Phonology and speech applications with weighted automata

Natural Language Processing
LING/CSCI 5832

Mans Hulden
Dept. of Linguistics
mans.hulden@colorado.edu

Feb 19 2014

University of Colorado **Boulder**

Overview

(1) Recap unweighted finite automata
and transducers

(2) Extend to probabilistic weighted automata/transducers

(3) See how these can be used in natural language applications + a brief look at speech applications

RE: anatomy of a FSA

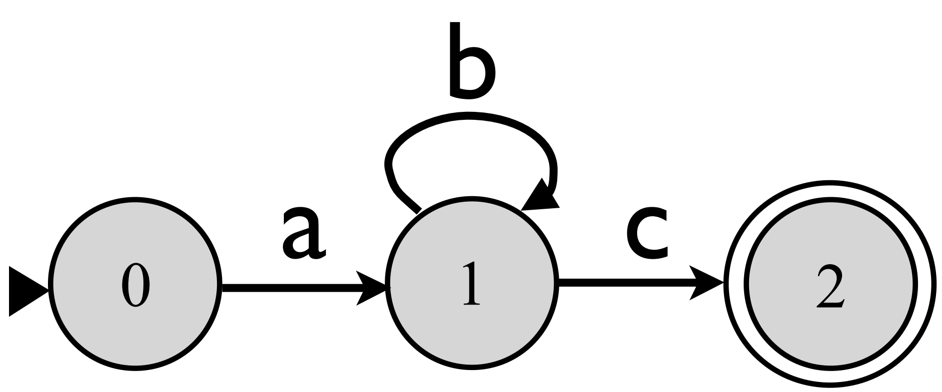
Regular expression

L = a b* c

Formal definition

$Q = \{0,1,2\}$ (set of states)
$\Sigma = \{a,b,c\}$ (alphabet)
$q_0 = 0$ (initial state)
$F = \{2\}$ (set of final states)
$\delta(0,a) = 1,\ \delta(1,b) = 1,\ \delta(1,c) = 2$
(transition function)

Graph representation

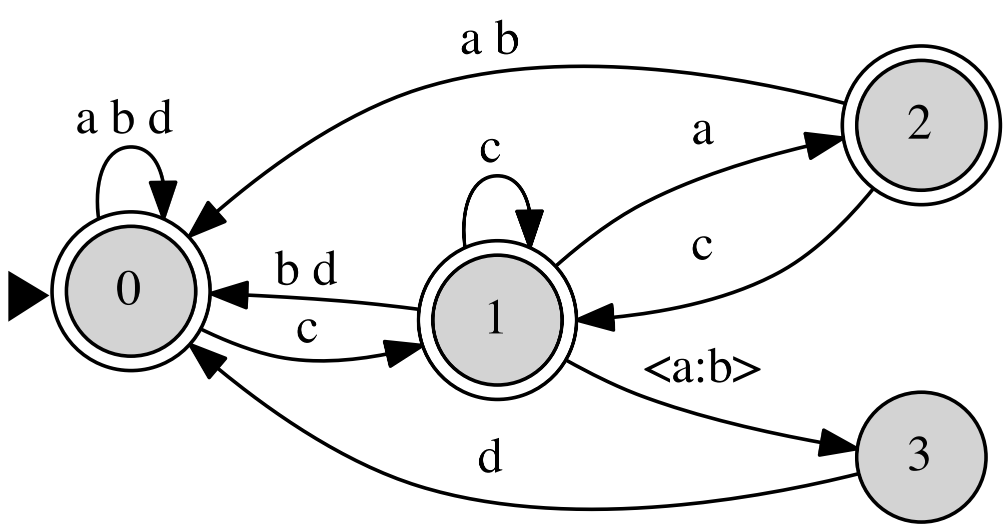


defines a set of strings

RE: anatomy of an FST

Formal definition

Graph representation



$Q = \{0,1,2,3\}$ (set of states)

$\Sigma = \{a,b,c,d\}$ (alphabet)

$q_0 = 0$ (initial state)

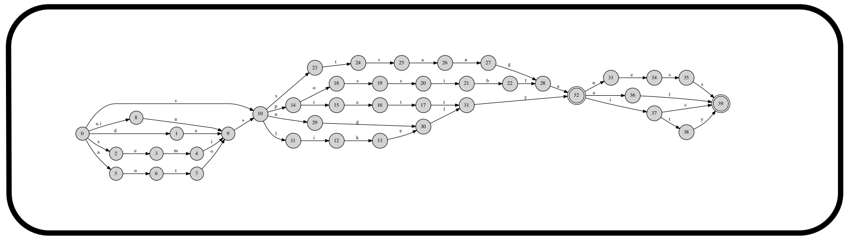
$F = \{0,1,2\}$ (set of final states)

δ (transition function)

