10-301/10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Hidden Markov Models (Part II)

Matt Gormley
Lecture 19
Mar. 27, 2023

Reminders

- Practice Problems: Exam 2
  - Out: Fri, Mar. 24
- Exam 2
  - Thu, Mar. 30, 6:30pm – 8:30pm
- Homework 7: Hidden Markov Models
  - Out: Fri, Mar. 31
  - Due: Mon, Apr. 10 at 11:59pm

SUPERVISED LEARNING FOR HMMS

Recipe for Closed-form MLE

1. Assume data was generated i.i.d. from some model (i.e. write the generative story)

   $x^{(i)} \sim p(x|\boldsymbol{\theta})$

2. Write log-likelihood

   $\ell(\boldsymbol{\theta}) = \log p(x^{(1)}|\boldsymbol{\theta}) + \ldots + \log p(x^{(N)}|\boldsymbol{\theta})$

3. Compute partial derivatives (i.e. gradient)

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_1 = \ldots$

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_2 = \ldots$

   …

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_M = \ldots$

4. Set derivatives to zero and solve for $\boldsymbol{\theta}$

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_m = 0$ for all $m \in \{1, \ldots, M\}$

   $\boldsymbol{\theta}^{MLE}$ = solution to system of M equations and M variables

5. Compute the second derivative and check that $\ell(\boldsymbol{\theta})$ is concave down at $\boldsymbol{\theta}^{MLE}$

MLE of Categorical Distribution

1. Suppose we have a **dataset** obtained by repeatedly rolling a M-sided (weighted) die N times. That is, we have data

$$\mathcal{D} = \{x^{(i)}\}_{i=1}^{N}$$

where $x^{(i)} \in \{1, \ldots, M\}$ and $x^{(i)} \sim \text{Categorical}(\boldsymbol{\phi})$.

2. A random variable is **Categorical** written $X \sim \text{Categorical}(\boldsymbol{\phi})$ iff

$$P(X = x) = p(x; \boldsymbol{\phi}) = \phi_x$$

where $x \in \{1, \ldots, M\}$ and $\sum_{m=1}^{M} \phi_m = 1$. The **log-likelihood** of the data becomes:

$$\ell(\boldsymbol{\phi}) = \sum_{i=1}^{N} \log \phi_{x^{(i)}} \text{ s.t. } \sum_{m=1}^{M} \phi_m = 1$$

3. Solving this *constrained* optimization problem yields the **maximum likelihood estimator** (MLE):

$$\phi_m^{MLE} = \frac{N_{x=m}}{N} = \frac{\sum_{i=1}^{N} \mathbb{I}(x^{(i)} = m)}{N}$$

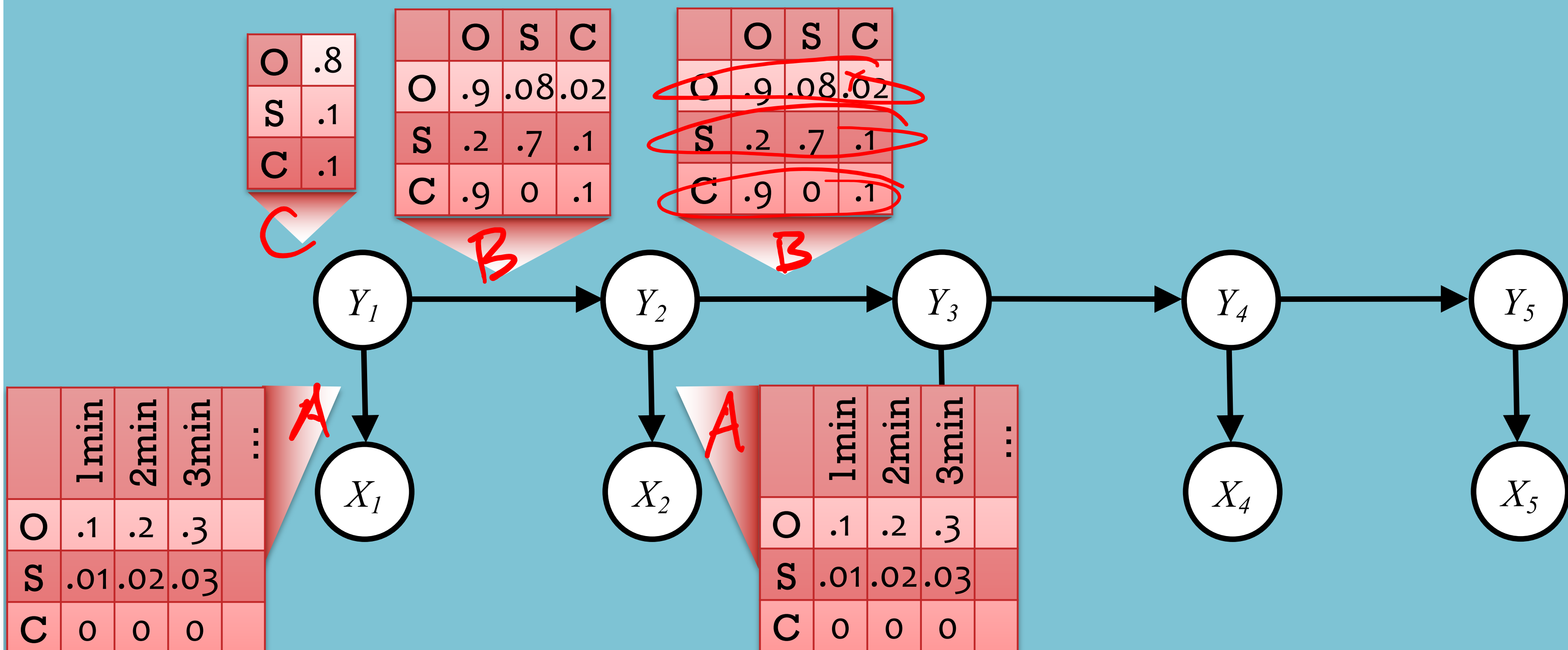
Hidden Markov Model (v1)

HMM Parameters:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

Initial probs, $\mathbf{C}$, where $P(Y_1 = k) = C_k, \forall k$



O	.8
S	.1
C	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1

	1min	2min	3min	…
O	.1	.2	.3	
S	.01	.02	.03	
C	0	0	0	

	1min	2min	3min	…
O	.1	.2	.3	
S	.01	.02	.03	
C	0	0	0	

$$P(\mathbf{X}, \mathbf{Y}) = P(Y_1) \left(\prod_{t=1}^{T} P(X_t | Y_t) \right) \left(\prod_{t=2}^{T} p(Y_t | Y_{t-1}) \right)$$

Hidden Markov Model (v1)

HMM Parameters:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

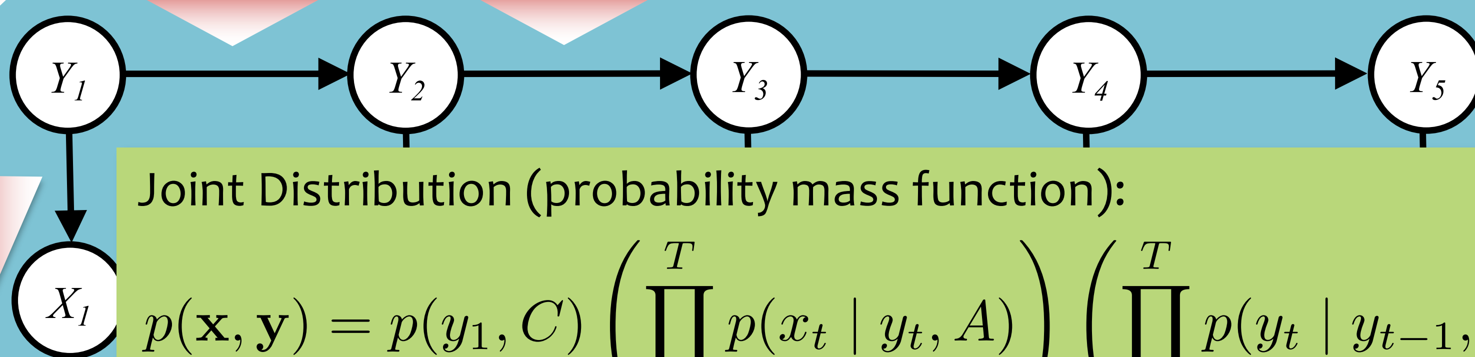
Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

Initial probs, $\mathbf{C}$, where $P(Y_1 = k) = C_k, \forall k$

O	.8
S	.1
C	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1

	O	S	C
O	.9	.08	.02
S	.2	.7	.1
C	.9	0	.1



	1min	2min	3min	…
O	.1	.2	.3	
S	.01	.02	.03	
C	0	0	0	

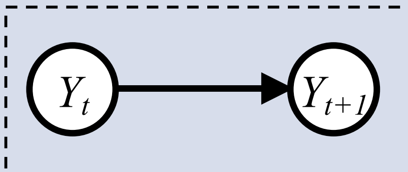
Joint Distribution (probability mass function):

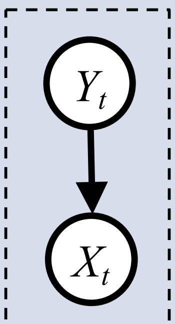
$$p(\mathbf{x}, \mathbf{y}) = p(y_1, C) \left(\prod_{t=1}^{T} p(x_t \mid y_t, A) \right) \left(\prod_{t=2}^{T} p(y_t \mid y_{t-1}, B) \right)$$

$$= C_{y_1} \left(\prod_{t=1}^{T} A_{y_t, x_t} \right) \left(\prod_{t=2}^{T} B_{y_{t-1}, y_t} \right)$$

Supervised Learning for HMM (v1)

Learning an HMM decomposes into solving two (independent) Mixture Models





Data: $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$ where $\mathbf{x} = [x_1, \ldots, x_T]^T$ and $\mathbf{y} = [y_1, \ldots, y_T]^T$

Likelihood:

$$\ell(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_{i=1}^N \log p(\mathbf{x}^{(i)}, \mathbf{y}^{(i)} \mid \mathbf{A}, \mathbf{B}, \mathbf{C})$$

$$= \sum_{i=1}^N \left[\underbrace{\log p(y_1^{(i)} \mid \mathbf{C})}_{\text{initial}} + \underbrace{\left(\sum_{t=2}^T \log p(y_t^{(i)} \mid y_{t-1}^{(i)}, \mathbf{B}) \right)}_{\text{transition}} + \underbrace{\left(\sum_{t=1}^T \log p(x_t^{(i)} \mid y_t^{(i)}, \mathbf{A}) \right)}_{\text{emission}} \right]$$

MLE:

$$\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}} = \underset{\mathbf{A},\mathbf{B},\mathbf{C}}{\text{argmax}}\, \ell(\mathbf{A}, \mathbf{B}, \mathbf{C})$$

$$\Rightarrow \hat{\mathbf{C}} = \underset{\mathbf{C}}{\text{argmax}} \sum_{i=1}^N \log p(y_1^{(i)} \mid \mathbf{C})$$

$$\hat{\mathbf{B}} = \underset{\mathbf{B}}{\text{argmax}} \sum_{i=1}^N \sum_{t=2}^T \log p(y_t^{(i)} \mid y_{t-1}^{(i)}, \mathbf{B})$$

$$\hat{\mathbf{A}} = \underset{\mathbf{A}}{\text{argmax}} \sum_{i=1}^N \sum_{t=1}^T \log p(x_t^{(i)} \mid y_t^{(i)}, \mathbf{A})$$

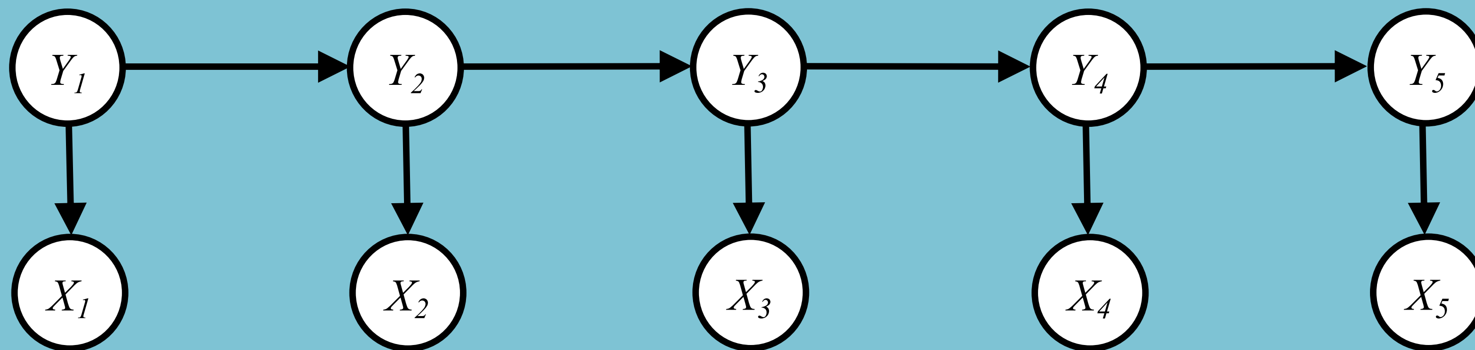
We can solve the above in closed form, which yields...

$$\hat{C}_k = \frac{\#(y_1^{(i)} = k)}{N}, \forall k$$

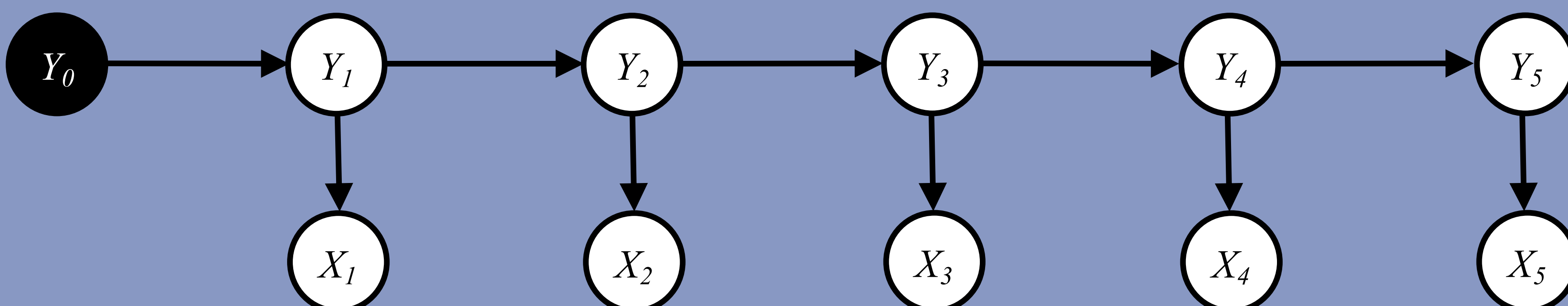
$$\hat{B}_{j,k} = \frac{\#(y_t^{(i)} = k \text{ and } y_{t-1}^{(i)} = j)}{\#(y_{t-1}^{(i)} = j)}, \forall j, k$$

$$\hat{A}_{j,k} = \frac{\#(x_t^{(i)} = k \text{ and } y_t^{(i)} = j)}{\#(y_t^{(i)} = j)}, \forall j, k$$

HMM (two ways)



HMM (v1): $$P(\mathbf{X}, \mathbf{Y}) = P(Y_1)\left(\prod_{t=1}^{T} P(X_t|Y_t)\right)\left(\prod_{t=2}^{T} p(Y_t|Y_{t-1})\right)$$



HMM (v2): $$P(\mathbf{X}, \mathbf{Y}|Y_0) = \prod_{t=1}^{T} P(X_t|Y_t)p(Y_t|Y_{t-1})$$

Hidden Markov Model (v2)

HMM Parameters:

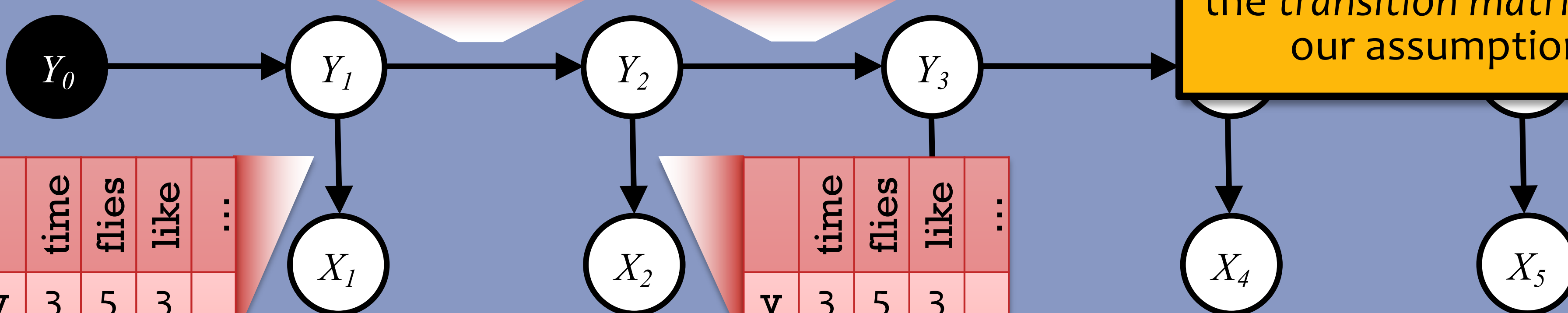
Emission matrix, $\mathbf{A}$, where $P(X_k = w | Y_k = t) = A_{t,w}, \forall k$

Transition matrix, $\mathbf{B}$, where $P(Y_k = t | Y_{k-1} = s) = B_{s,t}, \forall k$

	O	S	C	Start
O	.9	.08	.02	0
S	.2	.7	.1	0
C	.9	0	.1	0
Start	0.8	0.1	0.1	0

	O	S	C	Start
O	.9	.08	.02	0
S	.2	.7	.1	0
C	.9	0	.1	0
Start	0.8	0.1	0.1	0

For notational convenience, we fold the *initial probabilities* **C** into the *transition matrix* **B** by our assumption.



