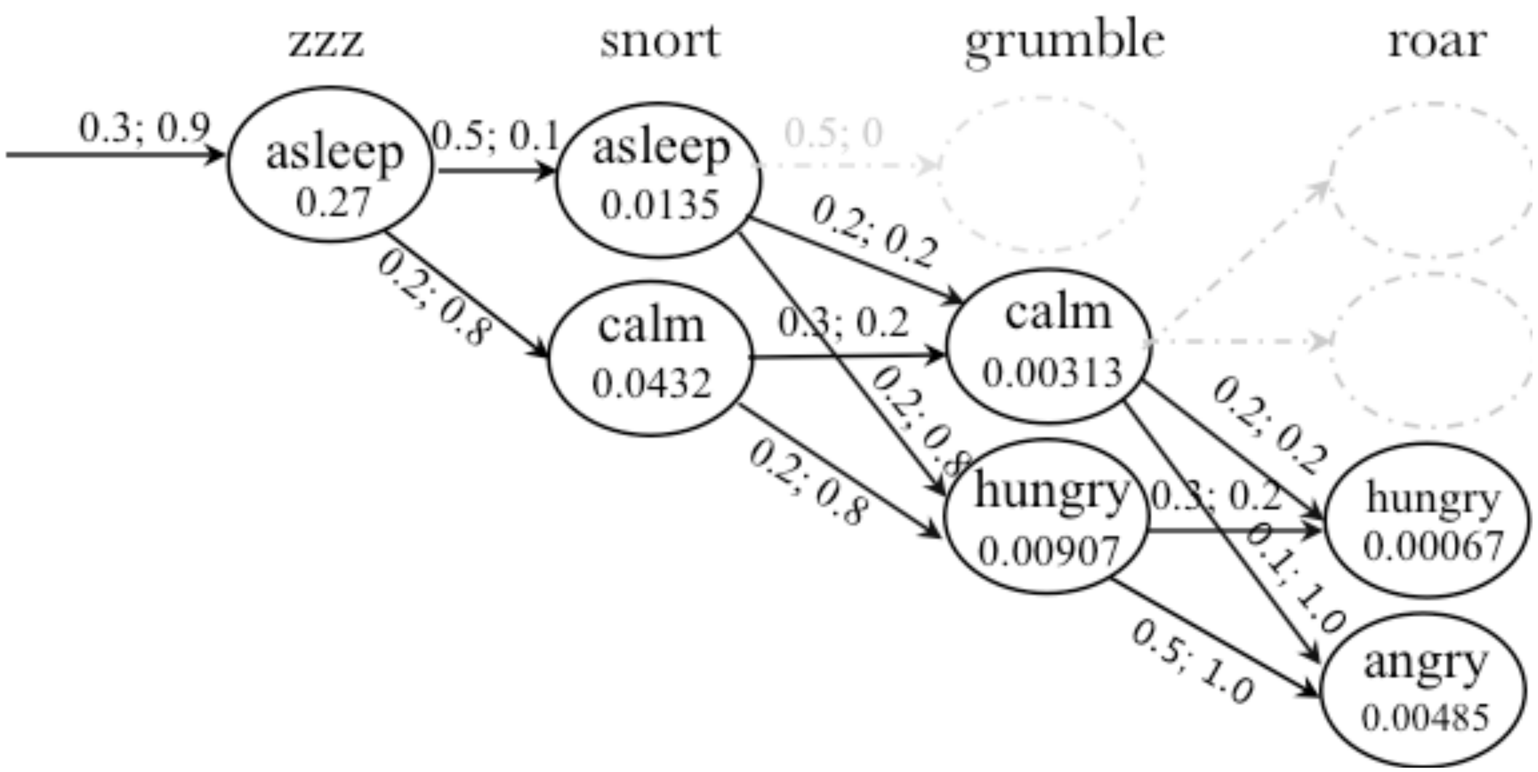
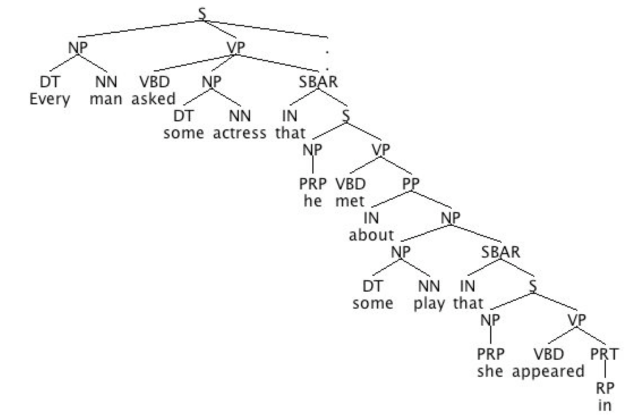
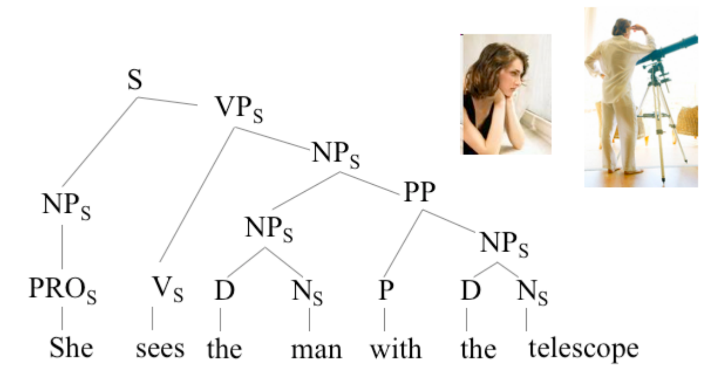
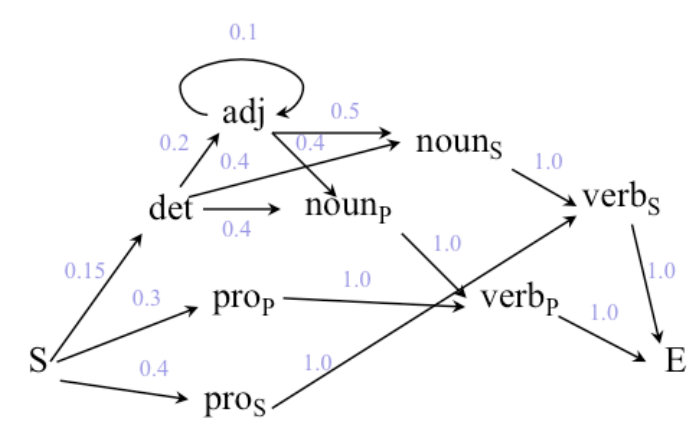
Computational Cognitive Science









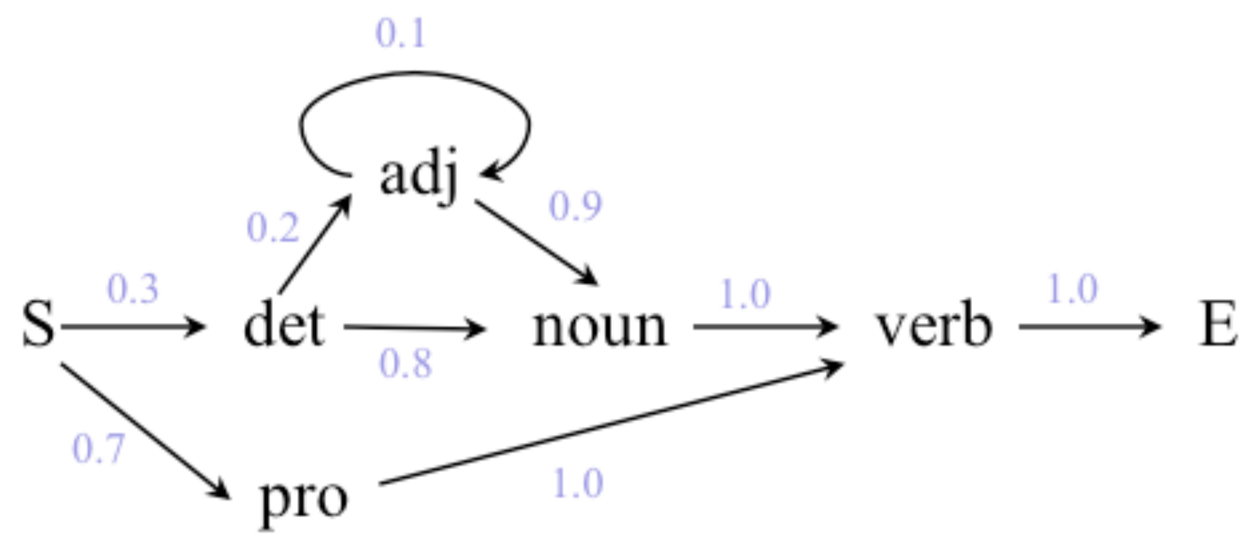
Lecture 19: HMMs and more complex grammars

Last time

- Because of the problem of long-distance dependencies, Markov models are not good models of language: they need to be too large to capture its regularities

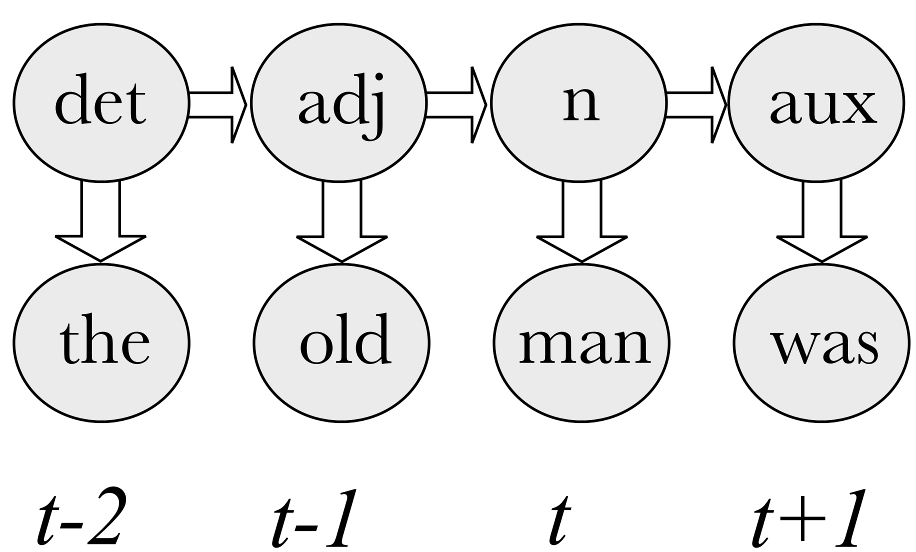
Last time

- Because of the problem of long-distance dependencies, Markov models are not good models of language: they need to be too large to capture its regularities
- Grammars that incorporate parts of speech can be useful for greatly minimising the size of the grammar required



Last time

- Because of the problem of long-distance dependencies, Markov models are not good models of language: they need to be too large to capture its regularities
- Grammars that incorporate parts of speech can be useful for greatly minimising the size of the grammar required
- Hidden Markov models, which involve hidden states that generate observations, can capture parts of speech



Last time

- Because of the problem of long-distance dependencies, Markov models are not good models of language: they need to be too large to capture its regularities
- Grammars that incorporate parts of speech can be useful for greatly minimising the size of the grammar required
- Hidden Markov models, which involve hidden states that generate observations, can capture parts of speech
- We can use such models to generate sequences of observations in both linguistic and non-linguistic contexts

Plan

- Last time: introduction to HMMs
  - Limitations of n-grams applied to language
  - Basics of HMMs
- Today: finishing HMMs, and more complex structures
  - Determining the likelihood of a given observation
  - Calculating the most likely state sequence
  - Finding the best HMM for given data
  - More complex models of language

Plan

- Last time: introduction to HMMs
  - Limitations of n-grams applied to language
  - Basics of HMMs
- ➡ Today: finishing HMMs, and more complex structures
  - Determining the likelihood of a given observation
  - Calculating the most likely state sequence
  - Finding the best HMM for given data
  - More complex models of language

Three fundamental questions for HMMs

- Given a model $M = (A,B,\Pi)$, how do we efficiently compute how likely a certain observation is?
- Given a sequence of observations Y and a model M, how do we infer the state sequence that best explains the observations?
- Given an observation sequence Y and a space of possible models found by varying the model parameters $M = (A,B,\Pi)$, how do we find the model that best explains the observed data?

Forward* algorithm

Viterbi* algorithm

Baum-Welch** algorithm

* You should be able to implement this; ** You don't need to be able to implement this

Three fundamental questions for HMMs

- ➡ Given a model $M = (A,B,\Pi)$, how do we efficiently compute how likely a certain observation is?
- Given a sequence of observations Y and a model M, how do we infer the state sequence that best explains the observations?
- Given an observation sequence Y and a space of possible models found by varying the model parameters $M = (A,B,\Pi)$, how do we find the model that best explains the observed data?

Forward* algorithm

Viterbi* algorithm

Baum-Welch** algorithm

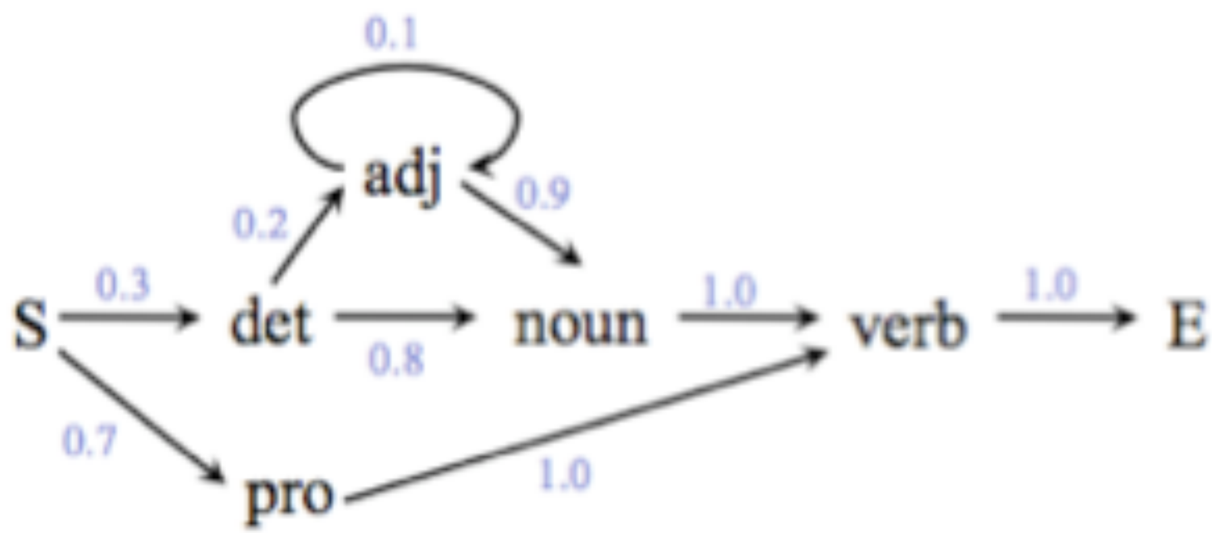
* You should be able to implement this; ** You don't need to be able to implement this

Computing likelihood of observations

For any output sequence $Y = (y_1, \ldots, y_T)$ we can calculate the probability of observing it by summing over all possible sequences of hidden states that could have generated it:

$$p(Y) = \sum_X p(Y|X)p(X)$$

Example: simple language
How likely are you to see “he eats”?



(0.5) verb → eats
(0.5) verb → runs
(0.3) pro → he
(0.3) pro → she
(0.4) pro → it
(0.7) det → the
(0.3) det → a
(0.4) noun → boy
(0.4) noun → dog
(0.2) noun → tiger
(1.0) adj → happy

$P($ he eats $| A,B,\Pi)$

= P(pro|S) P(he|pro) P(verb|pro) P(eats| verb) P(E|verb)

= 0.7 * 0.3 * 1.0 * 0.5 * 1.0
= 0.105

But that was easy, because there was just one way to generate that observation

Example: Mitee the warrior
How likely are you to see "zzz snort"?

Initial state probabilities Π:

Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8

$$P(\text{zzz snort} \mid A,B,\Pi)$$

$$= P(\text{zzz} \mid \text{asleep})\, P(\text{asleep})\, P(\text{calm} \mid \text{asleep})\, P(\text{snort} \mid \text{calm}) + P(\text{zzz} \mid \text{asleep})\, P(\text{asleep})\, P(\text{asleep} \mid \text{asleep})\, P(\text{snort} \mid \text{asleep})$$

$$= (0.9)\,(0.3)\,(0.2)\,(0.8) + (0.9)\,(0.3)\,(0.5)\,(0.1)$$

$$= 0.0675 + 0.0135$$

$$= 0.081$$

Computing likelihood of observations

You can see that this will grow increasingly difficult as the HMM grows increasingly larger (or there are fewer zeros in the transition matrix)

$$p(Y) = \sum_X p(Y|X)p(X)$$

Having to sum over every possible set of hidden states, in general, requires on the order of N^{T+1} multiplications, where T = # of time steps, and N = the number of states. The complexity is thus $O(N^T)$

Simplifying the computation

Luckily, in order to calculate the most likely path we don't have to sum over all possible state sequences

$$\cancel{p(Y) = \sum_{X} p(Y|X)p(X)}$$

Because of the limited horizon property, the probability of the path at any one point only depends on the probability of the current point and the probability of the previous point

Forward algorithm

An algorithm for efficiently calculating the probability of a sequence of observations

Incremental: at each observation step, you find the most likely path until that point

Complexity is $O(N^2T)$, assuming a fully connected model – a big improvement over $O(N^T)$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8

There is only one state that outputs zzz

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

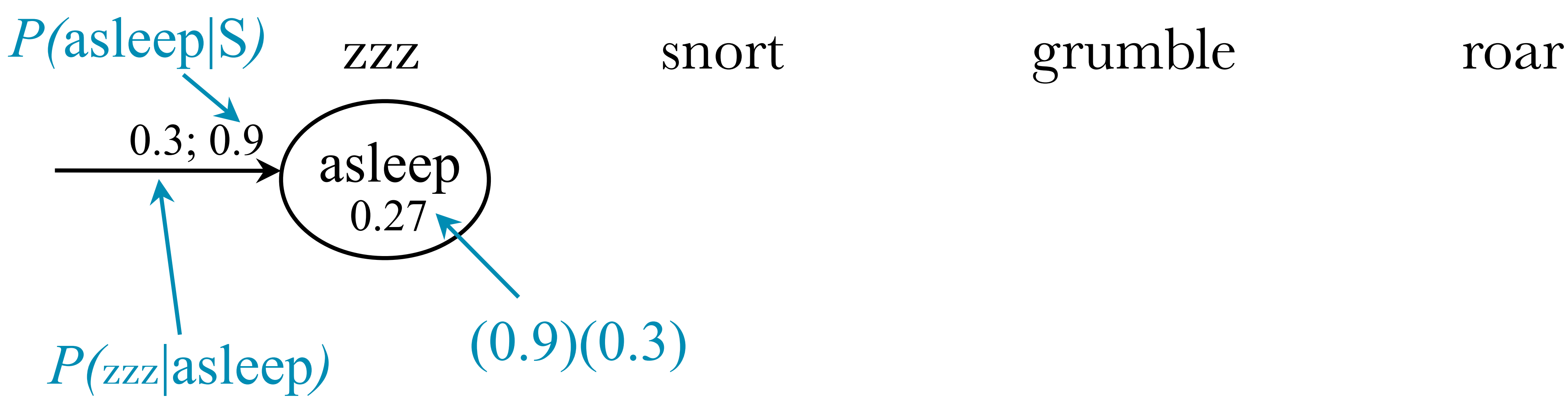
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



There is only one state that outputs zzz

Example: Mitee the warrior

How likely are you to see “zzz snort grumble roar”?