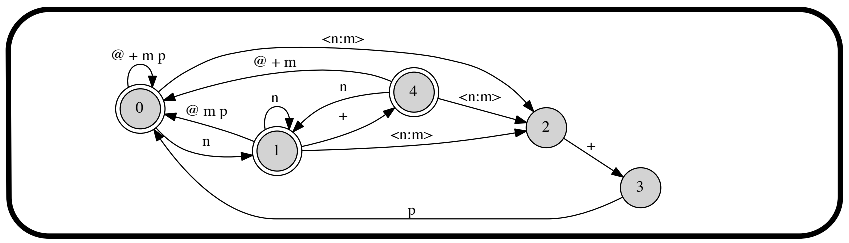
string-to-string mapping

RE: composition

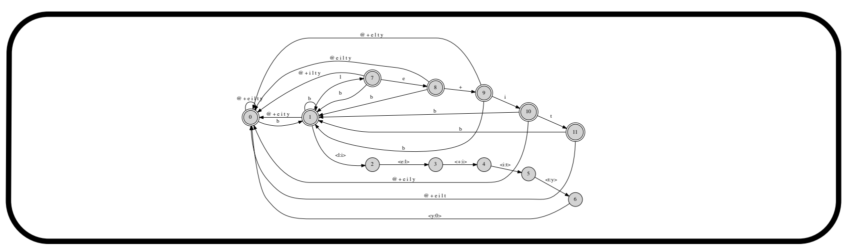
NEG+possible+ity+NOUN+PLURAL



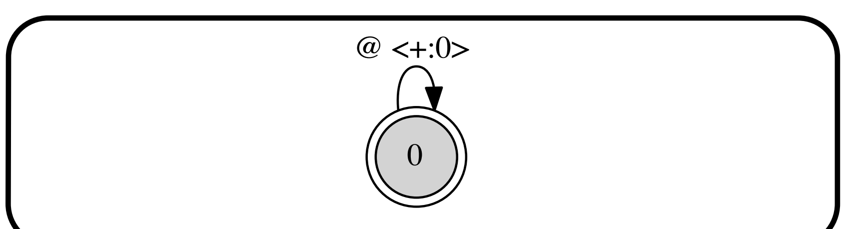
in+possible+ity+s



im+possible+ity+s

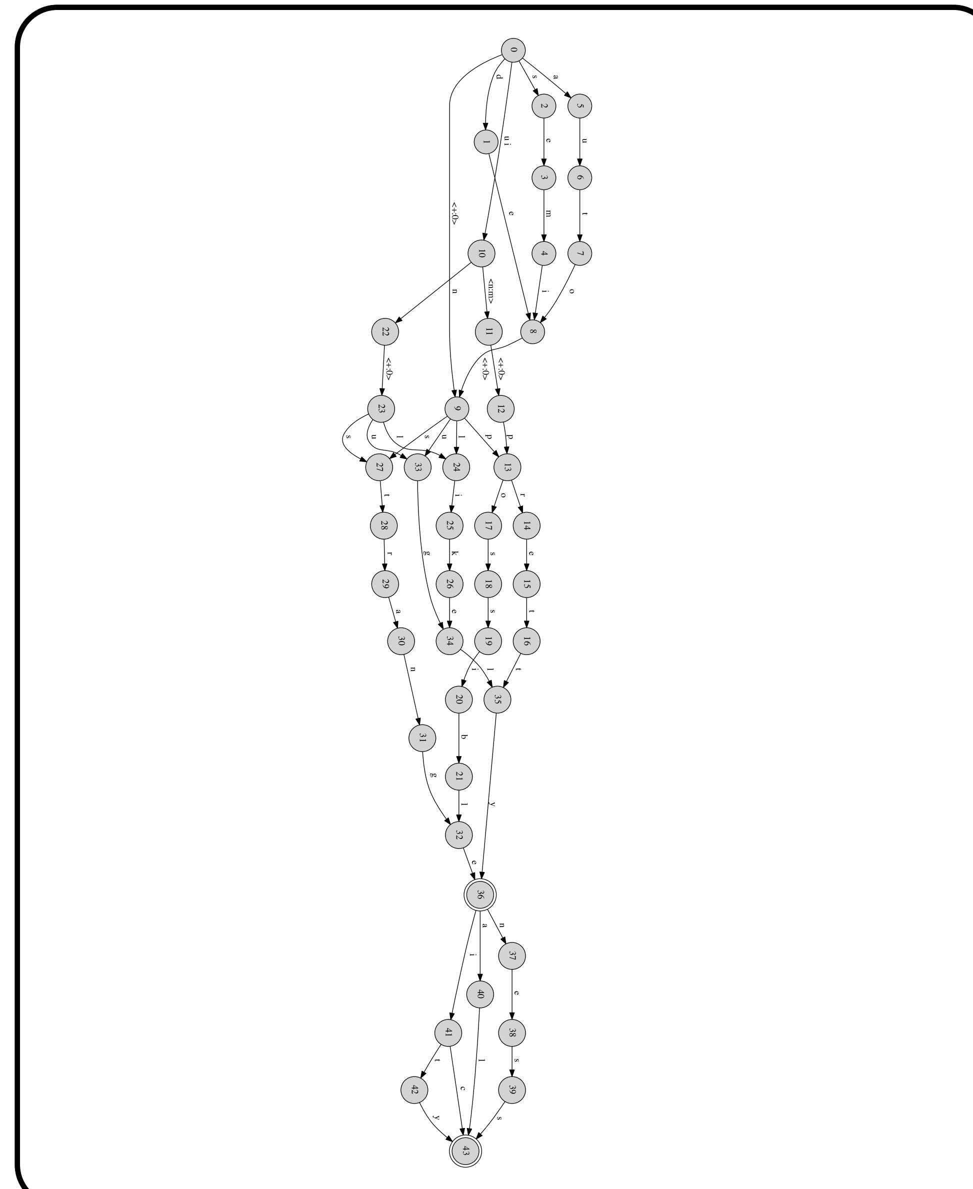


im+possibility+s



impossibilities

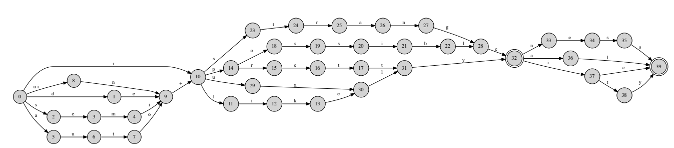
NEG+possible+ity+NOUN+PLURAL



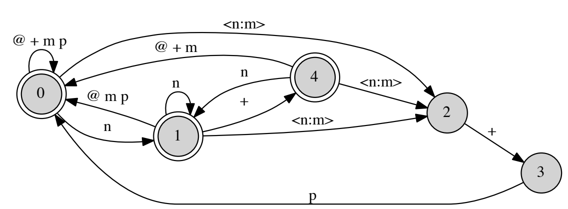
impossibilities

Orthographic vs. phonetic representation

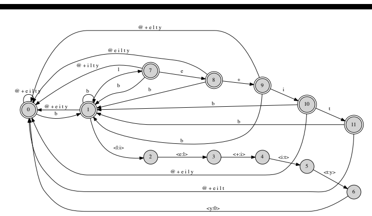
NEG+possible+ity+NOUN+PLURAL



in+possible+ity+s



im+possible+ity+s

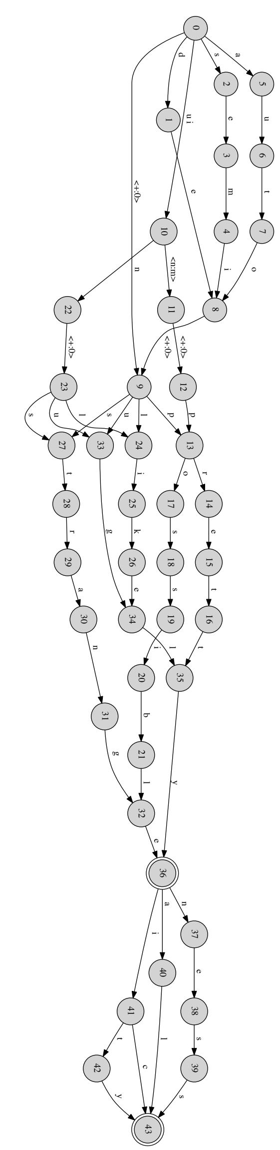


impossibilities

G2P

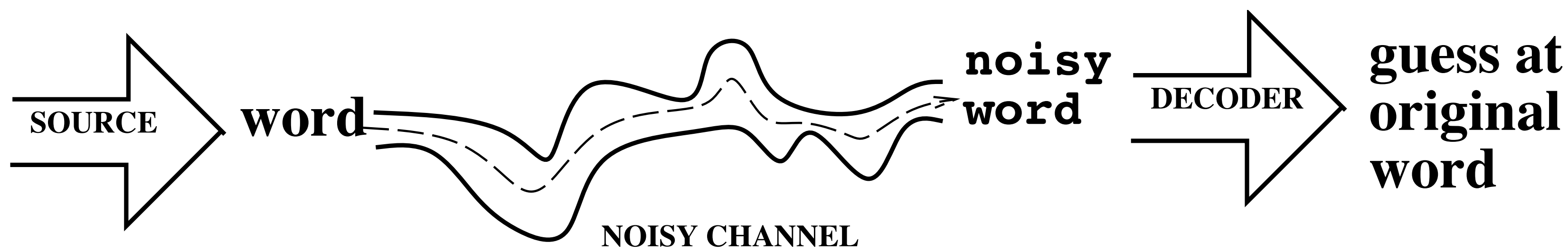
[ɪmpɑsəbɪlətis]

NEG+possible+ity+NOUN+PLURAL



[ɪmpɑsəbɪlətis]

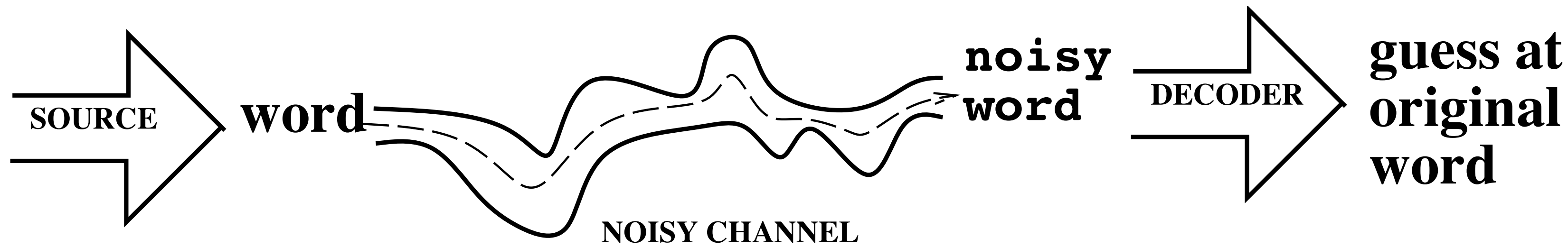
Noisy channel models



A general framework for thinking about spell checking, speech recognition, and other problems that involve decoding in probabilistic models

Similar problem to morphology 'decoding'

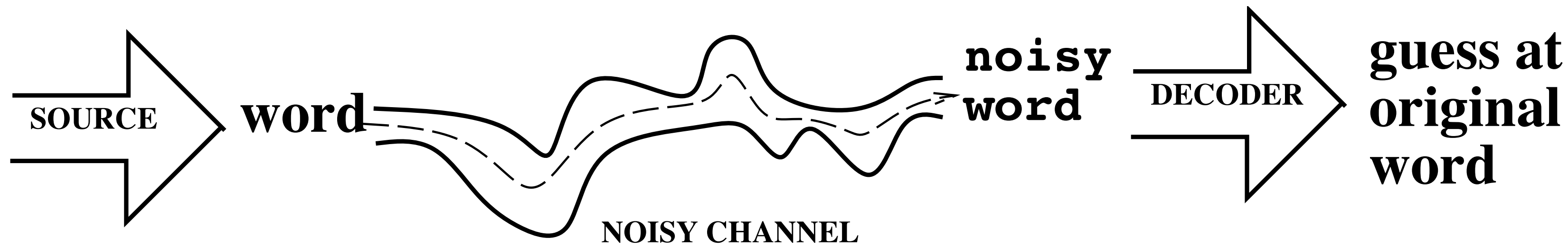
Example: spell checking



Problem form

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}}\, P(w|O)$$

Noisy channel models

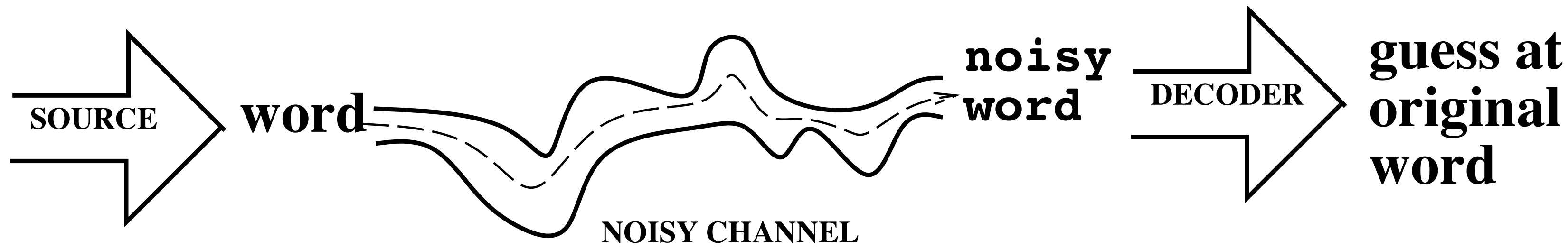


Problem form

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(w|O)$$

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$ (Bayes' Rule)

Noisy channel models

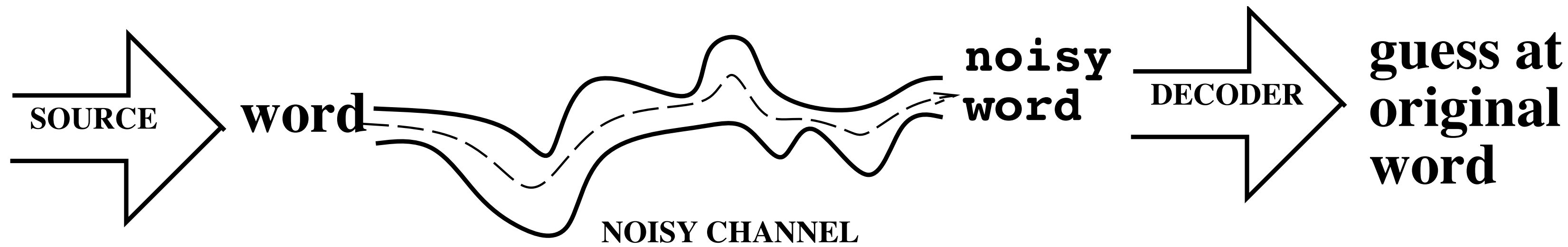


Problem form

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}}\, P(w|O)$$

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}}\, \frac{P(O|w)P(w)}{P(O)}$$

Noisy channel models



Problem form

language model

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(w|O)$$

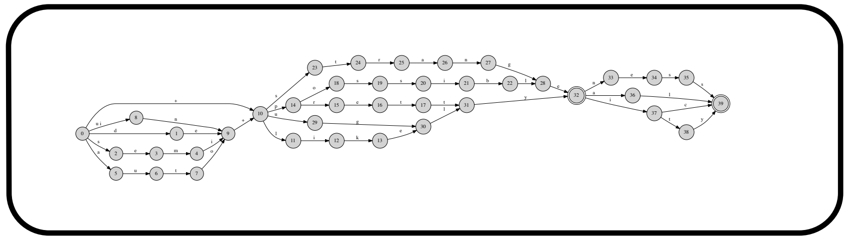
$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \frac{P(O|w)P(w)}{P(O)} = \underset{w \in V}{\operatorname{argmax}} P(O|w)\, P(w)$$

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \; \overbrace{P(O|w)}^{\text{likelihood}} \;\; \overbrace{P(w)}^{\text{prior}}$$

