Finite-State Methods in Morphology, Text-Analysis and the Analysis of Writing Systems

Richard Sproat

Bell Laboratories

600 Mountain Avenue

Murray Hill, NJ, 07974, USA

rws@bell-labs.com

http://www.bell-labs.com/project/tts/rws.html

Lucent Technologies
Bell Labs Innovations

Why the resurgence of interest in finite-state methods?

- Efficiency (Mohri, 1997)
- Wider coverage than previously understood.
  - Morphology (Koskenniemi, 1983, et seq.)
  - Phonology (Bird and Ellison, 1994)
  - Syntactic analysis (Mohri, 1994; Voutilainen, 1994; Roche, 1997)

  Addition of *weights* allows for modeling of gradient phenomena

  - Speech recognition (Pereira, Riley, and Sproat, 1994; Pereira and Riley, 1996)
  - Speech synthesis (Sproat, 1997b; Sproat, 1997c)

- Uniformity
-
- Bidirectionality

What this tutorial will cover

- Quick review of properties of regular languages/relations and regular automata/transducers

- "Converters": how to transform various kinds of descriptions into finite-state devices

- A particular application: computational morphology and phonology

- A more extensive application: "text analysis" for multilingual text-to-speech synthesis (TTS)

- Implications of the text analysis model for theories of writing systems

PART I: Review of Formal Issues

Formal issues: introduction

- Definitions of finite automata/transducers:

- Definitions of *weighted* finite automata/transducers:

- Some closure properties

- Some efficiency issues

Regular languages

1. $\emptyset$ is a regular language

2. For all symbols $a \in \Sigma \cup \epsilon$, $\{a\}$ is a regular language

3. If L_1, L_2 and L are regular languages, then so are

   (a) $L_1 \cdot L_2$, the *concatenation* of L_1 and L_2: for every $w_1 \in L_1$ and $w_2 \in L_2$, $w_1 w_2 \in L_1 \cdot L_2$

   (b) $L_1 \cup L_2$, the *union* of L_1 and L_2

   (c) L^*, the *Kleene closure* of L. Using L^i to denote L concatenated with itself i times, $L^* = \cup_{i=0}^{\infty} L^i$.

- *Intersection*: if L_1 and L_2 are regular languages then so is $L_1 \cap L_2$.

- *Difference*: if L_1 and L_2 are regular languages then so is $L_1 - L_2$, the set of strings in L_1 that are not in L_2.

- *Complementation*: if L is a regular language, then so is $\Sigma^* - L$, the set of all strings over Σ that are *not* in L. (Of course, complementation is merely a special case of difference.)

- *Reversal*: if L is a regular language, then so is $Rev(L)$, the

(cf. (Kaplan and Kay, 1994))

Regular n-relations

1. $\emptyset$ is a regular n-relation

2. For all symbols $a \in [(\Sigma \cup \epsilon) \times \ldots \times (\Sigma \cup \epsilon)]$, $\{a\}$ is a regular n-relation

3. If R_1, R_2 and R are regular n-relations, then so are

   (a) $R_1 \cdot R_2$, the *(n-way)concatenation* of R_1 and R_2: for every $r_1 \in R_1$ and $r_2 \in R_2$, $r_1 r_2 \in R_1 \cdot R_2$

   (b) $R_1 \cup R_2$

   (c) R^*, the *n-way Kleene closure* of R.

- *Composition*: if R_1 is a regular $k \times m$-relation and R_2 is a regular $m \times p$-relation, then $R_1 \circ R_2$ is a regular $k \times p$-relation.

- *Reversal*: if R is a regular n-relation, then so is $Rev(R)$.

- *Inversion*: if R is a regular $m \times n$-relation, then R^{-1}, the *inverse* of R, is a regular $n \times m$-relation.

(cf. (Kaplan and Kay, 1994))

Finite automata: definitions

$A = (\Sigma, Q, \delta, I, F)$

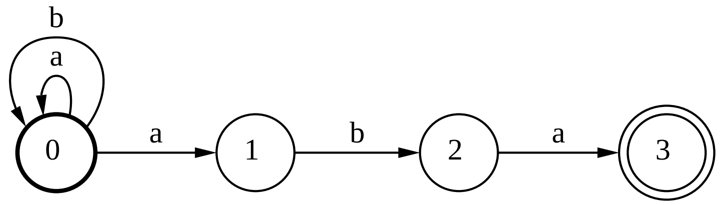
- Alphabet Σ (finite set of symbols),
- Finite set of states Q,
- Transition function δ: $Q \times \Sigma \rightarrow 2^Q$,
- $I \subseteq Q$ set of initial states,
- $F \subseteq Q$ set of final states.

A recognizes $L(A) = \{w \in \Sigma^* : \delta(I, w) \cap F \neq \emptyset\}$

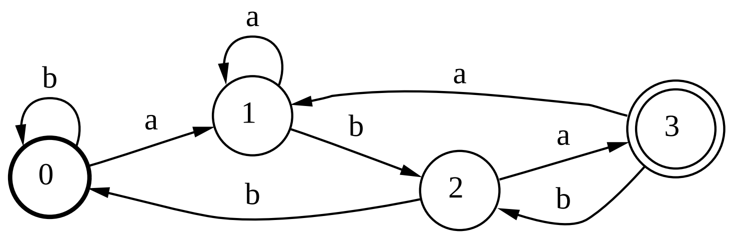
(Harrison, 1978; Hopcroft and Ullman, 1979; Lewis and Papadimitriou, 1981)

Theorem (Kleene, 1956). A set is regular (or rational) iff it can be recognized by a finite automaton.

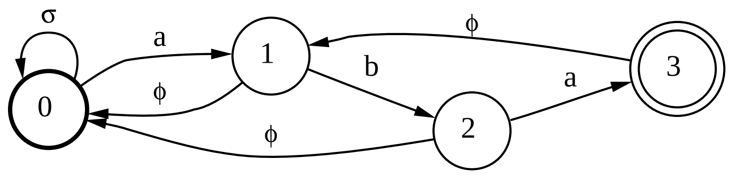
Finite automata: $L(A) = \Sigma^* aba$



non-deterministic



deterministic



failure-function

(Crochemore and Rytter, 1994) (Mohri, 1994)

Weighted automata: definitions

$A = (\Sigma, Q, \lambda, \delta, \sigma, \rho, I, F)$

- $(\Sigma, Q, \delta, I, F)$ is an automaton,
- Initial output function λ,
- Output function σ: $Q \times \Sigma \times Q \rightarrow K$,
- Final output function ρ,
- Function $f : \Sigma^* \longrightarrow (K, +, \cdot)$ associated with A:
  $\forall u \in Dom(f), f(u) = \sum_{(i,q)\in[I\times(\delta(i,u)\cap F)]} (\lambda(i) \cdot \sigma(i, u, q) \cdot \rho(q)).$

Semirings

- $(K, +)$ is a commutative monoid with neutral element denoted by 0
- $(K, \cdot)$ is a monoid with neutral element denoted by 1
- the product is distributive with respect to the sum
- for all a in K, $0a = a0 = 0$

(from (Berstel and Reutenauer, 1988))

Some common semirings:

- $(\cdot, +)$
- $(+, min)$ ('tropical') $\Leftarrow$ *commonly used in language modeling*

```
#define EXTEND(x,y) ((x)+(y))

#define COLLECT(x,y) (CHEAPER(x,y) ? (x) : (y))
```

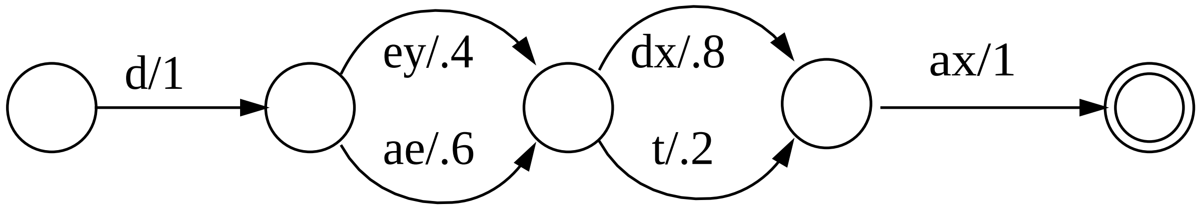
Rational power series

- *Power series*: functions mapping Σ^* to a semiring $(K, +, \cdot)$
- *Rational power series*: closure under rational operations of polynomials (polynomial power series) (Salomaa and Soittola, 1978; Berstel and Reutenauer, 1988)

Theorem (Schützenberger, 1961). A power series is rational iff it can be represented by a weighted finite automaton.

Weighted automata: example

Pronunciations of *data*



Transducers: definitions

$T = (\Sigma, \Delta, Q, \delta, \sigma, I, F)$

- Finite alphabets Σ and Δ,
- Finite set of states Q,
- Transition function δ: $Q \times \Sigma \rightarrow 2^Q$,
- Output function σ: $Q \times \Sigma \times Q \rightarrow \Sigma^*$,
- $I \subseteq Q$ set of initial states,
- $F \subseteq Q$ set of final states.

T defines a relation:

$$R(T) = \{(u, v) \in (\Sigma^*)^2 : v \in \bigcup_{q \in (\delta(I,u) \cap F)} \sigma(I, u, q)\}$$

Weighted transducers: example

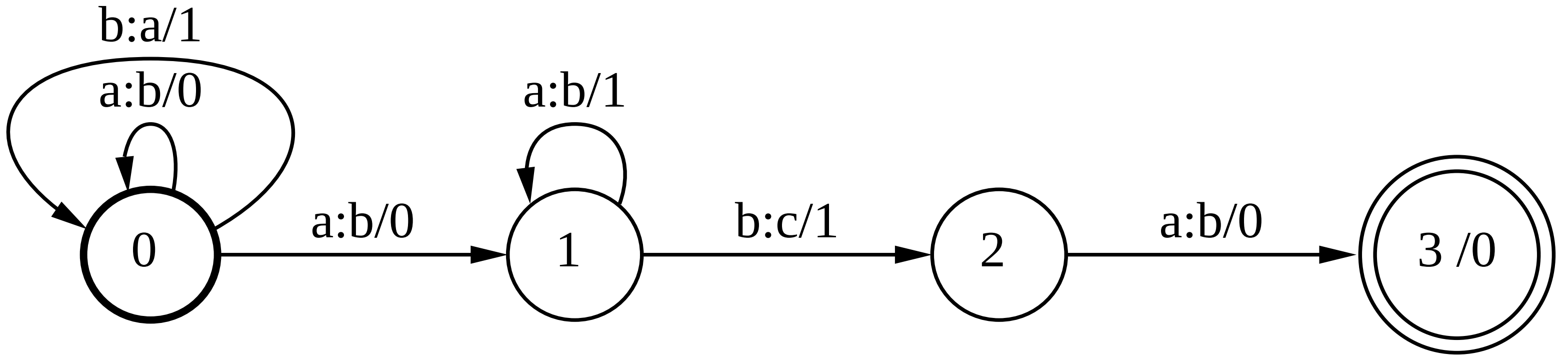


Figure 1: Example, $aaba \rightarrow (bbcb, (0 \odot 0 \odot 1 \odot 0) \oplus (0 \odot 1 \odot 1 \odot 0))$.

$$(\min, +): \quad aaba \rightarrow \min\{1, 2\} = 1$$

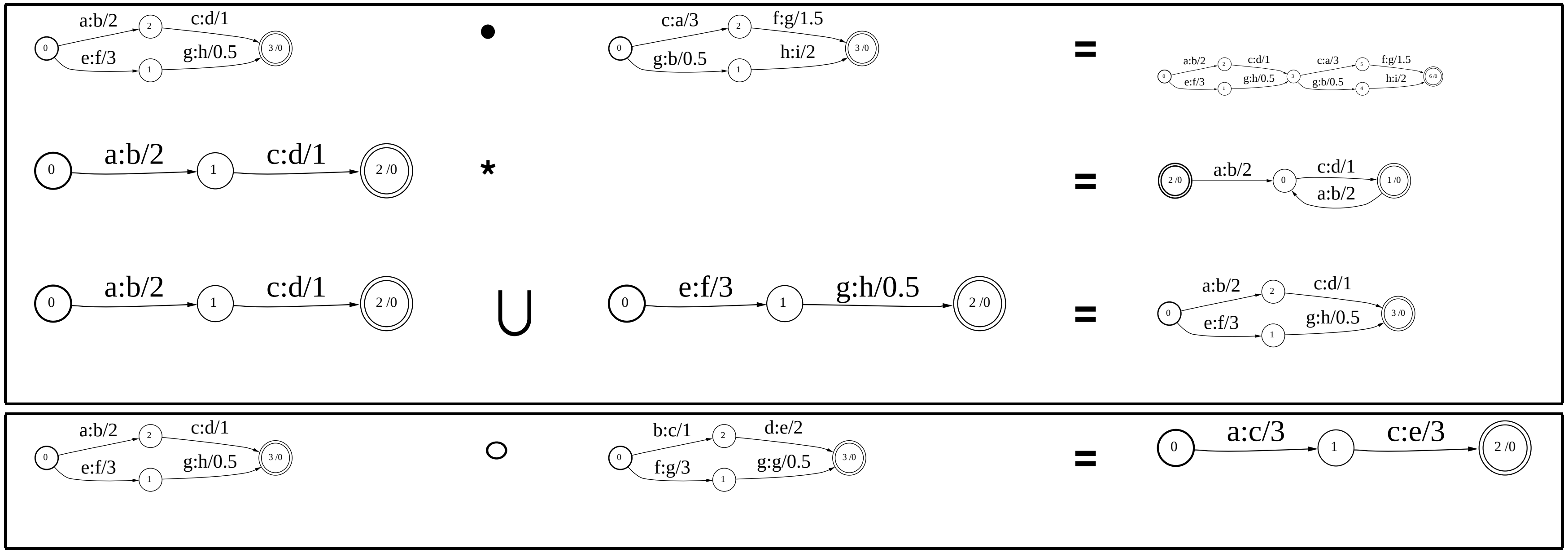
$$(+, \cdot): \quad aaba \rightarrow 0 + 0 = 0$$

