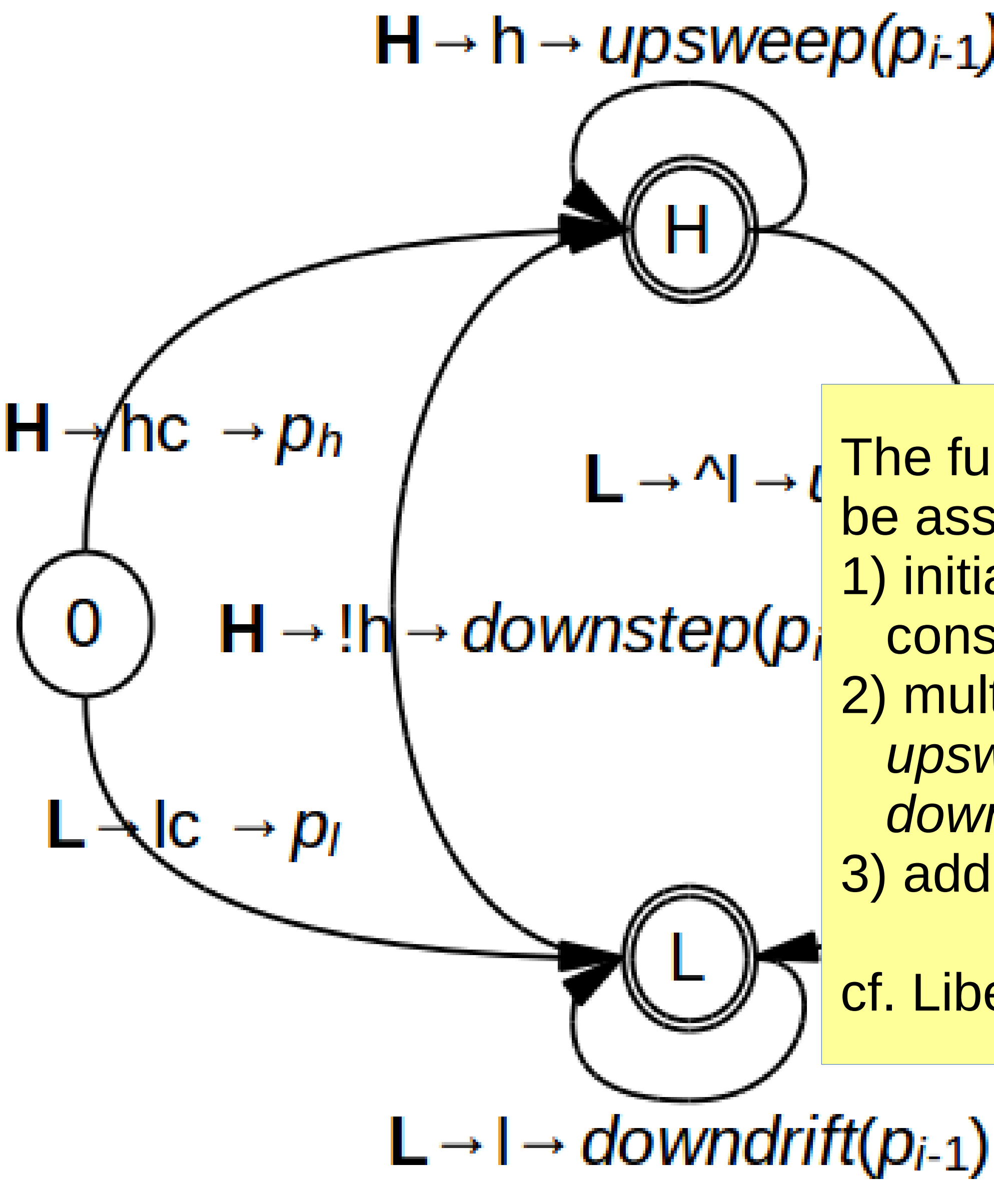
Abstract Oscillator: Niger-Congo Languages with 2 Lexical Tones

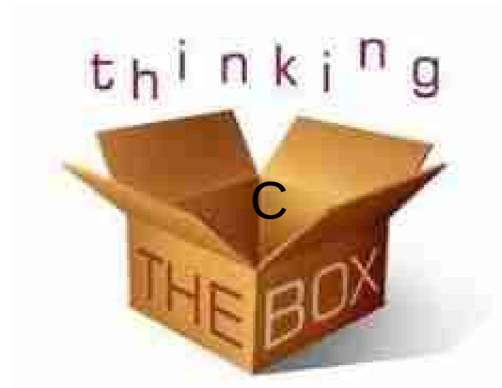
3-tape (3-level) transition network



Abstract Oscillator: Niger-Congo Languages with 2 Lexical Tones

3-tape (3-level) transition network





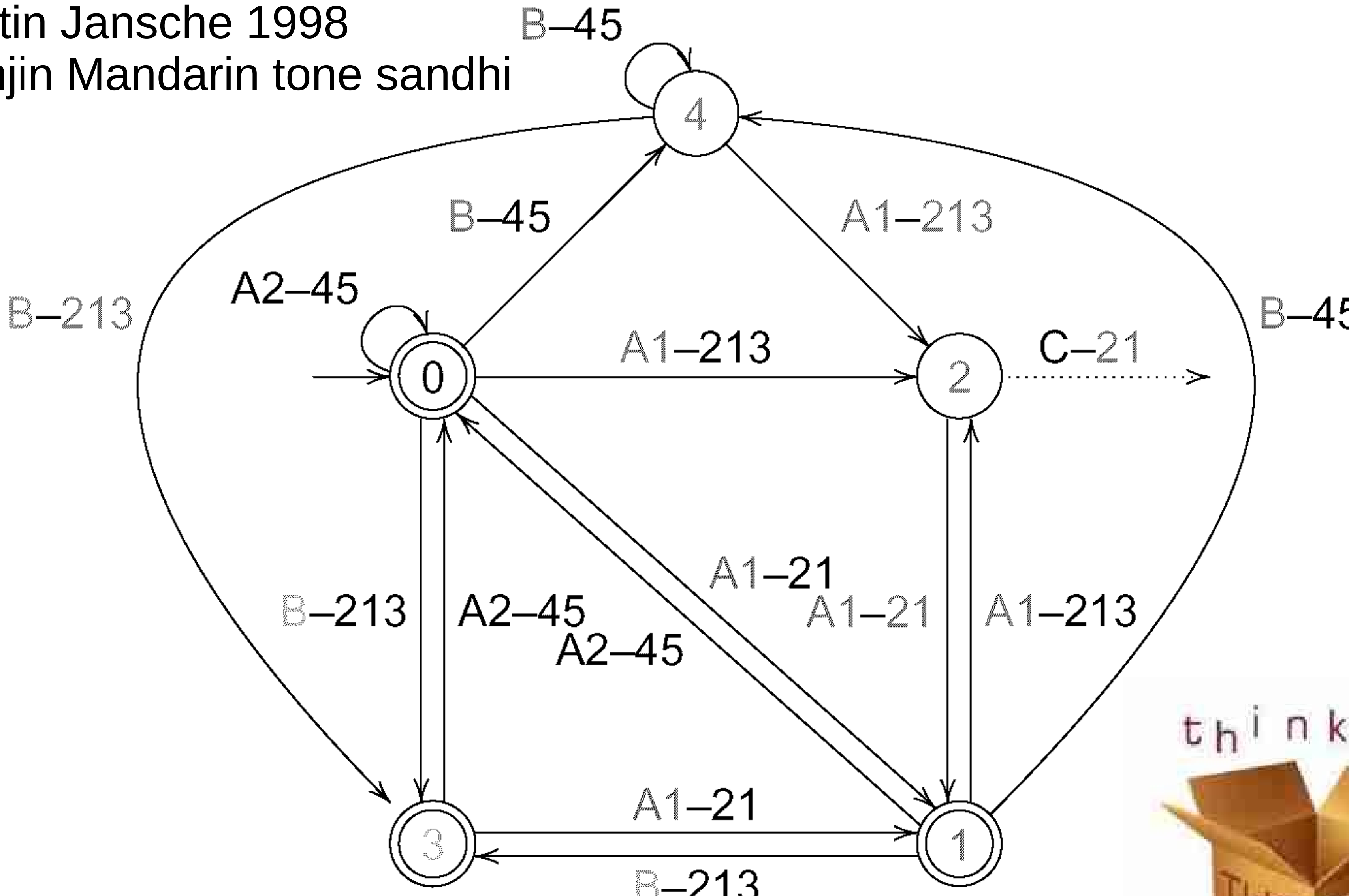
The functions on the third level can easily be assigned numerical values:

1) initial ‘start-up’ high or low fuzzy pitch constant
2) multiplication of previous value by an *upsweep*, *downdrift*, *upstep*, or *downstep* value
3) addition of a baseline value

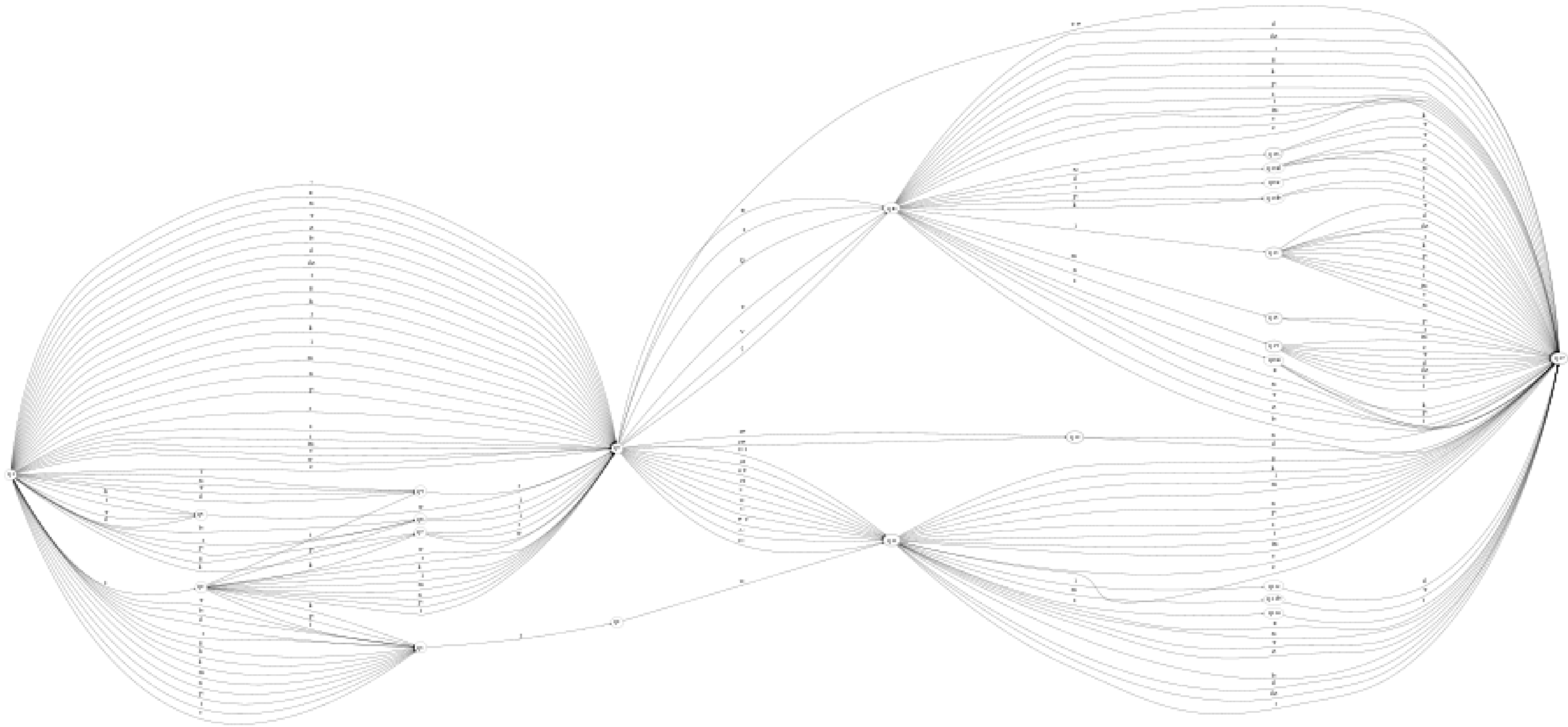
cf. Liberman & Pierrehumbert (e.g. 1984)

Abstract Oscillator: Tianjin Mandarin Tone Sandhi

Martin Jansche 1998
Tianjin Mandarin tone sandhi

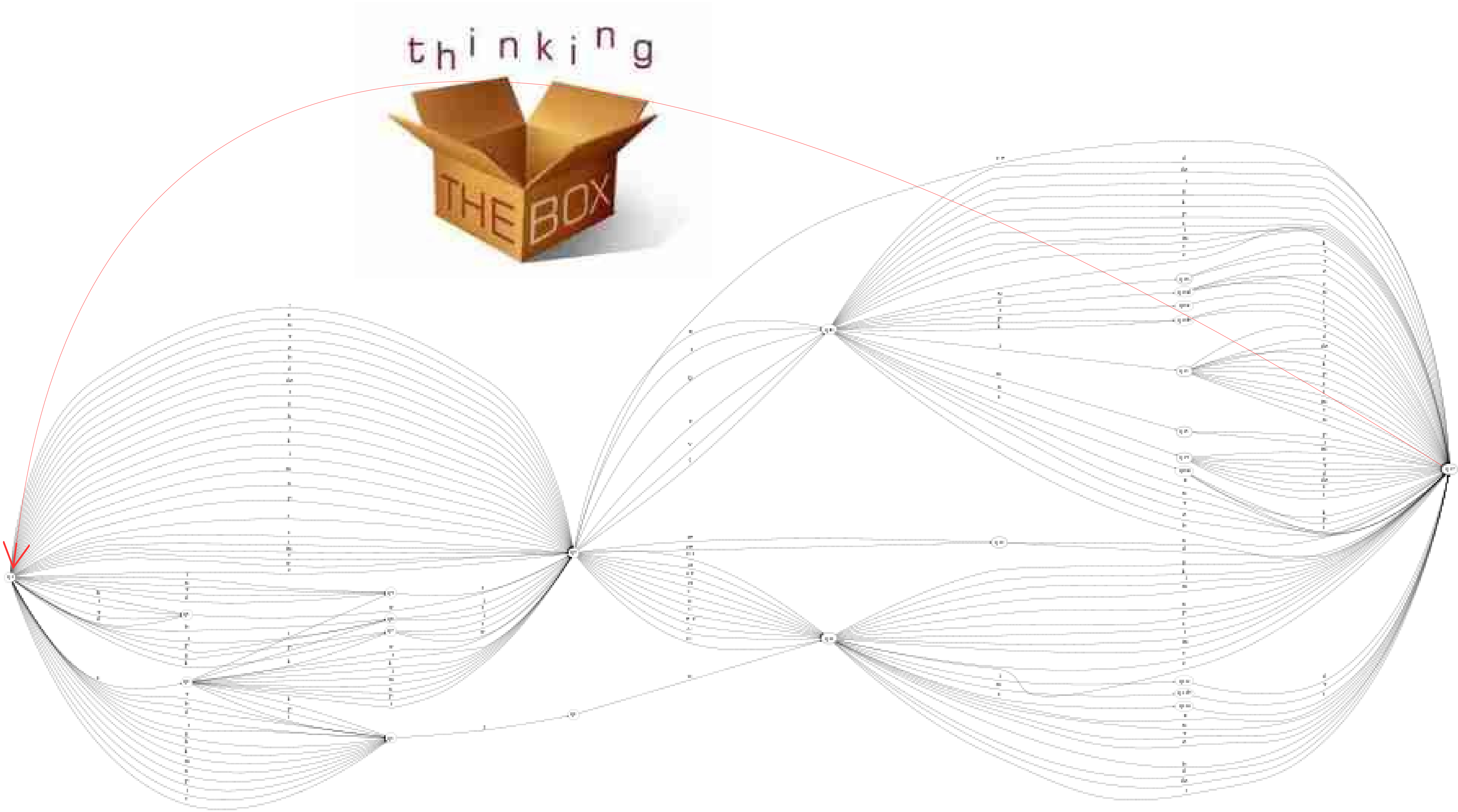


The Basis of English Rhythm: the Syllable ‘Abstract Oscillator’