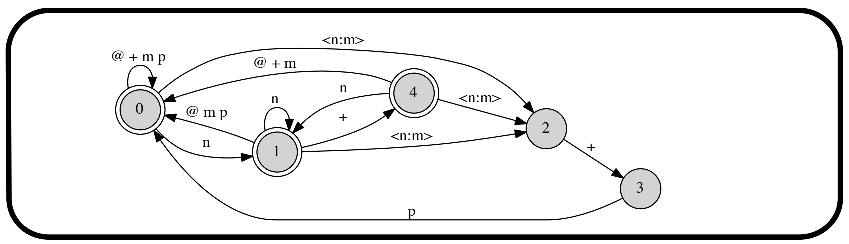
error model

Decoding

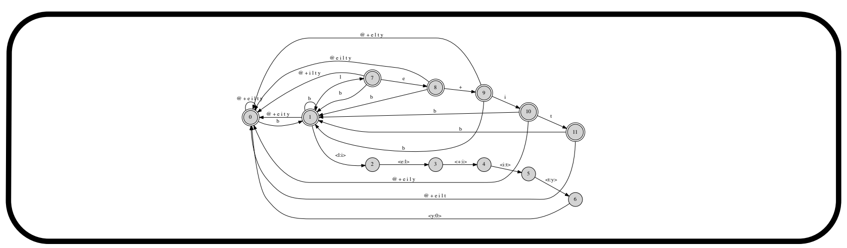
NEG+possible+ity+NOUN+PLURAL



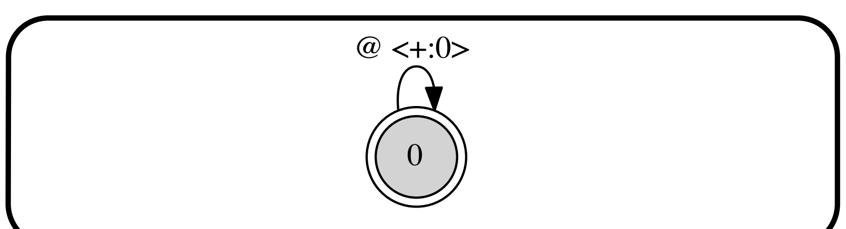
in+possible+ity+s



im+possible+ity+s

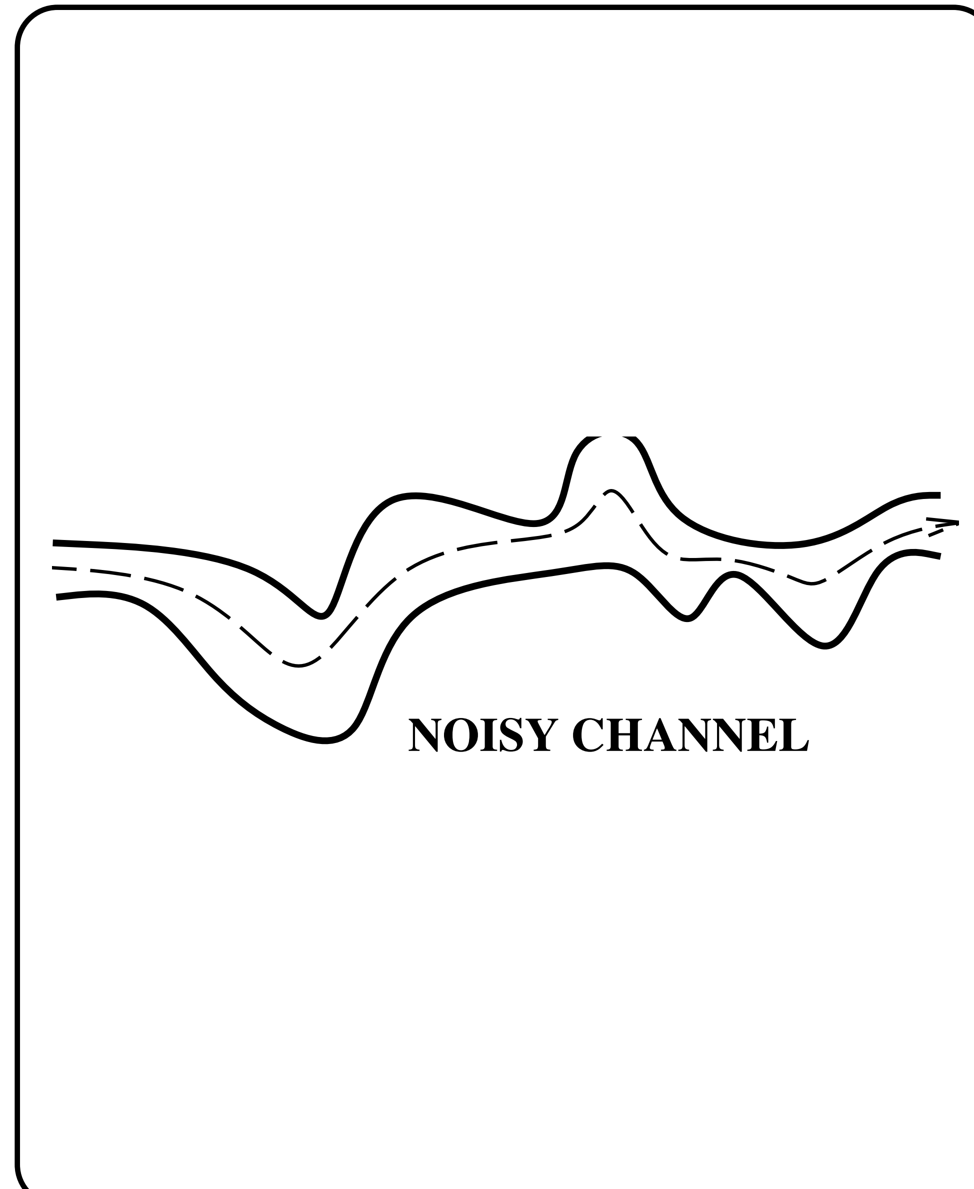


im+possibility+s



impossibilities

impossibility



impssblity

Decoding

NEG+possible+ity+NOUN+PLURAL

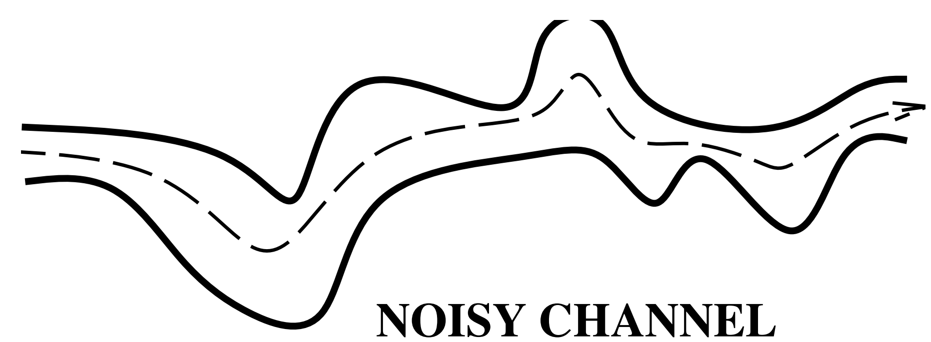
impossibility

non-probabilistic changes

Morphology/ phonology

decode

probabilistic changes (errors)



impossibilities

impssblity

Decoding/speech processing

NEG+possible+ity+NOUN+PLURAL

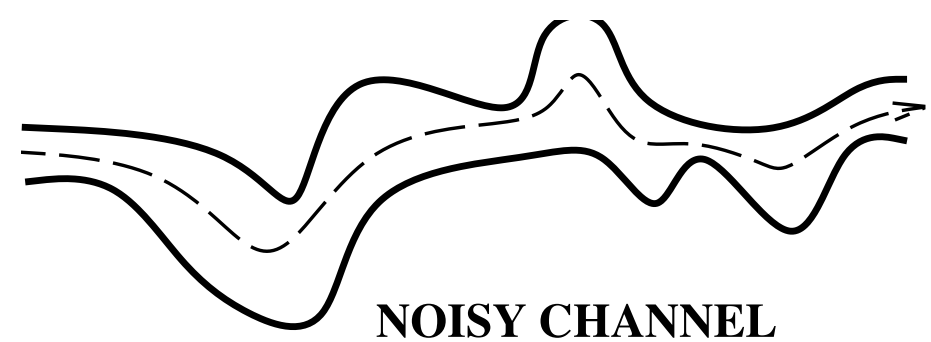
decoding is a problem

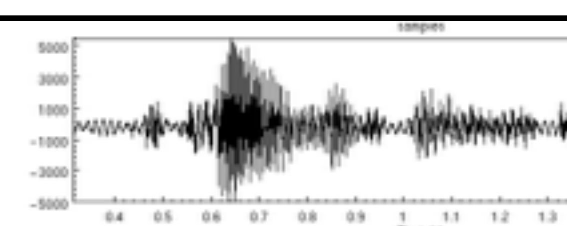
non-probabilistic changes

Morphology/ phonology

decode

probabilistic changes

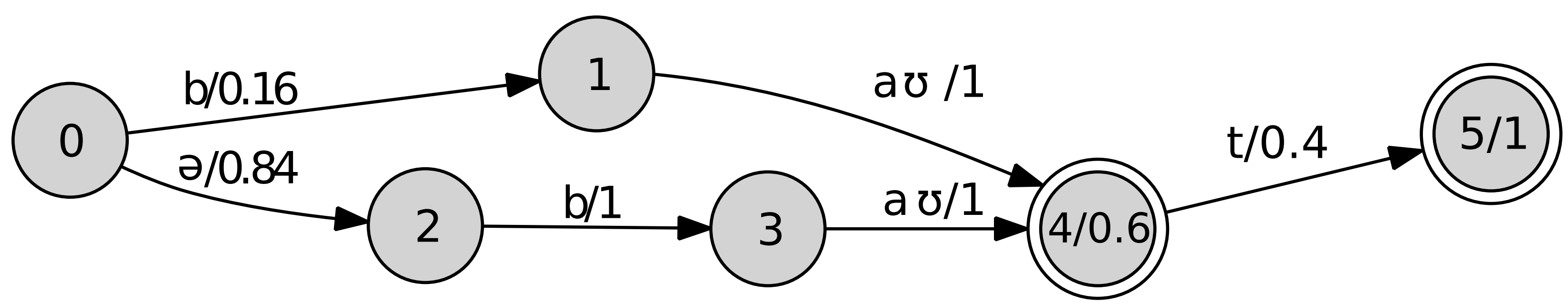




impossibilities

Probabilistic automata

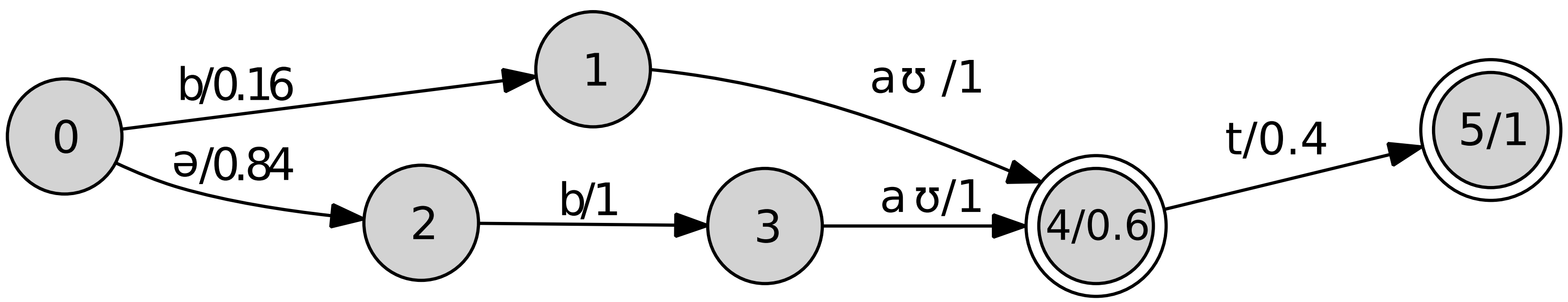
Intuition



- define probability distributions over strings
- symbols have transition probabilities
- states have final/halting probabilities
- probabilities are multiplied along paths
- probabilities are summed for several parallel paths

