TTIC 31210:
Advanced Natural Language Processing

Kevin Gimpel

Spring 2017

Lecture 14:
Finish up Bayesian/Unsupervised NLP, Start Structured Prediction

- Today and Wednesday: structured prediction
- No class Monday May 29 (Memorial Day)
- Final class is Wednesday May 31

- Assignment 3 has been posted, due Thursday June 1
- Final project report due Friday, June 9

Key Quantities

$$p(x, z, \theta \mid \alpha) = p(\theta \mid \alpha)\, p(z \mid \theta)\, p(x \mid z, \theta)$$

Our data is a set of samples: $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$

joint: $p(x^{(1)}, \ldots, x^{(n)}, z^{(1)}, \ldots, z^{(n)}, \theta \mid \alpha)$

$$= p(\theta \mid \alpha) \left(\prod_{i=1}^{n} p(z^{(i)} \mid \theta)\, p(x^{(i)} \mid z^{(i)}, \theta) \right)$$

posterior: $p(z^{(1)}, \ldots, z^{(n)}, \theta \mid x^{(1)}, \ldots, x^{(n)}, \alpha)$

collapsed posterior: $p(z^{(1)}, \ldots, z^{(n)} \mid x^{(1)}, \ldots, x^{(n)}, \alpha)$

Gibbs Sampling Template

$U_1, ..., U_p =$ latent variables

$U_{-i} =$ all latent variables other than U_i

$\boldsymbol{X} =$ all observed data and hyperparameters

Gibbs sampling:

initialize all U_i to values u_i

repeat until convergence:

sample u from $p(U_i \mid u_{-i}, \boldsymbol{X})$

set $U_i \leftarrow u$

LDA

Generative Story:

$\beta_k \sim \text{Dirichlet}(\psi)$

$\theta^{(i)} \sim \text{Dirichlet}(\alpha)$

$Z^{(i,j)} \sim \text{Multinomial}(\theta^{(i)})$

Posteriors:

$\beta_k \mid \text{ everything else} \sim \text{Dirichlet}(\psi + n_k)$

$\theta^{(i)} \mid \text{ everything else} \sim \text{Dirichlet}(\alpha + m^{(i)})$

$Z^{(i,j)} \mid \text{ everything else} \sim \text{Multinomial}(\theta^{(i)} \odot \beta_{\cdot, w^{(i,j)}})$

Expectation Maximization (EM)

$$\max_{\theta} \prod_{i} \sum_{z} p(x^{(i)}, z \mid \theta)$$

- EM is an algorithmic template that finds a local maximum of the marginal likelihood of the observed data

EM

- "E" step:
  - compute posteriors over latent variables:

$$\text{for each } i, \; q_i(z) = p(z \mid x^{(i)}, \theta)$$

- "M" step:
  - update parameters given posteriors:

$$\theta = \underset{\theta'}{\operatorname{argmax}} \sum_i \sum_z q_i(z) \log \frac{p(x^{(i)}, z \mid \theta')}{q_i(z)}$$

Different Views of the Dirichlet Process (DP)

- last time we discussed the “stick-breaking” view of the DP
- today we’ll briefly discuss the “Chinese Restaurant Process” view
- with both views, we still have the same DP hyperparameters

  (base distribution & concentration parameter)

Base Distribution G_0 for DP

- our unbounded distribution over items will choose them from the base distribution
- base distribution usually has infinite support
- simple example base distribution for our morph lexicon:

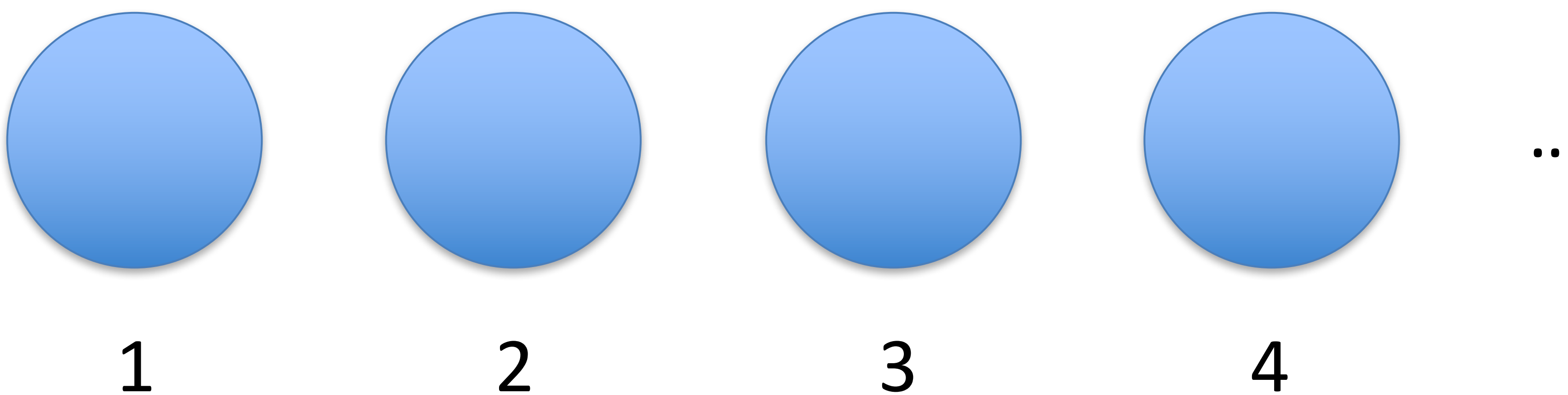
$$G_0(m) = p_{\text{len}}(|m|) \prod_{i=1}^{|m|} p_{\text{char}}(m_i)$$