Initial state probabilities Π:

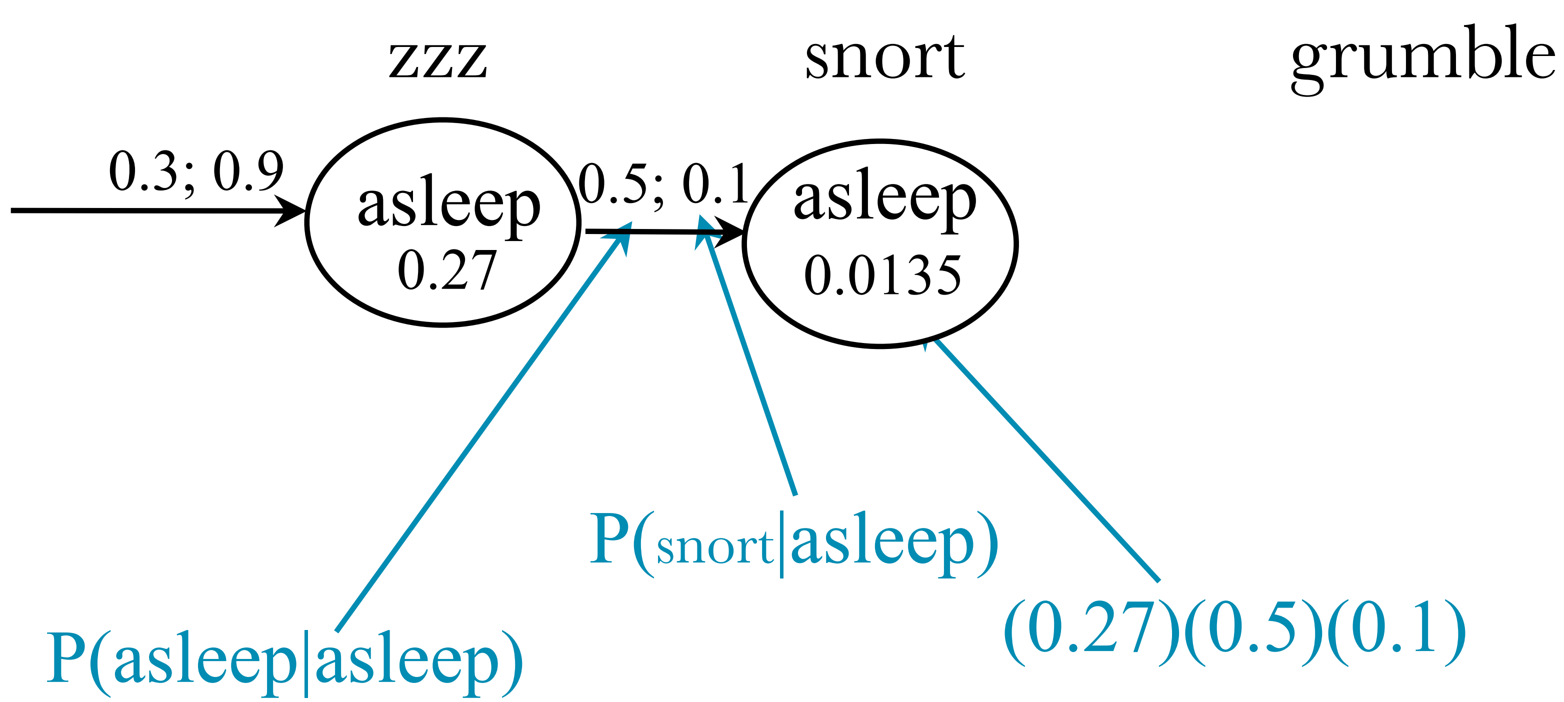
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

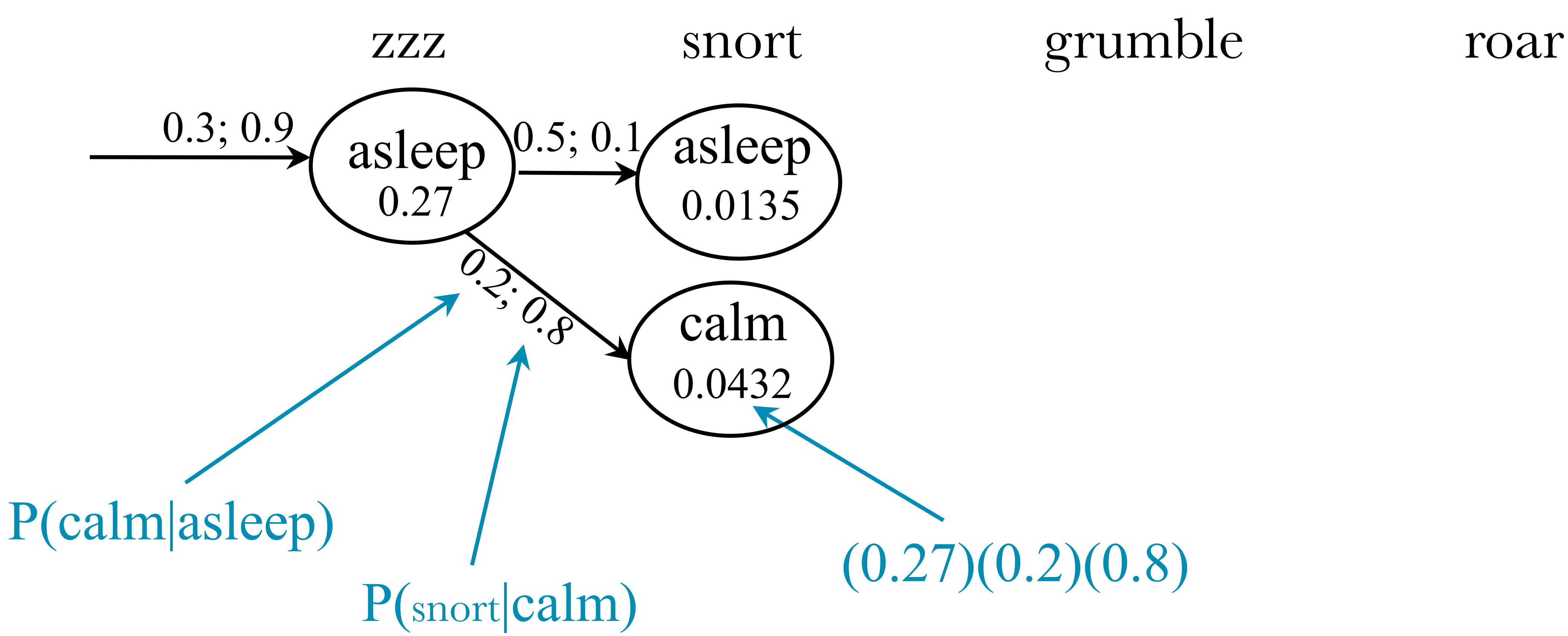
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

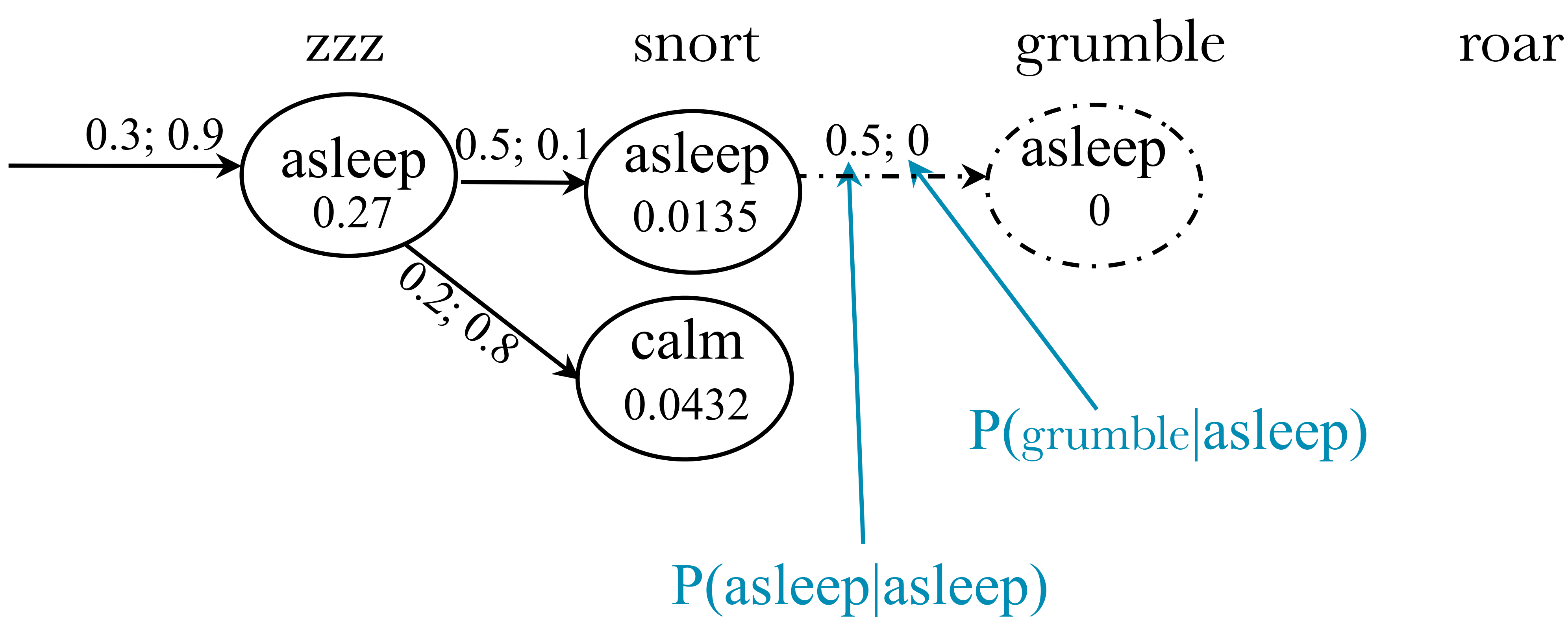
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

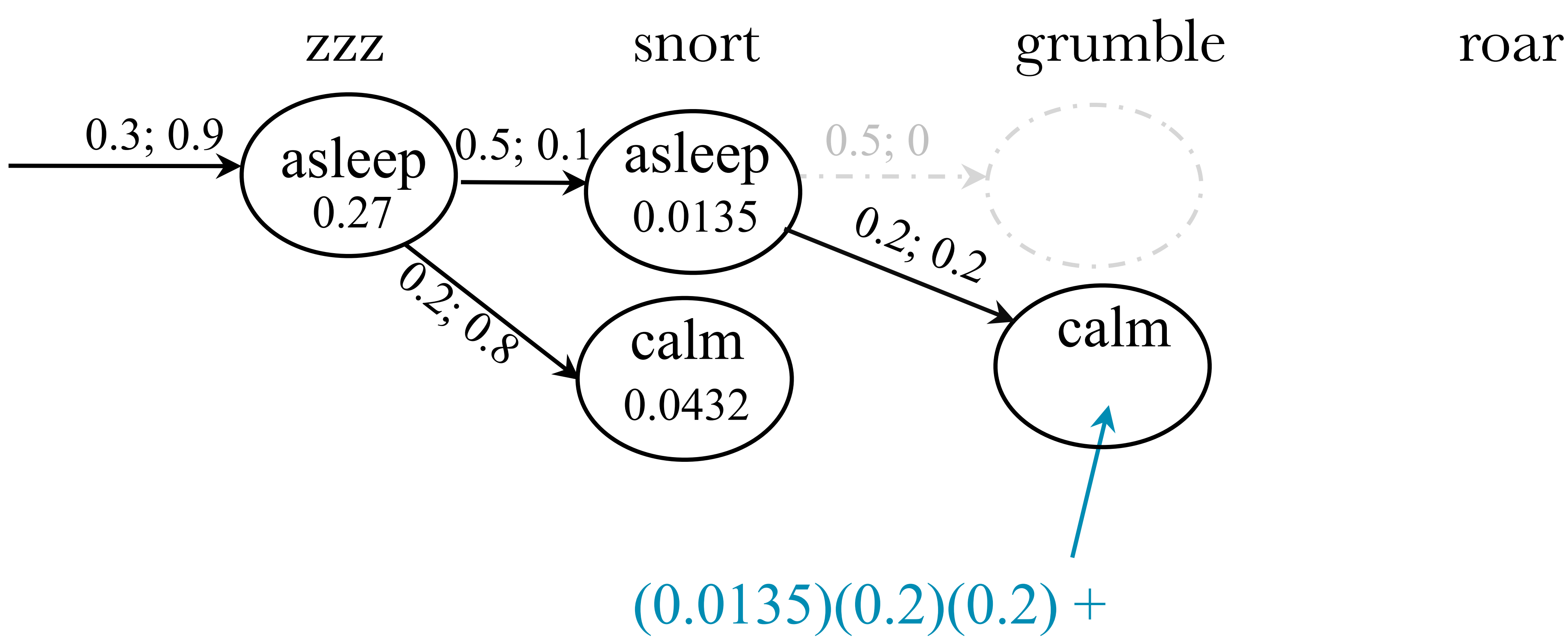
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see

"zzz snort grumble roar"?

Initial state probabilities Π:

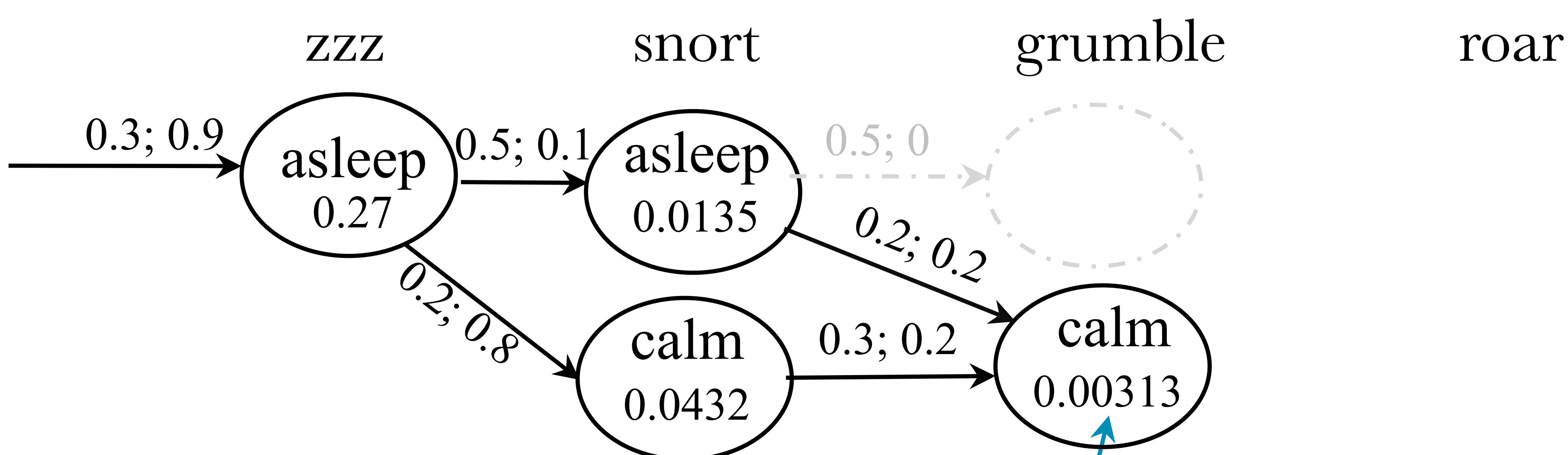
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



(0.0135)(0.2)(0.2) + (0.0432)(0.2)(0.3)