	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

	time	flies	like	…
v	3	5	3	
n	4	5	2	
p	0.1	0.1	3	
d	0.1	0.2	0.1	

Hidden Markov Model (v2)

HMM Parameters:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

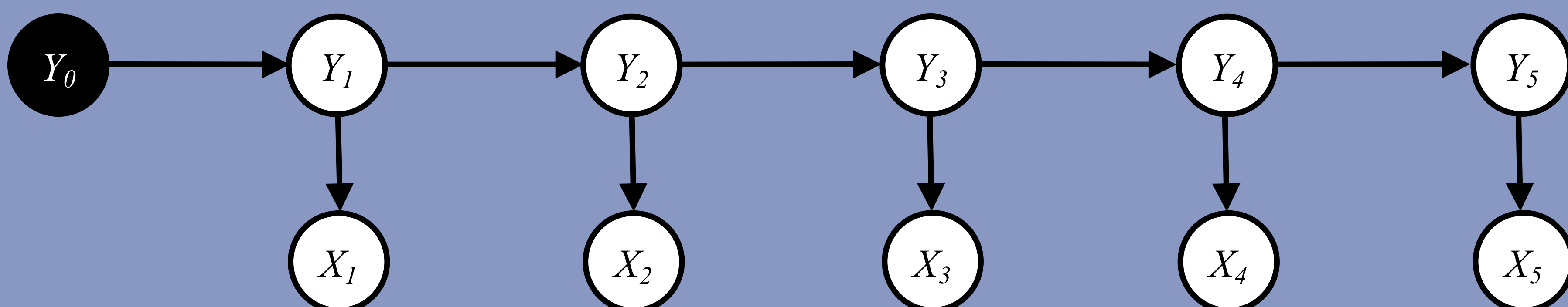
Assumption: $y_0 =$ START

Generative Story:

$$Y_t \sim \text{Multinomial}(\mathbf{B}_{Y_{t-1}}) \ \forall t$$

$$X_t \sim \text{Multinomial}(\mathbf{A}_{Y_t}) \ \forall t$$

For notational convenience, we fold the *initial probabilities* **C** into the *transition matrix* **B** by our assumption.



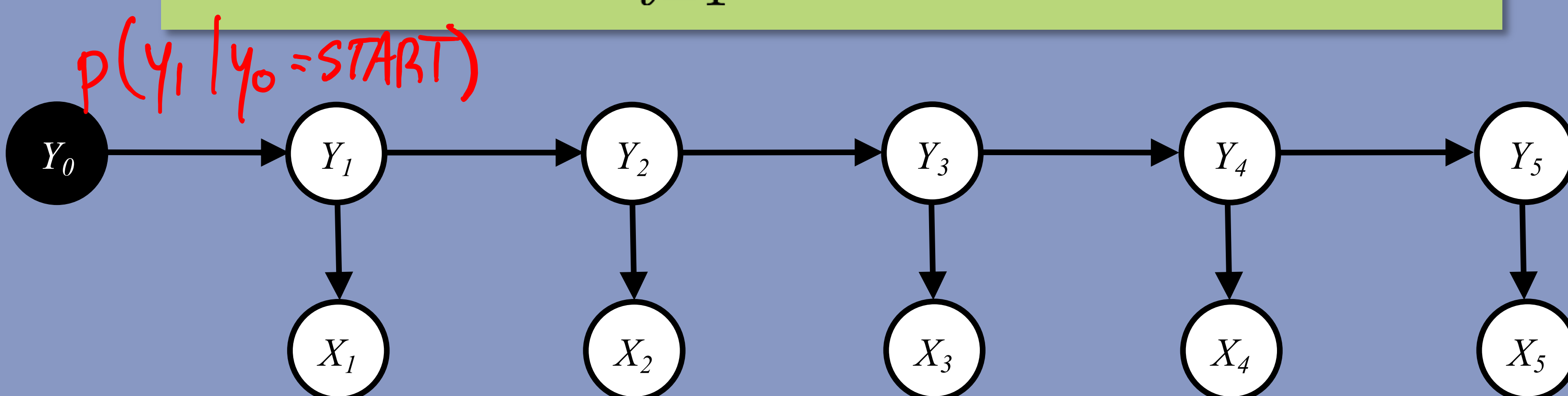
Hidden Markov Model (v2)

Joint Distribution (probability mass function):

$$y_0 = \text{START}$$

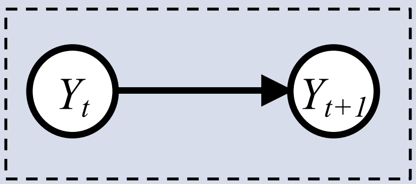
$$p(\mathbf{x}, \mathbf{y} | y_0) = \prod_{t=1}^{T} p(x_t | y_t) p(y_t | y_{t-1})$$

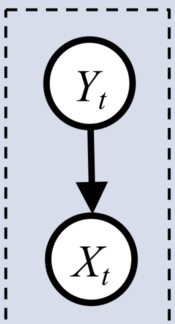
$$= \prod_{t=1}^{T} A_{y_t, x_t} B_{y_{t-1}, y_t}$$

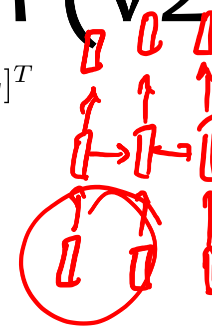


Supervised Learning for HMM (v2)

Learning an HMM decomposes into solving two (independent) Mixture Models







Data: $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ where $\mathbf{x} = [x_1, \ldots, x_T]^T$ and $\mathbf{y} = [y_1, \ldots, y_T]^T$

We assume $y_0^{(i)} = \text{START}$ for all i

Likelihood:

$$\ell(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}, \mathbf{y}^{(i)} \mid \mathbf{A}, \mathbf{B})$$

$$= \sum_{i=1}^{N} \left[\sum_{t=1}^{T} \underbrace{\log p(y_t^{(i)} \mid y_{t-1}^{(i)}, \mathbf{B})}_{\text{transition}} + \underbrace{\log p(x_t^{(i)} \mid y_t^{(i)}, \mathbf{A})}_{\text{emission}} \right]$$

MLE:

$$\hat{\mathbf{A}}, \hat{\mathbf{B}} = \underset{\mathbf{A},\mathbf{B},\mathbf{C}}{\text{argmax}}\, \ell(\mathbf{A}, \mathbf{B})$$

$$\Rightarrow \hat{\mathbf{B}} = \underset{\mathbf{B}}{\text{argmax}} \sum_{i=1}^{N} \sum_{t=1}^{T} \log p(y_t^{(i)} \mid y_{t-1}^{(i)}, \mathbf{B})$$

$$\hat{\mathbf{A}} = \underset{\mathbf{A}}{\text{argmax}} \sum_{i=1}^{N} \sum_{t=1}^{T} \log p(x_t^{(i)} \mid y_t^{(i)}, \mathbf{A})$$

We can solve the above in closed form, which yields...

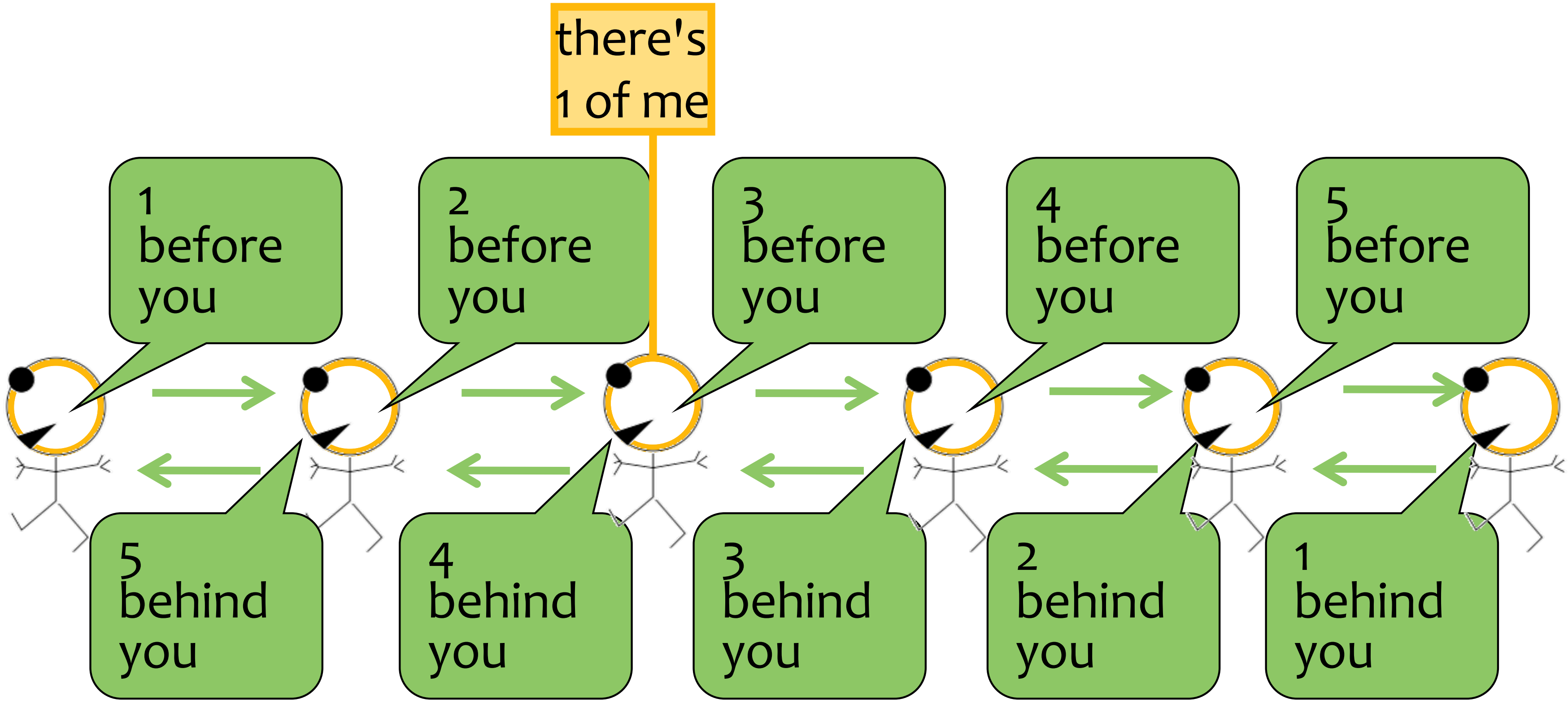
$$\hat{B}_{j,k} = \frac{\#(y_t^{(i)} = k \text{ and } y_{t-1}^{(i)} = j)}{\#(y_{t-1}^{(i)} = j)}, \ \forall j, k$$

$$\hat{A}_{j,k} = \frac{\#(x_t^{(i)} = k \text{ and } y_t^{(i)} = j)}{\#(y_t^{(i)} = j)}, \ \forall j, k$$

