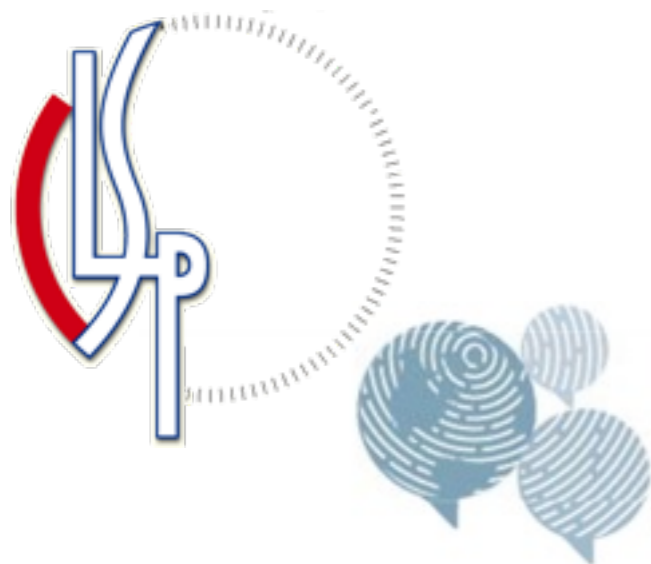


# **A Non-Parametric Model for the Discovery of Inflectional Paradigms from Plain Text using Graphical Models over Strings**

**Markus Dreyer**

Center for Language and Speech Processing (CLSP)  
Human Language Technology Center of Excellence (HLT/COE)  
Johns Hopkins University (JHU)



human language technology  
center of excellence

Dissertation Defense, November 5, 2010

# Motivation

- Rich morphology

## English text

■ ■ ■ break ■  
■■■■ ■ ■ ■ ■ ■  
■ ■ break ■ ■ ■  
jump ■ ■ ■ ■ ■  
■ ■ break ■ ■ ■  
■ ■■■ ■ ■ ■ ■ ■  
jump ■ ■ ■ ■ ■  
■ ■ ■■■ break  
■■■■ ■ break ■ ■  
■ break ■ ■ ■ ■ ■

## German text

■ ■ ■ brichst ■  
■■■■ ■ ■ ■ ■ ■  
■ ■ brecht ■ ■ ■ ■ ■  
springst ■ ■ ■ ■ ■  
■ ■ brechen ■ ■ ■ ■ ■  
■ ■■■ ■ ■ ■ ■ ■  
springe ■ ■ ■ ■ ■  
■ ■ ■■■ breche  
■■■■ ■ brichst ■ ■ ■  
■ breche ■ ■ ■ ■ ■

# Motivation

- **Analyzing** text:
  - lack of generalization
  - data sparseness
- **Generating** text:
  - need to generate correct forms
  - produce correctly inflected text

## German text

■ ■ ■ brichst ■  
■ ■ ■ ■ ■ ■ ■ ■ ■ ■  
■ brecht ■ ■ ■ ■  
springst ■ ■ ■ ■  
■ ■ brechen ■ ■ ■ ■  
■ ■ ■ ■ ■ ■ ■ ■ ■ ■  
springe ■ ■ ■ ■ ■ ■ ■ ■  
■ ■ ■ ■ breche  
■ ■ ■ ■ ■ ■ ■ ■ ■ ■  
■ breche ■ ■ ■ ■ ■ ■ ■ ■

# Motivation

- In NLP, we often **need** to analyze or generate text in such languages.
- So there is a need for a general **morphology model** that knows **how to inflect words**.
- Since annotations are always expensive, it would be best to **learn from unannotated text**.

# Motivation

So how do you inflect a verb?

You look it up in such a table, for example:

Inflectional Paradigm

infinitive	treffen			
1st	treffe	treffen	traf	trafen
2nd	triffst	trefft	trafst	traft
3rd	trifft	treffen	traf	trafen
	singular	plural	singular	plural
	present		past	

But creating such supervised data is **expensive.**

# Motivation

- This talk is about a comprehensive **model for inflectional morphology**.
- **Main goal:**
  - Given some **unannotated text**, can we **learn how to inflect** the verbs of a language (incl. irregularities and exceptions)?
  - Discover the **inflectional paradigms** (tables) of a language, using minimal supervision

# Motivation

I. Identify the different lexemes in text

German text

brichst  
 brecht  
 springst  
 brechen  
 springe  
 breche  
 brichst  
 breche

**Tokens**

Paradigm

infinitive				
1st				
2nd				
3rd				
	singular	plural	singular	plural
	present		past	

**Types**

# Motivation

I. Identify the different lexemes in text

German text

brichst  
 brecht  
 springst  
 brechen  
 springe  
 breche  
 brichst  
 breche

**Tokens**

Paradigm

infinitive				
1st				
2nd				
3rd				
	singular	plural	singular	plural
	present		past	

**Types**



# Motivation

I. Identify the different lexemes in text

German text

brichst  
brect  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

Paradigm

infinitive				
1st				
2nd				
3rd				
	singular	plural	singular	plural
	present		past	

**Types**

# Motivation

2. Place each form of a lexeme into its paradigm

German text

brichst  
brecht  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

Paradigm

infinite	brechen			
1st	breche			
2nd	brichst	brecht		
3rd				
	singular	plural	singular	plural
	present		past	

**Types**

# Motivation

2. Sort each lexeme into a paradigm

Paradigm

German text

brichst

brecht

springst

brechen

springe

breche

brichst

breche

**Tokens**

	infinitive	brechen			
1st	breche	brichen? brechen?	brichte? brach?	brichten? brachen?	
2nd	brichst	brecht	brichtest? brachst?	brichtet? bracht?	
3rd	bricht? brecht?	brichen? brechen?	brichte? brach?	brichten? brachen?	
	singular	plural	singular	plural	
	present		past		

**Types**

# Motivation

2. Sort each lexeme into a paradigm

Paradigm

German text

brichst  
brecht  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

infinitive				
1st				
2nd				
3rd				
	singular	plural	singular	plural
	present		past	

**Types**

# Motivation

2. Sort each lexeme into a paradigm

Paradigm

German text

brichst  
brecht  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

infinitive					
1st	springe				
2nd	springst				
3rd					
	singular	plural	singular	plural	
	present		past		

**Types**

# Motivation

2. Sort each lexeme into a paradigm

Paradigm

German text

■	■	■	brichst	■
■	■	■	■	■
■	brecht	■	■	■
springst	■	■	■	■
■	■	brechen	■	■
■	■	■	■	■
springe	■	■	■	■
■	■	■	breche	■
■	■	■	brichst	■
■	breche	■	■	■

**Tokens**

		springen? sprengen?		brichen? brichte? brichten?	
infinitive					
1st	springe	springen? sprengen?	springte? sprang?	springte? sprang?	
2nd	springst	springt? sprengt?	springtest? sprangst?	springtet? sprangt?	
3rd	springt? sprengt?	springen? sprengen?	springte? sprang?	springten? sprangen?	
	singular	plural	singular	plural	
	present		past		

**Types**

# Motivation

German text

brichst  
brecht  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

Paradigm

infinite	springen? sprengen?	brichen?	brichte?	brichten?
1st	springe	springen? sprengen?	springte? sprang?	springte? sprang?
2nd	springst	springt? sprengt?	springtest? sprangst?	springtet? sprangt?
3rd	springt? sprengt?	springen? sprengen?	springte? sprang?	springten? sprangen?
	singular	plural	singular	plural
	present		past	

**Types**

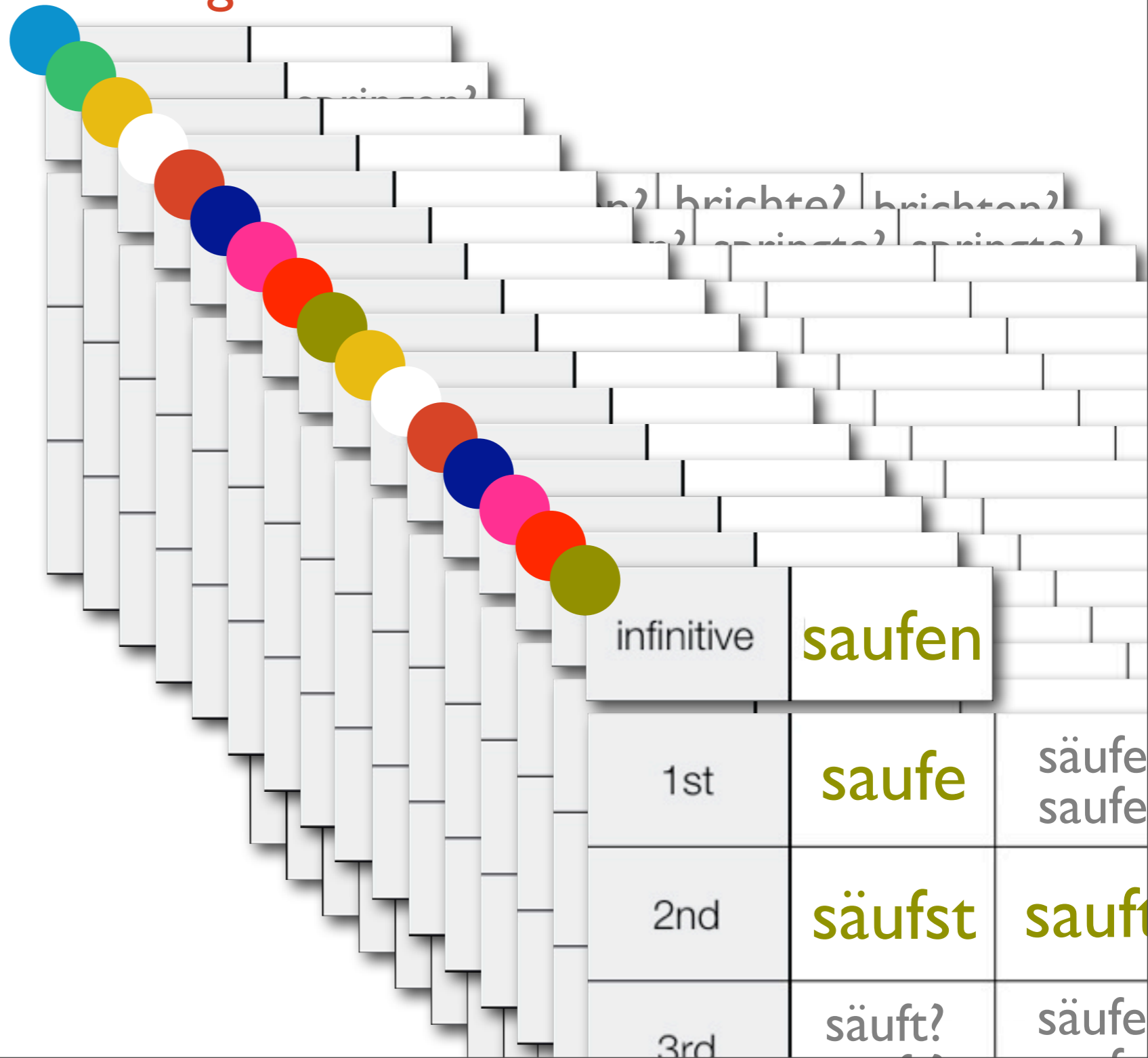
# Motivation

German text

brichst  
brecht  
springst  
brechen  
springe  
breche  
brichst  
breche

**Tokens**

Paradigm



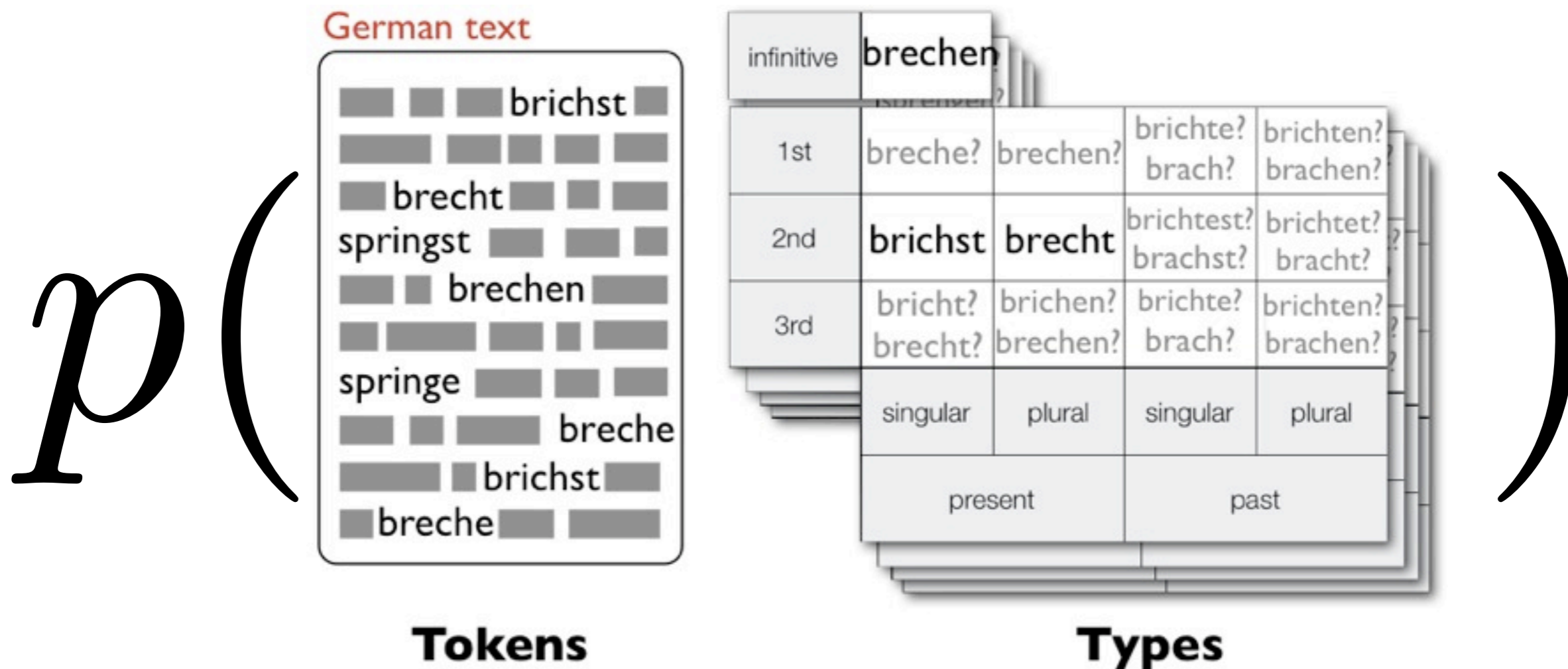


# Motivation

- Similar to **information extraction** tasks:
  - Find information in **text**,
  - Put it in **database**,
  - Make **deductions**,
  - Find **more** information in text,
  - *iterate ...*

# Motivation

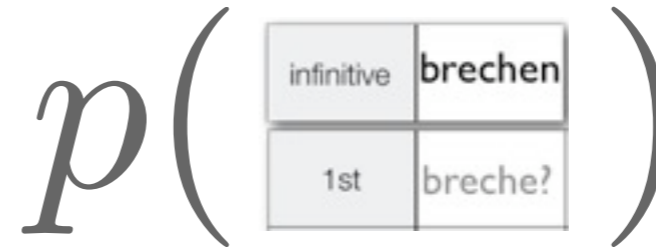
In order to perform this morphological knowledge discovery, we define a **probability distribution** over a **text corpus** and its (hidden) inflectional **paradigms**:



# Overview

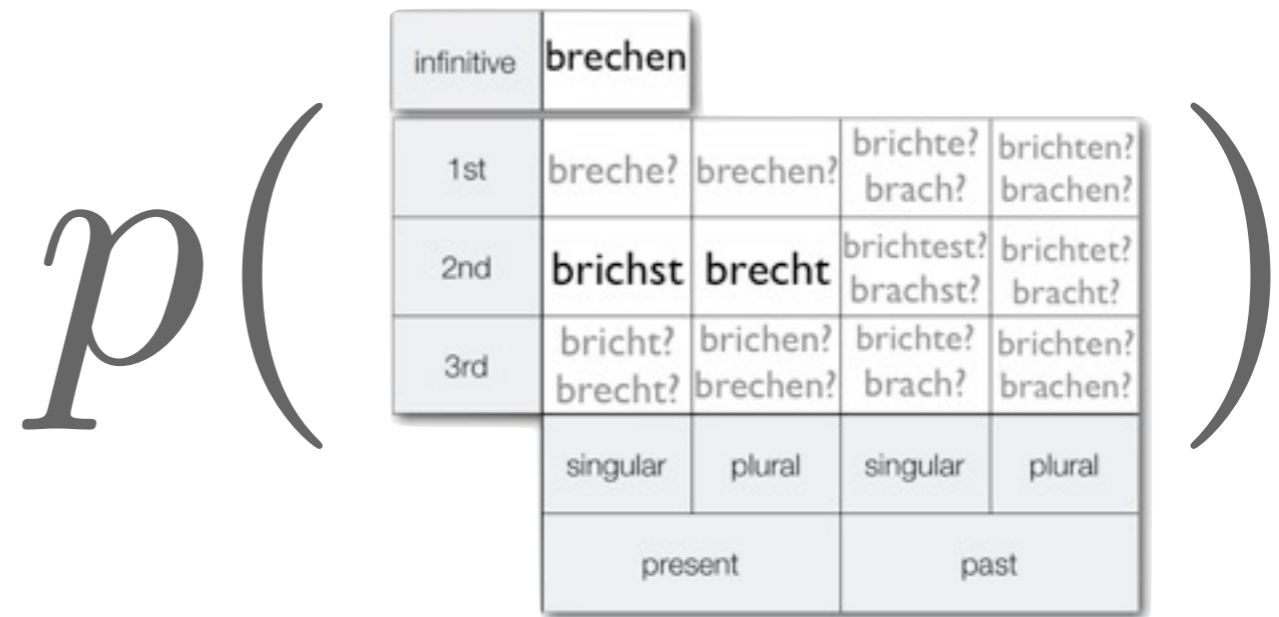
1

String pairs



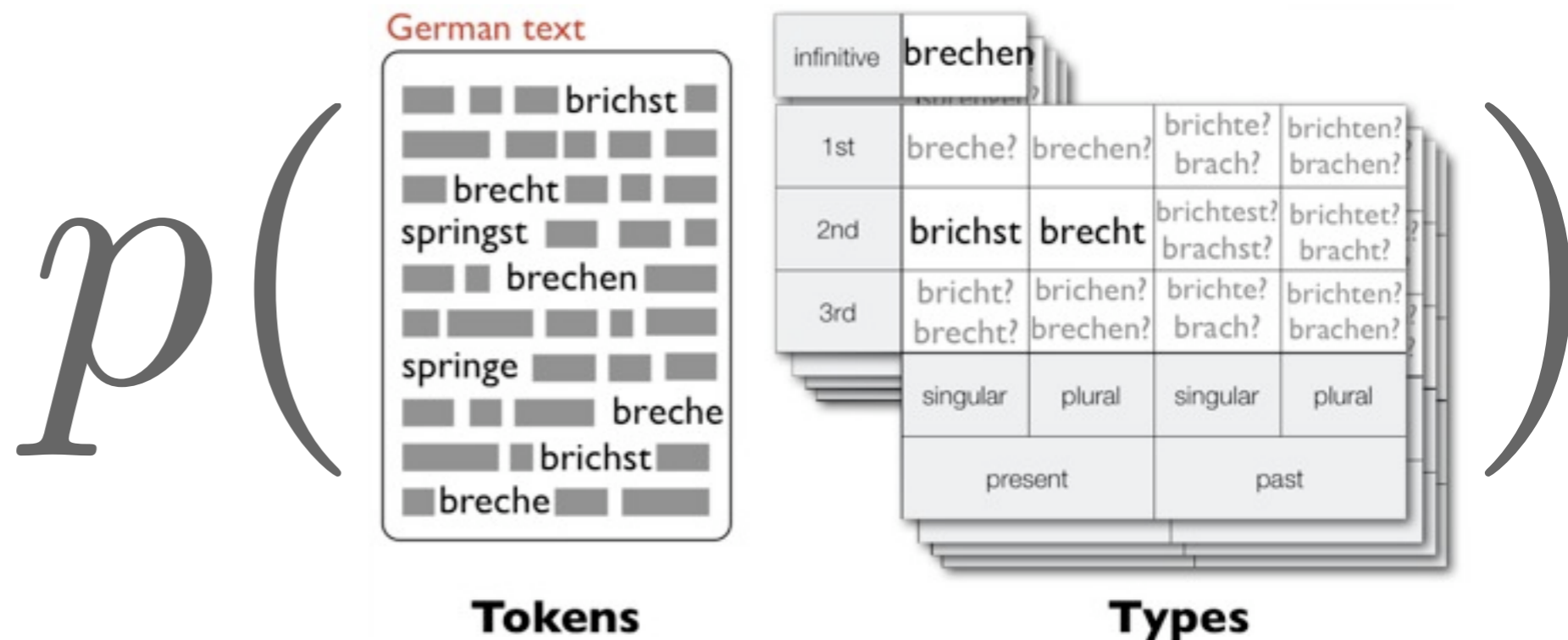
2

Multiple strings  
(paradigms)



3

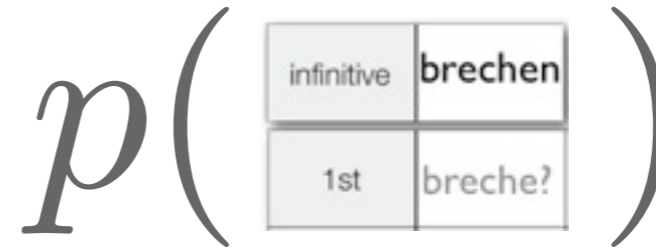
Text and  
paradigms



# Overview

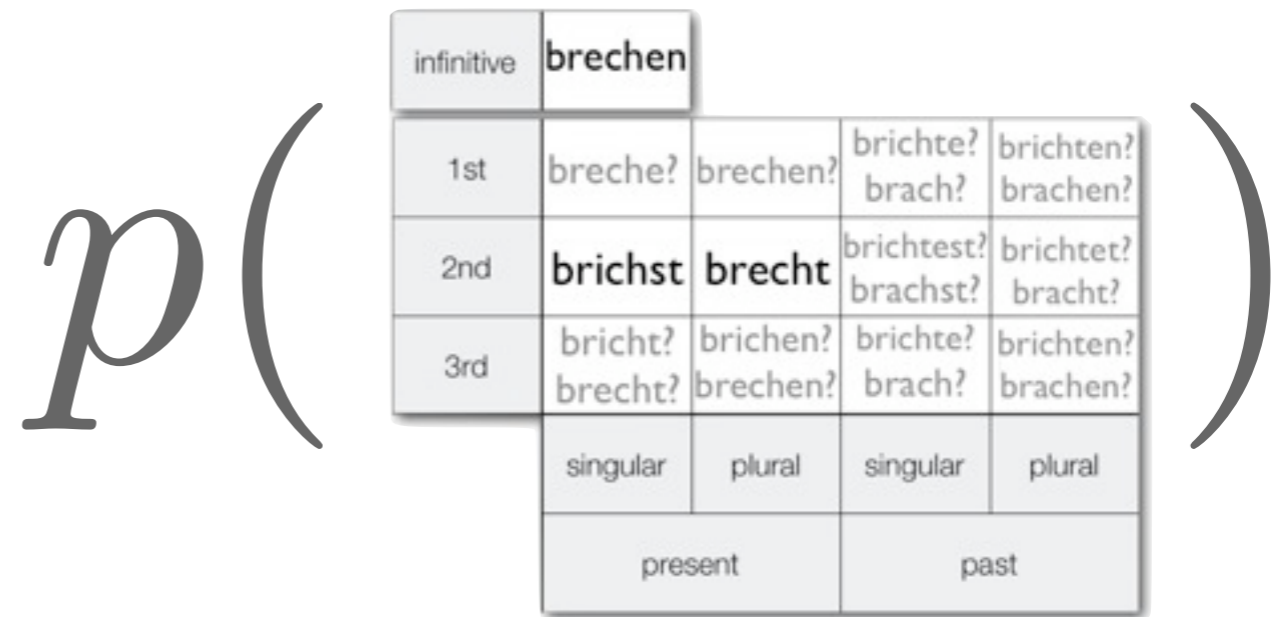
## 1 String pairs

Dreyer, Smith & Eisner, 2008



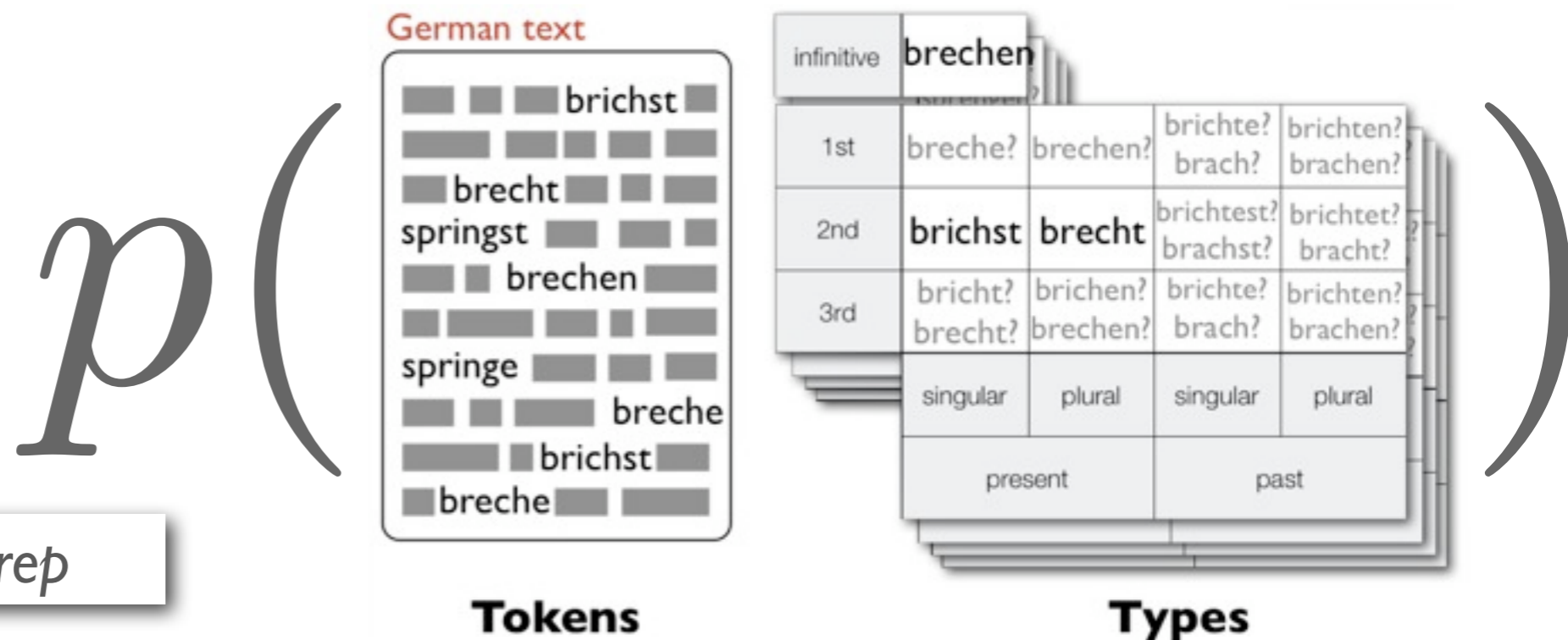
## 2 Multiple strings (paradigms)

Dreyer & Eisner, 2009



## 3 Text and paradigms

Dreyer & Eisner, *in prep*





1

# String Pairs

String pair problems are common in NLP:

Morphology:

infinitive	brechen
1st	breche?

# String Pairs

String pair problems are common in NLP:

Morphology:

infinitive	brechen
1st	breche?

## Transliteration

john hardt - yue han ha te

patrick johnson - pa te li ke yue han xun

frederick william mulley - fu lei de li ke wei lian ma li

## Pronunciation

Sternanisöl - /'stɛrnʔani:sʔø:l/

loophole - /'lu:p,həʊl/

## Spelling correction

Honululu - Honolulu

braek- break

# String Pairs

- We want to build a **probability model** over string pairs.
- Such a model can produce k-best output, can be **plugged** into bigger models later, etc.
- We would like to make use of **flexible features**, be able to look at **linguistic properties** of the strings,
- and **train** the parameters from data.



# String Pairs

$$\Pr(s_1, s_2) = \frac{1}{Z} F(s_1, s_2)$$

$$\Pr(s_2 | s_1) = \frac{1}{Z} F(s_1, s_2)$$

- Function  $F$  evaluates how well the two strings go together.
- It looks at properties (“features”) of the string pair and assigns some score.

1

# String Pairs

$s_1 =$  breaking  
 $s_2 =$  broke

1

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

1

# String Pairs

#breaking#  
#broke#

1

# String Pairs

#breaking#  
#br€oke€€#

# String Pairs

#breaking#  
#br€oke€€#

#breaking#  
#bro€ke€€#

#brea€king#  
#br€€oke€€#

#break€ing#  
#bro€ke€€€#

...

1

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

1

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$



1

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$



1

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$

$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#br}\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right)$$

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$

$$\left( \begin{aligned}
 & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#br}\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 + & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\text{\#} \end{array} \right)
 \end{aligned} \right)$$

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$

$$\left( \begin{aligned}
 & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#br}\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 + & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 + & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#brea}\epsilon\text{king\#} \\ \text{\#br}\epsilon\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right)
 \end{aligned} \right)$$

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$

$$\left( \begin{aligned}
 & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#br}\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#brea}\epsilon\text{king\#} \\ \text{\#br}\epsilon\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#break}\epsilon\text{ing\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\epsilon\text{\#} \end{array} \right)
 \end{aligned} \right)$$

# String Pairs

$s_1 = \text{\#breaking\#}$   
 $s_2 = \text{\#broke\#}$

$F(s_1, s_2) =$

$$\left( \begin{aligned}
 & \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#br}\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#breaking\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#brea}\epsilon\text{king\#} \\ \text{\#br}\epsilon\epsilon\text{oke}\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \exp \sum_i \theta_i f_i \left( \begin{array}{c} \text{\#break}\epsilon\text{ing\#} \\ \text{\#bro}\epsilon\text{ke}\epsilon\epsilon\epsilon\text{\#} \end{array} \right) \\
 & + \dots
 \end{aligned} \right)$$

1

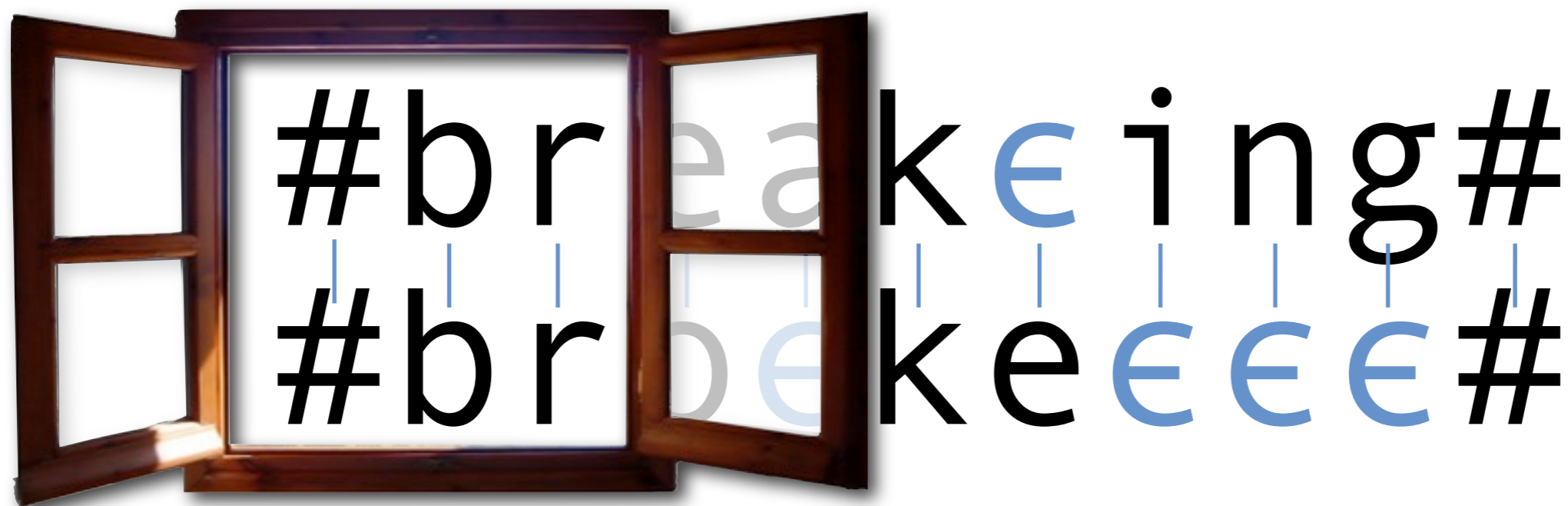
# String Pairs

$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{break} \epsilon \text{ing} \# \\ \# \text{bro} \epsilon \text{ke} \epsilon \epsilon \epsilon \# \end{array} \right)$$

#break€ing#  
#bro€ke€€€#

$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{break} \epsilon \text{ing} \# \\ \# \text{bro} \epsilon \text{ke} \epsilon \epsilon \epsilon \# \end{array} \right)$$

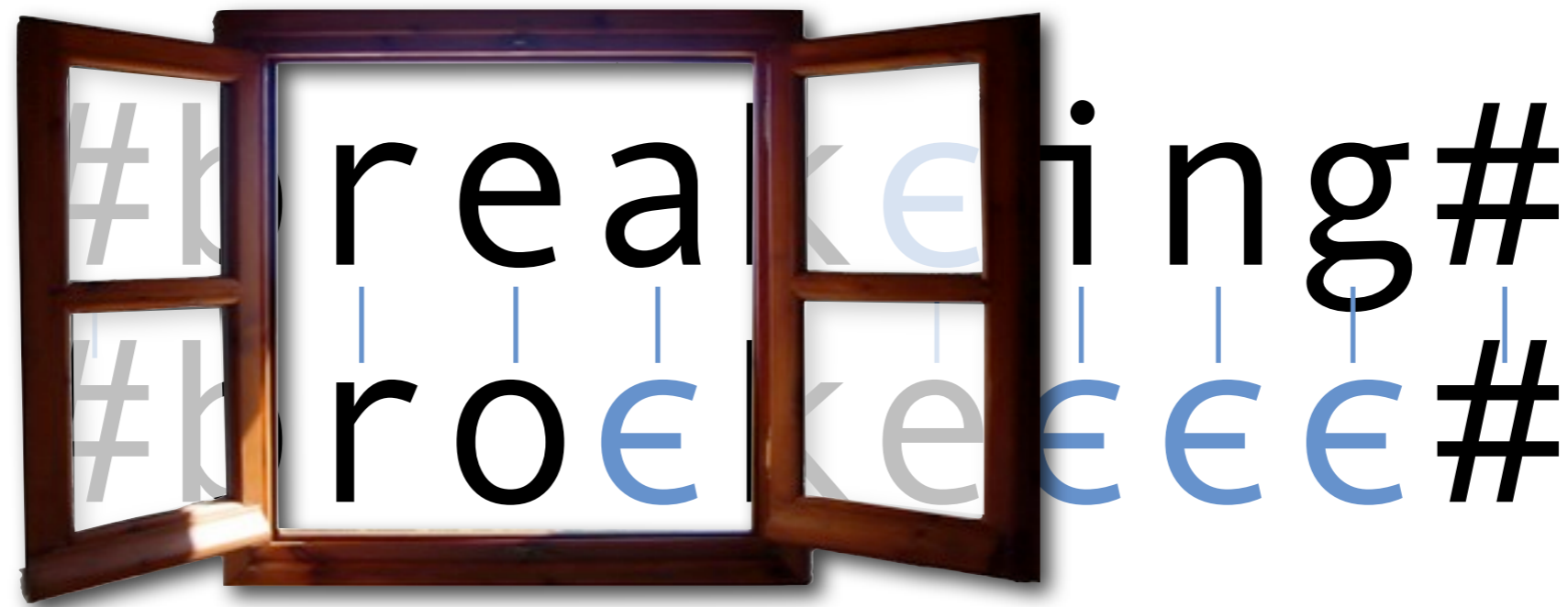




$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{br} \text{e} \text{a} \text{k} \text{€} \text{i} \text{n} \text{g} \# \\ \# \text{br} \text{o} \text{€} \text{k} \text{e} \text{€} \text{€} \text{€} \# \end{array} \right)$$