Concentration Parameter

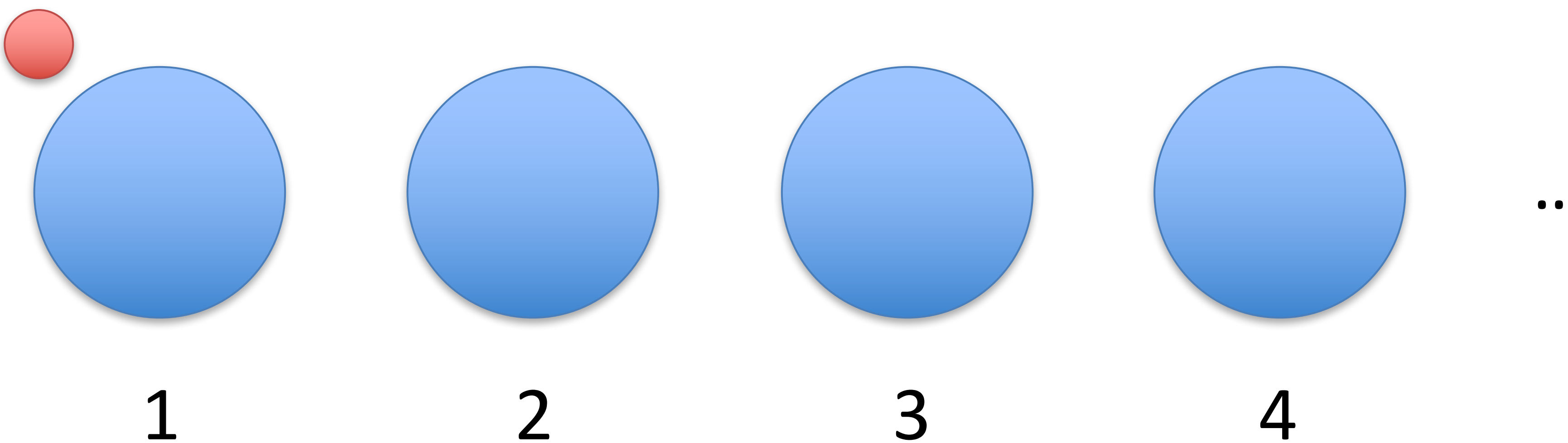
- in stick-breaking process, concentration parameter determines how much of the stick we break off each time
- high concentration == small parts of stick

- the stick-breaking construction of the DP is useful for specifying models and defining inference algorithms
- another useful way of representing a draw from a DP is with the Chinese Restaurant Process (CRP)
  - CRP provides a distribution over partitions with an unbounded number of parts

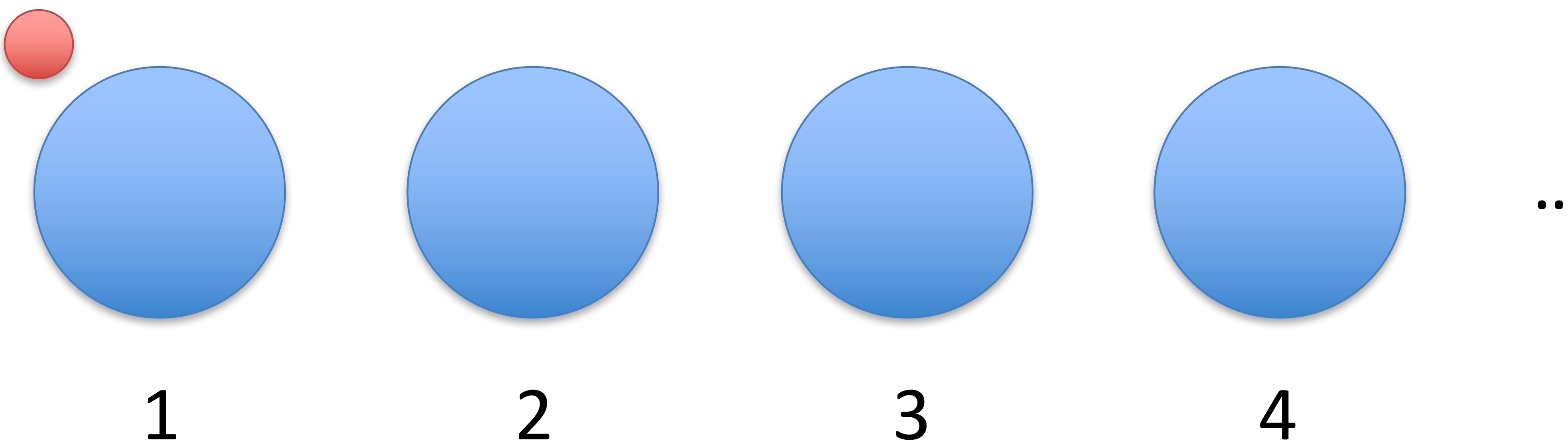
- imagine a Chinese restaurant with an infinite number of tables…



- first customer sits at first table:

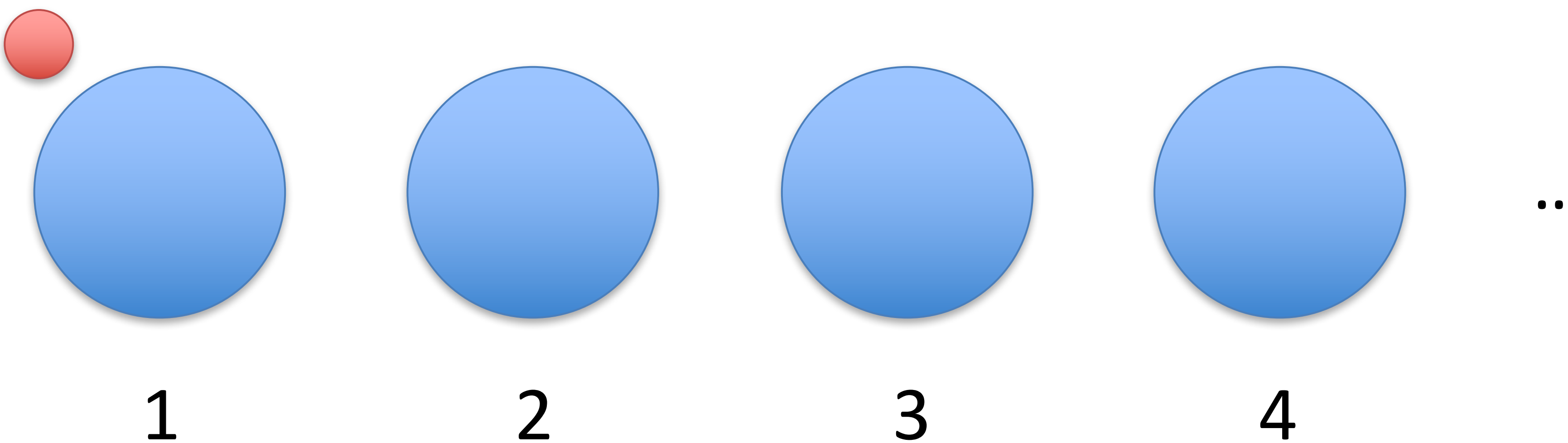


- second customer enters, chooses a table:



- second customer enters,

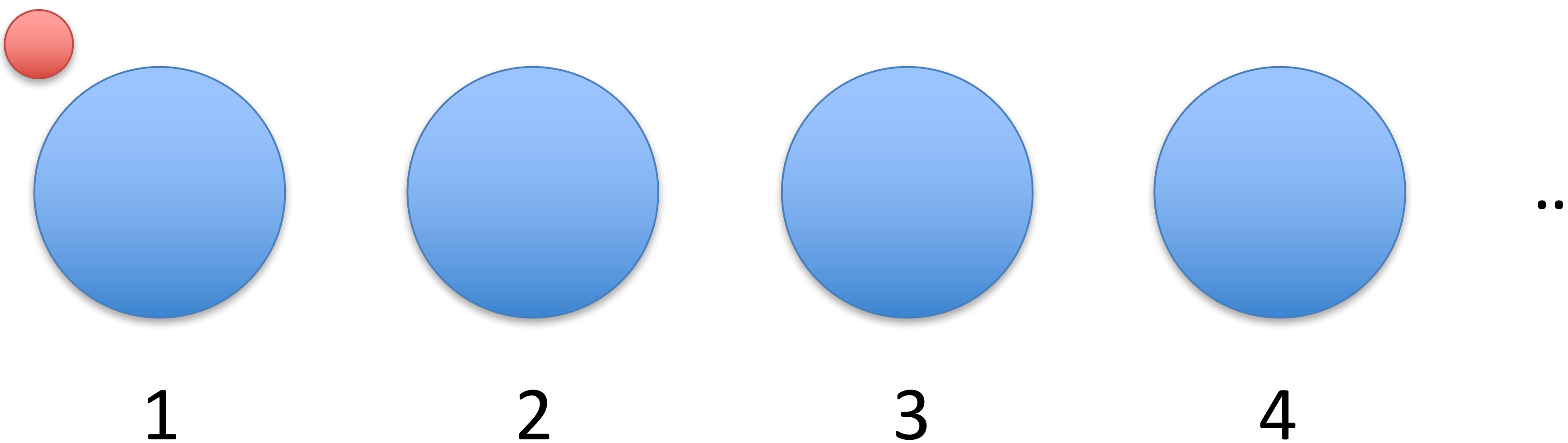
chooses table 1: $p(Y^{(2)} = 1 \mid Y^{(1)}, s) = \dfrac{1}{1+s}$



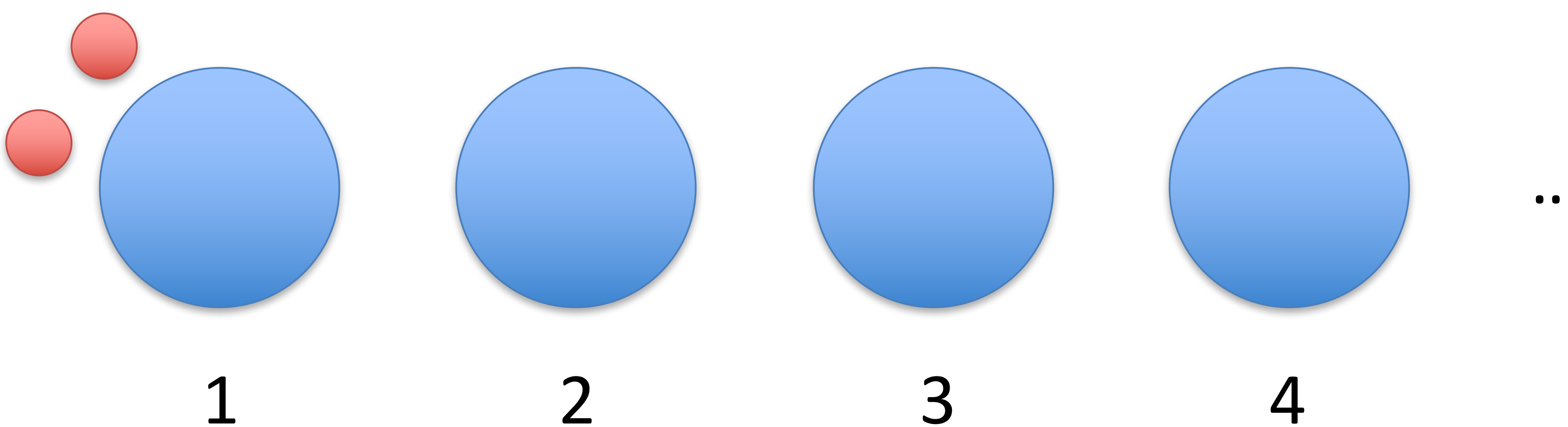
- second customer enters,

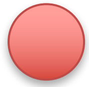
chooses table 1: $p(Y^{(2)} = 1 \mid Y^{(1)}, s) = \dfrac{1}{1+s}$

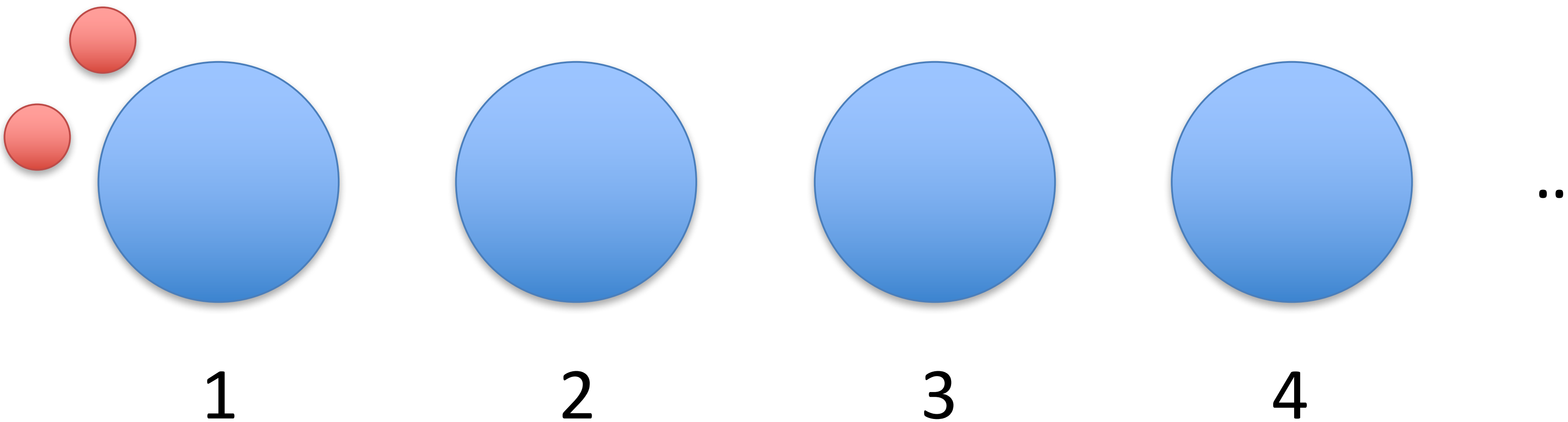
chooses new table: $p(Y^{(2)} = 2 \mid Y^{(1)}, s) = \dfrac{s}{1+s}$