Probabilistic automata

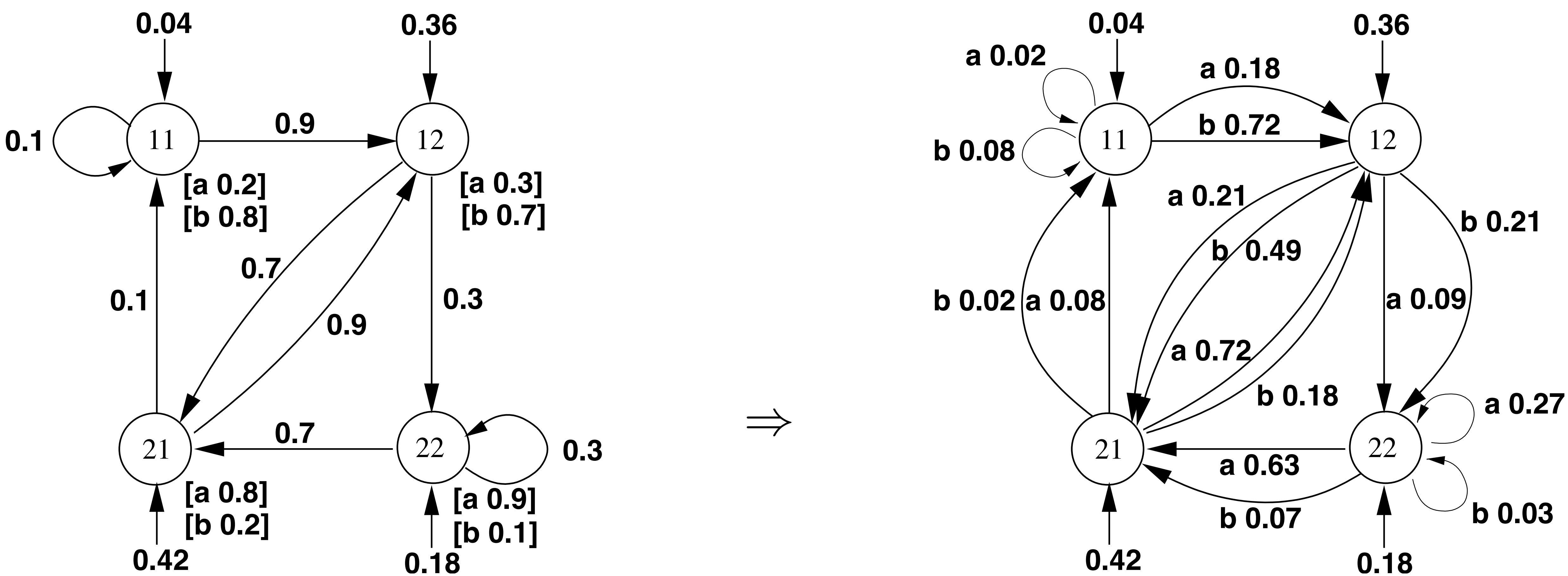
Intuition



p([əbaʊt]) = 0.336 $(0.84 \times 1 \times 1 \times 0.4 \times 1)$
p([əbaʊ]) = 0.504 $(0.84 \times 1 \times 1 \times 0.6)$
p([baʊt]) = 0.064 $(0.16 \times 1 \times 0.4 \times 1)$
p([baʊ]) = 0.096 $(0.16 \times 1 \times 0.6)$

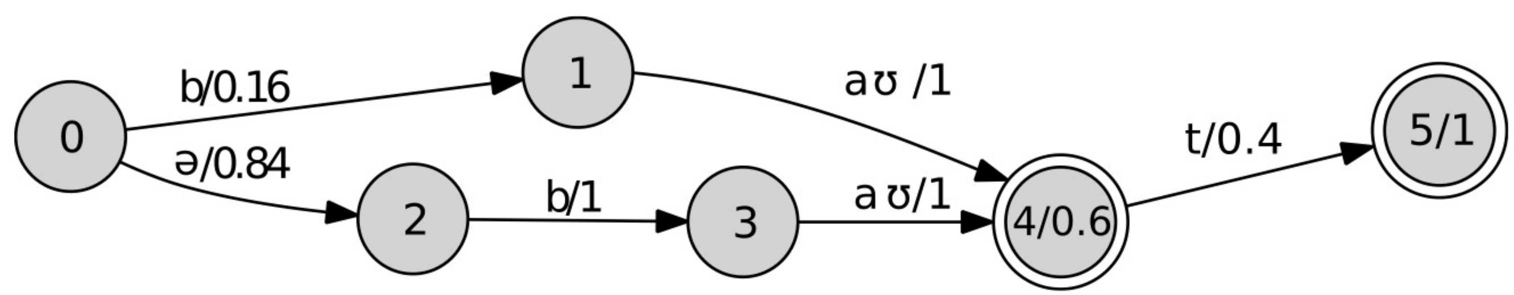
Aside: HMMs and prob. automata

Are equivalent (though automata may be more compact)



Probabilistic automata

from probabilistic to weighted



As always, we would prefer using(negative) logprobs, since this makes calculations easier:

-log(0.16) ≈ 1.8326

-log(0.84) ≈ 0.1744

-log(1) = 0

-log(0) = ∞

Since the more probable is now numerically smaller, we call them weights

Semirings

A *semiring* $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ = a ring that may lack negation.

- Sum: to compute the weight of a sequence (sum of the weights of the paths labeled with that sequence).
- Product: to compute the weight of a path (product of the weights of constituent transitions).

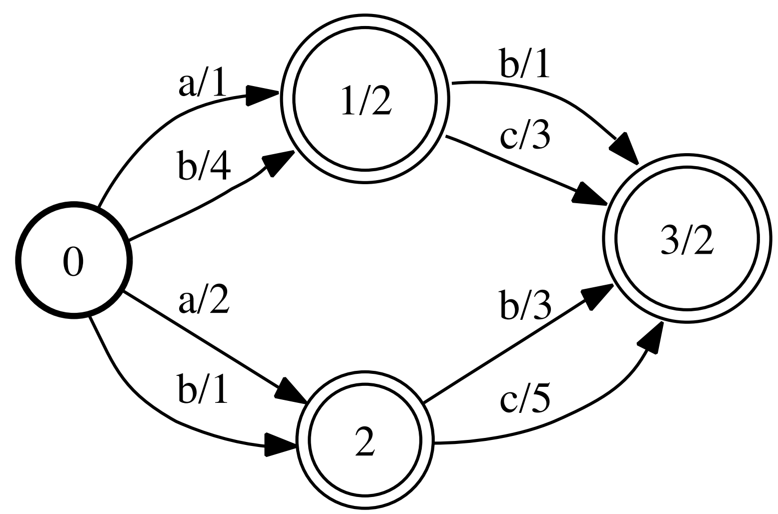
Semiring	Set	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1
Probability	$\mathbb{R}_+$	$+$	$\times$	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{\log}$	$+$	$+\infty$	0
Tropical	$\mathbb{R} \cup \{-\infty, +\infty\}$	min	$+$	$+\infty$	0
String	$\Sigma^* \cup \{\infty\}$	$\wedge$	$\cdot$	∞	ϵ

$\oplus_{\log}$ is defined by: $x \oplus_{\log} y = -\log(e^{-x} + e^{-y})$ and $\wedge$ is longest common prefix. The string semiring is a *left semiring*.

$$s \otimes \overline{1} = s \qquad s \oplus \overline{0} = s$$

$$s \otimes \overline{0} = \overline{0}$$

Semirings



Probability semiring $(\mathbb{R}_+, +, \times, 0, 1)$	Tropical semiring $(\mathbb{R}_+ \cup \{\infty\}, \min, +, \infty, 0)$
$[\![A]\!](ab) = 14$	$[\![A]\!](ab) = 4$
$(1 \times 1 \times 2 + 2 \times 3 \times 2 = 14)$	$(\min(1 + 1 + 2, 3 + 2 + 2) = 4)$