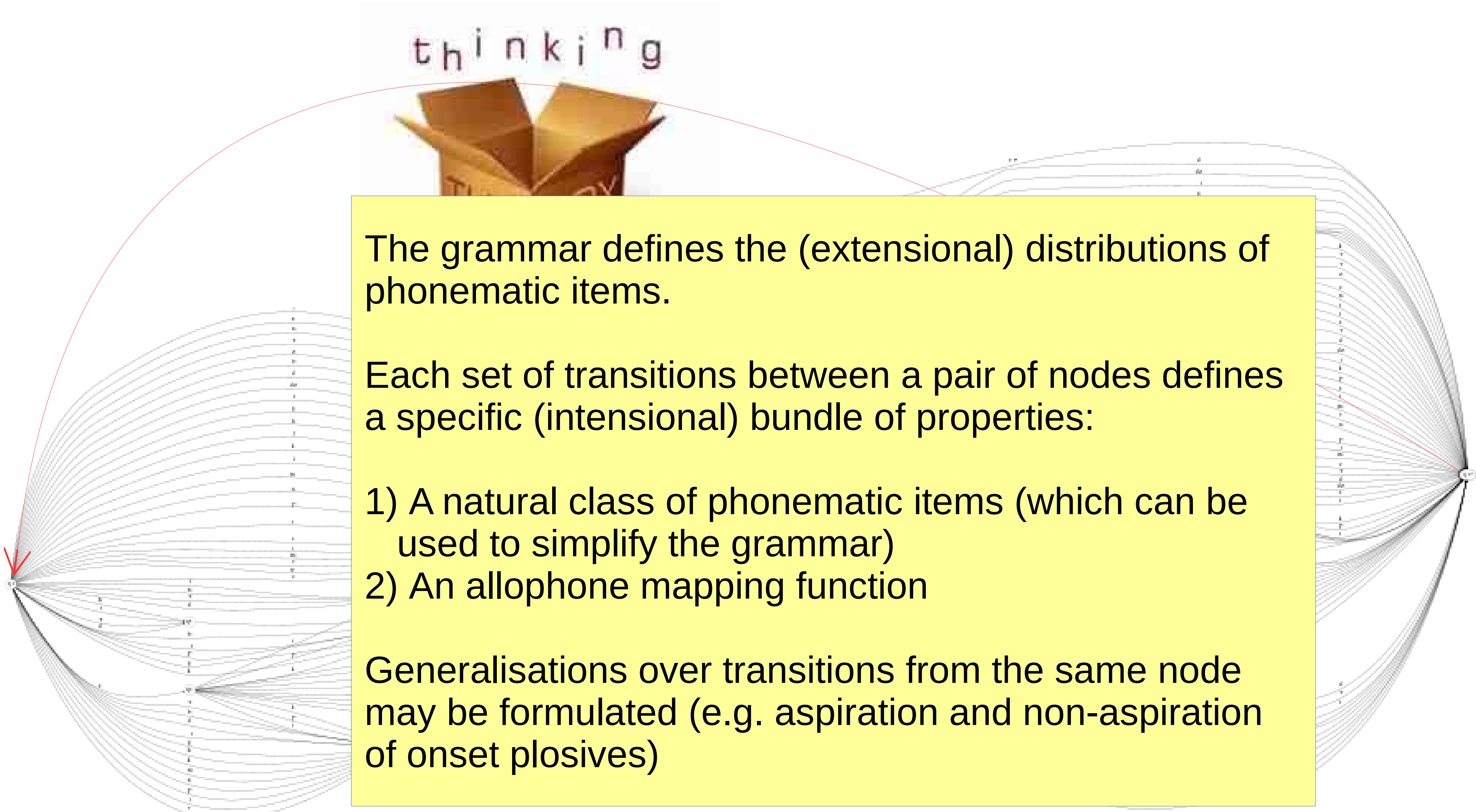
Linear Syllable Grammar for English

The Basis of English Rhythm: the Syllable ‘Abstract Oscillator’



Linear Syllable Grammar for English

The Basis of English Rhythm: the Syllable 'Abstract Oscillator'



The grammar defines the (extensional) distributions of phonematic items.

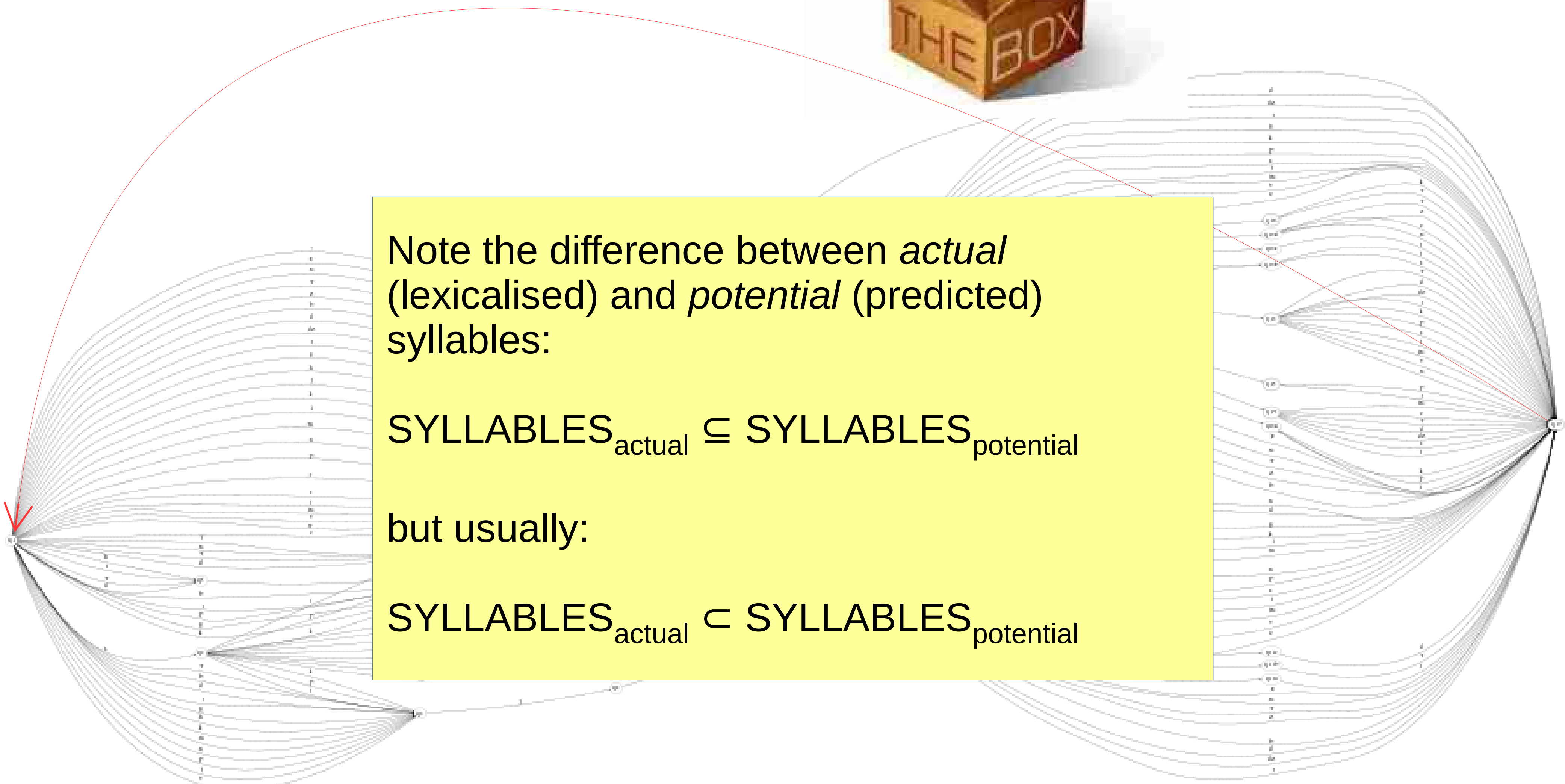
Each set of transitions between a pair of nodes defines a specific (intensional) bundle of properties:

1) A natural class of phonematic items (which can be used to simplify the grammar)
2) An allophone mapping function

Generalisations over transitions from the same node may be formulated (e.g. aspiration and non-aspiration of onset plosives)

Linear Syllable Grammar for English

The Basis of English Rhythm: the Syllable 'Abstract Oscillator'



Note the difference between *actual* (lexicalised) and *potential* (predicted) syllables:

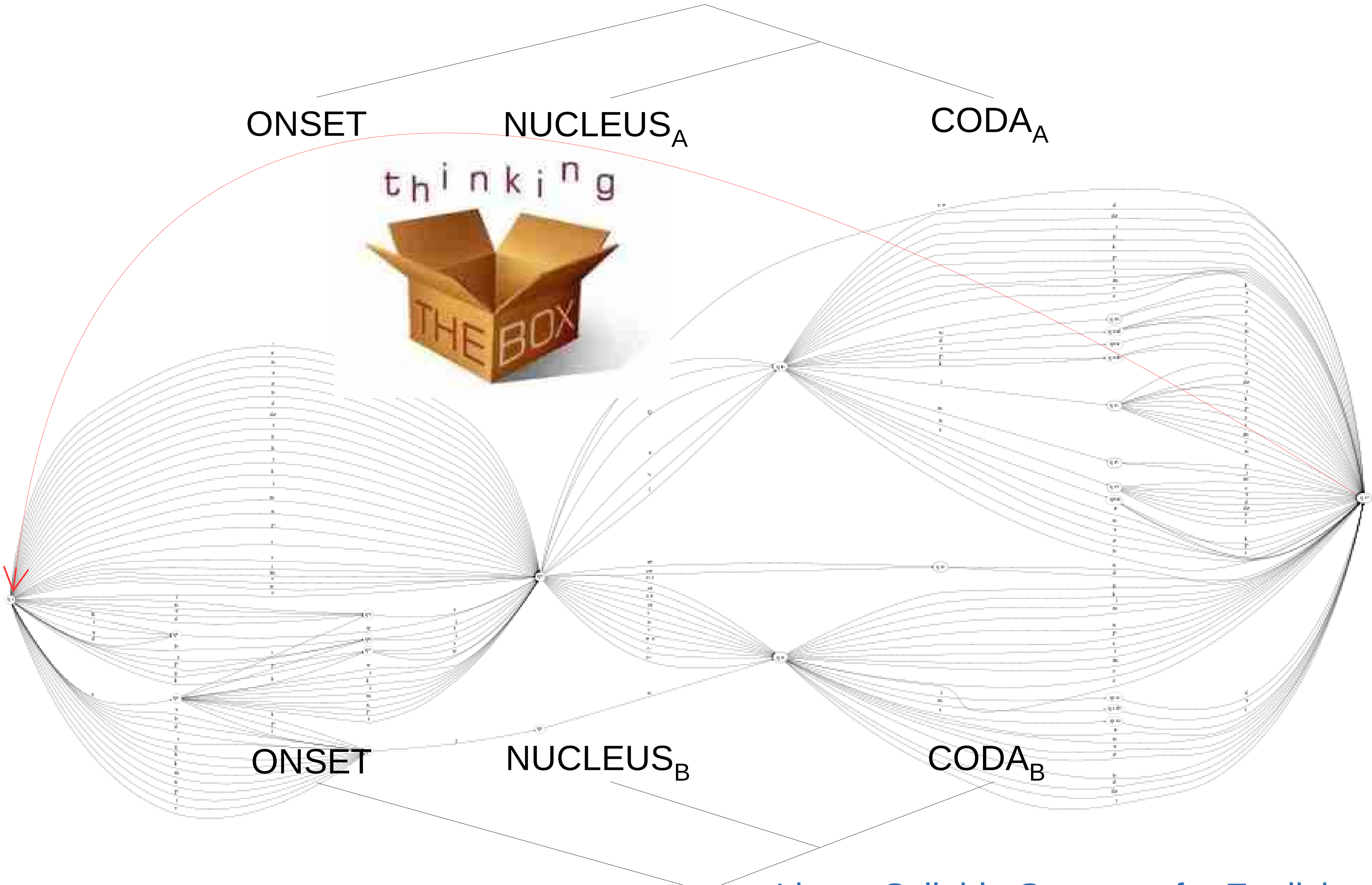
$$\text{SYLLABLES}_{\text{actual}} \subseteq \text{SYLLABLES}_{\text{potential}}$$

but usually:

$$\text{SYLLABLES}_{\text{actual}} \subset \text{SYLLABLES}_{\text{potential}}$$

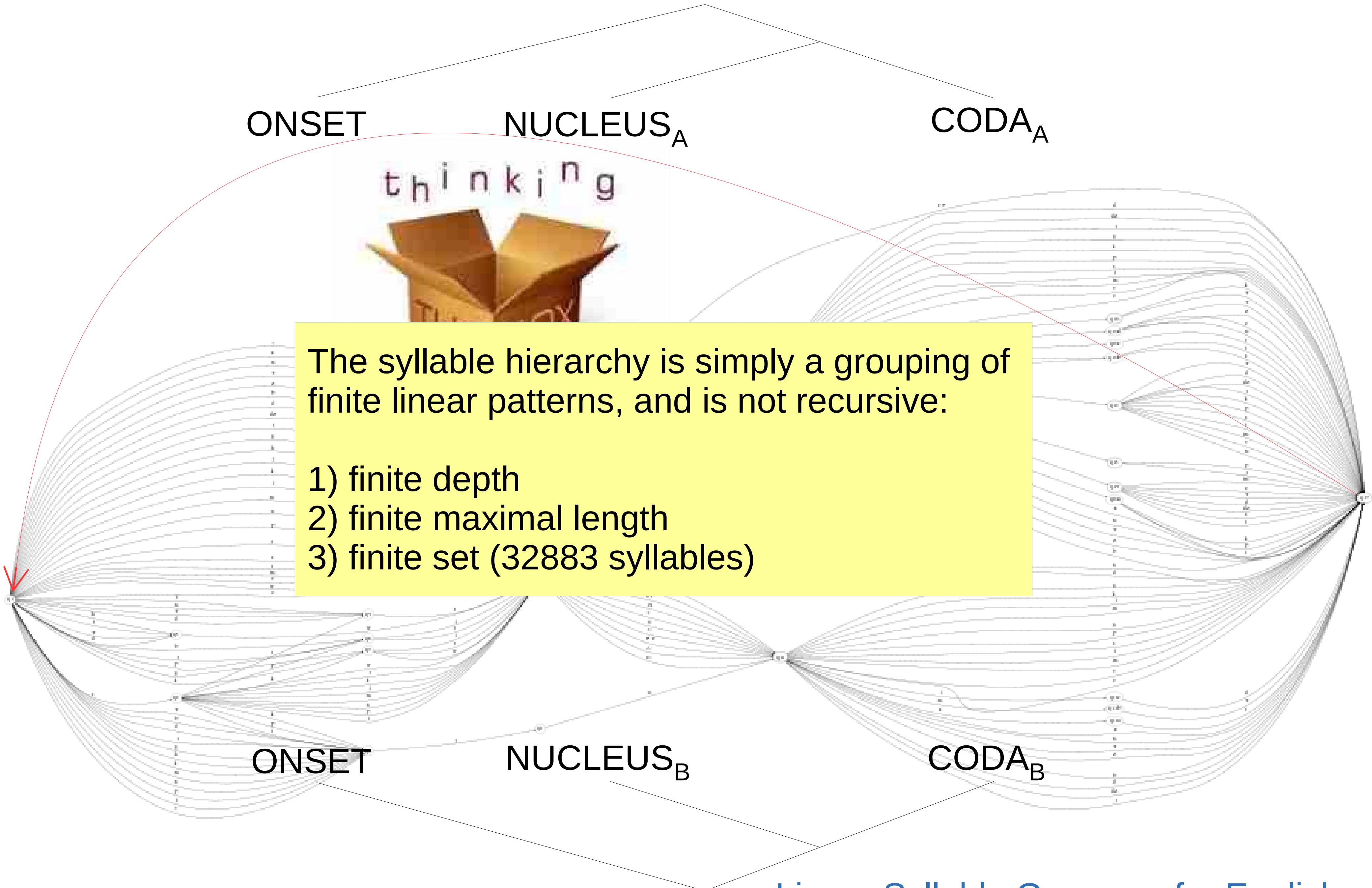
Linear Syllable Grammar for English

The Basis of English Rhythm: the Syllable 'Abstract Oscillator'



Linear Syllable Grammar for English

The Basis of English Rhythm: the Syllable 'Abstract Oscillator'



The syllable hierarchy is simply a grouping of finite linear patterns, and is not recursive:

1) finite depth
2) finite maximal length
3) finite set (32883 syllables)

Linear Syllable Grammar for English

Linear Phrasal Grammar of Mandarin Syllable Phonotactics:

A Computational Perspective

The Basis of Mandarin Rhythm: the Syllable 'Abstract Oscillator'

33 vowels with addition of o and ueng(ong) =35

the missing vowel o is place under uo and ueng under ong

22 consonants including 0 consonant

Pinyin	a	ai	ao	an	ang	ou	ong	e	ei	en	eng	i	ia	iao	ie	iu (iou)	ian	in	iang	ing	iong	u	ua	uo	uai	ui (uei)	uan	un (uen)	uang	ü	üe	üan	ün
ø	a	ai	ao	an	ang	ou	weng	e	ei	en	eng	yi	ya	yao	ye	you	yan	yin	yang	ying	yong	wu	wa	wo	wai	wei	wan	wen	wang	yu	yue	yuan	yun
b	ba	bai	bao	ban	bang				bei	ben	beng	bi		biao	bie		bian	bin		bing		bu		bo									
p	pa	pai	pao	pan	pang	pou			pei	pen	peng	pi		piao	pie		pian	pin		ping		pu		po									
m	ma	mai	mao	man	mang	mou		me	mei	men	meng	mi		miao	mie	miu	mian	min		ming		mu		mo									
f	fa			fan	fang	fou			fei	fen	feng											fu		fo									
d	da	dai	dao	dan	dang	dou	dong	de	dei		deng	di		diao	die	diu	dian			ding		du		duo		dui	duan	dun					
t	ta	tai	tao	tan	tang	tou	tong	te			teng	ti		tiao	tie		tian			ting		tu		tuo		tui	tuan	tun					
n	na	nai	nao	nan	nang	nou	nong	ne	nei	nen	neng	ni		niao	nie	niu	nian	nin	niang	ning		nu		nuo			nuan			nü	nüe		
l	la	lai	lao	lan	lang	lou	long	le	lei		leng	li	lia	liao	lie	liu	lian	lin	liang	ling		lu		luo			luan	lun		lü	lüe		
g	ga	gai	gao	gan	gang	gou	gong	ge	gei	gen	geng											gu	gua	guo	guai	gui	guan	gun	guang				
k	ka	kai	kao	kan	kang	kou	kong	ke	kei	ken	keng											ku	kua	kuo	kuai	kui	kuan	kun	kuang				
h	ha	hai	hao	han	hang	hou	hong	he	hei	hen	heng											hu	hua	huo	huai	hui	huan	hun	huang				
j												ji	jia	jiao	jie	jiu	jian	jin	jiang	jing	jiong									ju	jue	juan	jun
q												qi	qia	qiao	qie	qiu	qian	qin	qiang	qing	qiong									qu	que	quan	qun
x												xi	xia	xiao	xie	xiu	xian	xin	xiang	xing	xiong									xu	xue	xuan	xun
zh	zha	zhai	zhao	zhan	zhang	zhou	zhong	zhe	zhei	zhen	zheng	zhi										zhu	zhua	zhuo	zhuai	zhui	zhuan	zhun	zhuang				
ch	cha	chai	chao	chan	chang	chou	chong	che		chen	cheng	chi										chu	chua	chuo	chuai	chui	chuan	chun	chuang				
sh	sha	shai	shao	shan	shang	shou		she	shei	shen	sheng	shi										shu	shua	shuo	shuai	shui	shuan	shun	shuang				
r			rao	ran	rang	rou	rong	re		ren	reng	ri										ru		ruo		rui	ruan	run					
z	za	zai	zao	zan	zang	zou	zong	ze	zei	zen	zeng	zi										zu		zuo		zui	zuan	zun					
c	ca	cai	cao	can	cang	cou	cong	ce		cen	ceng	ci										cu		cuo		cui	cuan	cun					
s	sa	sai	sao	san	sang	sou	song	se		sen	seng	si										su		suo		sui	suan	sun					

The Basis of Mandarin Rhythm: the Syllable ‘Abstract Oscillator’

33 vowels with addition of o and ueng(ong) =35

the missing vowel o is place under uo and ueng under ong

22 consonants including 0 consonant

Pinyin	a	ai	ao	an	ang	ou	ong	e	ei	en	eng	i	ia	iao	ie	iu (iou)	ian	in	iang	ing	iong	u	ua	uo	uai	ui (uei)	uan	un (uen)	uang	ü	üe	üan	ün
ø	a	ai	ao	an	ang	ou	weng	e	ei	en	eng	yi	ya	yao	ye	you	yan	yin	yang	ying	yong	wu	wa	wo	wai	wei	wan	wen	wang	yu	yue	yuan	yun
b	ba	bai	bao	ban	bang																												
p	pa	pai	pao	pan	pang	pou																											
m	ma	mai	mao	man	mang	mou																											
f	fa			fan	fang	fou																											
d	da	dai	dao	dan	dang	dou																						dun					
t	ta	tai	tao	tan	tang	tou																						tun					
n	na	nai	nao	nan	nang	nou																								nü	nüe		
l	la	lai	lao	lan	lang	lou																						lun		lü	lüe		
g	ga	gai	gao	gan	gang	gou																						gun	guang				
k	ka	kai	kao	kan	kang	kou																						kun	kuang				
h	ha	hai	hao	han	hang	hou																						hun	huang				
j																														ju	jue	juan	jun
q																														qu	que	quan	qun
x																														xu	xue	xuan	xun
zh	zha	zhai	zhao	zhan	zhang	zhou																						zhun	zhuang				
ch	cha	chai	chao	chan	chang	chou																						chun	chuang				
sh	sha	shai	shao	shan	shang	shou																						shun	shuang				
r			rao	ran	rang	rou																						run					
z	za	zai	zao	zan	zang	zou																						zun					
c	ca	cai	cao	can	cang	cou	cong	ce		cen	ceng	ci										cu		cuo		cui	cuan	cun					
s	sa	sai	sao	san	sang	sou	song	se		sen	seng	si										su		suo		sui	suan	sun					

Something to think more about:

Note the difference between ***actual* syllables** (lexicalised, in Mandarin: corresponding to characters) and ***potential* syllables** (predicted, in Mandarin: without characters):

$$\text{SYLLABLES}_{actual} \subseteq \text{SYLLABLES}_{potential}$$

but usually:

$$\text{SYLLABLES}_{actual} \subset \text{SYLLABLES}_{potential}$$

Can you invent new Mandarin syllables which are not associated with characters?