BACKGROUND: MESSAGE PASSING

Great Ideas in ML: Message Passing

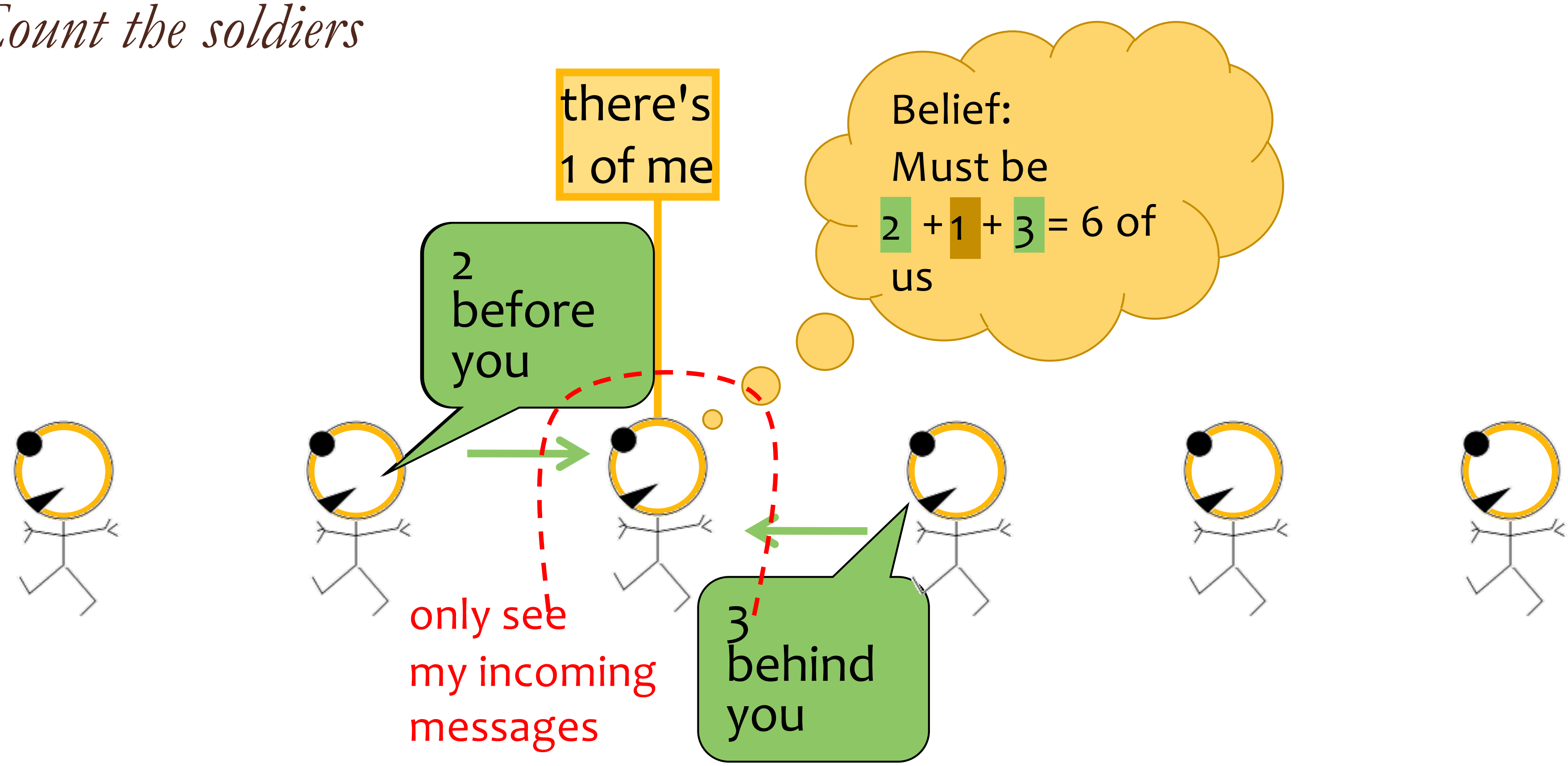
Count the soldiers



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

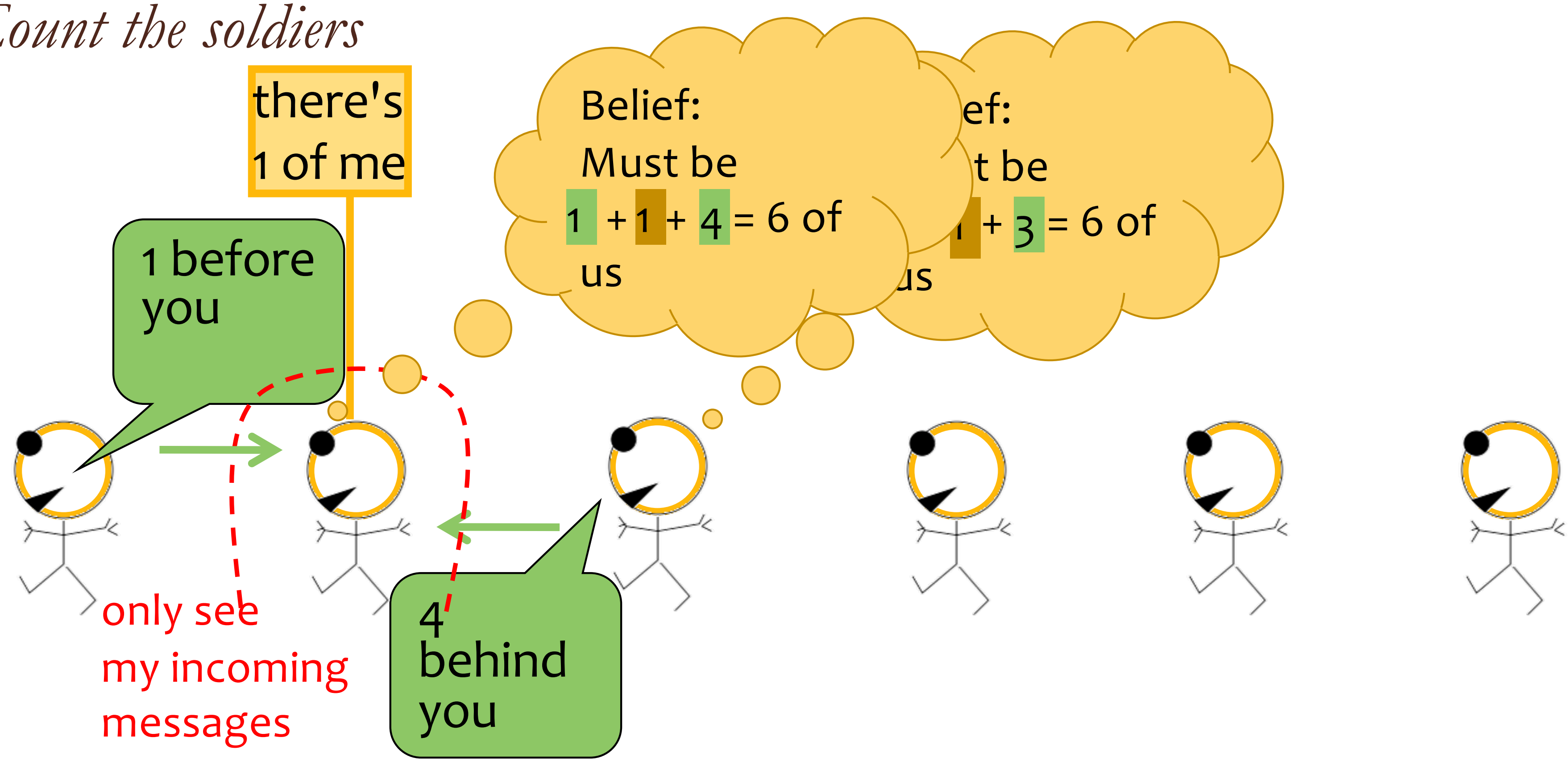
Count the soldiers



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

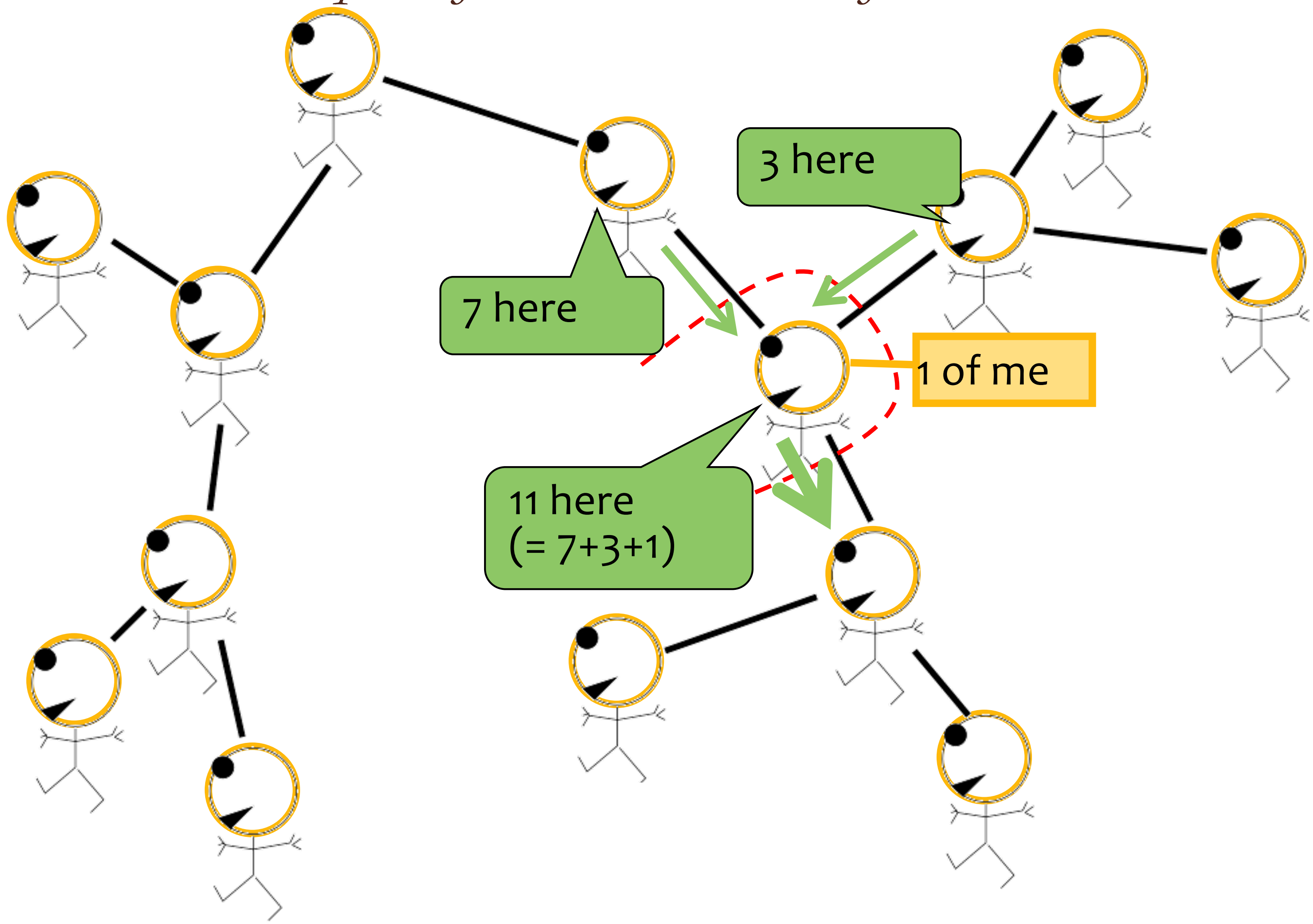
Count the soldiers



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

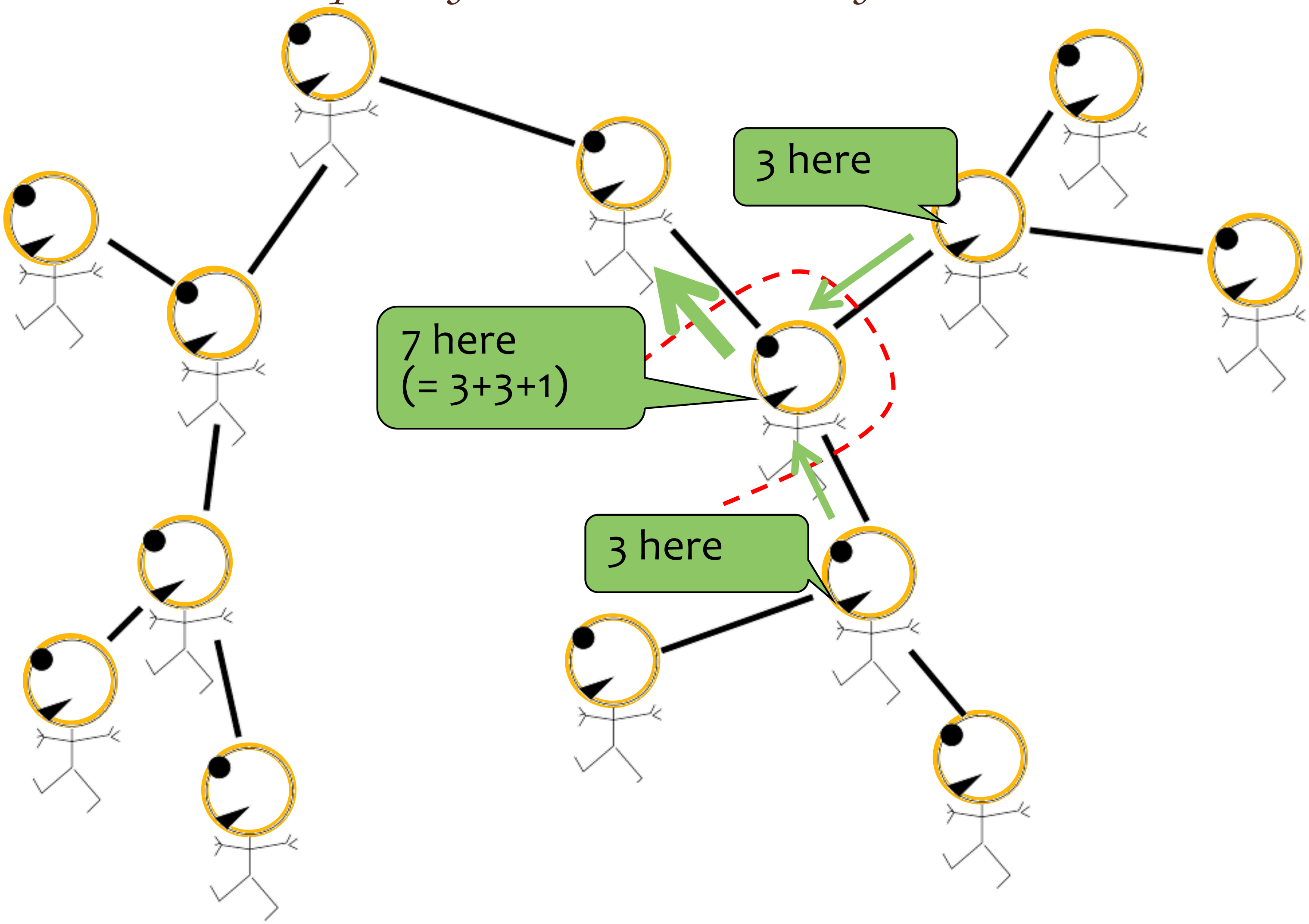
Each soldier receives reports from all branches of tree



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

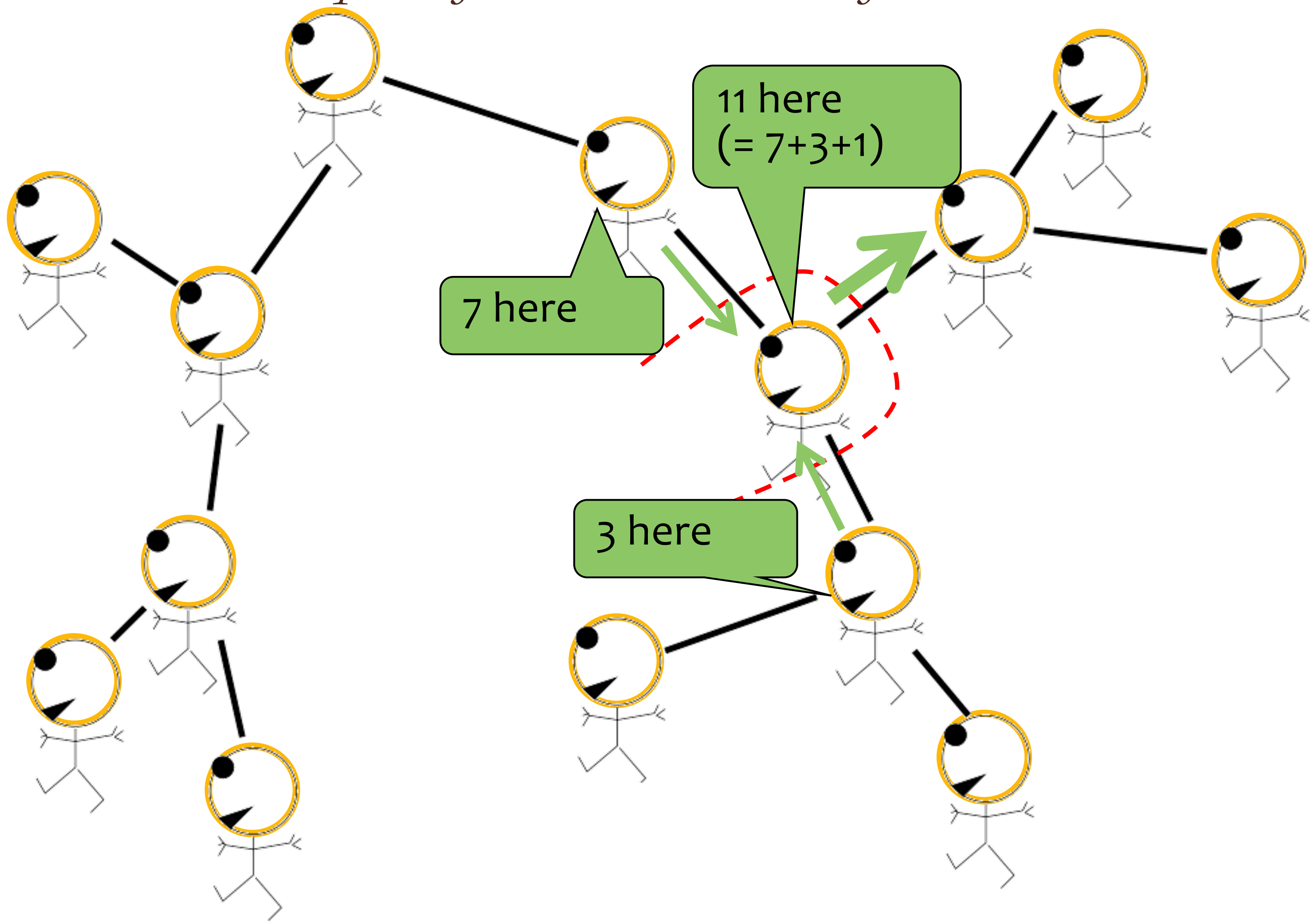
Each soldier receives reports from all branches of tree



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

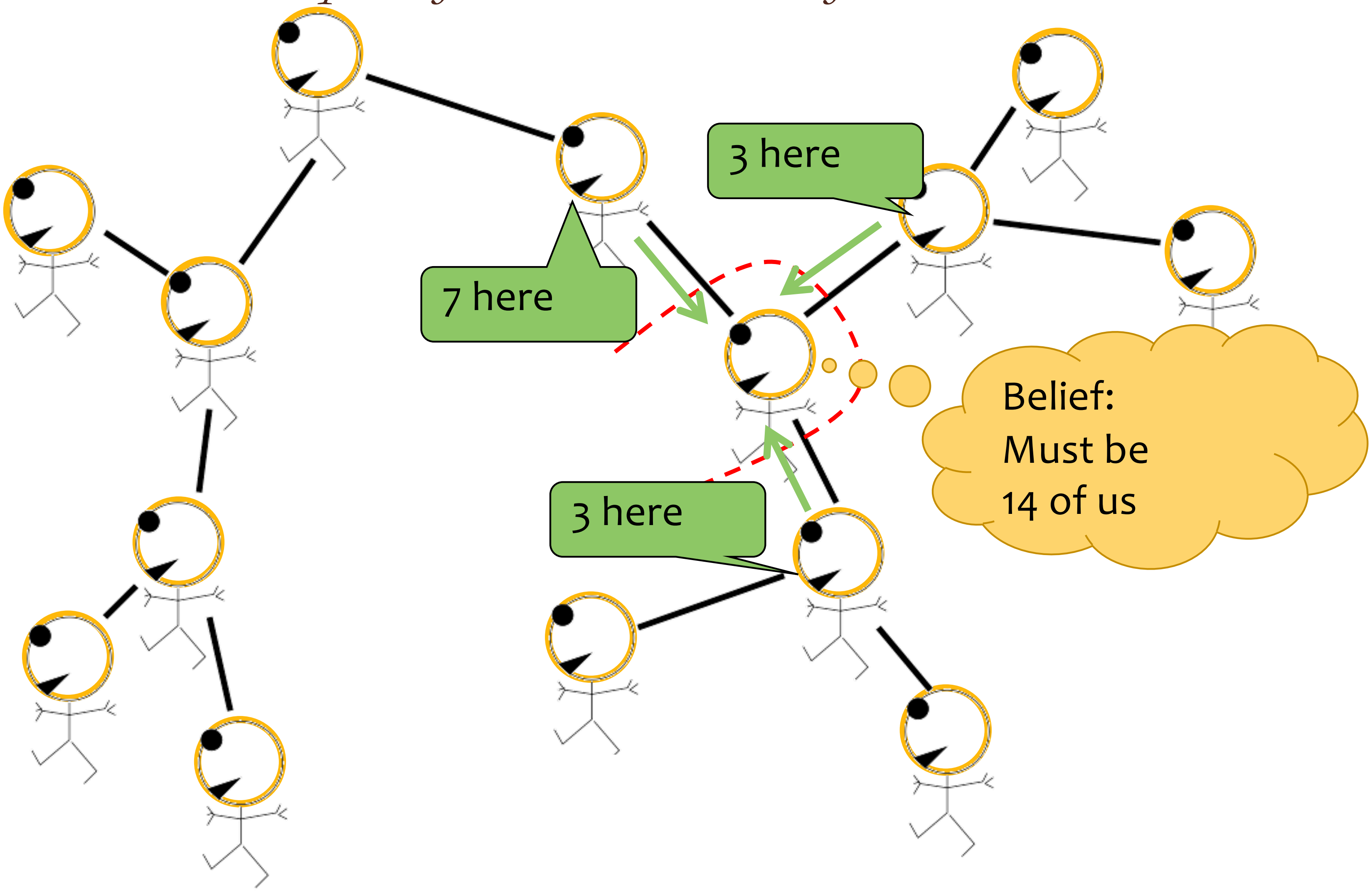
Each soldier receives reports from all branches of tree



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

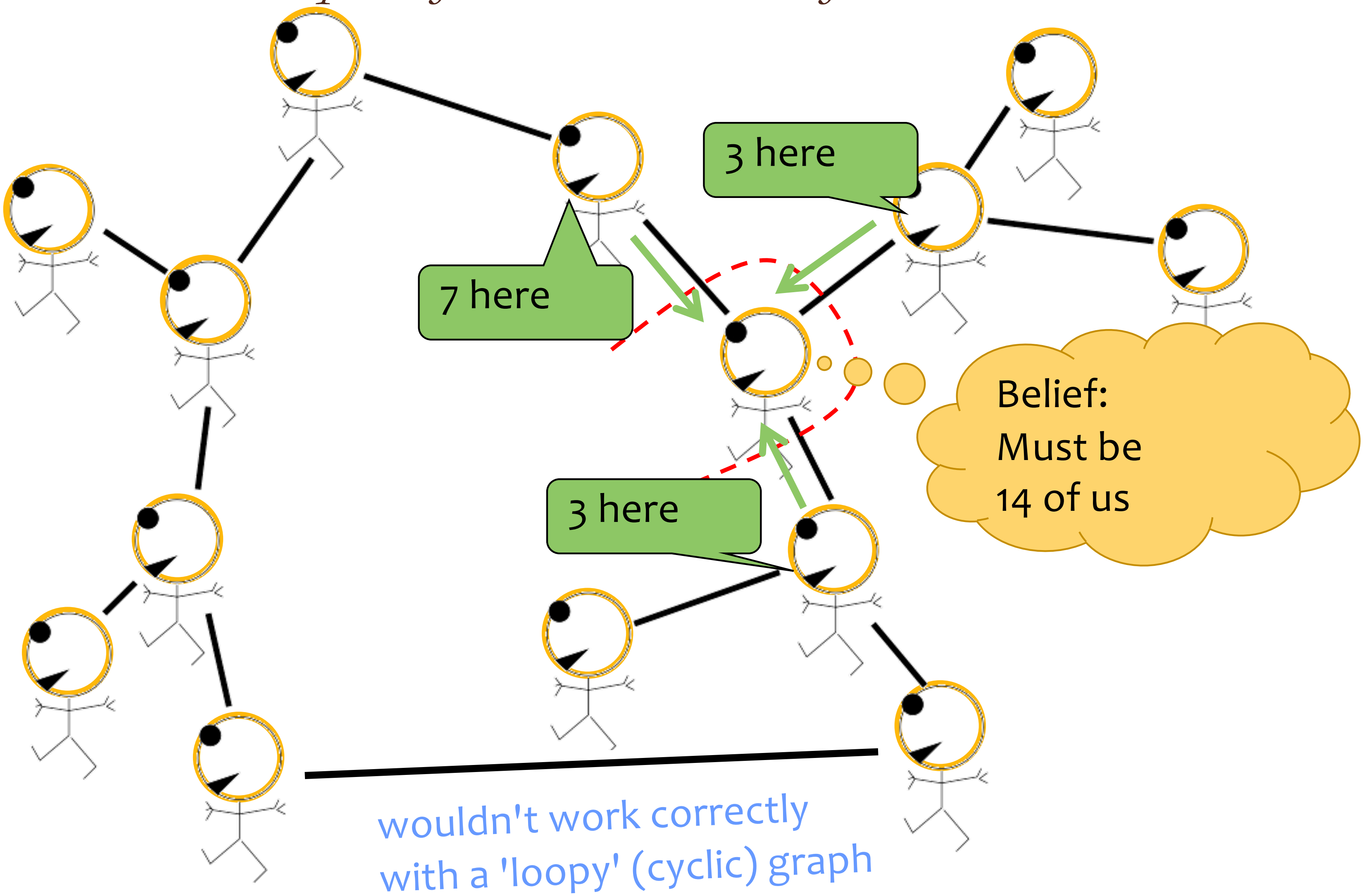
Each soldier receives reports from all branches of tree



adapted from MacKay (2003) textbook

Great Ideas in ML: Message Passing

Each soldier receives reports from all branches of tree



adapted from MacKay (2003) textbook

INFERENCE FOR HMMS

Inference

67%

Question: Q1 A = True B = toxic C = False

True or False: The **joint probability of the observations and the hidden states** in an HMM is given by:

$$P(\mathbf{X} = \mathbf{x}, \mathbf{Y} = \mathbf{y}) = C_{y_1} \left[\prod_{t=1}^{T} A_{y_t, x_t} \right] \left[\prod_{t=1}^{T-1} B_{y_t, y_{t+1}} \right]$$