- second customer enters, chooses table 1



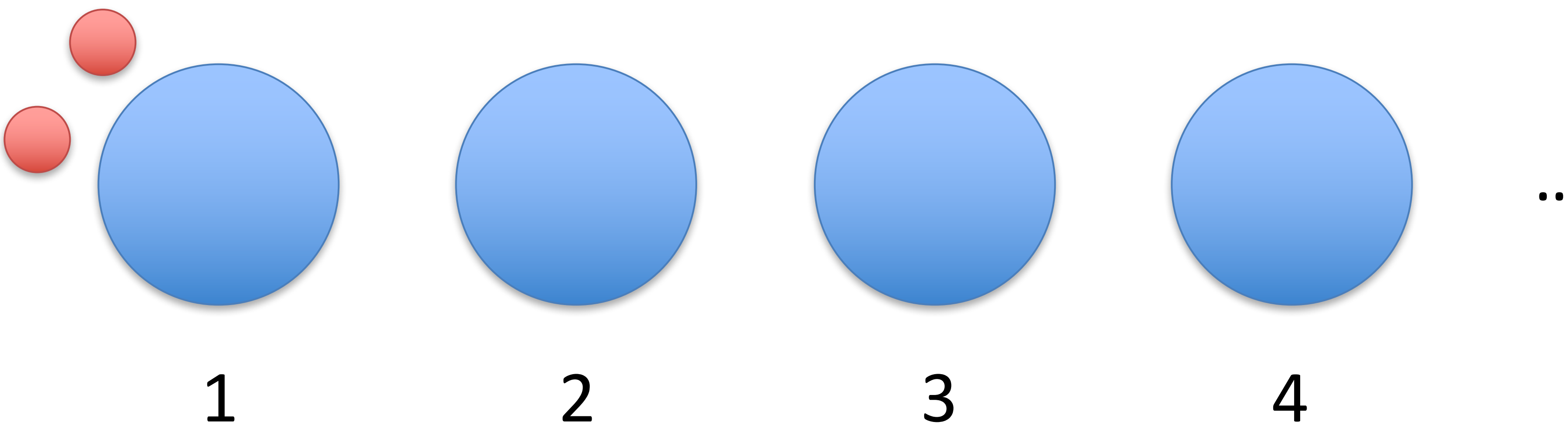
- third customer  enters,

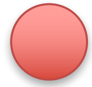


- third customer enters,

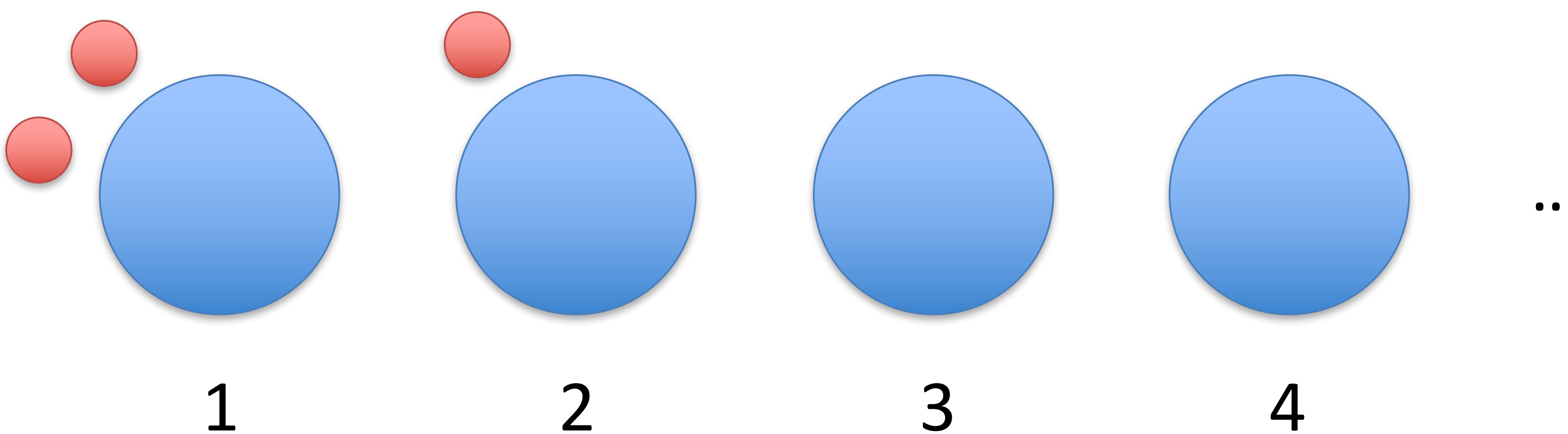
chooses table 1: $p(Y^{(3)} = 1 \mid Y^{(1)}, Y^{(2)}, s) = \dfrac{2}{2+s}$