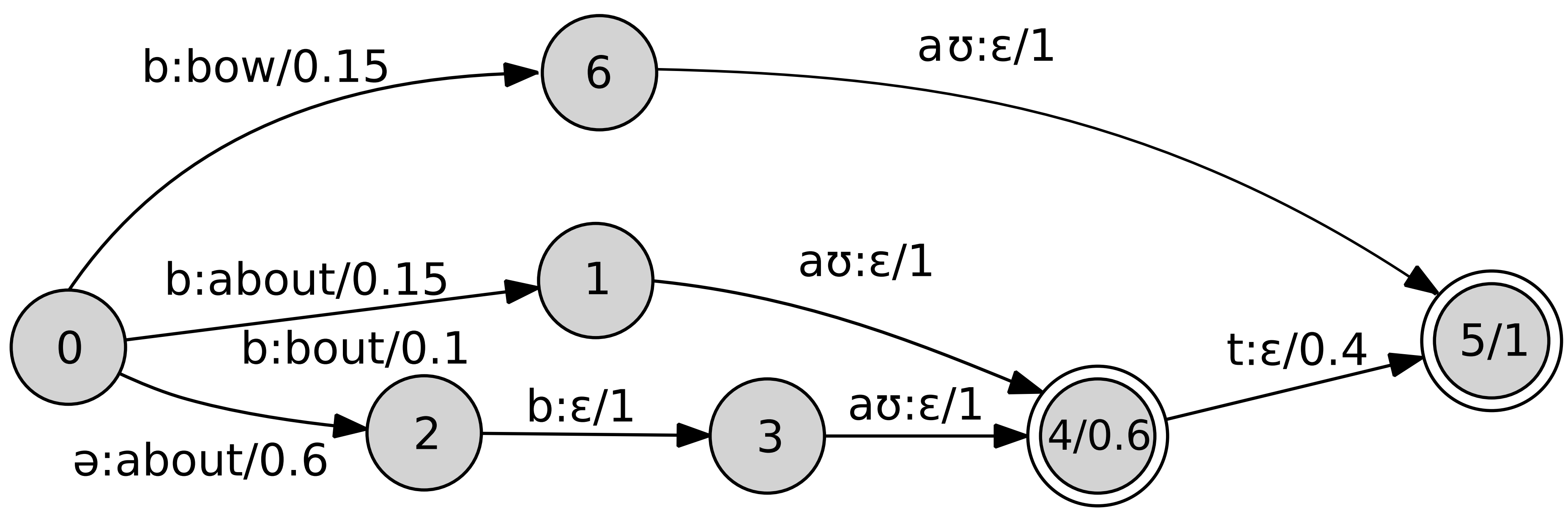
Formal definition

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

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

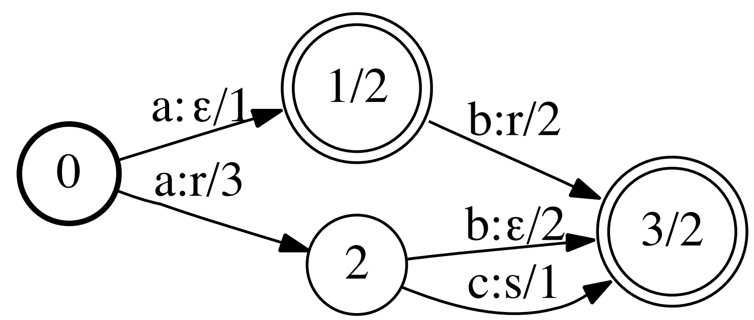
Weighted transducers

Intuition



Weighted transducers

Semirings



Probability semiring $(\mathbb{R}_+, +, \times, 0, 1)$	Tropical semiring $(\mathbb{R}_+ \cup \{\infty\}, \min, +, \infty, 0)$
$[\![T]\!](ab, r) = 16$	$[\![T]\!](ab, r) = 5$
$(1 \times 2 \times 2 + 3 \times 2 \times 2 = 16)$	$(\min(1 + 2 + 2, 3 + 2 + 2) = 5)$

Weighted transducers

Formal definition

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

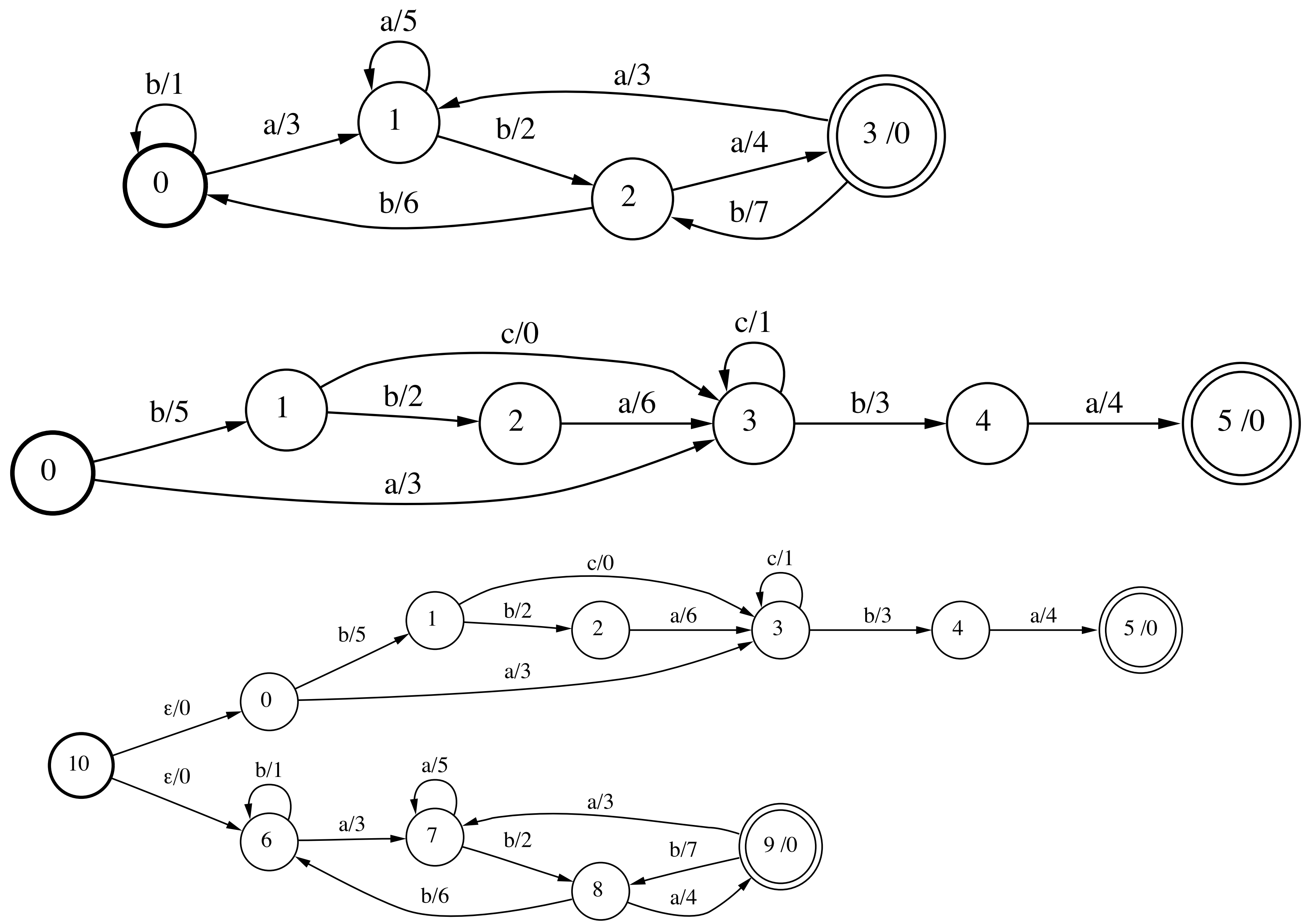
- Finite alphabets Σ and Δ,
- Finite set of states Q,
- Transition function δ: $Q \times \Sigma \to 2^Q$,
- Output function σ: $Q \times \Sigma \times Q \to \Sigma^*$,
- $I \subseteq Q$ set of initial states,
- $F \subseteq Q$ set of final states.