Recall:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

Initial probs, $\mathbf{C}$, where $P(Y_1 = k) = C_k, \forall k$

Inference

Question:

True or False: The **probability of the observations** in an HMM is given by:

$$P(\mathbf{X} = \mathbf{x}) = \prod_{t=1}^{T} A_{x_t, x_{t-1}}$$

Recall:

Emission matrix, $\mathbf{A}$, where $P(X_t = k | Y_t = j) = A_{j,k}, \forall t, k$

Transition matrix, $\mathbf{B}$, where $P(Y_t = k | Y_{t-1} = j) = B_{j,k}, \forall t, k$

Initial probs, $\mathbf{C}$, where $P(Y_1 = k) = C_k, \forall k$

Inference for HMMs

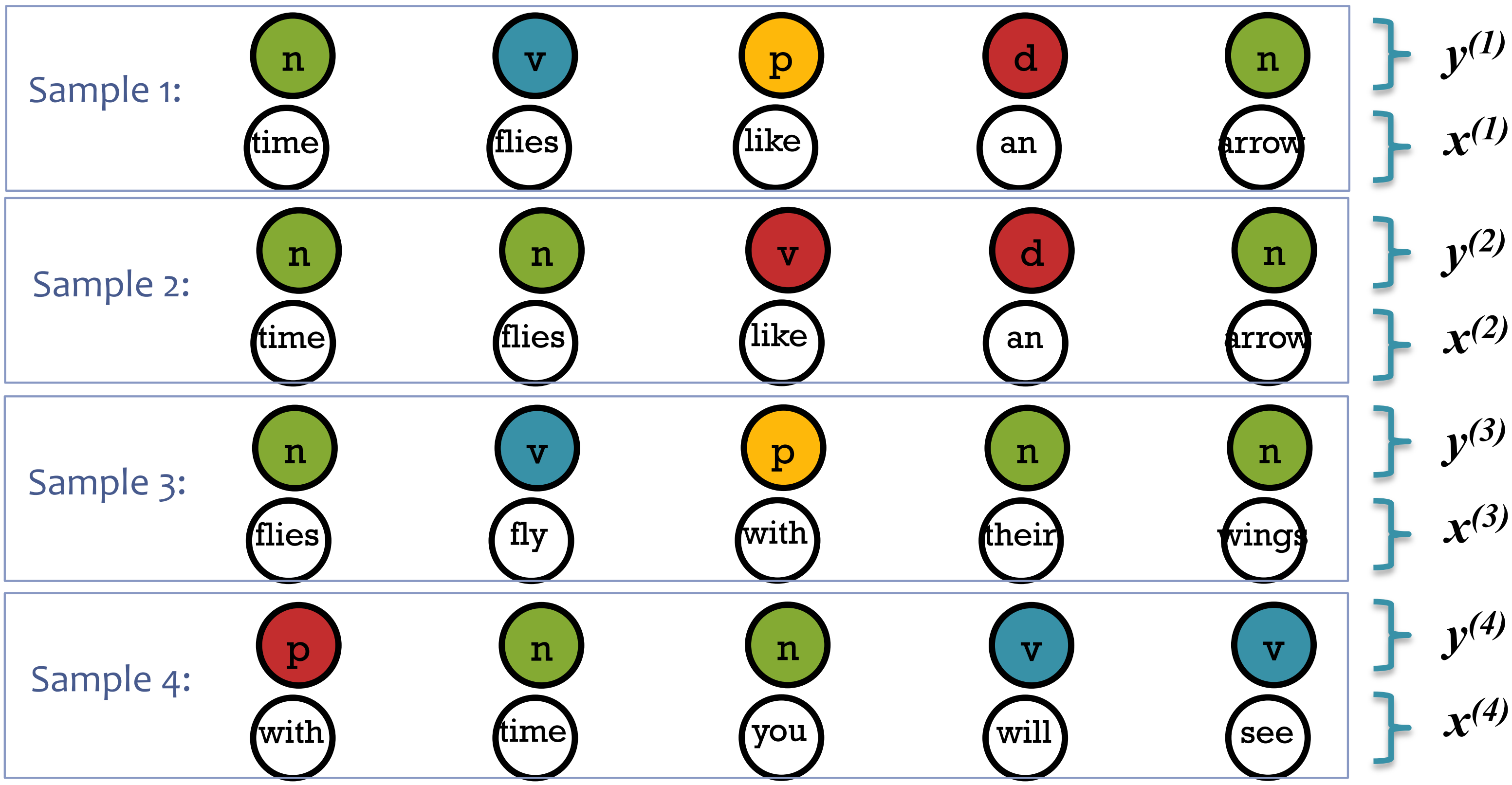
Whiteboard

- Three Inference Problems for an HMM
  1. Evaluation: Compute the probability of a given sequence of observations
  2. Viterbi Decoding: Find the most-likely sequence of hidden states, given a sequence of observations
  3. Marginals: Compute the marginal distribution for a hidden state, given a sequence of observations

THE SEARCH SPACE FOR FORWARD-BACKWARD

Dataset for Supervised Part-of-Speech (POS) Tagging

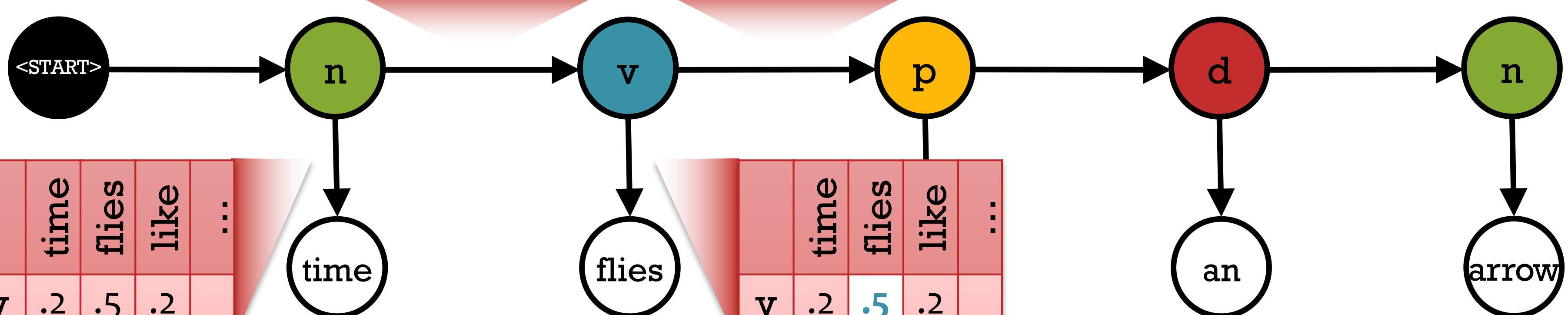
Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$



Example: HMM for POS Tagging

A Hidden Markov Model (HMM) provides a joint distribution over the the sentence/tags with an assumption of dependence between adjacent tags.

$$p(\text{n, v, p, d, n, time, flies, like, an, arrow}) = (.3 * .8 * .2 * .5 * \ldots)$$



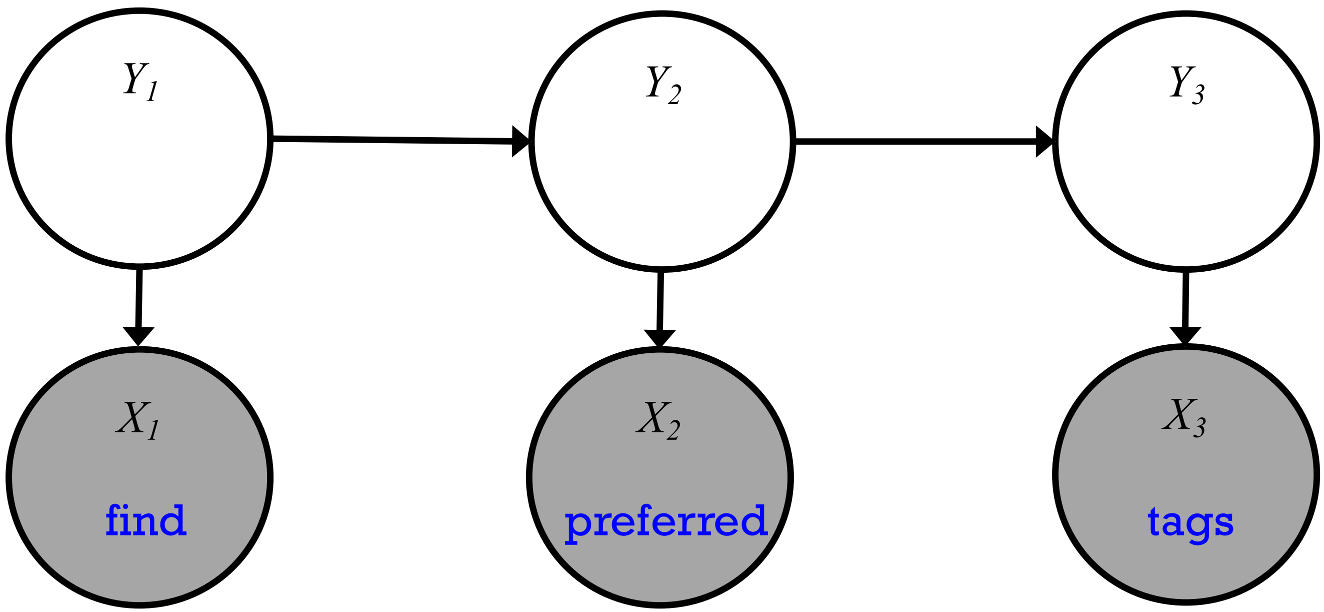
	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

	v	n	p	d
v	.1	.4	.2	.3
n	.8	.1	.1	0
p	.2	.3	.2	.3
d	.2	.8	0	0

	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

	time	flies	like	…
v	.2	.5	.2	
n	.3	.4	.2	
p	.1	.1	.3	
d	.1	.2	.1	

Example: HMM for POS Tagging



Could be verb or noun *Could be adjective or verb* *Could be noun or verb*

Inference for HMMs

Whiteboard

- Brute Force Evaluation
- Forward-backward search space

THE FORWARD-BACKWARD ALGORITHM

How is efficient computation even possible?

- The short answer is **dynamic programming**!

- The key idea is this:
  - We first come up with a **recursive definition** for the quantity we want to compute
  - We then observe that many of the recursive intermediate terms are **reused** across timesteps and tags
  - We then perform **bottom-up dynamic programming** by running the recursion in reverse, **storing the intermediate quantities** along the way!

- This enables us to search the **exponentially large** space in **polynomial time**!

Inference for HMMs

Whiteboard

- Forward-backward algorithm
  (edge weights version)

Forward-Backward Algorithm

Definitions

$\alpha_t(k) \triangleq p(x_1, \ldots, x_t, y_t = k)$

$\beta_t(k) \triangleq p(x_{t+1}, \ldots, x_T \mid y_t = k)$

Assume

$y_0 = \text{START}$

$y_{T+1} = \text{END}$

1. Initialize

$$\alpha_0(\text{START}) = 1 \qquad \alpha_0(k) = 0, \ \forall k \neq \text{START}$$

$$\beta_{T+1}(\text{END}) = 1 \qquad \beta_{T+1}(k) = 0, \ \forall k \neq \text{END}$$

2. Forward Algorithm

for $t = 1, \ldots, T+1$:

for $k = 1, \ldots, K$:

$$\alpha_t(k) = \sum_{j=1}^{K} p(x_t \mid y_t = k)\alpha_{t-1}(j)p(y_t = k \mid y_{t-1} = j)$$

3. Backward Algorithm

for $t = T, \ldots, 0$:

for $k = 1, \ldots, K$:

$$\beta_t(k) = \sum_{j=1}^{K} p(x_{t+1} \mid y_{t+1} = j)\beta_{t+1}(j)p(y_{t+1} = j \mid y_t = k)$$

4. Evaluation $p(\mathbf{x}) = \alpha_{T+1}(\text{END})$

5. Marginals $p(y_t = k \mid \mathbf{x}) = \frac{\alpha_t(k)\beta_t(k)}{p(\mathbf{x})}$

Forward-Backward Algorithm

Definitions

$\alpha_t(k) \triangleq p(x_1, \ldots, x_t, y_t = k)$

$\beta_t(k) \triangleq p(x_{t+1}, \ldots, x_T \mid y_t = k)$

Assume

$y_0 = \text{START}$

$y_{T+1} = \text{END}$

1. Initialize

$\alpha_0(\text{START}) = 1 \qquad \alpha_0(k) = 0, \forall k \neq \text{START}$

$\beta_{T+1}(\text{END}) = 1 \qquad \beta_{T+1}(k) = 0, \forall k \neq \text{END}$

2. Forward Algorithm

for $t = 1, \ldots, T+1$:

for $k = 1, \ldots, K$:

$$\alpha_t(k) = \sum_{j=1}^{K} p(x_t \mid y_t = k)\alpha_{t-1}(j)p(y_t = k \mid y_{t-1} = j)$$

O(K²T)

O(K)

kward Algorithm

for $t = T, \ldots, 0$:

or $k = 1, \ldots, K$:

$$\beta_t(k) = \sum_{j=1}^{K} p(x_{t+1} \mid y_{t+1} = j)\beta_{t+1}(j)p(y_{t+1} = j \mid y_t = k)$$

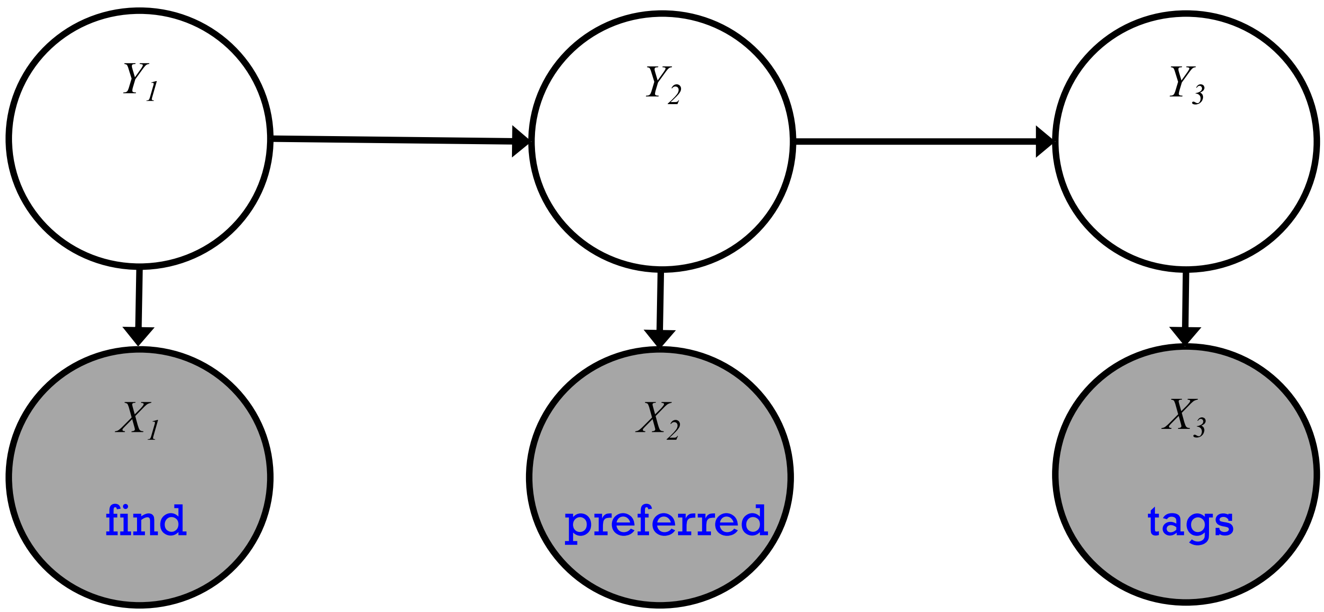
Brute force algorithm would be O(K^T)