The Basis of Mandarin Rhythm: the Syllable ‘Abstract Oscillator’

33 vowels with addition of o and ueng(ong) =35

the missing vowel o is place under uo and ueng under ong

22 consonants including 0 consonant

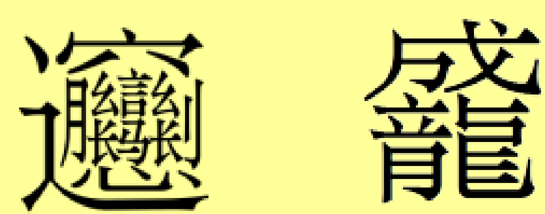
Pinyin	a	ai	ao	an	ang	ou	ong	e	ei	en	eng	i	ia	iao	ie	iu (iou)	ian	in	iang	ing	iong	u	ua	uo	uai	ui (uei)	uan	un (uen)	uang	ü	üe	üan	ün
ø	a	ai	ao	an	ang	ou	weng	e	ei	en	eng	yi	ya	yao	ye	you	yan	yin	yang	ying	yong	wu	wa	wo	wai	wei	wan	wen	wang	yu	yue	yuan	yun
b	ba	bai	bao	ban	bang				bei	ben	beng	bi		biao	bie		bian	bin		bing		bu		bo									
p	pa	pai	pao	pan	pang	pou			pei	pen	peng	pi		piao	pie		pian	pin		ping		pu		po									
m	ma	mai	mao	man	mang	mou		me	mei	men	meng	mi		miao	mie	miu	mian	min		ming		mu		mo									
f	fa			fan	fang	f																											
d	da	dai	dao	dan	dang	d																											
t	ta	tai	tao	tan	tang	t																											
n	na	nai	nao	nan	nang	n																								nü	nüe		
l	la	lai	lao	lan	lang	l																								lü	lüe		
g	ga	gai	gao	gan	gang	g																											
k	ka	kai	kao	kan	kang	k																											
h	ha	hai	hao	han	hang	h																											
j																														ju	jue	juan	jun
q																														qu	que	quan	qun
x																														xu	xue	xuan	xun
zh	zha	zhai	zhao	zhan	zhang	zh																											
ch	cha	chai	chao	chan	chang	ch																											
sh	sha	shai	shao	shan	shang	sh																											
r			rao	ran	rang	rou	rong	re		ren	reng	ri										ru		ruo		rui	ruan	run					
z	za	zai	zao	zan	zang	zou	zong	ze	zei	zen	zeng	zi										zu		zuo		zui	zuan	zun					
c	ca	cai	cao	can	cang	cou	cong	ce		cen	ceng	ci										cu		cuo		cui	cuan	cun					
s	sa	sai	sao	san	sang	sou	song	se		sen	seng	si										su		suo		sui	suan	sun					

So how about syllables

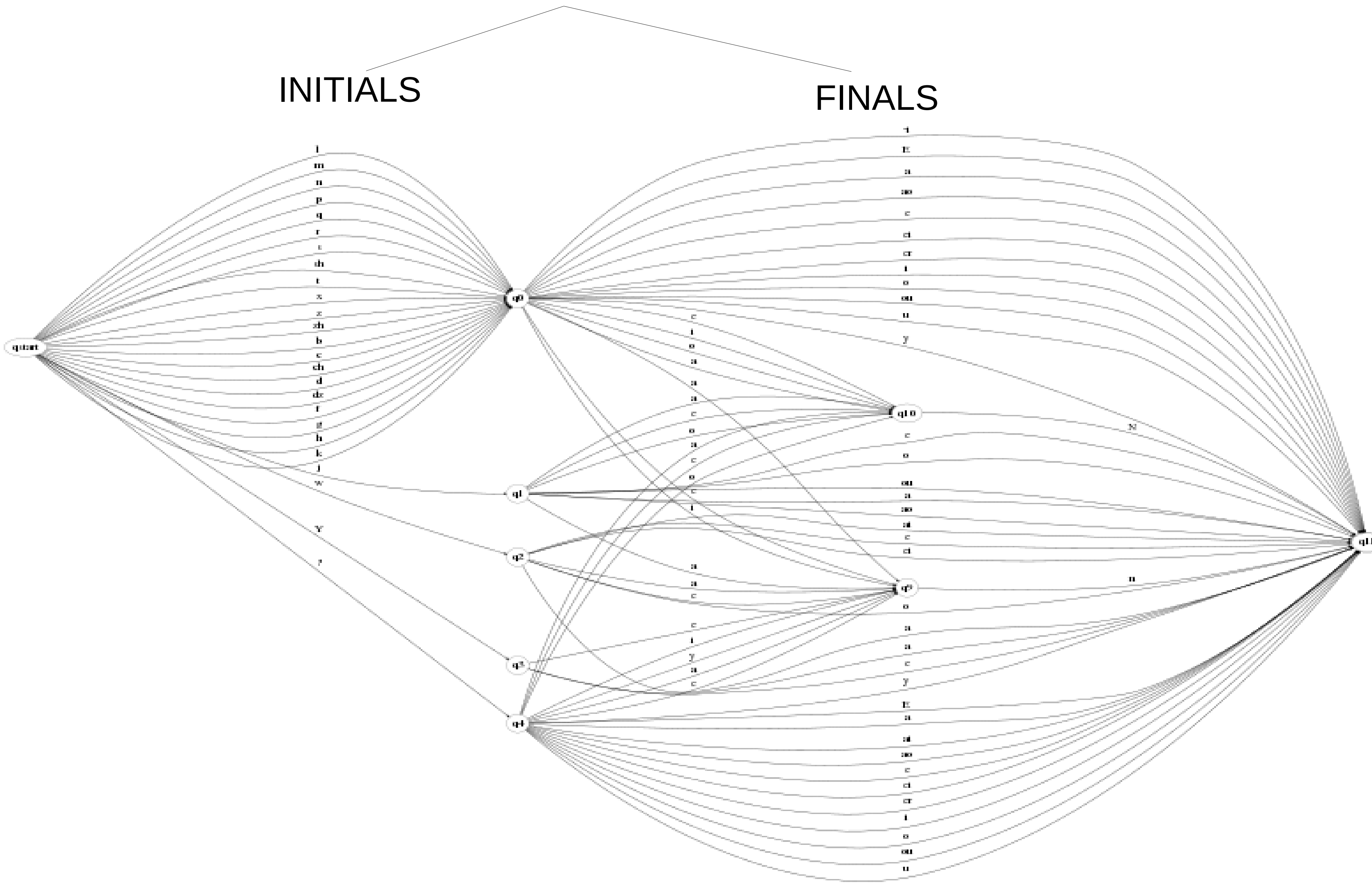
- **which were pronunced before they were written,**
- **and for which characters were invented later (which is historically the case with all characters)**