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

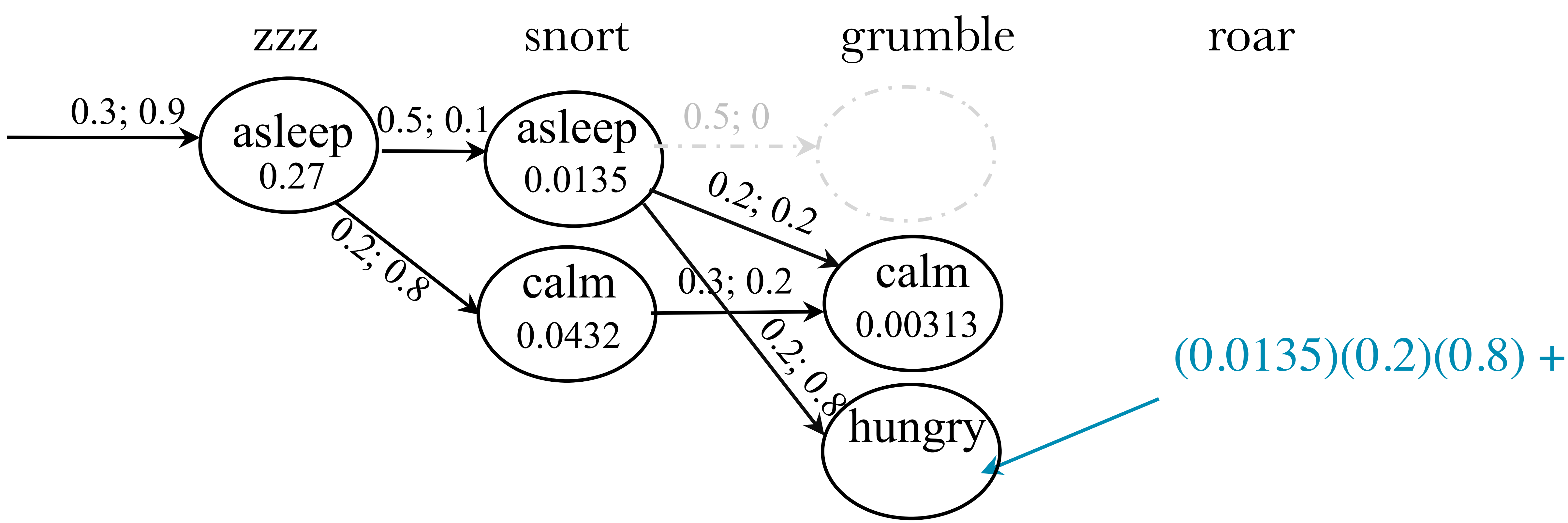
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

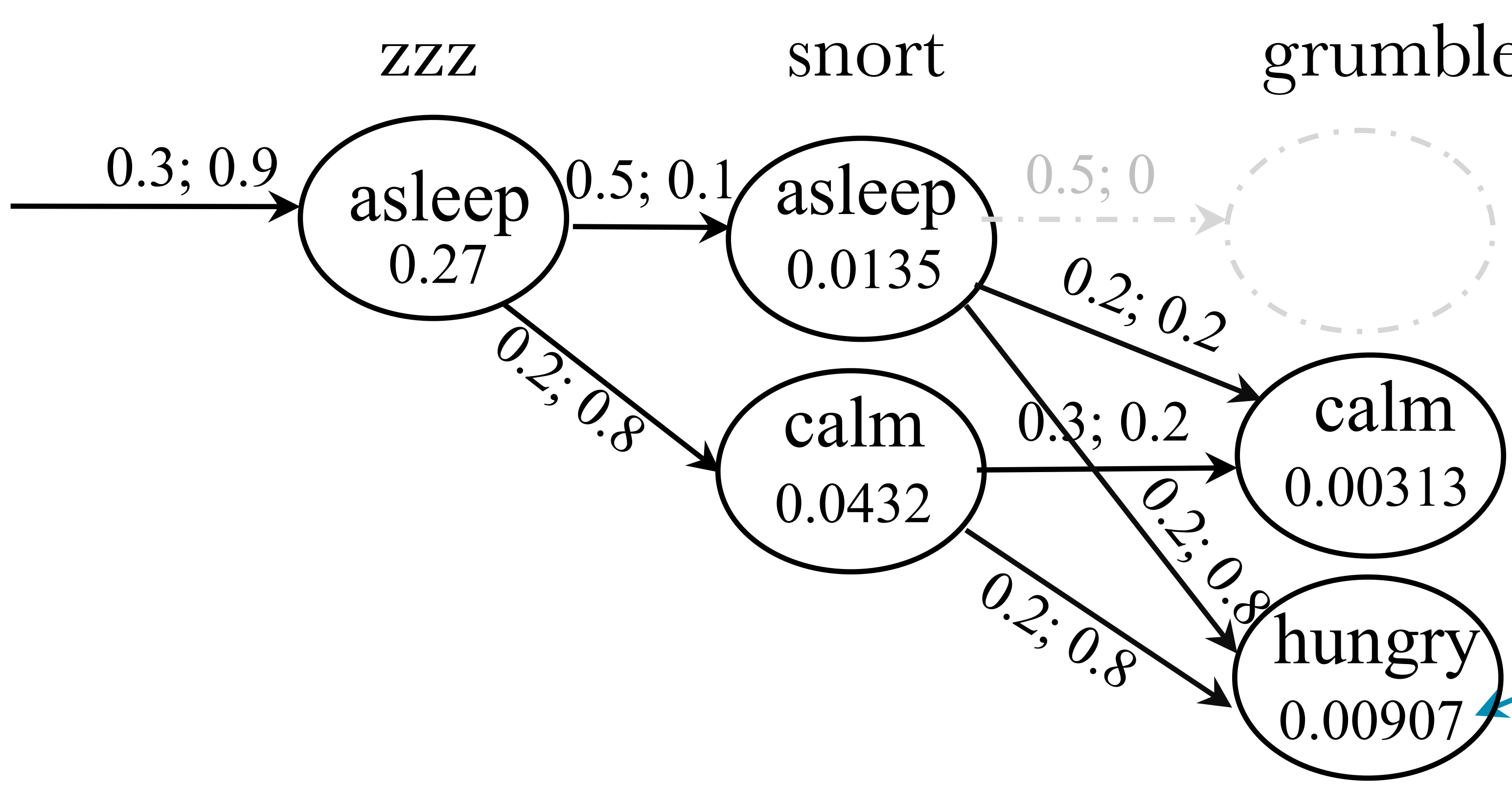
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



(0.0135)(0.2)(0.8) +
(0.0432)(0.2)(0.8)

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

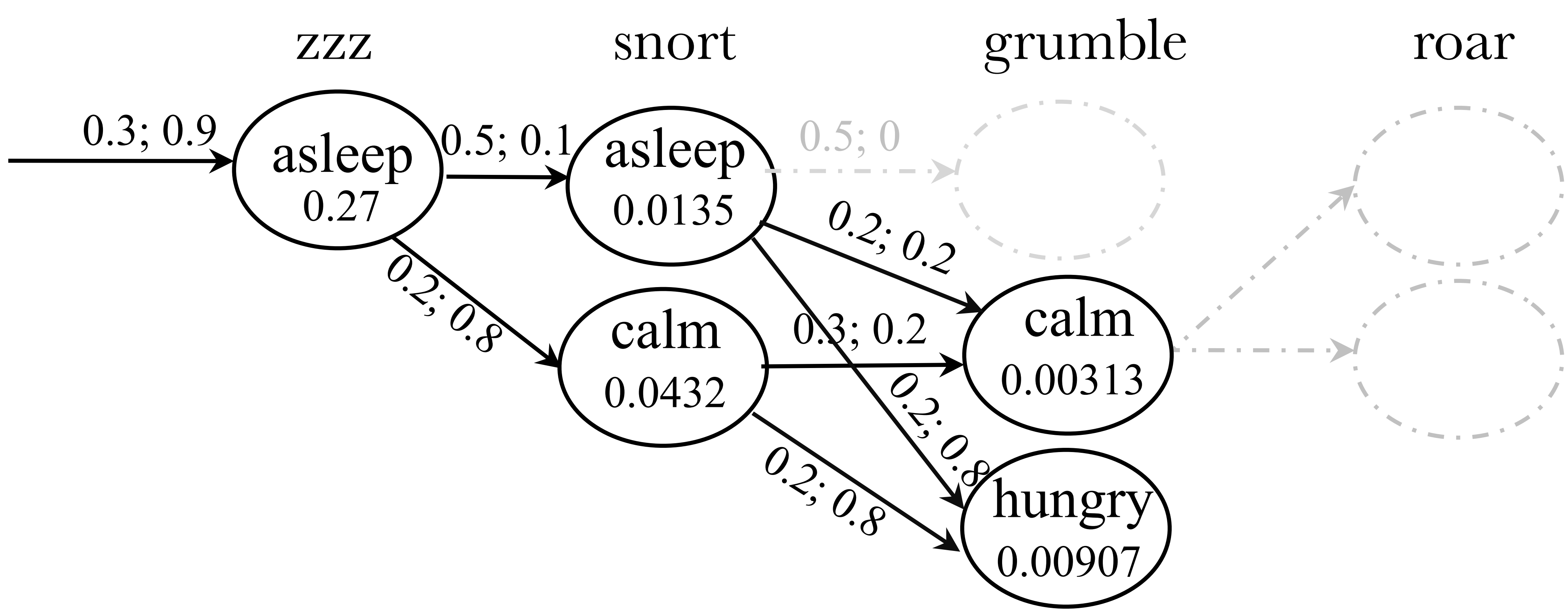
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

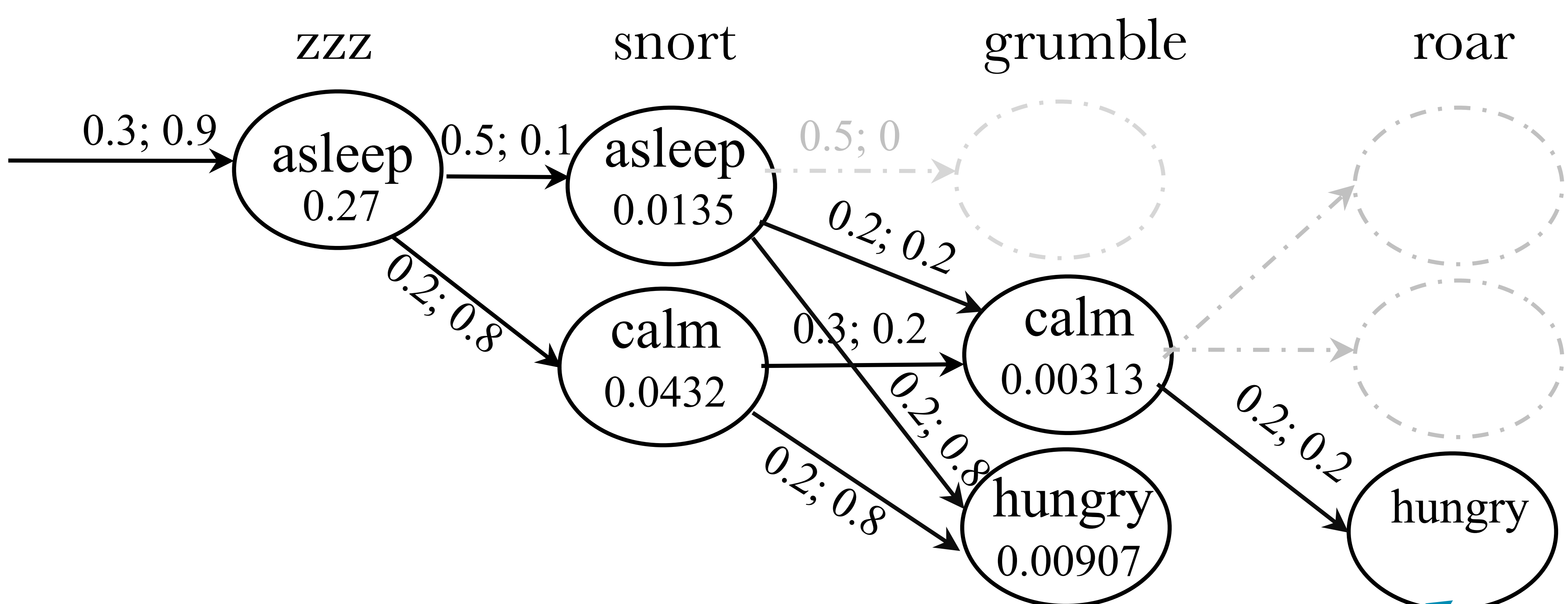
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



0.00313*0.2*0.2 +

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

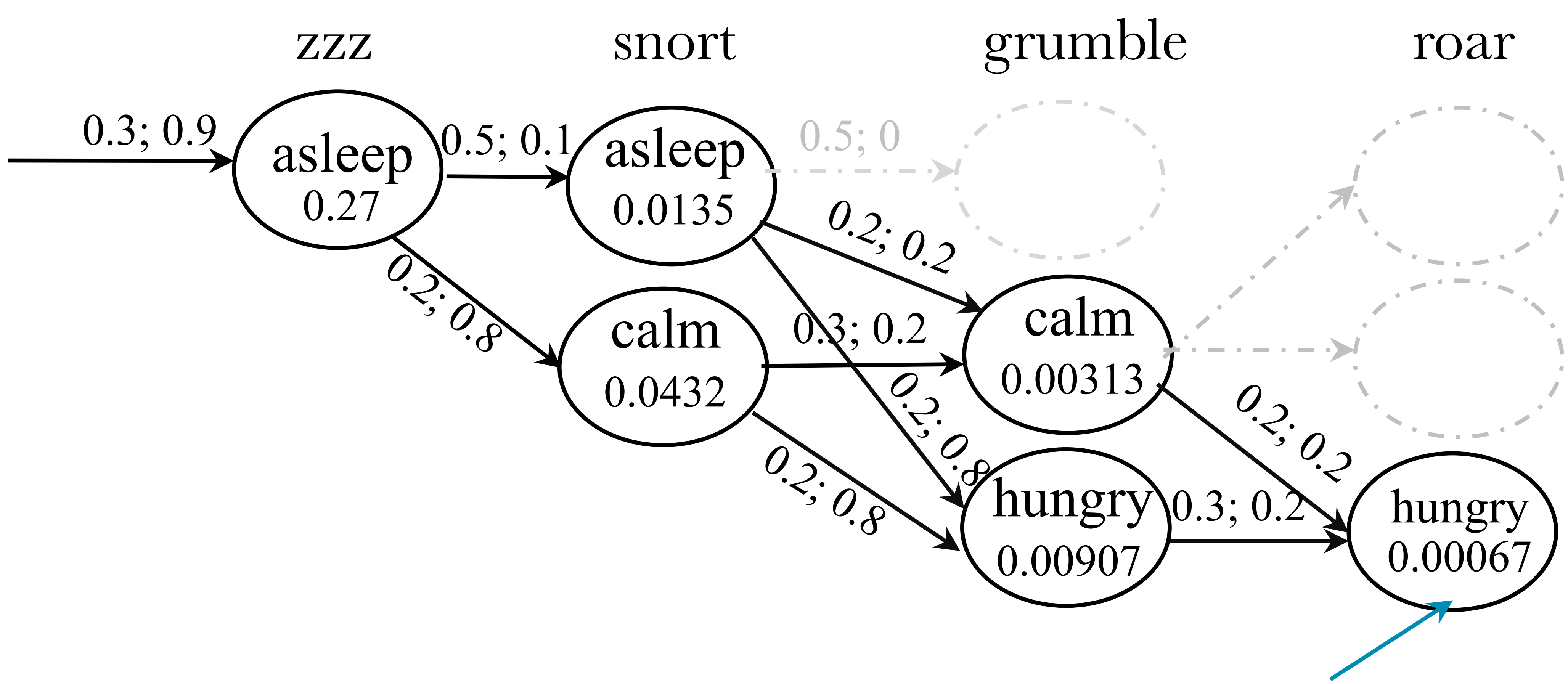
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



0.00313*0.2*0.2 + 0.00907*0.3*0.2

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

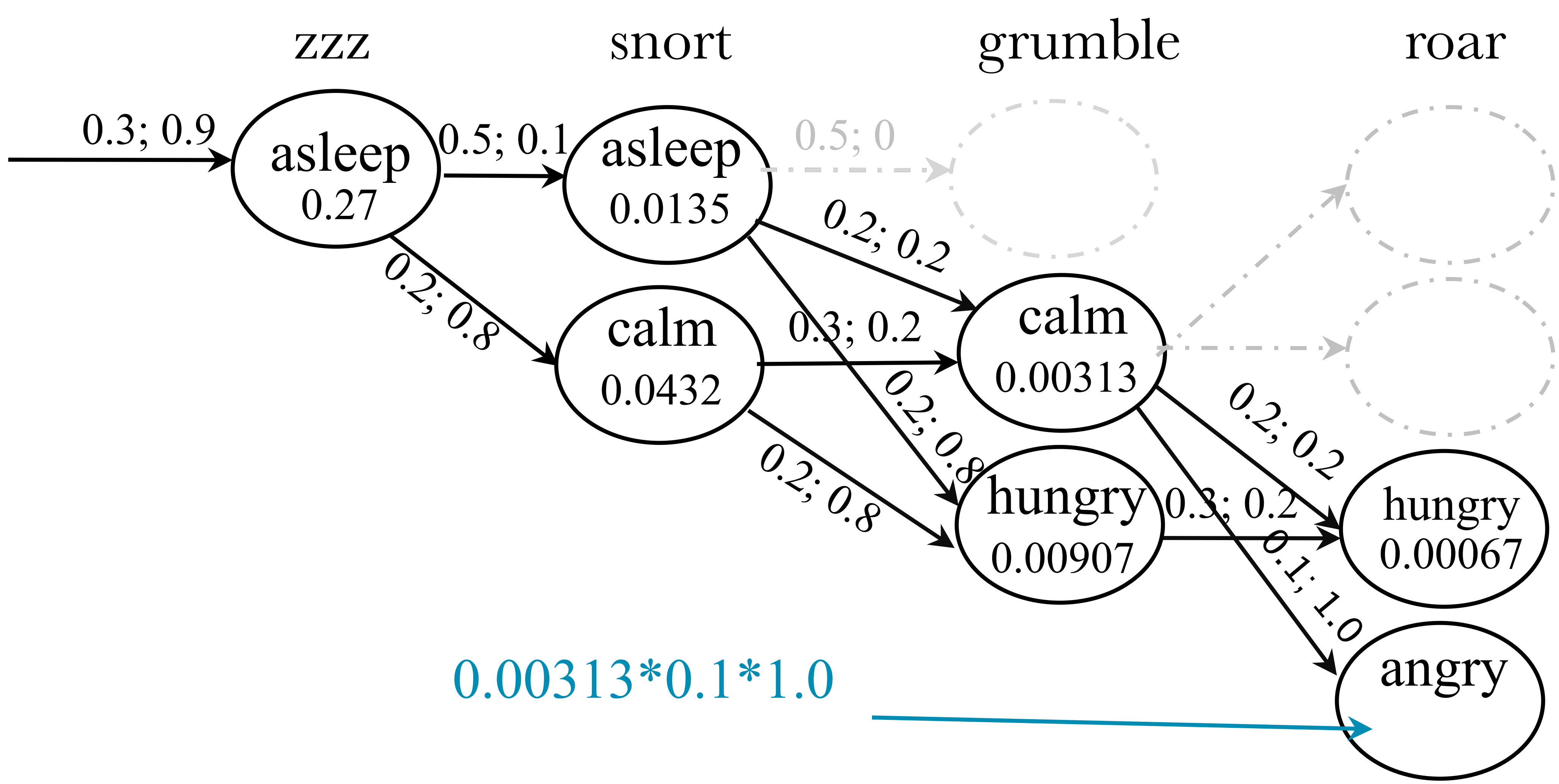
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