4. Evaluation $p(\mathbf{x}) = \alpha_{T+1}(\text{END})$

5. Marginals $p(y_t = k \mid \mathbf{x}) = \frac{\alpha_t(k)\beta_t(k)}{p(\mathbf{x})}$

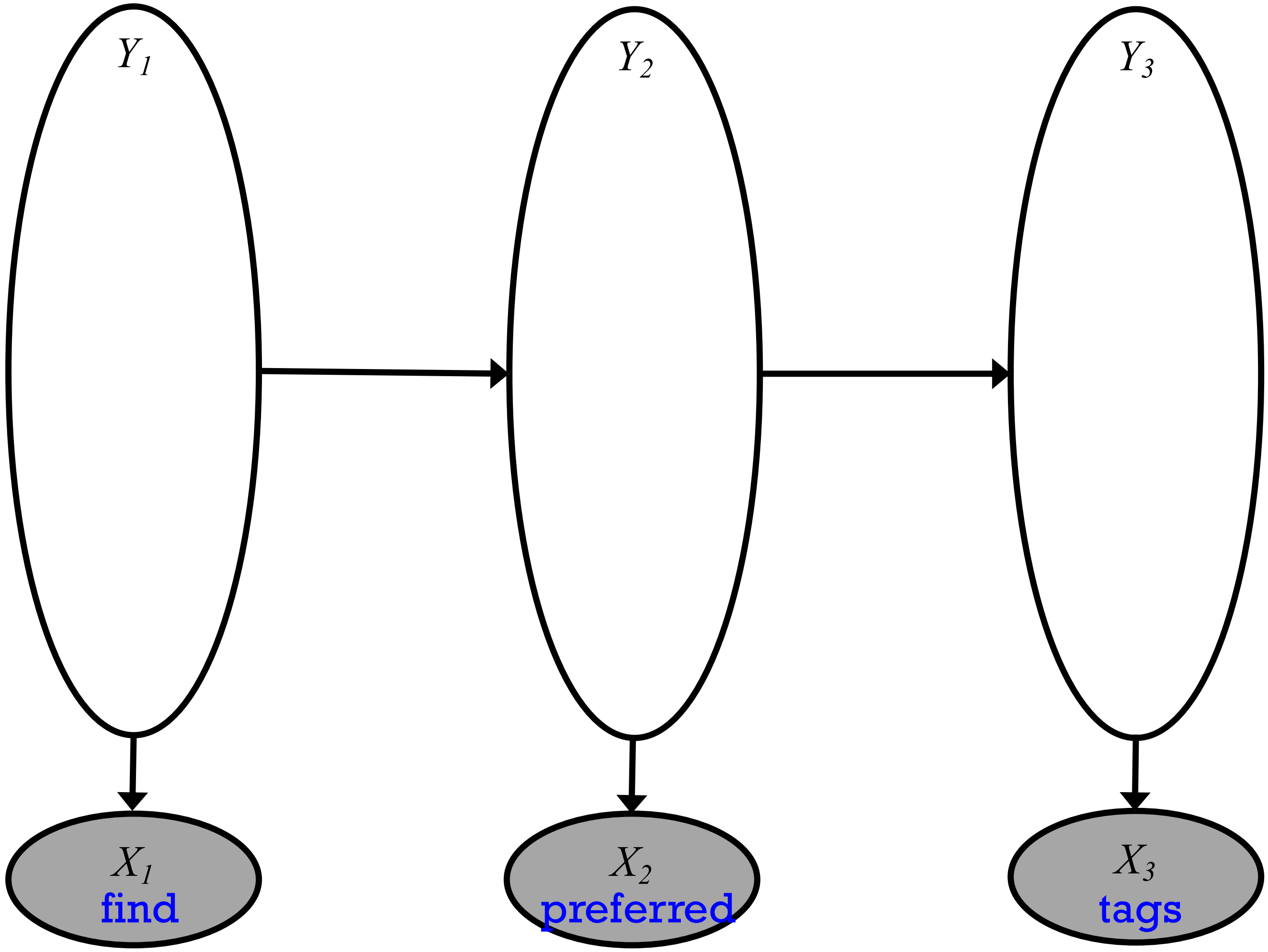
EXAMPLE: FORWARD-BACKWARD ON THREE WORDS

Forward-Backward Algorithm

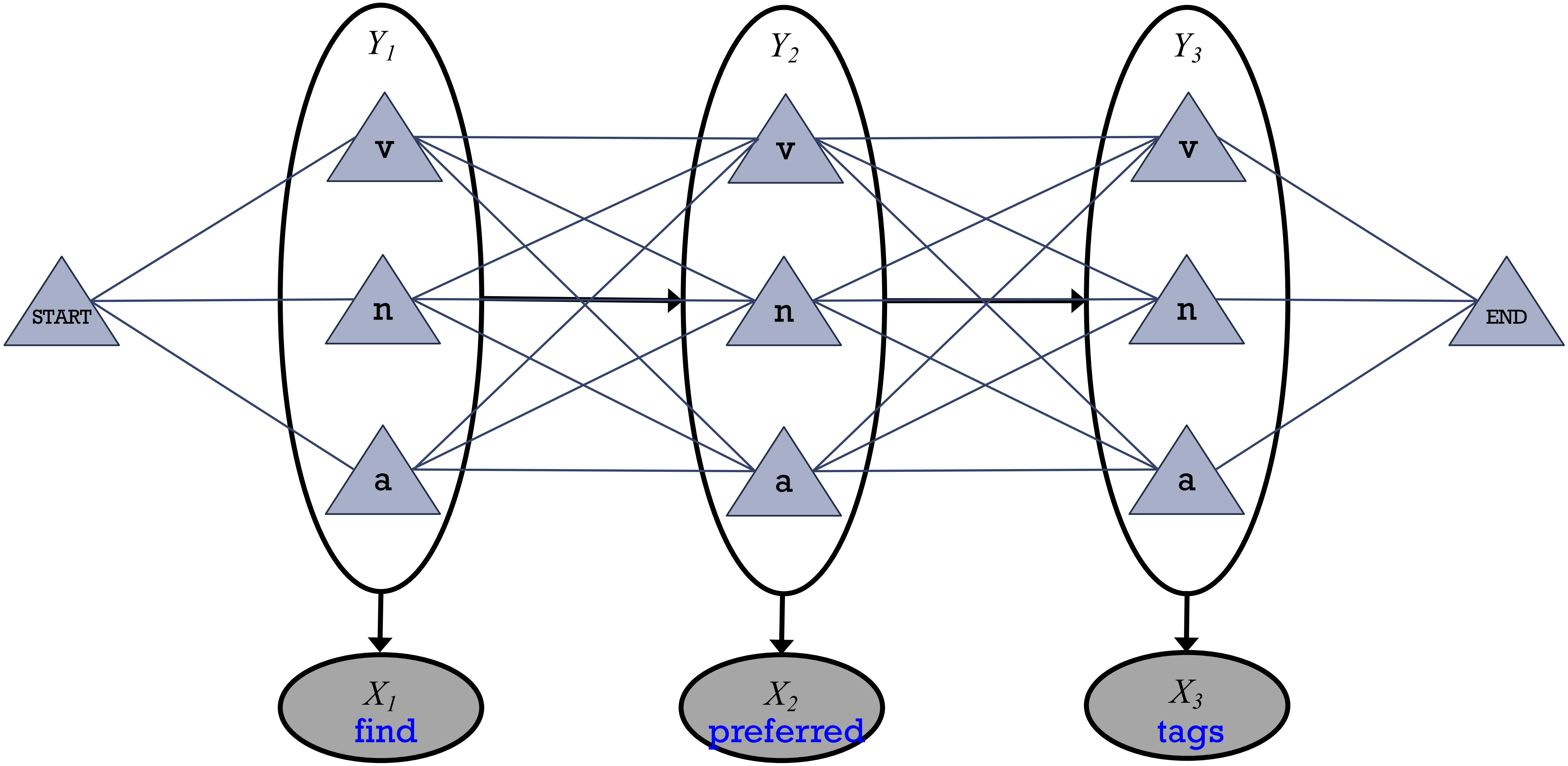


Could be verb or noun *Could be adjective or verb* *Could be noun or verb*

Forward-Backward Algorithm

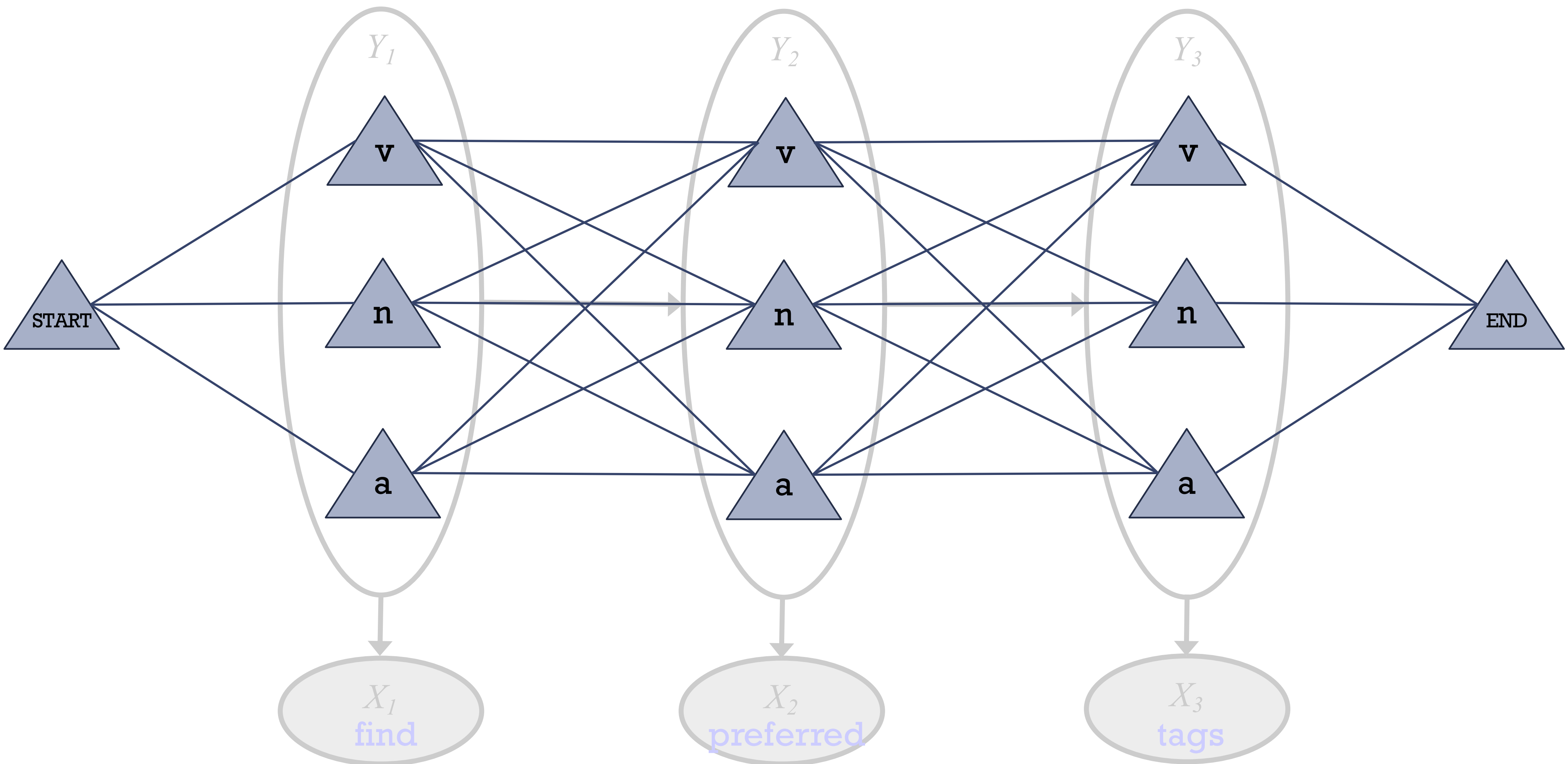


Forward-Backward Algorithm



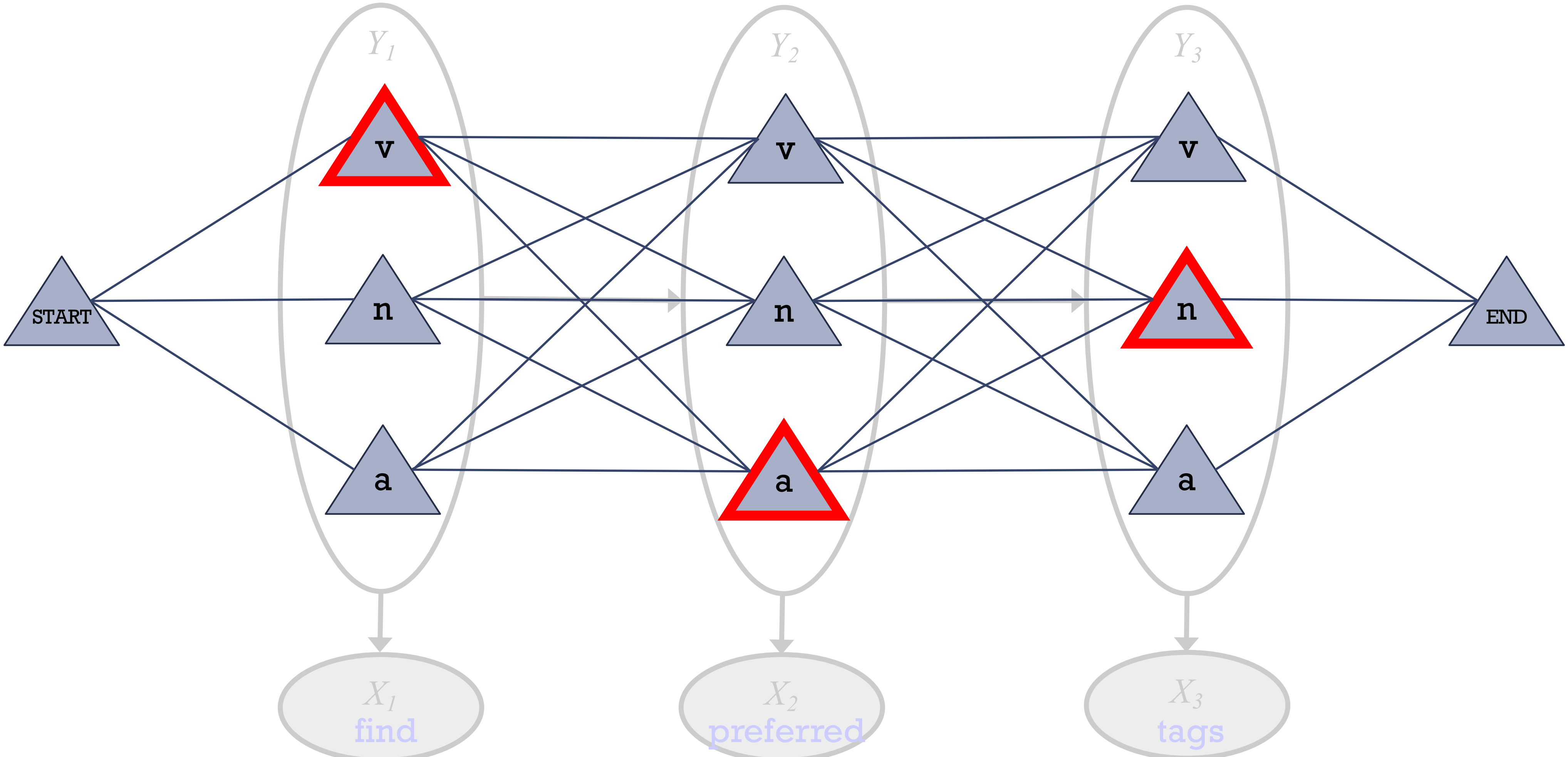
- Let's show the possible *values* for each variable

Forward-Backward Algorithm



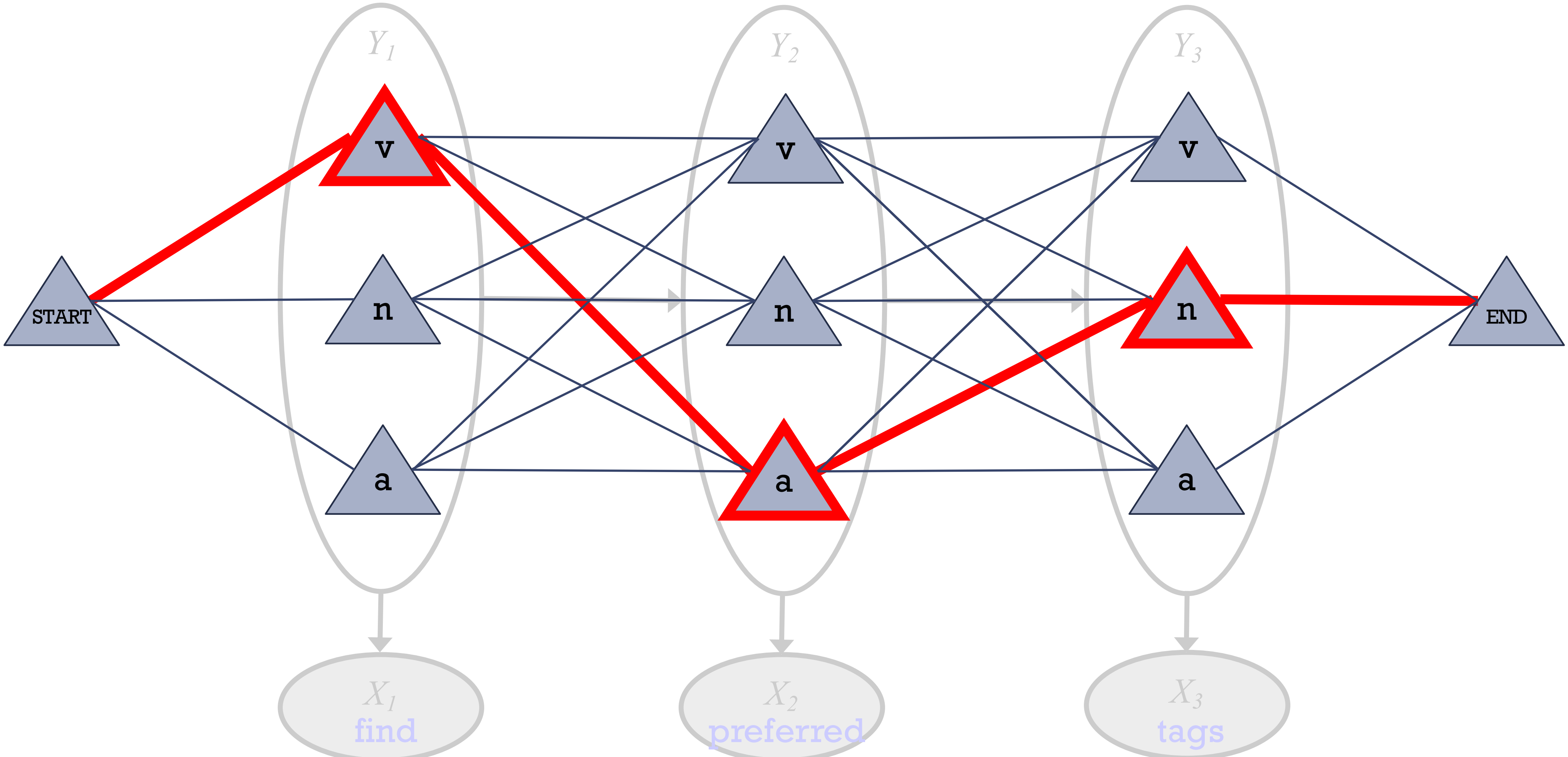
- Let's show the possible *values* for each variable