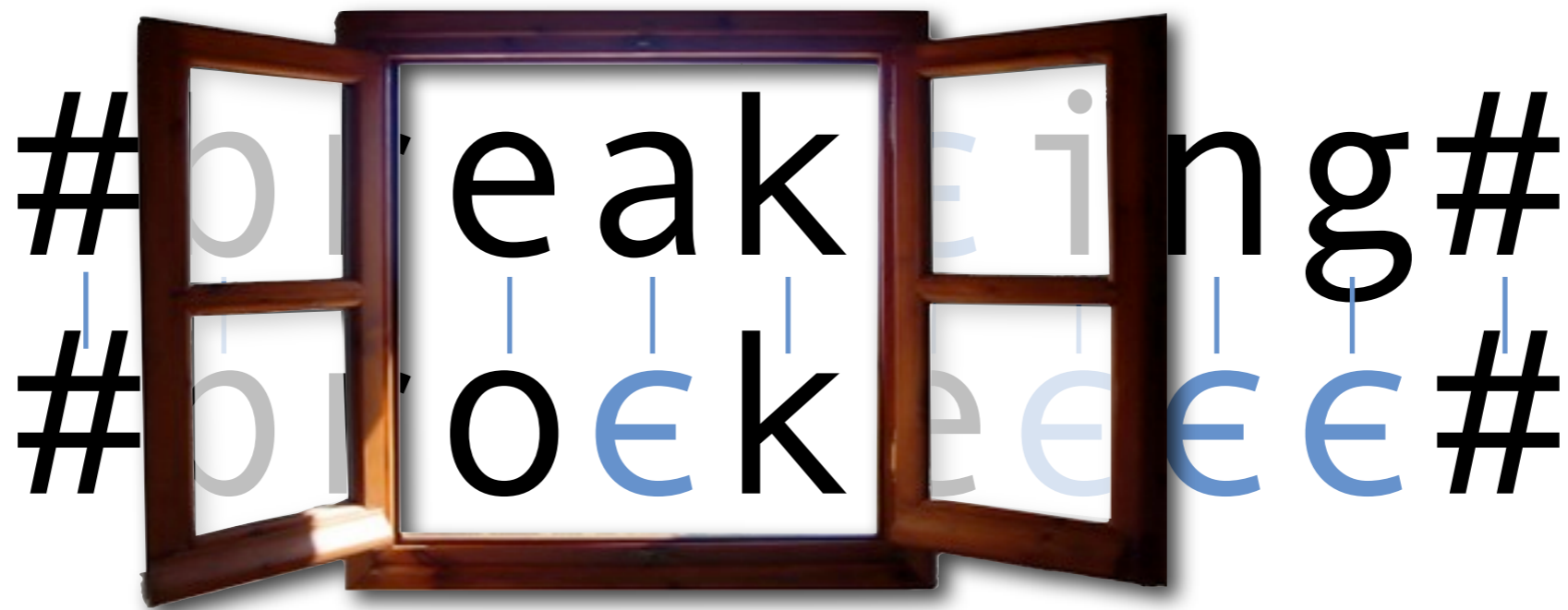
$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{break} \epsilon \text{ing} \# \\ \# \text{brok} \epsilon \epsilon \epsilon \# \end{array} \right)$$



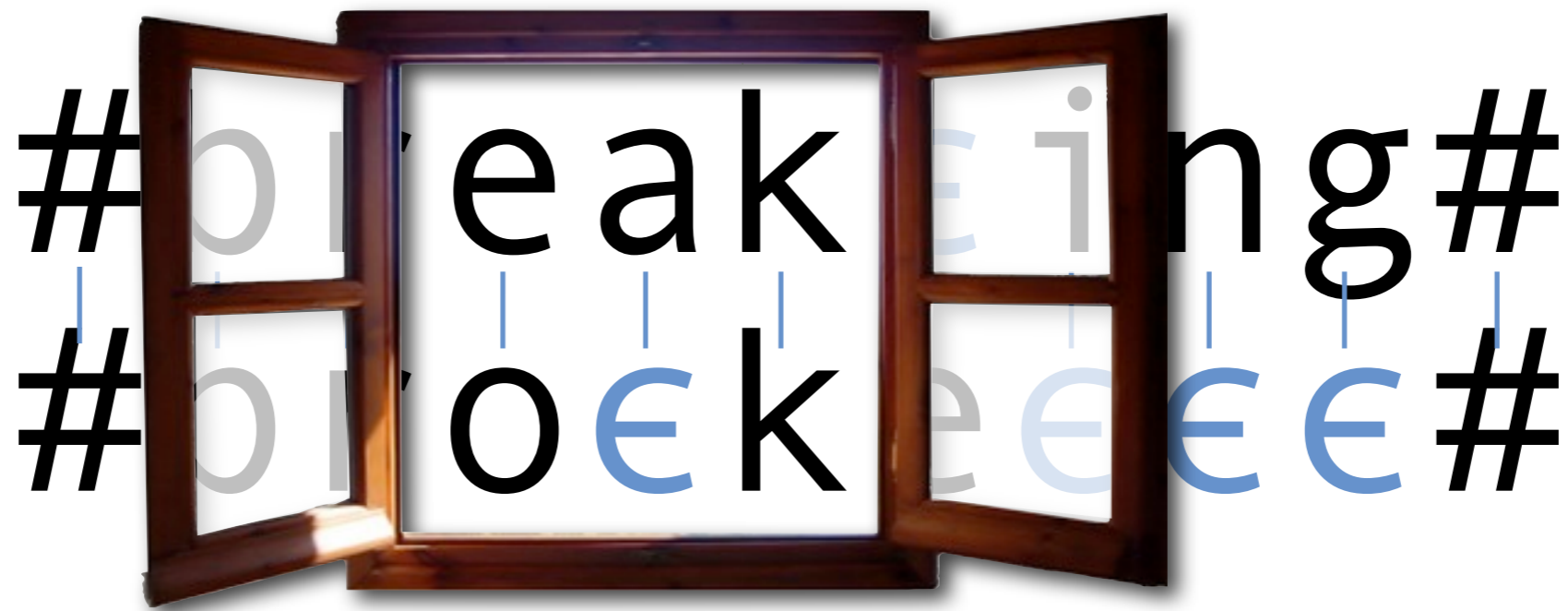
$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{br} \text{e} \text{a} \text{k} \text{e} \text{i} \text{n} \text{g} \# \\ \# \text{br} \text{o} \text{e} \text{k} \text{e} \text{e} \text{e} \text{e} \# \end{array} \right)$$

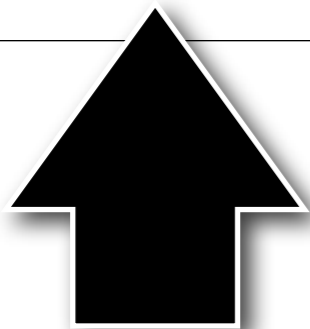
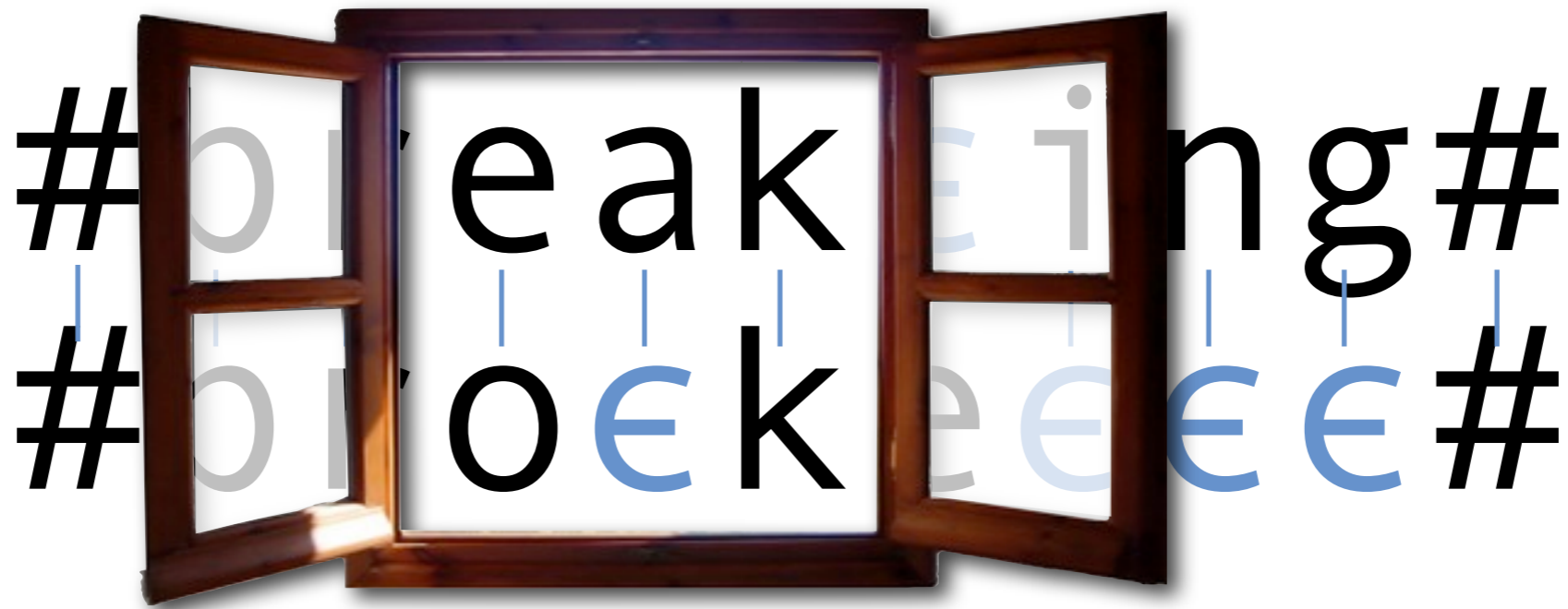


$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{br} \text{e} \text{a} \text{k} \text{e} \text{i} \text{n} \text{g} \# \\ \# \text{br} \text{o} \text{ε} \text{k} \text{e} \text{ε} \text{ε} \text{ε} \# \end{array} \right)$$

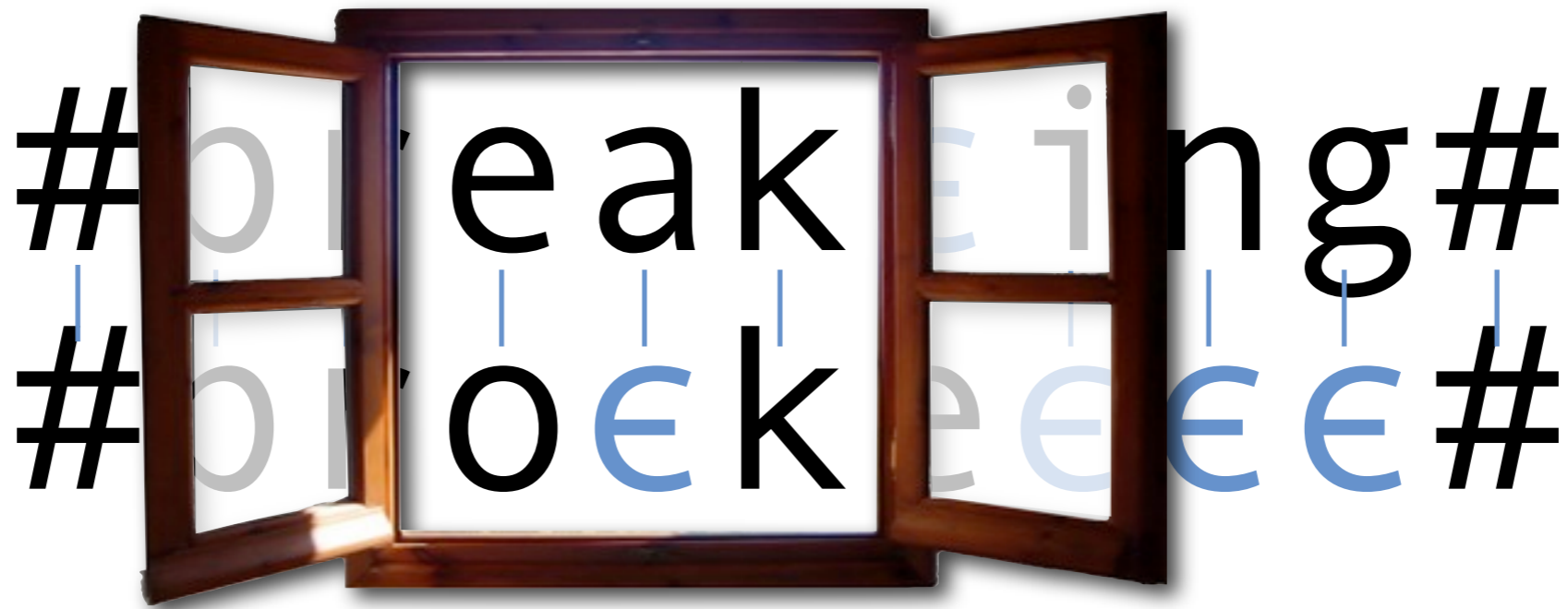


$$\exp \sum_i \theta_i f_i \left( \begin{array}{c} \# \text{break} \epsilon \text{ing} \# \\ \# \text{bro} \epsilon \text{ke} \epsilon \epsilon \epsilon \# \end{array} \right)$$

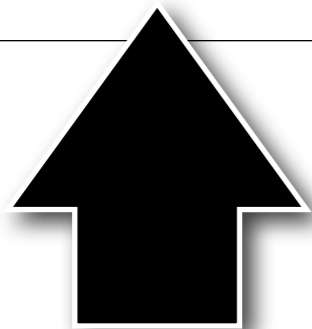




full  
window



e a k  
| | |  
o € k



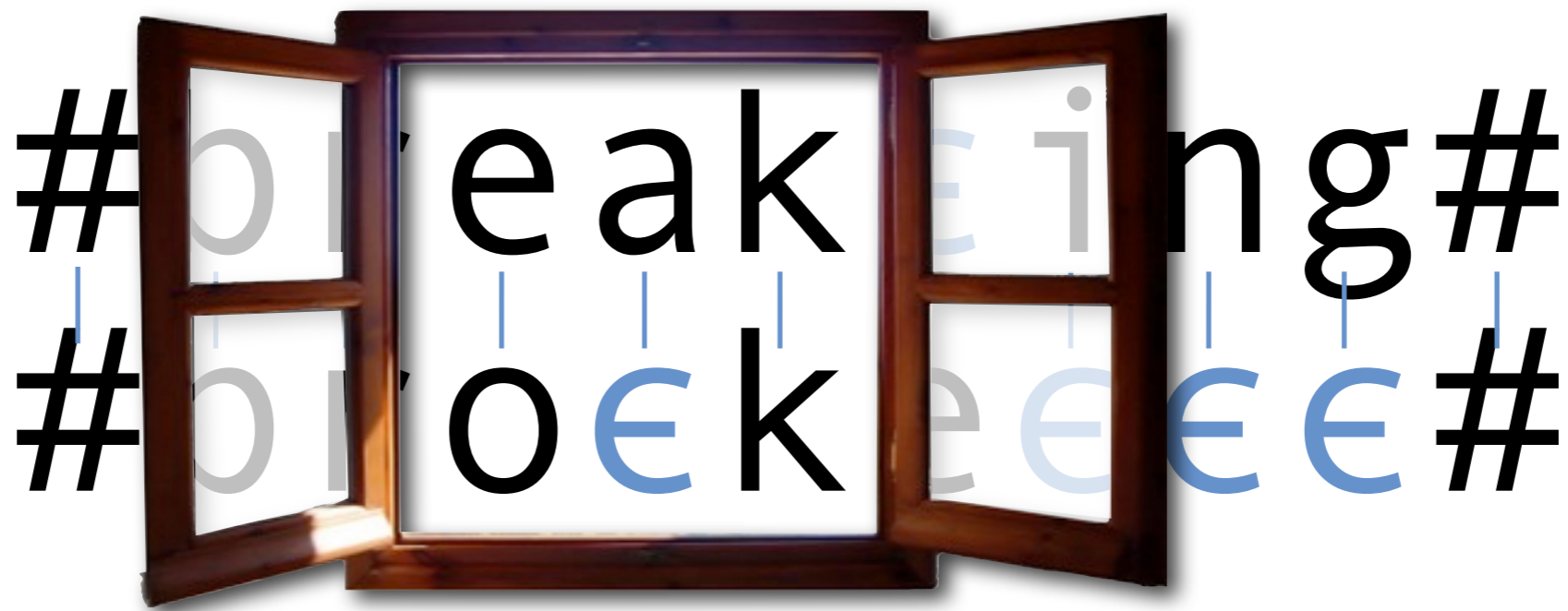
full  
window

V V C  
| | |  
V € C

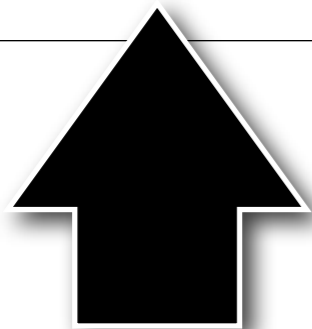


vowels,  
consonants





e a k  
o € k



full  
window

V V C  
V € C

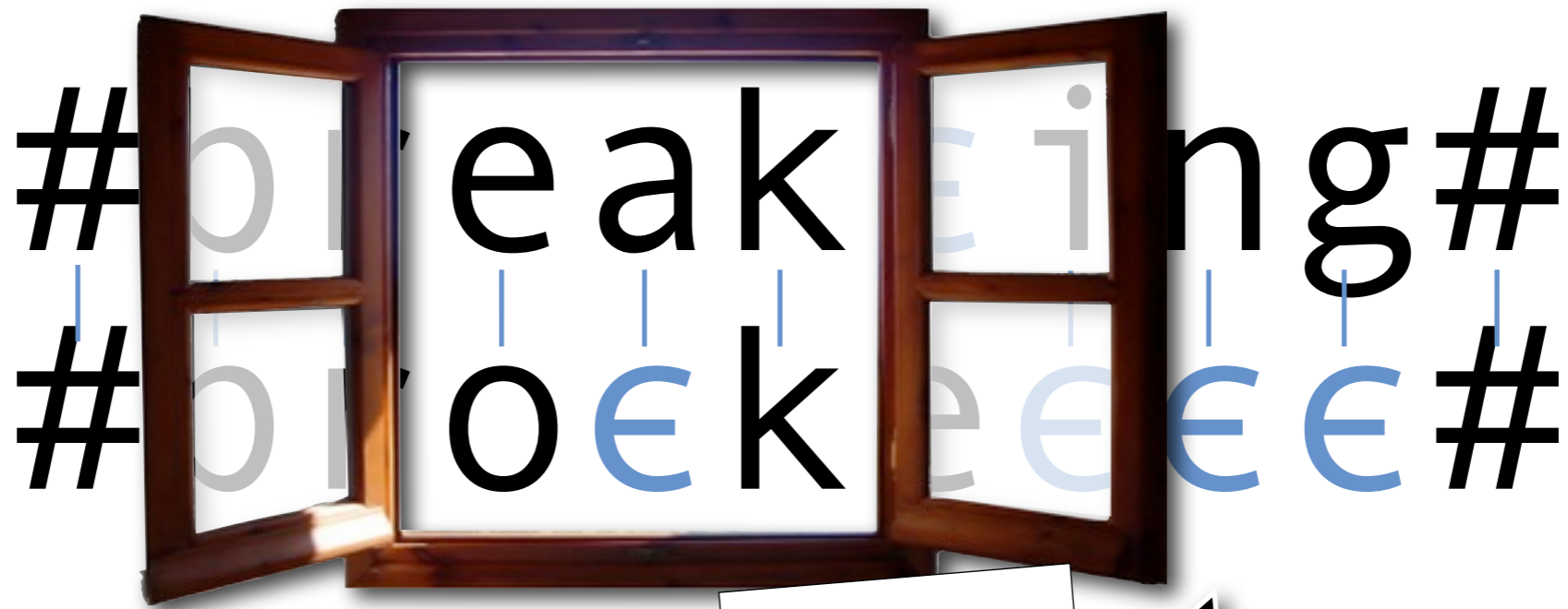


vowels,  
consonants

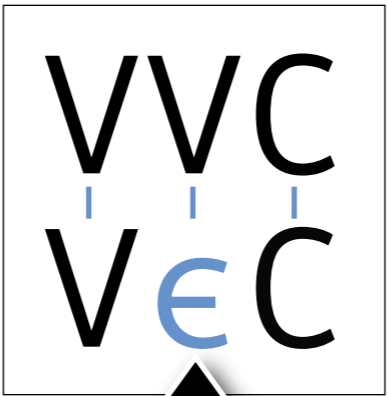
? ? ?  
o € k



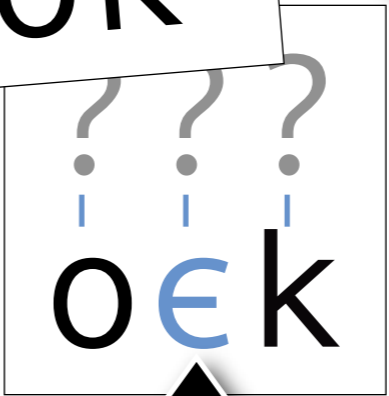
target  
language



full window



vowels, consonants



target language

# o r e a k e i n g #  
# o r o € k e € € #

???  
ok

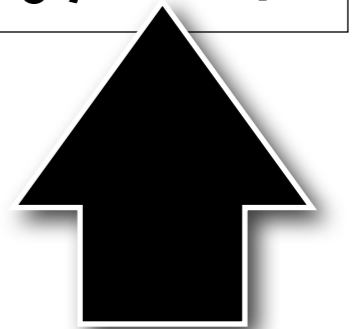
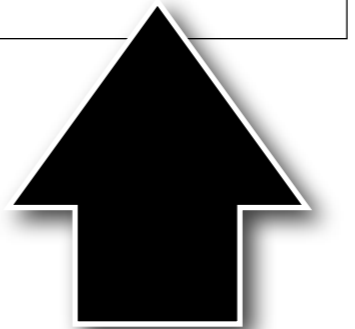
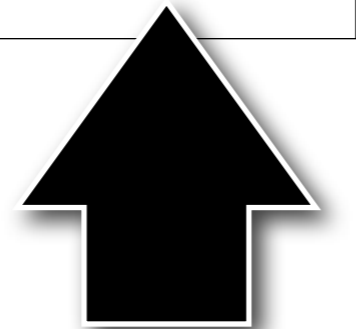
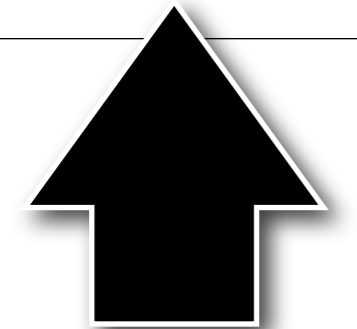
“collapsed”

e a k  
o € k

V V C  
V € C

???  
o € k

subst  
del  
ident



full  
window

vowels,  
consonants

target  
language

subst, del, ins,  
ident

Also add versions of these features that are backed off to bigrams!

king#  
€€#

???  
?k

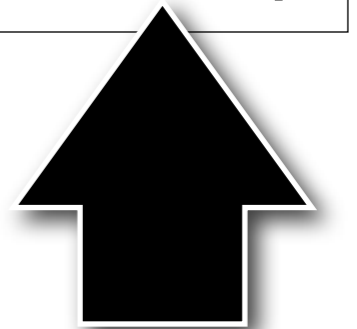
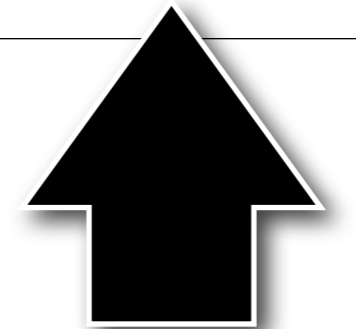
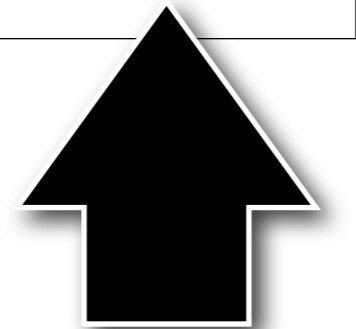
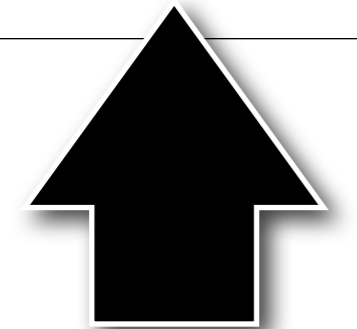
“collapsed”

? ak  
? € k

? VC  
? € C

? ? ?  
? € k

???  
? del  
ident



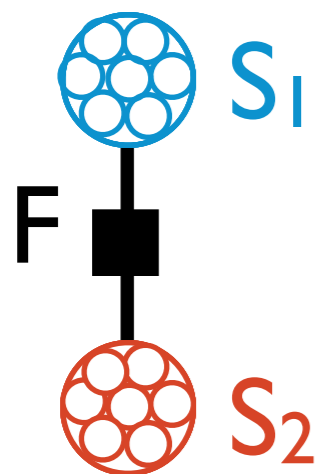
full window

vowels, consonants

target language

subst, del, ins, ident

# String Pairs



- To compute such feature-based scores for two string variables  $S_1$  and  $S_2$ , we construct a **weighted finite-state transducer F**
- It can assign a **score** to any string pair  $S_1, S_2$

$$\Pr(S_1, S_2) = \frac{1}{Z} F(S_1, S_2)$$

## **Background:** Finite-state machines

What is a **finite-state acceptor (FSA)** ?

An automaton with a finite number of states and arcs.  
Can be used to assign a score to any **string**.

What is a **finite-state transducer (FST)** ?

Same as FSA, but used to assign score to any **string pair**  
(e.g. evaluating how well they go together).

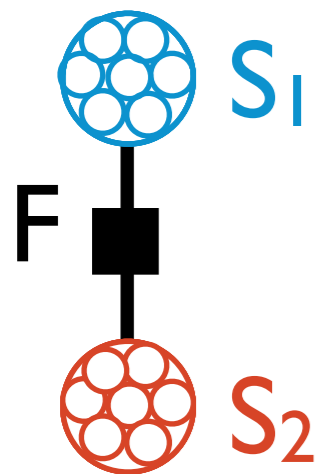
# String Pairs

- Specific kind of **grammar** that describes and scores one or more strings
- **Closure** properties under many useful operations (we will use composition, intersection, projection)
- **Useful** for many tasks in natural language processing

1

# String Pairs

$S_1 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{e} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{e} \text{O} \xrightarrow{n} \text{O}$

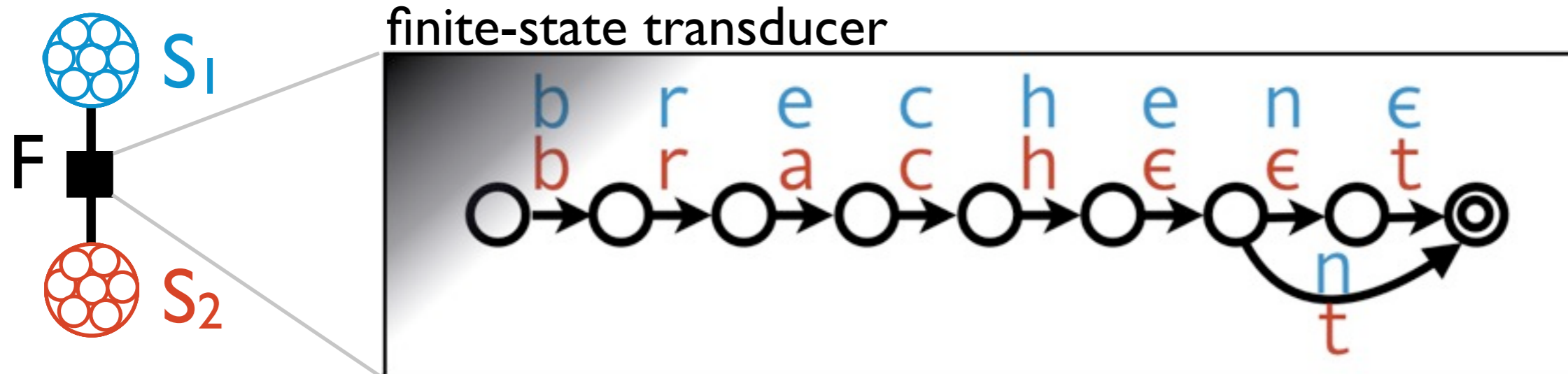
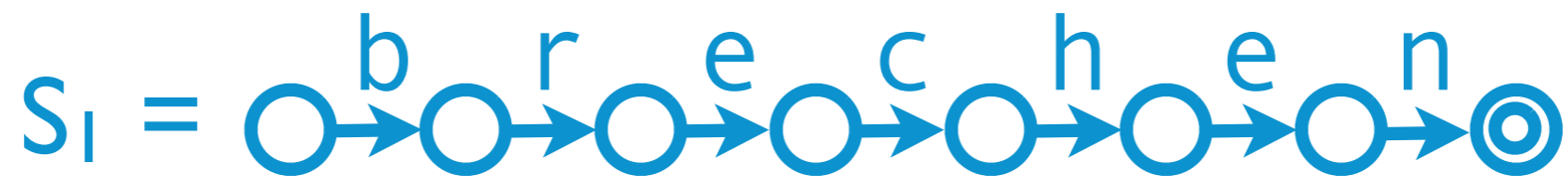


$S_2 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{a} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{t} \text{O}$



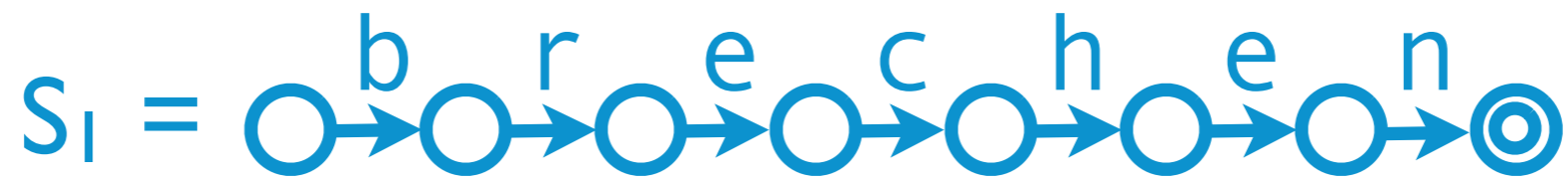
1

# String Pairs

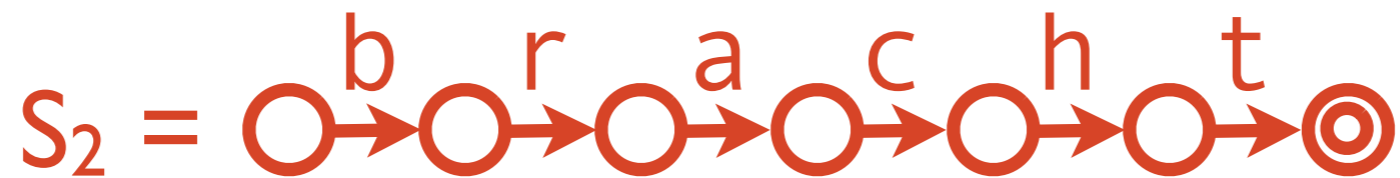
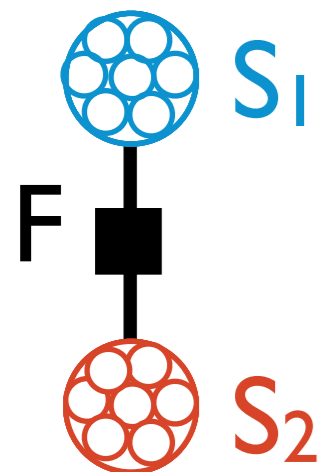
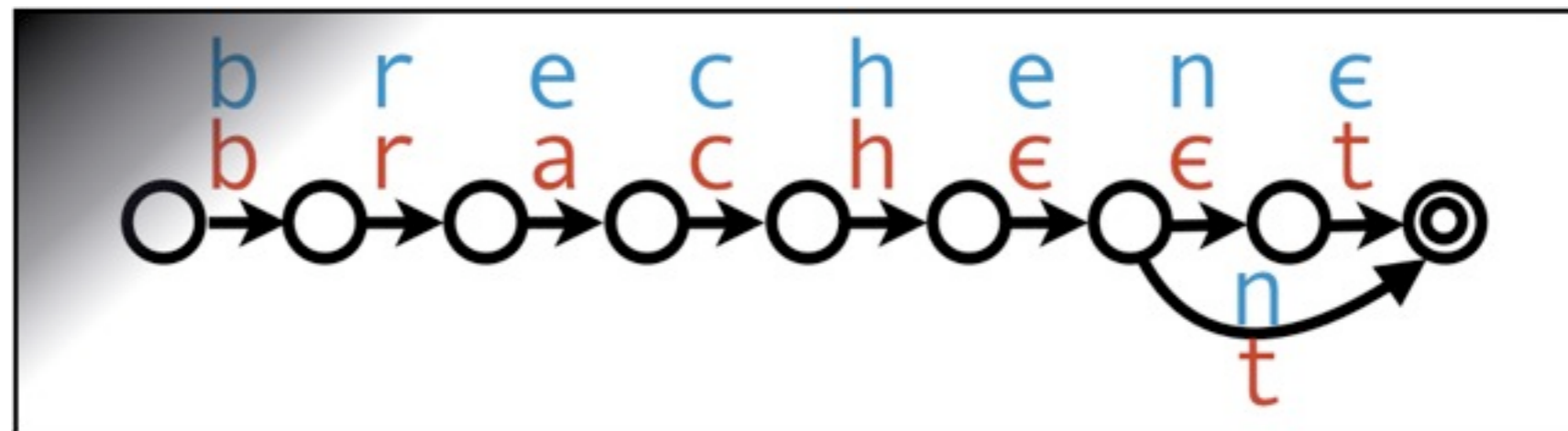


1

# String Pairs

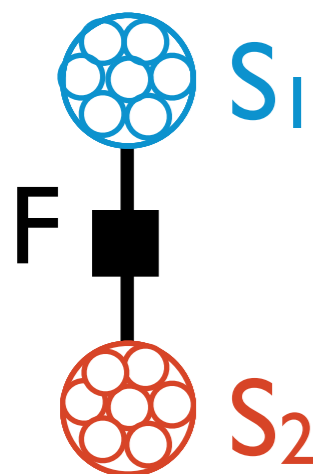
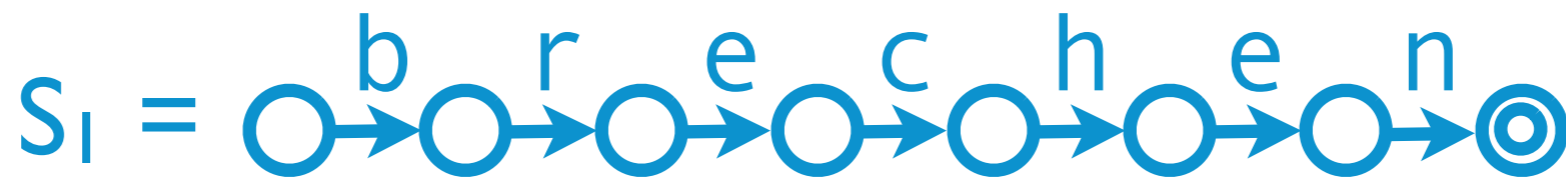


finite-state transducer

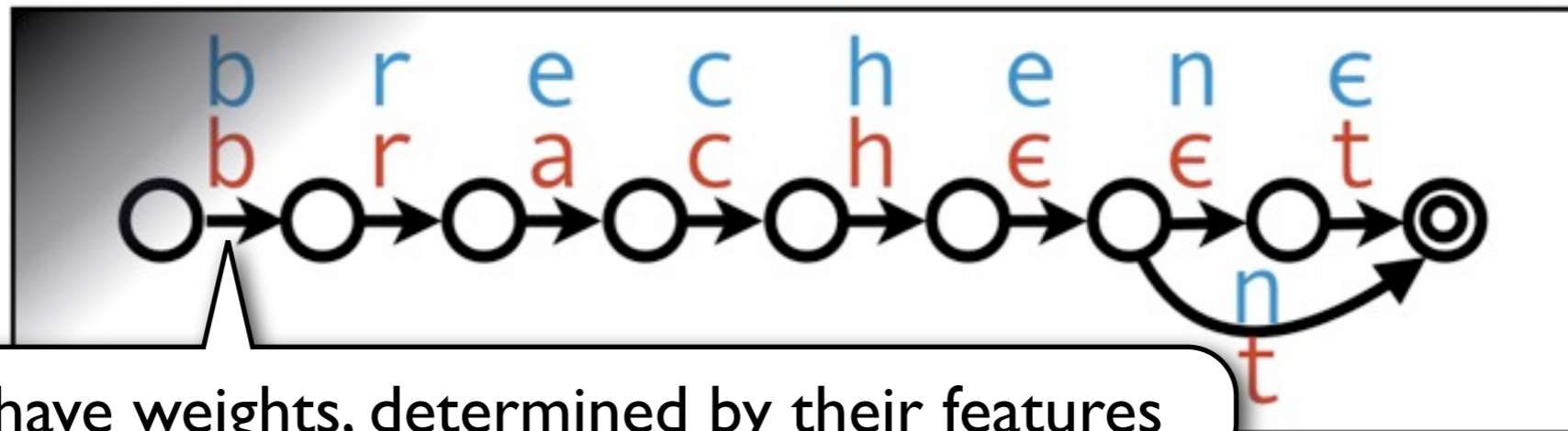


1

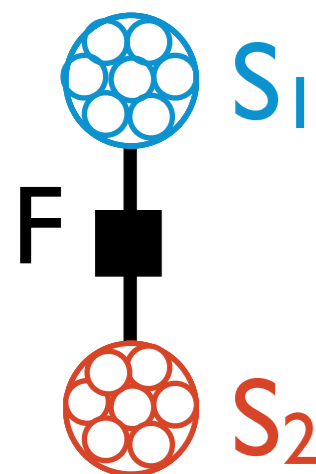
# String Pairs



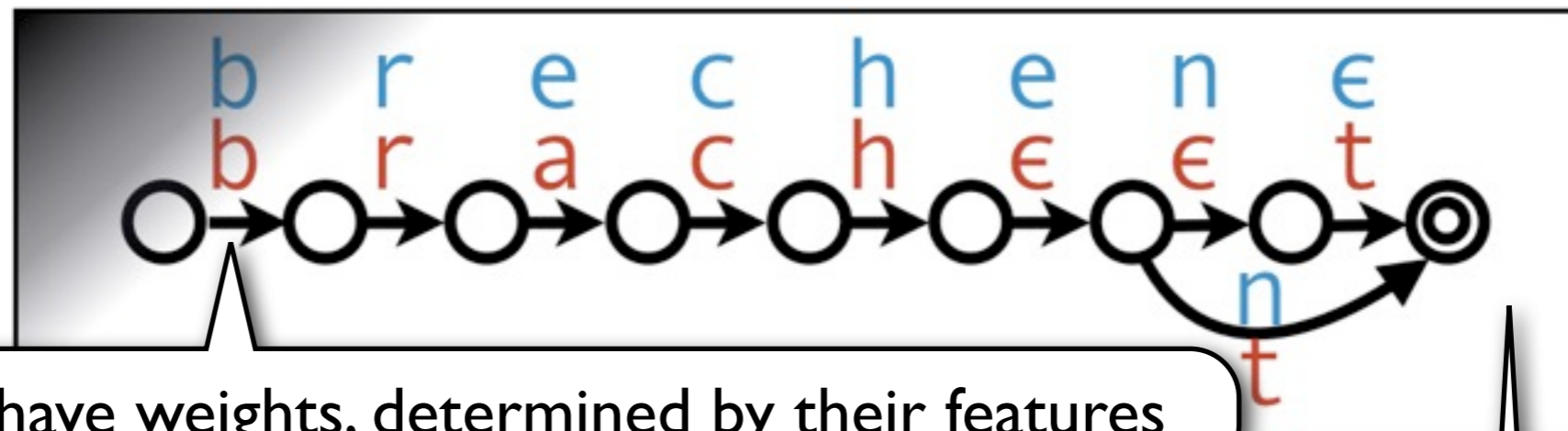
finite-state transducer



# String Pairs

$$S_1 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{e} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{e} \text{O} \xrightarrow{n} \text{O} \xrightarrow{\epsilon} \text{O}$$


finite-state transducer



arc have weights, determined by their features

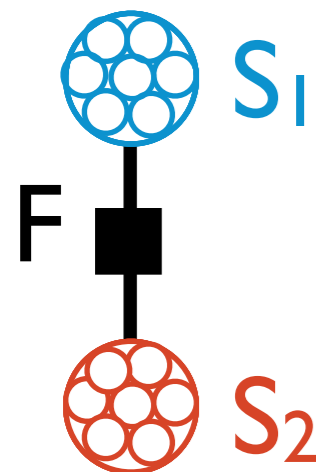
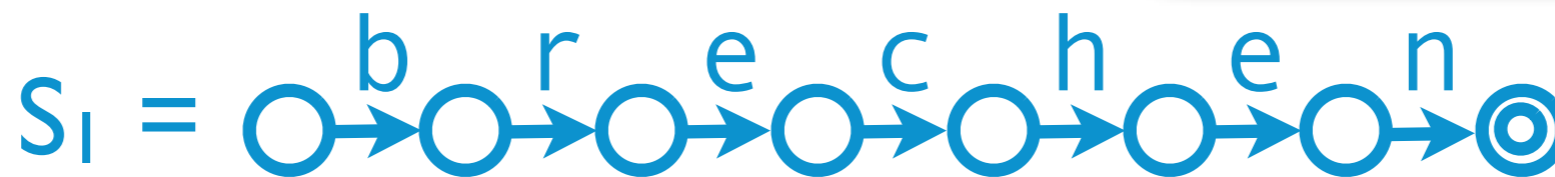
$$S_2 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{a} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{t} \text{O} \xrightarrow{\epsilon} \text{O}$$