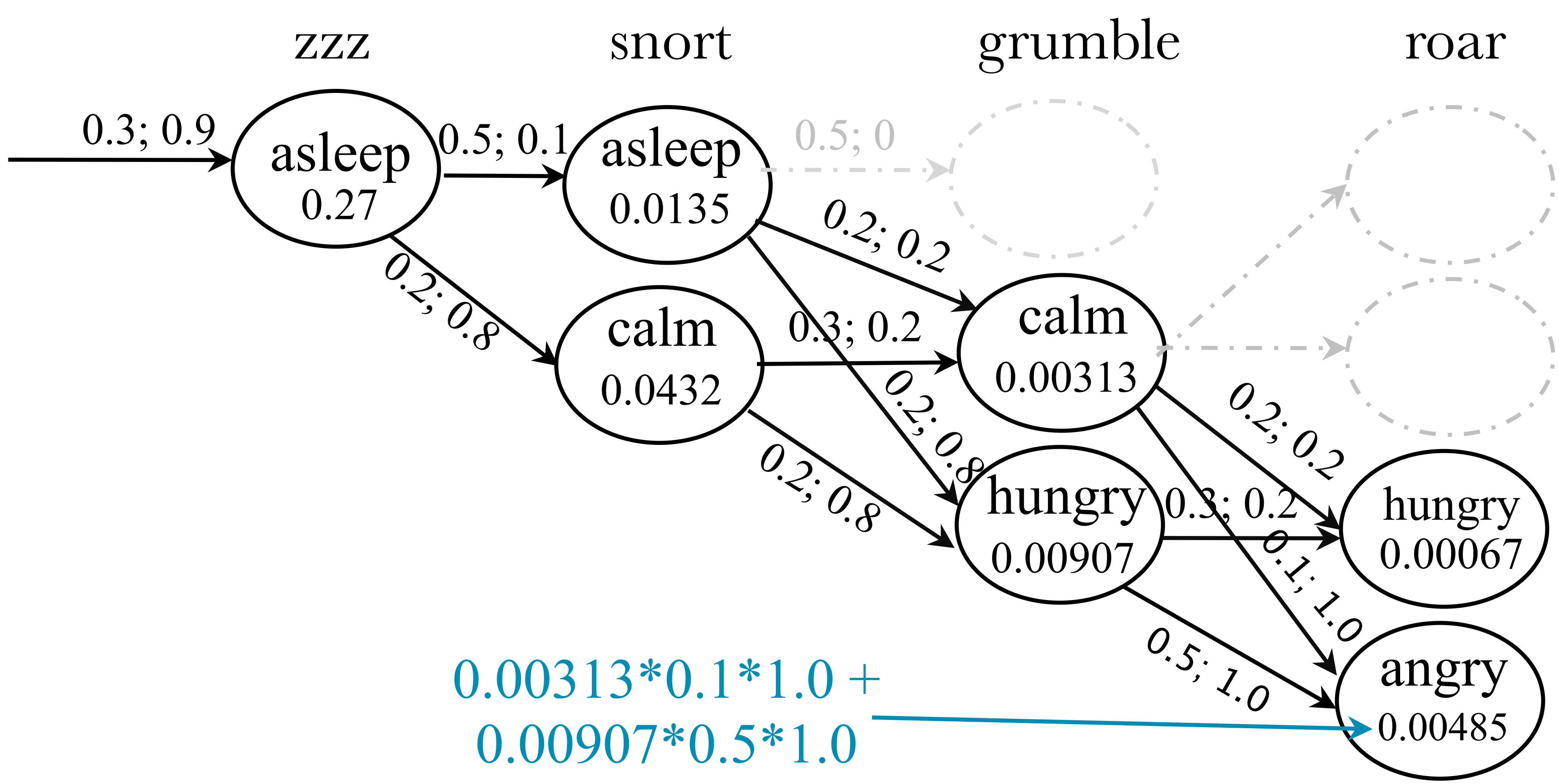
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



0.00313*0.1*1.0 +
0.00907*0.5*1.0

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

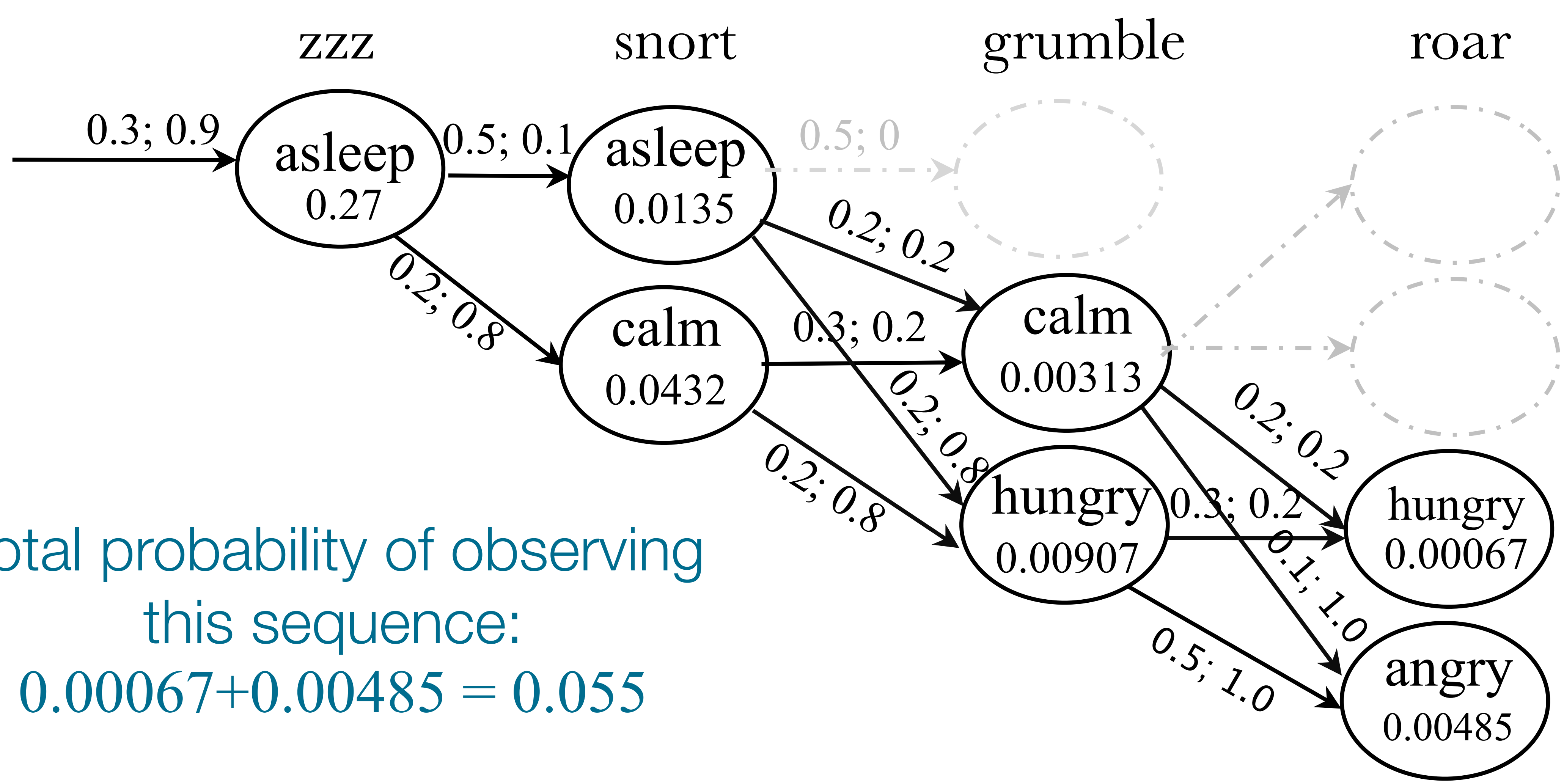
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Total probability of observing this sequence:
$0.00067+0.00485 = 0.055$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8

zzz snort grumble roar

This is called the forward algorithm, because we calculated incrementally moving forward in time

Three fundamental questions for HMMs

- Given a model $M = (A,B,\Pi)$, how do we efficiently compute how likely a certain observation is?
- ➡ Given a sequence of observations Y and a model M, how do we infer the state sequence that best explains the observations?
- Given an observation sequence Y and a space of possible models found by varying the model parameters $M = (A,B,\Pi)$, how do we find the model that best explains the observed data?

Forward* algorithm

Viterbi* algorithm

Baum-Welch** algorithm

Three fundamental questions for HMMs

- Given a model $M = (A,B,\Pi)$, how do we efficiently compute how likely a certain observation is?

➡ Given a sequence of observations Y and a model M, how do we infer the state sequence that best explains the observations?

Forward* algorithm

Viterbi* algorithm

Idea: what if we maximise as we go through the trellis, rather than sum up all of the states?

Viterbi algorithm

An algorithm for efficiently calculating the most likely path through an HMM, given a sequence of observations

Incremental: at each observation step, you find the most likely path until that point

Complexity is $O(N^2T)$, assuming a fully connected model – a big improvement over $O(N^T)$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

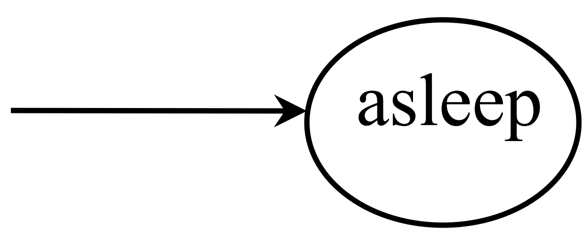
State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8

zzz snort grumble roar



There is only one state that outputs zzz

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

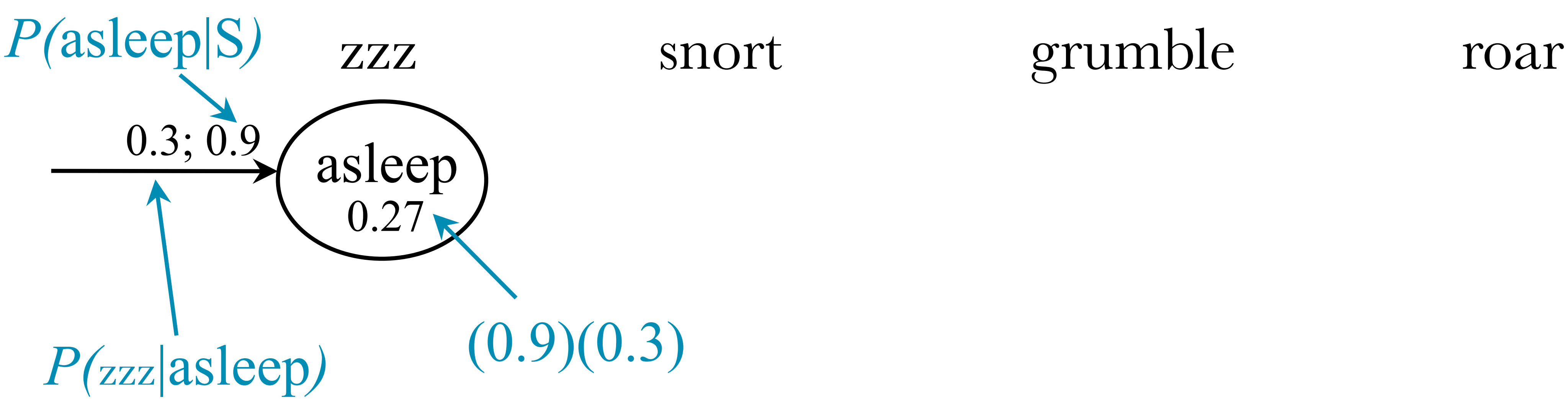
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



There is only one state that outputs zzz

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

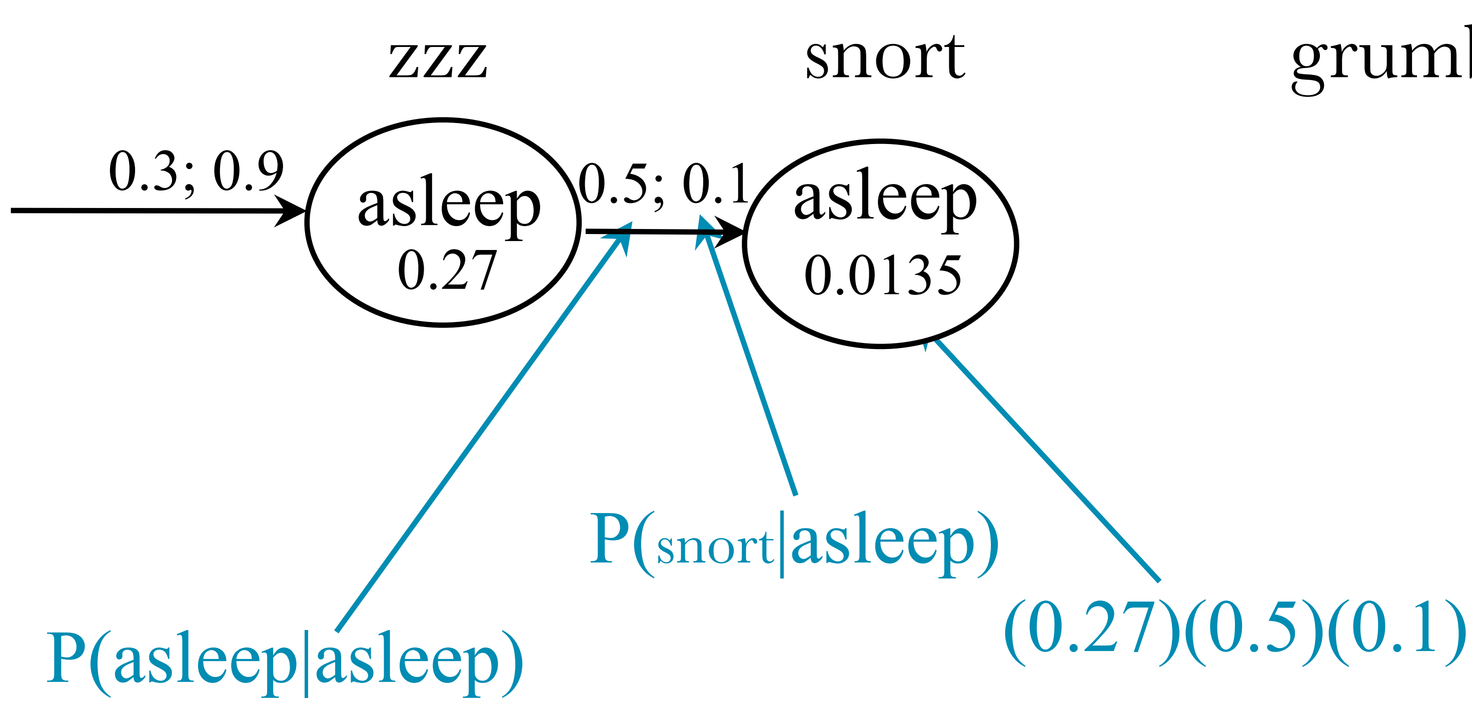
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