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

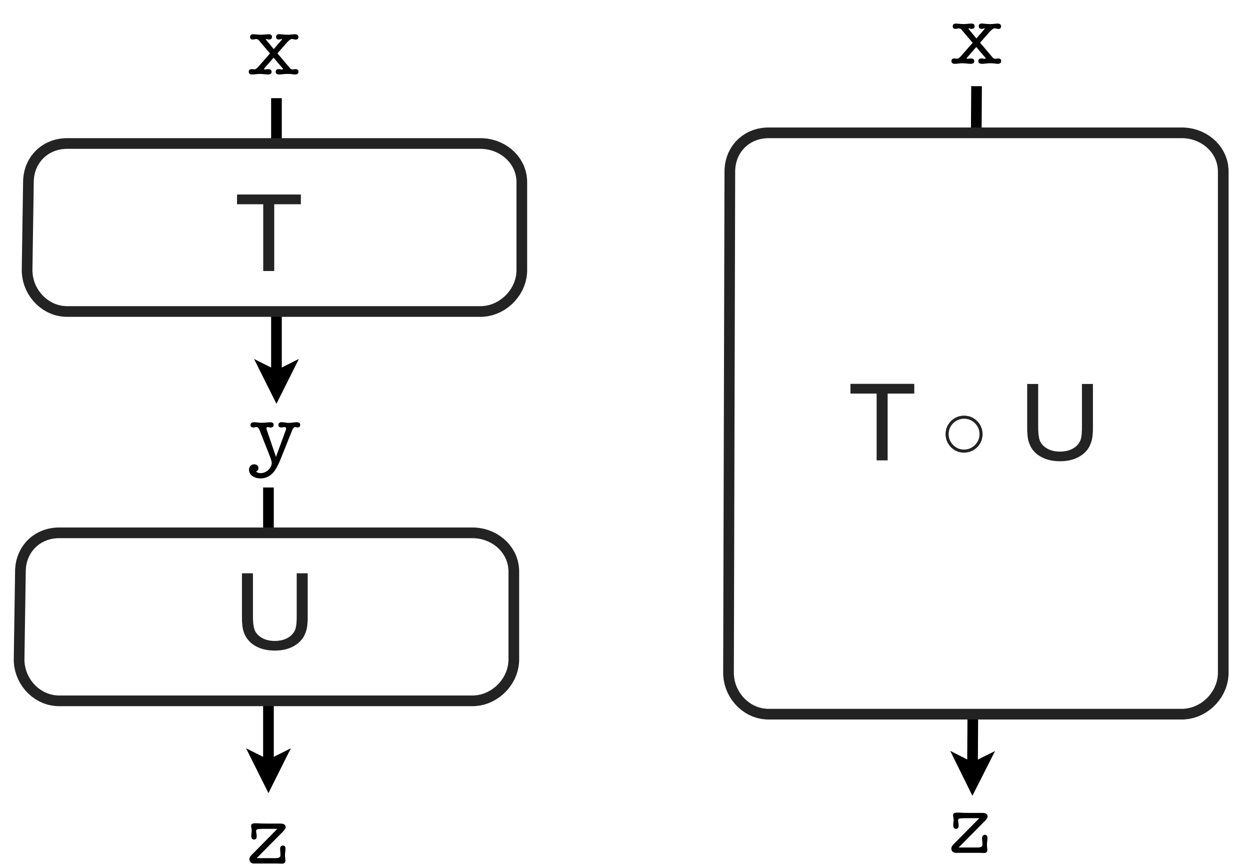
Operations on weighted automata

Booleans

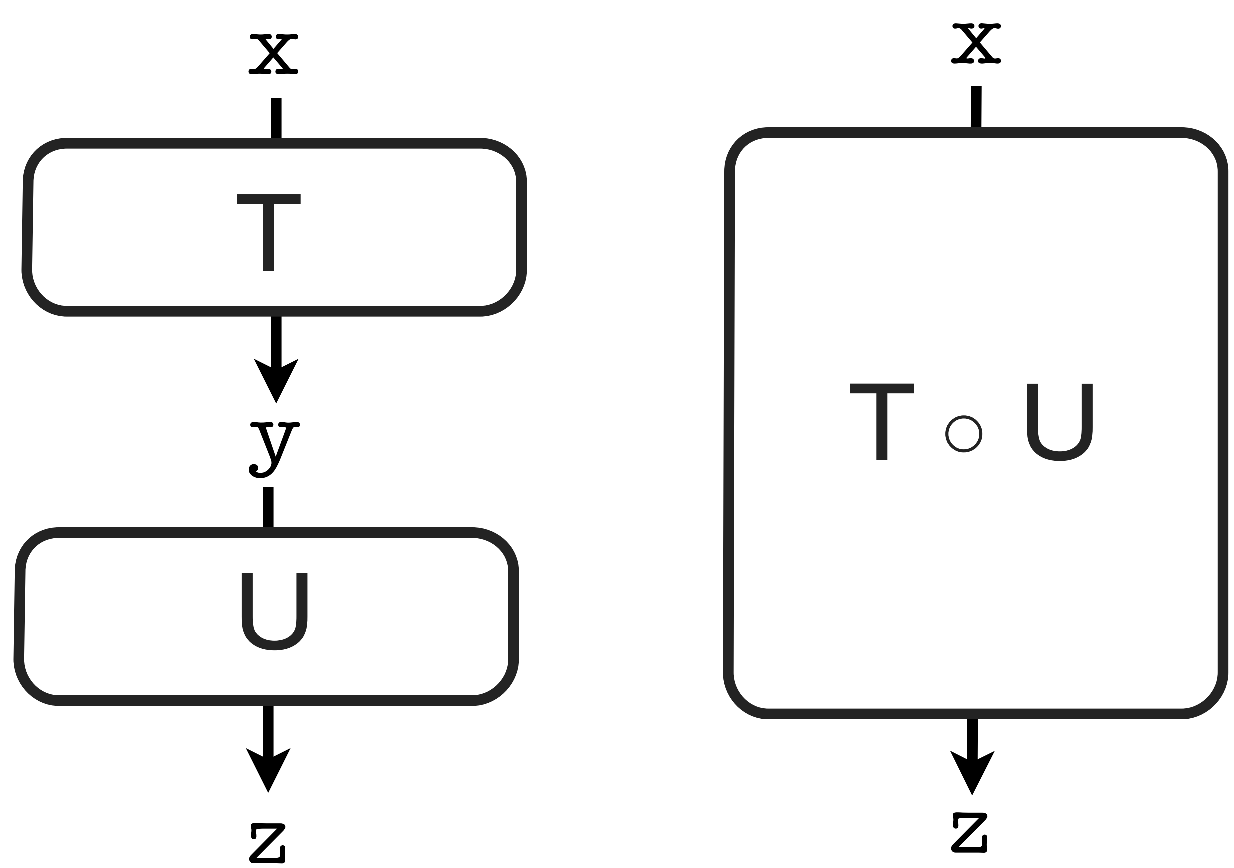
Union: Example



Composition



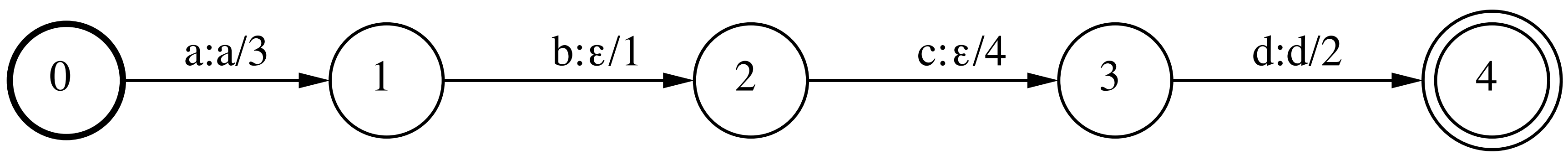
Composition



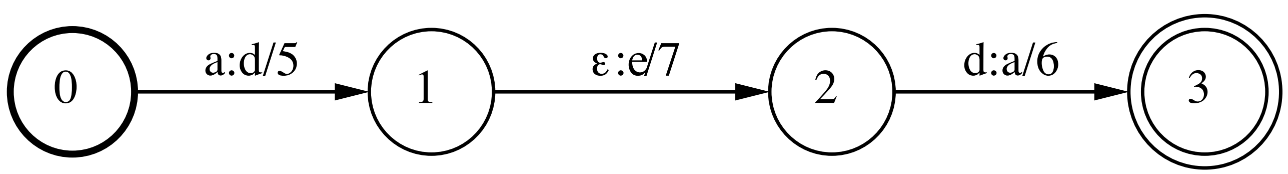
Multiplicative ~ p(y|x) p(z|y)