Closure properties: review

	unweighted langs	weighted langs	relations
concatenation	yes	yes	yes
Kleene closure	yes	yes	yes
union	yes	yes	yes
intersection	yes	yes	**no**†
complementation	yes	**no**	**no**†
composition	—	—	yes

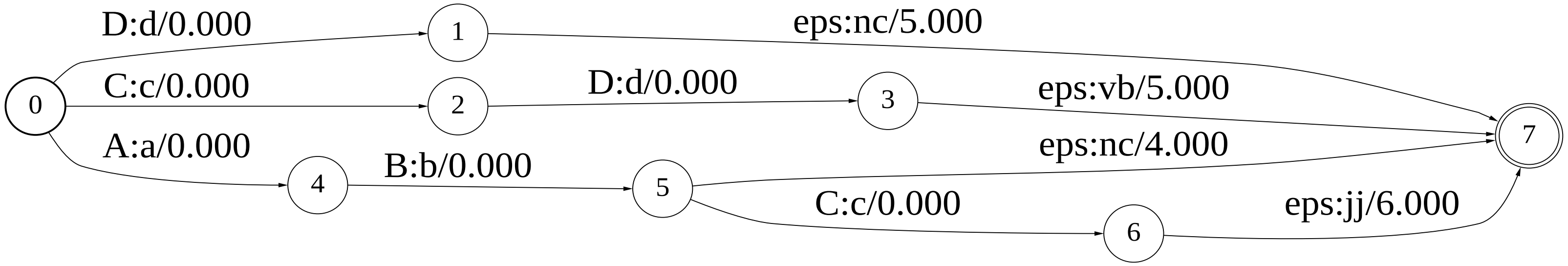
†Same-length relations *are* closed under intersection (and unweighted same-length relations are closed under complementation) an important property for "Two-Level" systems.

Weighted finite state transducers: some operations

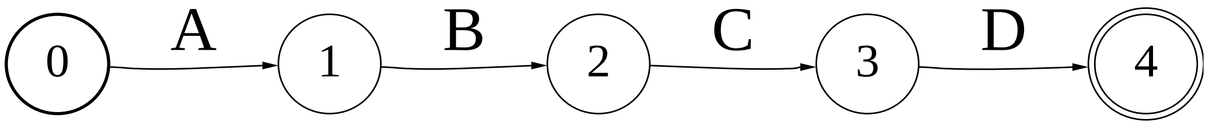


Composition: weighted transducers

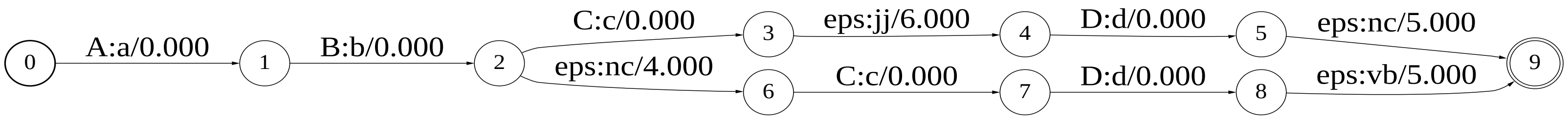
Dictionary D



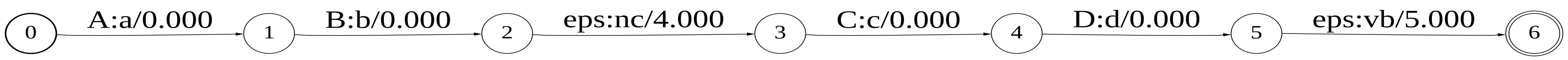
Input I



$I \circ D^*$



$BestPath(Id(I) \circ D^*)$



Efficiency issues: determinization

- Any FSA can be determinized
- Any *acyclic* WFSA or WFST can be determinized
- Any WFSA on the tropical semiring that has the *twins property*†can be determinized (Mohri, 1997)

Twins property†: any pair of states q, $q\prime$ that can be reached from the initial state by the same string u, and for which the minimum cost of any string v that reaches q *from* q (i.e. via a loop) is identical to the minimum cost for v reaching $q\prime$ from $q\prime$. A WFSA *has the twins property* when *any* pair of states are twins.

PART II: Converters

Introduction

- *Compilation from regular expressions*: a^*b, $(c{:}d)^+(d{:}c)$.
  These are useful in the representation of lexicons, declarative constraints ...

- Rewrite rule compilation

- Decision tree compilation

- Decision list compilation

- Compilation from neural nets (Omlin and Giles, 1996)

Rewrite rule compilation

- **Context-dependent rewrite rules**, general form:

$$\phi \rightarrow \psi / \lambda \_\_\_ \rho$$

  $\phi, \psi, \lambda, \rho$ regular expressions.

- Constraint: ψ cannot be rewritten but can be used as a context (Johnson, 1972; Kaplan and Kay, 1994)

- Example:

$$a \rightarrow b / c \_\_\_ b$$

- Composition of transducers allows for modeling ordered rules: $R_1 \circ R_2 \ldots R_n$

- Various algorithms for performing the compilation: (Karttunen and Beesley, 1992; Kaplan and Kay, 1994; Mohri and Sproat, 1996; Grimley-Evans, Kiraz, and Pulman, 1996; Karttunen, 1995)

Rule compilation method

(Mohri and Sproat, 1996)

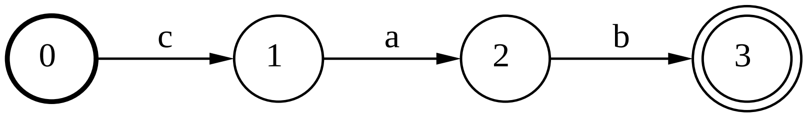
$$r \circ f \circ replace \circ l_1 \circ l_2$$

$$
\begin{array}{ll}
r: & \Sigma^* \rho \rightarrow \Sigma^* > \rho \\
f: & (\Sigma \cup \{>\})^* \phi > \rightarrow (\Sigma \cup \{>\})^* \{<_1, <_2\} \phi > \\
replace: & <_1 \phi > \rightarrow <_1 \psi \\
l_1: & \Sigma^* \lambda <_1 \rightarrow \Sigma^* \lambda \\
l_2: & \Sigma^* \overline{\lambda} <_2 \rightarrow \Sigma^* \overline{\lambda}
\end{array}
$$

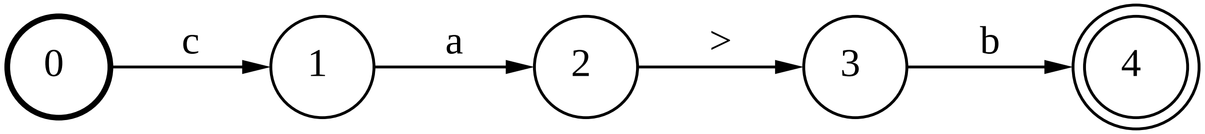
Example

$$a \rightarrow b / c \underline{\quad} b, \quad w = cab$$

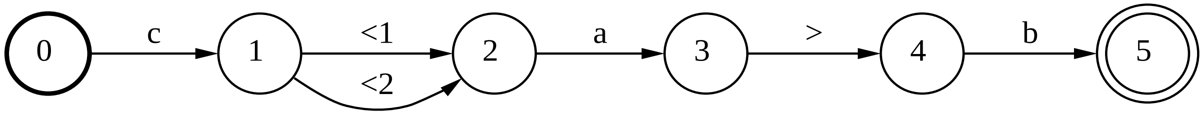
Input:



After r**:**

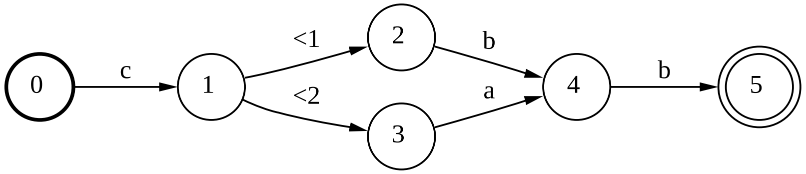


After f**:**

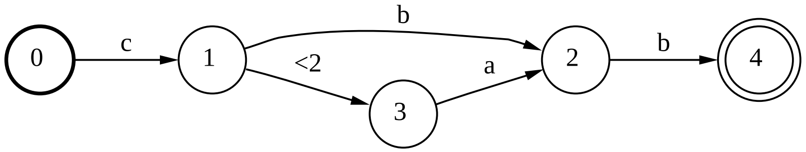


Example

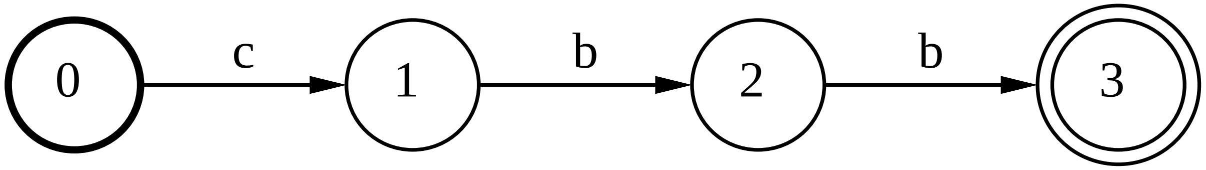
After *replace*:



After l_1:



After l_2:



Extension to weighted rules

(Mohri and Sproat, 1996)

Weighted context-dependent rules:

$$\phi \rightarrow \psi / \lambda \underline{\quad} \rho$$

- ϕ, λ, ρ regular expressions,
- ψ a weighted regular expression

Example:

$$c \rightarrow (0.15c) + (3.32t)/a \underline{\quad} t$$

Decision trees

- Well-known algorithms for growing decision trees from data (Breiman et al., 1984; Riley, 1989).

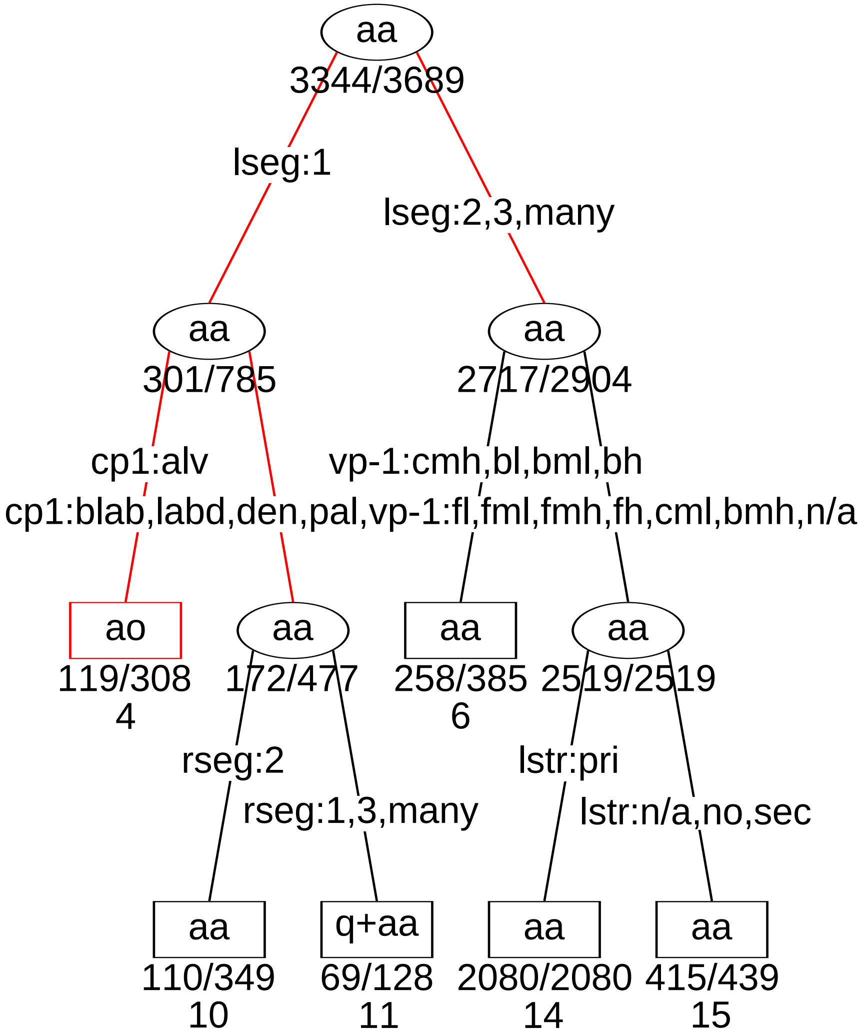
- Many applications to speech and natural language in the literature:

  - Modeling of phonetic variation (Riley, 1991; Ljolje and Riley, 1992)

  - Phrasing prediction in TTS (Wang and Hirschberg, 1992; Ostendorf and Veilleux, 1994)
  - Syntactic parsing (Magerman, 1995).