For example

- ***biangbiang***
- ***duang***

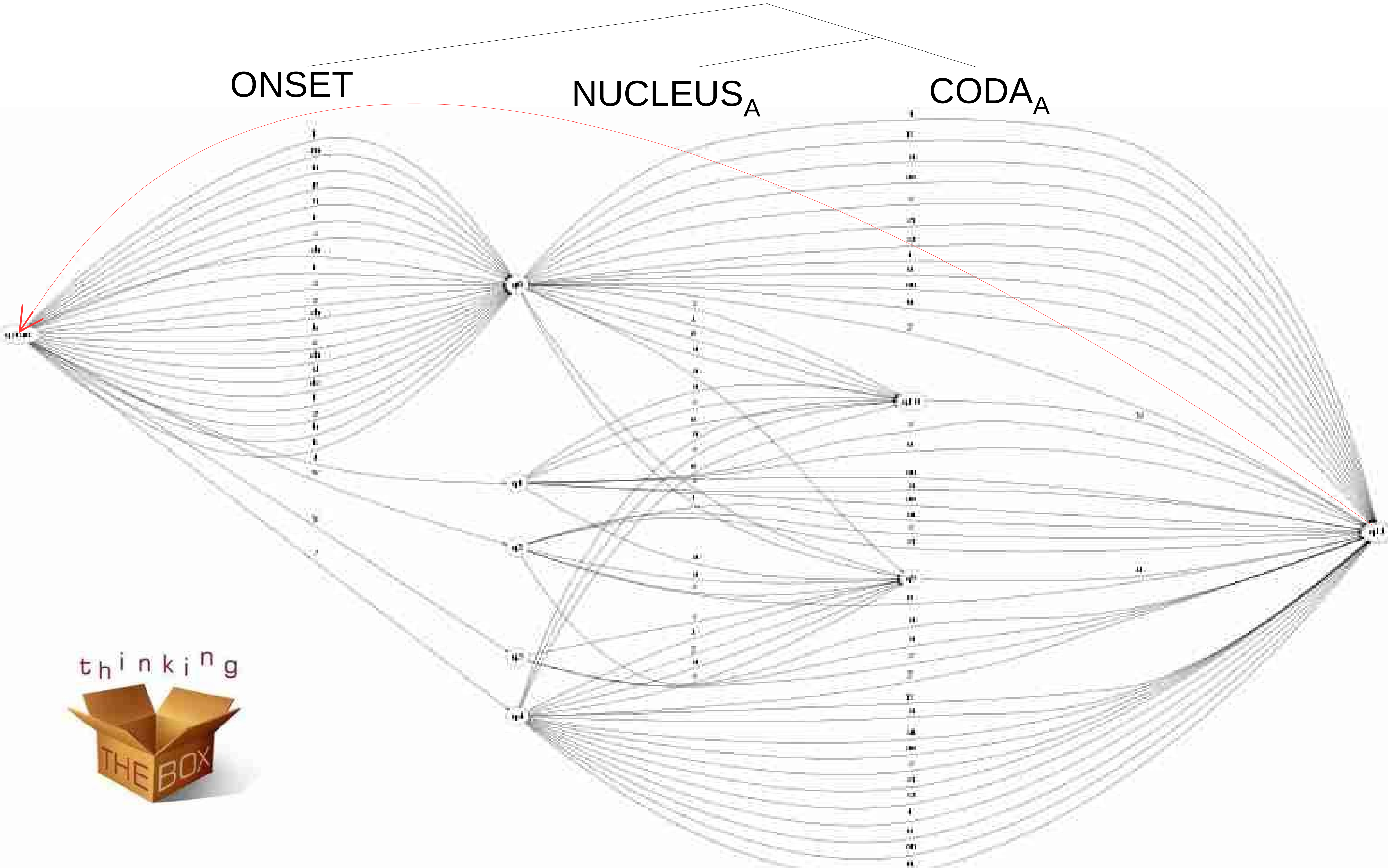


The Basis of Mandarin Rhythm: the Syllable ‘Abstract Oscillator’



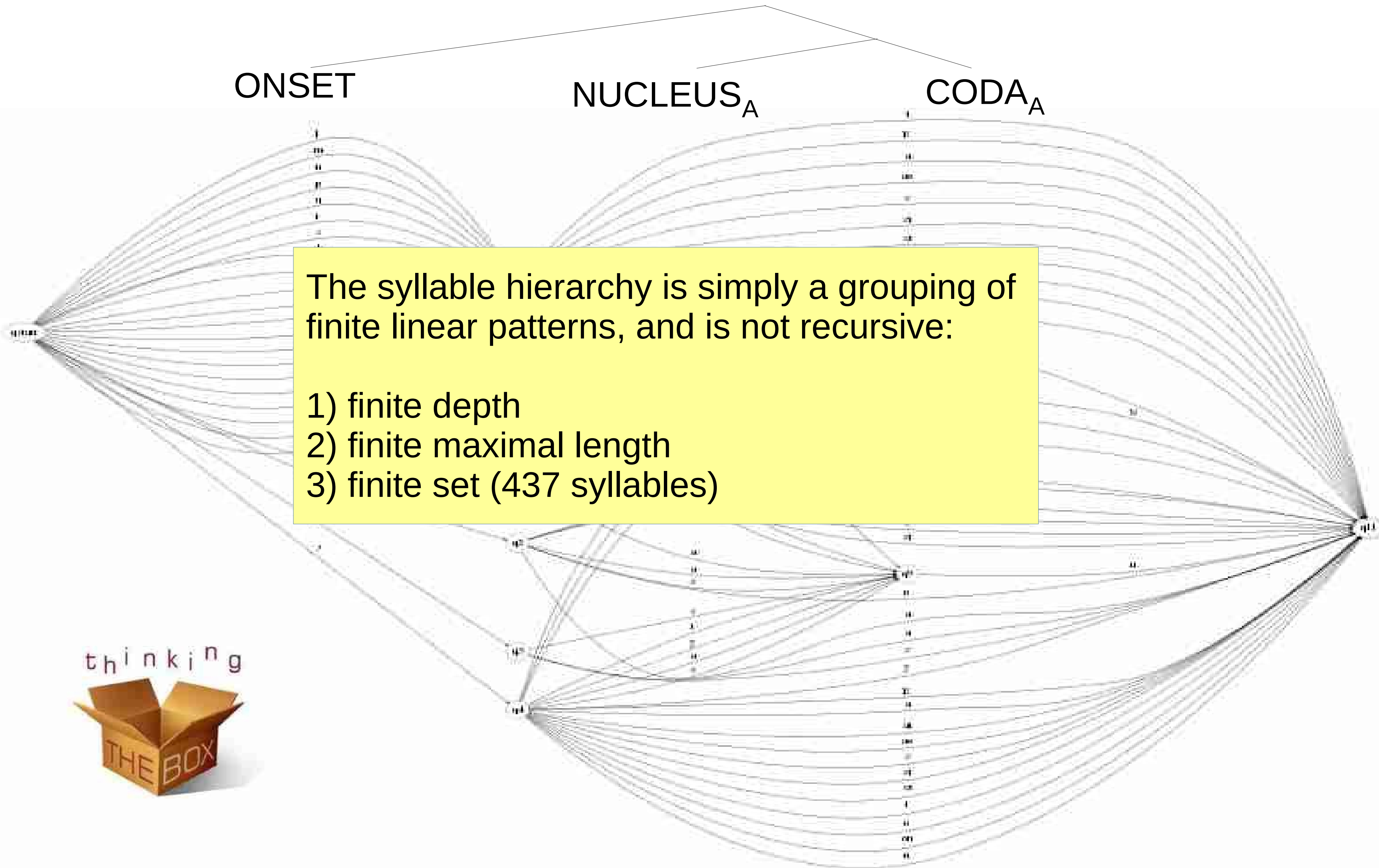
Linear Syllable Grammar for Mandarin

The Basis of Mandarin Rhythm: the Syllable ‘Abstract Oscillator’



Linear Syllable Grammar for Mandarin

The Basis of Mandarin Rhythm: the Syllable 'Abstract Oscillator'



The syllable hierarchy is simply a grouping of finite linear patterns, and is not recursive:

1) finite depth
2) finite maximal length
3) finite set (437 syllables)

Linear Syllable Grammar for Mandarin

A note on Oscillation, Iteration and Recursion

Processing Prosody: a Computational Perspective

Rhythm as Oscillation is based on iteration, cycles, loops

(or on a linear variety of recursion)

Computational requirements for real time processing:

(the recursion issue):

- finite memory space
- finite or linear processing time

Fulfilment of real time processing requirements:

- iterative grammars have linear processing requirements
- right-branching, or left-branching grammars have linear processing time
- finite-depth grammars have constant finite processing time

Nonfulfilment of real time processing requirements:

- non-deterministic grammars (e.g. grammars like A→a b | a c
- centre-embedding phrase structure grammars

Processing Time and Processing Space: Rhythm and Recursion

Food for thought:

- recursion is not just about a node dominating another node with the same name – that name may be ill-defined and ambiguous, or a generalisation, or vague; this criterion is necessary but not sufficient
- recursion is about describing an infinite number of objects (sentences, words, numbers, …)
- a recursive theory of language and speech must also be realistic:
  - the Linear Processing Time Constraint:

    The time required for processing speech must be linear in relation to the length of the input.
  - the Finite Processing Space Constraint:

    The memory required for processing speech must be finite.