Composition

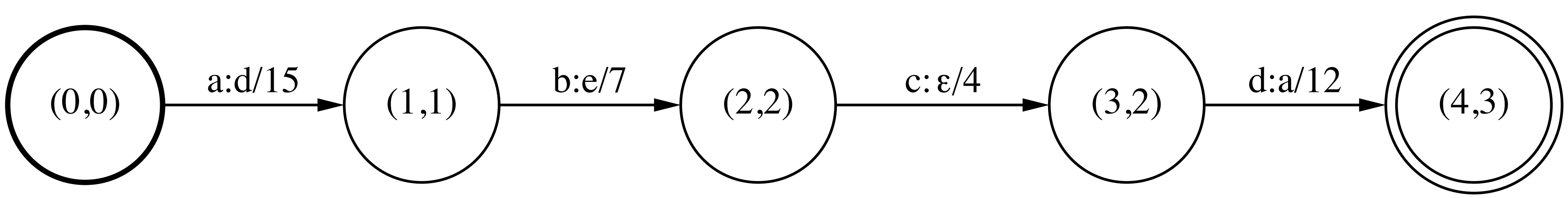
A



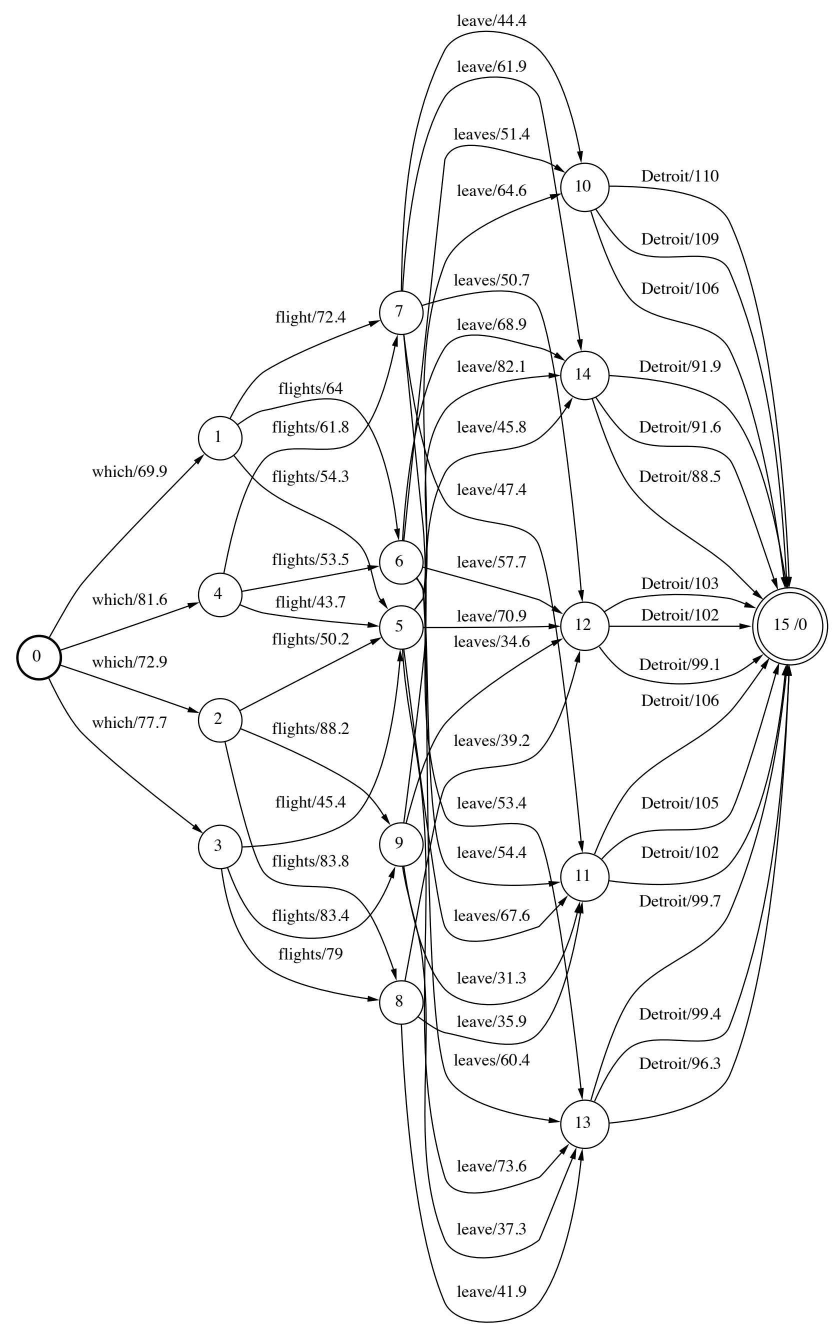
B



A o B

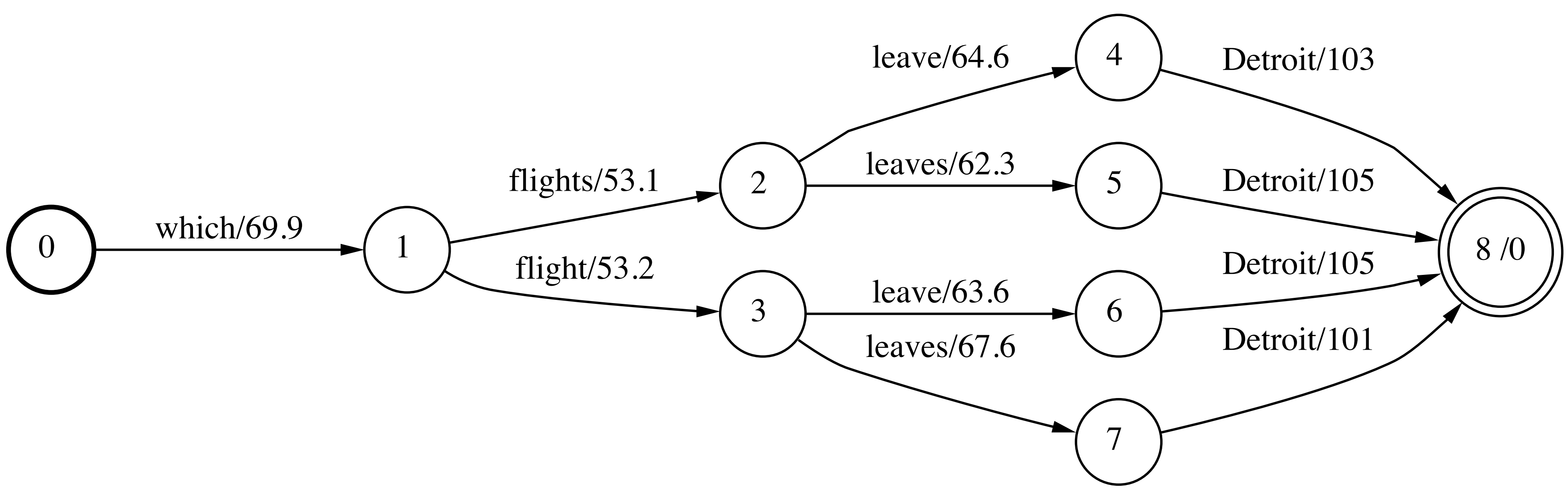


Determinization



Language model: 16 states, 53 transitions

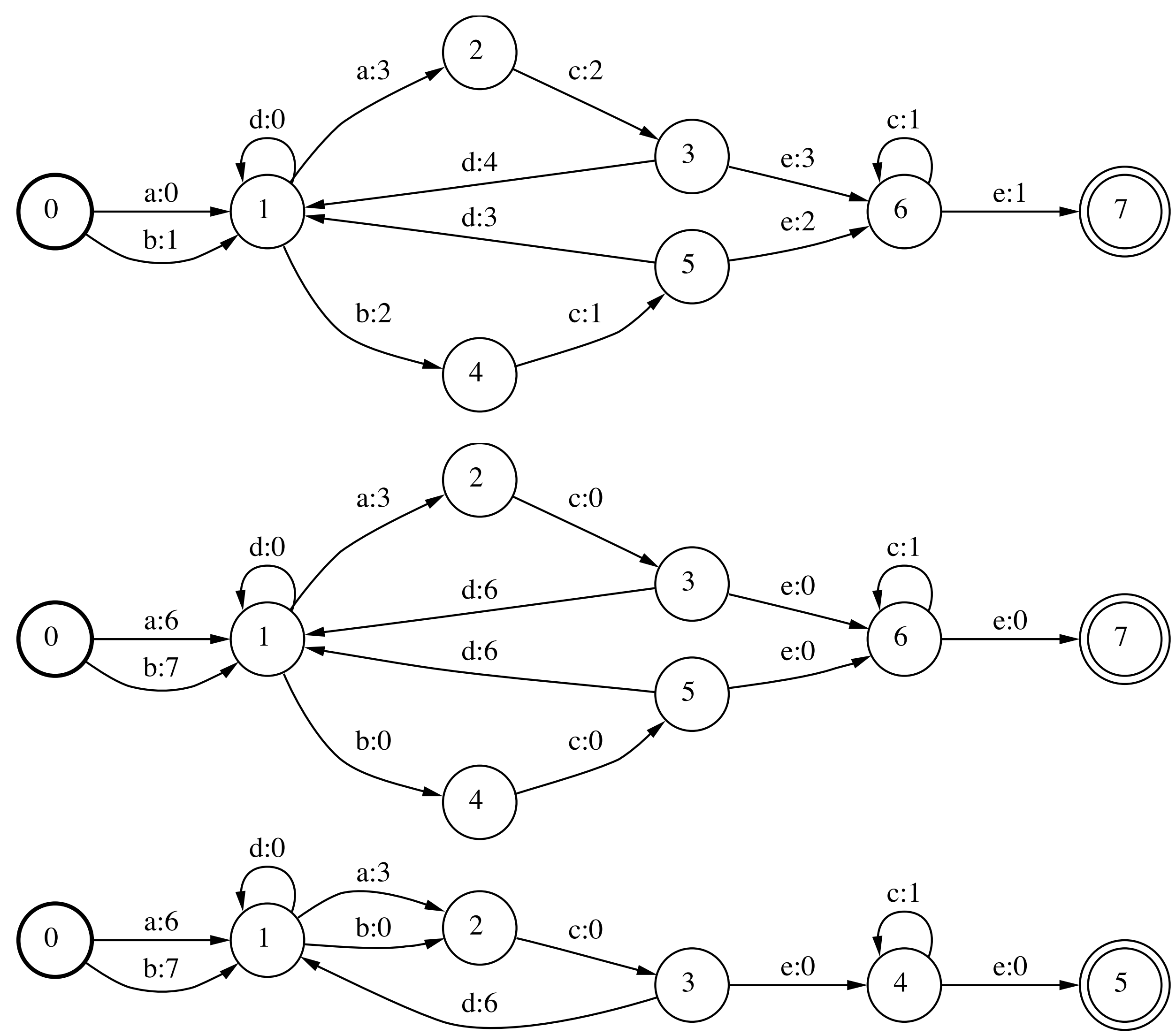
Determinization



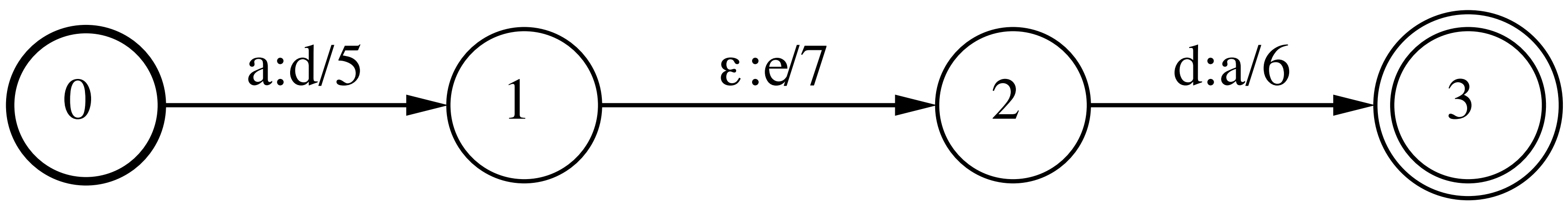
Same language model: 9 states, 11 transitions

Minimization

by weight pushing



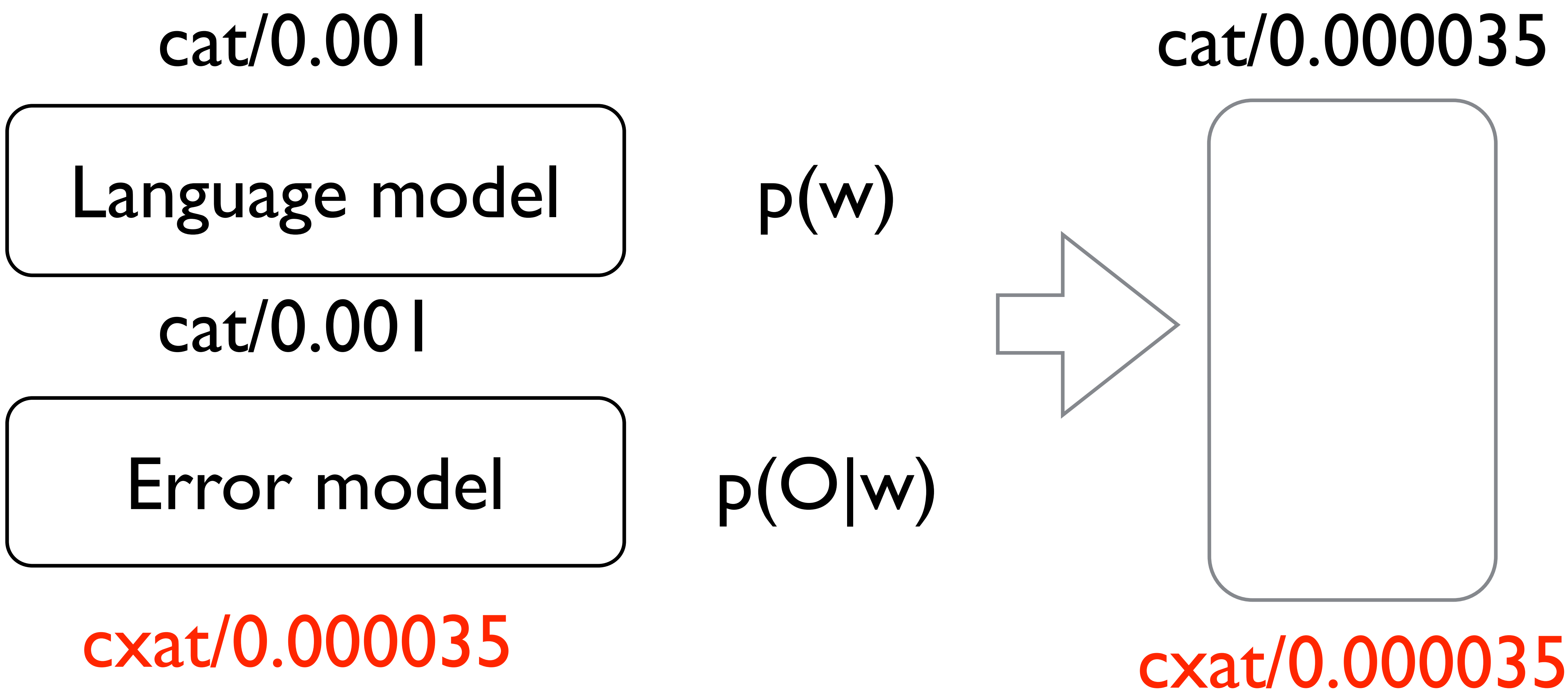
Projection



Trivial: just delete at in/out labels

Example application

probabilistic spell checking



Example application

constructing p(w) and p(O|w)

p(w) can be a n-gram language model
converted to a transducer, easily estimated from data