A quick overview of tree-based modeling

Tree modeling the phonetic realization of /aa/



Some ARPABET/IPA correspondences

aa	ɑ
ao	ɔ
ax	ə
ah	ʌ
q+aa	ʔɑ
q+ao	ʔɔ

Outputs at node 4

phone	probability	$-\log$ prob. (weight)
ao	**0.385**	**0.95**
aa	0.289	1.24
q+aa	0.103	2.27
q+ao	0.096	2.34
ah	0.069	2.68
ax	0.058	2.84

Some important assumptions about decision trees

- Each decision tree is a *complete* description: for any new data point, there will be some leaf node that corresponds to it

- For the purposes of the compilation algorithm, each decision tree has the property that its predictions specify how to rewrite a symbol in an input string

  - More concretely, each leaf represents a contextually determined *two-level* mapping between an input symbol and a set of output symbols

The tree compilation algorithm

(Sproat and Riley, 1996)

- Each *leaf* node corresponds to *single* rule defining a *constrained weighted mapping* for the input symbol associated with the tree

- Decisions at each node are stateable as regular expressions restricting the left or right context of the rule(s) dominated by the branch

- The full left/right context of the rule at a leaf node are derived by *intersecting* the expressions traversed between the root and leaf node

- The transducer for the entire tree represents the conjunction of all the constraints expressed at the leaf nodes; it is derived by *intersecting* together the set of WFSTs corresponding to each of the leaves
  - Note that intersection is defined for transducers that express same-length relations

- The *alphabet* is defined to be an alphabet of all correspondence pairs that were determined empirically to be possible

Regular-expression interpretation of decisions to node 4

(a)	left branch	λ	$=$	$\#$
		ρ	$=$	Σ^*
	right branch	λ	$=$	$(\#\alpha) \cup (\#\alpha\alpha) \cup$
				$(\#\alpha\alpha(\alpha)^+)$
			$=$	$(\alpha)^+$
		ρ	$=$	Σ^*
(b)	left branch	λ	$=$	Σ^*
		ρ	$=$	(alv)
	right branch	λ	$=$	Σ^*
		ρ	$=$	$((blab)) \cup ((labd)) \cup ((den)) \cup$
				$((pal)) \cup ((vel)) \cup ((pha))$
				$\cup((n/a))$

Weighted regular expression for outputs at node 4

$$\psi = ao_{0.95} \cup aa_{1.24} \cup q{+}aa_{2.27} \cup q{+}ao_{2.34} \cup ah_{2.68} \cup ax_{2.84}$$

Weighted rules at node 4 and 6

Node 4:

$$\boxed{aa \rightarrow (ao_{0.95} \cup aa_{1.24} \cup q{+}aa_{2.27} \cup q{+}ao_{2.34} \cup ah_{2.68} \cup ax_{2.84})/\#\underline{\quad}(alv)}$$

Node 6:

$$aa \rightarrow (aa_{0.40} \cup ao_{1.11})/\ (\#(\alpha)^{+}) \cap (\Sigma^{*}((cmh) \cup (bl) \cup (bml) \cup (bh)))\underline{\quad}\Sigma^{*}$$

Summary of compilation algorithm

Each rule represents a *weighted two-level surface coercion rule*

$$Rule_L = Compile(\phi_T \rightarrow \psi_L / \bigcap_{p \in P_L} \lambda_p \_\_\_ \bigcap_{p \in P_L} \rho_p)$$

Each tree/forest represents a set of simultaneous weighted two-level surface coercion rules

$$Rule_T = \bigcap_{L \in T} Rule_L$$

$$Rule_F = \bigcap_{T \in F} Rule_T$$

Decision lists

- Decision lists are an alternative to decision trees (Rivest, 1987; Yarowsky, 1996)

- Have been used in various *sense disambiguation* applications by Yarowsky and others:

  - Semantic ambiguities: *crane* (bird vs. construction equipment)

  - Case restoration: *NICE MAYOR TO RESIGN*

  - Accent restoration: *cote* (*côté* 'side' vs. *côte* 'coast')

  - Homograph disambiguation: *bass* (/beɪs/ 'musical range' vs. /bæs/ 'fish')

Decision lists: homograph disambiguation

Choose single best piece of matching evidence.

Decision List for ***lead***		
Logprob	Evidence	Pronunciation
11.40	*follow/V* + lead	$\Rightarrow$ lid
11.20	*zinc* $\leftrightarrow$ lead	$\Rightarrow$ lɛd
11.10	lead *level/N*	$\Rightarrow$ lɛd
10.66	*of* lead *in*	$\Rightarrow$ lɛd
10.59	*the* lead *in*	$\Rightarrow$ lid
10.51	lead *role*	$\Rightarrow$ lid
10.35	*copper* $\leftrightarrow$ lead	$\Rightarrow$ lɛd
10.28	lead *time*	$\Rightarrow$ lid
10.16	lead *poisoning*	$\Rightarrow$ lɛd

Decision lists as WFSTs: general approach

1. Find and tag (H0, H1 ...) instances of homograph in input.

2. For each piece of contextual evidence, mark

   - what type of evidence it is
   - which pronunciation it triggers

   Transduce other symbols to Δ

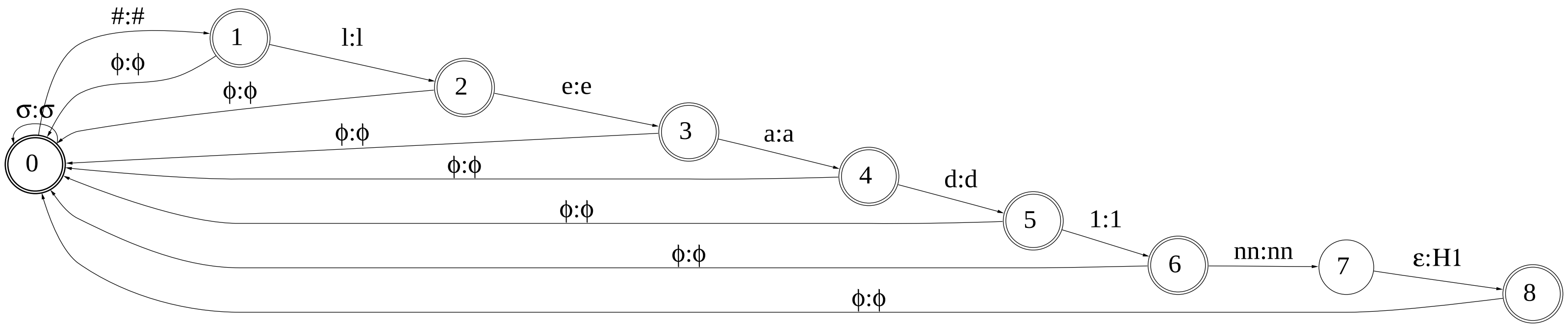
3. Optionally change H0, H1 ... into *ok* in appropriate environments

4. Filter unchanged H0, H1 ...

Decision lists as WFSTs: 1

The *lead* example

- Construct 'homograph taggers' H_0, H_1 … that find and tag instances of a homograph set in a lexical analysis. For example, H_1 can be represented by the following failure-function automaton:



Decision lists as WFSTs: 2

- Construct an *environmental classifier* consisting of a pair of transducers C_1 and C_2, where
  - C_1 optionally rewrites any symbol except the word boundary or the homograph tags H0, H1 ..., as a single dummy symbol Δ
  - C_2 classifies contextual evidence from the decision list according to its type, and assigns a cost equal to the position of the evidence in the list; and otherwise passes Δ, word boundary and H0, H1 ... through:

## follow vb ##	$\rightarrow$	## Δ V0 ## <1>
## zinc nn ##	$\rightarrow$	## Δ C1 ## <2>
## level(s?) nn ##	$\rightarrow$	## Δ R1 ## <3>
## of pp ##	$\rightarrow$	## Δ [1 ## <2>
## in pp ##	$\rightarrow$	## Δ 1] ## <2>
	$\vdots$	

Decision lists as WFSTs: 3

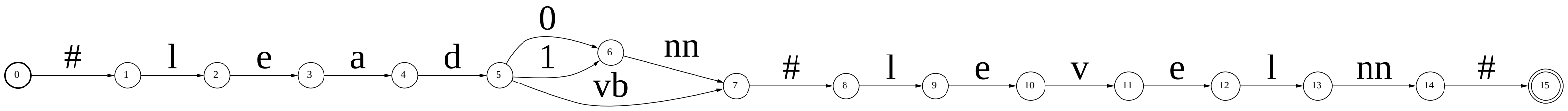
- Construct a *disambiguator* D from a set of *optional* rules of the form:

$$
\begin{array}{llll}
\mathrm{H0} & \rightarrow & ok & / \quad \mathrm{V0}\ \Sigma^*\ \underline{\quad} \\
\mathrm{H1} & \rightarrow & ok & / \quad \mathrm{C1}\ \Sigma^*\ \underline{\quad} \\
\mathrm{H1} & \rightarrow & ok & / \quad \underline{\quad}\ \Sigma^*\ \mathrm{C1} \\
\mathrm{H0} & \rightarrow & ok & / \quad \underline{\quad}\ \#\#\ \Delta^*\ \mathrm{R0} \\
\mathrm{H1} & \rightarrow & ok & / \quad \underline{\quad}\ \#\#\ \Delta^*\ \mathrm{R1} \\
\mathrm{H0} & \rightarrow & ok & / \quad [0\ \#\#\ \Delta^*\ \underline{\quad}\ \#\#\ \Delta^*\ 0] \\
\mathrm{H1} & \rightarrow & ok & / \quad [1\#\#\ \Delta^*\ \underline{\quad}\ \#\#\ \Delta^*\ 1] \\
 & \vdots & & \\
\mathrm{H0} & \rightarrow & ok < 20 > & \\
\mathrm{H1} & \rightarrow & ok < 40 > &
\end{array}
$$

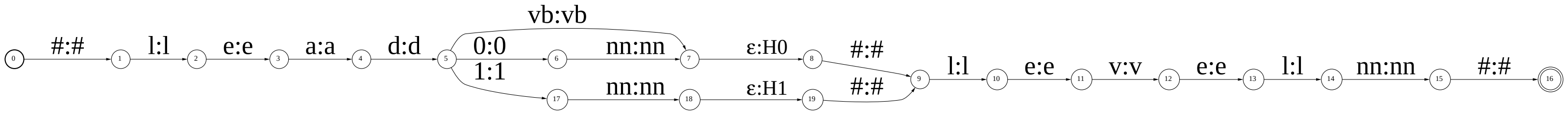
- Construct a *filter* F that removes all paths containing H0, H1 . . . .

Decision lists as WFSTs: 4

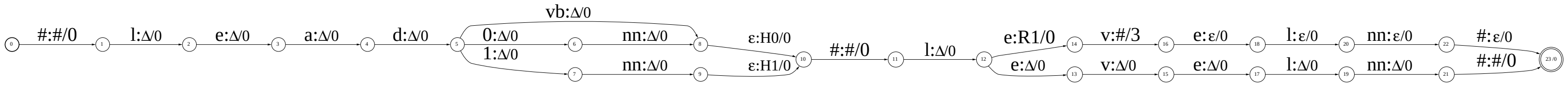
Let T be *lead* $\underline{\#\#}$ *level* $\underline{\#\#}$:



$T \circ H_0 \circ H_1$ is:

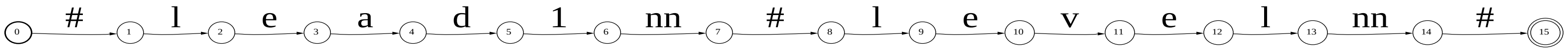


$T \circ H_0 \circ H_1 \circ C_1 \circ C_2$ is:



The disambiguated input

$T \cap \pi_1[\ BestPath\ [\ T \circ H_0 \circ H_1 \circ C_1 \circ C_2 \circ D \circ F\ ]\ ]$ is:



/lɛd/ $\underline{\#\#}$ *level* $\underline{\#\#}$

Why factor the problem in this way?