Forward-Backward Algorithm



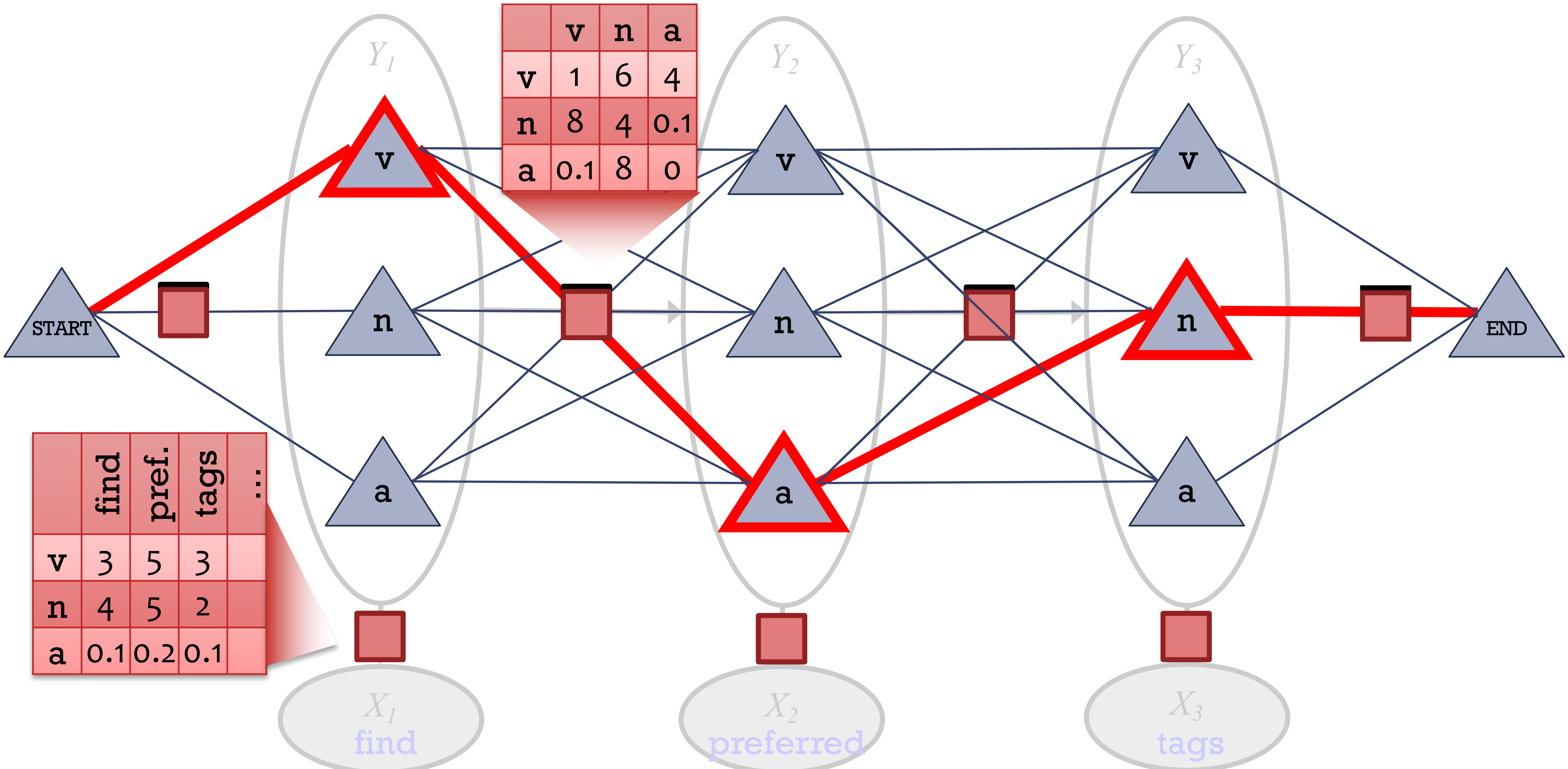
- Let's show the possible *values* for each variable
- One possible assignment

Forward-Backward Algorithm



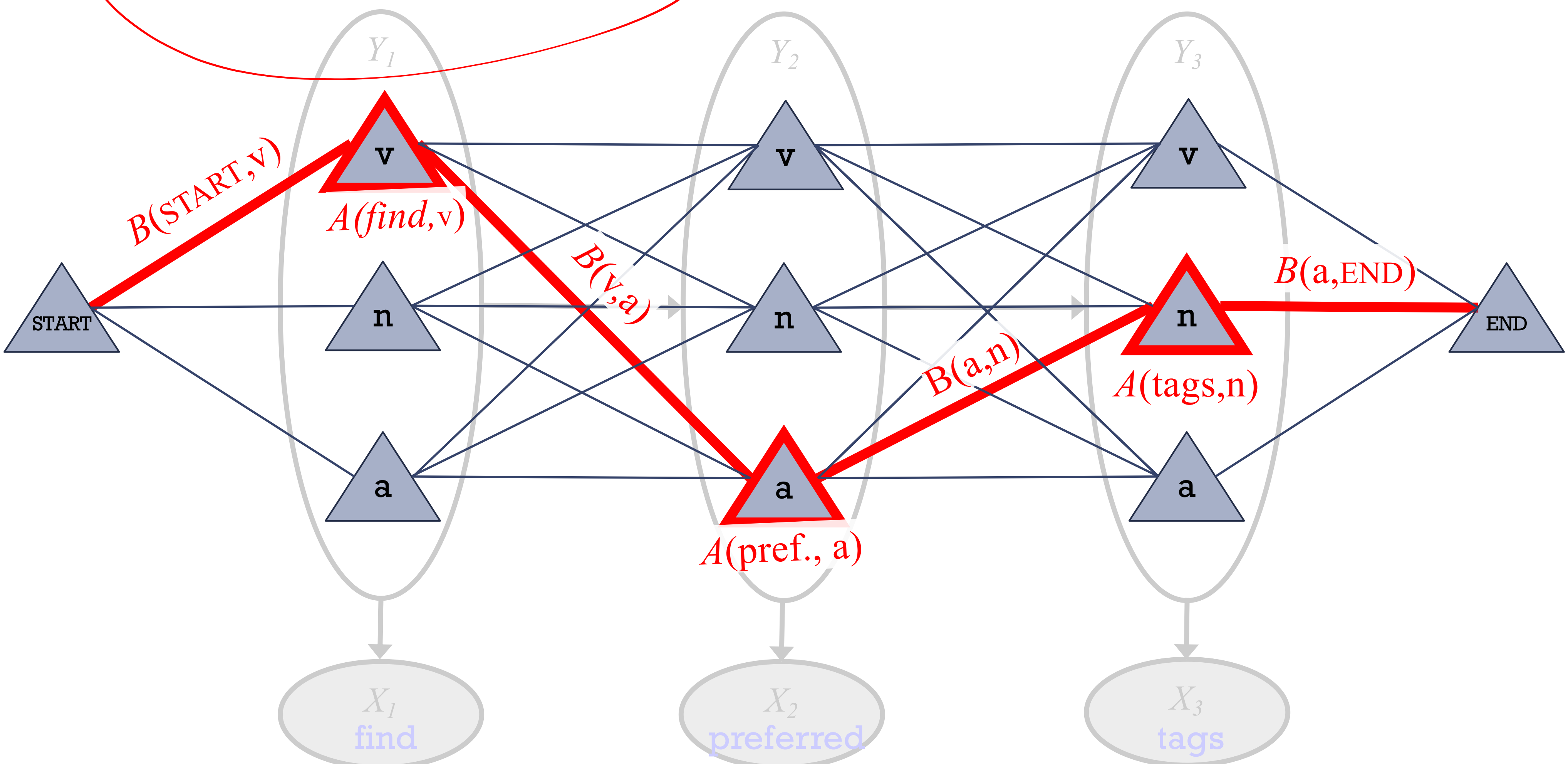
- Let's show the possible *values* for each variable
- One possible assignment
- And what the 7 transition / emission factors think of it …

Forward-Backward Algorithm



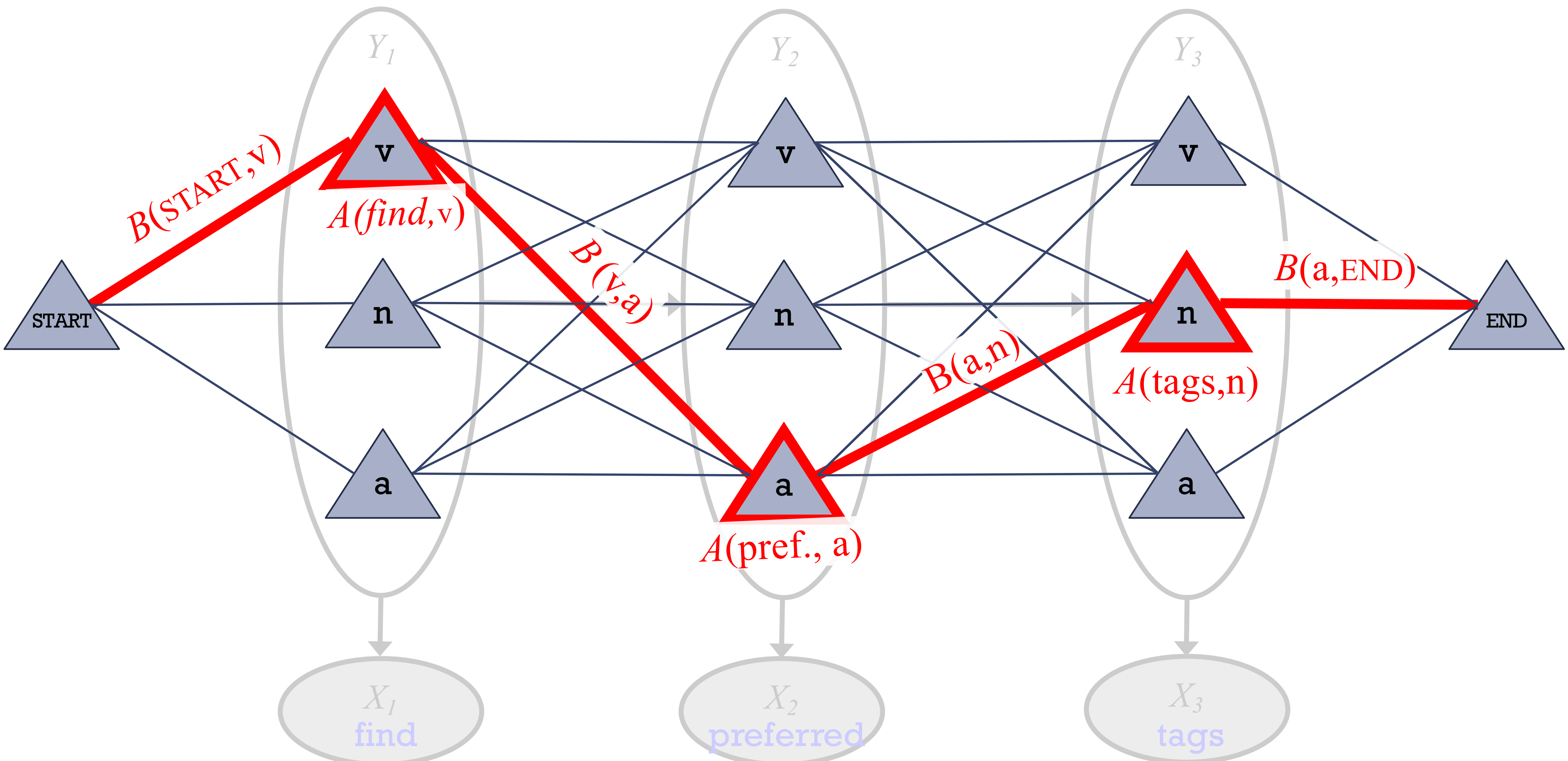
- Let's show the possible *values* for each variable
- One possible assignment
- And what the 7 transition / emission factors think of it …

Viterbi Algorithm: Most Probable Assignment



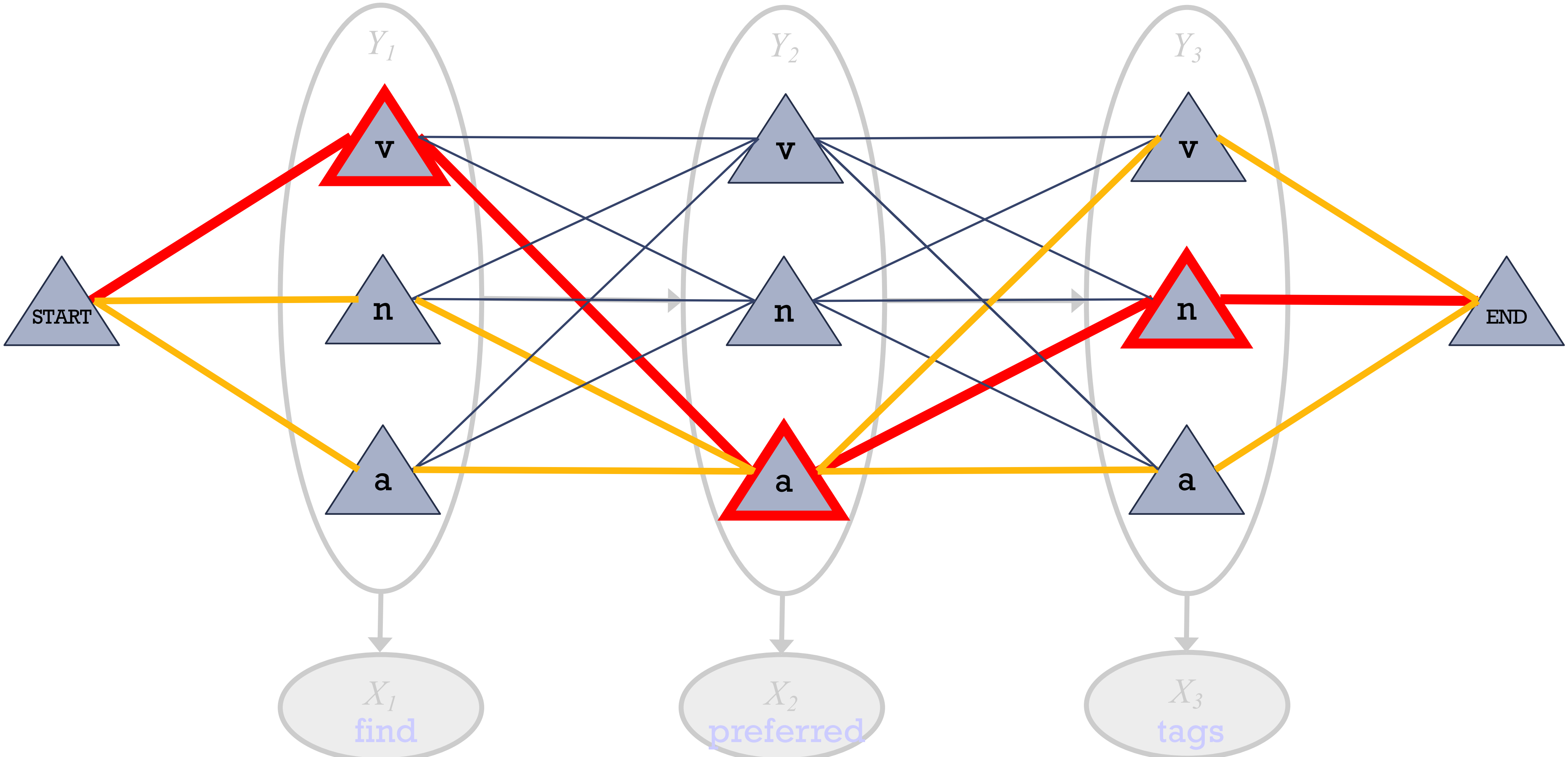
- So p(**v a n**) = (1/Z) * product of 7 numbers
- Numbers associated with edges and nodes of path
- Most probable assignment = **path with highest product**