Transducer  $F$  computes score by looking at **all** alignments

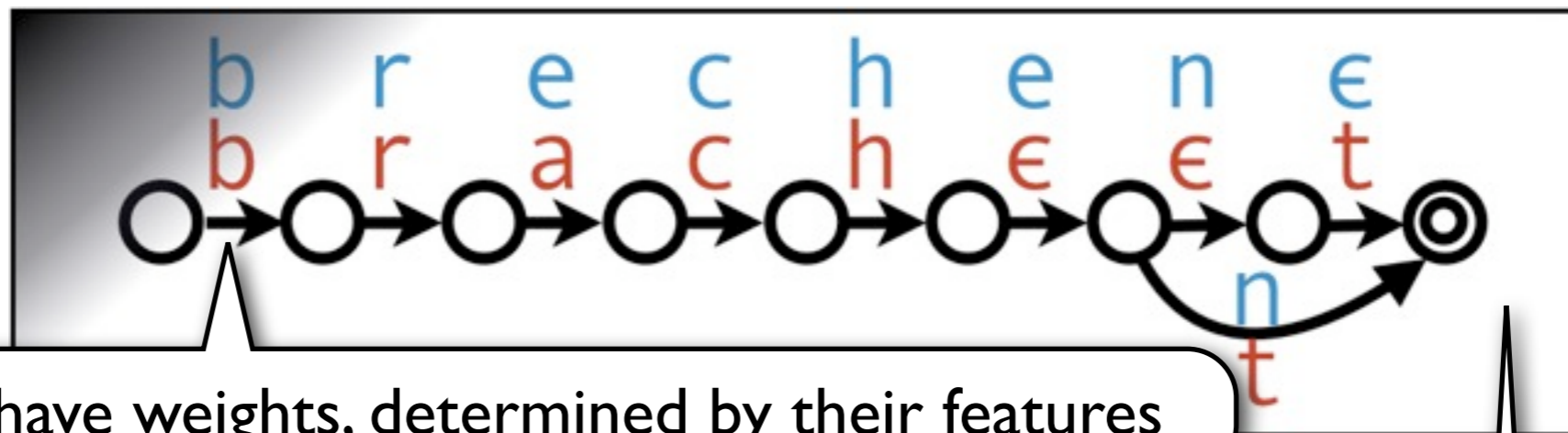
1

# String Pairs

Sum over all paths in the finite-state transducer



finite-state transducer



= 13.26

arc have weights, determined by their features



Transducer F computes score by looking at **all** alignments

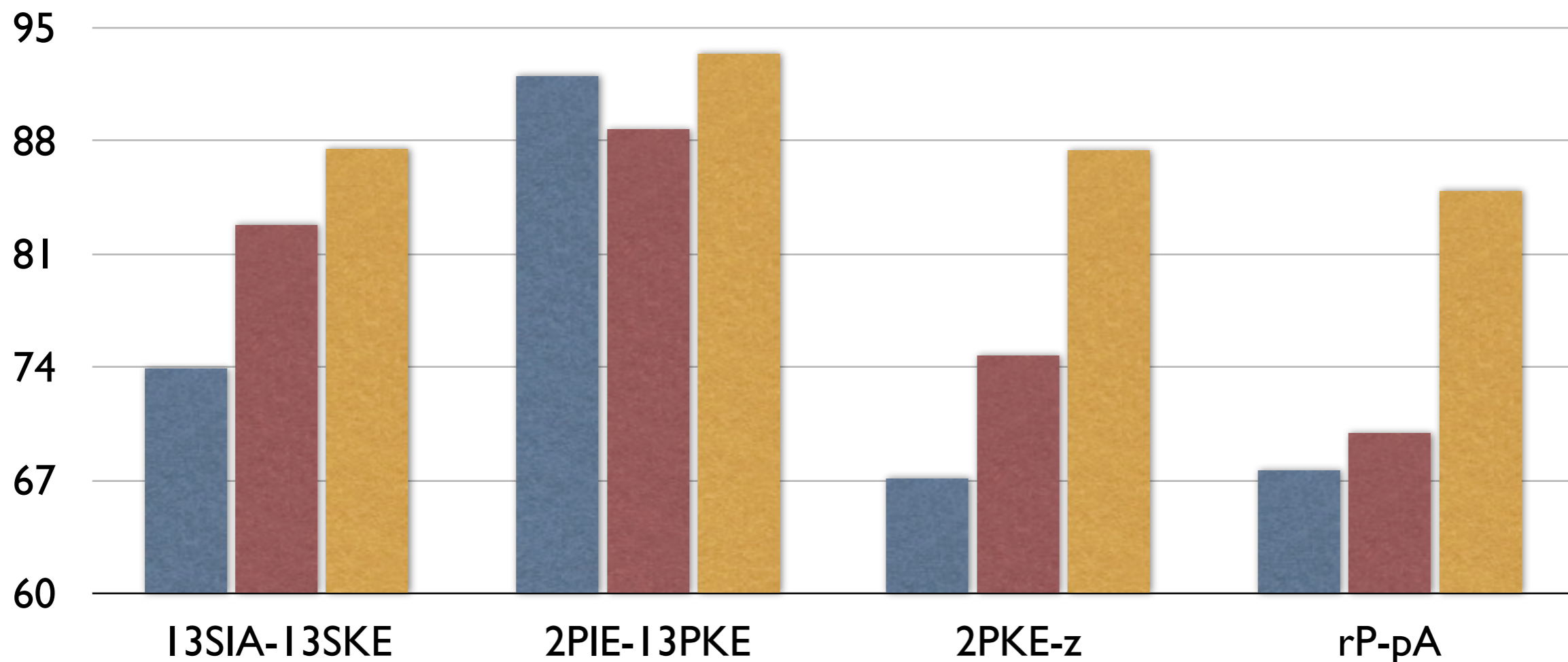
# String Pairs

- The alignment between the string pair is a latent variable.
- We add more latent variables to the model:
  - Change regions
  - Conjugations classes

For details, see my thesis, and  
Dreyer, Smith & Eisner, 2008

# String Pairs

## Inflection (on German verbs)



■ Moses3  
(baseline)

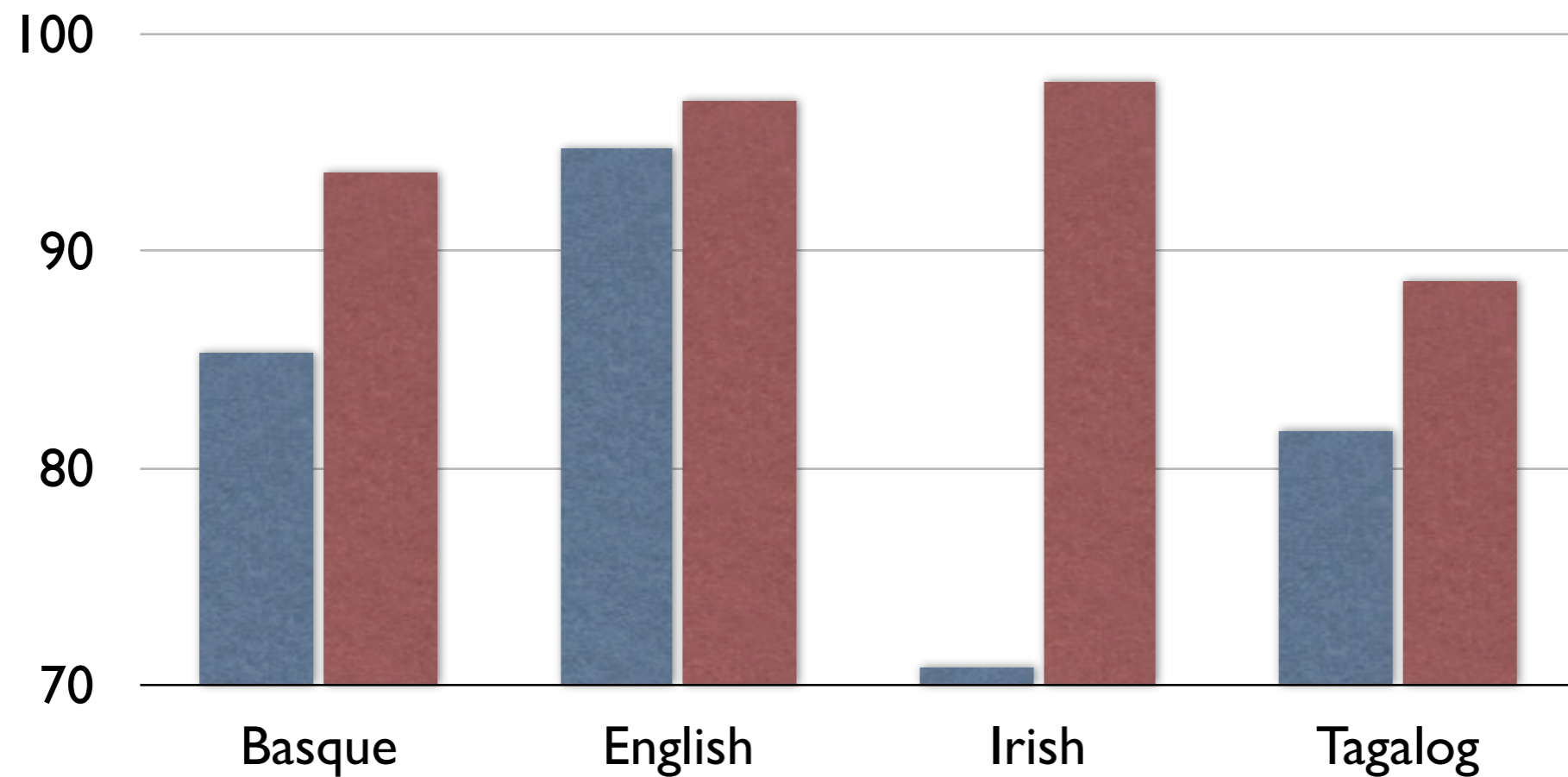
■ FST  
(this talk)

■ FST (+latent)

See my thesis, and Dreyer,  
Smith & Eisner, 2008

# String Pairs

## Lemmatization



■ Wicentowski (2002)  
■ This talk

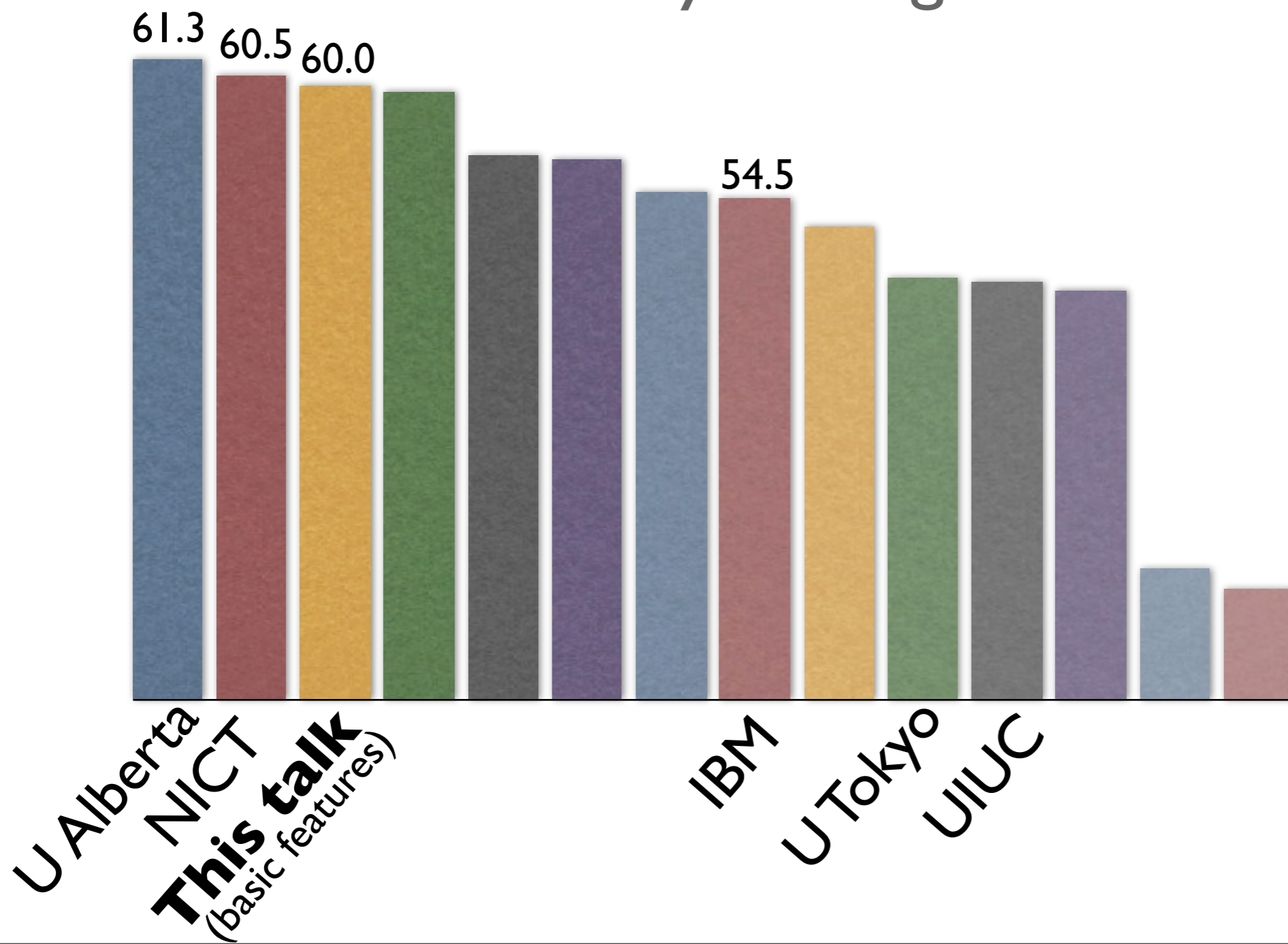
See my thesis, and Dreyer,  
Smith & Eisner, 2008



# String Pairs

## Transliteration competition, NEWS 2009

Accuracy on English-to-Russian



# 1 Conclusions / Contributions

- Presented a novel, well-defined **probability model** over string pairs (or single strings)
- **General** enough to model many string-to-string problems in NLP (and neighboring disciplines)
- Achieved high-scoring results in **different tasks** (inflection, lemmatization, transliteration) in **multiple languages** (*German, Basque, English, Irish, Tagalog, Russian*)

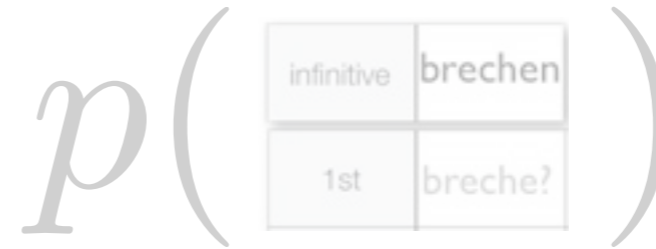
# 1 Conclusions / Contributions

- **Linguistic properties** and soft constraints can be expressed and learned (*prefer certain vowel/consonant sequences, prefer identities, ...*)
- **Arbitrary-length output** is handled elegantly (eliminates need for limiting structure insertion)
- Much information does not need to be annotated; it is inferred as **hidden variables** (alignments, conjugation classes, regions)

# Overview

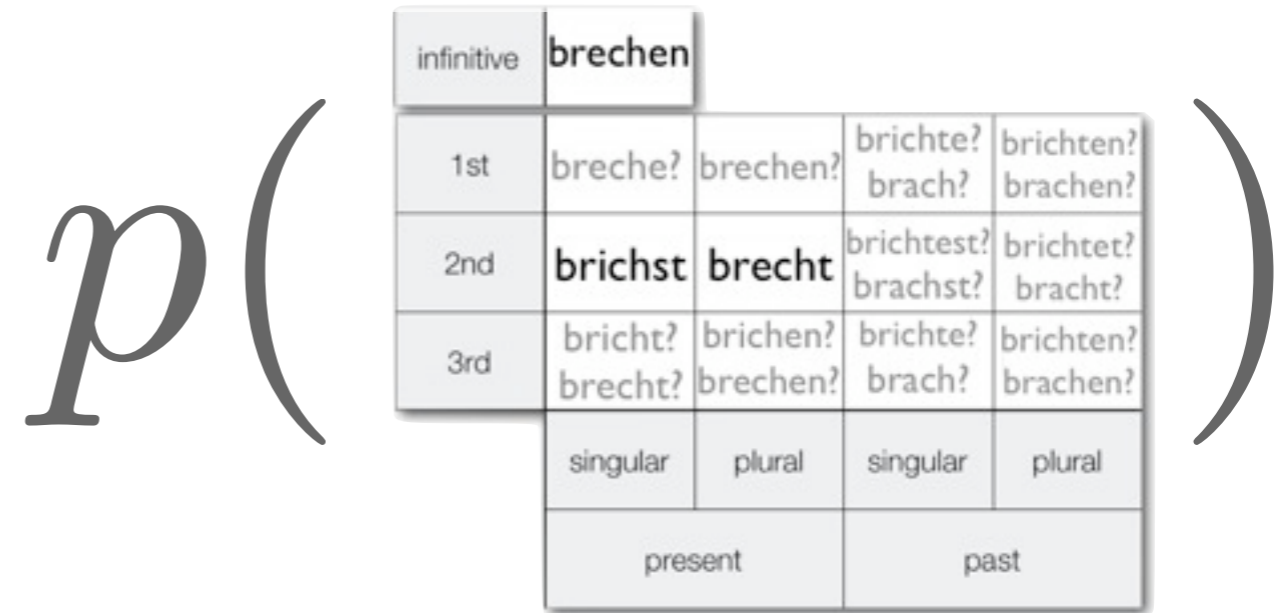
1

String pairs



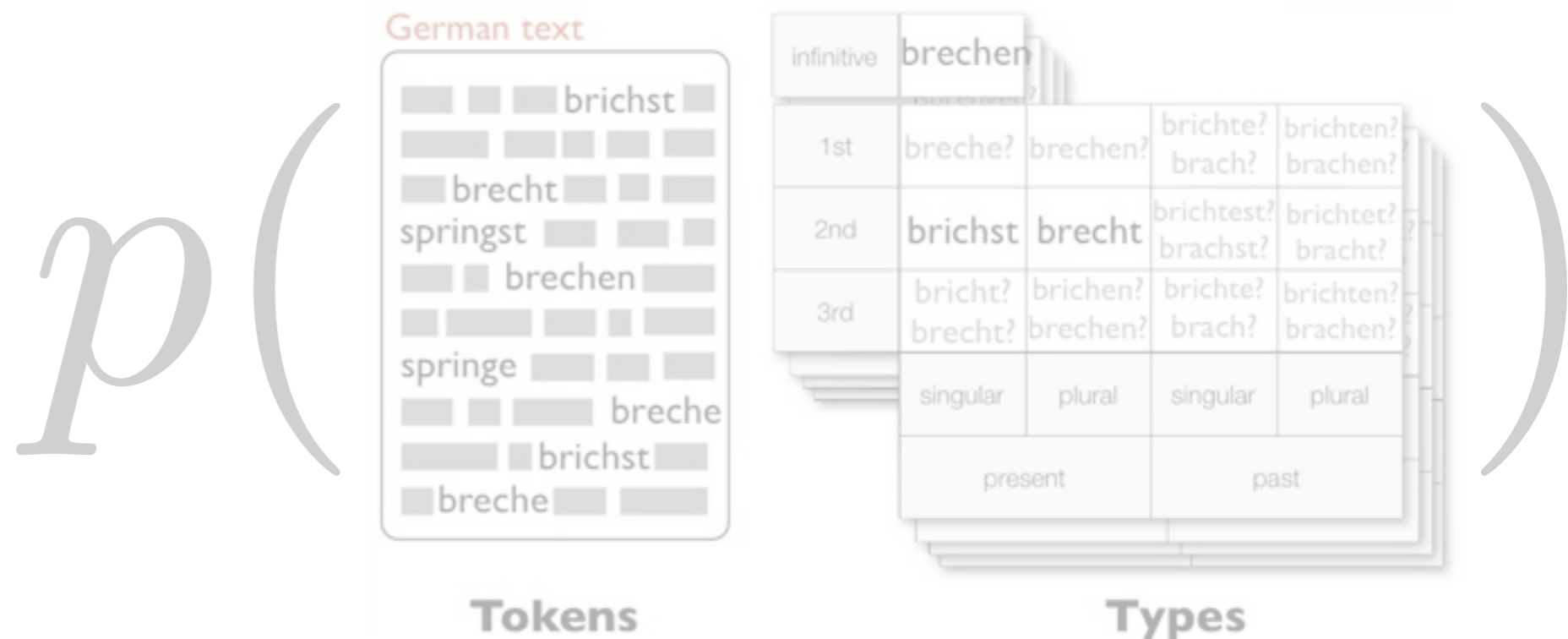
2

**Multiple strings (paradigms)**



3

Text and paradigms



# Multiple Strings

infinitive	brechen
1st	breche?

- We've seen how to model 2 strings, using feature-based finite-state machines
- But we have bigger goals ...

2

# Multiple Strings

infinitive	<b>brechen</b>
1st	breche?

## 2

# Example applications

## Inflectional paradigms

infinitive	<b>brechen</b>			
1st	breche?	brechen?	brichte? brach?	brichten? brachen?
2nd	<b>brichst</b>	<b>brecht</b>	brichstest? brachst?	brichtest? bracht?
3rd	bricht? brecht?	brichen? brechen?	brichte? brach?	brichten? brachen?
	singular	plural	singular	plural
	present		past	

2

# Example applications

**Inflectional paradigms**



## 2

# Example applications

## Inflectional paradigms

infinitive	brechen			
1st	brech <sup>?</sup> e	brech <sup>?</sup> en	brach <sup>?</sup>	brach <sup>?</sup> en
2nd	brich <sup>?</sup> st	brech <sup>?</sup> t	brach <sup>?</sup> st	brach <sup>?</sup> t
3rd	brich <sup>?</sup> t	brech <sup>?</sup> en	brach <sup>?</sup>	brach <sup>?</sup> en
	singular	plural	singular	plural
	present		past	

# 2

# Example applications

## Inflectional paradigms

infinitive	<b>brechen</b>			
1st	breche <b>e</b>	breche <b>n</b>	brach	<b>brachen</b>
2nd	brich <b>st</b>	<b>brecht</b>	brach <b>st</b>	brach <b>t</b>
3rd	bricht <b>t</b>	breche <b>n</b>	brach	brach <b>e</b> n
	singular	plural	singular	plural
	present		past	

The diagram illustrates the inflectional paradigm for the German verb 'brechen'. The table shows the conjugation for the present and past tenses, singular and plural. Red circles highlight the infinitive form 'brechen', the 1st person singular present form 'breche', the 2nd person singular present form 'brecht', and the 1st person plural past form 'brachen'. Red arrows labeled 'predict' point from the infinitive form to the 1st person singular present form and from the 1st person singular present form to the 1st person plural past form.

# 2

# Example applications

## Inflectional paradigms

infinitive	<b>brechen</b>			
1st	breche <b>e</b>	breche <b>en</b>	brach	<b>brachen</b>
2nd	brich <b>st</b>	<b>brecht</b>	brach <b>st</b>	brach <b>t</b>
3rd	bricht <b>t</b>	breche <b>en</b>	brach	brach <b>en</b>
	singular	plural	singular	plural
	present		past	

The diagram illustrates the inflectional paradigm for the German verb 'brechen'. The infinitive form 'brechen' is circled in red. Two red arrows labeled 'predict' point from 'brechen' to the 1st person plural present form 'brachen' and the 2nd person singular present form 'brecht', which are also circled in red. The table below shows the full paradigm for 'brechen'.

# 2

# Example applications

## Inflectional paradigms

infinitive	<b>brechen</b>			
1st	breche <b>e</b>	breche <b>en</b>	brach	<b>brachen</b>
2nd	brich <b>st</b>	<b>brecht</b>	brache <b>t</b>	<b>bracht</b>
3rd	bricht <b>t</b>	breche <b>n</b>	brach	brache <b>n</b>
	singular	plural	singular	plural
	present		past	

The diagram illustrates the inflectional paradigm for the German verb 'brechen'. The table shows the conjugation for the present and past tenses, singular and plural. Red circles highlight the infinitive 'brechen', the 1st person singular present 'breche', the 2nd person singular present 'brecht', the 1st person plural present 'brachen', and the 2nd person singular past 'bracht'. Red arrows labeled 'predict' point from 'brechen' to 'brachen' and from 'brechen' to 'bracht', indicating the prediction of these forms from the infinitive.

# 2

# Example applications

## Inflectional paradigms

infinitive	<b>brechen</b>			
1st	breche	<b>brechen</b>	brach	<b>brachen</b>
2nd	brichst	<b>brecht</b>	brachet	<b>bracht</b>
3rd	<b>bricht</b>	brechen	brach	brachen
	singular	plural	singular	plural
	present		past	

The diagram illustrates the inflectional paradigm for the verb 'brechen' (to break). The table shows the forms for the present and past tenses, singular and plural, and first and second person. Red arrows and circles highlight the following relationships:

- predict**: From the infinitive 'brechen' to the 1st person singular present 'breche' and the 1st person plural present 'brachen'.
- predict**: From the 1st person singular present 'breche' to the 2nd person singular present 'brecht'.
- reinforce**: From the 1st person plural present 'brachen' to the 2nd person plural present 'bracht'.
- reinforce**: From the 2nd person singular present 'brecht' to the 2nd person plural present 'bracht'.

2

# Example applications

## Transliteration (using phonology)

ice cream

English orthography

? aɪ s k r iː m

English phonology

アイスクリーム

Japanese orthography

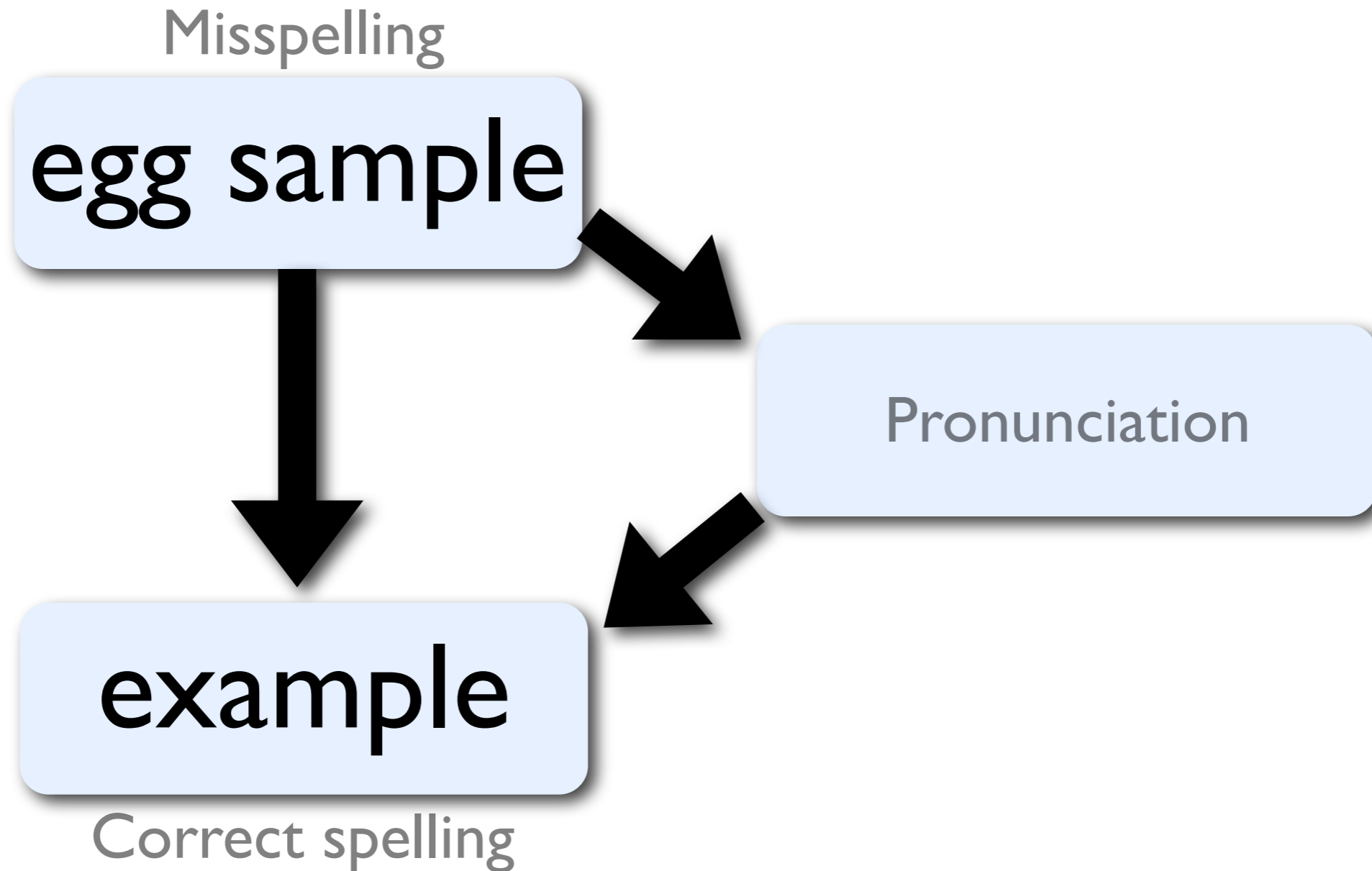
? aɪ s u k u ɹ iː m u

Japanese phonology

2

# Example applications

## Spelling correction



# 2 Example applications

**... and all other tasks** where word forms and representations interact:

- Cognate modeling
- Multiple-string alignment
- System combination

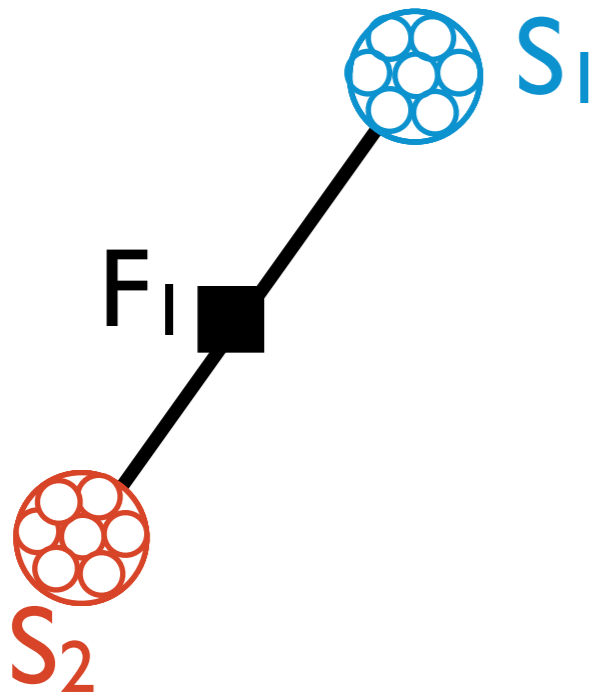


- Let's build a general **probability model over multiple strings**
- It extends the string-pair model we saw in the last part.
- We will later be able to use it to learn how to inflect verbs.