	States	Arcs
H_0	9	17
H_1	9	16
C_1	1	90
C_2	40	57
D	28	228
F	1	11
Σ	88	419
$H_0 \circ H_1 \circ C_1 \circ C_2 \circ D \circ F$	260	5984

(alphabet size: 47)

PART III:

Application 1:
Computational Morphology and Phonology

Computational morphology: introduction

- Models of computational *morphology* have mostly taken one of two approaches:
  - Finite state (Koskenniemi, 1983; Antworth, 1990; Tzoukermann and Liberman, 1990; Kay, 1987; Kiraz, 1996)
  - Unification-based (Bear, 1986; Trost, 1990; Ritchie et al., 1992)

- Models of computational *phonology* have essentially been exclusively finite-state (Koskenniemi, 1983; van Leeuwen, 1989; Antworth, 1990; Karttunen and Beesley, 1992; Karttunen, Kaplan, and Zaenen, 1992; Ritchie et al., 1992; Bird and Ellison, 1994; Kaplan and Kay, 1994; Mohri and Sproat, 1996)

  (But cf. "affix-stripping" models such as (Byrd et al., 1986, et seq.).)

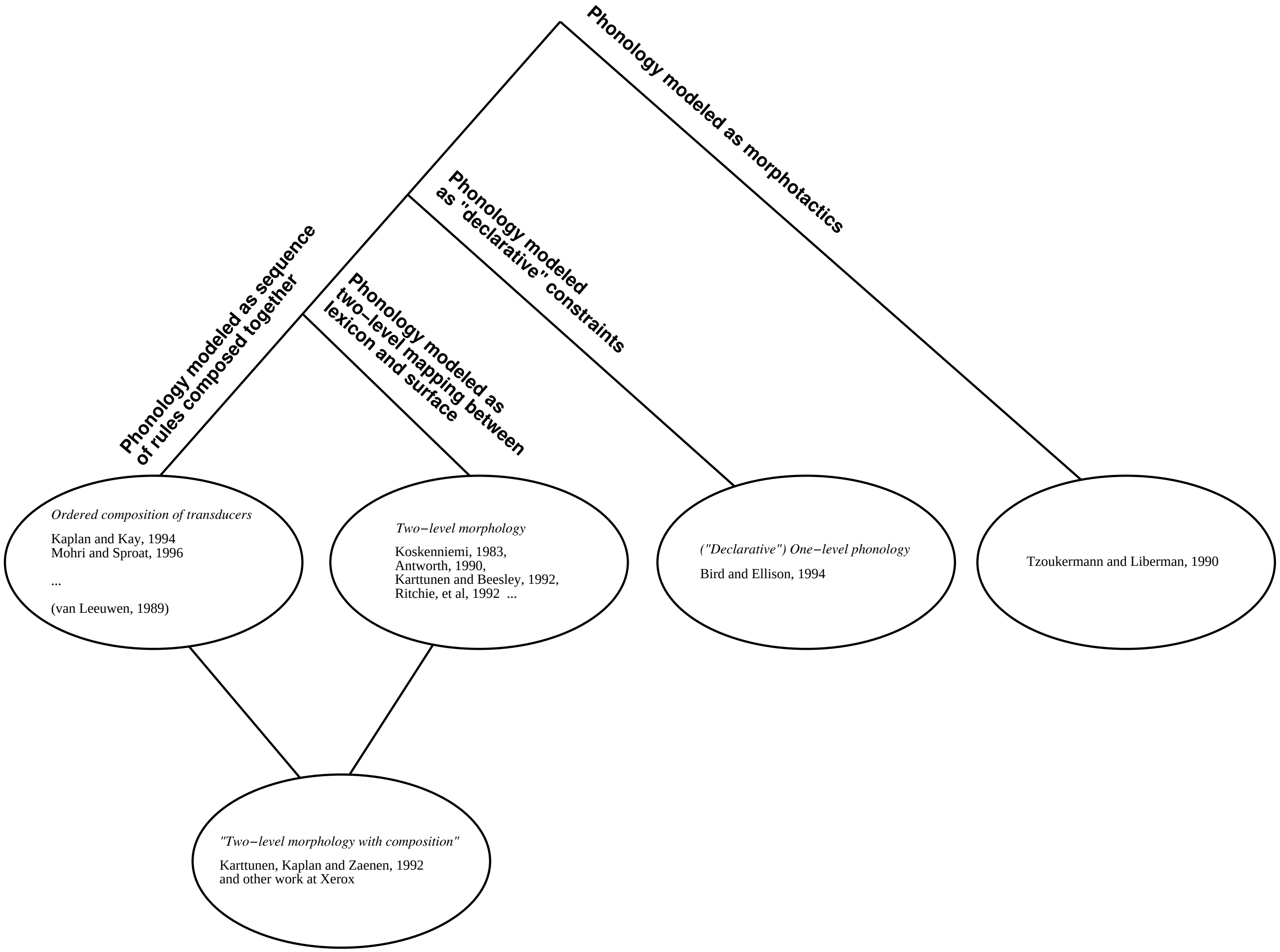
No discussion of computational morphology can ignore computational phonology.

Computational phonology: an example

Finnish Vowel Harmony

Nominative	**Partitive**	**Gloss**
taivas	taivas+ta	‘sky’
puhelin	puhelin+ta	‘telephone’
lakeus	lakeut+ta	‘plain’
syy	syy+tä	‘reason’
lyhyt	lyhyt+tä	‘short’
ystävällinen	ystävällinen+tä	‘friendly’

Computational phonology: approaches



Computational phonology: approaches

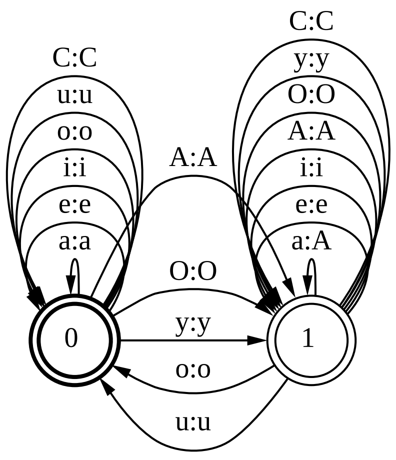
- Kaplan/Kay model depends upon the observation that a series of FST's composed together is also an FST. (Mohri and Sproat extended this to the weighted case.)

- Two-level morphology (Koskenniemi, 1983, et seq.) depends upon the observation that *certain classes of* transducers are closed under intersection ...

- One-level phonology depends upon the observation that most phonological constraints can be modeled as regular expressions.

Rewrite rule approach

$$a \rightarrow ä \;/\; [ä,ö,y]\ C^*\ ([i,e]\ C^*)^*\ \underline{\qquad}$$

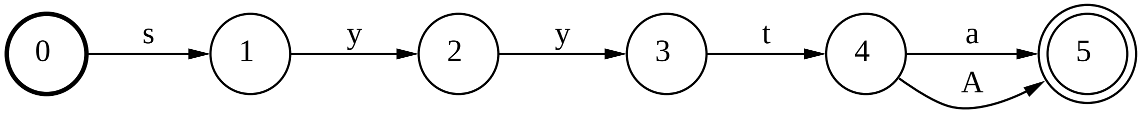


$$Surface = \pi_2[Lexical \circ Rule]$$

Declarative approach

$$\neg(FH\ (C \vee NV)^{*}\mathrm{a}) \wedge \neg(BH\ (C \vee NV)^{*}\ddot{\mathrm{a}})$$

FH	"front harmonizing"	/ä, ö, y/
NV	neutral vowels	/i, e/
BH	"back harmonizing"	all others



$$\textit{Correct Form} = \textit{Lexical} \cap \textit{Declaration}$$

Two-level vs. "multi-level" approaches

Rule ordered approach

$$w \rightarrow \epsilon \;/\; V \;\_\_\_\; V$$

$$a \rightarrow i \;/\; \_\_\_\; i$$

Two-level approach (notation of (Koskenniemi, 1983))

$$w{:}\epsilon \Leftarrow V \;\_\_\_\; V$$

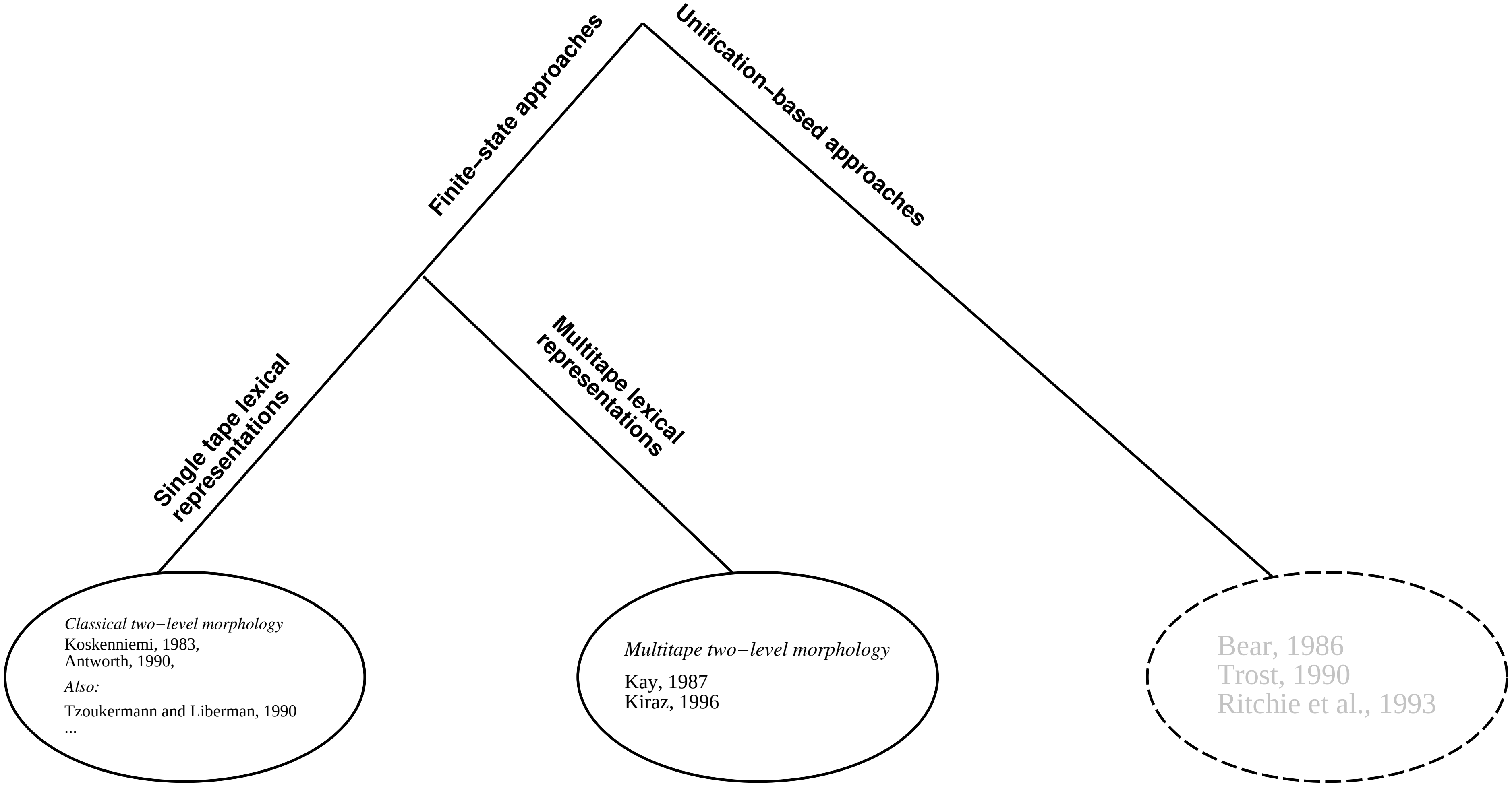
$$a{:}i \Leftarrow \;\_\_\_\; ({=}{:}\epsilon)?\;\; i$$

Computational morphology: an example

Spanish verbal morphology

Features	*hablar* 'speak'	*cerrar* 'close'	*cocer* 'cook'
ind pres 1 sg	hablo	c**ie**rro	c**uez**o
ind pres 2 sg	hablas	c**ie**rras	c**ue**ces
ind pres 3 sg	habla	c**ie**rra	c**ue**ce
ind pres 1 pl	hablamos	cerramos	cocemos
ind pres 2 pl	habláis	cerráis	cocéis
ind pres 3 pl	hablan	c**ie**rran	c**ue**cen
ind pret 1 sg	hablé	cerré	cocí
ind pret 2 sg	hablaste	cerraste	cociste
ind pret 3 sg	habló	cerró	coció
ind pret 1 pl	hablamos	cerramos	cocimos
ind pret 2 pl	hablasteis	cerrasteis	cocisteis
ind pret 3 pl	hablaron	cerraron	cocieron