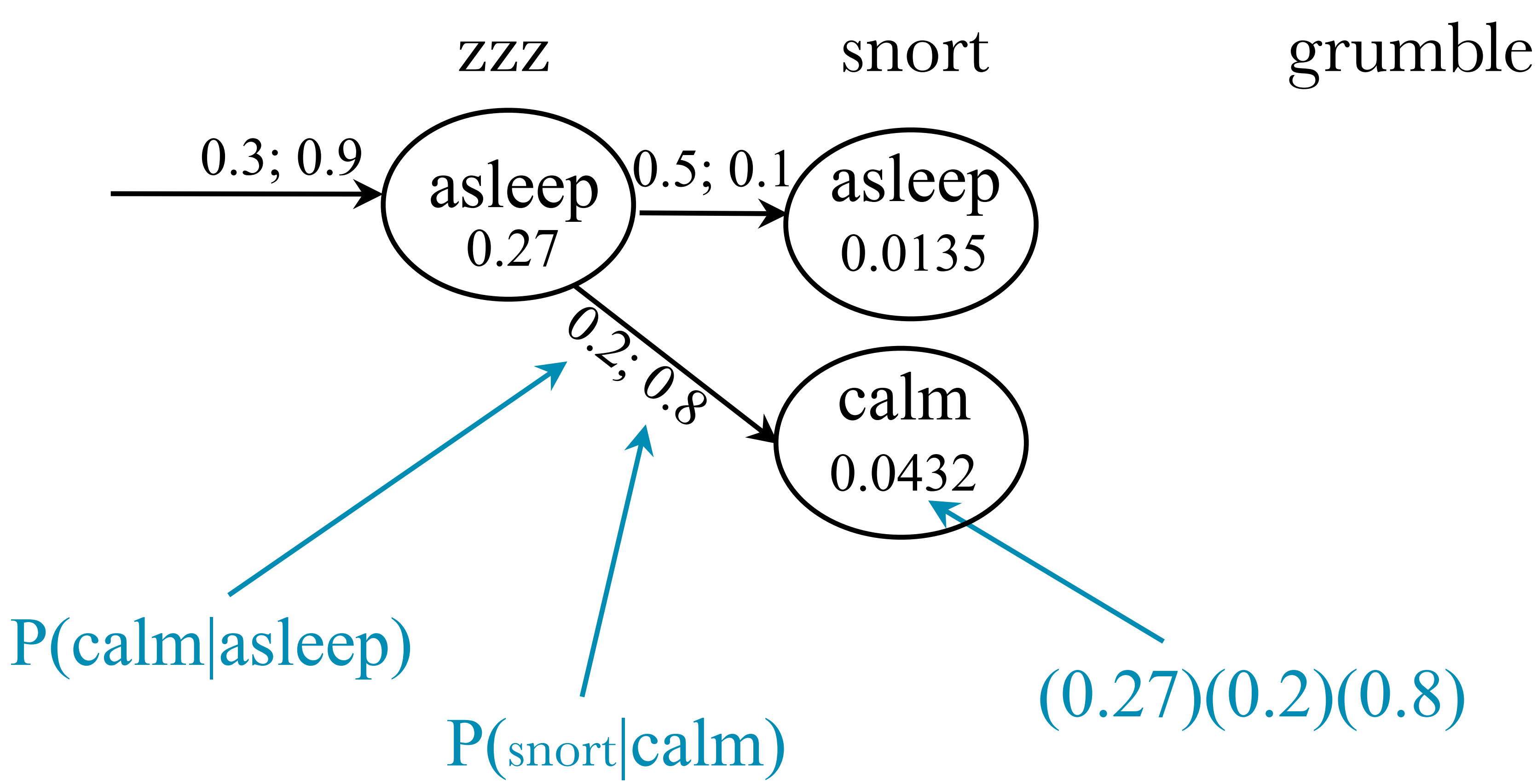
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

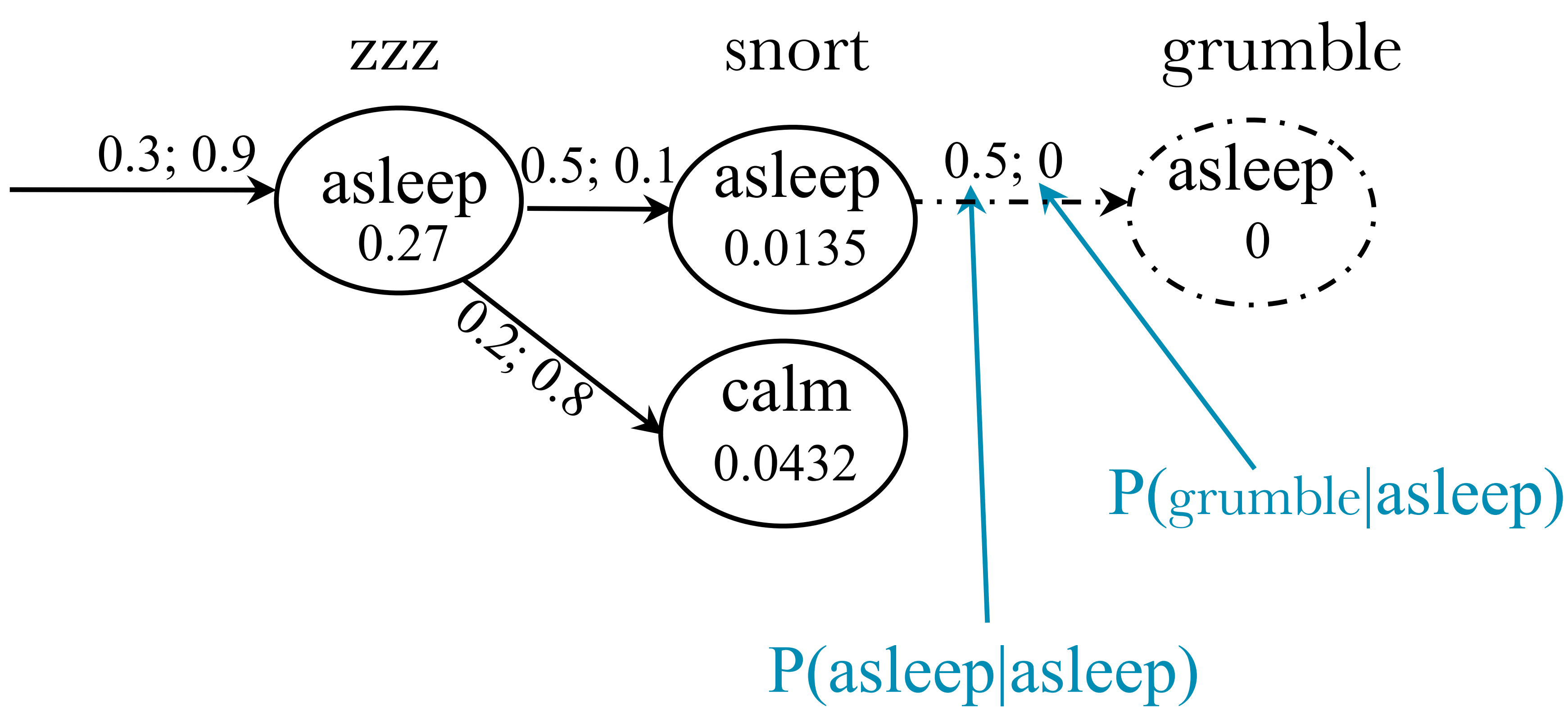
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

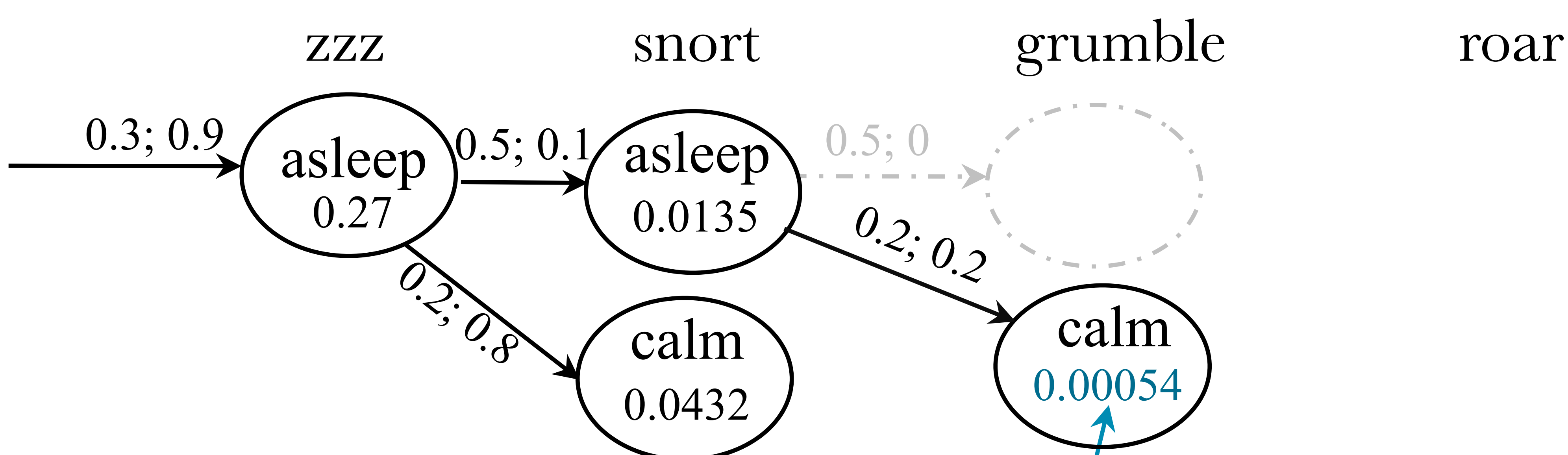
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



$(0.0135)(0.2)(0.2) = 0.00054$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

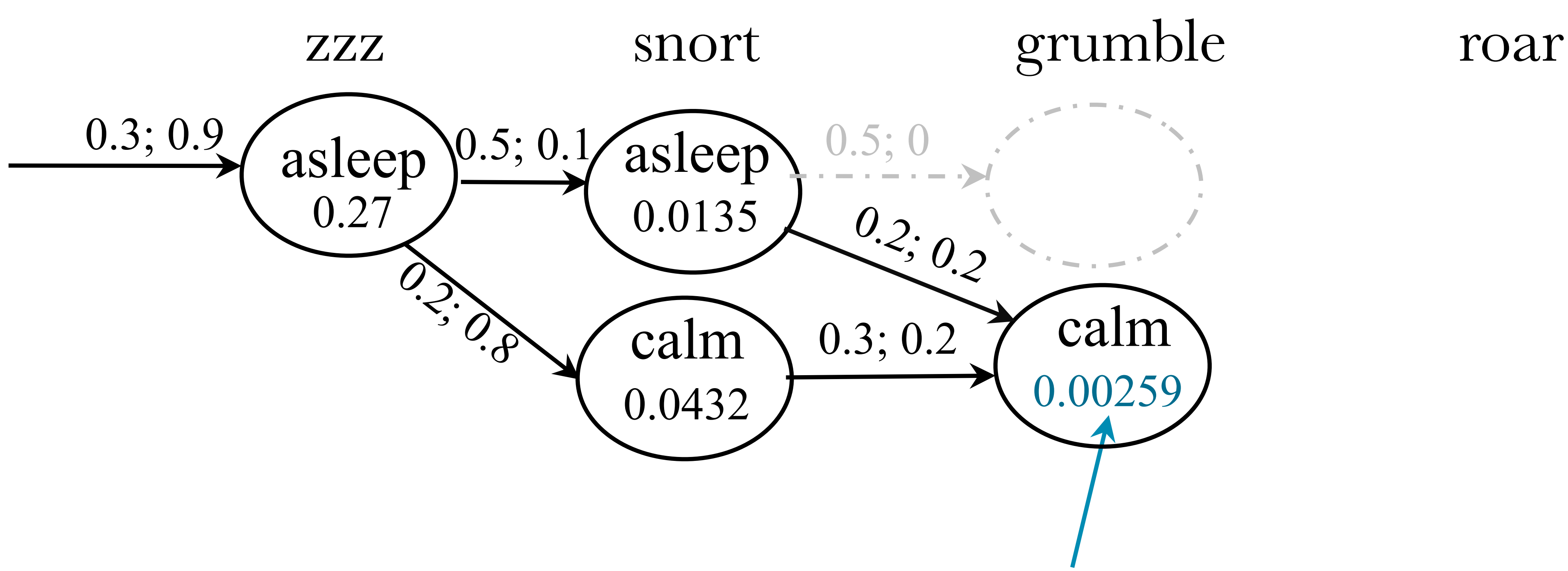
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



$(0.0135)(0.2)(0.2) = 0.00259$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

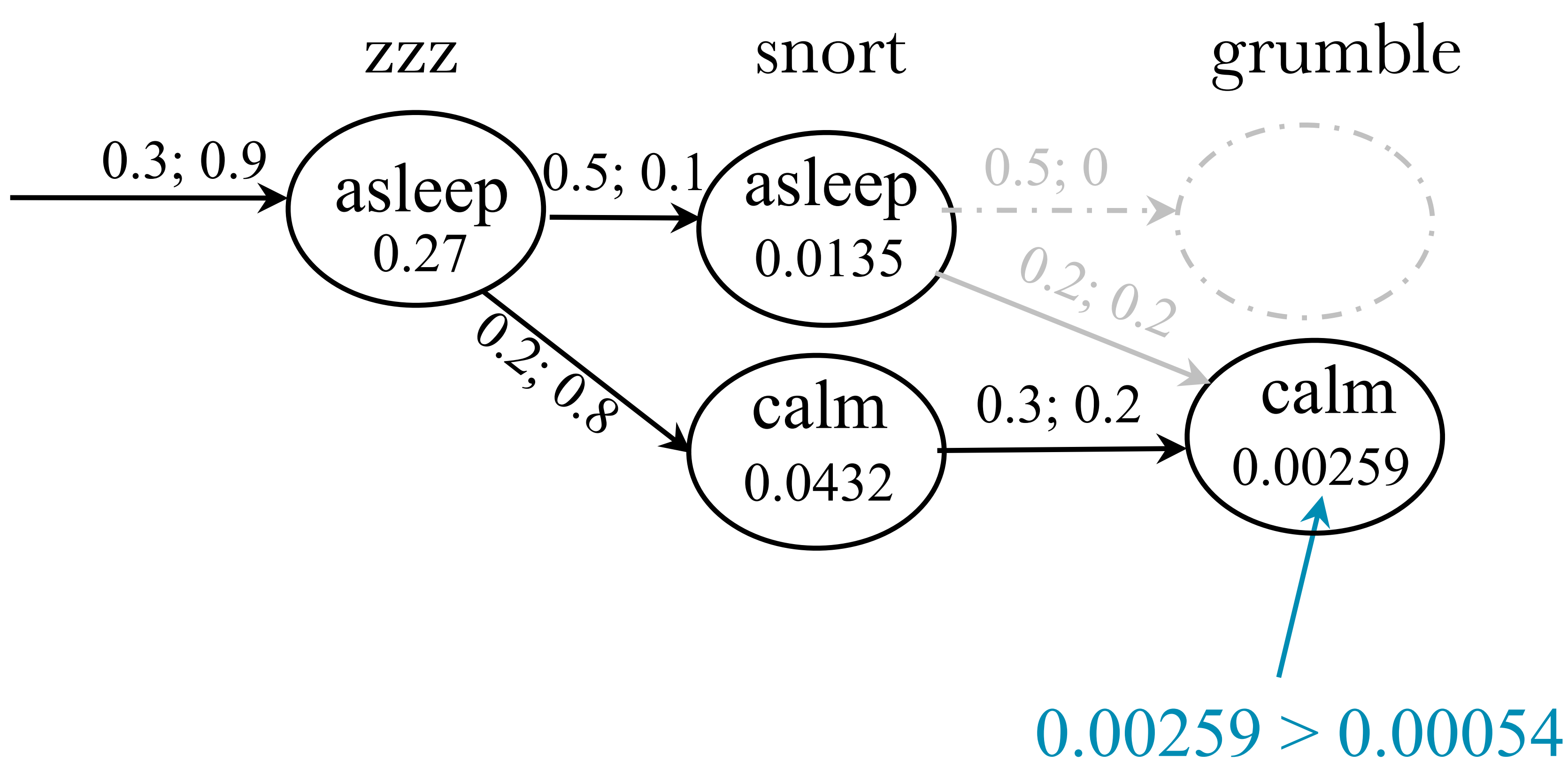
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see “zzz snort grumble roar”?

Initial state probabilities Π:

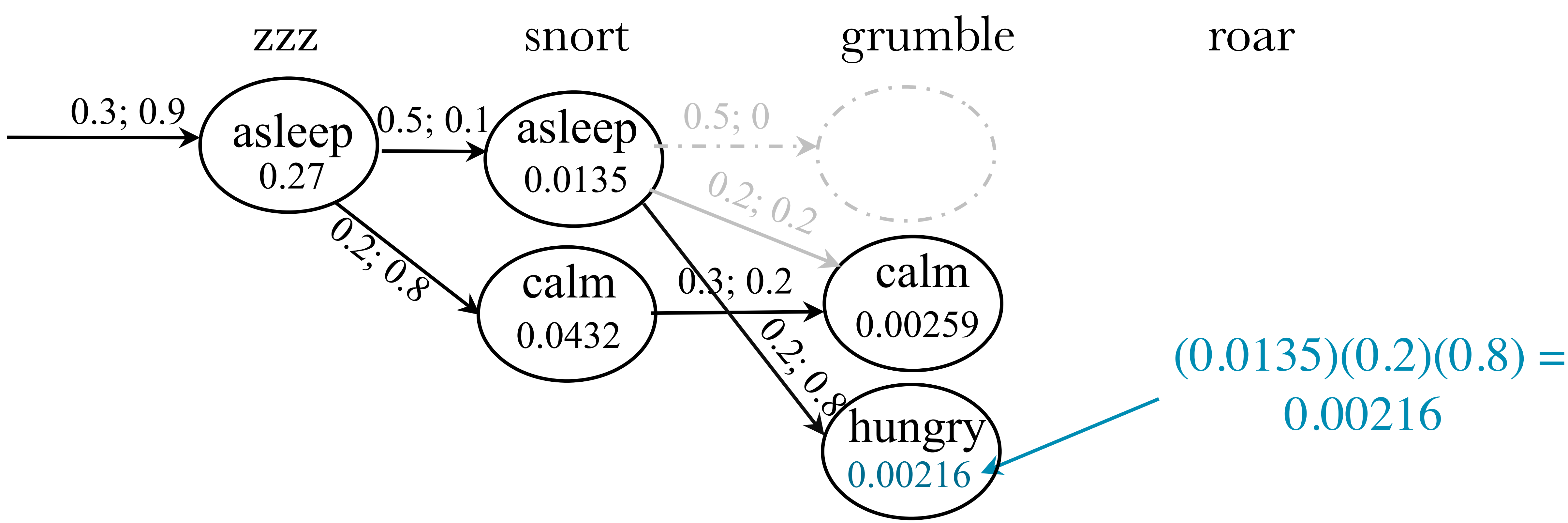
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