chooses new table: $p(Y^{(3)} = 2 \mid Y^{(1)}, Y^{(2)}, s) = \dfrac{s}{2+s}$



- third customer  enters,

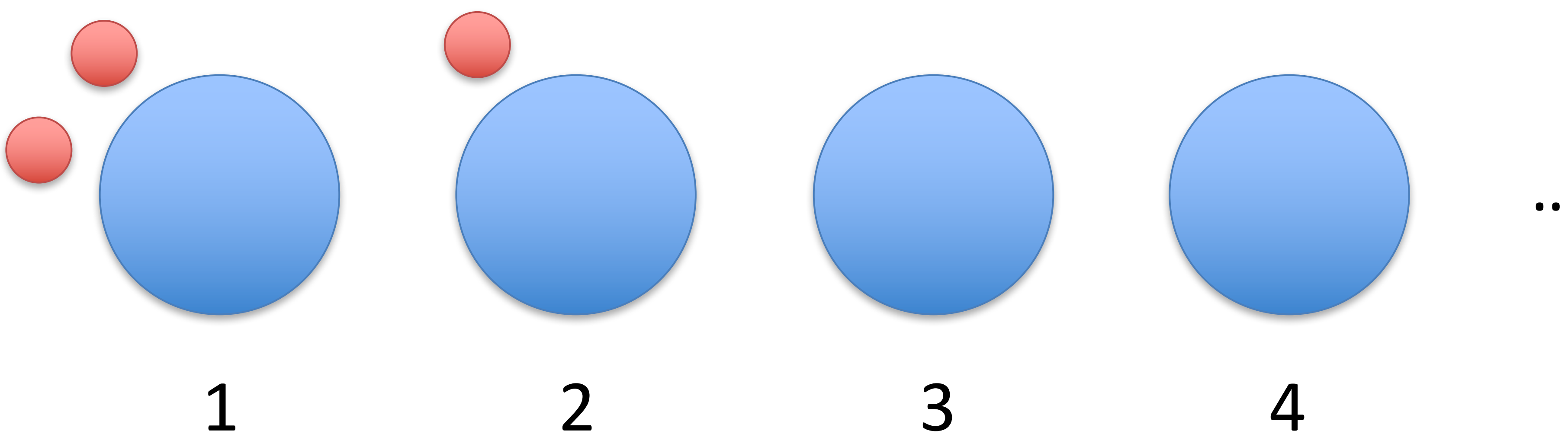
chooses new table

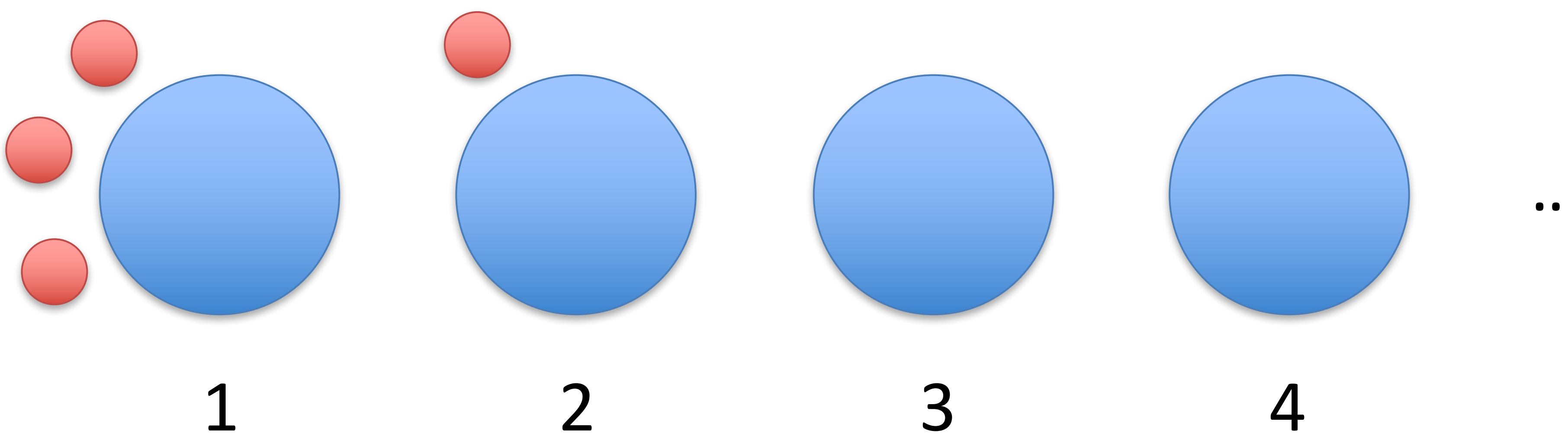


- fourth customer enters,

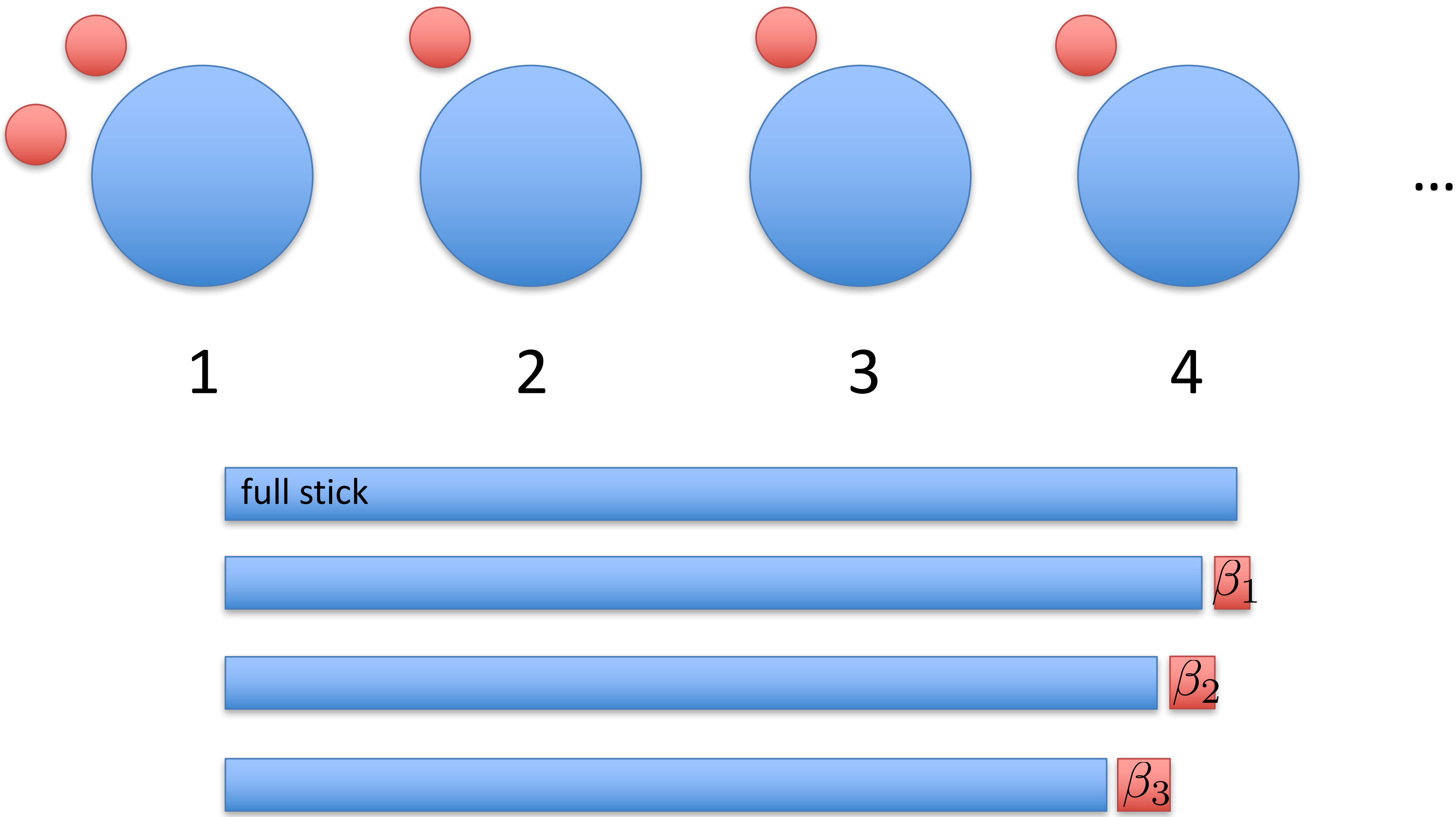
p(choose table 1): $\frac{2}{3+s}$ p(choose table 2): $\frac{1}{3+s}$