Computational morphology: approaches



A simple 'arclist' model

START	ar	habl	hablar **vb**
START	ar	cerr **diph**	cerrar **vb**
START	er	coc **diph c/z**	cocer **vb**
ar	WORD	+o#	+ **ind pres 1 sg**
ar	WORD	+as#	+ **ind pres 2 sg**
ar	WORD	+a#	+ **ind pres 3 sg**
ar	WORD	+amos#	+ **ind pres 1 pl**
ar	WORD	+'ais#	+ **ind pres 2 pl**
ar	WORD	+an#	+ **ind pres 3 pl**
ar	WORD	+'e#	+ **ind pret 1 sg**
ar	WORD	+aste#	+ **ind pret 2 sg**
ar	WORD	+'o#	+ **ind pret 3 sg**
ar	WORD	+amos#	+ **ind pret 1 pl**
ar	WORD	+asteis#	+ **ind pret 2 pl**
ar	WORD	+aron#	+ **ind pret 3 pl**

er	WORD	+o#	**+ ind pres 1 sg**
er	WORD	+es#	**+ ind pres 2 sg**
er	WORD	+e#	**+ ind pres 3 sg**
er	WORD	+emos#	**+ ind pres 1 pl**
er	WORD	+'eis#	**+ ind pres 2 pl**
er	WORD	+en#	**+ ind pres 3 pl**
er	WORD	+'i#	**+ ind pret 1 sg**
er	WORD	+iste#	**+ ind pret 2 sg**
er	WORD	+i'o#	**+ ind pret 3 sg**
er	WORD	+imos#	**+ ind pret 1 pl**
er	WORD	+isteis#	**+ ind pret 2 pl**
er	WORD	+ieron#	**+ ind pret 3 pl**
WORD			

Some spelling rules for Spanish

$\mathrm{e} \rightarrow \mathrm{ie}\ /\ \underline{\quad}\ \mathrm{C}^*\ \mathbf{feat}^*\ \mathbf{diph}\ \mathbf{feat}^* + \mathrm{V}^*\ \mathrm{C}^*\ \#$

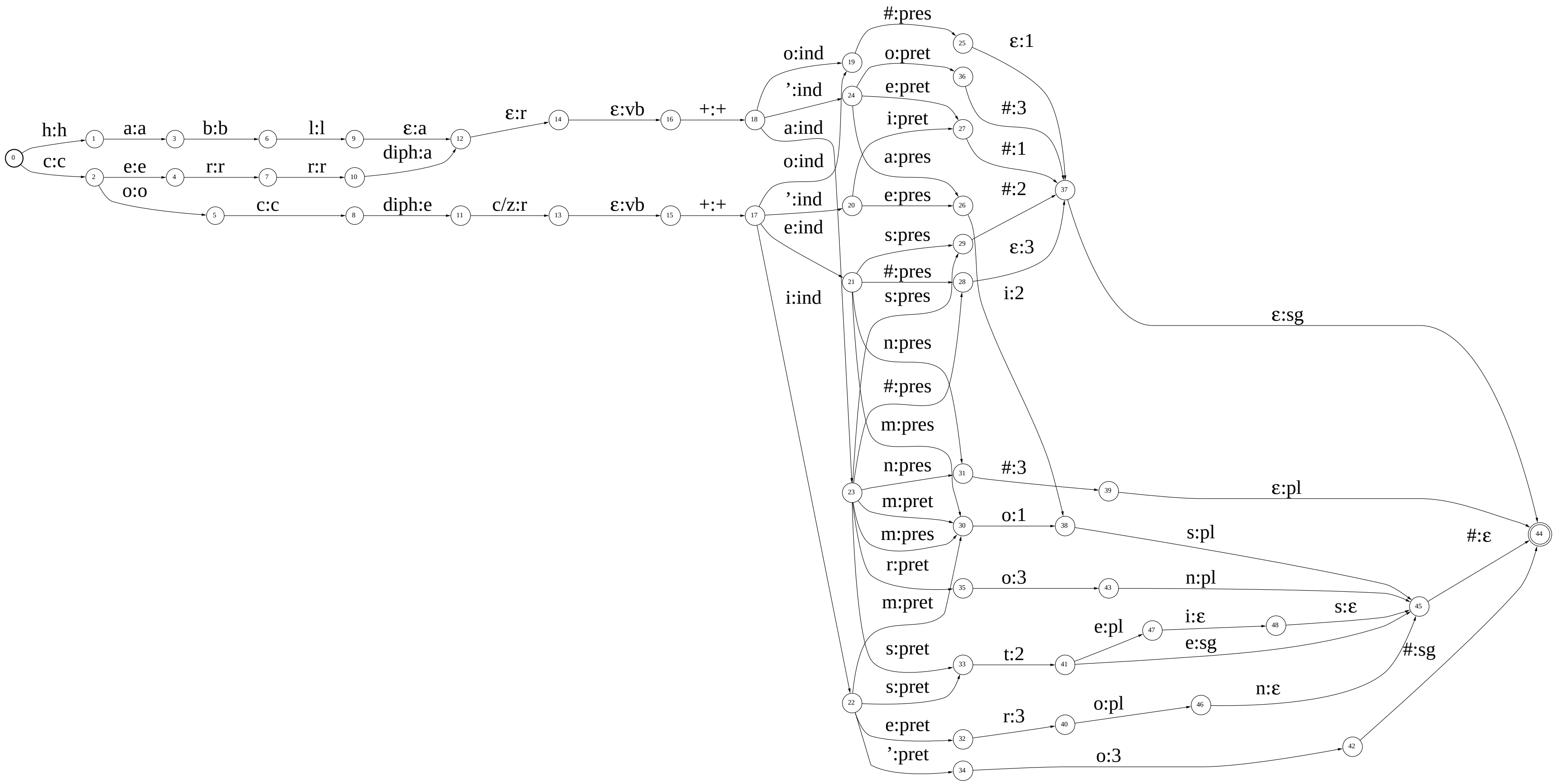
$\mathrm{o} \rightarrow \mathrm{ue}\ /\ \underline{\quad}\ \mathrm{C}^*\ \mathbf{feat}^*\ \mathbf{diph}\ \mathbf{feat}^* + \mathrm{V}^*\ \mathrm{C}^*\ \#$

$\mathrm{c} \rightarrow \mathrm{z}\ /\ \underline{\quad}\mathbf{feat}^*\ \mathbf{c/z} + [+\mathrm{back}]$

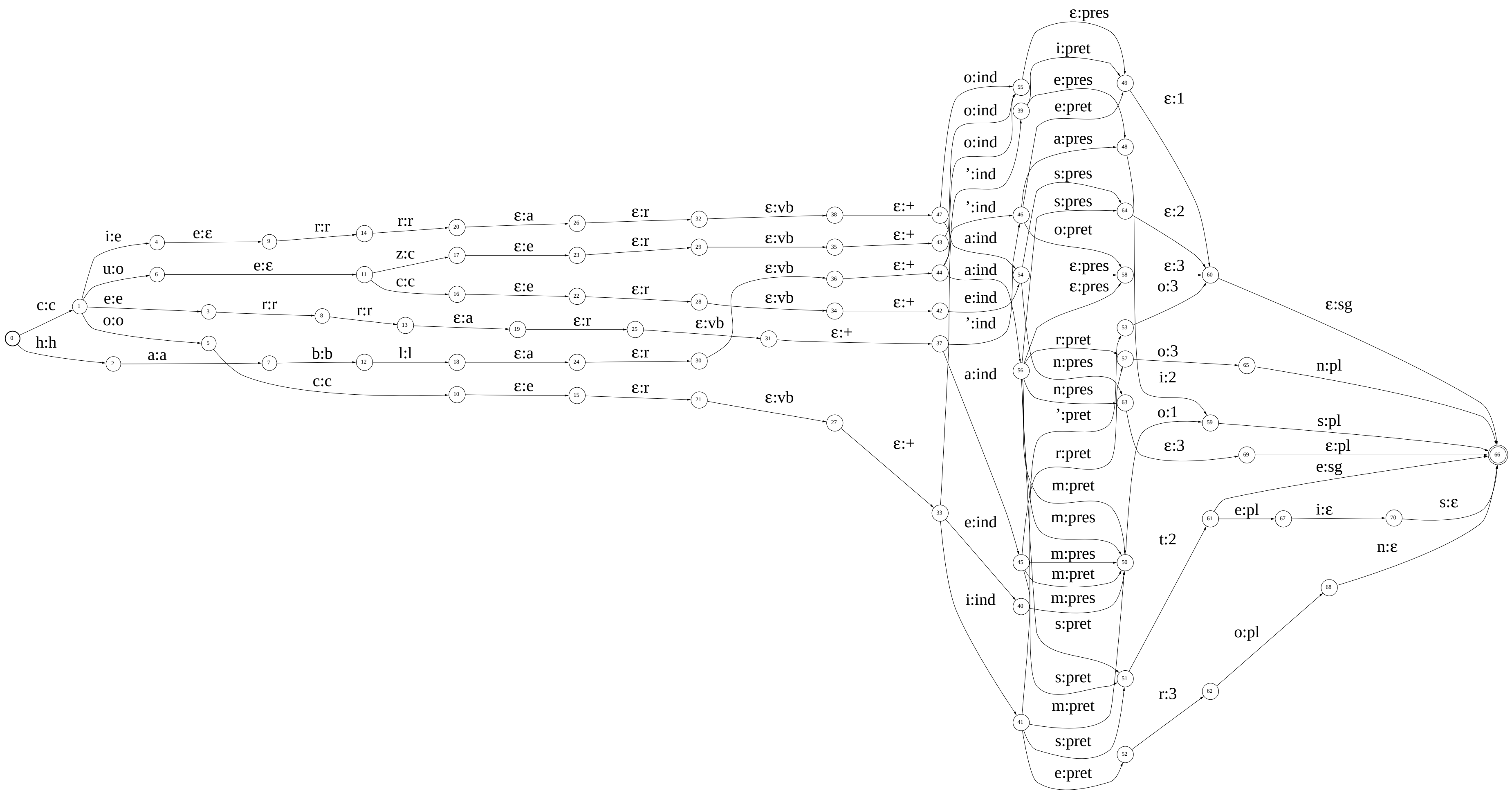
$(\mathbf{feature} \vee \mathbf{boundary}\ ) \rightarrow \epsilon$

$\boxed{\text{Call the FST representing these } R}$

Spanish lexicon transducer fragment D



Spanish lexicon transducer fragment $(D \circ R)^{-1}$



Problems with finite-state approaches

- Long-distance dependencies: *empowerable*, *enrichable* ...

  Possible solutions:

  - Use unification-based or context-free approaches
  - "Hide" the complexity using grammar tools — e.g. finite-state approximation (Pereira and Wright, 1997).

- Traditional FS models — e.g. (Koskenniemi, 1983) — are best suited for *concatenative* morphology.

  - Multi-tape approaches (Kay, 1987; Kiraz, 1996) offer one solution to this

- Reduplication seems to have no elegant analysis (cf. Warlbiri **pangu***pangurnu* 'dug repeatedly' (Nash, 1980))

Multi-tape approach (Kiraz, 1996)

kattab, Class II (perfect) form of *ktb* ‘write’

	a					*vocalism tape*
k		t			b	*root tape*
C	V	C	C	V	C	*pattern tape*
k	a	t	t	a	b	*surface tape*

Each of the correspondences between the surface and the lexical tapes is licensed by a rule

Reduplication

See appendix

Interaction of rules with grammatical features

(Kiraz, 1997) (cf. (Bear, 1988))

- English [e] epenthesis: /potato/ $\rightarrow$ /potato*e*s/, but /piano/ $\rightarrow$ /pianos/.
  $\epsilon \rightarrow$ e / ___ s [epenthesis +]
  potato [epenthesis +]
  piano [epenthesis -]

- If
  - the attributes and values in the feature structures are atomic
  - variables in feature structures can be represented as a disjunction of a *predefined* set of value specifications

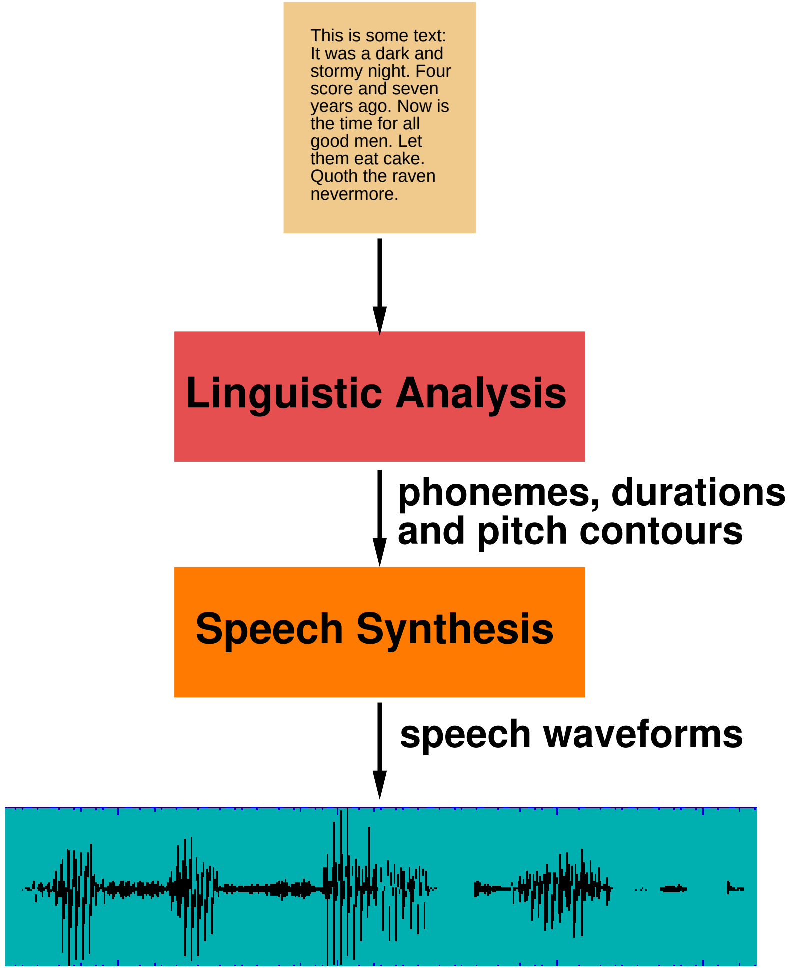
  then:

- One can compile out the feature structure as a regular expression and make it part of the (righthand) context of the rule.