2

# Model. *Factor graph examples*

*Factor Graph:*

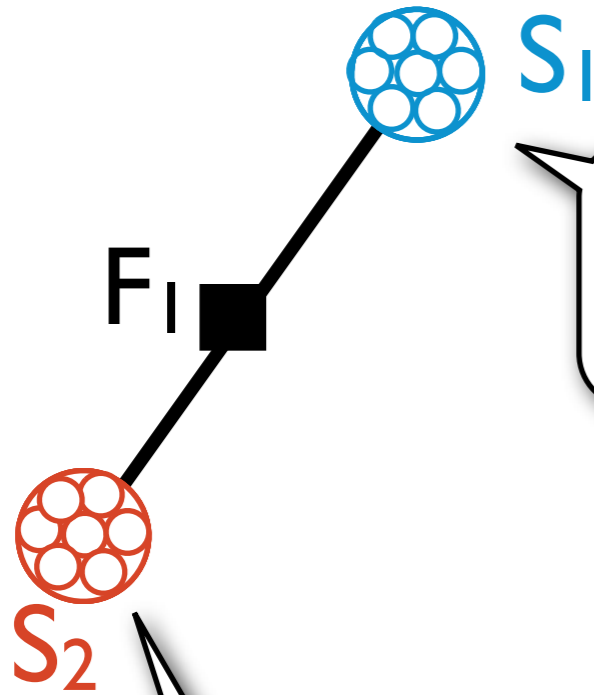


$$\Pr(s_1, s_2) = \frac{1}{Z} \times F_1(s_1, s_2)$$

2

# Model. *Factor graph examples*

*Factor Graph:*



Random variable,  
ranges over any string

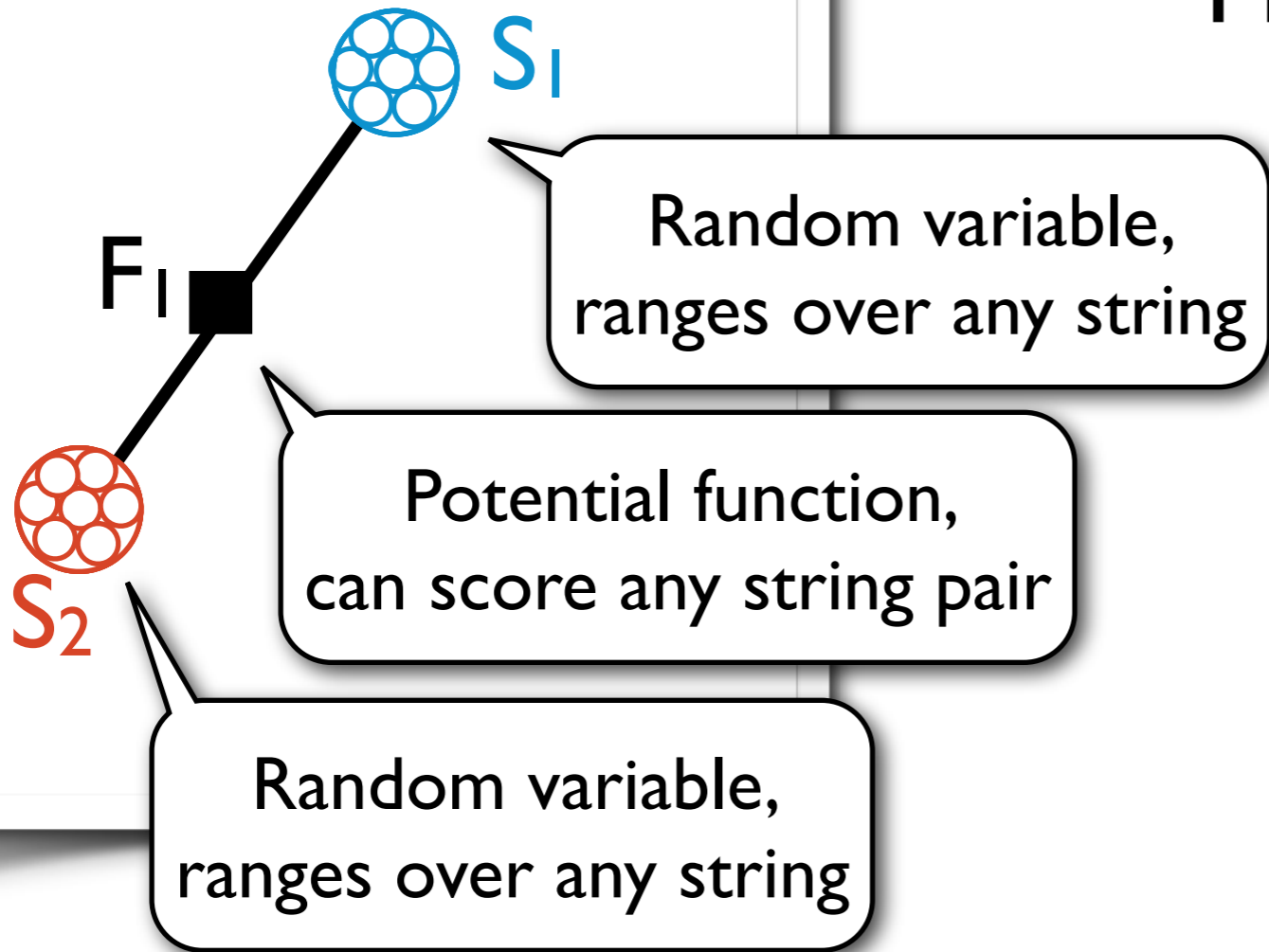
Random variable,  
ranges over any string

$$\Pr(s_1, s_2) = \frac{1}{Z} \times F_1(s_1, s_2)$$

## 2

Model. *Factor graph examples*

*Factor Graph:*

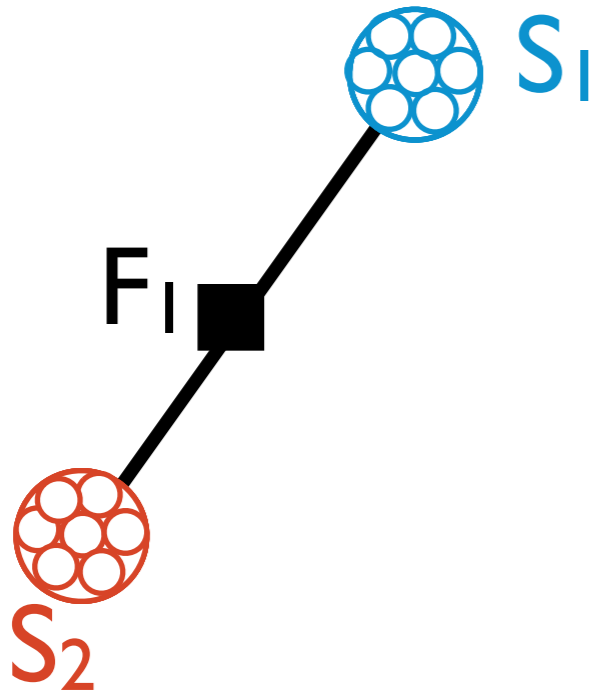


$$\Pr(s_1, s_2) = \frac{1}{Z} \times F_1(s_1, s_2)$$

2

# Model. *Factor graph examples*

*Factor Graph:*

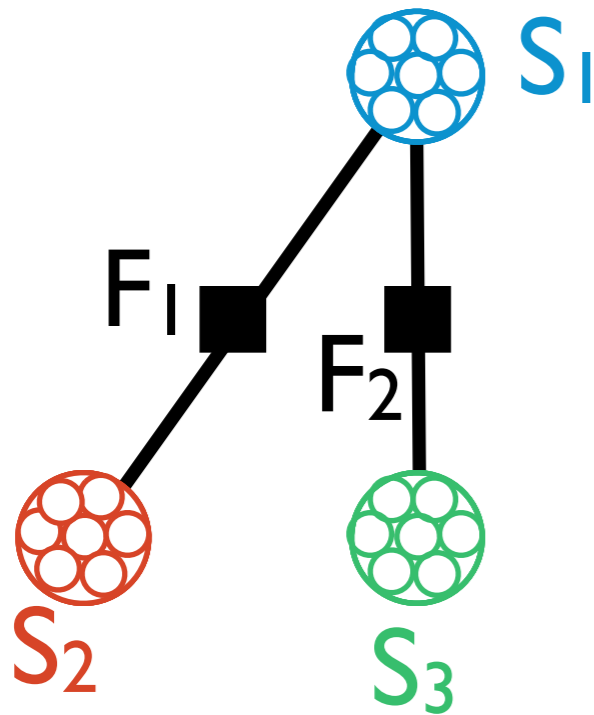


$$\Pr(s_1, s_2) = \frac{1}{Z} \times F_1(s_1, s_2)$$

2

# Model. *Factor graph examples*

*Factor Graph:*

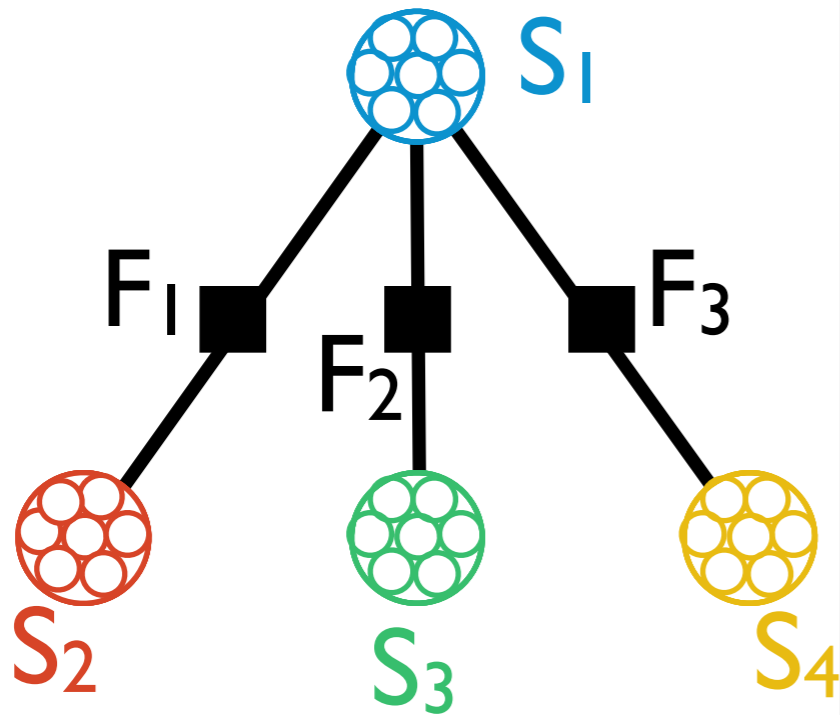


$$\Pr(s_1, s_2, s_3) = \frac{1}{Z} \times F_1(s_1, s_2) \times F_2(s_1, s_3)$$

2

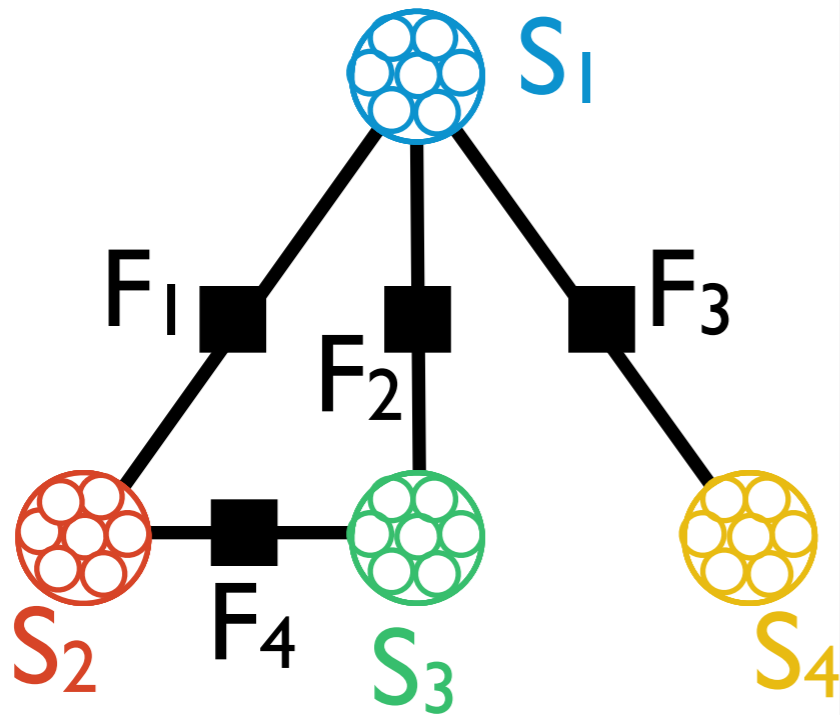
# Model. *Factor graph examples*

*Factor Graph:*



$$\Pr(s_1, s_2, s_3, s_4) = \frac{1}{Z} \\ \times F_1(s_1, s_2) \\ \times F_2(s_1, s_3) \\ \times F_3(s_1, s_4)$$

## 2

Model. *Factor graph examples**Factor Graph:*

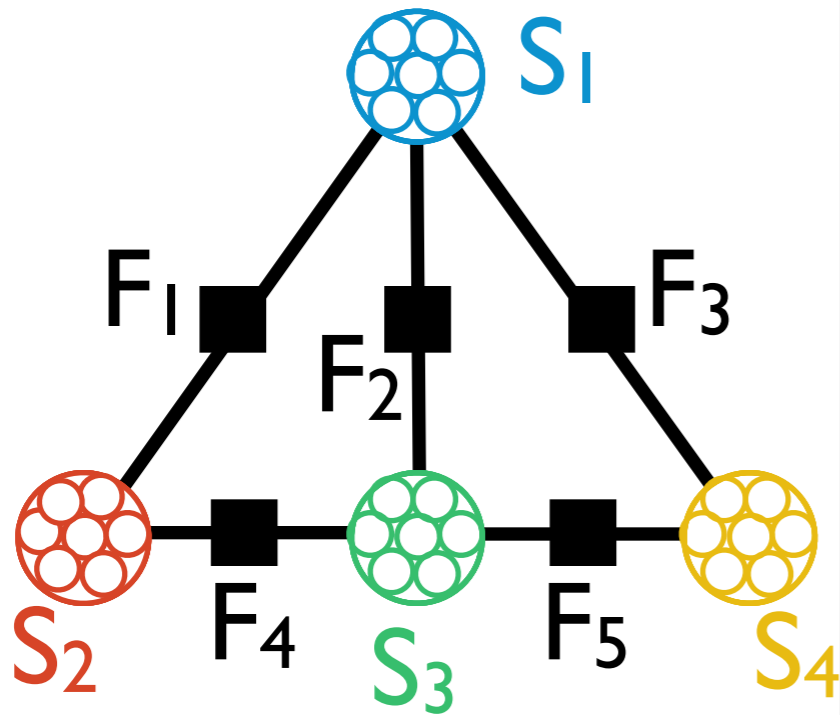
$$\Pr(s_1, s_2, s_3, s_4) = \frac{1}{Z} \times F_1(s_1, s_2) \times F_2(s_1, s_3) \times F_3(s_1, s_4) \times F_4(s_2, s_3)$$



2

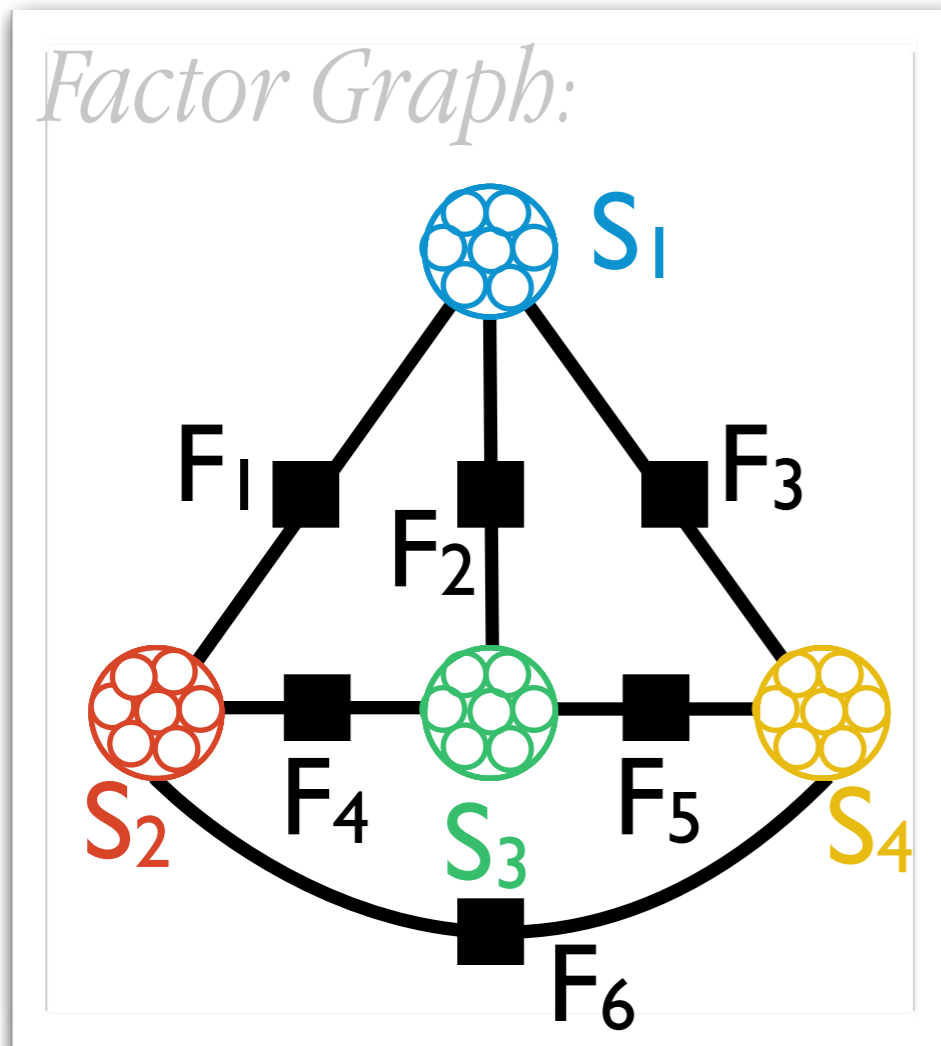
# Model. *Factor graph examples*

*Factor Graph:*



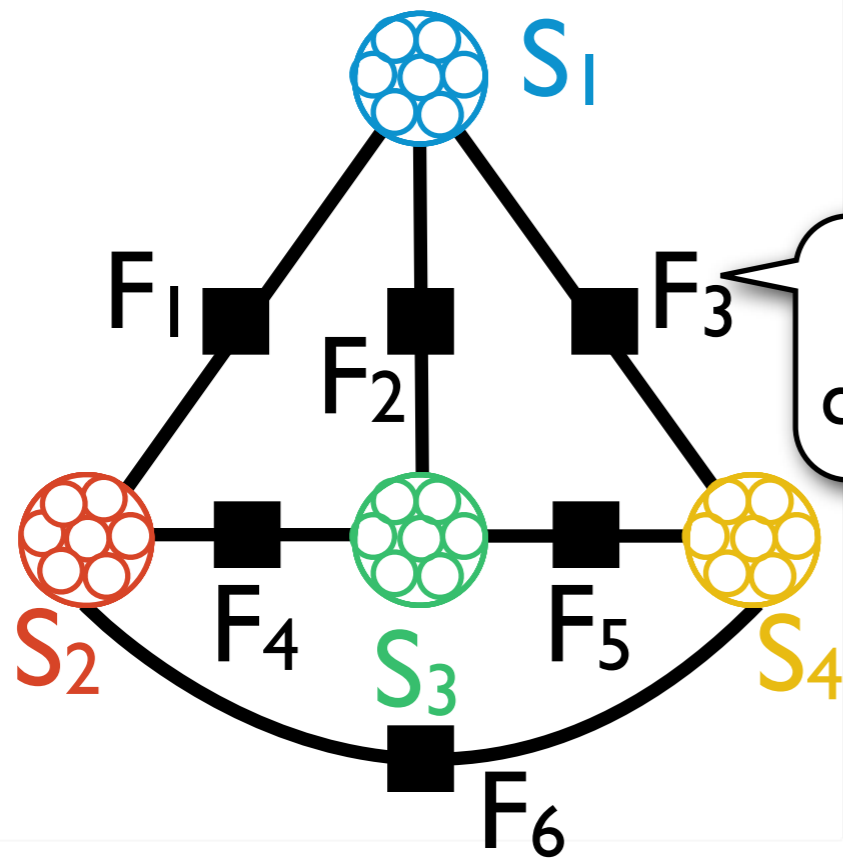
$$\begin{aligned} \Pr(s_1, s_2, s_3, s_4) = & \frac{1}{Z} \\ & \times F_1(s_1, s_2) \\ & \times F_2(s_1, s_3) \\ & \times F_3(s_1, s_4) \\ & \times F_4(s_2, s_3) \\ & \times F_5(s_3, s_4) \end{aligned}$$

## 2

Model. *Factor graph examples*

$$\Pr(s_1, s_2, s_3, s_4) = \frac{1}{Z} \\ \times F_1(s_1, s_2) \\ \times F_2(s_1, s_3) \\ \times F_3(s_1, s_4) \\ \times F_4(s_2, s_3) \\ \times F_5(s_3, s_4) \\ \times F_6(s_2, s_4)$$

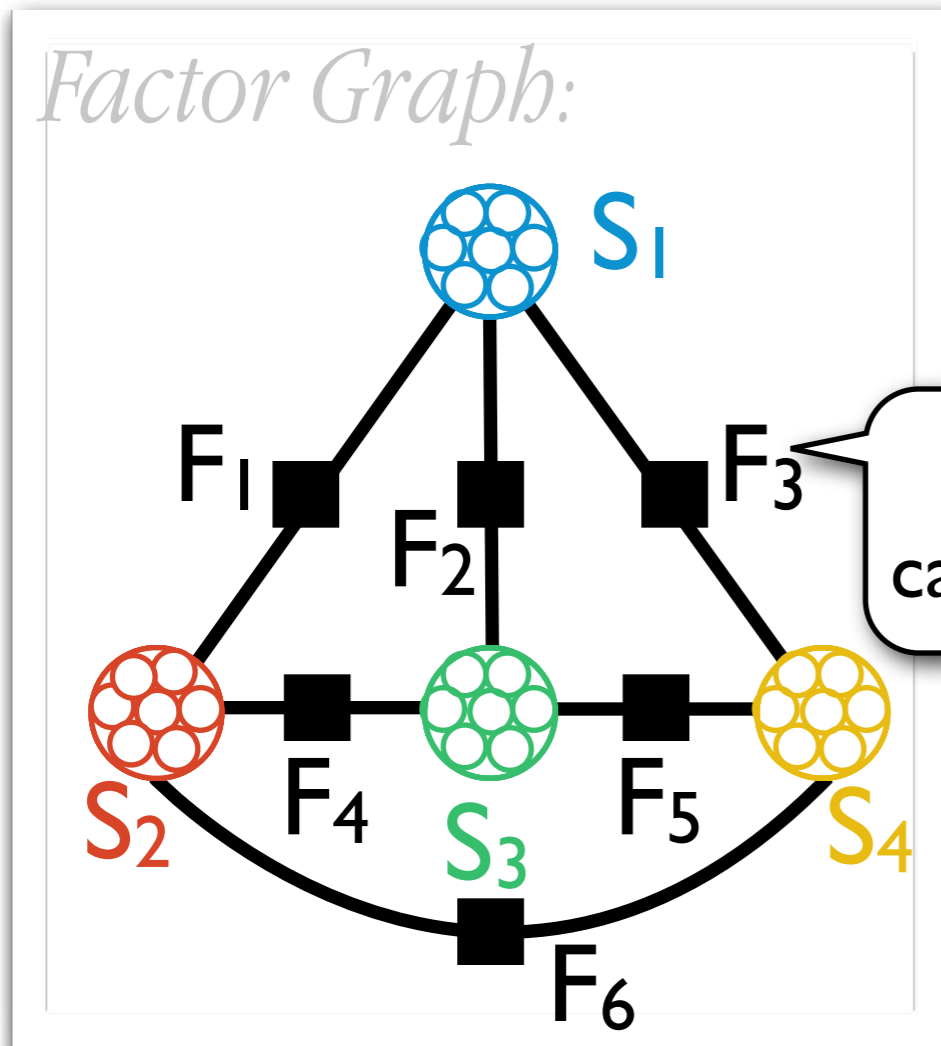
## 2

Model. *Factor graph examples**Factor Graph:*

Potential function,  
can score any string pair

$$\Pr(s_1, s_2, s_3, s_4) = \frac{1}{Z} \times F_1(s_1, s_2) \times F_2(s_1, s_3) \times F_3(s_1, s_4) \times F_4(s_2, s_3) \times F_5(s_3, s_4) \times F_6(s_2, s_4)$$

## 2

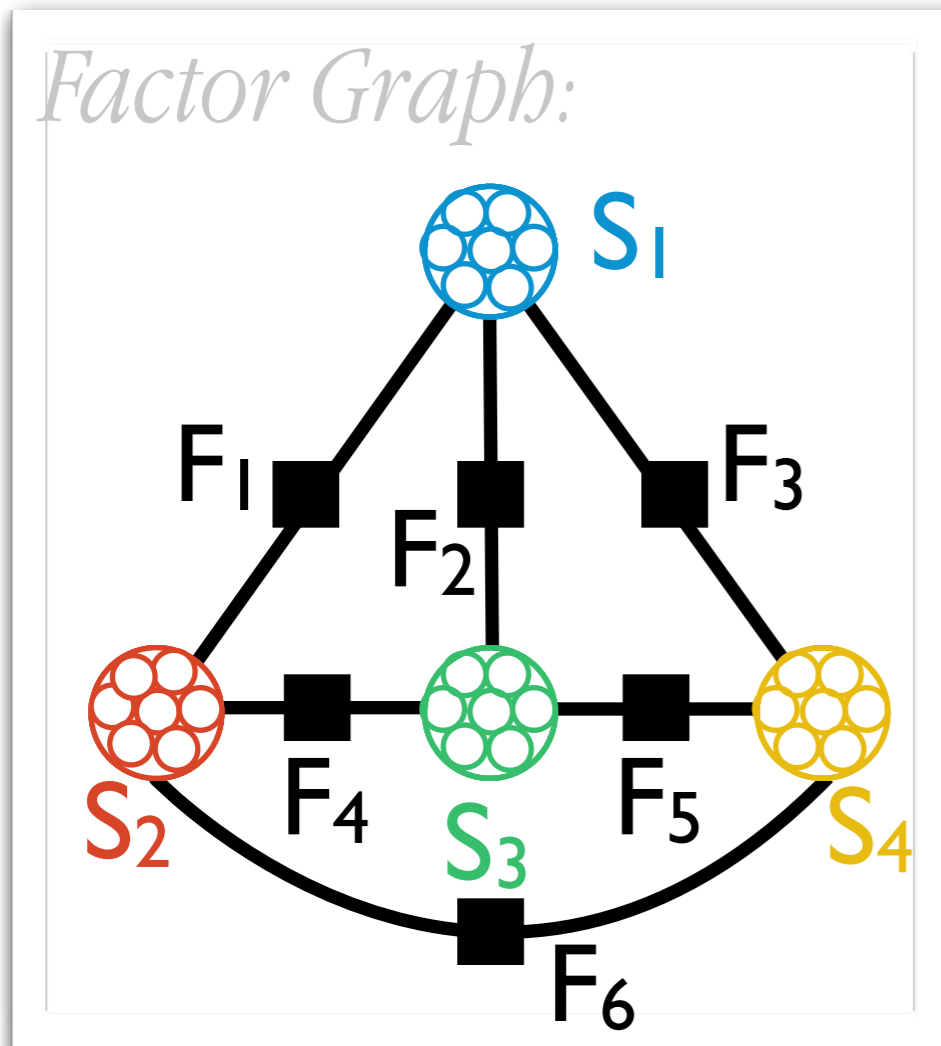
Model. *Factor graph examples*

Potential function,  
can score any string pair

Each potential function  $\mathbf{F}$   
is computed by a **finite-  
state transducer**.

2

# Model. *Factor graph examples*



$$\begin{aligned}
 \Pr(s_1, s_2, s_3, s_4) = & \frac{1}{Z} \\
 & \times F_1(s_1, s_2) \\
 & \times F_2(s_1, s_3) \\
 & \times F_3(s_1, s_4) \\
 & \times F_4(s_2, s_3) \\
 & \times F_5(s_3, s_4) \\
 & \times F_6(s_2, s_4)
 \end{aligned}$$

A formal description of such a model ...

- It is formally an undirected **graphical model** (a.k.a. Markov Random Field, *MRF*),
- in which the **variables are string-valued**, and the **factors** (potential functions) **are finite-state transducers.**

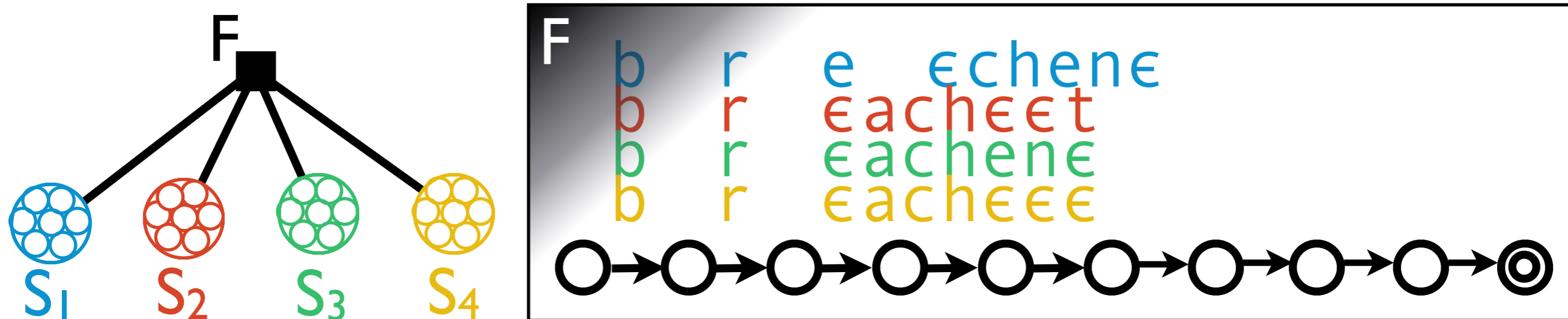
To **model multiple strings** and their various interactions, I ...

- use many **finite-state transducers**,
- have each of them look at a different string pair,
- **plug them together** into a big network,
- and **coordinate them** to predict all strings jointly (also: *train* the transducers jointly).

# 2

# Model. *Comparison with k-tape FSM*

- Model k strings with a k-tape finite-state machine?



- Factored model more powerful:
  - ☺ Encode swaps and other useful models
  - ☹ Encode undecidable models



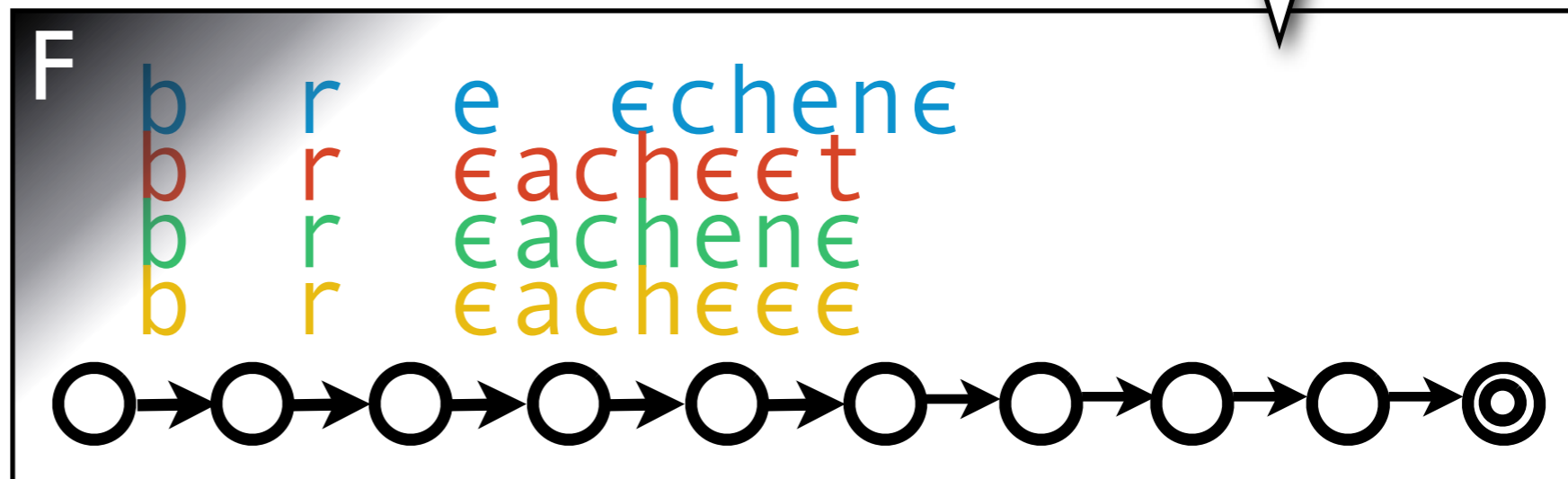
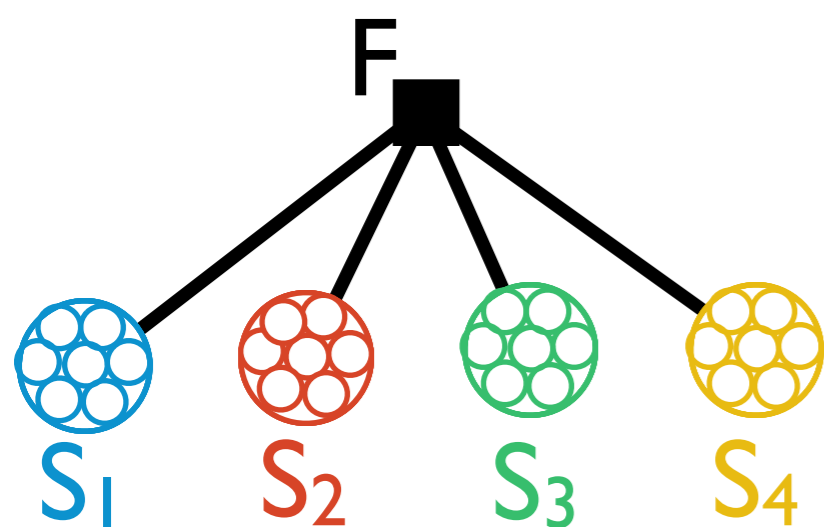
# 2

# Model. *Comparison with k-tape FSM*

- Model k strings with a k-tape finite-state machine?

- $>26^k$  arcs, intractable!