Viterbi Algorithm: Most Probable Assignment



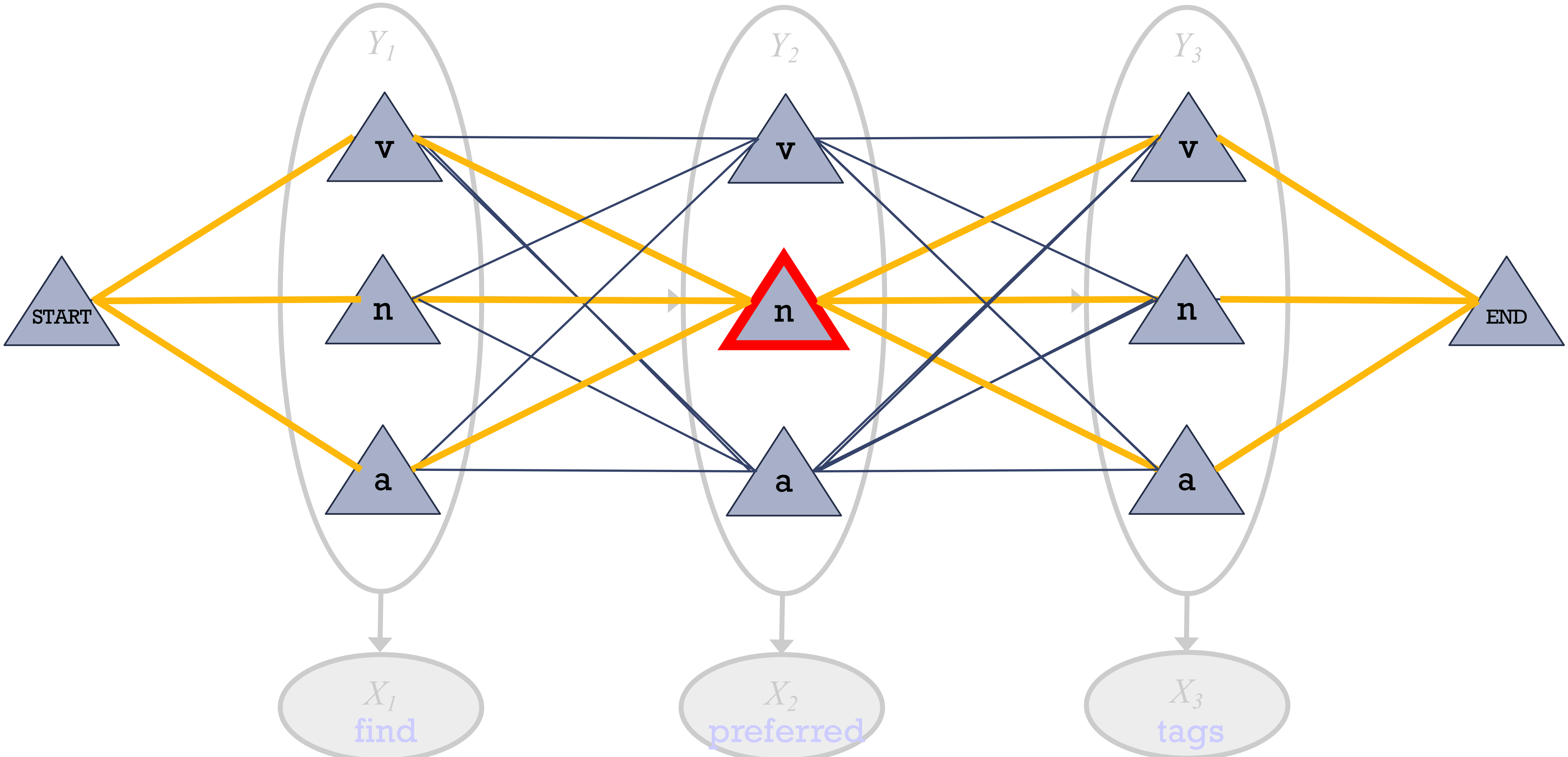
- So p(**v a n**) = (1/Z) * product weight of one path

Forward-Backward Algorithm: Finds Marginals



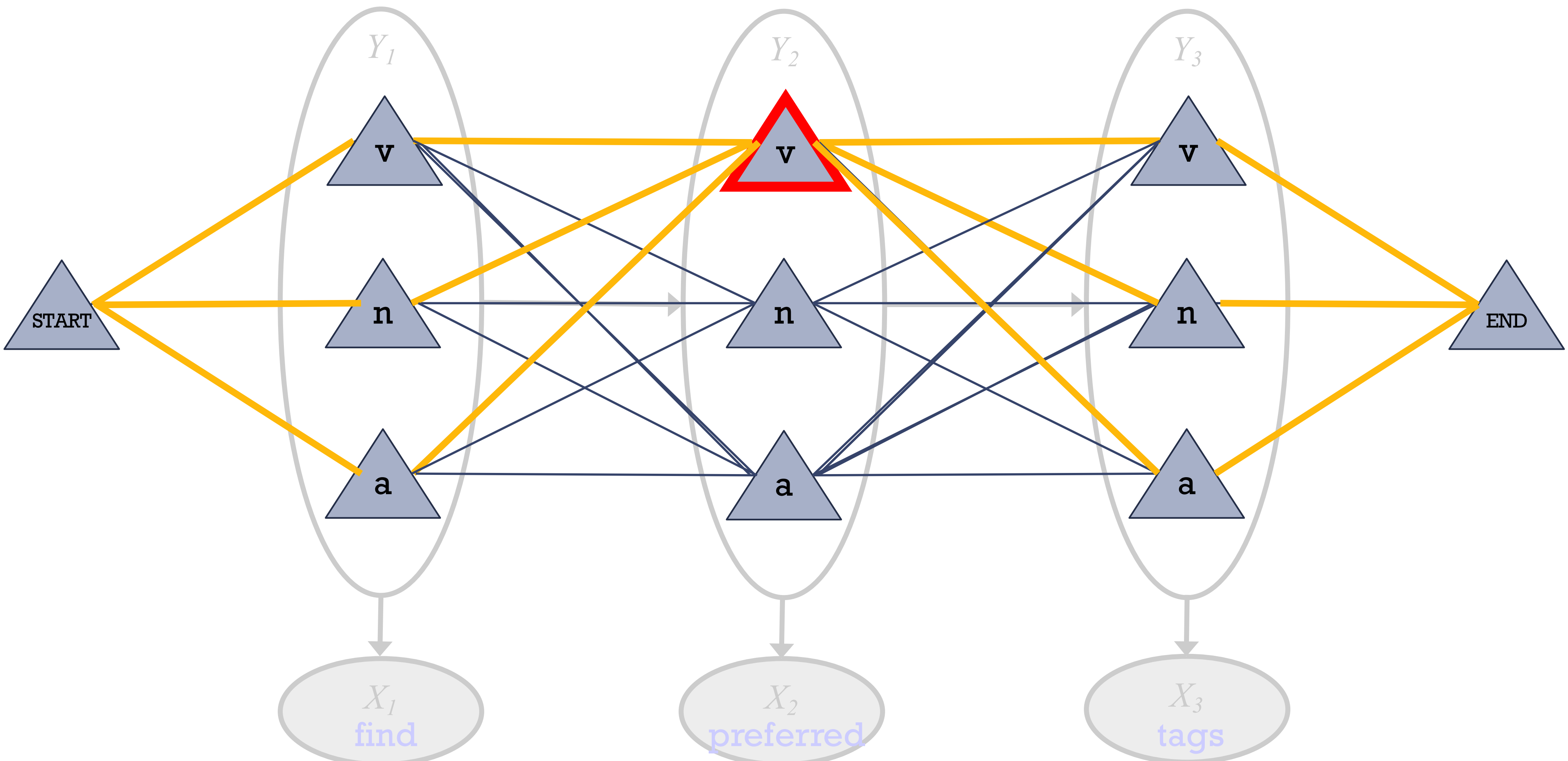
- So p(**v a n**) = (1/Z) * product weight of one path
- Marginal probability p(Y_2 = a)
  = (1/Z) * total weight of *all* paths through **a**

Forward-Backward Algorithm: Finds Marginals



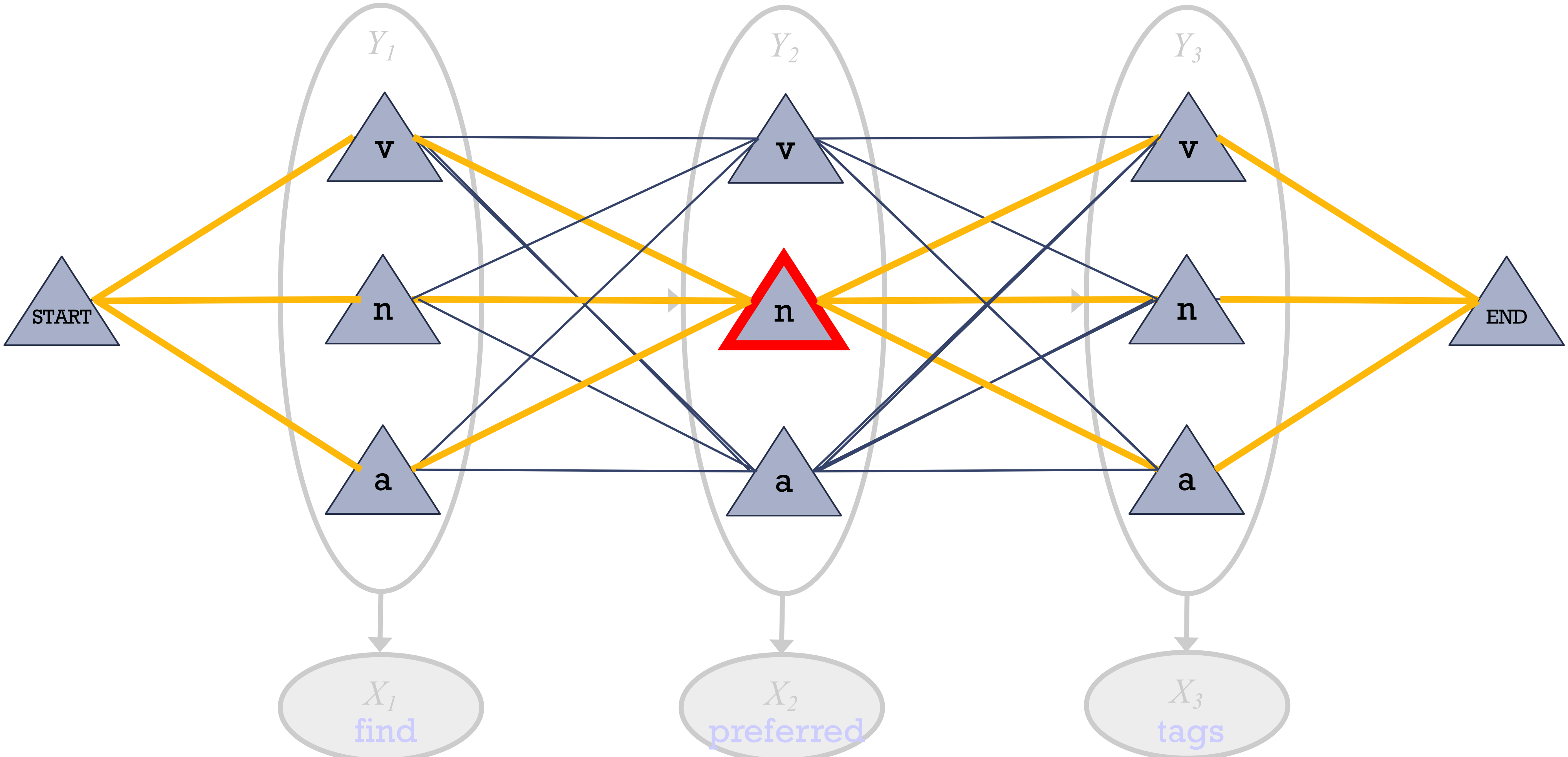
- So p(**v a n**) = (1/Z) * product weight of one path
- Marginal probability $p(Y_2 = n)$
  = (1/Z) * total weight of *all* paths through n

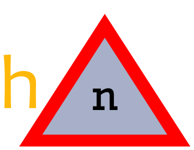
Forward-Backward Algorithm: Finds Marginals



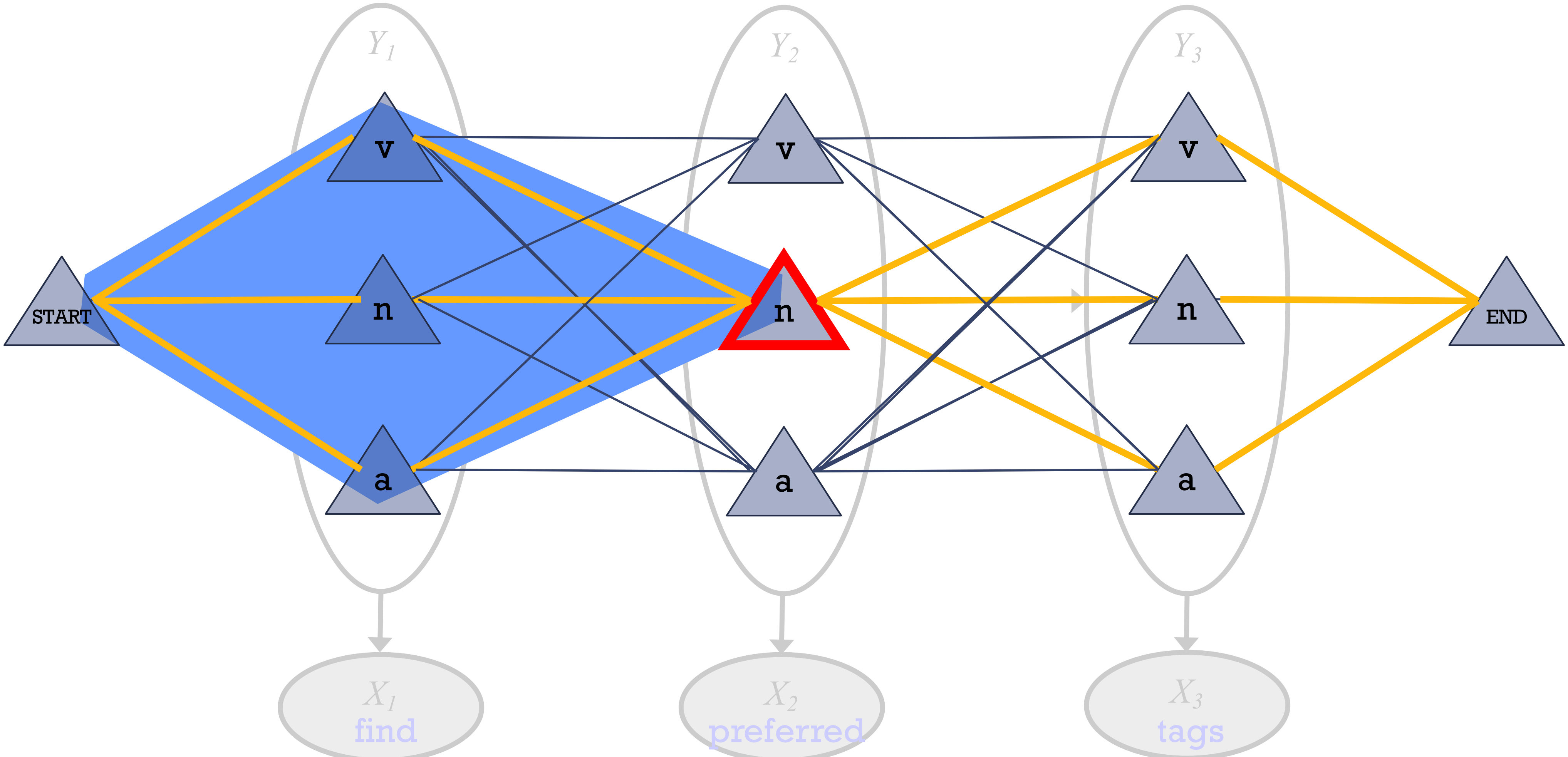
- So p(**v a n**) = (1/Z) * product weight of one path
- Marginal probability p(Y_2 = v)
  = (1/Z) * total weight of *all* paths through v

Forward-Backward Algorithm: Finds Marginals



- So p(**v a n**) = (1/Z) * product weight of one path
- Marginal probability $p(Y_2 = n)$
  = (1/Z) * total weight of *all* paths through 

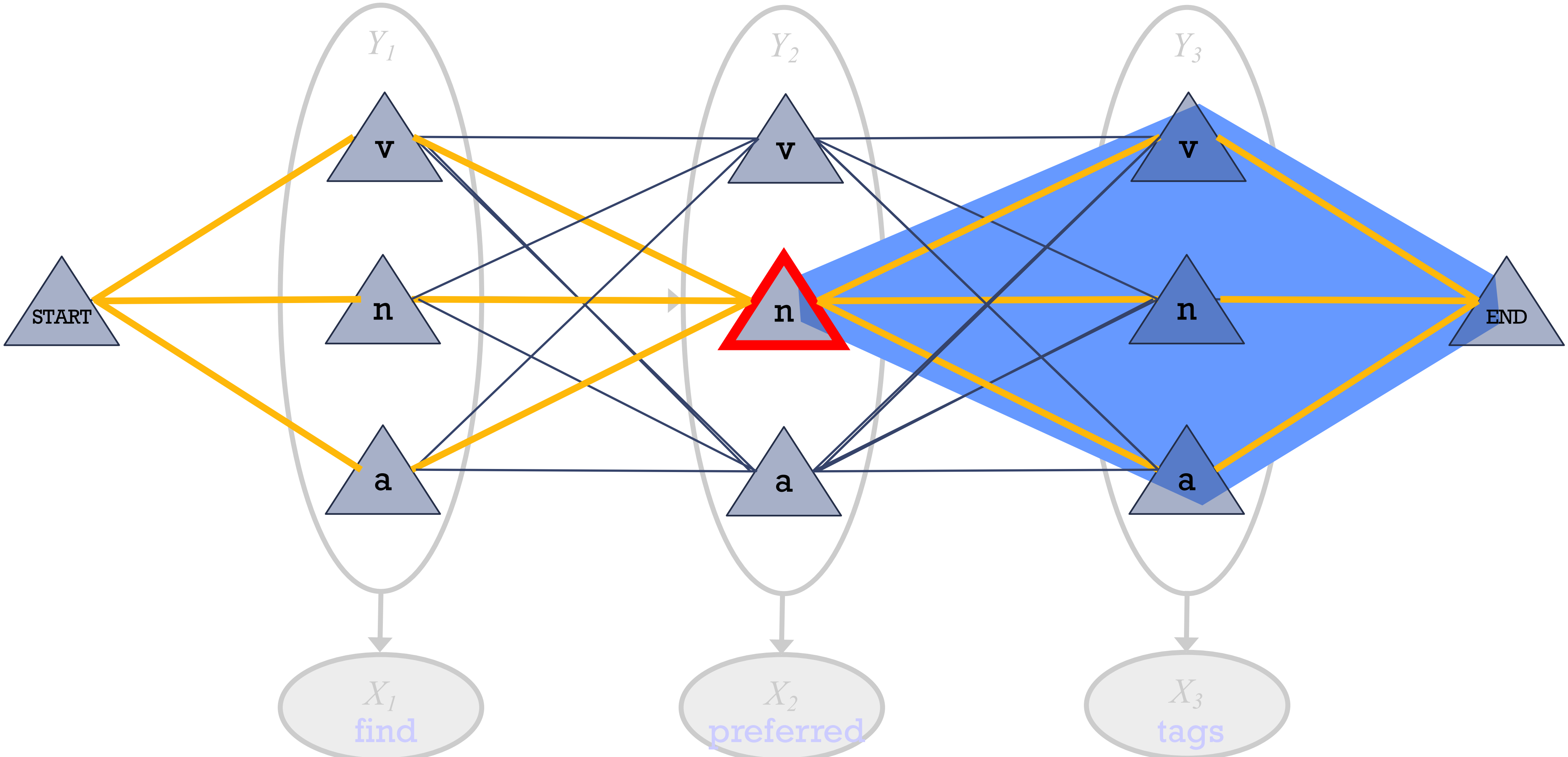
Forward-Backward Algorithm: Finds Marginals