PART IV:

Application 2:
Text Analysis for TTS

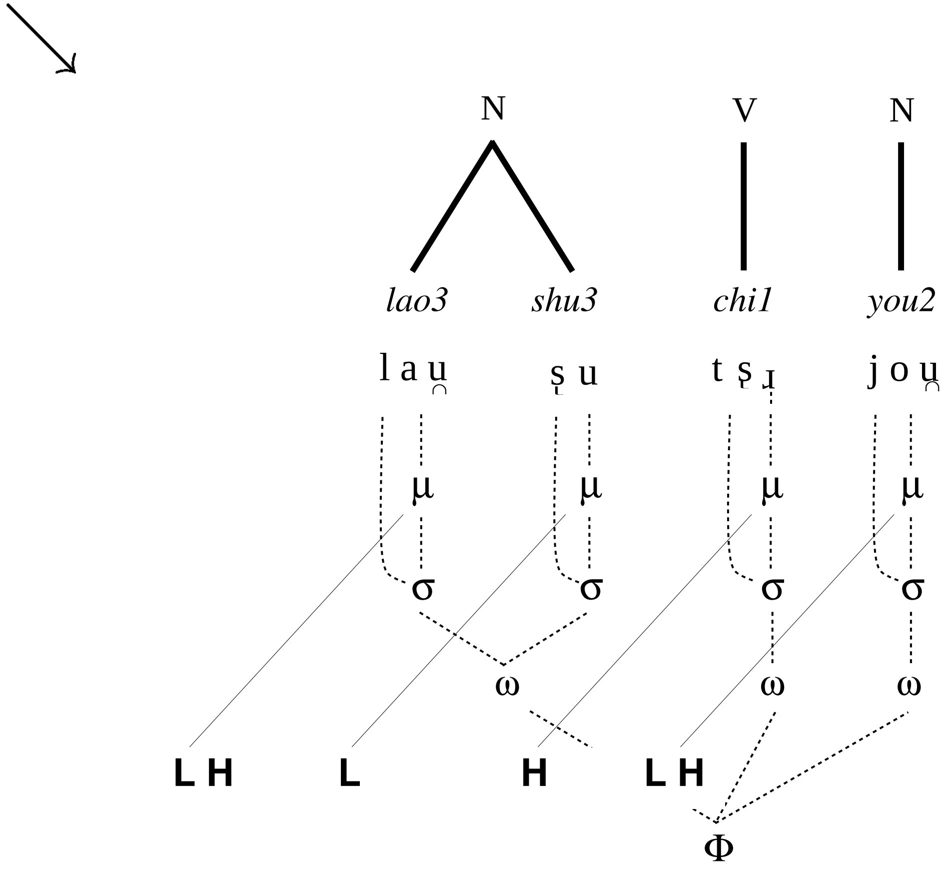
Text analysis for TTS: introduction



From text to linguistic representation

老鼠吃油。

'The rat is eating the oil'



Linguistic analysis

- Segment text into words.
- Segment text into sentences, checking for and expanding abbreviations.
- Expand numbers

- Lexical and morphological analysis
- Word pronunciation
  - Homograph disambiguation
- Prosodic Phrasing
- Accentuation

Preprocessing

Most TTS systems treat word segmentation, and 'normalization' of numbers and abbreviations in a preprocessing phase. *But:*

- word segmentation may require linguistic information that is normally only available at a 'later' stage

- 'text normalization' may require linguistic information that is normally only available at a 'later' stage

Word segmentation in Chinese

李春葉$_1$ 吃不下$_2$ 她$_3$ 的$_4$ 苦瓜$_5$

lǐ-chūn-yè chī-bú-xià tā de kǔguā

“Li Chunye can’t swallow her bitter melon”

Segmentation into words requires (at least) a model of the lexicon (3, 4, 5), morphology (2) and personal names (1).

Conclusion: for some languages lexical information is needed in order to segment words

How do you say '%' in Russian?

		Adjectival forms when modifying nouns
20% скидка	⇒	двадцат $\boxed{\text{и}}$ -процентн $\boxed{\text{ая}}$ скидка
'20% discount'		*dvadcat* $\boxed{i}$ *-procent* $\boxed{naja}$ *skidka*
с 20% раствором	⇒	с двадцат $\boxed{\text{и}}$ -процент $\boxed{\text{ным}}$ раствором
'with 20% solution'		*s dvadcat* $\boxed{i}$ *-procent* $\boxed{nym}$ *rastvorom*
		Nominal forms otherwise
21%	⇒	двадцать один процент
		dvadcat' odin procent
23%	⇒	двадцать три процент $\boxed{\text{а}}$
		dvadcat' tri procent $\boxed{a}$
20%	⇒	двадцать процент $\boxed{\text{ов}}$
		dvadcat' procent $\boxed{ov}$
с 20%	⇒	с двадцать $\boxed{\text{ю}}$ процент $\boxed{\text{ами}}$
'with 20%'		*s dvadcat'* $\boxed{ju}$ *procent* $\boxed{ami}$

Conclusion: contextual linguistic information may be required for abbreviation expansion

Furthermore . . .

A multilingual text analysis system should allow:

- for easy incorporation of models of morphological analysis of ordinary words, names, etc.
- for *ranking* of various analyses, where these analyses cannot be ruled out by categorical constraints

Chinese word segmentation again

日 文 章 魚 怎 麼 說

rìwén zhāngyú zěnmo shuō

can be segmented in at least two ways:

日$_1$ | 文 章$_2$ | 魚$_3$ | 怎 麼 | 說

'?how do you say "fish$_3$" in a Japanese$_1$ essay$_2$?'

日 文$_1$ | 章 魚$_2$ | 怎 麼 | 說

'how do you say "octopus$_2$" in Japanese$_1$?'

But the second interpretation is much more likely — has lower *cost*

Summary of desiderata

- There should be no distinguished 'preprocessing' phase: expansion of abbreviations, numbers, etc., should be done concurrently with other lexical analysis

- Easy incorporation of morphological models

- *Weights* or *costs*, for ranking alternative analyses

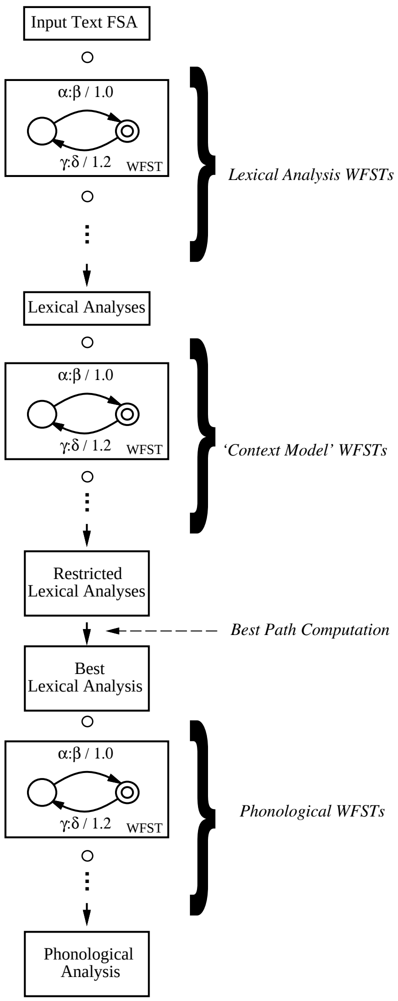
Weighted Finite State Transducers offer an attractive computational model

Model for TTS

(Sproat, 1997b; Sproat, 1997c)

- input text represented as an FSA
- lexical analyses derived by composing input with one or more WFSTs
- lexical analyses further restricted by composing them with one or more 'context model' WFSTs
- *phrasing and accentuation are handled as part of the context modeling*
- best restricted lexical analysis is then composed with one or more phonological WFSTs

Model


Input Text FSA
α:β / 1.0
γ:δ / 1.2
WFST
Lexical Analysis WFSTs
Lexical Analyses
α:β / 1.0
γ:δ / 1.2
WFST
'Context Model' WFSTs
Restricted Lexical Analyses
Best Path Computation
Best Lexical Analysis
α:β / 1.0
γ:δ / 1.2
WFST
Phonological WFSTs
Phonological Analysis

Finer-grained structure of lexical analysis WFST: 1

$$\left\|\begin{array}{c|c|c} \boxed{\text{Words : Lex. Anal.}} & & \boxed{\text{Punc. :Interp.}} \\ \bigcup & & \bigcup \\ \boxed{\text{Special Symbols : Expansions}} & \cdot & \boxed{\mathit{SPACE}\ \text{:Interp.}} \\ \bigcup & & \\ \boxed{\text{Numerals : Expansions}} & & \end{array}\right\|^{*}$$

SPACE: white space in English, Spanish, Russian ...

ϵ in Japanese, Chinese ...

Mandarin segmentation

我	:	我pro$_{4.88}$	wǒ	‘I’
忘	:	忘vb$_{11.77}$	wàng	‘forget’
不	:	不adv$_{4.58}$	bù	‘not’
了解	:	了$_2$解$_1$vb$_{8.11}$	liǎojiě	‘understand’
忘不了	:	忘vb+不了$_2$npot$_{12.23}$	wàng+bùliǎo	‘unable to forget’
解放	:	解$_1$放nc$_{10.92}$	jiěfàng	‘liberation’
放大	:	放大$_1$vb$_{10.70}$	fàngdà	‘enlarge’
大街	:	大$_1$街nc$_{11.45}$	dàjiē	‘avenue’

BestPath(我忘不了解放大街在那裡 $\circ$ Dictionary*) =
我pro$_{4.88}$#忘vb+不了$_2$npot$_{12.23}$# 解$_1$放nc$_{10.92}$大$_1$街nc$_{11.45}$ . . .
‘I couldn’t forget where Liberation Avenue is.’