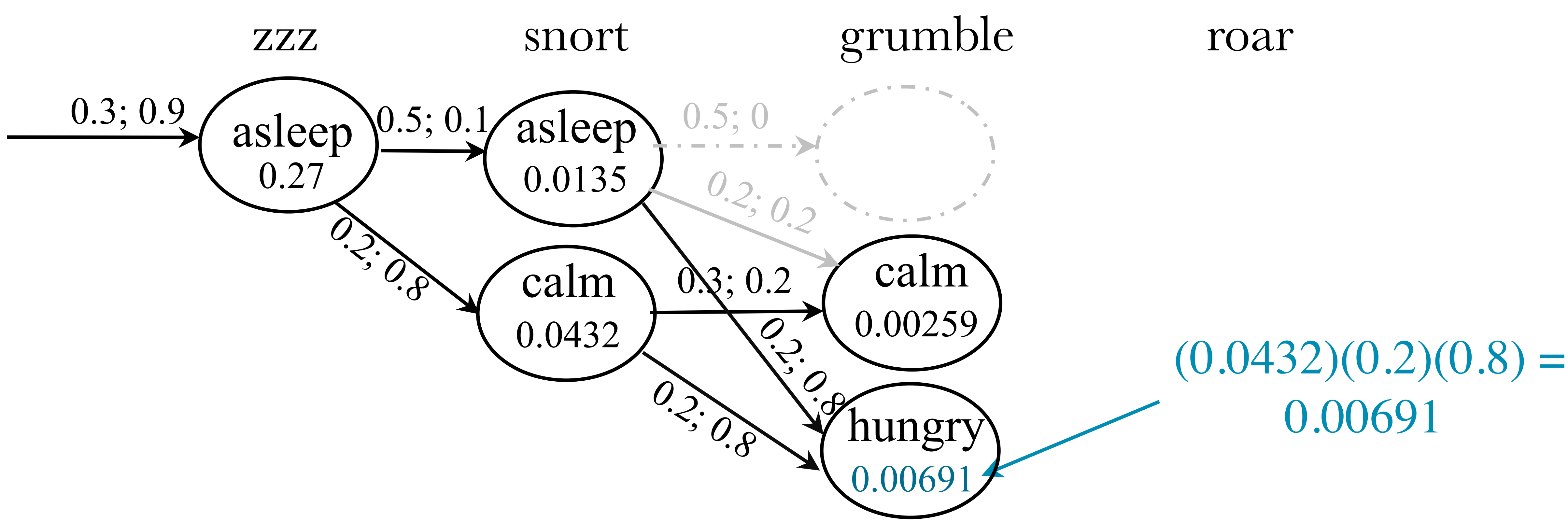
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

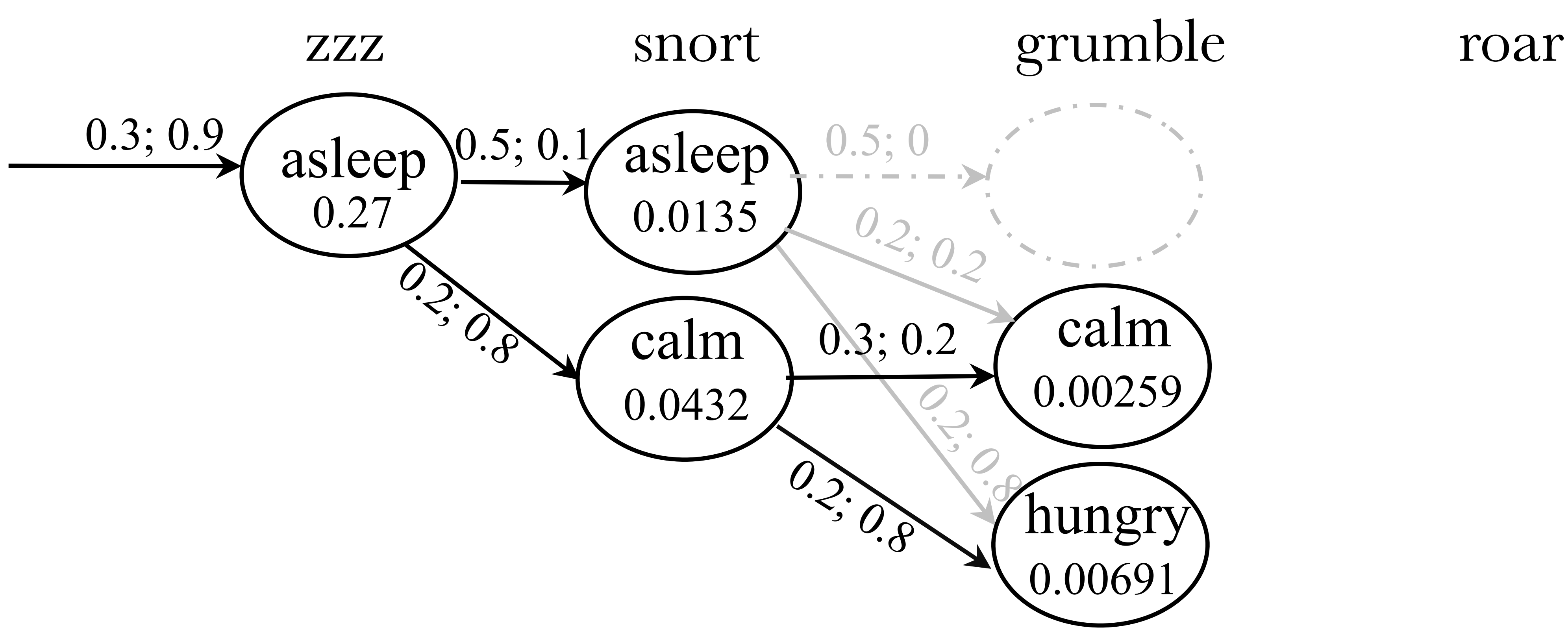
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



0.00691 > 0.00216

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

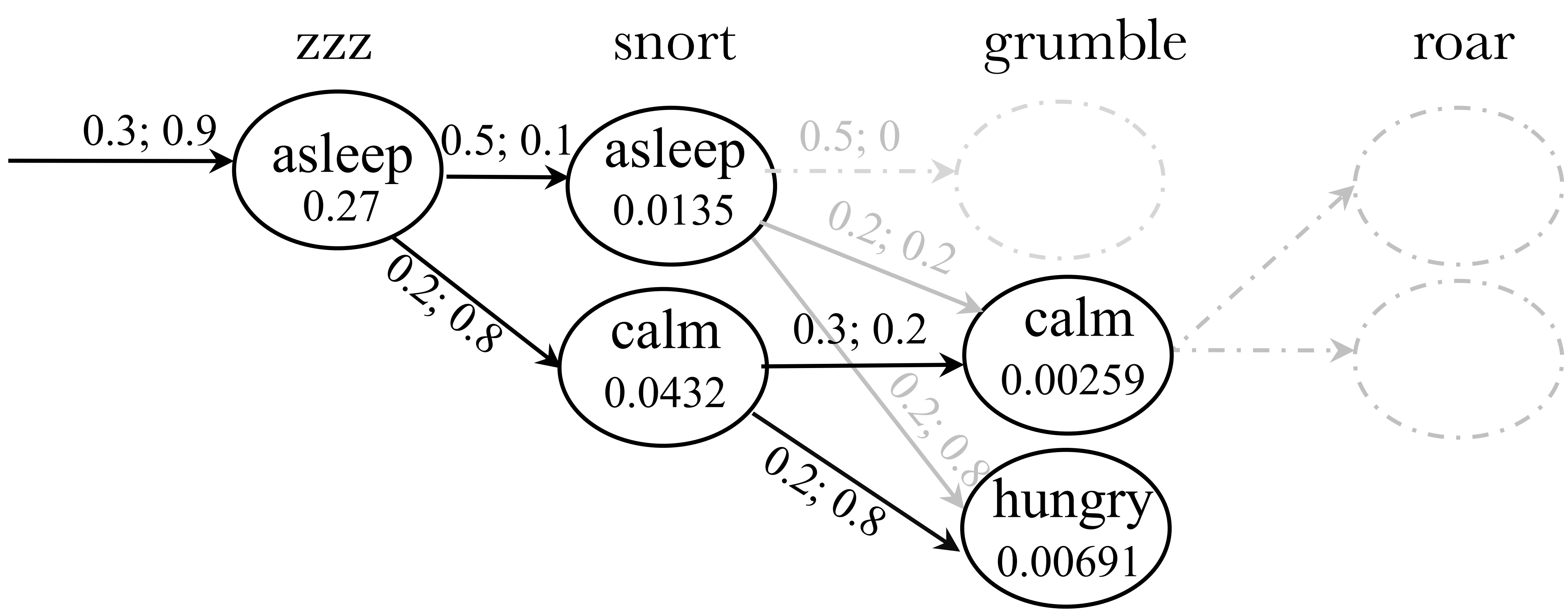
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

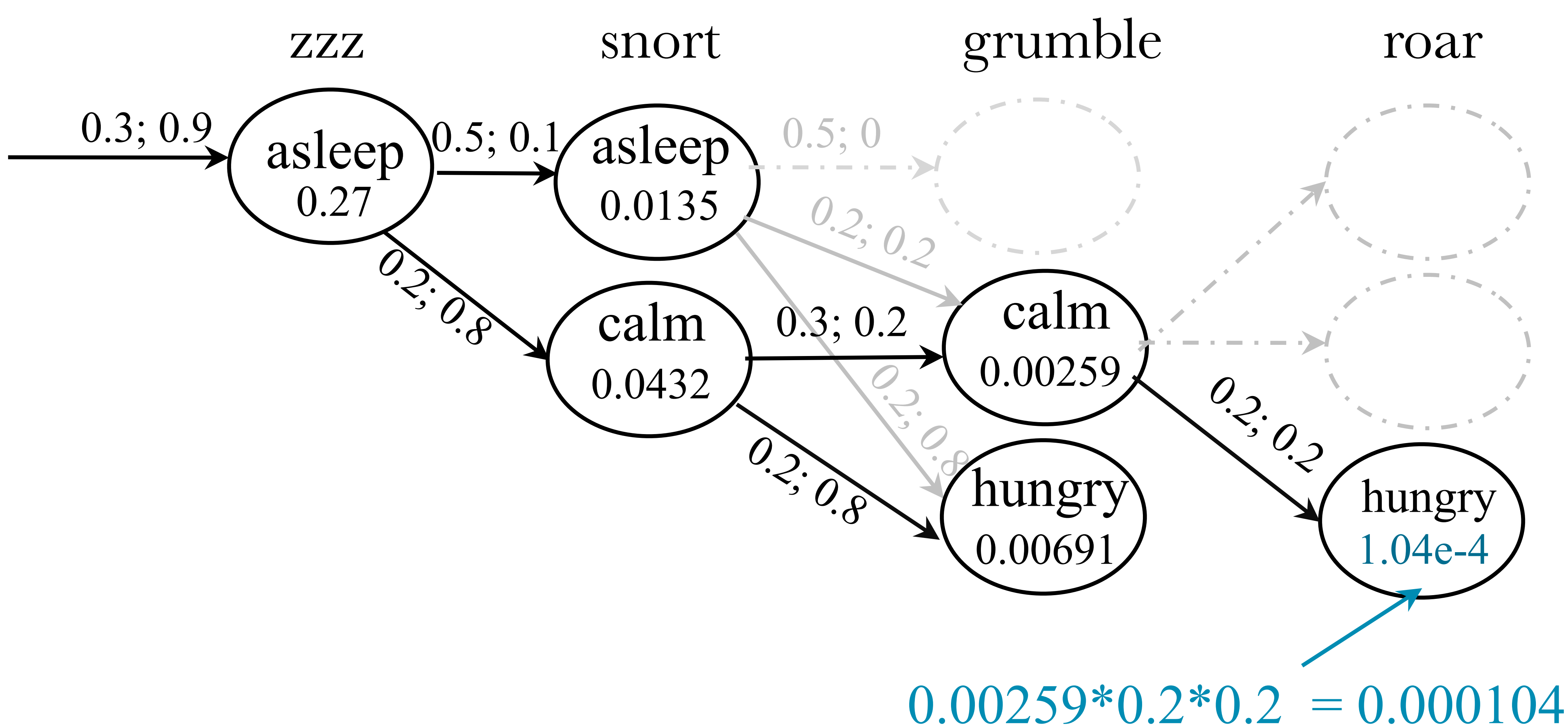
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

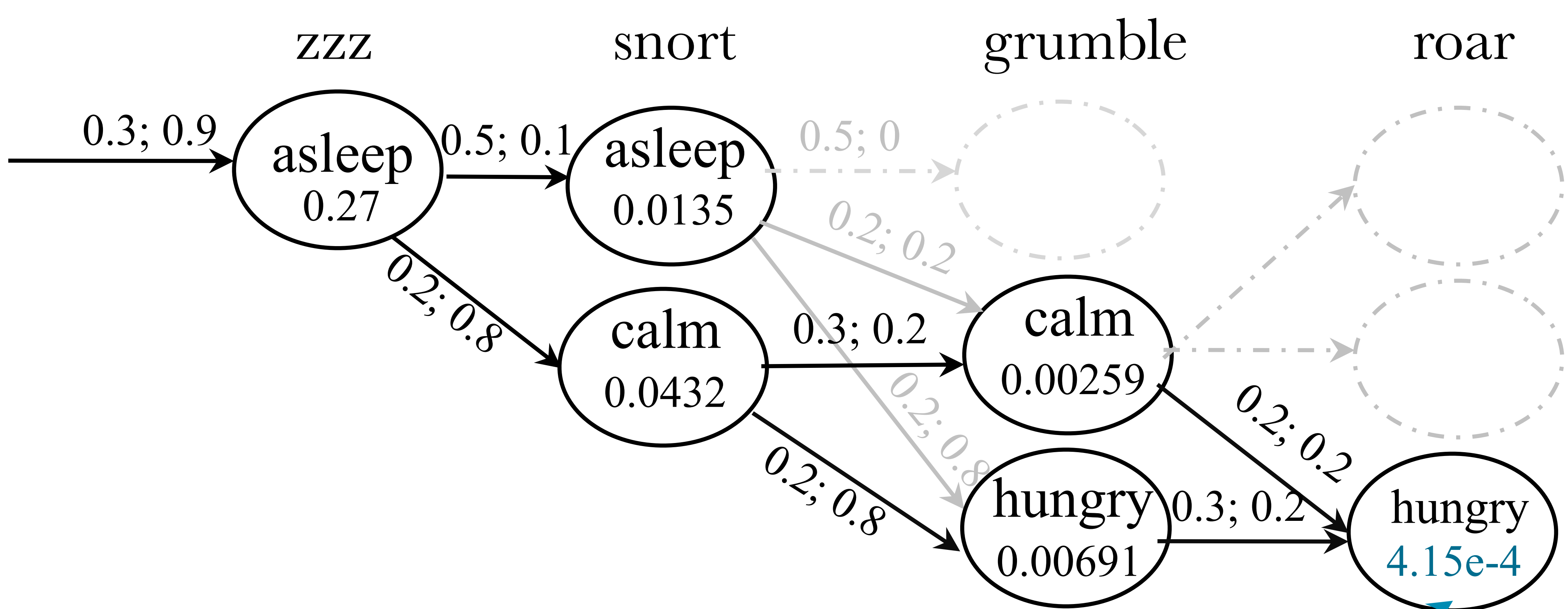
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



$0.00691*0.3*0.2 = 0.000415$

Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

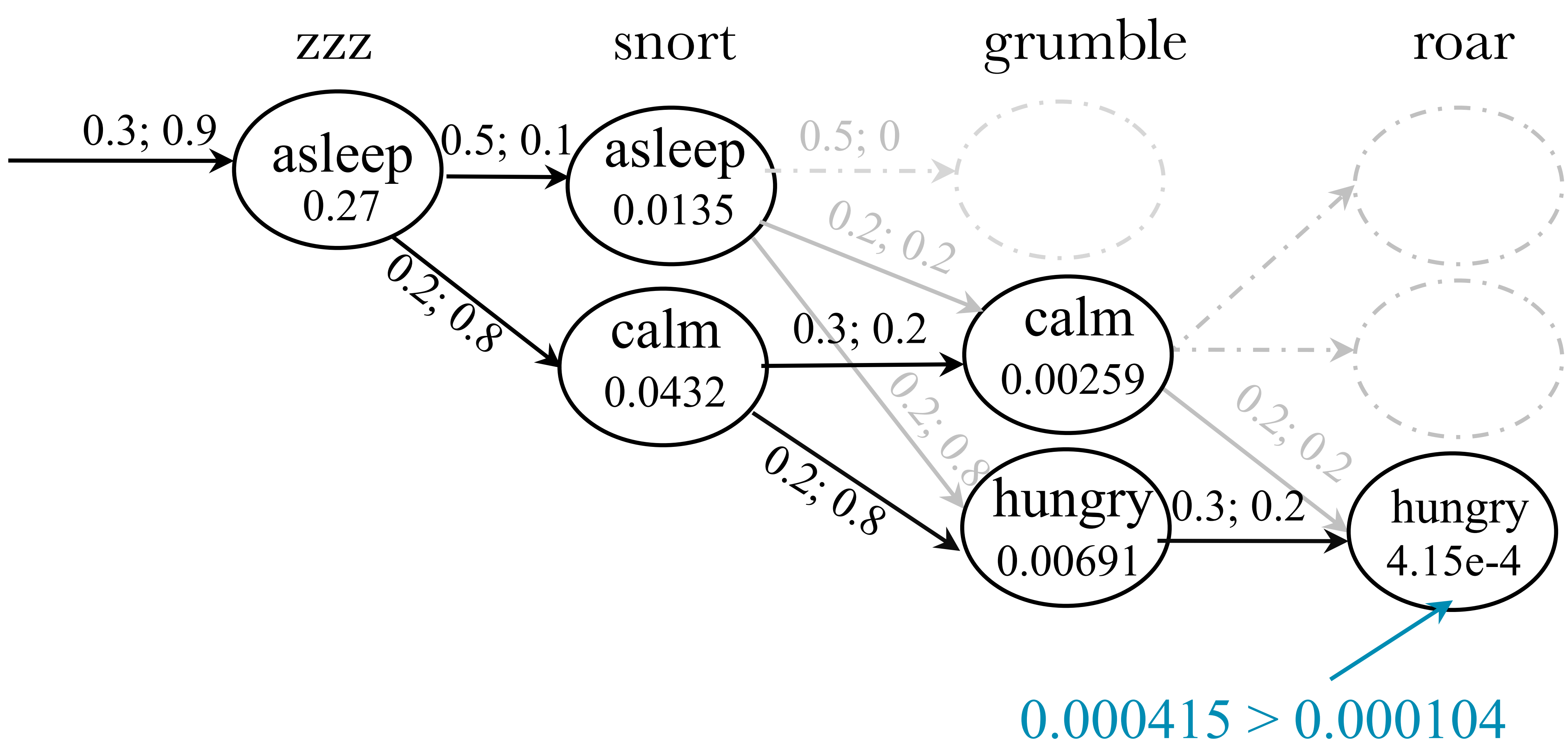
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