p(choose new table): $\frac{s}{3+s}$

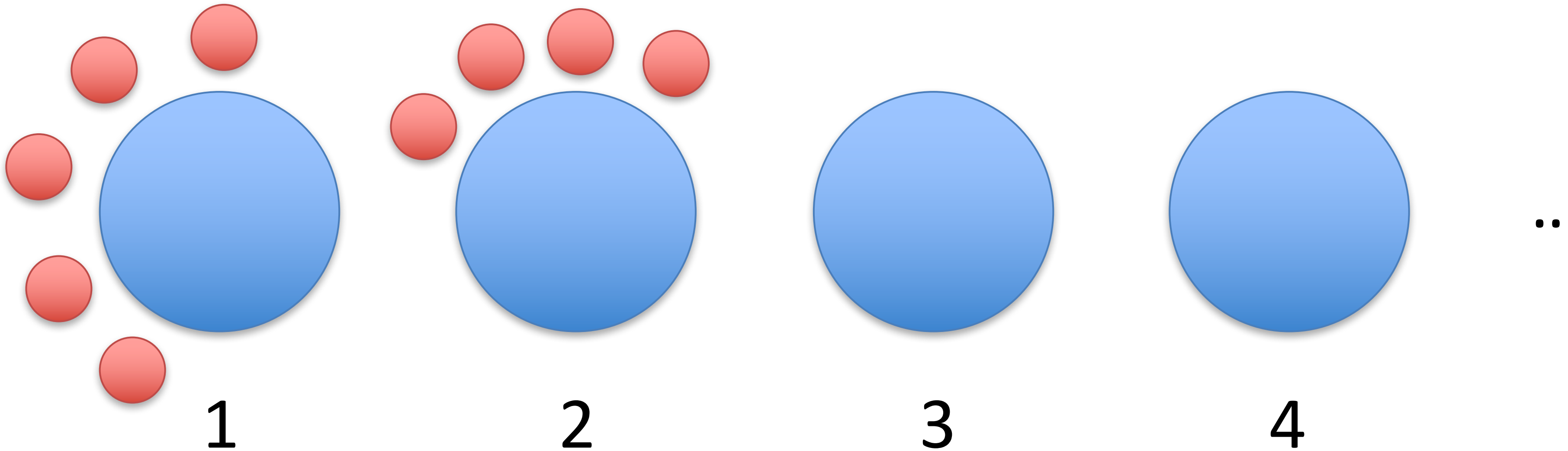



1
2
3
4
...

- large value of concentration parameter:



- small value of concentration parameter:



A Draw G from a DP
(Stick-Breaking Representation)

draw infinite probabilities from stick-breaking process with parameter s

1: $\beta \sim \mathrm{GEM}(s)$

draw atoms from base distribution

2: $\theta_1, \theta_2, \ldots \sim G_0$

atoms can be repeated!

3: the distribution G is defined as:

$$G(\theta) = \sum_{k=1}^{\infty} \beta_k \mathbb{I}[\theta = \theta_k]$$

$$G(\text{“ing”}) = \sum_{k=1}^{\infty} \beta_k \mathbb{I}[\text{“ing”} = \theta_k]$$

A Representation of G Drawn from a DP
(Chinese Restaurant Process Representation)

draw table assignments for *n* customers with parameter *s*

1: $y^{(1)}, \ldots, y^{(n)} \sim \mathrm{CRP}(s)$

for each occupied table, draw atom from base distribution

2: $\phi_1, \ldots, \phi_{y_{\max}} \sim G_0$

3: set each $\theta^{(i)}$ to $\phi_{y^{(i)}}$ for $i \in \{1, \ldots, n\}$

$$y_{\max} = \max_i y^{(i)}$$

each draw from G is an atom, where its probability comes from the number of customers at its table

number of tables occupied

When to be Bayesian?

- if you’re doing unsupervised learning or learning with latent variables
- if you want to marginalize out some model parameters
- if you want to learn the structure/architecture of your model
- if you want to learn a potentially-unbounded lexicon (Bayesian nonparametrics)

What is Structured Prediction?

Modeling, Inference, Learning

$$\text{classify}(x, \boldsymbol{\theta}) = \underset{y}{\text{argmax}} \;\; \text{score}(x, y, \boldsymbol{\theta})$$

Modeling, Inference, Learning

modeling: define score function

$$\mathrm{classify}(x, \boldsymbol{\theta}) = \underset{y}{\mathrm{argmax}}\ \mathrm{score}(x, y, \boldsymbol{\theta})$$

- **Modeling**: How do we assign a score to an (*x*,*y*) pair using parameters θ?

Modeling, Inference, Learning

inference: solve argmax

modeling: define score function

$$\mathrm{classify}(x, \boldsymbol{\theta}) = \underset{y}{\mathrm{argmax}} \; \mathrm{score}(x, y, \boldsymbol{\theta})$$

- **Inference**: How do we efficiently search over the space of all labels?

Modeling, Inference, Learning

inference: solve argmax

modeling: define score function

$$\operatorname{classify}(x, \boldsymbol{\theta}) = \operatorname*{argmax}_{y} \ \operatorname{score}(x, y, \boldsymbol{\theta})$$

learning: choose $\boldsymbol{\theta}$

- **Learning**: How do we choose $\boldsymbol{\theta}$?

Modeling, Inference, Learning

inference: solve argmax

modeling: define score function

$$\text{classify}(x, \boldsymbol{\theta}) = \operatorname*{argmax}_{y} \ \text{score}(x, y, \boldsymbol{\theta})$$

learning: choose $\boldsymbol{\theta}$

Structured Prediction:

size of output space is exponential in size of input

or is unbounded (e.g., machine translation)

(we can't just enumerate all possible outputs)

Simplest kind of structured prediction: Sequence Labeling

Part-of-Speech Tagging

determiner	verb (past)	prep.	proper noun	proper noun	poss.	adj.	noun
Some	questioned	if	Tim	Cook	’s	first	product

modal	verb	det.	adjective	noun	prep.	proper noun	punc.
would	be	a	breakaway	hit	for	Apple	.