$\alpha_2(\mathbf{n})$ = total weight of these path *prefixes*

(found by dynamic programming: matrix-vector products)

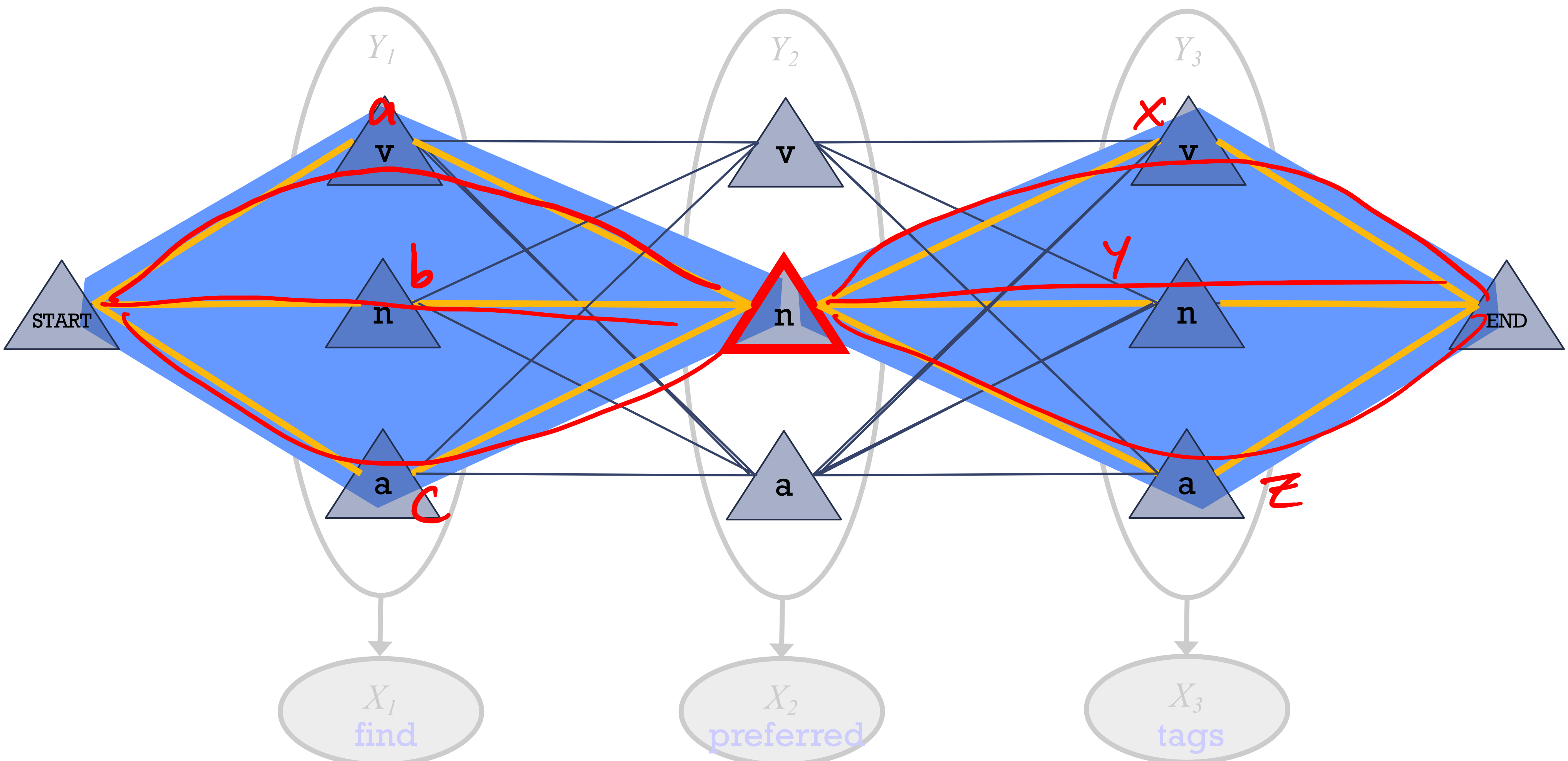
Forward-Backward Algorithm: Finds Marginals



$\beta_2(\mathbf{n})$ = total weight of these path *suffixes*

(found by dynamic programming: matrix-vector products)

Forward-Backward Algorithm: Finds Marginals



$\alpha_2(\mathbf{n})$ = total weight of these path *prefixes* (a + b + c)

$\beta_2(\mathbf{n})$ = total weight of these path *suffixes* (x + y + z)

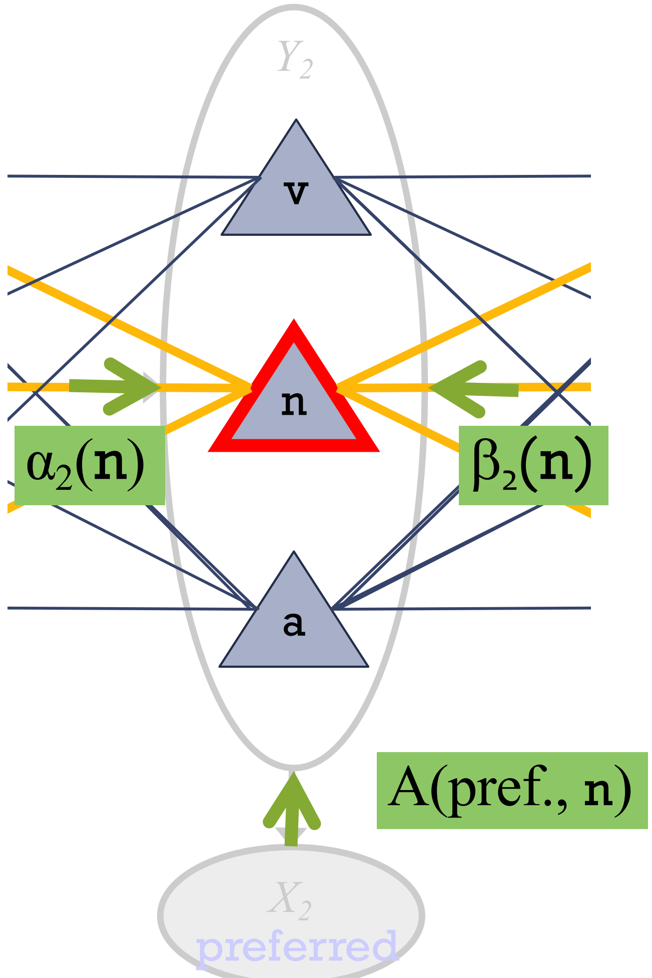
(a+b+c)(x+y+z)

Product gives ax+ay+az+bx+by+bz+cx+cy+cz = total weight of paths

Forward-Backward Algorithm: Finds Marginals

Oops! The weight of a path through a state also includes a weight at that state.
So $\alpha(\mathbf{n})\cdot\beta(\mathbf{n})$ isn't enough.

The extra weight is the opinion of the emission probability at this variable.

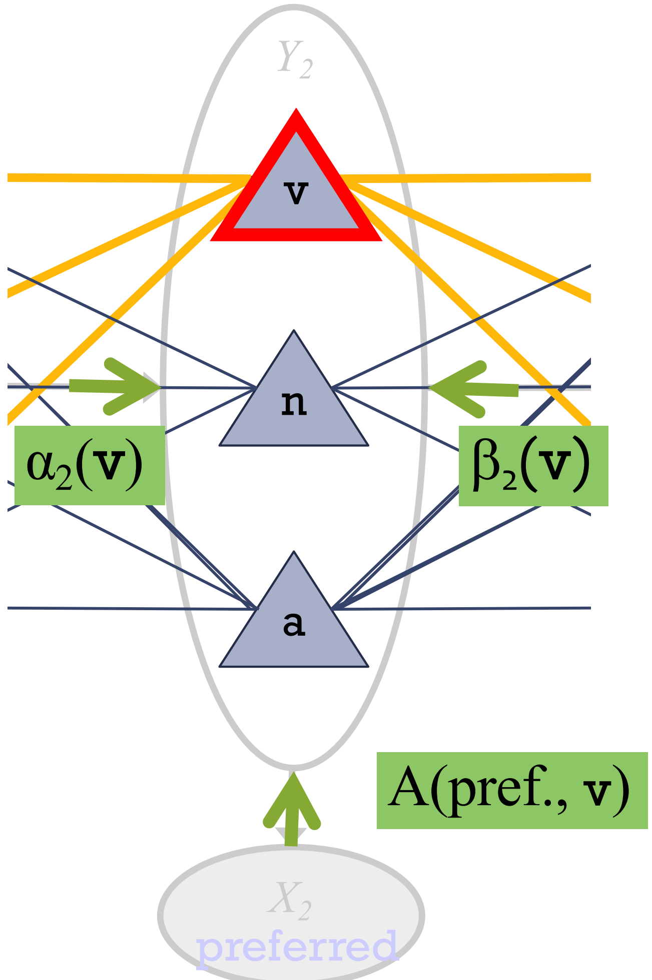


"belief that $Y_2 = \mathbf{n}$"

total weight of *all* paths through **n**

= $\alpha_2(\mathbf{n})$ A(pref., n) $\beta_2(\mathbf{n})$

Forward-Backward Algorithm: Finds Marginals



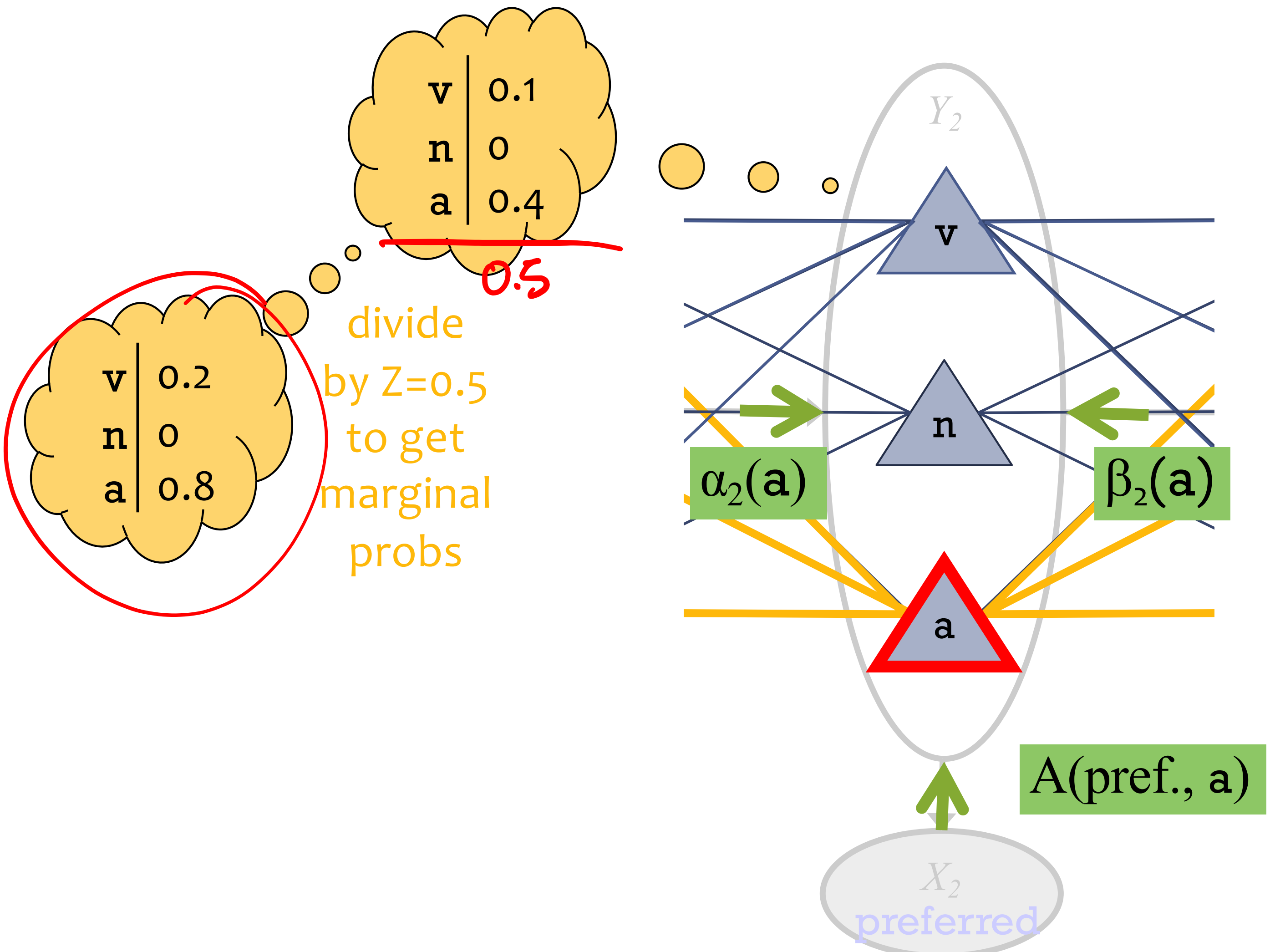
"belief that $Y_2 = \mathbf{v}$"

"belief that $Y_2 = \mathbf{n}$"

total weight of *all* paths through **v**

= $\alpha_2(\mathbf{v})$ A(pref., **v**) $\beta_2(\mathbf{v})$

Forward-Backward Algorithm: Finds Marginals



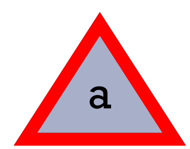
"belief that Y_2 = **v**"

"belief that Y_2 = **n**"

"belief that Y_2 = **a**"

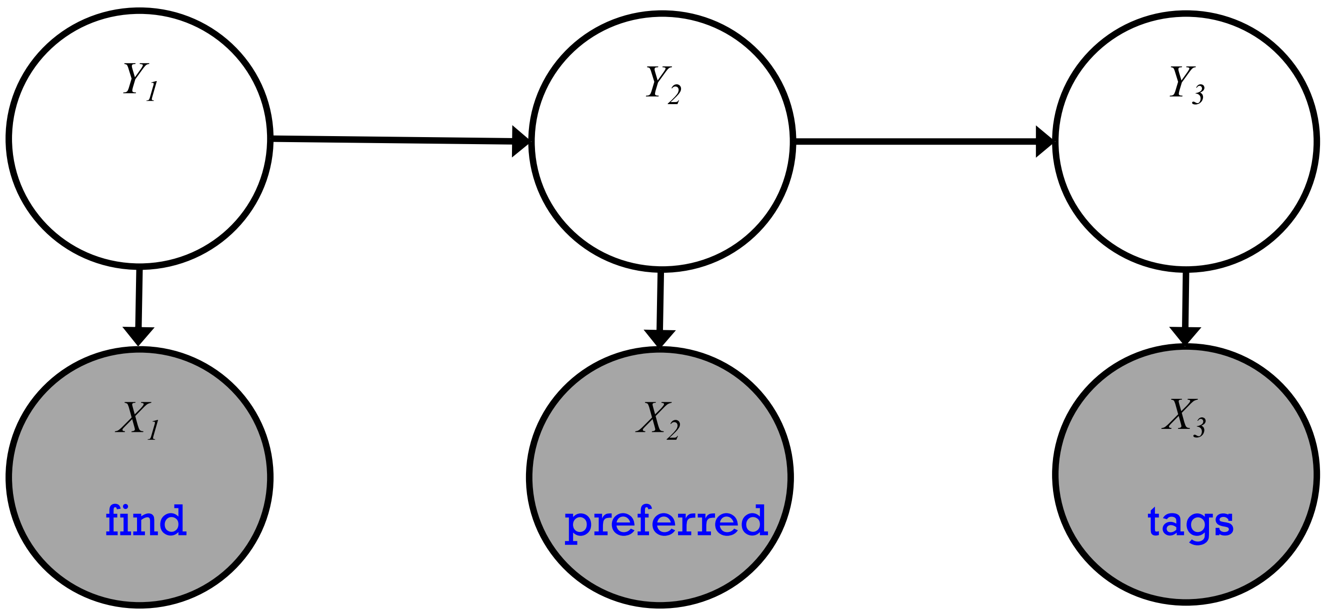
sum = Z
(total weight
of *all* paths)

→ $p(\vec{x})$

total weight of *all* paths through 

= α_2(a) A(pref., a) β_2(a)

Forward-Backward Algorithm



Could be verb or noun *Could be adjective or verb* *Could be noun or verb*

THE FORWARD-BACKWARD ALGORITHM

Forward-Backward Algorithm

Definitions

$\alpha_t(k) \triangleq p(x_1, \ldots, x_t, y_t = k)$

$\beta_t(k) \triangleq p(x_{t+1}, \ldots, x_T \mid y_t = k)$

Assume

$y_0 = \text{START}$

$y_{T+1} = \text{END}$

1. Initialize

$$\alpha_0(\text{START}) = 1 \qquad \alpha_0(k) = 0, \forall k \neq \text{START}$$

$$\beta_{T+1}(\text{END}) = 1 \qquad \beta_{T+1}(k) = 0, \forall k \neq \text{END}$$

2. Forward Algorithm

for $t = 1, \ldots, T+1$:

for $k = 1, \ldots, K$:

$$\alpha_t(k) = \sum_{j=1}^{K} p(x_t \mid y_t = k)\alpha_{t-1}(j)p(y_t = k \mid y_{t-1} = j)$$

3. Backward Algorithm

for $t = T, \ldots, 0$:

for $k = 1, \ldots, K$:

$$\beta_t(k) = \sum_{j=1}^{K} p(x_{t+1} \mid y_{t+1} = j)\beta_{t+1}(j)p(y_{t+1} = j \mid y_t = k)$$

4. Evaluation $p(\mathbf{x}) = \alpha_{T+1}(\text{END})$

5. Marginals $p(y_t = k \mid \mathbf{x}) = \frac{\alpha_t(k)\beta_t(k)}{p(\mathbf{x})}$