Multiple-sequence alignment

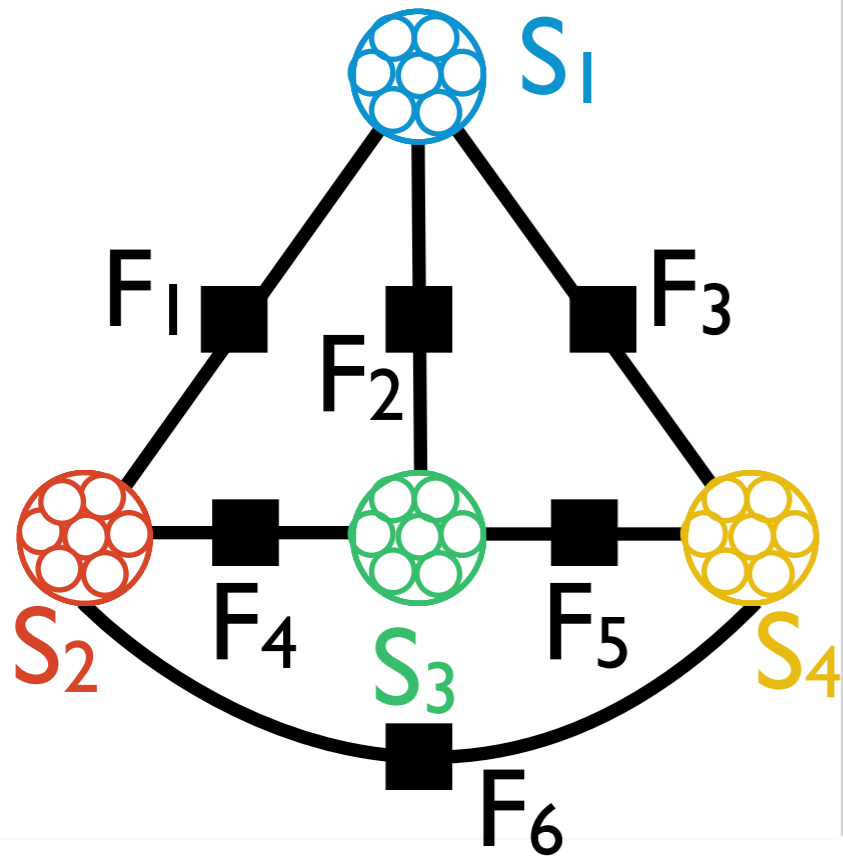


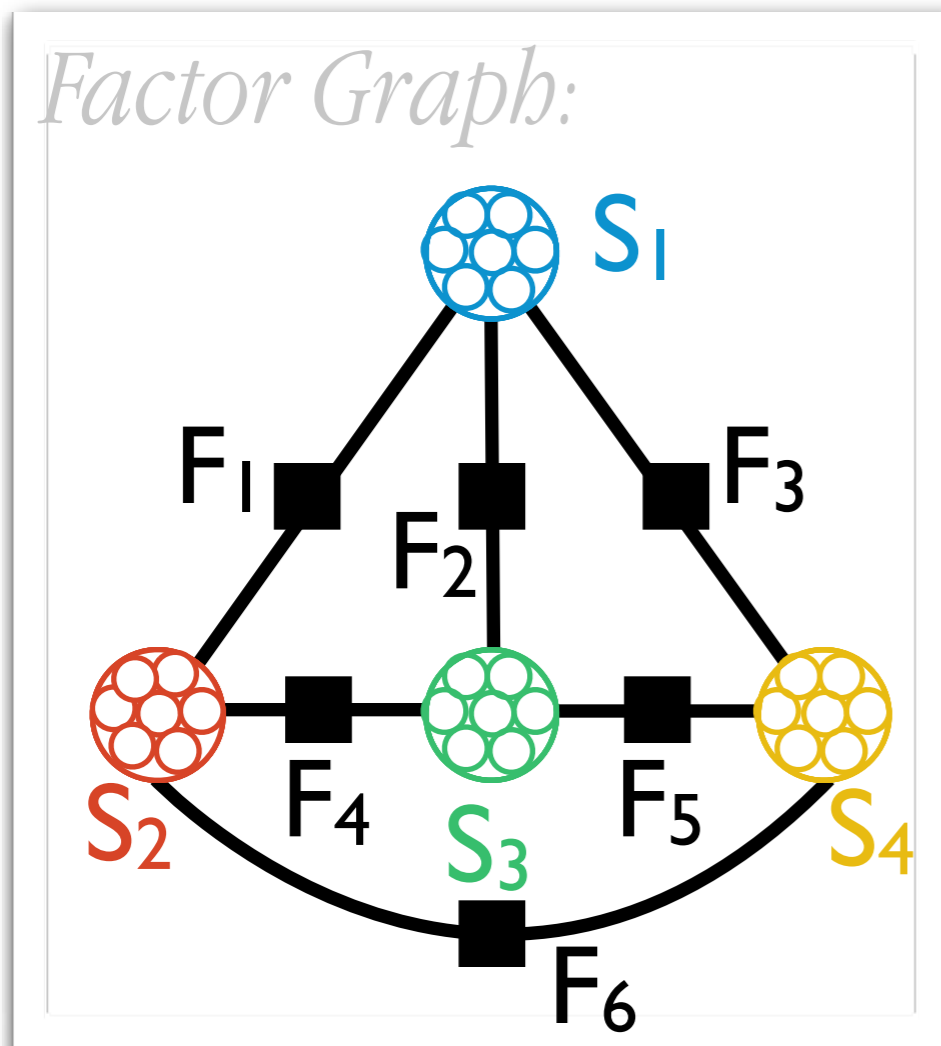
- Factored model more powerful:

☺ Encode swaps and other useful models

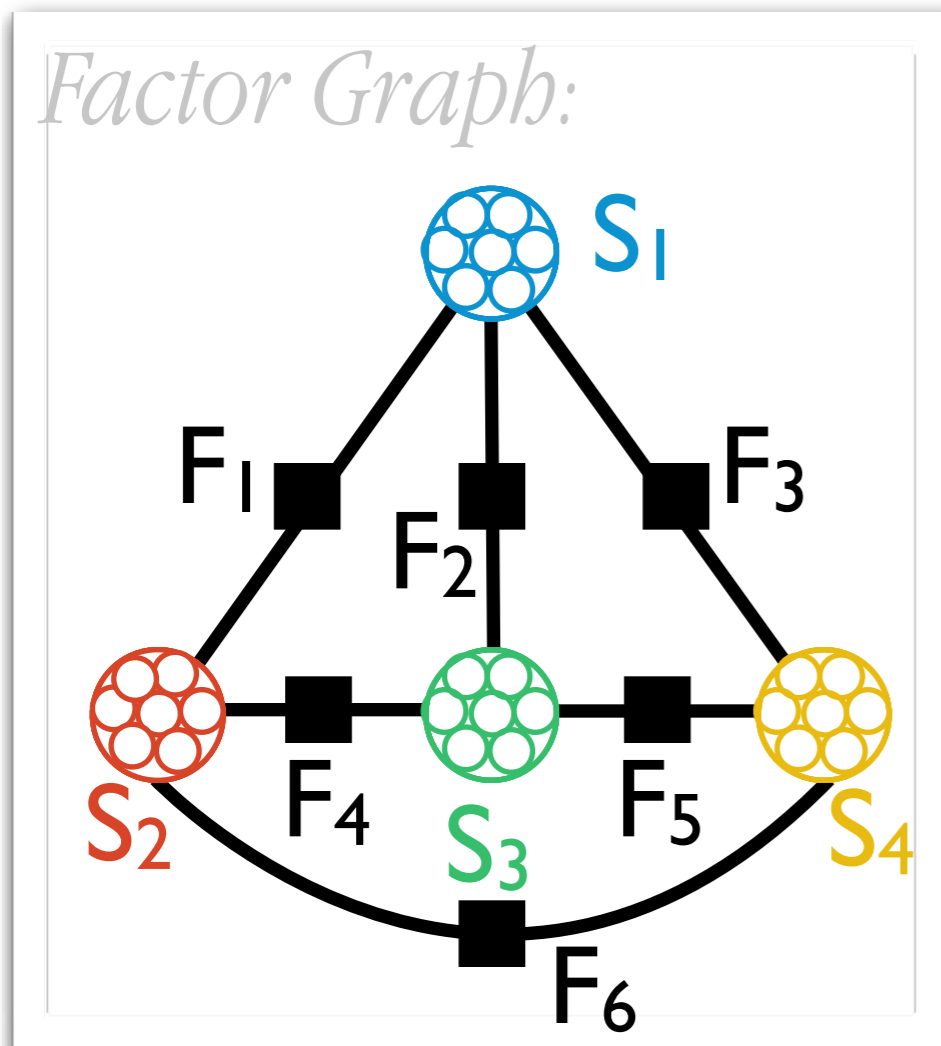
☹ Encode undecidable models

*Factor Graph:*

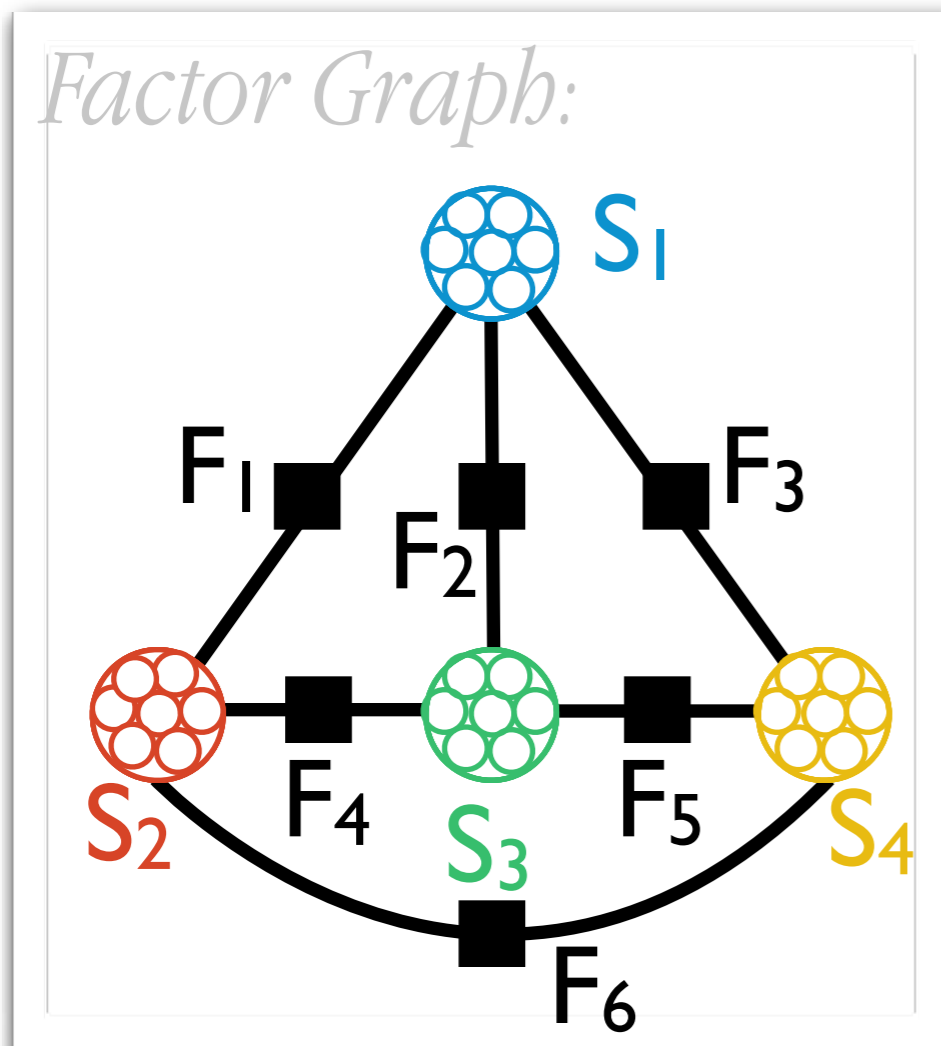




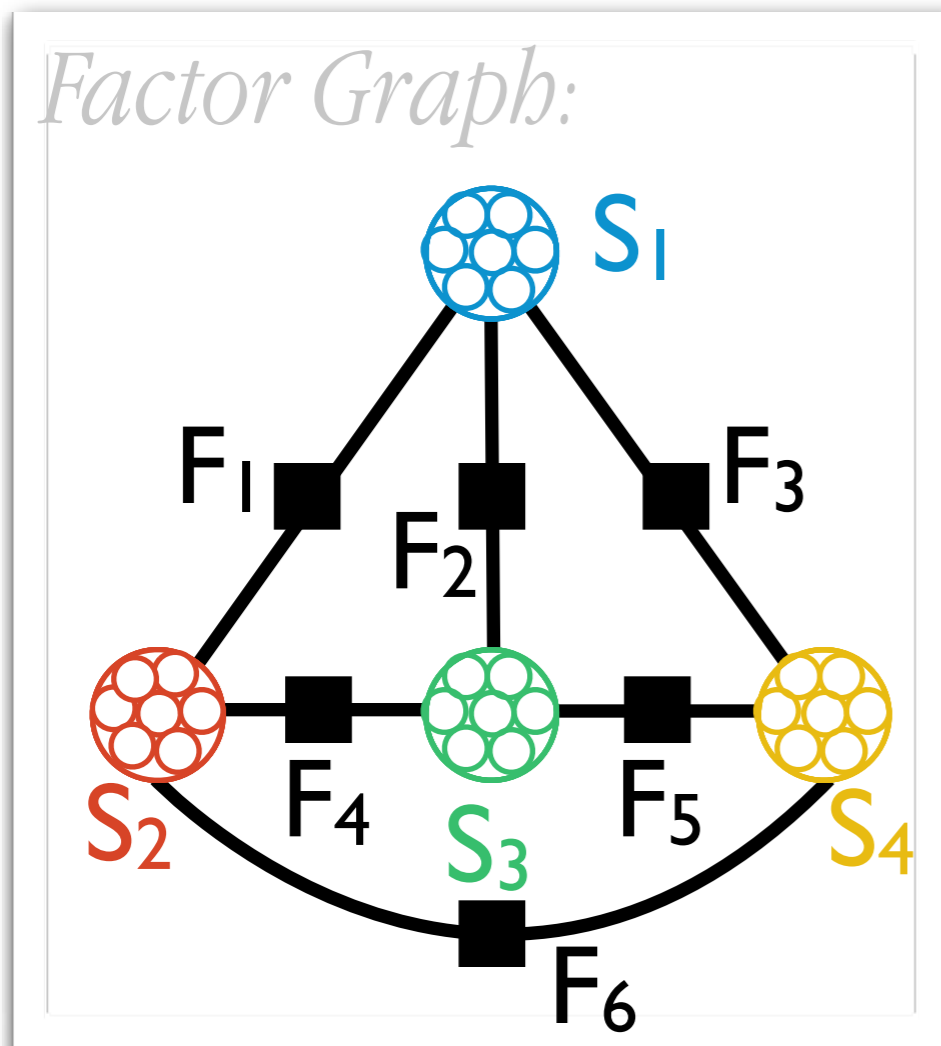
- Run **Belief Propagation (BP)**



- Run **Belief Propagation (BP)**
- BP is a message-passing algorithm, a **generalization of forward-backward.**



- Run **Belief Propagation (BP)**
- BP is a message-passing algorithm, a **generalization of forward-backward.**
- BP computes marginals



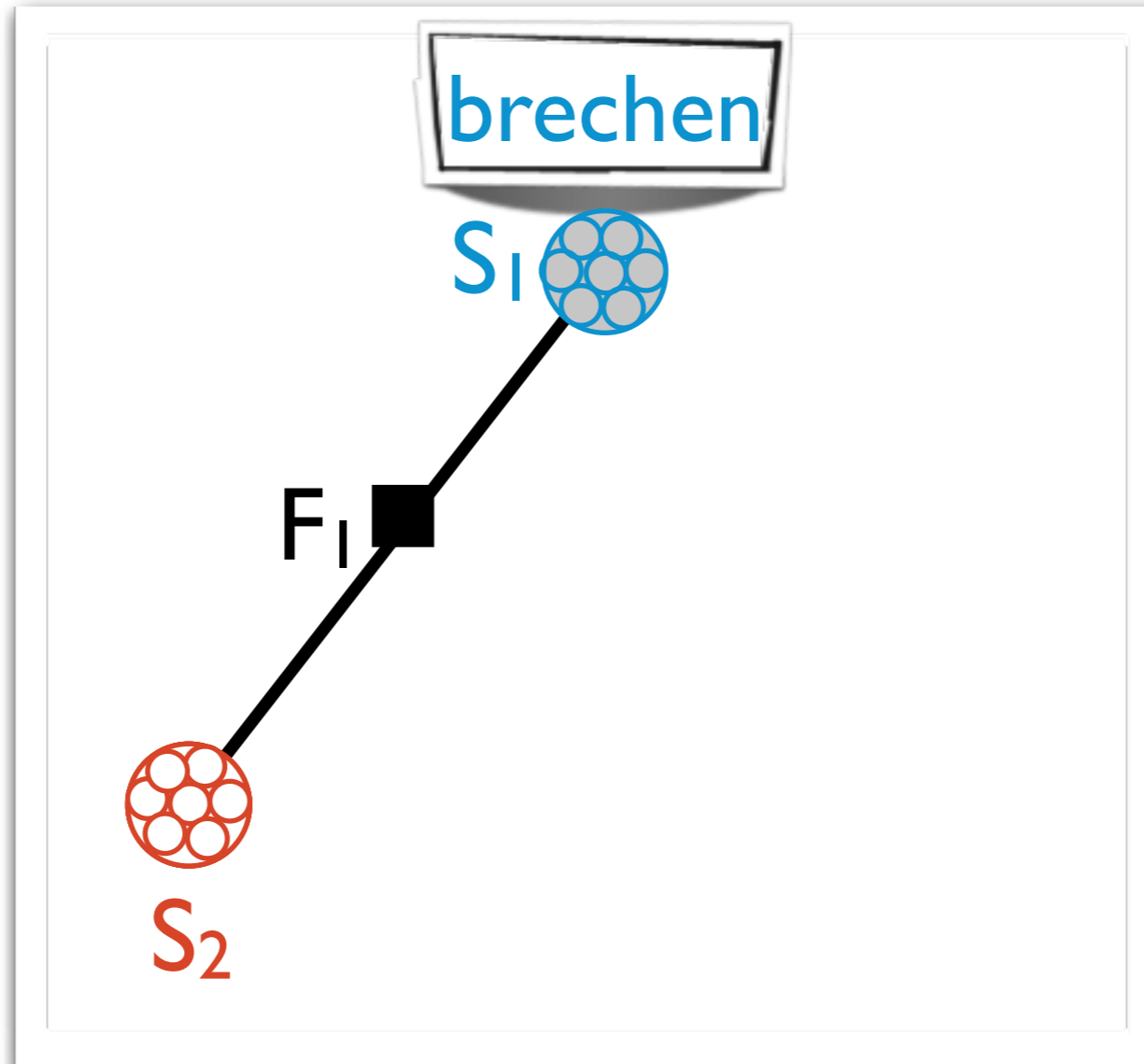
- Run **Belief Propagation (BP)**
- BP is a message-passing algorithm, a **generalization of forward-backward.**
- BP computes marginals

In our version of BP, **all messages and beliefs are finite-state machines**, which is novel.

# 2

# Inference. *Multiple strings*

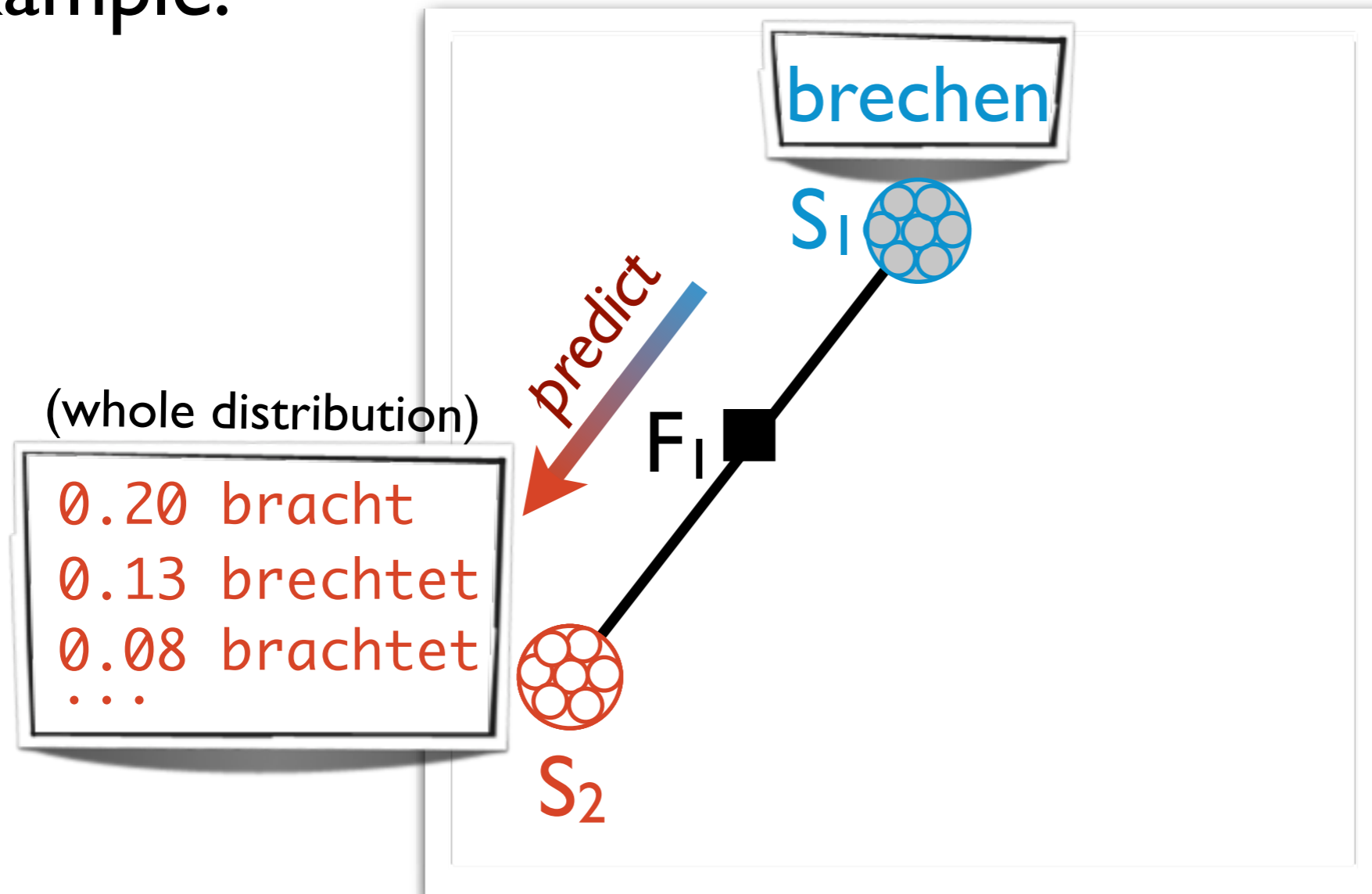
Example:



# 2

# Inference. *Multiple strings*

Example:

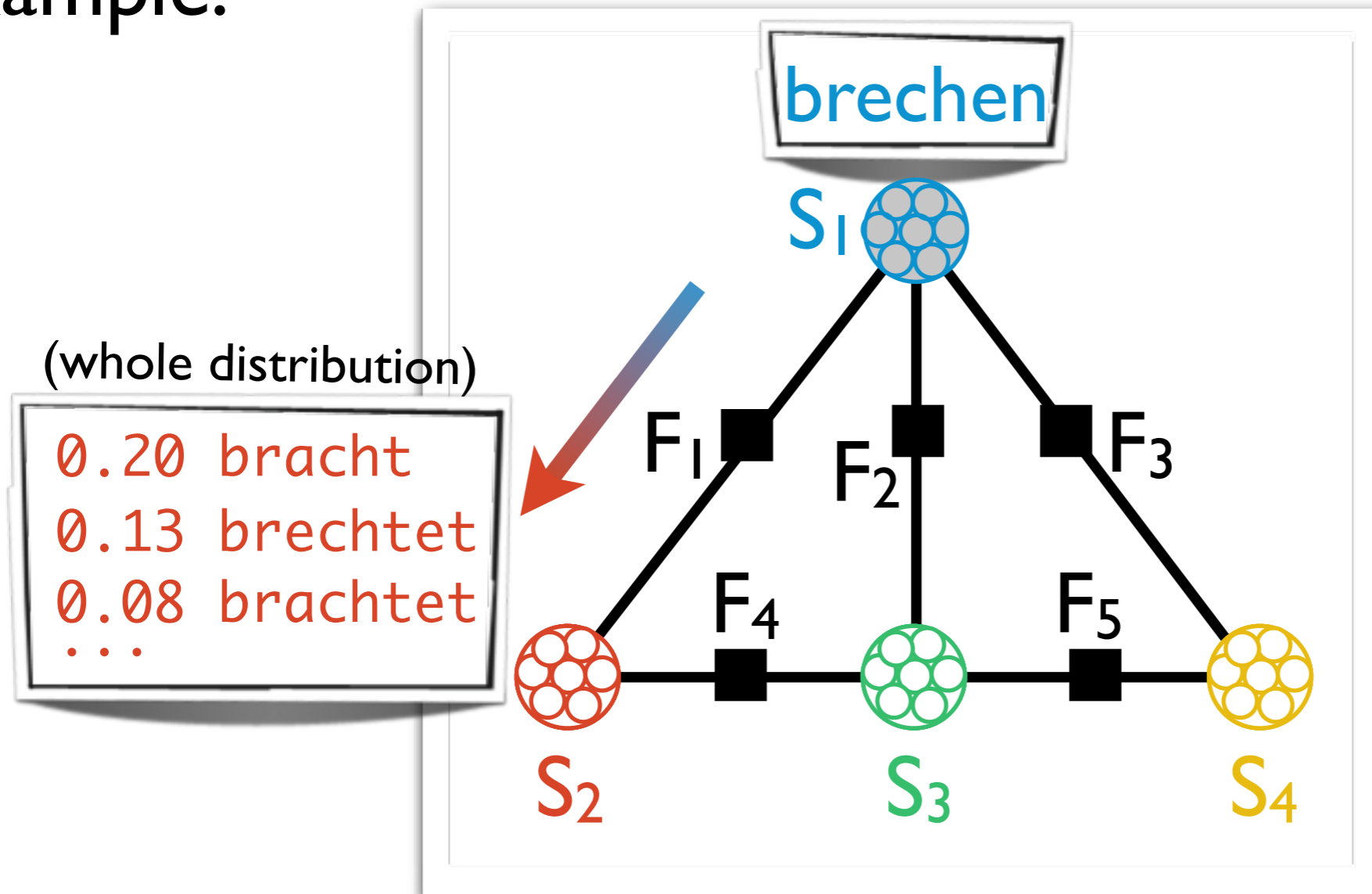




# 2

# Inference. *Multiple strings*

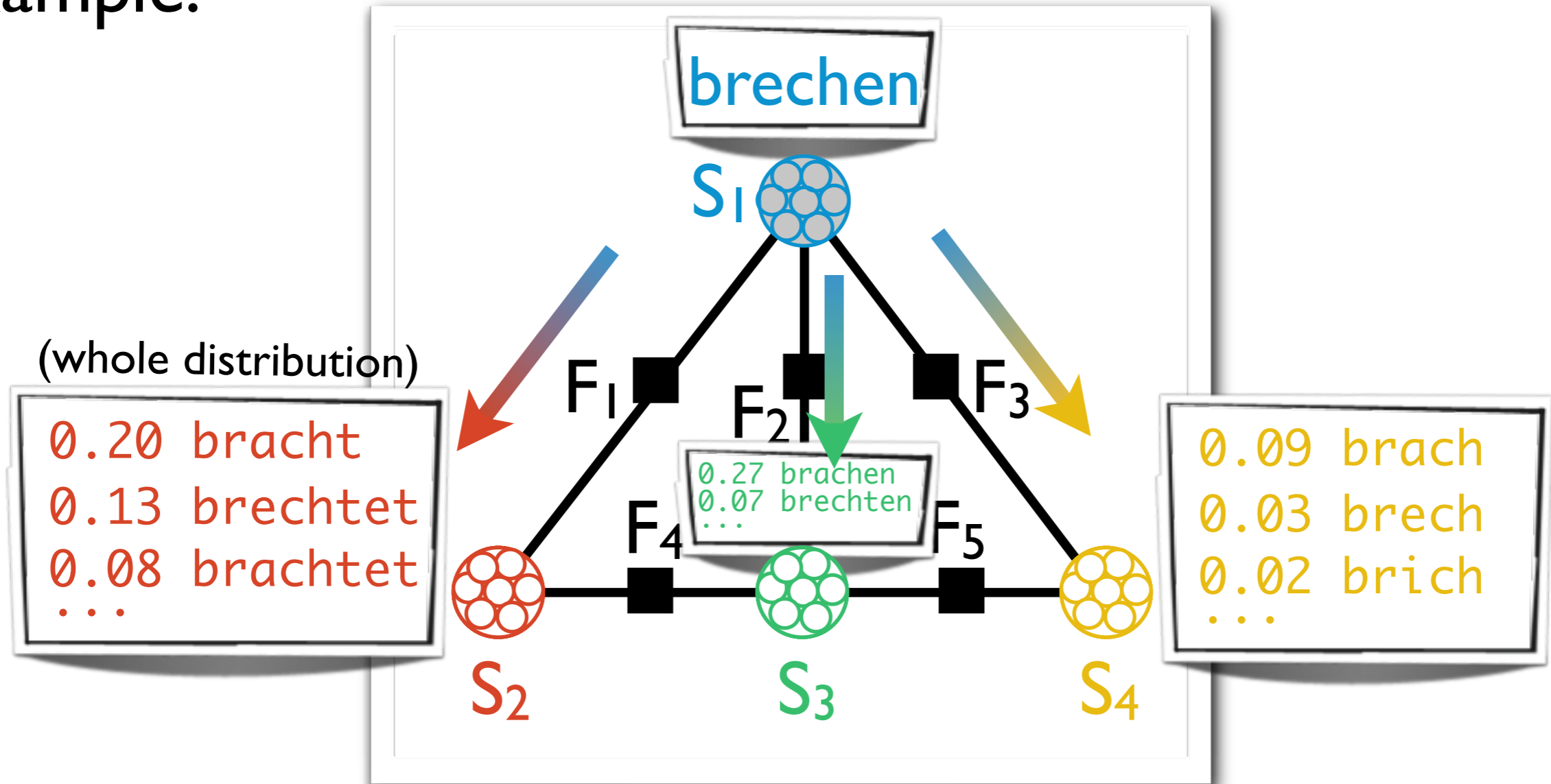
Example:



# 2

# Inference. *Multiple strings*

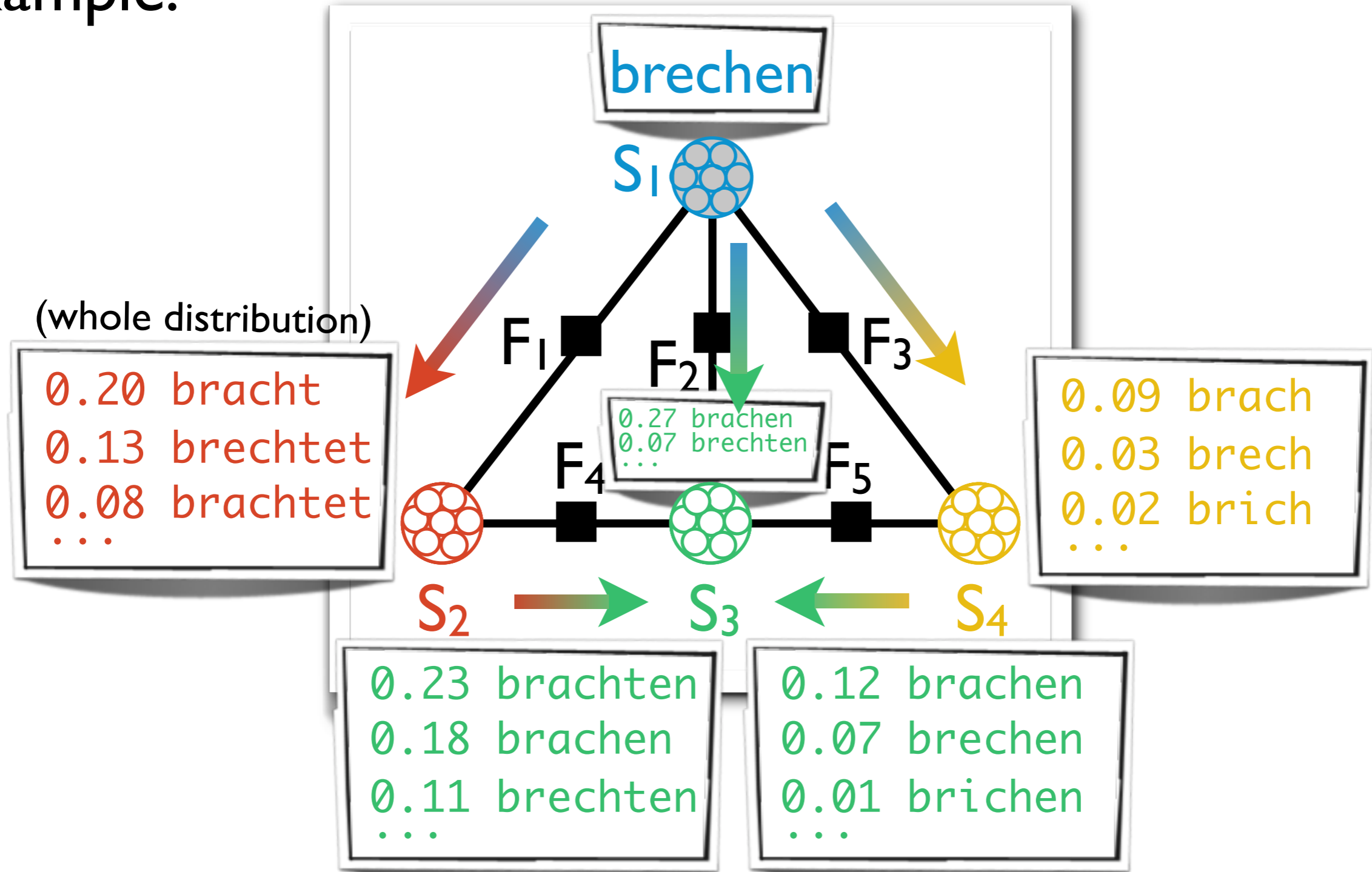
Example:



# 2

# Inference. *Multiple strings*

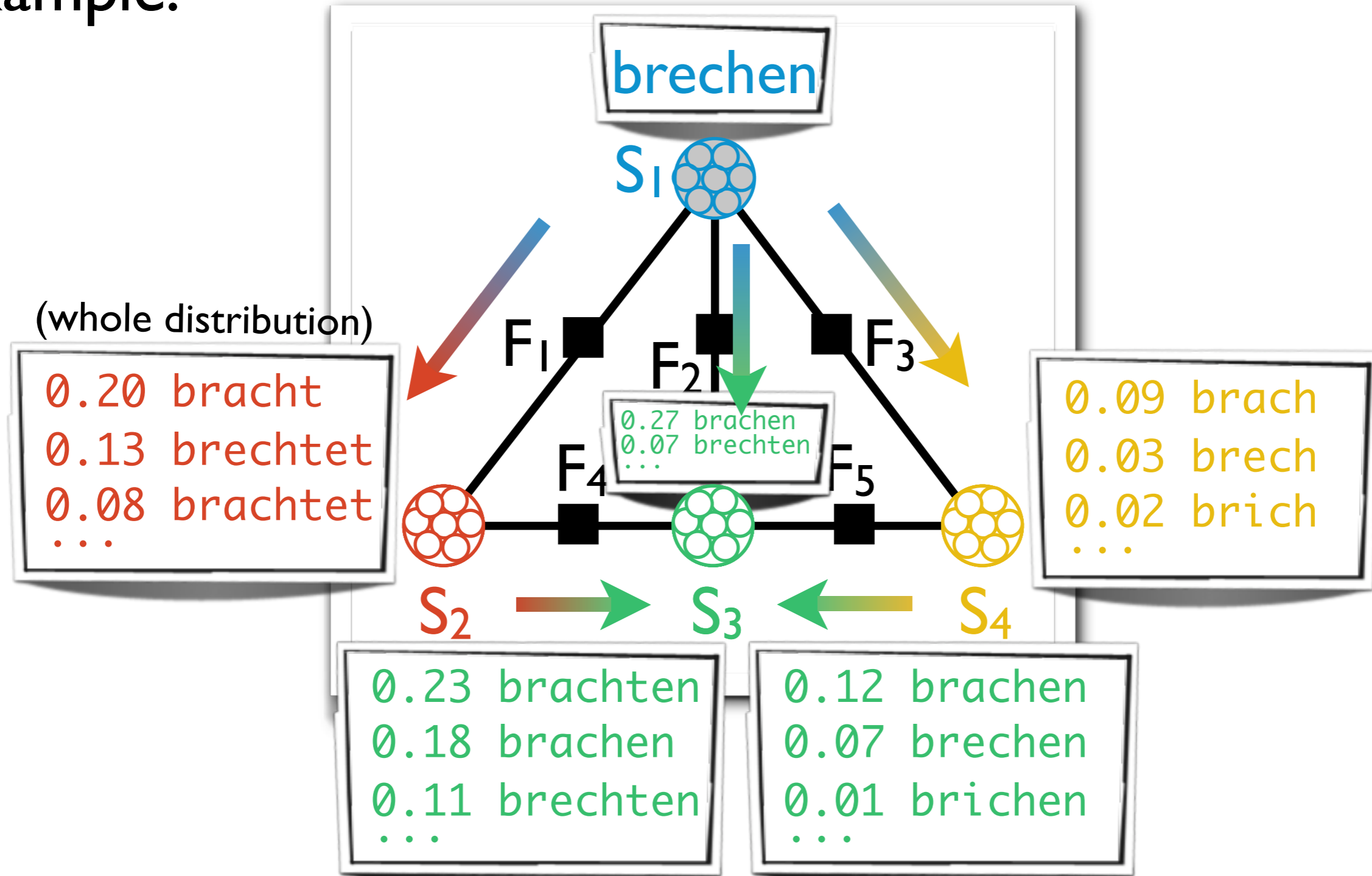
Example:



# 2

# Inference. *Multiple strings*

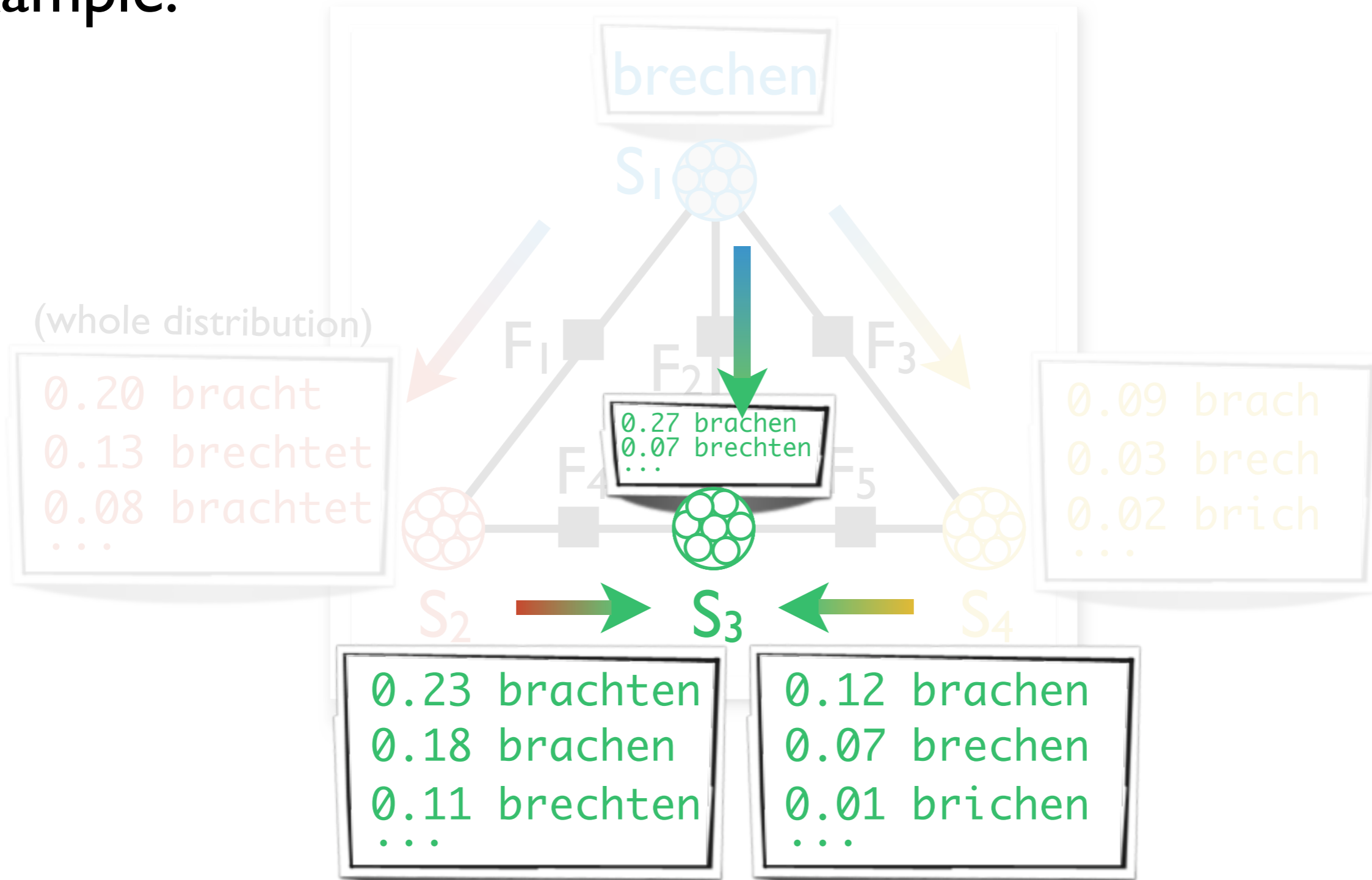
Example:



# 2

# Inference. *Multiple strings*

Example:

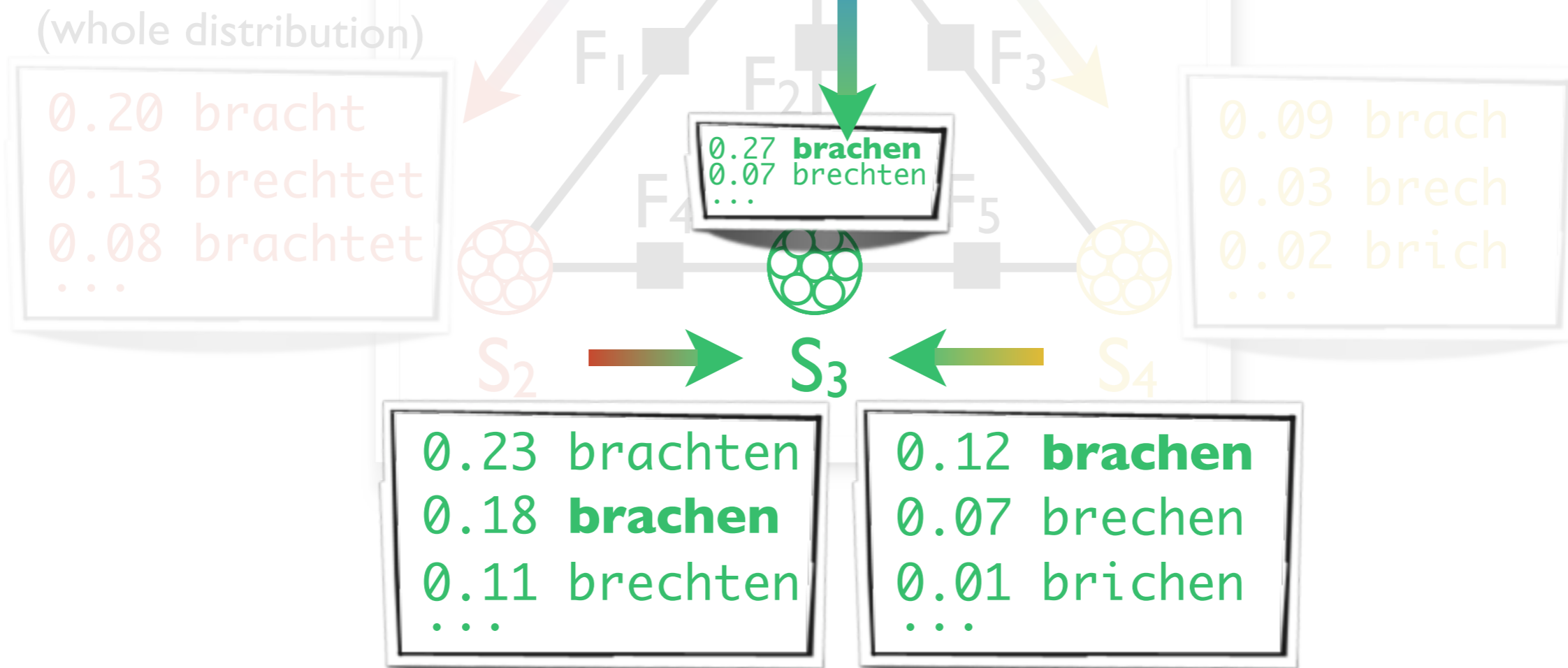


# 2

# Inference. *Multiple strings*

Example:

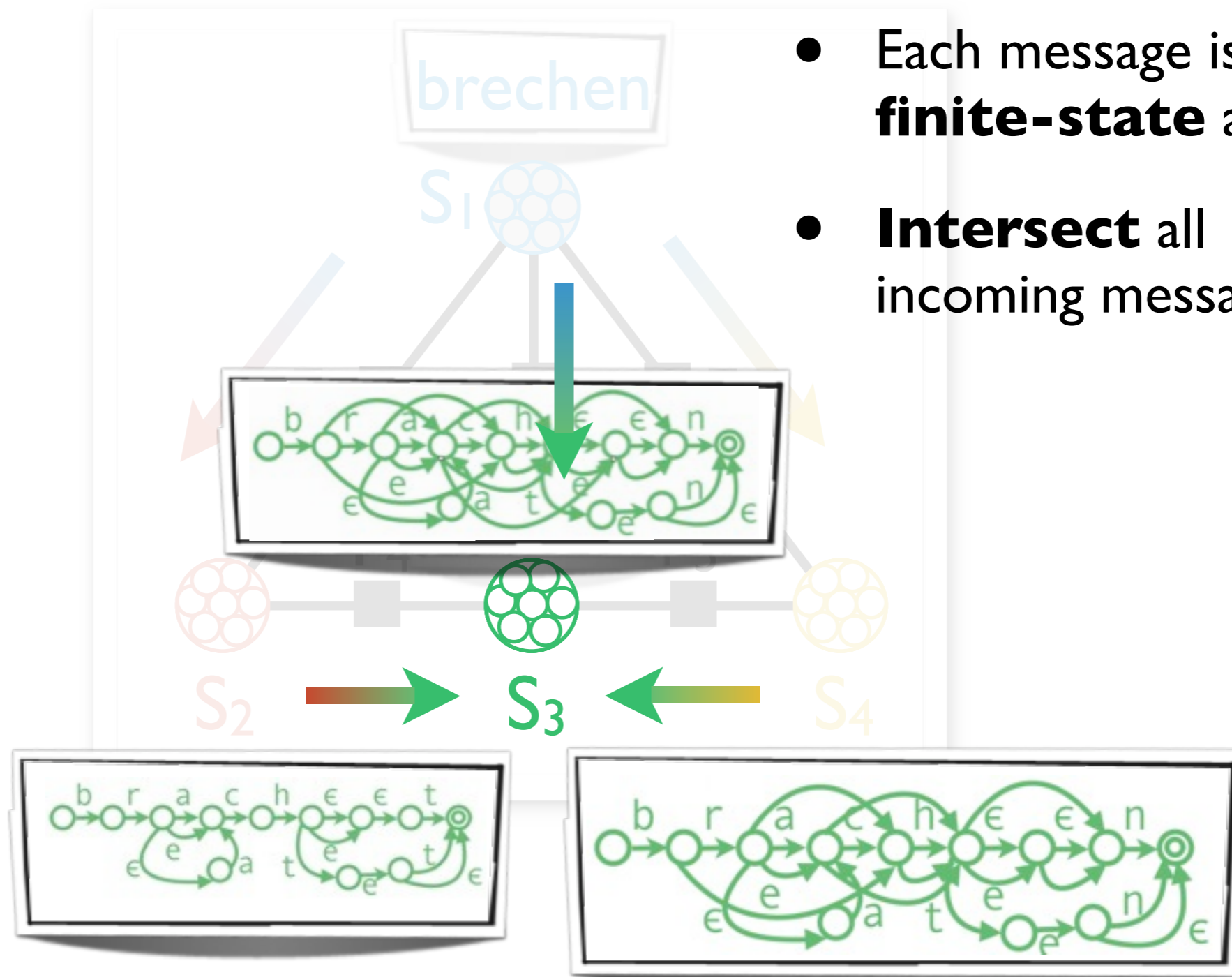
Decoding output for  $S_3$  (consensus): **brachen**



# 2

# Inference. *Multiple strings*

Example:

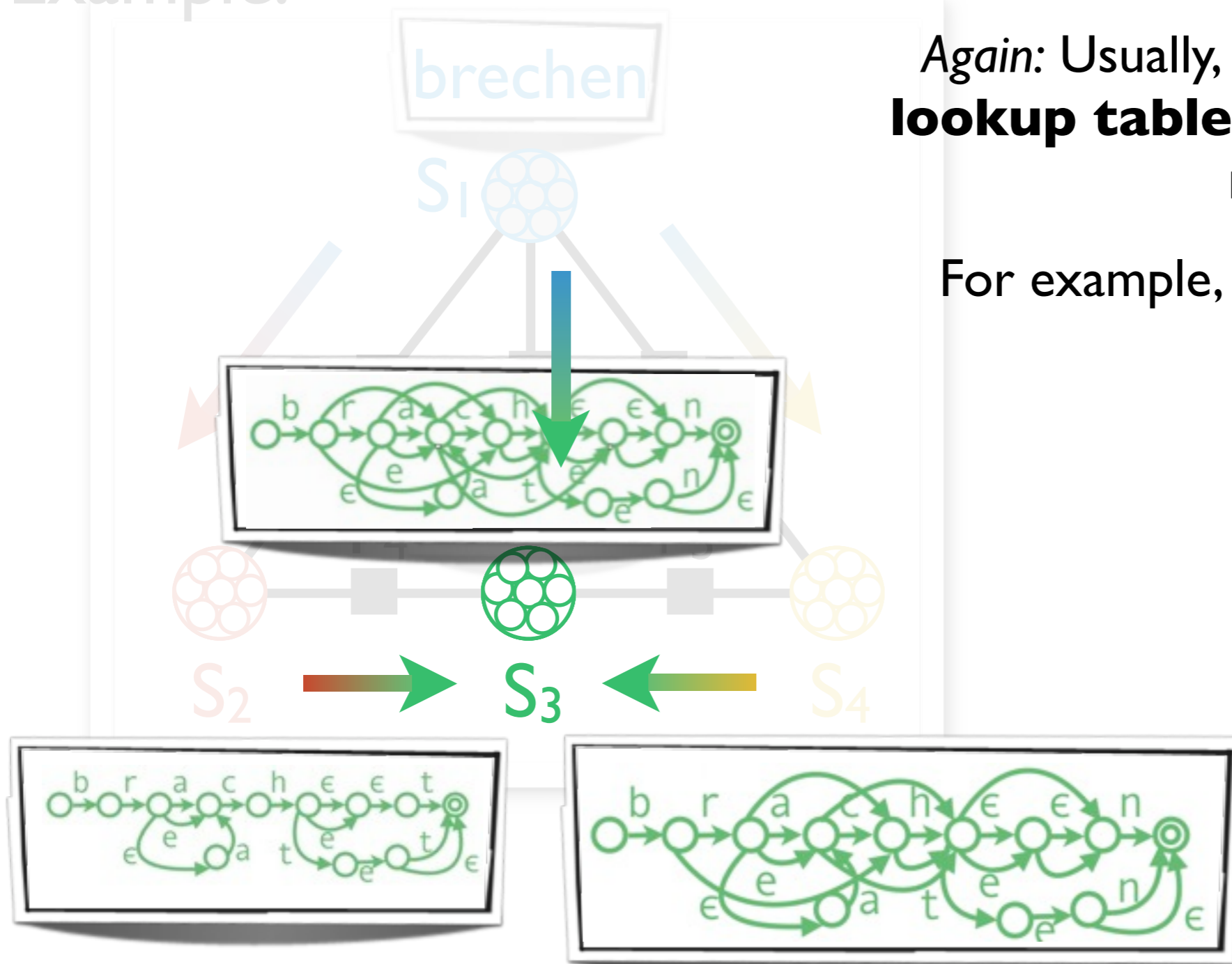


- Each message is a **finite-state** acceptor
- **Intersect** all incoming messages

# 2

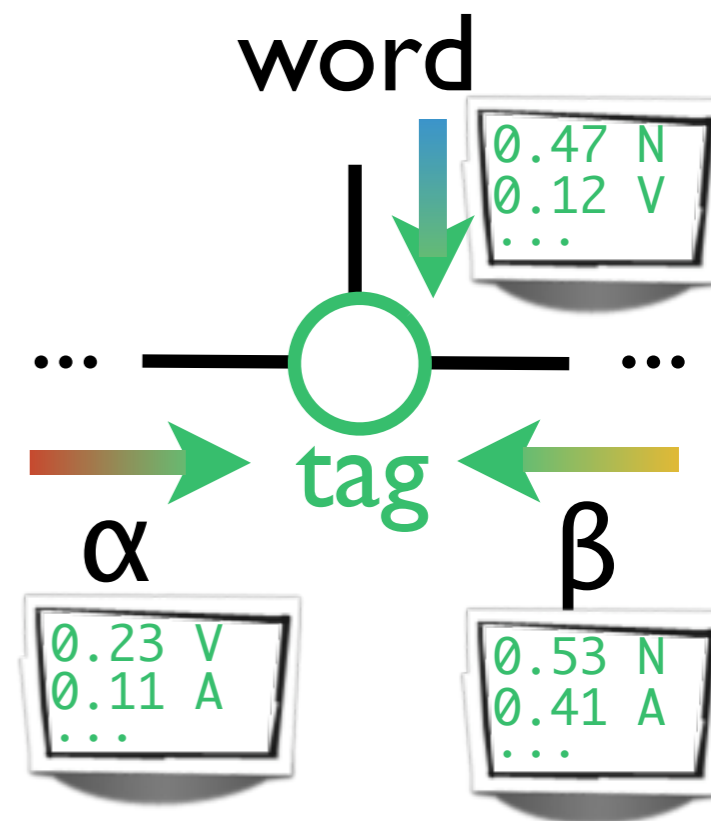
# Inference. *Multiple strings*

Example:



Again: Usually, BP just works with simple **lookup tables** as factors and messages, **not** finite-state machines.

For example, message passing in a CRF:

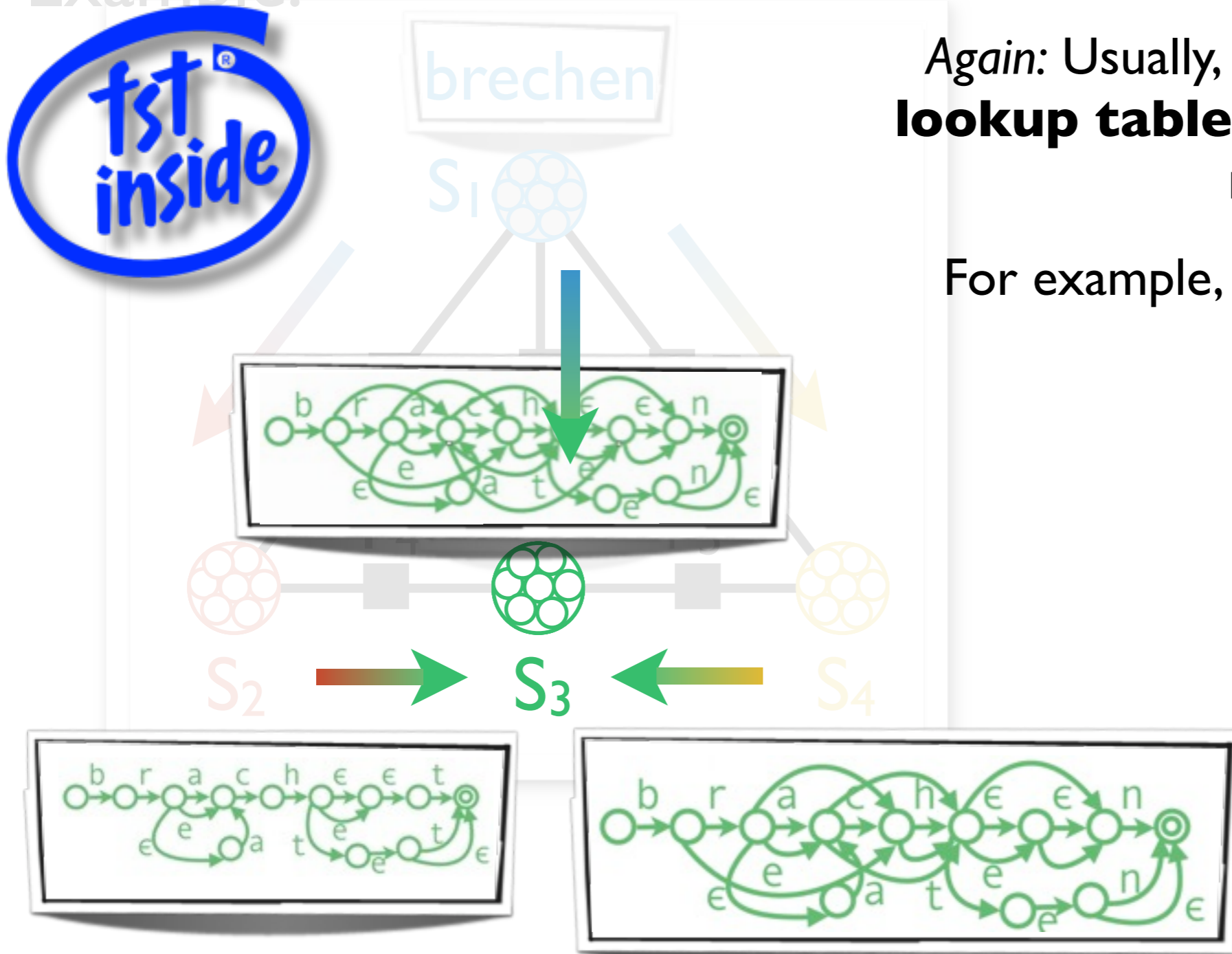




# 2

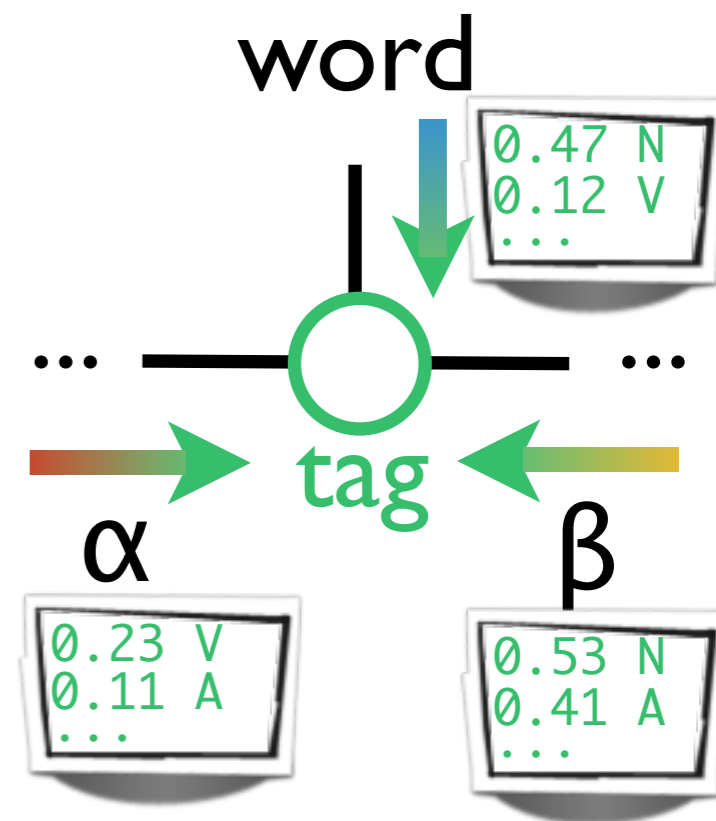
# Inference. *Multiple strings*

Example:



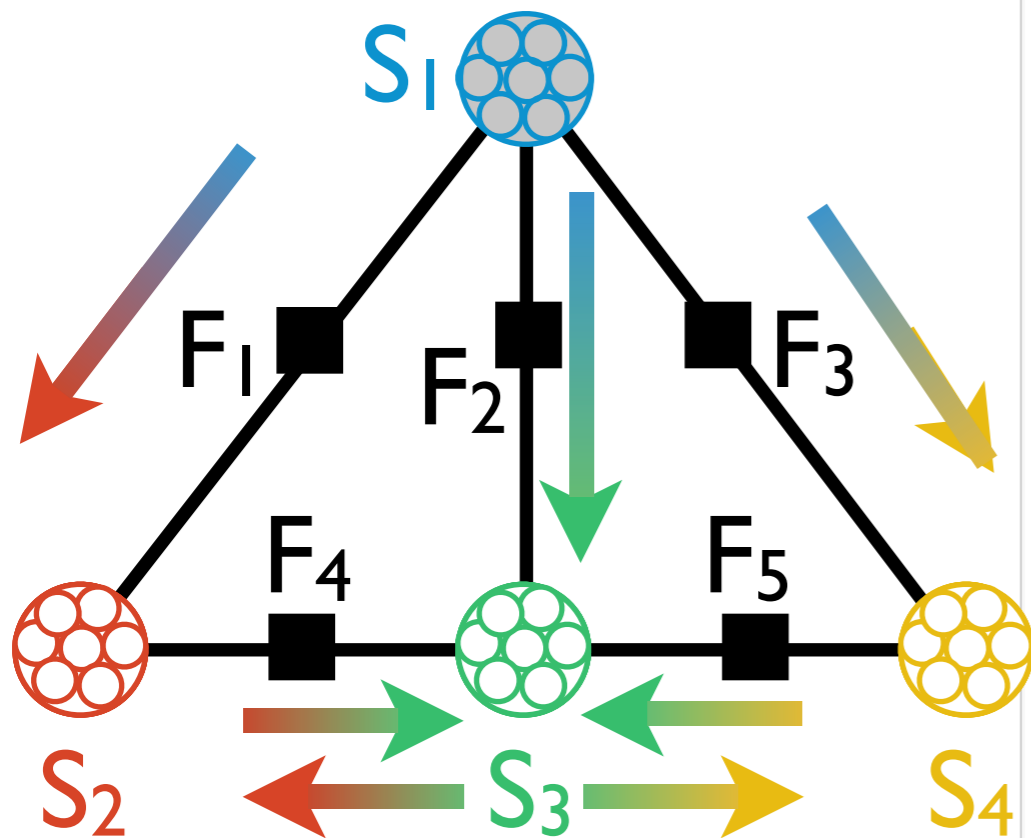
Again: Usually, BP just works with simple **lookup tables** as factors and messages, **not** finite-state machines.

For example, message passing in a CRF:



## 2

Inference. *Multiple strings*

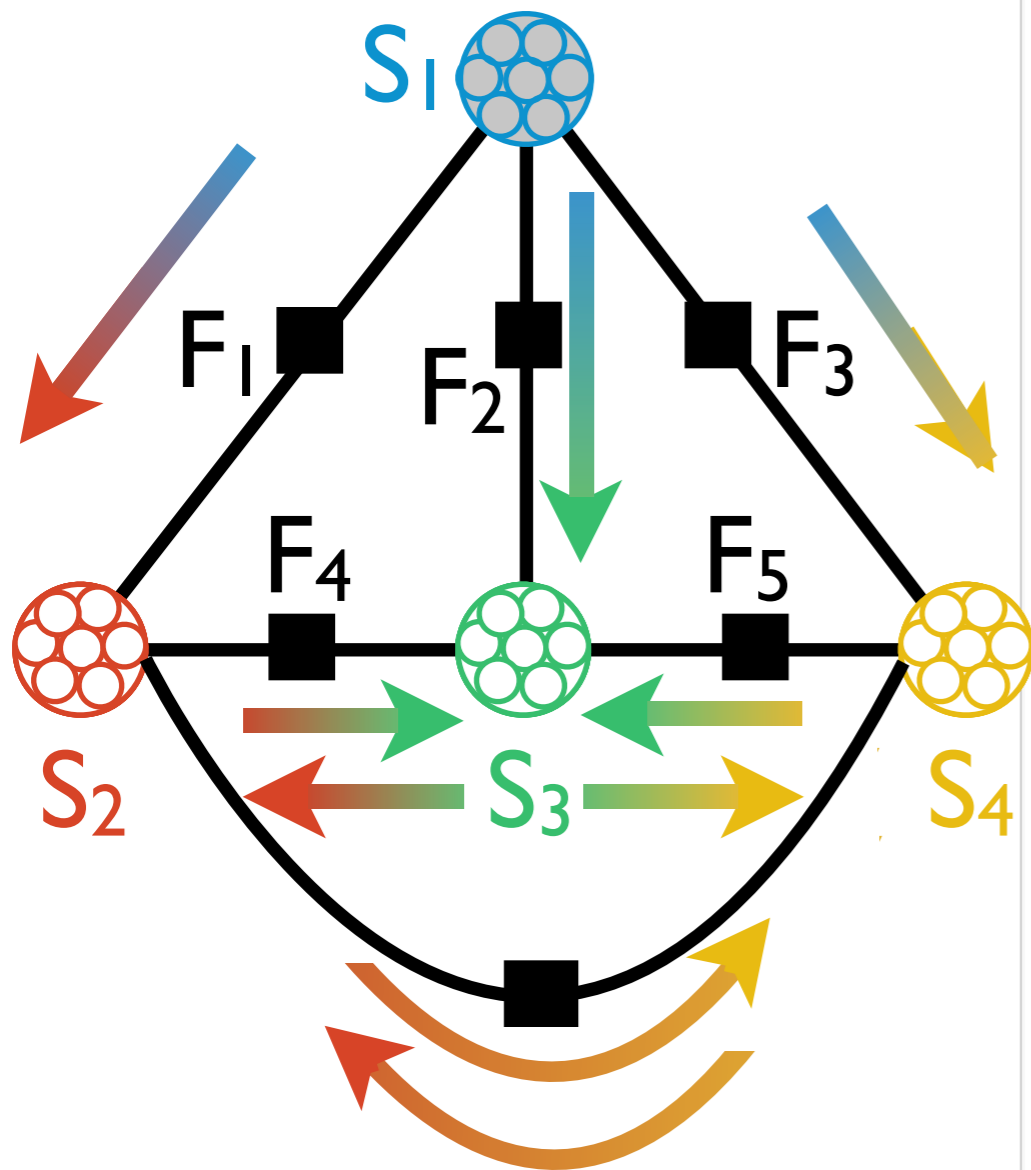
$$S_1 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{e} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{e} \text{O} \xrightarrow{n} \text{O} \quad (\text{observed})$$


- We can also run **loopy belief propagation** on these finite-state Markov Random Fields (MRFs)
- Just **iterate** the message passing
- Issues with **intractability**, see my thesis

2

# Inference. *Multiple strings*

$S_1 = \text{O} \xrightarrow{b} \text{O} \xrightarrow{r} \text{O} \xrightarrow{e} \text{O} \xrightarrow{c} \text{O} \xrightarrow{h} \text{O} \xrightarrow{e} \text{O} \xrightarrow{n} \text{O} \quad (\text{observed})$



- We can also run **loopy belief propagation** on these finite-state Markov Random Fields (MRFs)
- Just **iterate** the message passing
- Issues with **intractability**, see my thesis

# 2

# Inference. *Multiple strings*

- Joint inference can be used to train these models from data
- Training data consists of complete or incomplete tables of forms
- We present a method to induce factor graphs in a data-driven way
- See my thesis for the approach and results

## 2 Conclusions / Contributions

- Presented general, novel **joint probability model** over multiple strings
- Combines **NLP techniques** (*finite-state machines*) with **machine-learning techniques** (*graphical models*)
- Markov Random Field over strings (variables: *string-valued*, **potential functions: finite-state machines**)

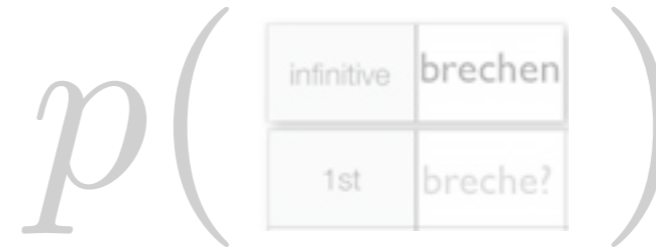
## 2 Conclusions / Contributions

- Novel variant of belief propagation with **finite-state messages**
- Presented **approximations**
- Presented novel way of **structure induction** for string-based models based on edit distance
- Achieved significant improvements through **staged joint training** of complex factor graphs

# Overview

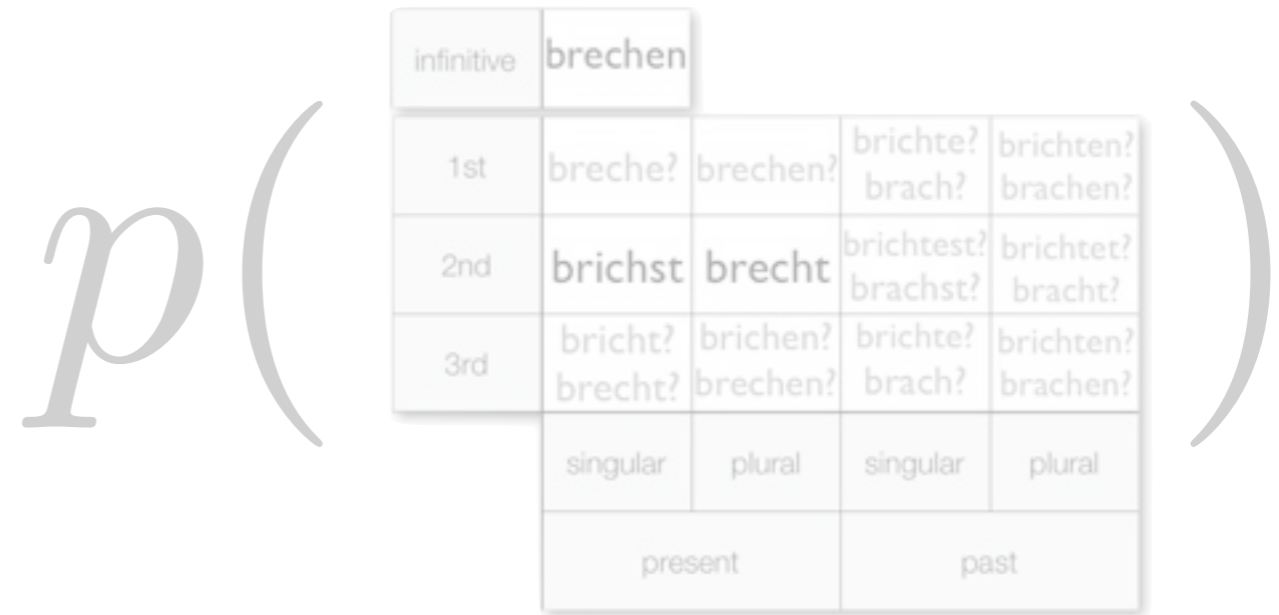
1

String pairs



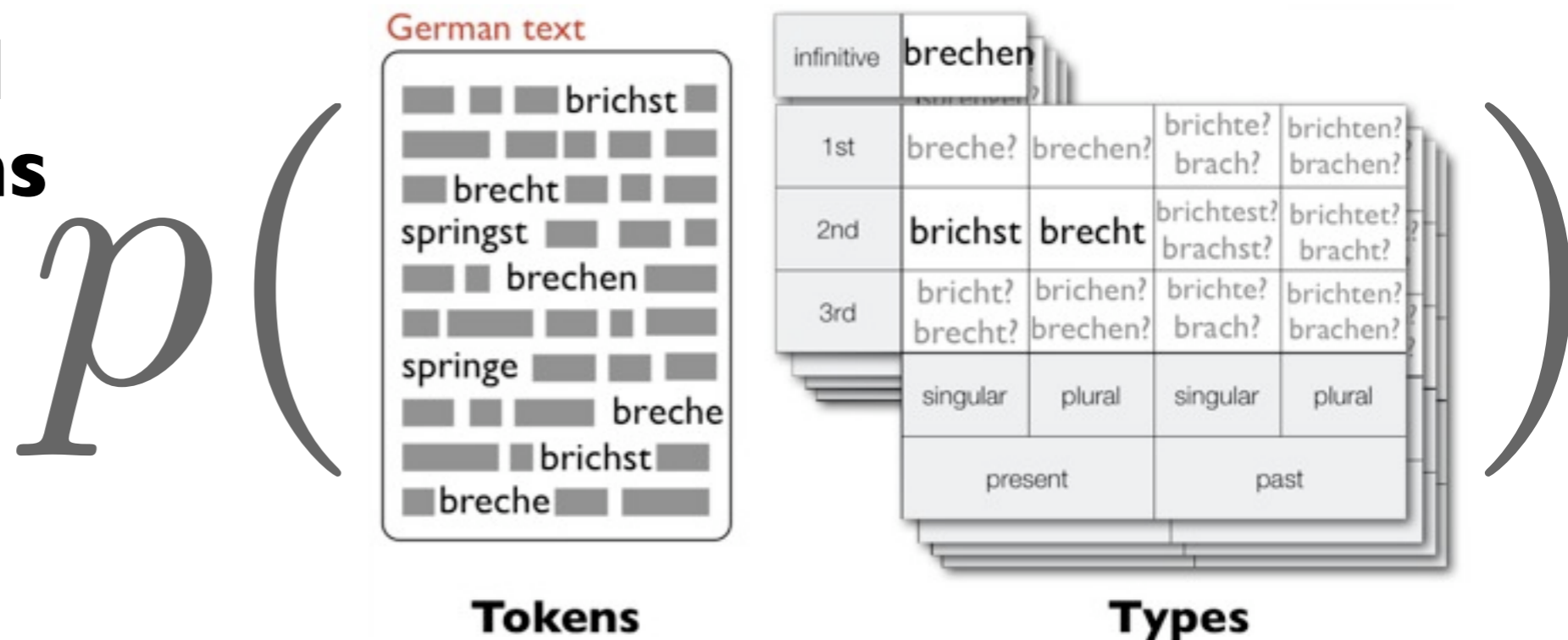
2

Multiple strings  
(paradigms)



3

**Text and  
paradigms**

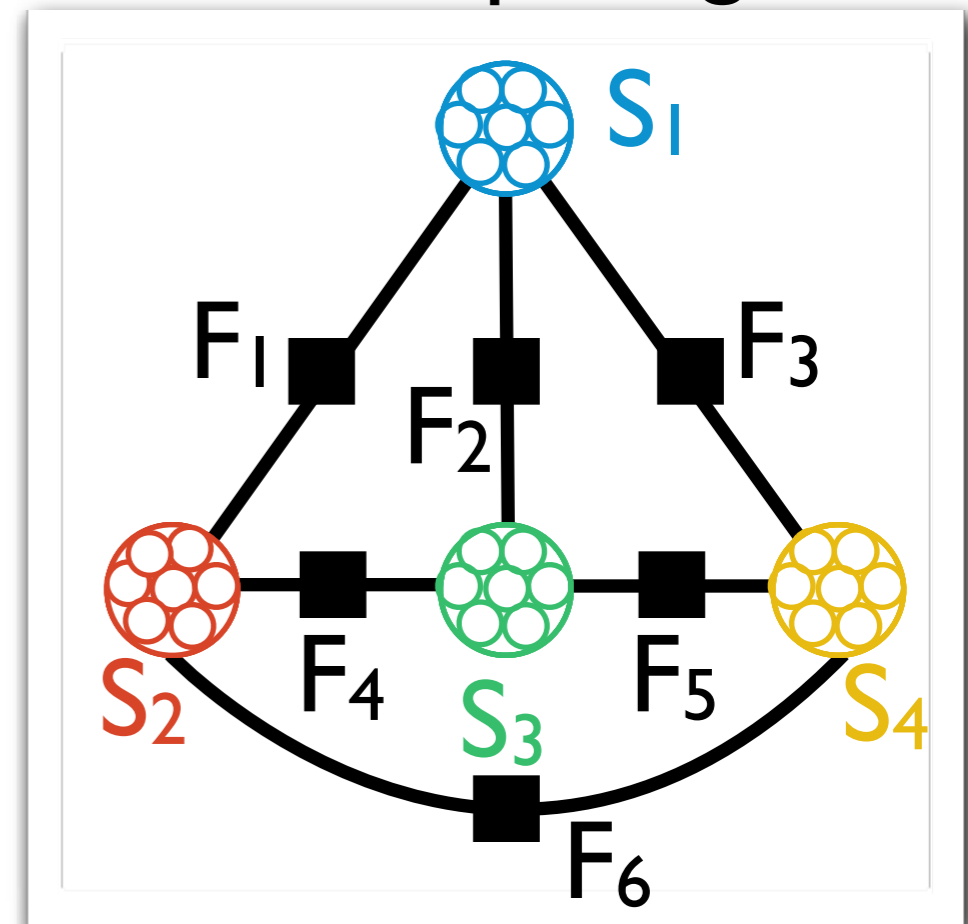


## 3

## Text &amp; Paradigms

- We have seen how an **inflectional paradigm** (“multiple strings”) can be modeled by **finite-state Markov Random Fields (MRFs)**
- Now we will build a joint model over **inflectional paradigms** and a **text corpus**
- **Goal:** Learn how to inflect words in the language, using clues from the text corpus

4 systematically related spellings:



$$\Pr(s_1, s_2, s_3, s_4)$$



**Why** do we want to use a text corpus?

- In **Part 2:**

We learn how to inflect words, given some observed paradigms (complete or incomplete) that someone created as training data (*expensive supervision*) ☹️

- Here in **Part 3:**

We *also* want to learn how to inflect words, we'll use a few observed paradigms as well, but mainly learn from plain text (*cheap data*) 😊

## **How can a text corpus help?**

It can potentially **fix** erroneous MRF string predictions.

Intuition:

If a spelling predicted by the MRF **cannot be found in the corpus**, it was probably an incorrect prediction.