Russian percentage expansion

с 5% скидкой

$\circ$

$\boxed{\textbf{Lexical Analysis FST}}$

$\Downarrow$

$$\boxed{\begin{array}{c} \mathrm{s_{prep}\ pjat_{num}{'}_{nom}\text{-}procentn_{adj}}\boxed{\star}\mathrm{+aja_{fem+sg+nom}\ skidk_{fem}oj_{sg+instr}} \cup \\ \boxed{\mathrm{s_{prep}\ pjat_{num}i_{gen}\text{-}procentn_{adj}}\boxed{\star}\mathrm{+oj_{fem+sg+instr}\ skidk_{fem}oj_{sg+instr}\ 2.0}} \cup \\ \mathrm{s_{prep}\ pjat_{num}{'}ju_{instr}\text{-}procent_{noun}+ami_{pl+instr}\ skidk_{fem}oj_{sg+instr}\ 4.0} \cup \\ \vdots \end{array}}$$

$\circ$

Language Model FSTs:

$$\epsilon \rightarrow \boxed{\star} \ / \ \text{procent}_{\text{noun}} \ (\Sigma \cap \neg\#)^* \ \#\_\_ \ (\Sigma \cap \neg\#)^*_{\text{noun}}$$

$$\boxed{\star} \rightarrow \epsilon \ / \ \text{procentn}_{\text{adj}} \ (\Sigma \cap \neg\#)^* \ \#\_\_ \ (\Sigma \cap \neg\#)^*_{\text{noun}}$$

$\circ$

$$\epsilon \rightarrow \boxed{\star} \ / \ \text{procentn} \ (\Sigma \cap \neg\#)^*{}_{\text{Case}\cap\neg\text{instr}} \# \ \_\_ \ (\Sigma \cap \neg\#)^*_{\text{instr}}$$

$$\epsilon \rightarrow \boxed{\star} \ / \ \text{procentn} \ (\Sigma \cap \neg\#)^*{}_{\text{sg+Case}} \# \ \_\_ \ (\Sigma \cap \neg\#)^*_{\text{pl}}$$

$\circ$

$$\neg(\Sigma^* \boxed{\star} \Sigma^*)$$

$\Downarrow$

$$\text{s pjati}_{\text{gen}}\text{-procentn}_{\text{adj}}\text{oj}_{\text{sg+instr}} \ \text{skidkoj}$$

Percentage expansion: continued

с 5% **скидкой**

$\Downarrow$

s pjati$_{\mathrm{gen}}$-procentn$_{\mathrm{adj}}$oj$_{\mathrm{sg+instr}}$ skidkoj

$\circ$

$\boxed{\mathbf{L} \circ \mathbf{P}}$

$\Downarrow$

s pʲitʲí prʌtˢɛ́ntnəj skʲitkəj

Bell Labs tools and their uses

	Lexical Analysis			Contextual Analysis			Phonological Analysis	
	Words & Names	*Abbreviations*	*Numbers*	*Disambiguation*	*Accent*	*Phrasing*	*'Letter to Sound'*	*Syllabification*
compwl *(regular expression compiler)*	●	●						●
paradigm *(morphological paradigm compiler)*	●							
arclist *(finite−state grammar compiler)*	●							
numbuilder *(numeral−expansion compiler)*			●					
rulecomp *(rewrite−rule compiler)*	●		●	●	●		●	
tree2trans *(decision−tree compiler)*						●		
––– *(decision−list compilation)*				●		●		

Numeral expansion

$$234 \quad \circ \quad \boxed{\text{Factorization}} \quad \Rightarrow \quad 2 \cdot 10^2 + 3 \cdot 10^1 + 4$$

$$\circ \quad \boxed{\text{NumberLexicon}}^{*} \quad \Downarrow$$

two hundred thirty four

兩百三十四

Languages and sizes

Sizes of some selected lexical analysis WFSTs

	States	Arcs
Spanish	31,601	71,260
Mandarin	45,806	277,432
Japanese	93,720	415,381
German	111,190	623,643
Italian	112,659	269,487
Russian	139,592	495,847

Treatment of ordinary words: a closer look

- Text form: **костра**<kostra> (bonfire+genitive.singular)

- Morphological analysis:
  кост′{Ё}**р**{noun}{masc}{inan}+**′а**{sg}{gen}

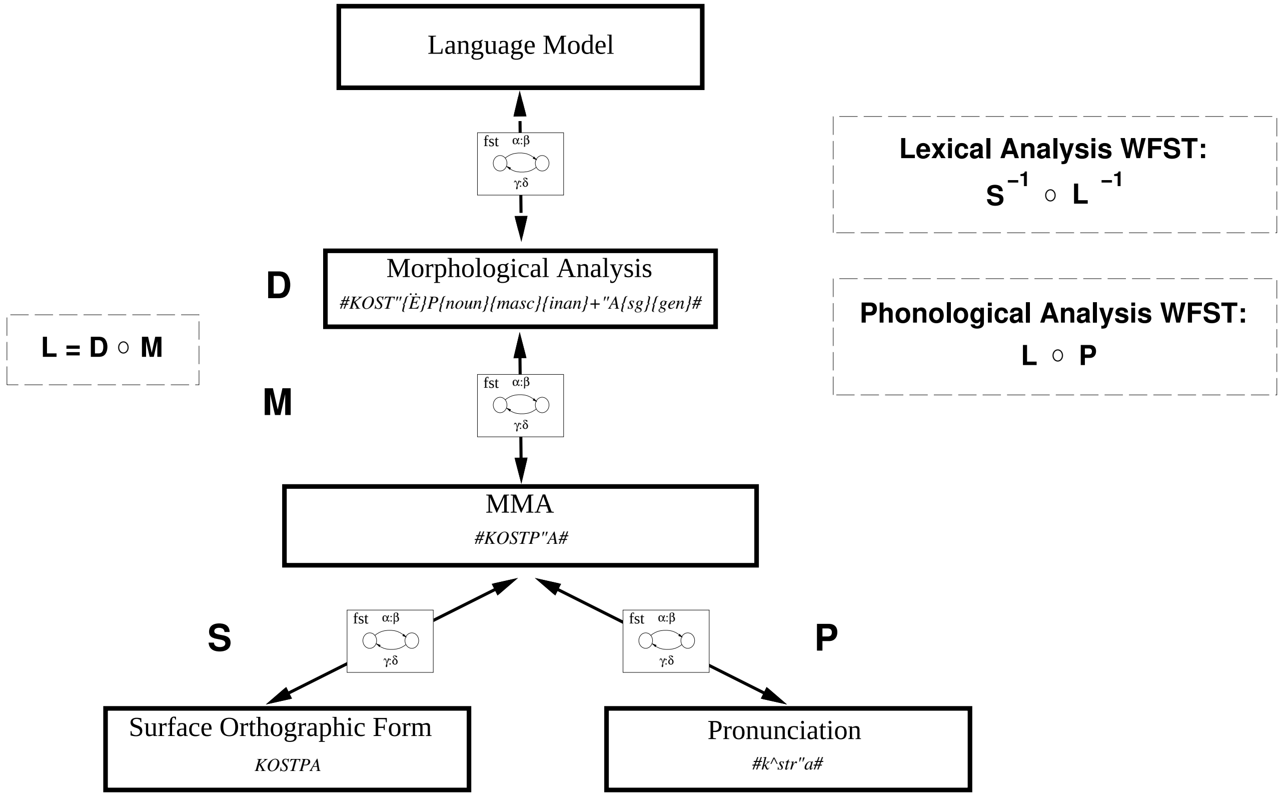
- Pronunciation: /kʌstr′a/

- ‘Minimal morphologically-motivated annotation (MMA)’: **костр′а**

Each of these levels can be related by (W)FSTs.

Conceptual structure for analysis of ordinary words

Levels of representation of the Russian word костра /kΛstr′a/ (bonfire's).



Finer-grained structure of lexical analysis WFST: 2

$$\left[\begin{array}{c|c|c} \boxed{\boxed{S^{-1}} \circ \boxed{L^{-1}_{word}}} & \cdot & \boxed{L^{-1}_{punc}} \\ \cup & & \cup \\ \boxed{L^{-1}_{special\ symbol}} & & \boxed{L^{-1}_{\mathrm{SPACE}}} \\ \cup & & \\ \boxed{L^{-1}_{numbers}} & & \\ \vdots & & \end{array}\right]^{*}$$

PART V: Writing Systems

Models of text analysis, and models of reading

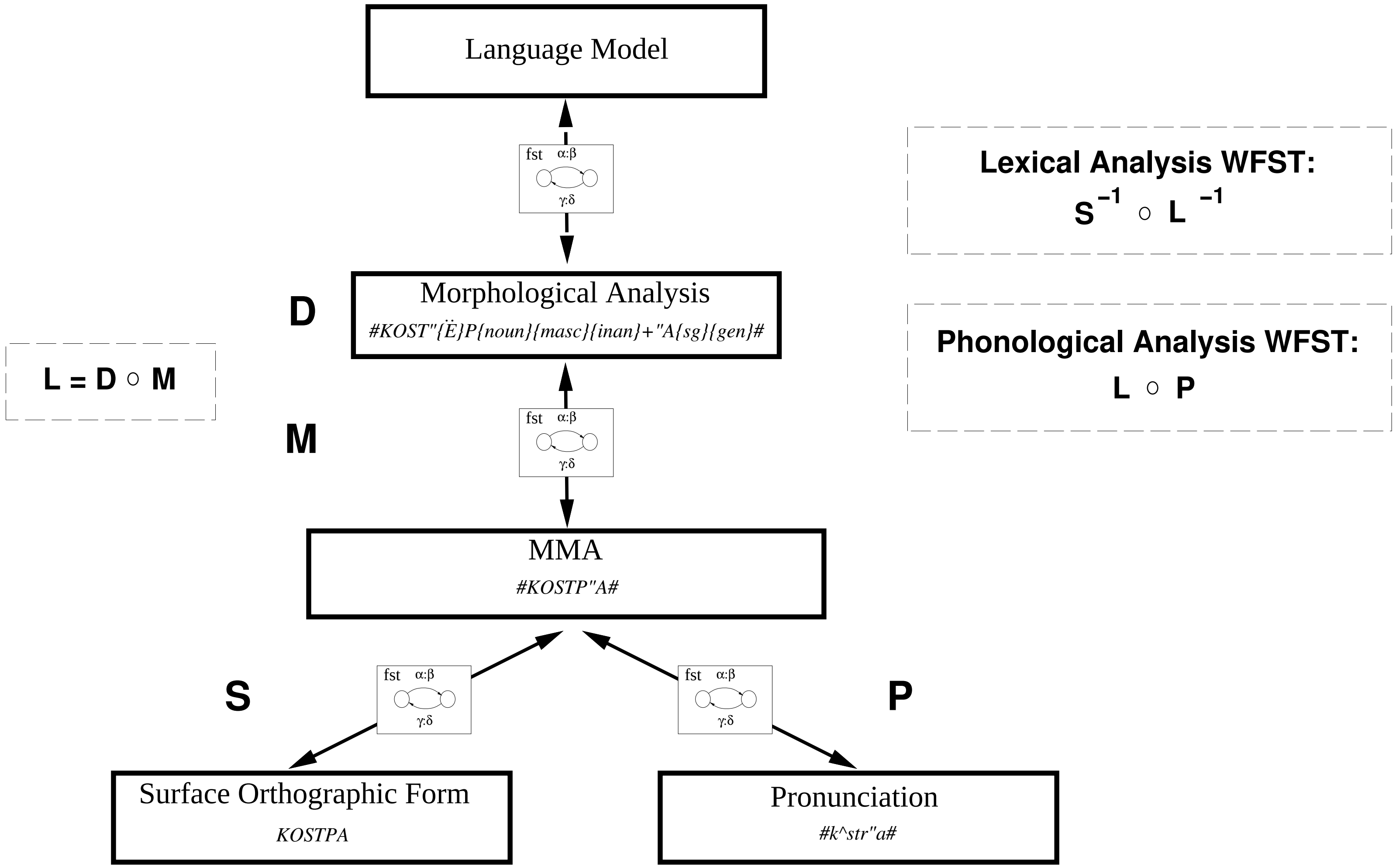
- Most work on "letter-to-sound" conversion (at least implicitly) treats orthography as a kind of glorified phonetic transcription (Sejnowski and Rosenberg, 1987; Adamson and Damper, 1996; Luk and Damper, 1993; Luk and Damper, 1996)

- But we know that this is not correct: cf. stress placement in Russian, which is highly dependent on lexical information that is not directly represented in the orthography:

карандаш	*karandáš*	'pencil'
город	*górod*	'city'
города	*góroda*	'city's'
города	*gorodá*	'cities'

- The model presented here takes a less naive view

The text analysis model again



Some assumptions implicit in the model

- Orthography relates neither directly to phonology, nor directly to lexical representation, but rather to an "annotated phonological representation"

- Furthermore, this is a universal property of *all* orthographic systems: there is no division between orthographies that represent (exclusively) phonological representation, and orthographies that represent other linguistic levels

- The mapping between orthography and linguistic representation is regular

Is a universal treatment of orthography possible?

Orthographic Depth Hypothesis (Strong Version)

In "shallow" orthographies word *naming* (pronouncing aloud) is achieved not from memory — i.e., from the lexicon — but by direct mapping to phonology. In contrast, "deep" orthographies, by necessity, require lexical access for naming in most cases.

(cf. (Turvey, Feldman, and Lukatela, 1984))

Examples of shallow orthographies:	Spanish, Serbo-Croatian
Examples of deep orthographies:	English, Chinese

Corollary

Word frequency and priming effects should be absent in naming tasks in shallow orthographies (Katz and Feldman, 1983)

Evidence against the ODH

- The evidence that lexical access does not take place when reading a shallow orthography is equivocal — see e.g. (Besner and Smith, 1992).

- Contrariwise, for deep orthographies such as Chinese one may find evidence of *pre-lexical* mapping to phonology (Tzeng and colleagues, 1994)

Conclusion: the available evidence favors a cross-linguistically uniform model of reading whereby both the lexicon and phonology are accessible from the orthography — for any orthography. In terms of the MMA model, the difference between orthographies (the "depth" of the orthographies) relates to how close the MMA is to either the lexical or phonological representation.

Tzeng et al.'s experiment

- Rapidly present a series of graphically distinct characters:

All pronounced differently:	國 — 琵 — 東	*guó—pí—dōng*
Some homophonic:	<u>皮</u> — <u>琵</u> — 東	*pí—pí—dōng*

- Subjects are required to say how many characters they saw

- Subjects report the number of distinct *syllables* they "saw", rather than the number of distinct characters

Is the mapping between orthography and linguistic representation regular?