Processing Time and Processing Space: a Note on Recursion

In the many discussions of recursion over the past 20 years or so, this crucial distinction between two types of recursion with different processing time and space properties has been neglected:

- linear recursion:
  - left & right branching (computationally equivalent to iteration)
  - linear recursion is realistic, requiring finite working memory, and processing time which is a linear function of the size of the input
- non-linear recursion:
  - centre-embedding, cross-serial dependencies
  - non-linear recursion is unrealistic, requiring unrestricted memory and at least quadratic processing time, thus implausible for speech

Processing Time and Processing Space: a Note on Recursion

Non-linear recursion is unproblematic: the basic principle of **rhythm** and of **creativity** in language.

But speakers fail at producing and understanding centre-embedding in spontaneous speech. How can this then be a feature of language?

In rehearsed speech, writing and read speech, a small amount of centre-embedding is possible, due to the additional time and memory space provided by this kind of register.

Processing Time and Processing Space: a Note on Recursion

Where did centre-embedding come from?

Speakers were trying to be clever: generalising *linearly recursive* sentence-final nominal clauses (e.g. relative clauses, that clauses) to *centre-embedding* non-final positions.

So centre-embedding is

- derived from right or left recursion
- *plus* a generalisation:
  - “Use right (or left) branching anywhere”

Unfortunately, processing capacity is too limited to permit more than one application of this generalisation, unless rehearsal or writing are involved. And speakers fail.

Processing Time and Processing Space: a Note on Recursion

So where did centre-embedding really come from?

Speakers were trying to be clever: generalising *linearly recursive* sentence-final nominal clauses (e.g. relative clauses, that clauses) to *centre-embedding* non-final positions.

But this really only (partly) works with extra time and memory:

- rehearsal
- writing

1. Linear (right-branching):
   - Jim saw <u>the man who found the boy</u>
2. Centre-embedding experiment – tough to process:
   - <u>the man who found the boy</u> saw Jim
3. **Linear right-branching solution – use the passive:**
   - Jim was seen by <u>the man who found the boy</u>

Processing Time and Processing Space: a Note on Recursion

Try pronouncing this:

I met the lady who the girl who the teacher who my friend saw was teaching was visiting had in fact left town.

Processing Time and Processing Space: a Note on Recursion

Try pronouncing this:

I met the lady who the girl who the teacher who my friend saw was teaching was visiting had in fact left town.

Now try pronouncing this:

I met the lady who was being visited by the girl who was being taught by the teacher who was seen by my friend.

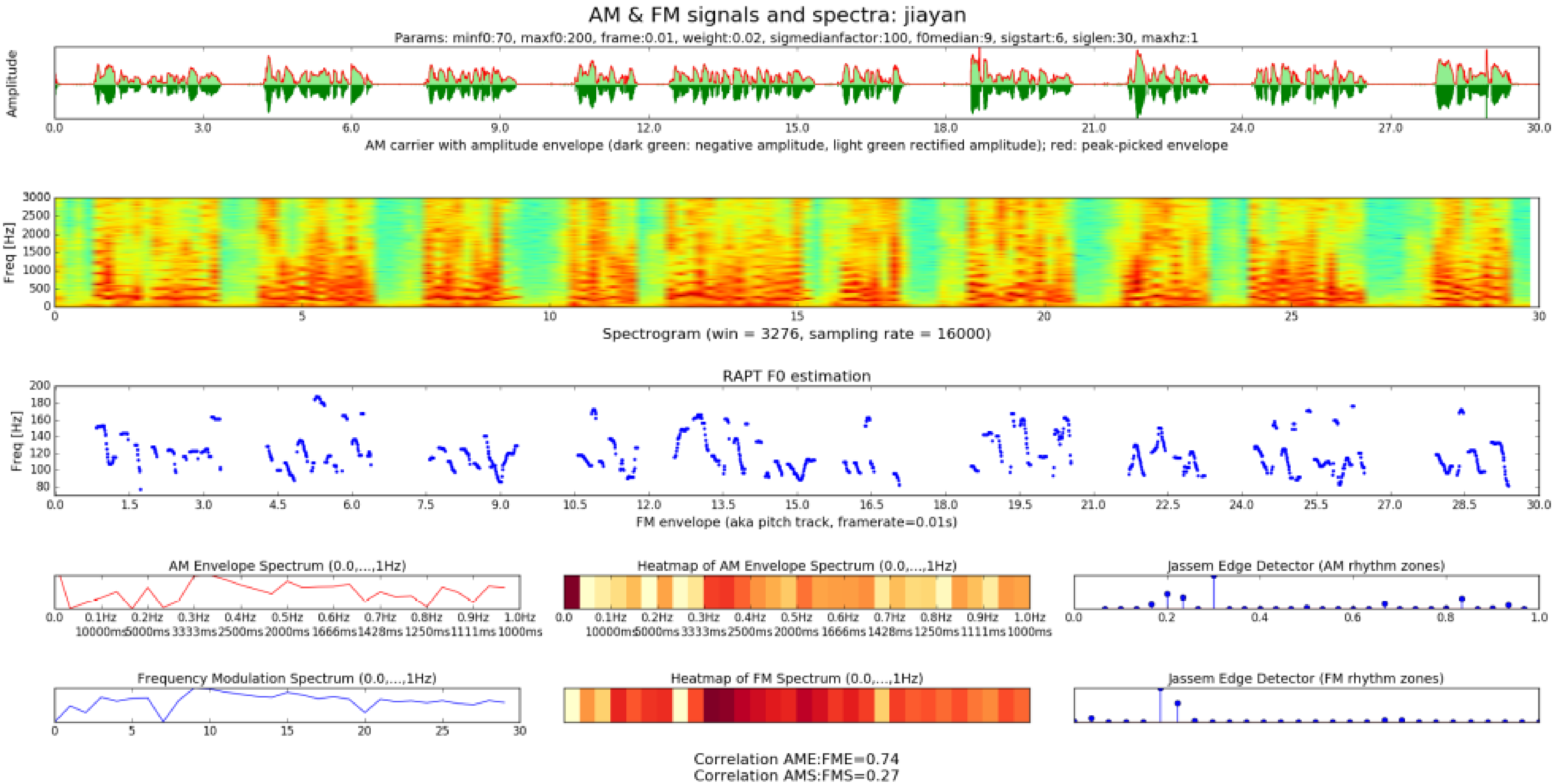
Looking Ahead: from Deduction to Induction

Automatic generalisation from data
Machine Learning
Artificial Intelligence

The Physical Basis of Speech Oscillations:

Modulation Theory

The Physical Basis of Speech Oscillations: Modulation Theory



Summary:

Aspects of Prosody and Time

Time Epochs

Time Types

The architecture of language:

Ranks and Interpretations

The Phonology of Prosody:

A computational perspective of different ranks

Summary:

Aspects of Prosody and Time
Time Epochs
Time Types

The architecture of language:
Ranks and Interpretations

The Phonology of Prosody:
A computational perspective of different ranks

Conclusion:

… thinking outside the box

Thank you!

谢谢！