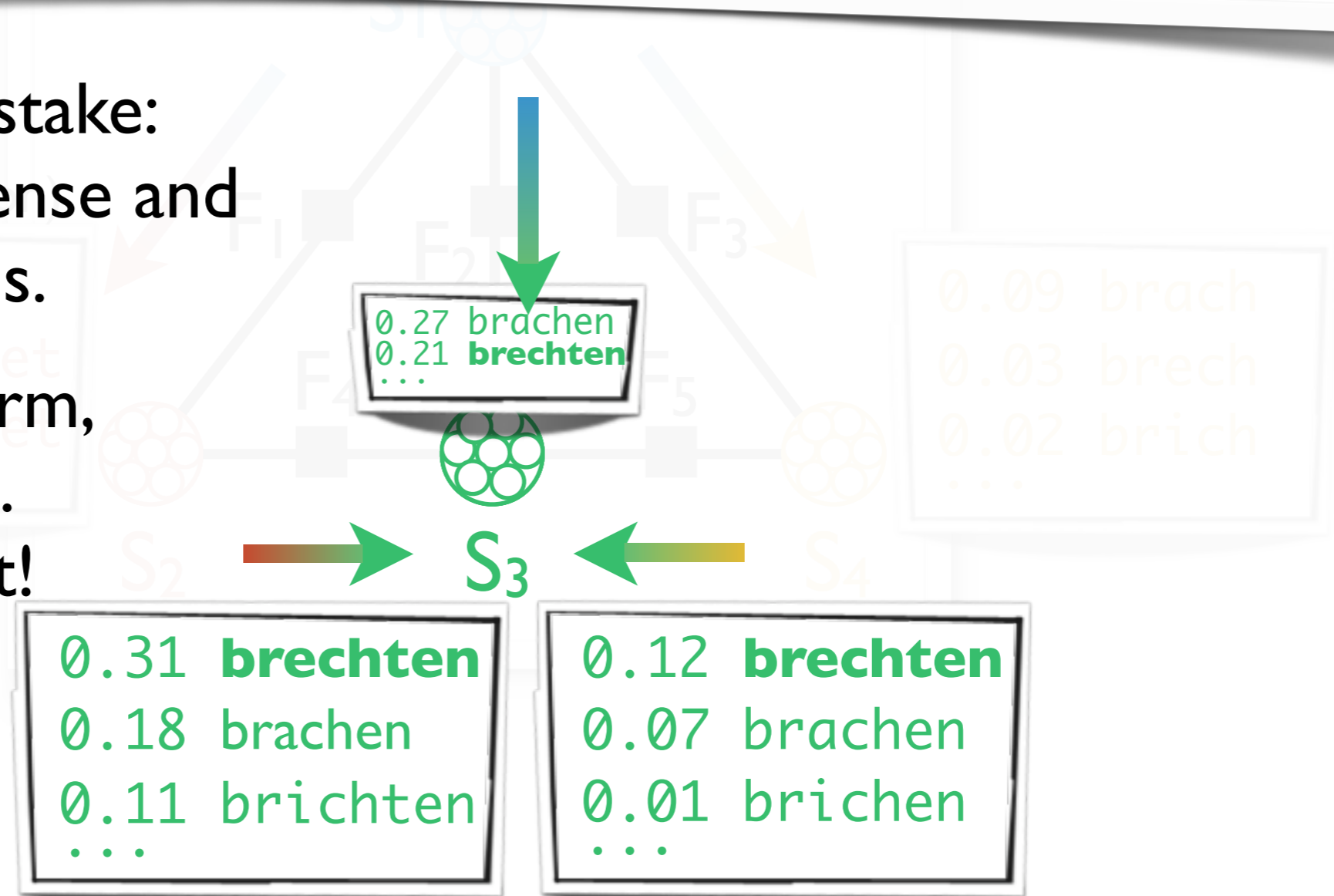
# Text & Paradigms

Decoding output for  $S_3$  (consensus): **brechten**

MRF is making a mistake:  
**brechten** is nonsense and  
**not** found in corpus.

But the 2nd-best form,  
**brachen**, is frequent.

It's probably correct!



We will make such decisions  
using **statistical inference**,  
under a **probability model** that uses the finite-  
state MRFs, but models the text corpus as well.

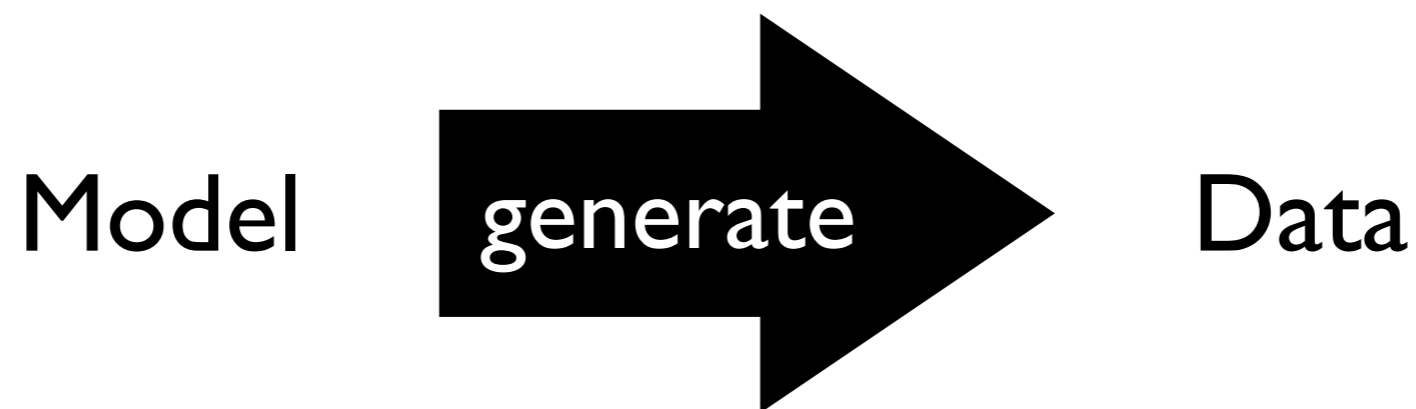
What kind of probability model do we want?

- Keep **tokens** and **types** separate
- Tokens are in the **text corpus**
- Types are in the **paradigms**
- We also model **abstract morphological knowledge** about how forms are related

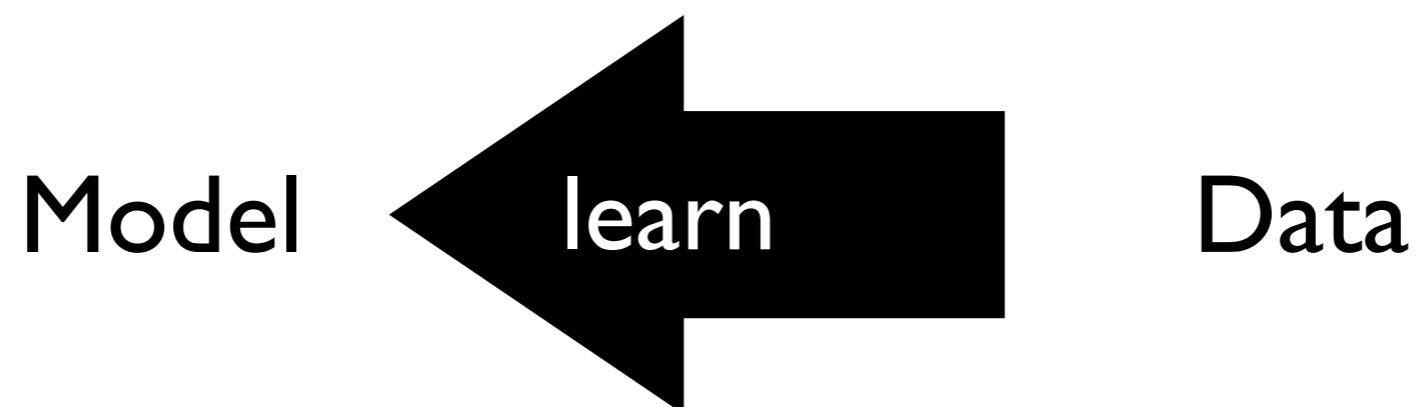
- Each paradigm contains the systematically related spellings of a **lexeme** (modeled by finite-state MRFs).
- Assume an **unbounded** number of possible lexemes and paradigms (“non-parametric”) in the text.

# Text & Paradigms

## **Generative story**



## **Inference (Sampling)**



3

# Text & Paradigms

## **Generative story**

Model

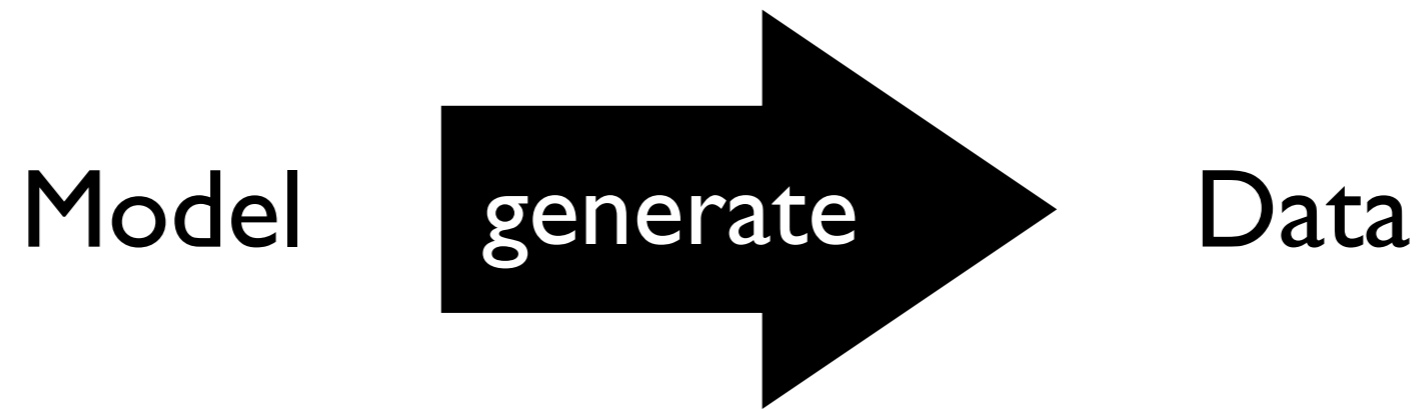


generate

Data



## Generative story



To generate from our model:

First, generate the **types** of the language.

Then, use them to generate the corpus **tokens**.

# 3

# Text & Paradigms

- (1) Generate infinitely many lexemes

# 3

# Text & Paradigms

(1) Generate infinitely many lexemes

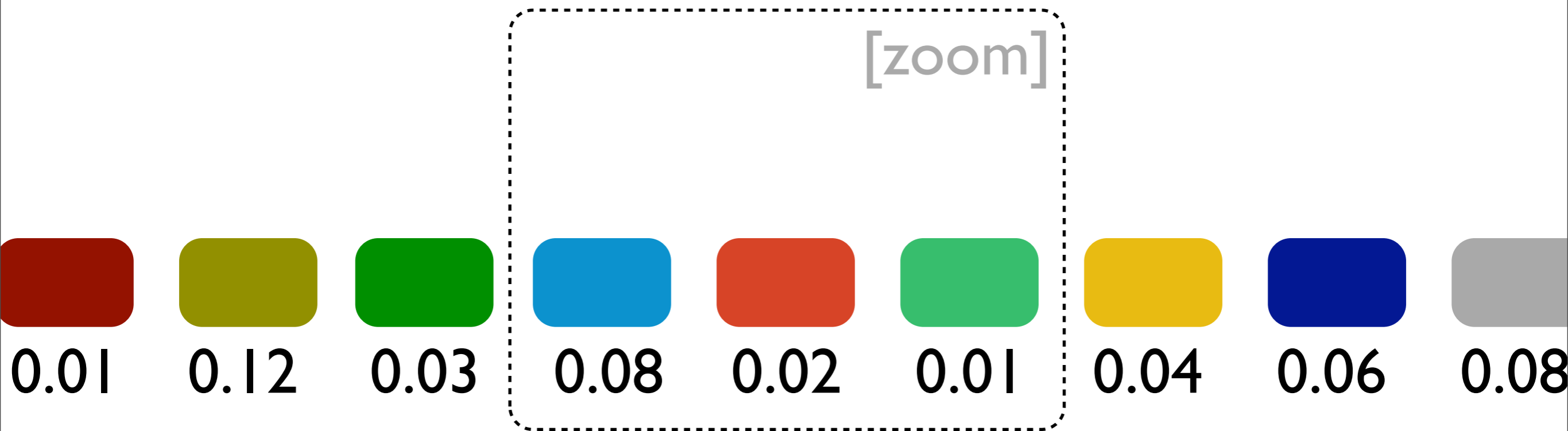


Stick-breaking process

# 3

# Text & Paradigms

(1) Generate infinitely many lexemes

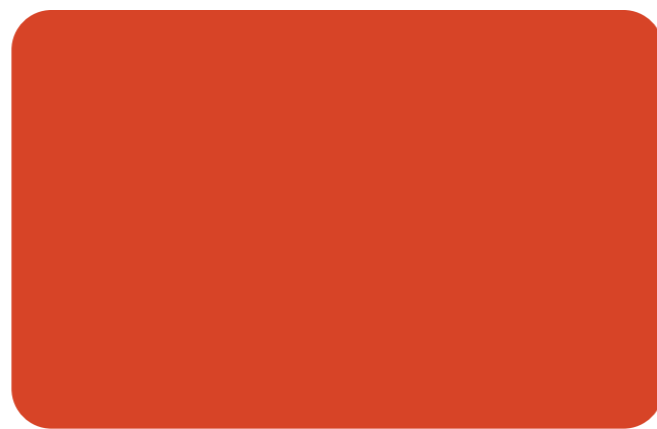


Stick-breaking process

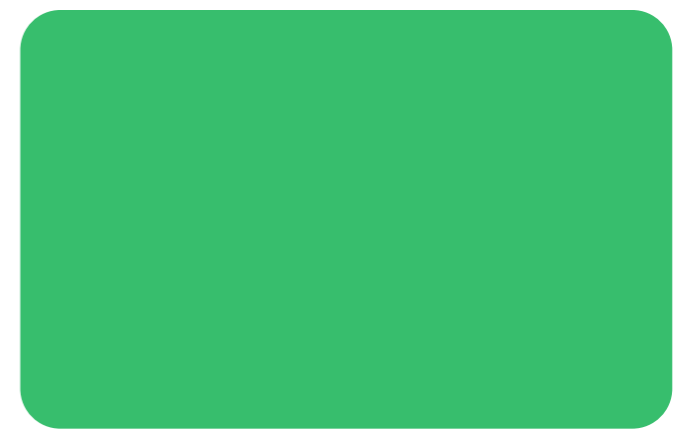
(1) Generate infinitely many lexemes



0.08



0.02



0.01

3

# Text & Paradigms

(1) Generate infinitely many lexemes



0.08



0.02



0.01

## 3

# Text & Paradigms

- (2) Each lexeme has a paradigm with slots for the different inflections

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

0.08



0.02



0.01

## 3

## Text &amp; Paradigms

- (2) Each lexeme has a paradigm with slots for the different inflections

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

0.08

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

0.02

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

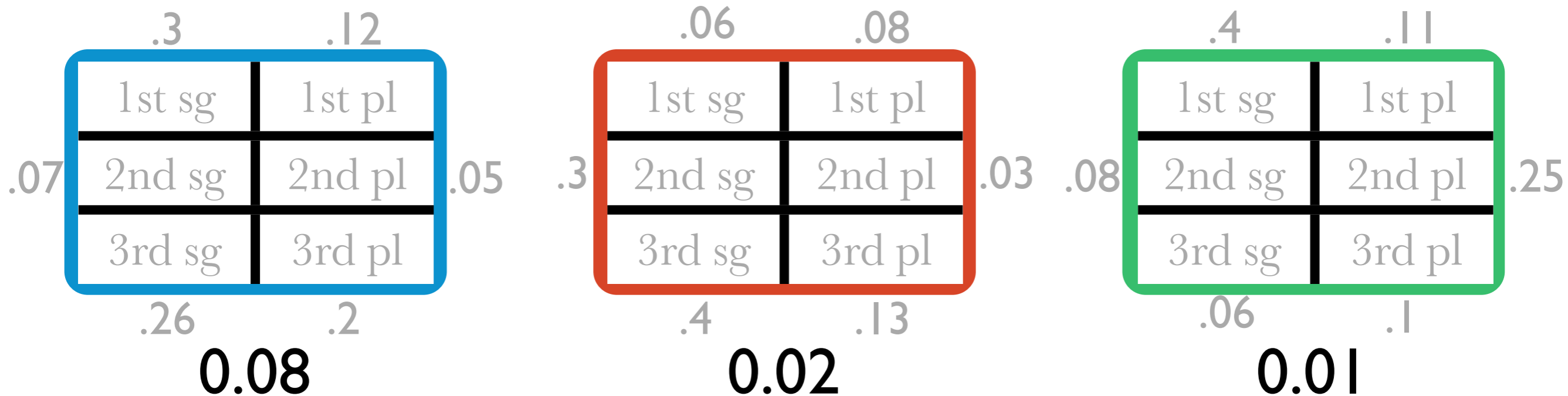
0.01



# 3

# Text & Paradigms

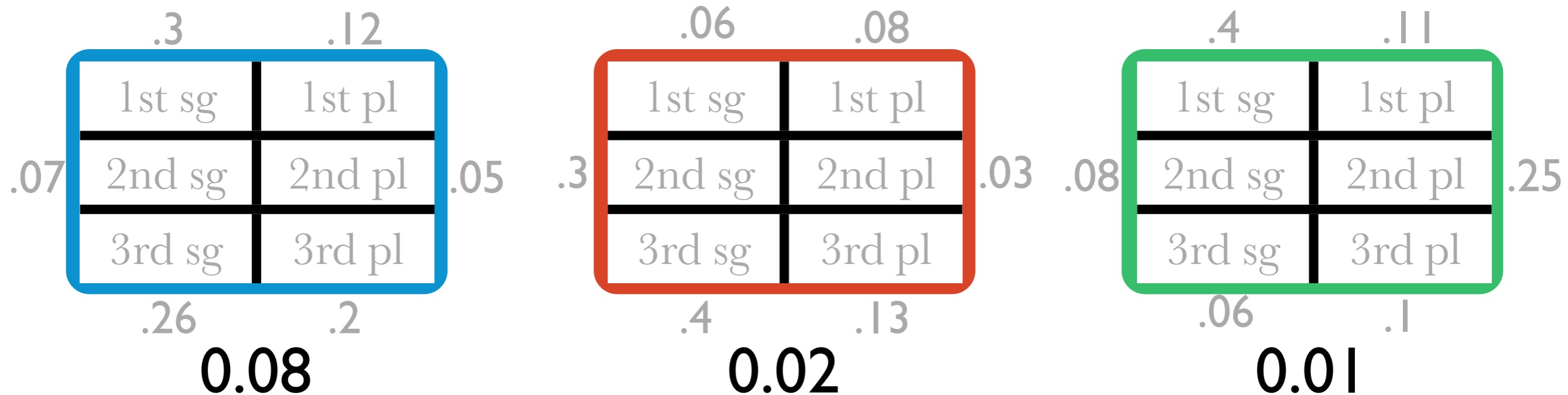
(3) Each paradigm has a distribution over slot frequencies



## 3

## Text &amp; Paradigms

- (4) All paradigms generate their spellings using the same finite-state MRF parameterized by  $\theta$



## 3

## Text &amp; Paradigms

- (4) All paradigms generate their spellings using the same finite-state MRF parameterized by  $\theta$

	.3	.12	
	breche	brechen	
.07	brichst	brecht	.05
	bricht	brechen	
	.26	.2	
	0.08		

	.06	.08	
	1st sg	1st pl	
.3	2nd sg	2nd pl	.03
	3rd sg	3rd pl	
	.4	.13	
	0.02		

	.4	.11	
	1st sg	1st pl	
.08	2nd sg	2nd pl	.25
	3rd sg	3rd pl	
	.06	.1	
	0.01		

## 3

## Text &amp; Paradigms

- (4) All paradigms generate their spellings using the finite-state MRF parameterized by  $\theta$

	.3	.12	
	breche	brechen	
.07	brichst	brecht	.05
	bricht	brechen	
	.26	.2	
	0.08		

	.06	.08	
	treffe	treffen	
.3	triffst	trefft	.03
	trifft	treffen	
	.4	.13	
	0.02		

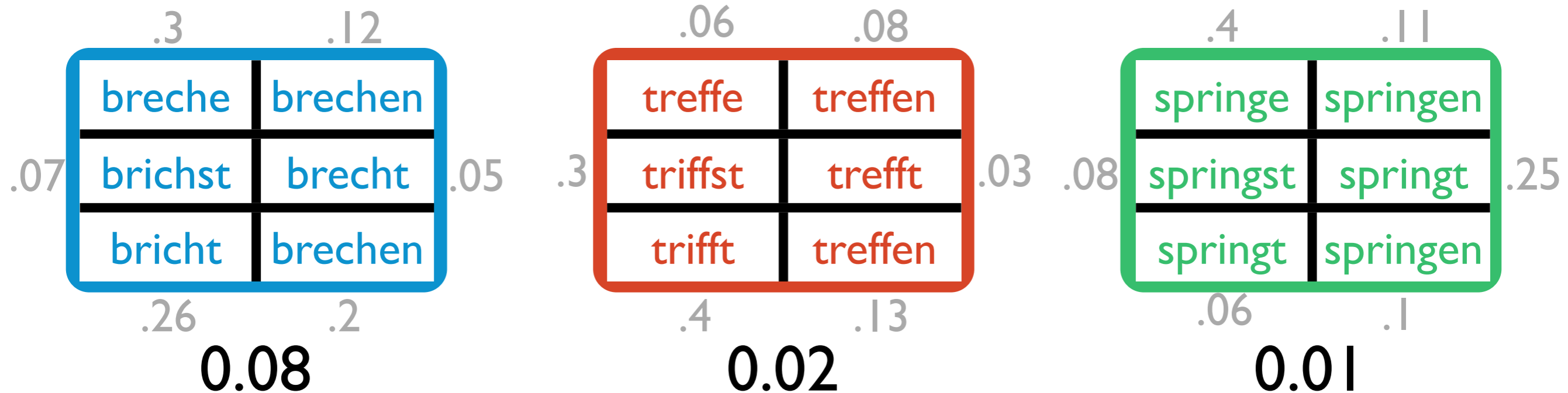
  

	.4	.11	
	1st sg	1st pl	
.08	2nd sg	2nd pl	.25
	3rd sg	3rd pl	
	.06	.1	
	0.01		

# 3

# Text & Paradigms

(4) All paradigms generate their spellings using the finite-state MRF parameterized by  $\theta$



# Text & Paradigms

- (4) All paradigms generate their spellings using the finite-state MRF parameterized by  $\theta$

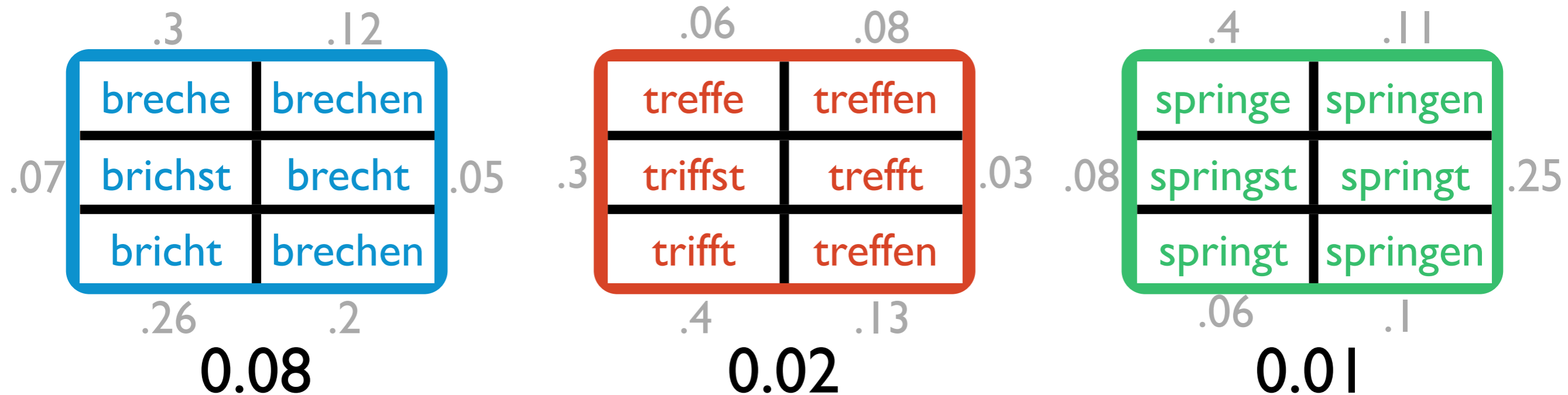
$\theta$  is a “morphological grammar”  
 The true  $\theta$  for German morphology will know, for example,  
 “from *1st singular* to *plural*, just append the **suffix** *n*”, or  
 “between *3rd sg* and *pl*, a **vowel change** is likely”

	.3	.12			.06	.08		.4	.11	
	breche	brechen			treffe	treffen		springe	springen	
.07	brichst	brecht	.05	.3	triffst	trefft	.03	springst	springt	.25
	bricht	brechen			trifft	treffen		springt	springen	
	.26	.2		.4	.13		.06	.1		
	0.08			0.02			0.01			

# 3

# Text & Paradigms

All types of the language have been generated.  
Now generate the corpus tokens.



# 3

# Text & Paradigms

	.3	.12	
	breche	brechen	
.07	brichst	brecht	.05
	bricht	brechen	
	.26	.2	
	<b>0.08</b>		

	.06	.08	
	treffe	treffen	
.3	triffst	trefft	.03
	trifft	treffen	
	.4	.13	
	<b>0.02</b>		

	.4	.11	
	springe	springen	
.08	springst	springt	.25
	springt	springen	
	.06	.1	
	<b>0.01</b>		

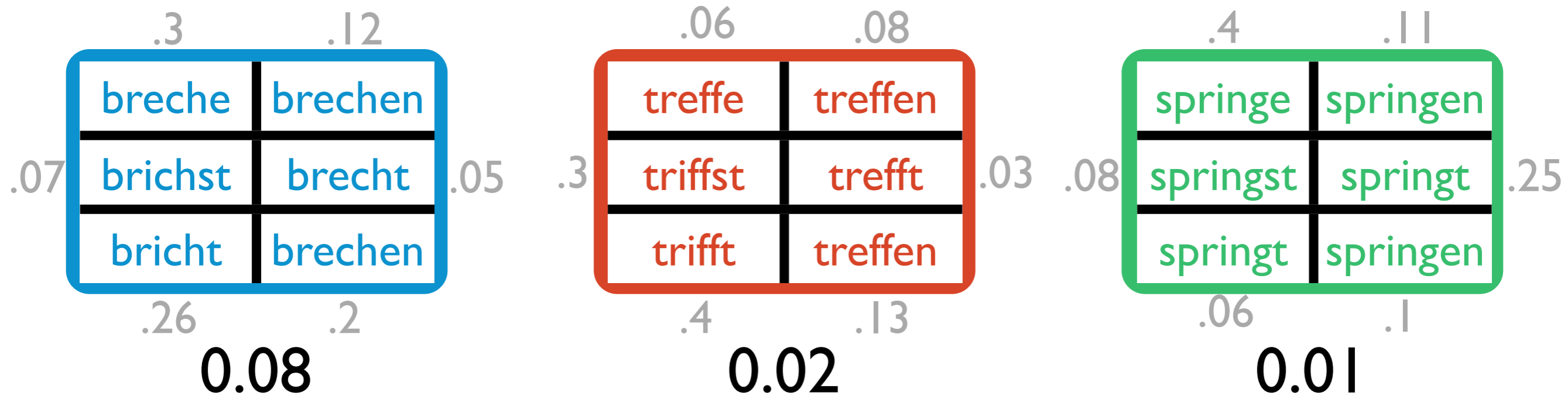


# 3

# Text & Paradigms

(5) Generate the corpus: *POS tags*

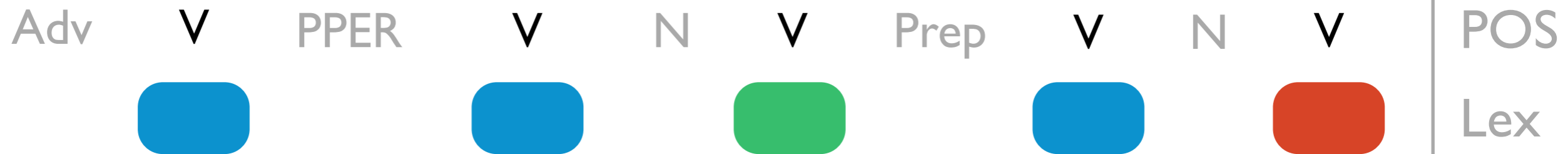
Adv    V    PPER    V    N    V    Prep    V    N    V



# 3

# Text & Paradigms

(5) Generate the corpus: *Lexemes*



		.3	.12			.06	.08			.4	.11		
		breche	brechen			treffe	treffen			springe	springen		
.07		brichst	brecht	.05		triffst	trefft	.03		springst	springt	.25	
		bricht	brechen			trifft	treffen			springt	springen		
		.26	.2			.4	.13			.06	.1		
		<b>0.08</b>				<b>0.02</b>				<b>0.01</b>			

# 3

# Text & Paradigms

(5) Generate the corpus: *Inflections*

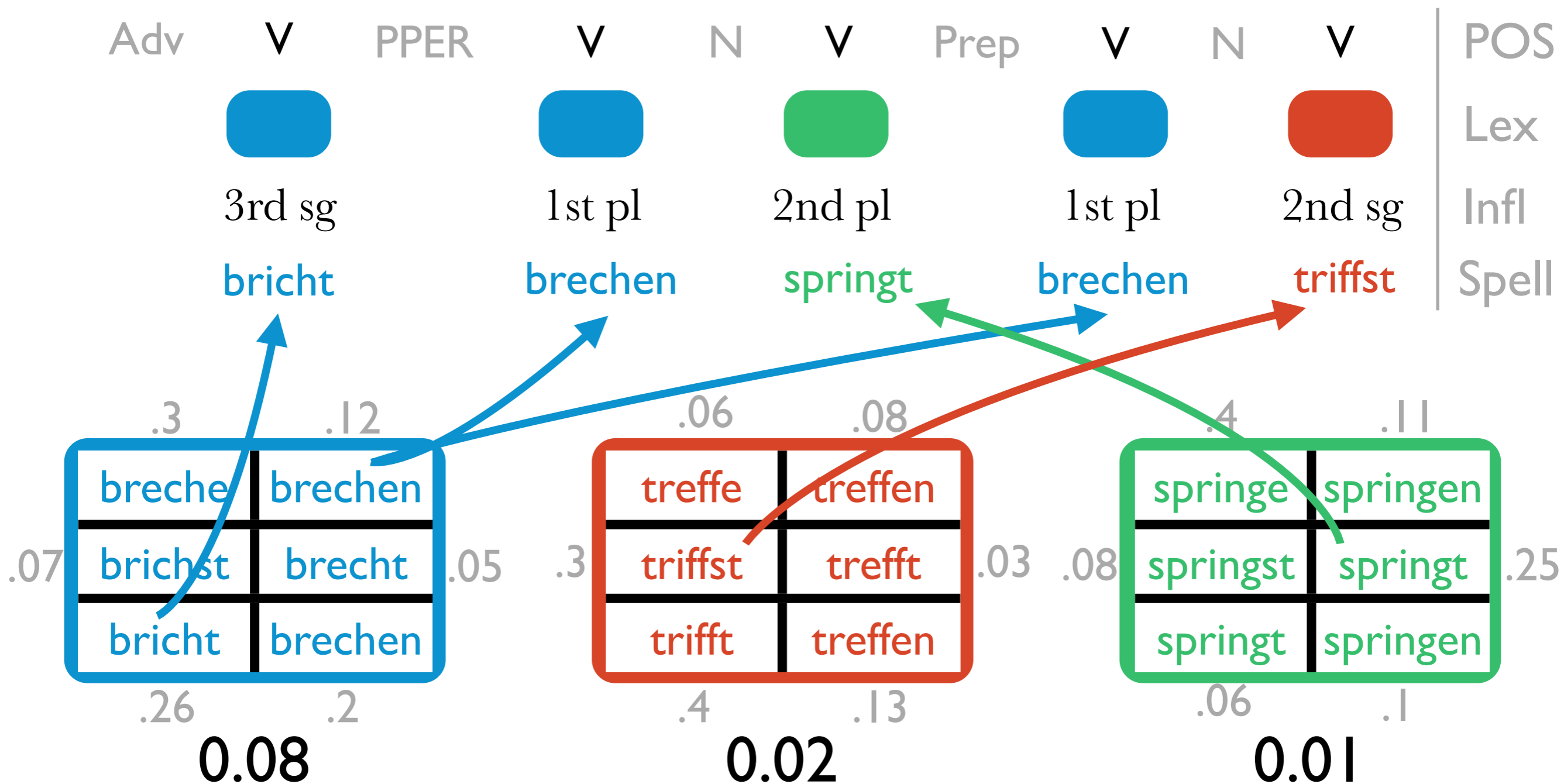


.3		.12		.06		.08		.4		.11	
breche	brechen	treffe	treffen	springe	springen						
brichst	brecht	triffst	trefft	springst	springt						
bricht	brechen	trifft	treffen	springt	springen						
.26		.2		.4		.13		.06		.1	
<b>0.08</b>				<b>0.02</b>				<b>0.01</b>			

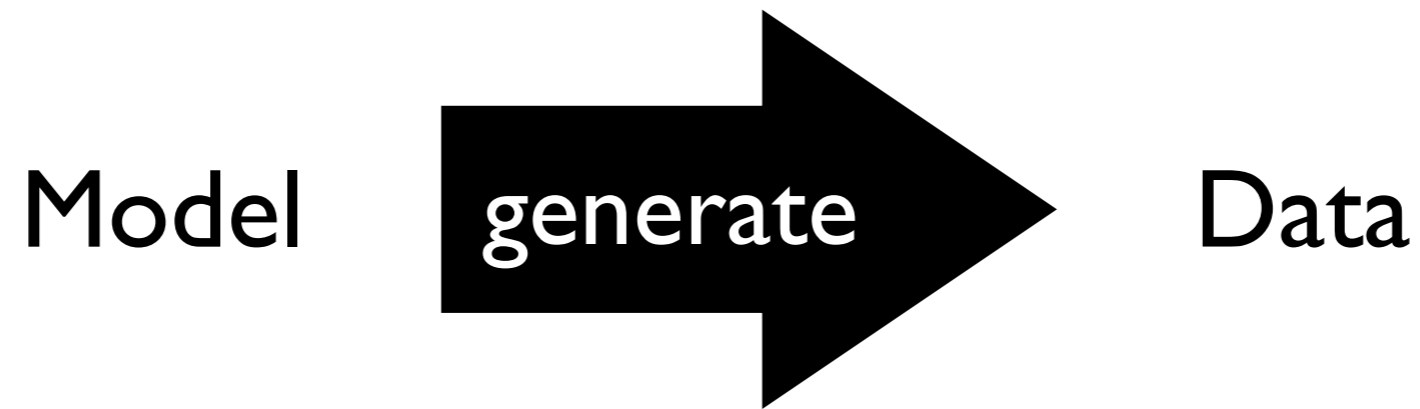
# 3

# Text & Paradigms

(5) Generate the corpus: Look up the *spellings*



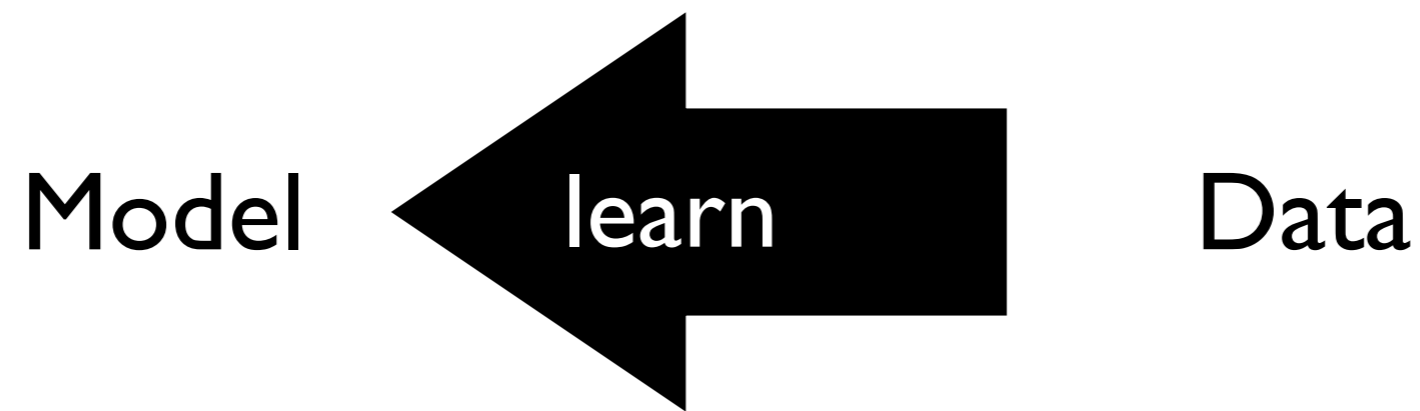
## **Generative story**



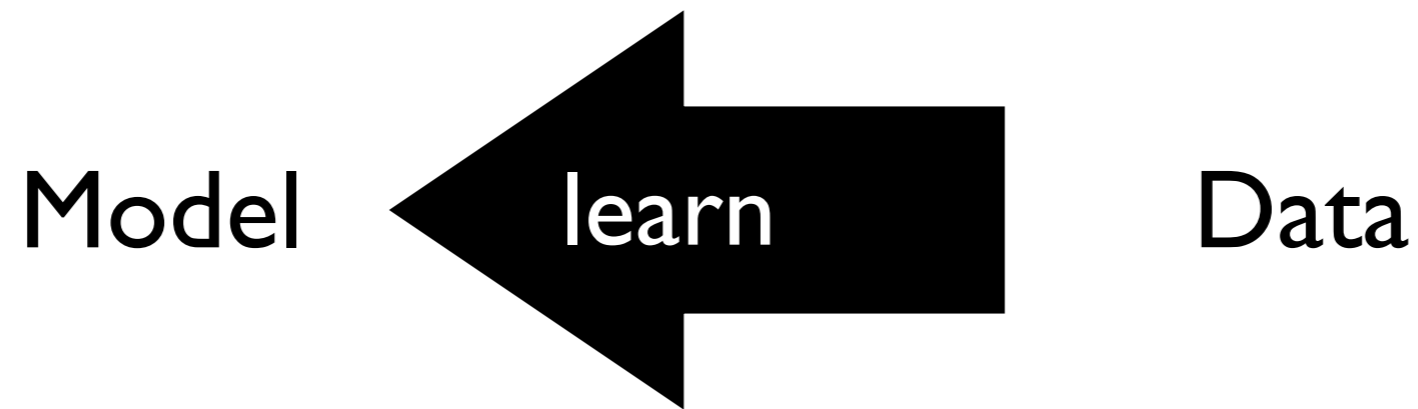
3

# Text & Paradigms

## **Inference (Sampling)**



## Inference (Sampling)

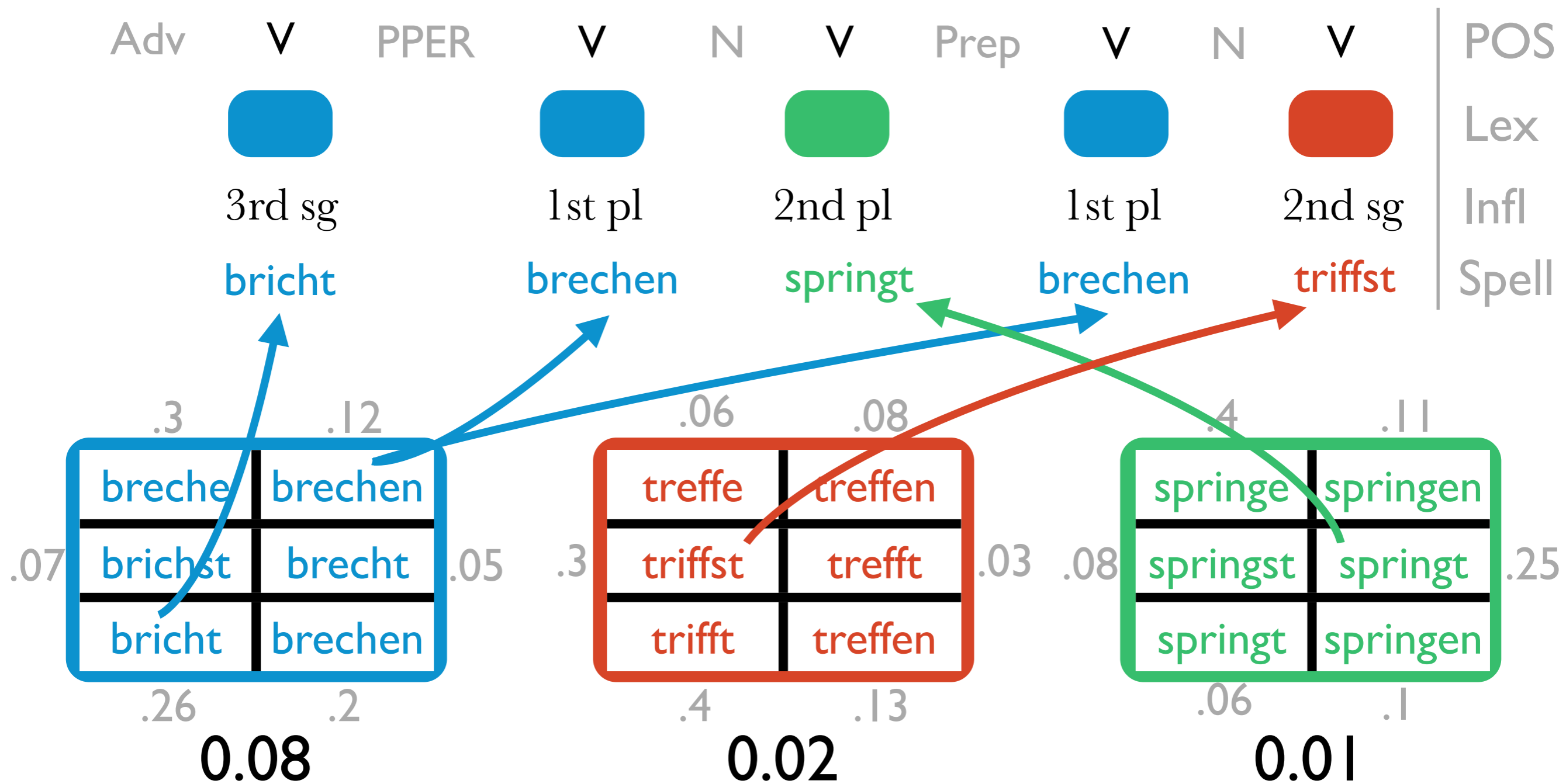


We start with **observed corpus tokens** and construct the paradigms and estimate all distributions