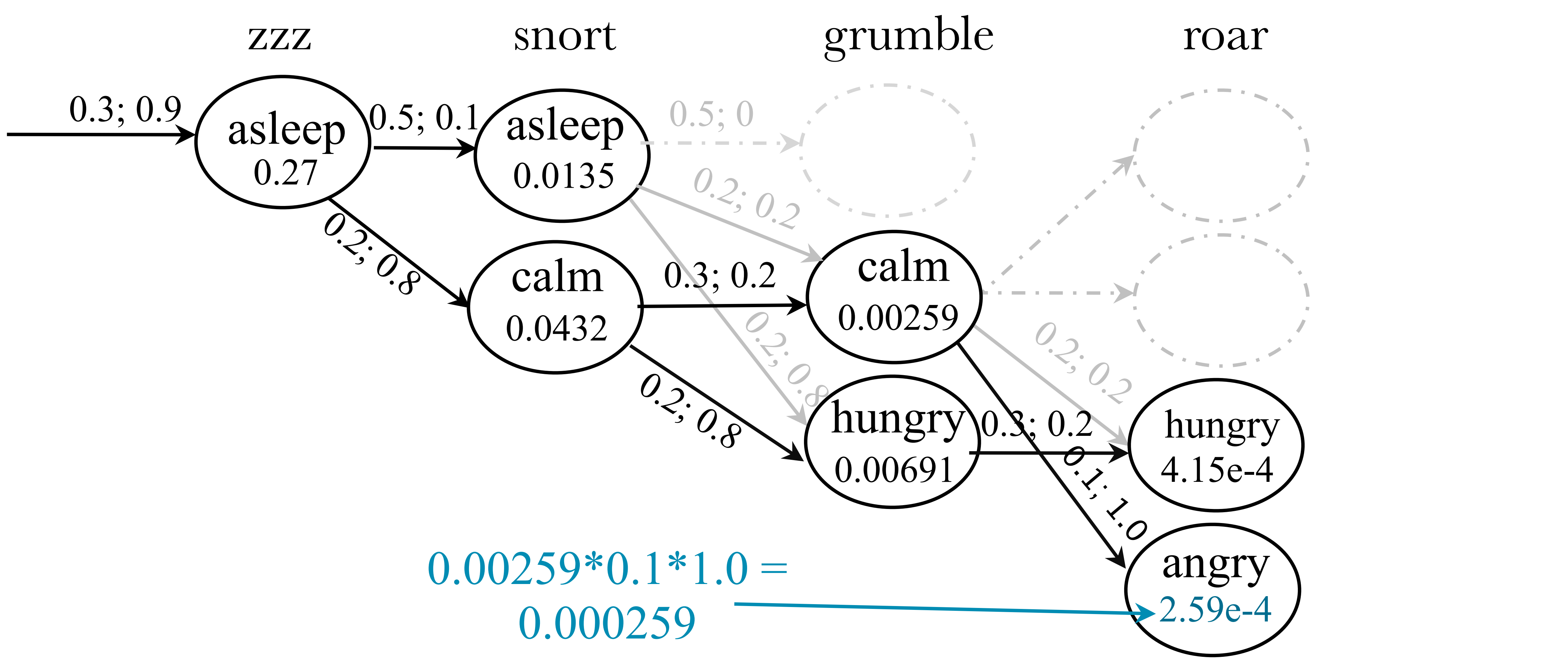
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

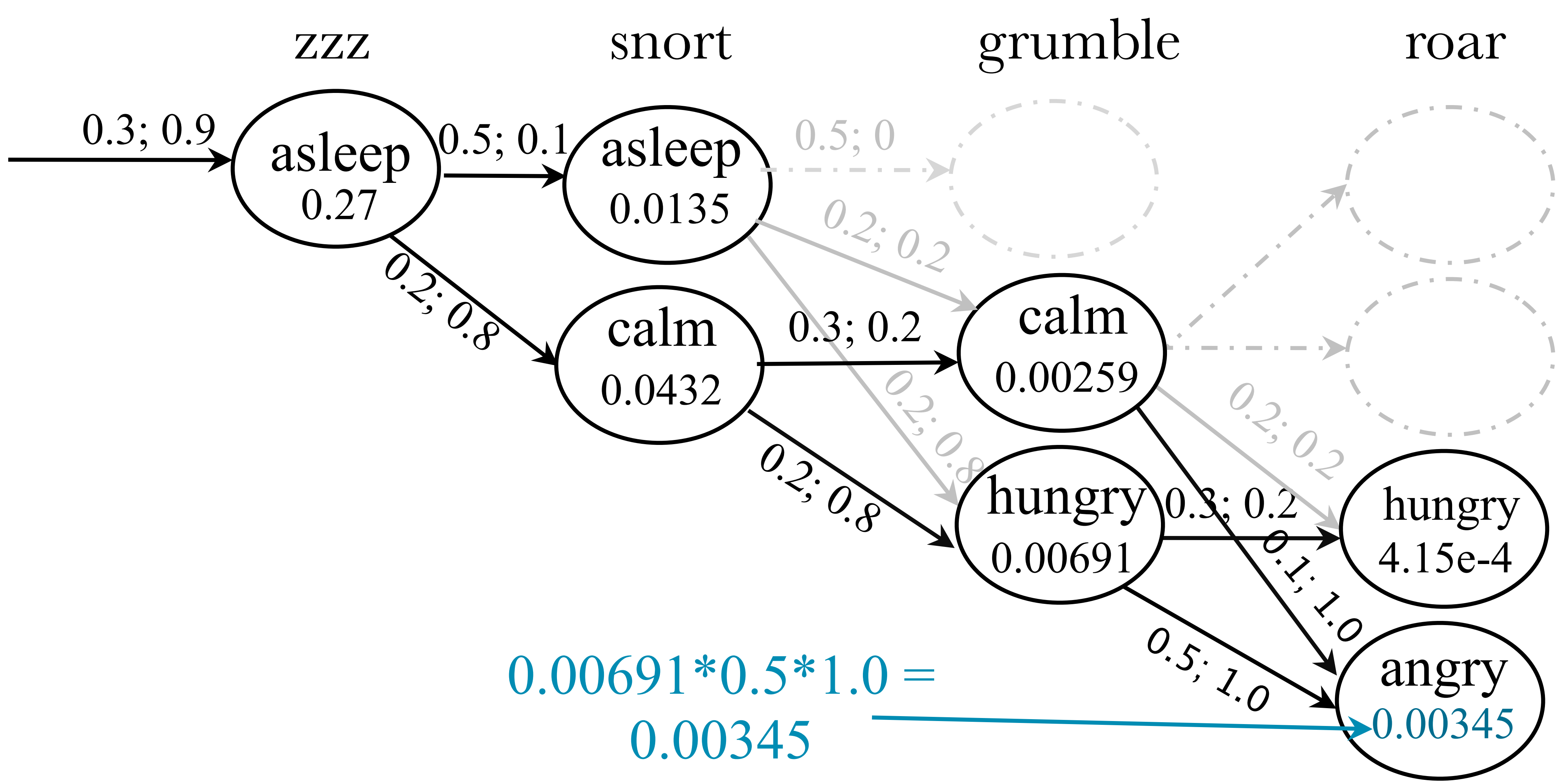
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



Example: Mitee the warrior

How likely are you to see "zzz snort grumble roar"?

Initial state probabilities Π:

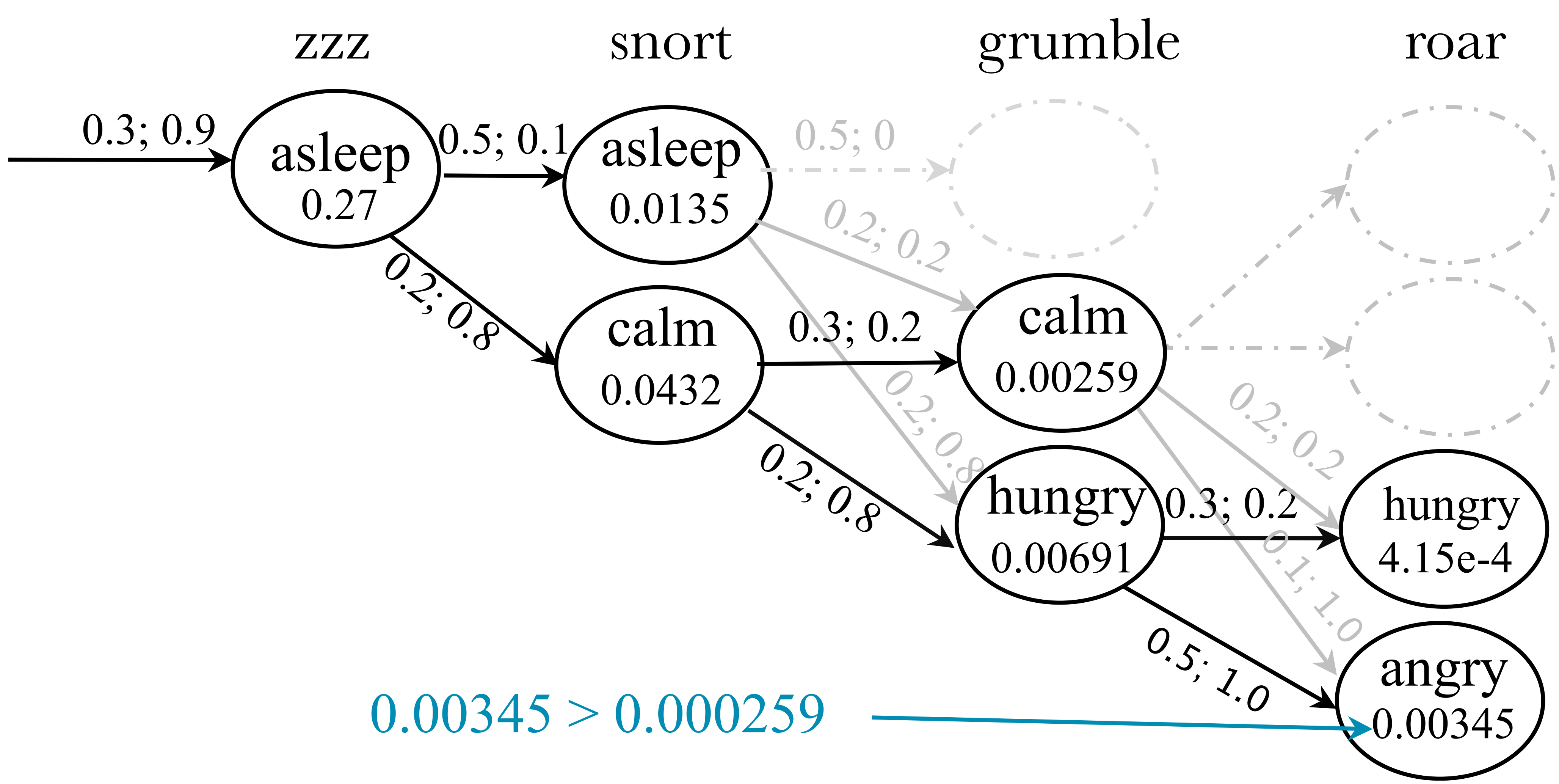
Asleep	Calm	Angry	Hungry
0.3	0.3	0.2	0.2

State transition matrix A:

	Asleep	Calm	Angry	Hungry
Asleep	0.5	0.2	0.1	0.2
Calm	0.4	0.3	0.1	0.2
Angry	0.1	0.2	0.6	0.1
Hungry	0.1	0.1	0.5	0.3

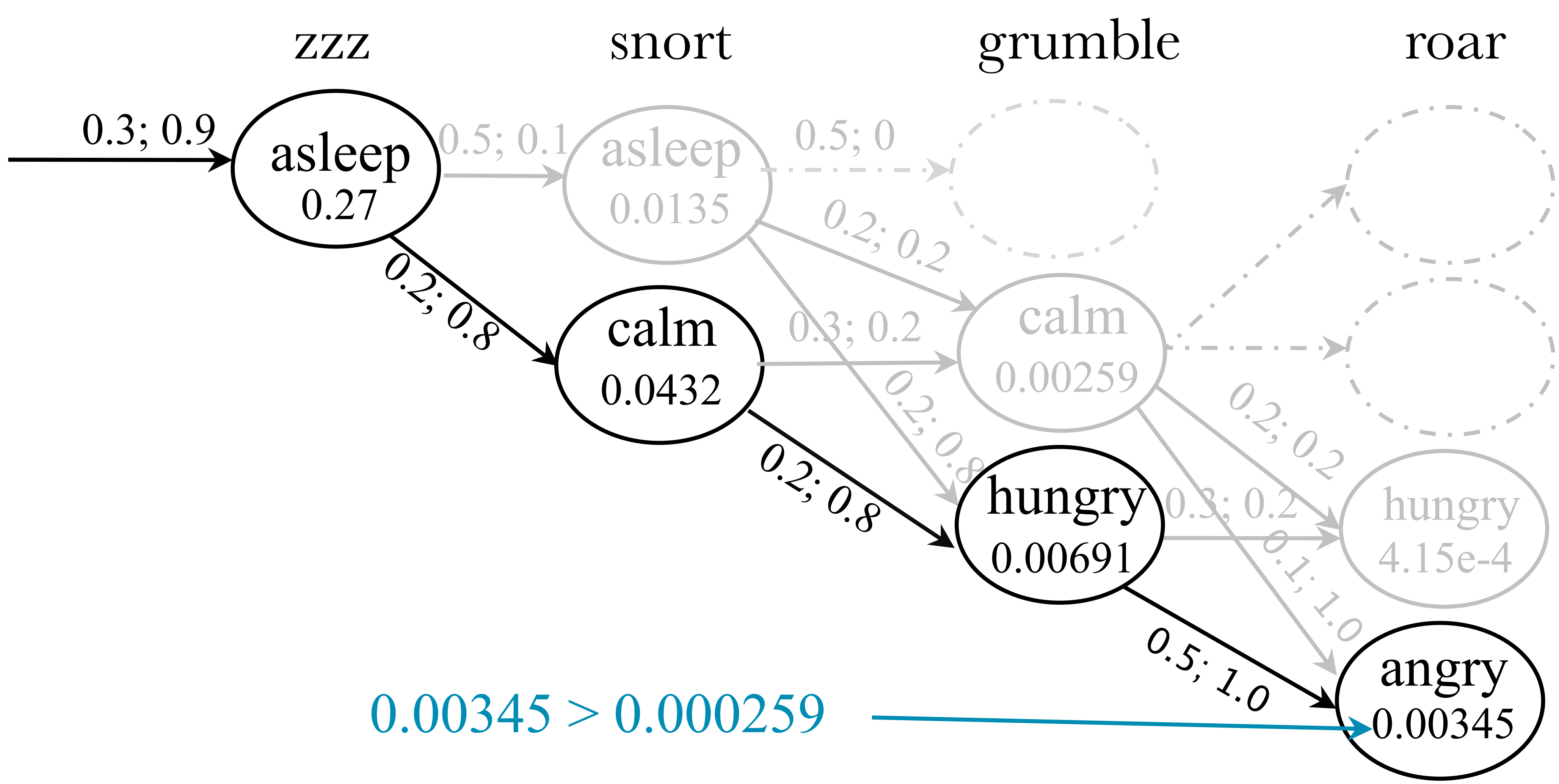
Output symbol matrix B:

	Roar	Zzz	Snort	Grumble
Asleep	0.0	0.9	0.1	0.0
Calm	0.0	0.0	0.8	0.2
Angry	1.0	0.0	0.0	0.0
Hungry	0.2	0.0	0.0	0.8



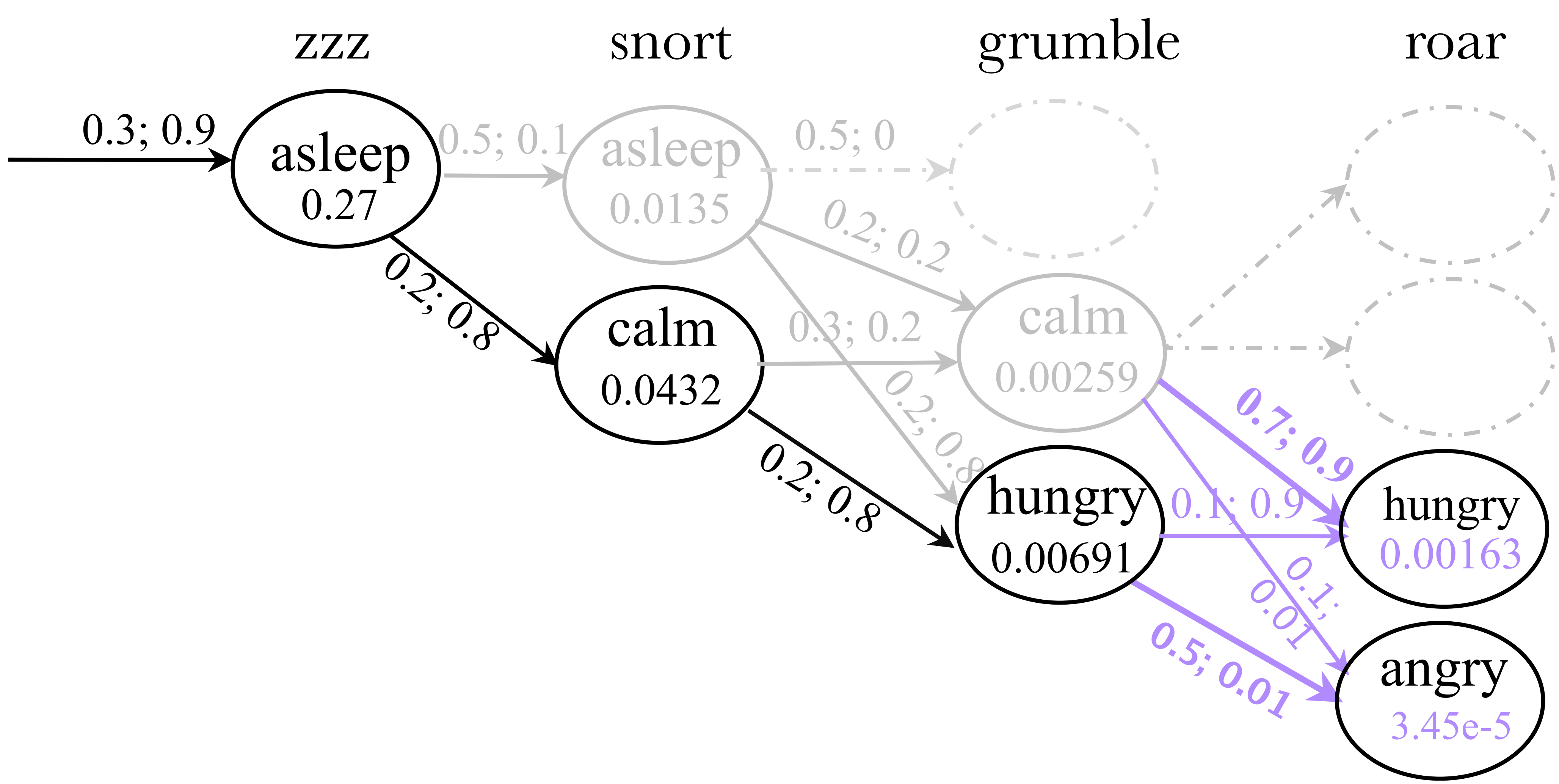
Given this, is the most likely state sequence just the one whose states are most probable at every point in time?