Formulating segmentation tasks as sequence labeling via B-I-O labeling:

Named Entity Recognition

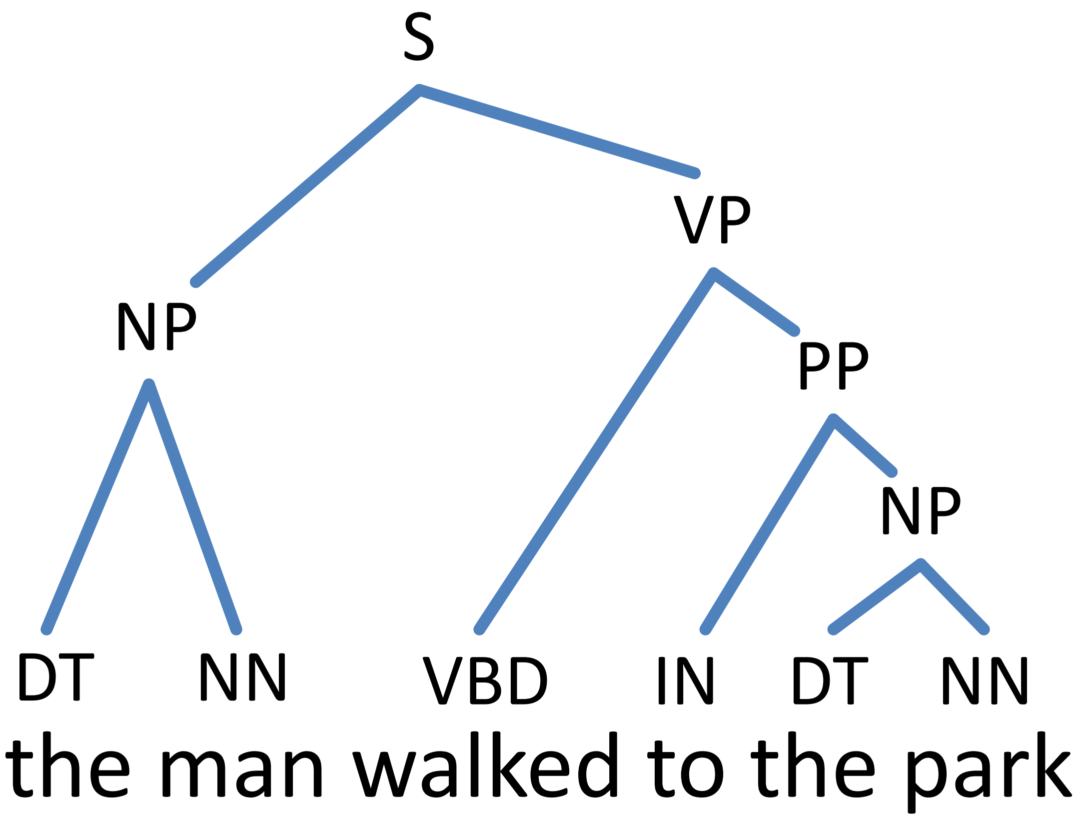
O	O	O	B-PERSON	I-PERSON	O	O	O
Some	questioned	if	Tim	Cook	’s	first	product

O	O	O	O	O	O	B-ORGANIZATION	O
would	be	a	breakaway	hit	for	Apple	.

B = “begin”
I = “inside”
O = “outside”

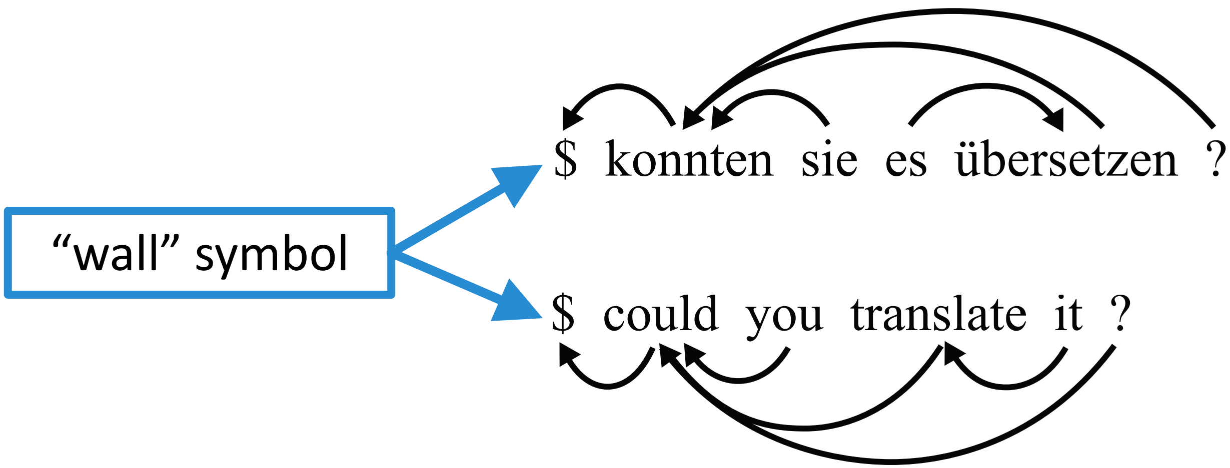
Constituent Parsing

(S (NP the man) (VP walked (PP to (NP the park))))



Key:
S = sentence
NP = noun phrase
VP = verb phrase
PP = prepositional phrase
DT = determiner
NN = noun
VBD = verb (past tense)
IN = preposition

Dependency Parsing



Coreference Resolution

As we head towards training camp, the **Philadelphia Eagles** have finally filled most of their needs on offense.

One of the main goals for this off-season was to find weapons for **the team's** franchise quarterback, **Carson Wentz**. **The Eagles** needed a wide receiver who could stretch the field and give **Wentz** the opportunity to throw the long ball.

They signed receiver **Torrey Smith** to a 3-year deal. While the signing of **Smith** was huge for **the team**, the biggest signing **the Eagles** made was former **Chicago Bears** receiver **Alshon Jeffery**. **He** had a solid 5-year stint in Chicago, but as **the team** started to fall apart, **Jeffery** was forced to explore other options.