Actually, what sense does this question make? After all:

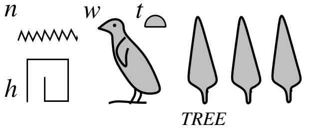
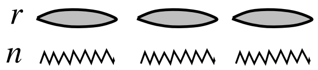
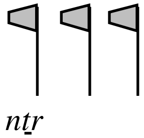
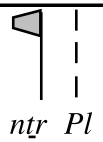
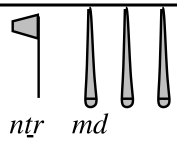
- Writing is a 2D arrangement of symbols; the symbols themselves may have a significant 2D structure (Chinese characters, Korean hangul). Also, readers do not read text "character-by-character" as an automaton does (Pollatsek and Rayner, 1990)
- Linguistic representation is hierarchical; and in some theories, at some level of abstraction, not even linearly ordered

Nevertheless:

- At some level of coarseness (most) orthographies admit of a sequential analysis
- Linguistic elements are necessarily sequentially ordered in the mapping to speech

Some apparent non-regular properties

Ancient Egyptian (Gardiner, 1957)

	$nh+w+t$ (tree+Pl+Fem) 'trees'
	$rn+w$ (name+Pl) 'names'
	$n\underline{t}r+w$ (god+Pl) 'gods'
	$n\underline{t}r+w$ (god+Pl) 'gods'
	$md+w$ $n\underline{t}r$ (word+Pl god) 'god's words'

Another case of systematic reversal

The Pahawh Hmong Messianic script

(Smalley, Vang, and Yang, 1990; Daniels and Bright, 1996)

Onsets are consistently written *after rimes*:

Soob	Lwj	yog	leej	neeg
ʃɔ́ŋ	lɨ̂	jɔ̤	lɛ̂ŋ	nɛ̤ŋ
Shong	Lue	be	CLF	person

A transducer of 1,322 states and 7,560 arcs is needed to perform the reversal for all onset-rime-diacritic pairs.

(cf. also expressions like *\$200*)

Some closing thoughts

- Finite-state models are widespread in NLP and speech technology, and are likely to become more so

- One can construct simple models using FS techniques that nonetheless have quite wide coverage

References

Adamson, Martin and Robert Damper. 1996. A recurrent network that learns to pronounce English text. In *Proceedings of the Fourth International Conference on Spoken Language Processing*, volume 3, pages 1704–1707, Philadelphia, PA. ICSLP.

Antworth, Evan. 1990. *PC-KIMMO: A Two-Level Processor for Morphological Analysis.* Occasional Publications in Academic Computing, 16. Summer Institute of Linguistics, Dallas, TX.

Bear, John. 1986. A morphological recognizer with syntactic and phonological rules. In *COLING-86*, pages 272–276, Morristown, NJ. Association for Computational Linguistics.

Bear, John. 1988. Morphology with two-level rules and negative rule features. In *COLING-88*, pages 28–31, Morristown, NJ. Association for Computational Linguistics.

Berstel, Jean and Christophe Reutenauer. 1988. *Rational Series and Their Languages.* Springer-Verlag: Berlin-New York.

Besner, Derek and Marilyn Chapnik Smith. 1992. Basic processes in reading: is the orthographic depth hypothesis sinking? In Ram Frost and Leonard Katz, editors, *Orthography, Phonology, Morphology and Meaning*, number 94 in Advances in Psychology. Elsevier, pages 45–66.

Bird, Steven and T. Mark Ellison. 1994. One-level phonology: Autosegmental representations and rules as finite automata. *Computational Linguistics*, 20(1):55–90.

Breiman, Leo, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees.* Wadsworth & Brooks, Pacific Grove CA.

Byrd, Roy, Judith Klavans, Mark Aronoff, and Frank Anshen. 1986. Computer methods for morphological analysis. In *ACL Proceedings, 24th Annual Meeting*, pages 120–127, Morristown, NJ. Association for Computational Linguistics.

Crochemore, Maxime and Wojciech Rytter. 1994. *Text Algorithms*. Oxford University Press, Oxford.

Daniels, Peter and William Bright. 1996. *The World's Writing Systems*. Oxford University Press, New York, NY.

Gardiner, Alan. 1957. *Egyptian Grammar*. Griffith Institute, Ashmolean Museum, Oxford, 3rd edition.

Grimley-Evans, Edmund, George Kiraz, and Stephen Pulman. 1996. Compiling a partition-based two-level formalism. In *Proceedings of COLING-96*, pages 454–459, Copenhagen, Denmark. COLING.

Harrison, Michael. 1978. *Introduction to Formal Language Theory*. Addison Wesley, Reading, MA.

Hopcroft, John and Jeffrey Ullman. 1979. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA.

Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. Mouton, Mouton, The Hague.

Kaplan, Ronald and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–378.

Karttunen, Lauri. 1995. The replace operator. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 16–23, Morristown, NJ. Association for Computational Linguistics.

Karttunen, Lauri and Kenneth Beesley. 1992. Two-level rule compiler. Technical Report P92–00149, Xerox Palo Alto Research Center.

Karttunen, Lauri, Ronald Kaplan, and Annie Zaenen. 1992. Two-level morphology with composition. In *COLING-92*, pages 141–148. COLING.

Katz, Leonard and Laurie Feldman. 1983. Relation between pronunciation and recognition of printed words in deep and shallow orthographies. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 9:157–166.

Kay, Martin. 1987. Nonconcatenative finite-state morphology. In *ACL Proceedings, 3rd European Meeting*, pages 2–10, Morristown, NJ. Association for Computational Linguistics.

Kiraz, George Anton. 1996. ṢemḤe: A generalized two-level system. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 159–166, Morristown, NJ. Association for Computational Linguistics.

Kiraz, George Anton. 1997. Compiling regular formalisms with rule features into finite-state automata. In *35th Annual Meeting of the Association for Computational Linguistics*, pages 329–336. Association for Computational Linguistics.

Kleene, S. C. 1956. Representation of events by nerve nets. In C. E. Shannon and J. McCarthy, editors, *Automata Studies.* Princeton University Press, Princeton, NJ, pages 129–153.

Koskenniemi, Kimmo. 1983. *Two-Level Morphology: a General Computational Model for Word-Form Recognition and Production.* Ph.D. thesis, University of Helsinki, Helsinki.

Lewis, Harry and Christos Papadimitriou. 1981. *Elements of the Theory of Computation.* Prentice-Hall, Englewood Cliffs, NJ.

Ljolje, Andrej and Michael D. Riley. 1992. Optimal speech recognition using phone recognition and lexical access. In *Proceedings of ICSLP*, pages 313–316, Banff, Canada, October.

Luk, Robert and Robert Damper. 1993. Experiments with silent-e and affix correspondences in stochastic phonographic transductions. In *Proceedings of the Third European Conference on Speech Communication and Technology*, volume 2, pages 917–920, Berlin. ESCA.

Luk, Robert and Robert Damper. 1996. Stochastic phonographic transduction for English. *Computer Speech and Language*, 10:133–153.

Magerman, David. 1995. Statistical decision-tree models for parsing. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 276–283, Morristown, NJ. Association for Computational Linguistics.

Mohri, Mehryar. 1994. Syntactic analysis by local grammars automata: an efficient algorithm. In *Papers in Computational Lexicography: COMPLEX '94*, pages 179–191, Budapest. Research Institute for Linguistics, Hungarian Academy of Sciences.

Mohri, Mehryar. 1996. On some applications of finite-state automata theory to natural language processing. *Natural Language Engineering*, 2, 61–80.

Mohri, Mehryar. 1997. Finite-state transducers in language and speech processing. *Computational Linguistics*, forthcoming.

Mohri, Mehryar and Richard Sproat. 1996. An efficient compiler for weighted rewrite rules. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 231–238, Morristown, NJ. Association for Computational Linguistics.

Nash, D. 1980. *Topics in Warlpiri Grammar*. Ph.D. thesis, The Massachusetts Institute of Technology, Cambridge, MA.

Omlin, Christian W. and C. Lee Giles. 1996. Constructing deterministic finite-state automata in recurrent neural networks. *Journal of the Association for Computing Machinery*.

Ostendorf, M. and N. Veilleux. 1994. A hierarchical stochastic model for automatic prediction of prosodic boundary location. *Computational Linguistics*, 20(1):27–54.

Pereira, Fernando and Michael Riley. 1996. Speech recognition by composition of weighted finite automata. `http://xxx.lanl.gov/ps/cmp-lg/9608018`.

Pereira, Fernando, Michael Riley, and Richard Sproat. 1994. Weighted rational transductions and their application to human language processing. In *Proceedings of the Human Language Technology Workshop*, pages 249–254, Plainsboro, NJ. ARPA.

Pereira, Fernando and Rebecca Wright. 1997. Finite-state approximation of phrase-structure grammars. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*. MIT Press, Cambridge, MA.

Pollatsek, Alexander and Keith Rayner. 1990. Eye movements and lexical access in reading. In David Balota, Giovanni Flores D'Arcais, and Keith Rayner, editors, *Comprehension Processes in Reading*. Lawrence Erlbaum Associates, Hillsdale, NJ, pages 143–163.

Riley, Michael. 1989. Some applications of tree-based modelling to speech and language. In *Proceedings of the Speech and Natural Language Workshop*, pages 339–352, Cape Cod MA, October. DARPA, Morgan Kaufmann.

Riley, Michael. 1991. A statistical model for generating pronunciation networks. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages S11.1.–S11.4. ICASSP91, October.

Ritchie, Graeme, Graham Russell, Alan Black, and Stephen Pulman. 1992. *Computational Morphology: Practical Mechanisms for the English Lexicon*. MIT Press, Cambridge, MA.

Rivest, R. 1987. Learning decision lists. *Machine Learning*, 2:229–246.

Roche, Emmanuel. 1997. Finite-state transducers: Parsing free and frozen sentences. *Journal of Natural Language Engineering*. To appear.

Salomaa, Arto and Matti Soittola. 1978. *Automata-Theoretic Aspects of Formal Power Series*. Springer-Verlag: New York.

Schützenberger, Marcel Paul. 1961. On the definition of a family of automata. *Information and Control*, 4.

Sejnowski, Terence and C. Rosenberg. 1987. Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168.

Shih, Chilin and Richard Sproat. 1996. Issues in text-to-speech conversion for Mandarin. *Computational Linguistics and Chinese Language Processing*, 1:37–86.

Smalley, William, Chia Koua Vang, and Gnia Yee Yang. 1990. *Mother of Writing: The Origin and Development of a Hmong Messianic Script.* University of Chicago Press, Chicago.

Sproat, Richard. 1997a. Computational morphology. In Robert Dale, Hermann Moisl, and Harold Somers, editors, *Handbook of Natural Language Processing.* Marcel Dekker, New York, NY. Forthcoming.

Sproat, Richard. 1997b. Multilingual text analysis for text-to-speech synthesis. *Journal of Natural Language Engineering.* To appear.

Sproat, Richard, editor. 1997c. *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach.* Kluwer, Boston, MA.

Sproat, Richard and Michael Riley. 1996. Compilation of weighted finite-state transducers from decision trees. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 215–222, Morristown, NJ. Association for Computational Linguistics.

Trost, Harald. 1990. The application of two-level morphology to non-concatenative German morphology. In *COLING-90, Volume 2*, pages 371–376, Morristown, NJ. Association for Computational Linguistics.

Turvey, M., Laurie Feldman, and Georgije Lukatela. 1984. The serbo-croatian orthography constrains the reader to a phonologically analytic strategy. In L. Henderson, editor, *Orthographies and Reading: Perspectives from Cognitive Psychology, Neuropsychology and Linguistics.* Lawrence Erlbaum Associates, Hillsdale, NJ, pages 81–89.

Tzeng, Ovid and colleagues. 1994. Paper presented at naccl v, university of southern california.

Tzoukermann, Evelyne and Mark Liberman. 1990. A finite-state morphological processor for Spanish. In *COLING-90, Volume 3*, pages 3: 277–286. COLING.

van Leeuwen, Hugo. 1989. *Too$_L$iP: A Development Tool for Linguistic Rules.* Ph.D. thesis, Technical University Eindhoven.

Voutilainen, Atro. 1994. Designing a parsing grammar. Technical Report 22, University of Helsinki.

Wang, Michelle and Julia Hirschberg. 1992. Automatic classification of intonational phrase boundaries. *Computer Speech and Language*, 6:175–196.

Yarowsky, David. 1994a. A comparison of corpus-based techniques for restoring accents in Spanish and French texts. In *Proceedings of the Second Annual Workshop on Very Large Corpora*, pages 19–32, Kyoto. ACL.

Yarowsky, David. 1994b. Decision lists for lexical ambiguity resolution. In *32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Las Cruces, NM. ACL.

Yarowsky, David. 1996. *Three Machine Learning Algorithms for Lexical Ambiguity Resolution.* Ph.D. thesis, University of Pennsylvania.

Yarowsky, David. 1997. Homograph disambiguation in text-to-speech synthesis. In Jan van Santen, Richard Sproat, Joseph Olive, and Julia Hirschberg, editors, *Progress in Speech Synthesis.* Springer, New York, NY, pages 157–172.