# 3

# Text & Paradigms







# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	bricht		brechen		springt		brechen		triffst	Lex
										Infl
										Spell

Minimal supervision: We do also observe a **few seed paradigms**, from which we can estimate an initial  $\theta$ , which parameterizes the finite-state MRFs

## 3

## Text &amp; Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	bricht		brechen	springt	brechen		triffst			Lex
										Infl
										Spell

Seed paradigm

treffe	treffen
triffst	trefft
trifft	treffen

Minimal supervision: We do also observe a **few seed paradigms**, from which we can estimate an initial  $\theta$ , which parameterizes the finite-state MRFs

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	bricht		brechen	springt	brechen		triffst			Lex
										Infl
										Spell

Seed paradigm

treffe	treffen
triffst	trefft
trifft	treffen

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	bricht		brechen		springt		brechen		triffst	Lex
										Infl
										Spell

Seed paradigm

treffe	treffen
triffst	trefft
trifft	treffen

Train initial  $\theta$  values



“morphological grammar”

“e” is likely to change into “i”

3rd sg ends in “t”

from 3rd sg to 1st pl, change vowel

...

# 3

# Text & Paradigms

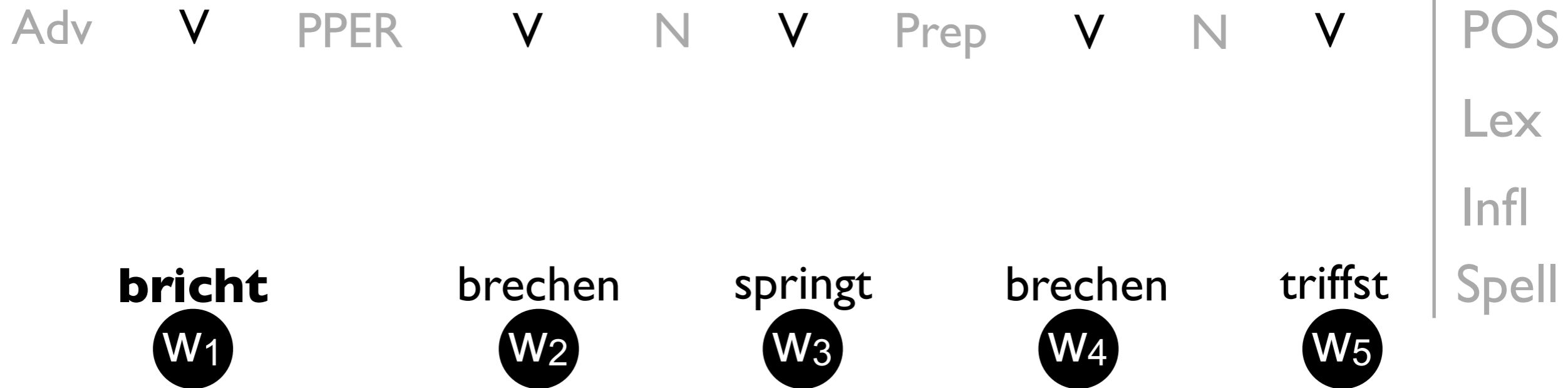
Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	bricht		brechen	springt	brechen		triffst			Lex
										Infl
										Spell

treffe	treffen
triffst	trefft
trifft	treffen



# 3

# Text & Paradigms



treffe	treffen
triffst	trefft
trifft	treffen

The red lexeme is completely specified and “bricht” does not fit in.

# 3

# Text & Paradigms

Adv

V

PPER

V

N

V

Prep

V

N

V

POS

Lex

Infl

Spell

**bricht**

W<sub>1</sub>

brechen

W<sub>2</sub>

springt

W<sub>3</sub>

brechen

W<sub>4</sub>

triffst

W<sub>5</sub>

treffe	treffen
triffst	trefft
trifft	treffen

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

Remember:  $\theta$  says,  
3rd sg ends in "t"

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
	<b>bricht</b>		brechen		springt		brechen		triffst	Lex
	W <sub>1</sub>		W <sub>2</sub>		W <sub>3</sub>		W <sub>4</sub>		W <sub>5</sub>	Infl
										Spell


treffe	treffen
triffst	trefft
trifft	treffen

1st sg	1st pl
2nd sg	2nd pl
W <sub>1</sub> 3rd sg	3rd pl

Remember:  $\theta$  says,  
3rd sg ends in "t"

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg									Infl
	<b>bricht</b>		brechen		springt		brechen		triffst	Spell
	<b>W<sub>1</sub></b>		<b>W<sub>2</sub></b>		<b>W<sub>3</sub></b>		<b>W<sub>4</sub></b>		<b>W<sub>5</sub></b>	

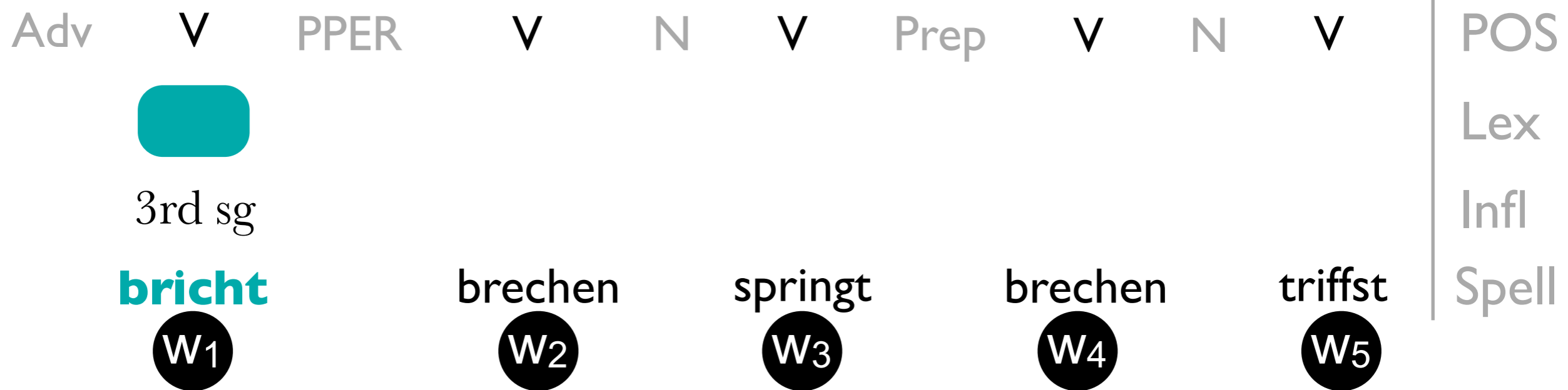
treffe	treffen
triffst	trefft
trifft	treffen

1st sg	1st pl
2nd sg	2nd pl
<b>W<sub>1</sub> bricht</b>	3rd pl

Remember:  $\theta$  says,  
3rd sg ends in "t"

# 3

# Text & Paradigms



treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brichen? brechen?

We immediately run  
finite-state-based  
**belief propagation**  
in this new paradigm.

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg									Infl
	<b>bricht</b>		<b>brechen</b>		springt		brechen		triffst	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? <b>brechen?</b>
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brichen? <b>brechen?</b>

# 3

# Text & Paradigms



Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl							Infl
	<b>bricht</b>		<b>brechen</b>		springt		brechen		triffst	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl							Infl
	<b>bricht</b>		<b>brechen</b>		<b>springt</b>		<b>brechen</b>		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	



treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>



# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl							Infl
	<b>bricht</b>		<b>brechen</b>		<b>springt</b>		<b>brechen</b>		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	




treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>

1st sg	1st pl
2nd sg	2nd pl
3rd sg	3rd pl

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg					Infl
	<b>bricht</b>		<b>brechen</b>		<b>springt</b>		<b>brechen</b>		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

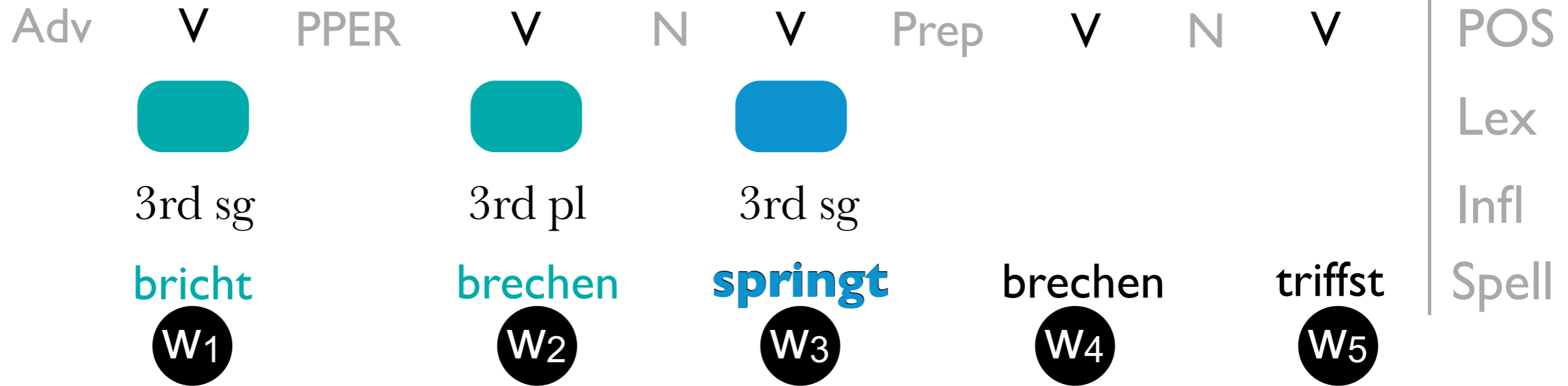
treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>

1st sg	1st pl
2nd sg	2nd pl
<b>W3</b> springt	3rd pl

# 3

# Text & Paradigms



Run belief propagation!




treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>

springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
<b>W3</b> springt	springen? sprengen?

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg					Infl
	bricht		brechen		springt		<b>brechen</b>		triffst	Spell
	W1		W2		W3		W4		W5	




treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
W1 bricht	brechen W2

springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
W3 springt	springen? sprengen?

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg					Infl
	bricht		brechen		springt		<b>brechen</b>		triffst	Spell
	W <sub>1</sub>		W <sub>2</sub>		W <sub>3</sub>		W <sub>4</sub>		W <sub>5</sub>	

It would fit well in two of the **cells**:

treffe	treffen
triffst	trifft
trifft	treffen





briche? breche?	brichen? <b>brechen?</b>
brichst? brechst?	bricht? brecht?
W <sub>1</sub> bricht	<b>brechen</b> W <sub>2</sub>

springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
W <sub>3</sub> springt	springen? sprengen?

# 3

# Text & Paradigms

Do not have to run BP now, because paradigm spellings are not changed.  
But frequency estimates change right away.

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl			Infl
	<b>bricht</b>		<b>brechen</b>		<b>springt</b>		<b>brechen</b>		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	










treffe	treffen
triffst	trefft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
<b>W1</b> bricht	brechen <b>W2</b>
	<b>W4</b>




springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
<b>W3</b> springt	springen? sprengen?


# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl			Infl
	bricht		brechen		springt		brechen		<b>triffst</b>	Spell
										






treffe	treffen
triffst	trifft
trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	bricht? brecht?
 bricht	brechen 
	

springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
 springt	springen? sprengen?

# 3

# Text & Paradigms

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl		2nd sg	Infl
	bricht		brechen		springt		brechen		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

	treffe	treffen
<b>W5</b>	triffst	trifft
	trifft	treffen

	briche? breche?	brichen? brechen?
	brichst? brechst?	bricht? brecht?
<b>W1</b>	bricht	brechen <b>W2</b>
		<b>W4</b>






	springe? sprenge?	springen? sprengen?
	springst? sprengst?	springt? sprengt?
<b>W3</b>	springt	springen? sprengen?



# 3

# Text & Paradigms

We will now re-estimate  $\theta$ , given our new “observations” (samples). This training method is called MCEM.

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl		2nd sg	Infl
	bricht		brechen		springt		brechen		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

	treffe	treffen
<b>W5</b>	triffst	trefft
	trifft	treffen

	briche? breche?	brichen? brechen?
	brichst? brechst?	bricht? brecht?
<b>W1</b>	bricht	brechen <b>W2</b>
		<b>W4</b>

	springe? sprenge?	springen? sprengen?
	springst? sprengst?	springt? sprengt?
<b>W3</b>	springt	springen? sprengen?

# 3

# Text & Paradigms

We go over the corpus over and over again, **re-analyzing words** in the light of **newly acquired knowledge** about table frequencies, inflection frequencies and the updated “morphological grammar”  $\theta$ .

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl		2nd sg	Infl
	bricht		brechen		springt		brechen		triffst	Spell
	W <sub>1</sub>		W <sub>2</sub>		W <sub>3</sub>		W <sub>4</sub>		W <sub>5</sub>	

	treffe	treffen
W <sub>5</sub>	triffst	trefft
	trifft	treffen






	briche? breche?	brichen? brechen?
	brichst? brechst?	bricht? brecht?
W <sub>1</sub>	bricht	brechen
		W <sub>2</sub>
		W <sub>4</sub>

	springe? sprenge?	springen? sprengen?
	springst? sprengst?	springt? sprengt?
W <sub>3</sub>	springt	springen? sprengen?

# 3

# Text & Paradigms

We go over the corpus over and over again, **re-analyzing words** in the light of **newly acquired knowledge** about table frequencies, inflection frequencies and the updated “morphological grammar”  $\theta$ .

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	3rd sg		3rd pl		3rd sg		3rd pl		2nd sg	Infl
	<b>bricht</b>		brechen		springt		brechen		<b>triffst</b>	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

	treffe	treffen
<b>W5</b>	triffst	trefft
	trifft	treffen

	briche? breche?	brichen? brechen?
	brichst? brechst?	<b>bricht?</b> brecht?
<b>W1</b>	<b>bricht</b>	brechen <b>W2</b>
		<b>W4</b>

	springe? sprenge?	springen? sprengen?
	springst? sprengst?	springt? sprengt?
<b>W3</b>	<b>springt</b>	springen? sprengen?

# 3

# Text & Paradigms

We go over the corpus over and over again, **re-analyzing words** in the light of **newly acquired knowledge** about table frequencies, inflection frequencies and the updated “morphological grammar”  $\theta$ .

Adv	V	PPER	V	N	V	Prep	V	N	V	POS
										Lex
	2nd pl		3rd pl		3rd sg		3rd pl		2nd sg	Infl
	<b>bricht</b>		brechen		springt		brechen		triffst	Spell
	<b>W1</b>		<b>W2</b>		<b>W3</b>		<b>W4</b>		<b>W5</b>	

	treffe	treffen
<b>W5</b>	triffst	trifft
	trifft	treffen

briche? breche?	brichen? brechen?
brichst? brechst?	<b>bricht</b> <b>W1</b>
bricht? brecht?	brechen <b>W2</b>
	<b>W4</b>

springe? sprenge?	springen? sprengen?
springst? sprengst?	springt? sprengt?
<b>W3</b>	springt <b>W3</b>
	springen? sprengen?

## Summary of the sampling process:

- Constantly update **frequency estimates** for lexemes and inflections
- Often update the “morphological grammar”  $\theta$
- Keep **re-analyzing words** accordingly
- Run **finite-state BP** to fill in missing paradigm cells
- **Important:** Often, BP will produce a regular and some more irregular candidates, one of them is found in the corpus and placed in the cell, so we “learn” it!

## Summary of the sampling process:

- Inflections and lexemes at the corpus positions are **sampled**.
- The missing paradigm cells are **marginalized** over.
- The “morphological grammar”  $\theta$  is **maximized**.
- We are using a collapsed **Gibbs** sampler, according to a hierarchical **Chinese Restaurant Process**, with interspersed finite-state **belief propagation** steps

## Sampling Speedup:

- Improve mixing and prevent “lock-in”
- Do not sample word by word
- Instead: Pick a whole lexeme, remove all its current words and perform Gibbs sampling just with those words (for one or more iterations)

## Obtaining results for evaluation

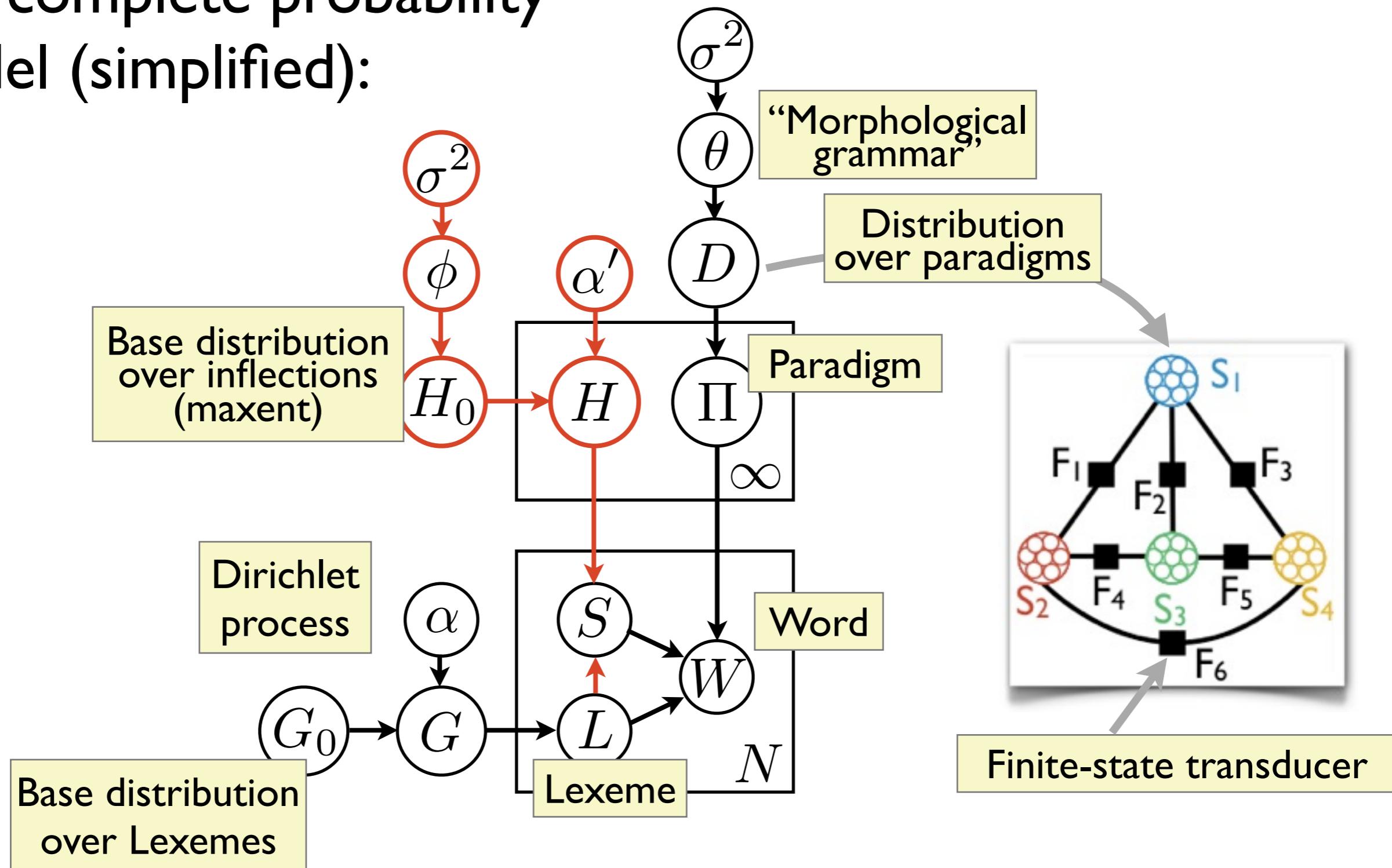
- We add many paradigms, in which **only the lemma form** is given, but the other slots are empty.
- Just **keep track** of what corpus tokens the sampler places in those empty cells, or what candidates will be suggested from belief propagation.
- To get an answer for particular cell, get its **marginal** probability distribution at end of each iteration. At the end, get **average prob. per spelling** and report highest-scoring one



# 3

# Text & Paradigms

The complete probability model (simplified):



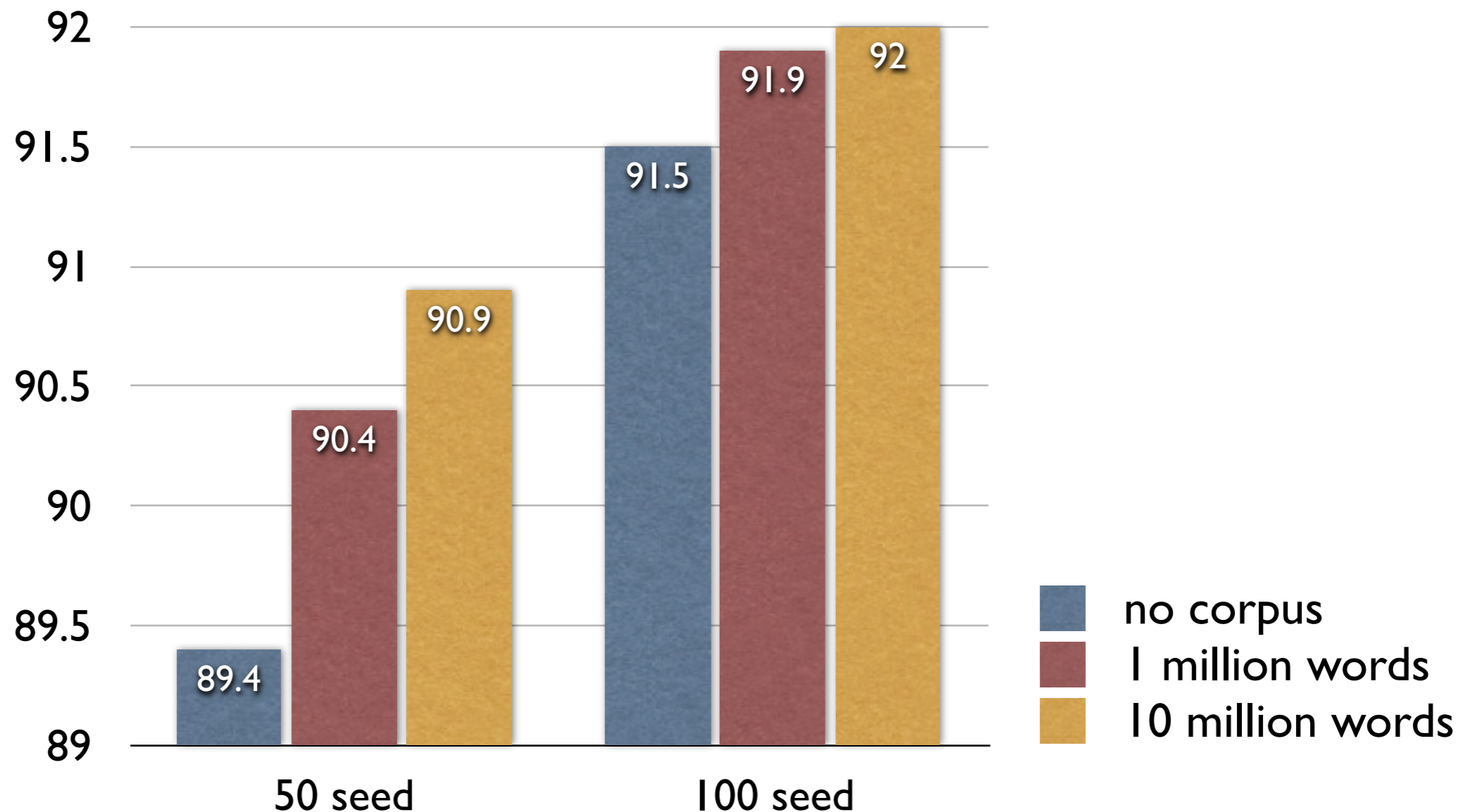
Experiment:

## **Learn German inflectional morphology**

- Given:
  - 50 seed paradigms (from CELEX)
  - German corpus of 10 million words (from “WaCKy” corpus)
- Test:  
For 5,415 German verbs, predict paradigms with 21 inflections each

# Text & Paradigms

Adding a large text corpus significantly improves prediction accuracy.



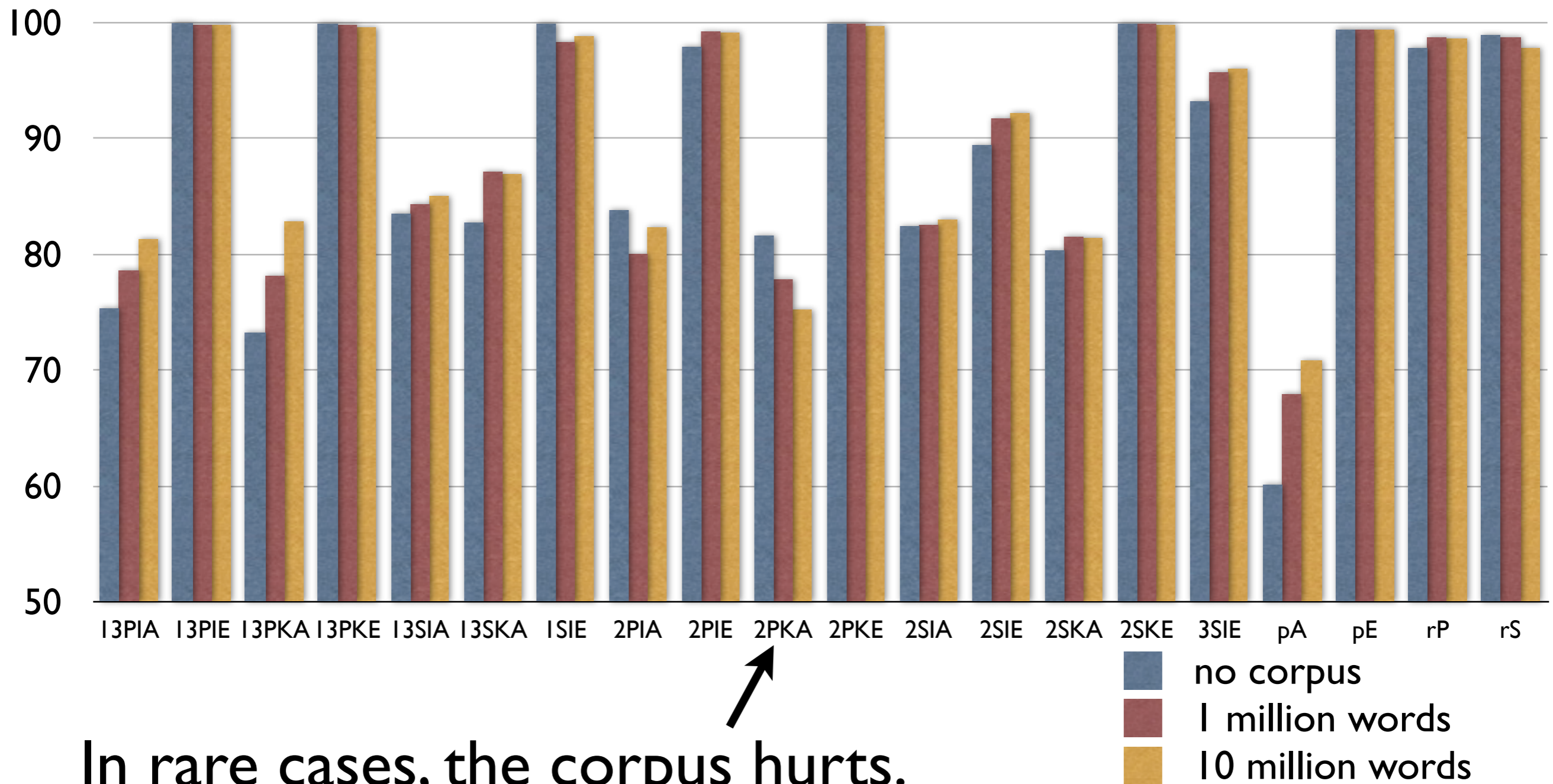
Regular:  
85.4

# 3

# Text & Paradigms

Many forms are easy (~100% acc.)

Large gains on some forms (irregularities)



- Simplifications:
  - The finite-state MRFs use a simple factor graph that just connects the lemma to all other forms, but not the forms among each other
  - Information flows from one form to the other through the lemma slot
  - Better factor graphs give orthogonal improvements, see Ch. 3

## Possible model extensions

- **Adding context:** Take neighboring words into account, so that *1st pl* can be more likely after “we” than after “she”, etc.
- **Adding topic variables:** Useful for deciding that a particular spelling belongs into one lexeme rather than another (singed does not fit into “sing” paradigm because it’s a different topic).

## **Remaining morphological issues**

- Reduplication
- Metathesis
- Consonant doubling
- Circumfixes
- Templatic morphology
- Interaction with phonology

# 3 Conclusions / Contributions

- Presented novel, principled approach to **learning inflectional morphology** of a language
- Developed joint **probability model** over **text corpus** and **inflectional paradigms**, which are hidden
- Presented **type-based sampling procedure** that discovers inflectional paradigms from plain text



# 3 Conclusions / Contributions

- Presented novel **generative story** for inflectional morphology, ...
- ... based on **ordinary linguistic notions** (lexemes, inflections, paradigms)
- Clusters corpus words into lexemes and inflections using hierarchical **Dirichlet process**
- Allows **unbounded number of lexemes**
- Handles **nonconcatenative**, irregular morphology

# Related Work

## 1 String pairs

- *Sherif & Kondrak (2007), Hong et al (2009), and others, get 1-best alignment, segment into chunks, and score chunks individually*
- Others get 1-best alignment and train conventional *n*-gram model (*Bisani & Ney (2008), and others*)
- In contrast, we sum over all alignments, use features, add latent variables, generate arbitrary-length output, use global normalization

# Related Work

2

## Multiple Strings

- Joint models over multiple strings have not been tackled much before
- Exception: *Bouchard-Cote et al (2007)*, who defines a directed graphical model, does not run BP inference and does not use FSTs

# Related Work

## 3

### **Text and Paradigms**

- No one has modeled structured inflectional paradigms before
- Typically, simple concatenative morphology is assumed (*Harris (1955), Chan (2008)*), but see *Yarowsky and Wicentowski (2002)*
- *Goldsmith (2001)* and others extract “suffix paradigms” (lists of verb endings)
- In contrast, we extract structured paradigms that seamlessly handle non-concatenative phenomena

# Conclusions / Contributions

- Presented several novel probability models step by step, each smaller one being a factor component in the next bigger one
- Developed a coherent, unified statistical approach to inflectional morphology, which advances the state-of-the-art in computational morphology
- Extracted detailed and structured morphological knowledge from plain text;
- Presented the most ambitious morphological knowledge discovery task and method to date

# Conclusions / Contributions

- All presented models have many further uses in NLP:
  - string-pair models for transliteration, pronunciation modeling, spelling correction, etc.
  - multiple-string models for bioinformatics, historical linguistics, phonology, transliteration, etc.
  - text & paradigms model for text generation, machine translation, etc.

# Conclusions / Contributions

- This thesis naturally brings together many different concepts from machine learning, NLP, and linguistics, in various novel ways:
  - In part 1, we use linguistically inspired features, latent variables, FSTs and dynamic programming.
  - In part 2, we combine FSTs with graphical models and belief propagation.
  - In part 3, we bring together all of the above with statistical tools like Dirichlet process and collapsed Gibbs sampling to tackle a novel task that people have not been able to tackle before.

# Publications

1. Dreyer and Eisner. *in prep.* Discovering Morphological Paradigms From Plain Text.
2. Dreyer and Eisner. 2009. Graphical Models over Multiple Strings. *EMNLP*.
3. McNamee, Dredze, Gerber, Garera, Finin, Mayfield, Piatko, Rao, Yarowsky, Dreyer. 2009. HLCOE Approaches to Knowledge Base Population. *TAC*.
4. Dreyer, Eisner and Smith. 2008. Finite-State Modeling of Log-Linear String Transductions with Latent Variables and Backoff Features. *EMNLP*.
5. Karakos, Eisner, Khudanpur, Dreyer. 2008. Machine Translation System Combination using ITG-based Alignments. *ACL*.
6. Dreyer and Shafran. 2007. Exploiting Prosody for PCFGs with Latent Annotations. *Interspeech*.
7. Dreyer, Hall, Khudanpur. 2007. Comparing Reordering Constraints for SMT Using Efficient BLEU Oracle Computation. *SSST*.
8. Dreyer and Eisner. 2006. Better Informed Training of Latent Syntactic Features. *EMNLP*.
9. Dreyer, Smith, Smith. 2006. Vine Parsing and Minimum Risk Reranking for Speed and Precision. *CoNLL*.
10. Burbank, Carpuat, Clark, Dreyer, Fox, Groves, Hall, Hearne, Melamed, Shen, Way, Wellington, Wu. 2005. Statistical Machine Translation by Parsing. *CLSP*.