Coreference Resolution

input: a document

output: a set of “mentions” (textual spans in document), and memberships of those mentions in clusters

Semantic Role Labeling

Who did *what* to *whom* at *where?*

The police officer detained the suspect at the scene of the crime

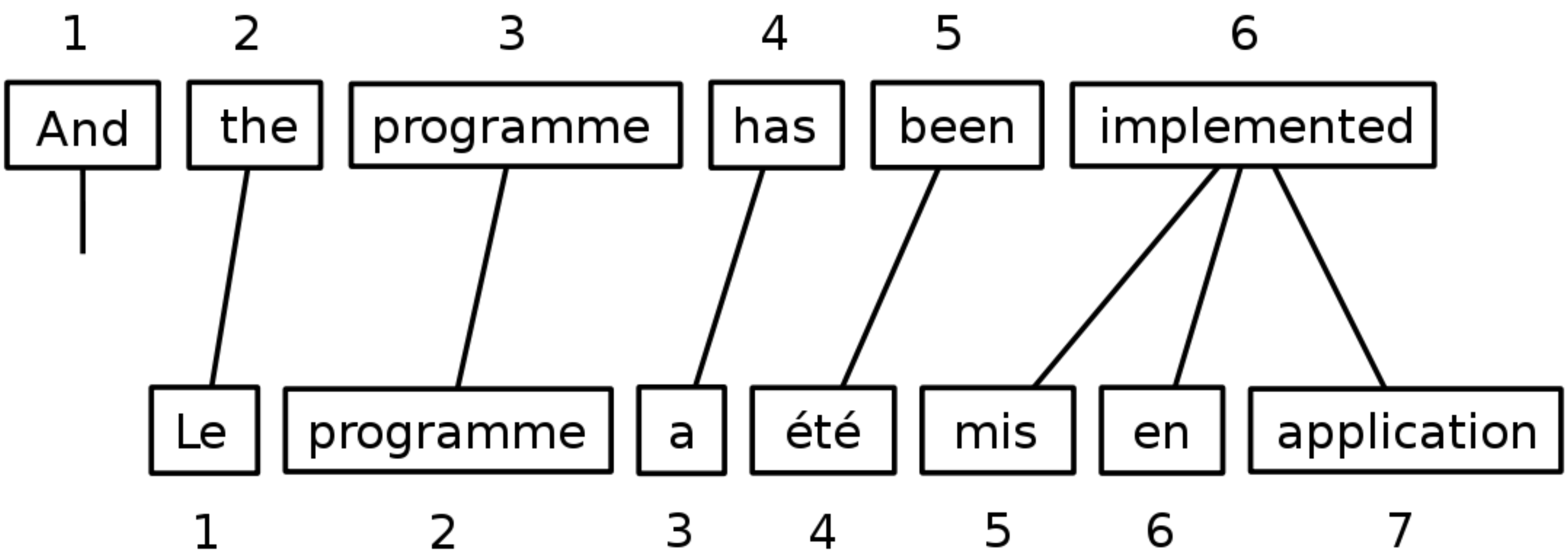
Agent Predicate Theme Location

input: a sentence

output: one span in the sentence identified as a *predicate*, and a set of other spans identified as particular *roles* for that predicate

Supervised Word Alignment

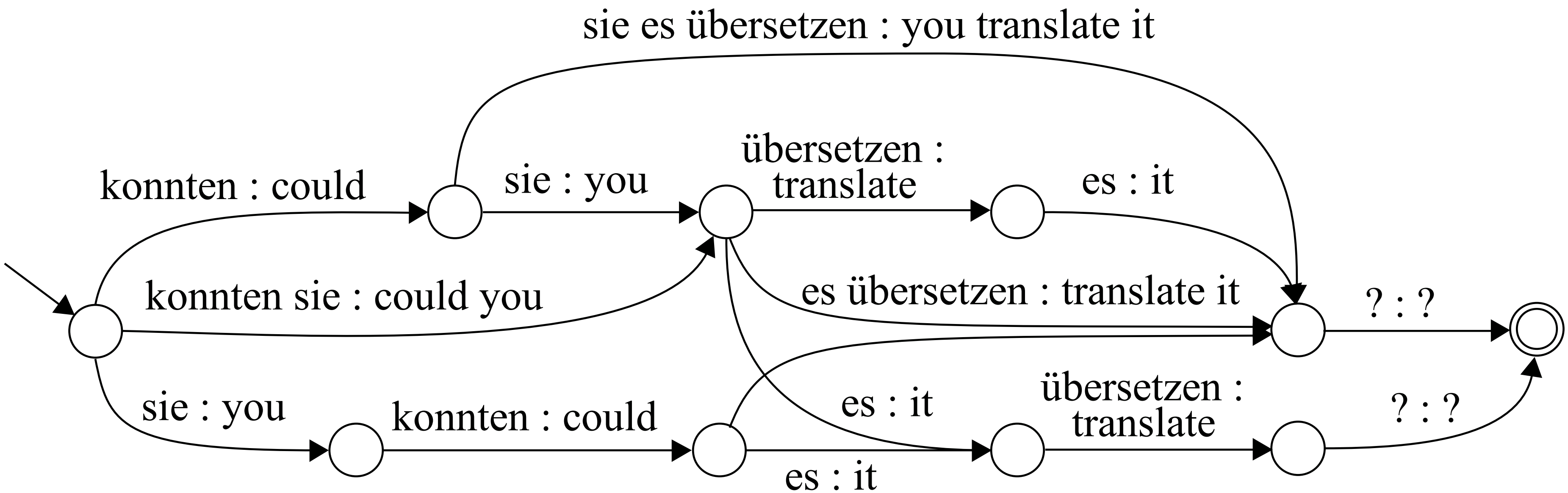
given parallel sentences, predict word alignments:



Brown et al. (1990)

Machine Translation

- phrase-based model (Koehn et al., 2003):



input: a sentence in the source language

output: a segmentation of the source sentence into segments, a translation of each segment, and an ordering of the translations

Key Categories of Structured Prediction

- I think of structured prediction methods in two primary categories:

  score-based and search-based

Score-Based Structured Prediction

- focus on defining the score function of the structured input/output pair:

$$\mathrm{score}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta})$$

- in dependency parsing, this is called "graph-based parsing" because minimum spanning tree algorithms can be used to find the globally-optimal max-scoring tree

Search-Based Structured Prediction

- focus on the procedure for searching through the structured output space (usually involves simple greedy or beam search)
- design a classifier to score a small number of decisions at each position in the search
  - this classifier can use information about the current state as well as the entire history of the search

- in dependency parsing, this is called "transition-based parsing" because it consists of greedily, sequentially deciding what parsing decision to make

Structured Prediction

- to make SP practical, we need to decompose the SP problem into **parts**
- this is true whether we are going to use search-based or score-based SP
  – score-based: score function decomposes additively into scores of parts
  – search-based: search factors into a sequence of decisions, each one adding a part to the final output structure