p(O|w) is much more difficult

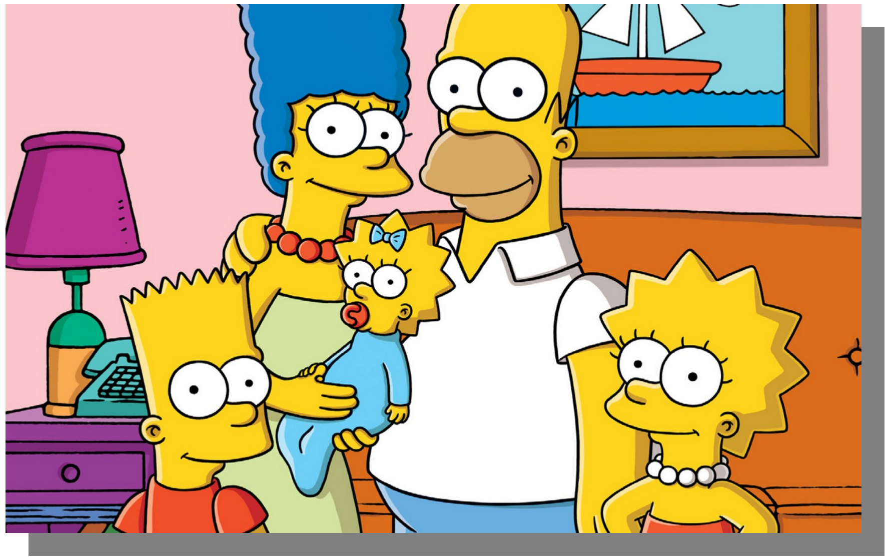
What's the probability of confusing "a" with "z"

Is this word-dependent? Context-dependent?

Example application

Example unigram language model (in Kleene* weighted FST language)

```
$LM = ( the<3.3123733563043>|
        you<3.40834334278697>
        i<3.47764362842074>|
        a<3.62151061674717>|
        to<3.74035111367985>|
        and<4.12455498051775>
        of<4.2521768299548>|
        ...
```



Unigram model from The Simpsons word frequency list (http://pastebin.com/anKcMdvk)

Example application

```
$rep = . ; $ins = "":.; $del = .:""; $chg = .:.-.;
$EM = ( $rep<0.0> | $ins<1.0> | $del<1.0> |
$chg<1.0> )*;
```

Simple error model (insertion/deletion/replacements have a weight of one)

```
$corr = $^shortestPath( (cxat) _o_$EM _o_ $LM );
```

“argmax”

composition

Example application

```
$rep = . ; $ins = "":.; $del = .:""; $chg = .:.-.;
$EM = ( $rep<0.0> | $ins<1.0> | $del<1.0> |
$chg<1.0> )*;
```

Simple error model (insertion/deletion/replacements have a weight of one)

```
$corr = $^shortestPath( (cxat) _o_$EM _o_ $LM );
```

= cat

“argmax” composition

Example application

```
$rep = . ; $ins = "":.; $del = .:""; $chg = .:.-.;
$EM = ( $rep<0.0> | $ins<1.0> | $del<1.0> |
$chg<1.0> )*;
```

Simple error model (insertion/deletion/replacements have a weight of one)

```
$corr = $^shortestPath( (cxat) _o_$EM _o_ $LM );
```

= cat

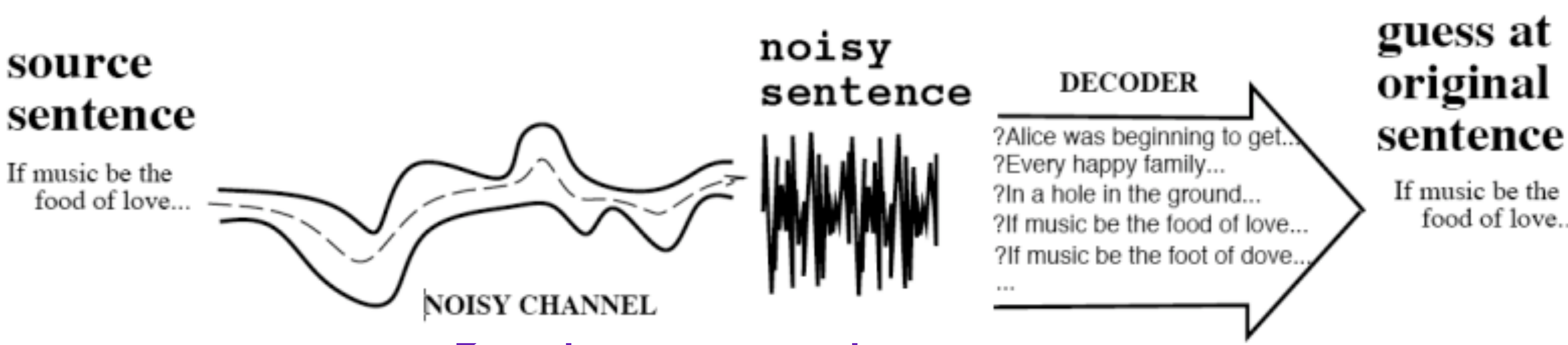
"argmax"

composition

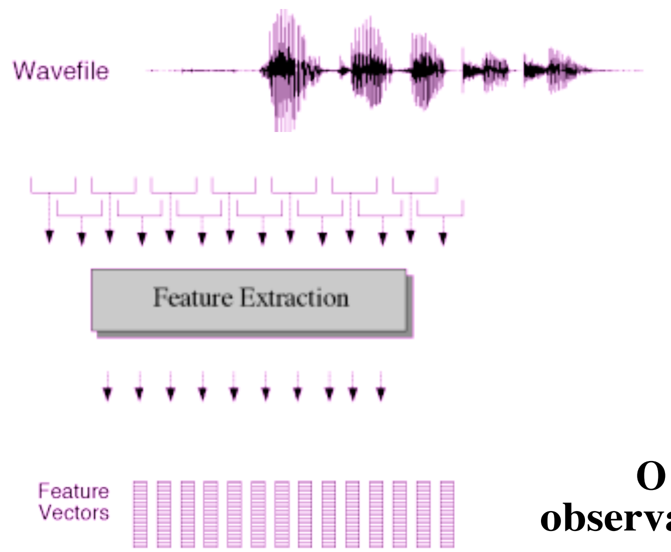
What about 'home'? Does that get corrected and how?

Speech recognition

Noisy channel model for ASR



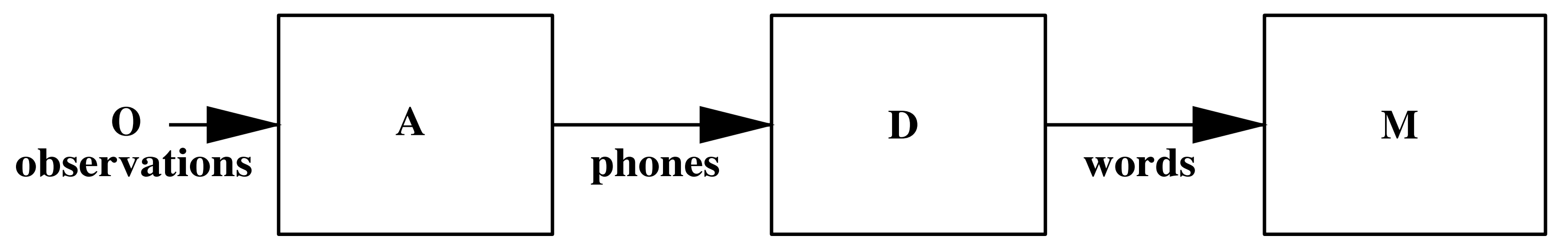
ASR birds-eye view



(MFCCs)

speech recognition

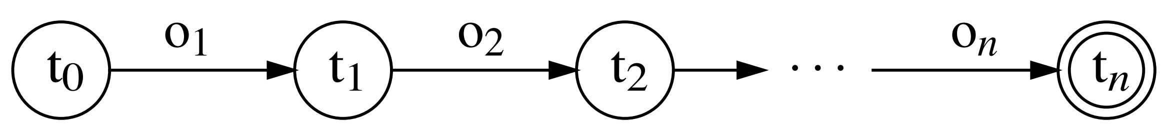
[ɪf]/0.0001 if/0.000034 if/0.0000045



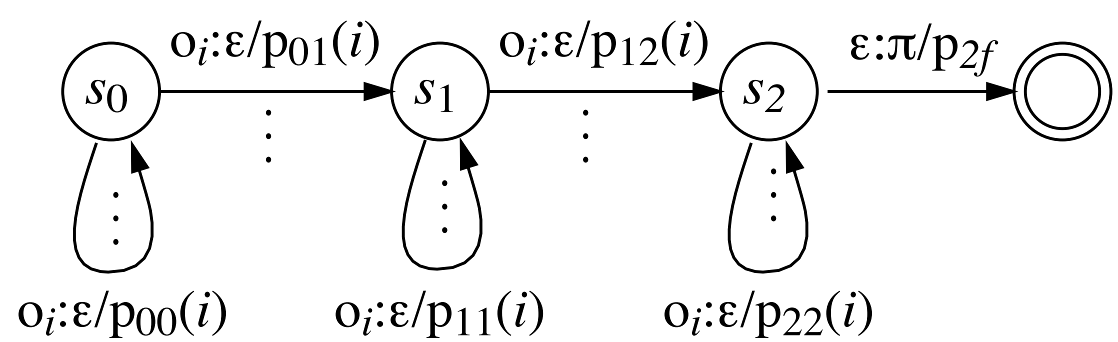
- Recognition from observations $\mathbf{o}$ by composition:
  - *Observations:* $O(\mathbf{s}, \mathbf{s}) = \begin{cases} 1 & \text{if } \mathbf{s} = \mathbf{o} \\ 0 & \text{otherwise} \end{cases}$
  - *Acoustic-phone transducer:* $A(\mathbf{a}, \mathbf{p}) = P(\mathbf{a}|\mathbf{p})$
  - *Pronunciation dictionary:* $D(\mathbf{p}, \mathbf{w}) = P(\mathbf{p}|\mathbf{w})$
  - *Language model:* $M(\mathbf{w}, \mathbf{w}) = P(\mathbf{w})$
- *Recognition:* $\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} (O \circ A \circ D \circ M)(\mathbf{o}, \mathbf{w})$

Slightly more detail

- Quantized observations:



- Phone model A_π : observations $\rightarrow$ phones



Acoustic transducer: $A = \left(\sum_\pi A_\pi\right)^*$

- Word pronunciations D_{data} : phones $\rightarrow$ words