This worked out…



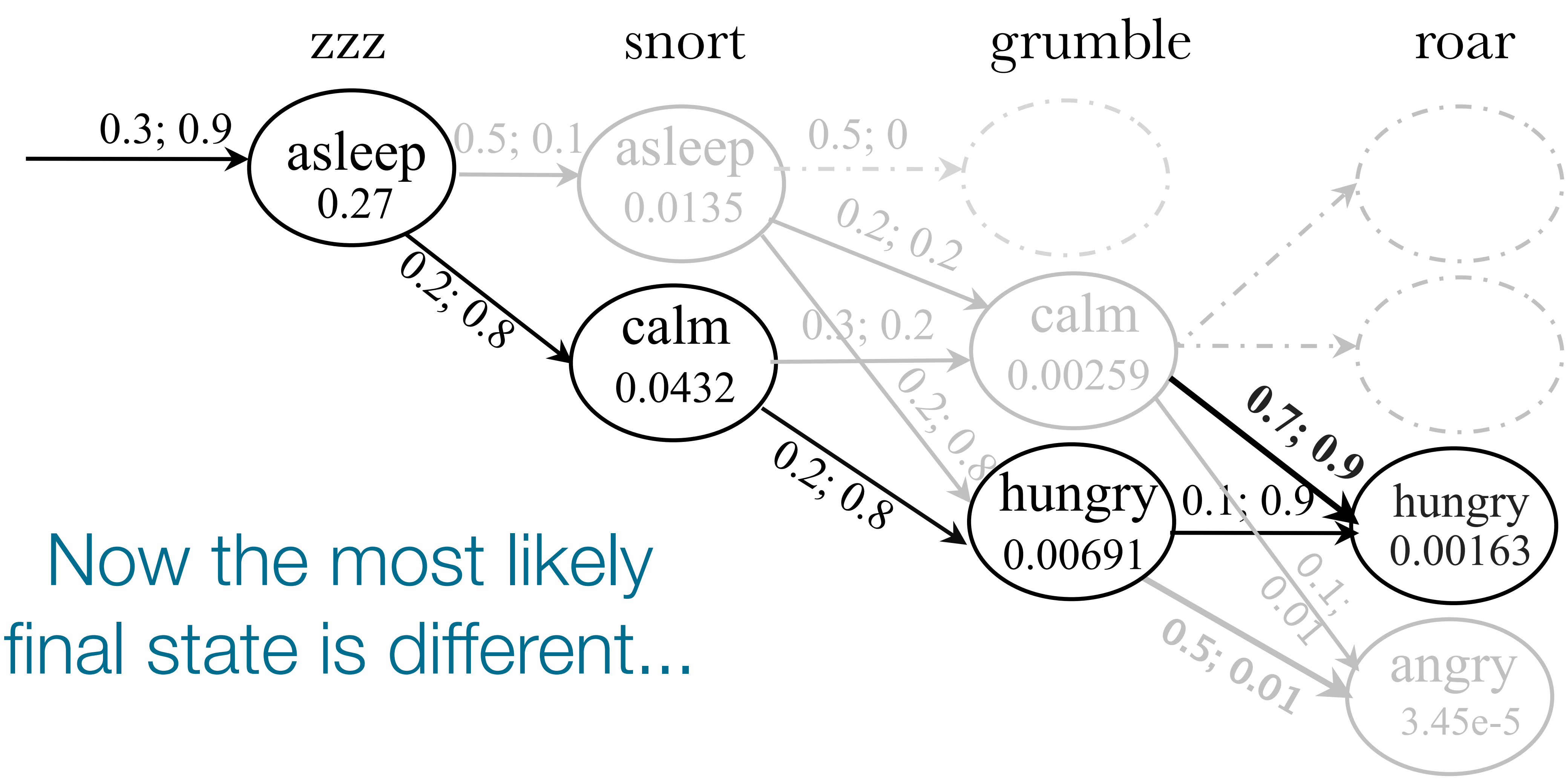
Given this, is the most likely state sequence just the one whose states are most probable at every point in time?

But imagine the transition probabilities were slightly different



Given this, is the most likely state sequence just the one whose states are most probable at every point in time?

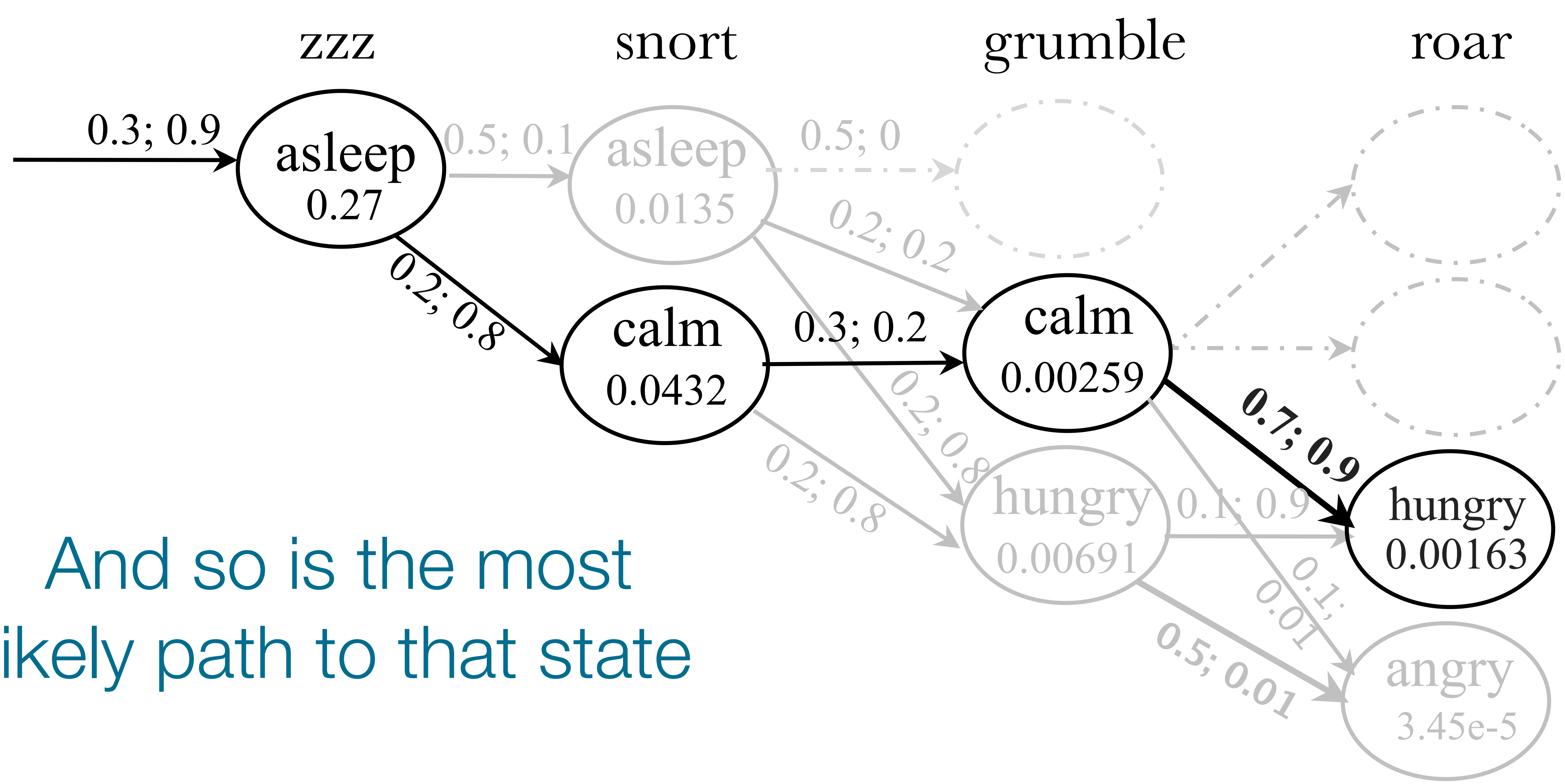
But imagine the transition probabilities were slightly different



Now the most likely final state is different...

Given this, is the most likely state sequence just the one whose states are most probable at every point in time?

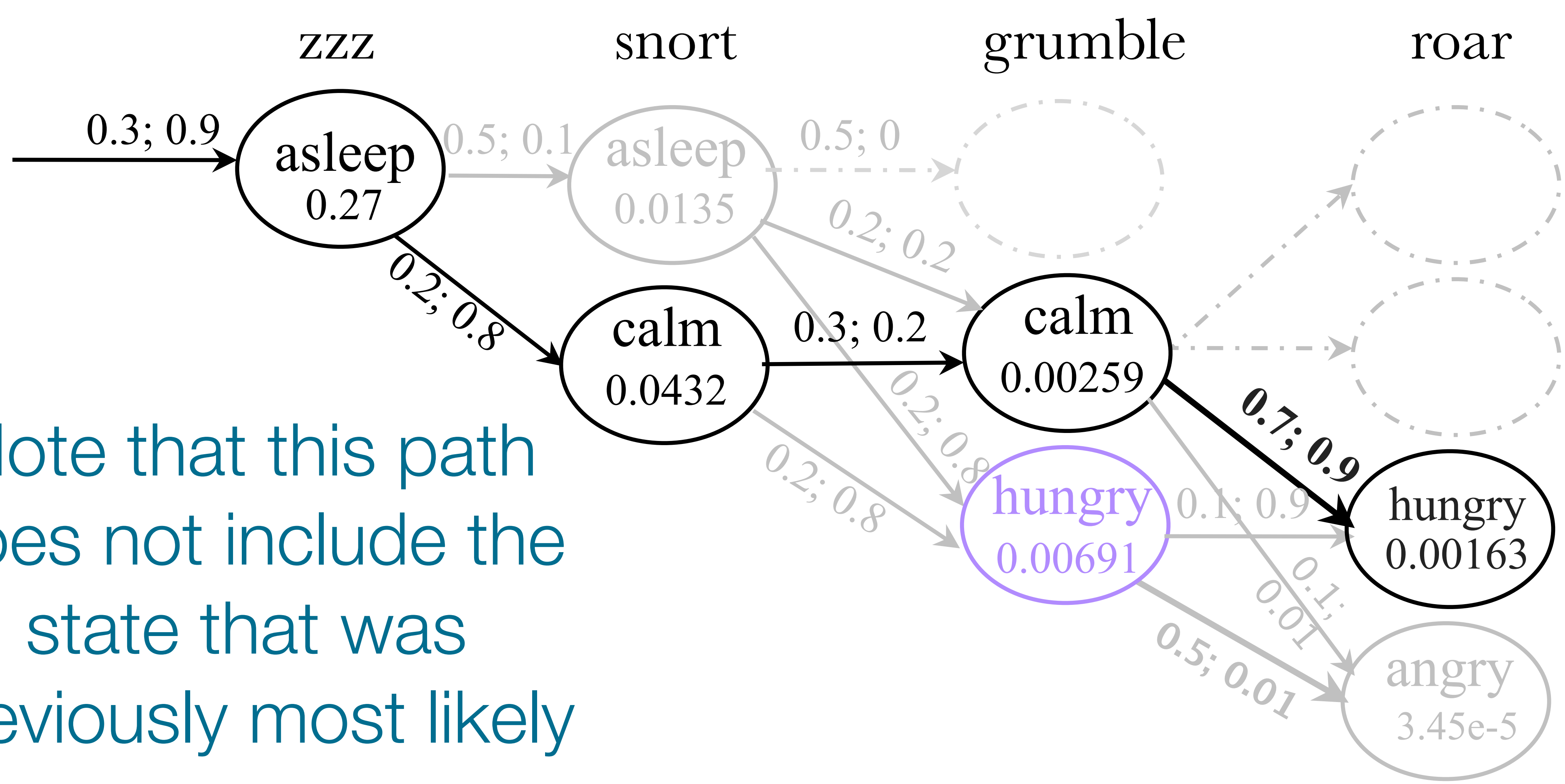
But imagine the transition probabilities were slightly different



And so is the most likely path to that state

Given this, is the most likely state sequence just the one whose states are most probable at every point in time?

But imagine the transition probabilities were slightly different



Note that this path does not include the state that was previously most likely

In order to calculate the most likely state sequence, you need to find the maximum transition at each point, get to the end, and then backtrack through

This algorithm – finding all of the forward probabilities (maxima, not sums), and then backtracking – is called the Viterbi algorithm.

Three fundamental questions for HMMs

- Given a model $M = (A,B,\Pi)$, how do we efficiently compute how likely a certain observation is?

  Forward* algorithm

- Given a sequence of observations Y and a model M, how do we infer the state sequence that best explains the observations?

  Viterbi* algorithm

- ➡ Given an observation sequence Y and a space of possible models found by varying the model parameters $M = (A,B,\Pi)$, how do we find the model that best explains the observed data?

  Baum-Welch algorithm**

* You should be able to implement this; ** You don't need to be able to implement this

Three fundamental questions for HMMs

I'll give the main idea of how it works, but not all of the nitty-gritty detail. You don't need to be able to implement this – I just want to get you started in case you ever want to.

➡ Given an observation sequence Y and a space of possible models found by varying the model parameters $M = (A,B,\Pi)$, how do we find the model that best explains the observed data?

Baum-Welch** algorithm

* You should be able to implement this; ** You don't need to be able to implement this

Baum-Welch algorithm

Basic idea: This is just an EM algorithm! But instead of:

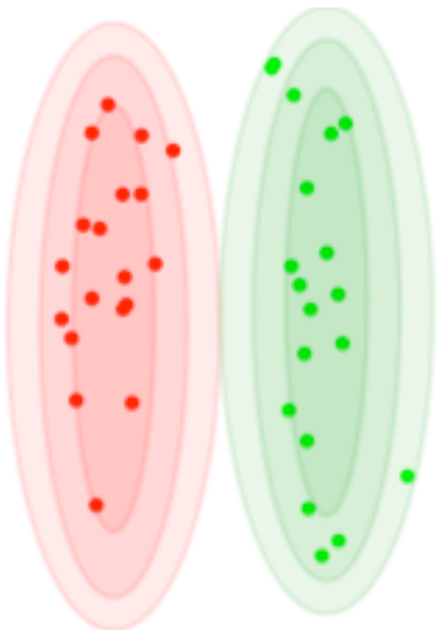
```
Assignment step (E-step):
   Calculate the likelihood of
each data point in each cluster,
assuming the cluster is a
Gaussian with the current mean,
standard deviation, and weight

Update step (M-step):
   Recalculate the means

   Recalculate the standard
       deviations

   Recalculate the weights
```

$$r_k^{(n)} = \frac{w_k \frac{1}{\prod_{i=1}^{I} \sqrt{2\pi}\sigma_i^{(k)}} \exp\left(-\sum_{i=1}^{I} (m_i^{(k)} - x_i^{(n)})^2 / 2(\sigma_i^{(k)})^2\right)}{\sum_{k'} w_{k'} \frac{1}{\prod_{i=1}^{I} \sqrt{2\pi}\sigma_i^{(k')}} \exp\left(-\sum_{i=1}^{I} (m_i^{(k')} - x_i^{(n)})^2 / 2(\sigma_i^{(k')})^2\right)}$$

$$\mathbf{m}^{(k)} = \frac{\sum_k r_k^{(n)} \mathbf{x}^{(n)}}{\sum_n r_k^{(n)}}$$

$$\sigma_k^2 = \frac{\sum_n r_k^{(n)} (x_i^{(n)} - m_i^{(k)})^2}{\sum_n r_k^{(n)}}$$

$$w_k = \frac{\sum_n r_k^{(n)}}{\sum_k \sum_n r_k^{(n)}}$$



Baum-Welch algorithm

Basic idea: This is just an EM algorithm! But instead of:

```
Assignment step (E-step):
   Calculate the probability of the observation
sequence given the current model (A, B, Π)

Update step (M-step):

   Recalculate A
   Recalculate B
   Recalculate Π
```

Forward algorithm

$$a_{ij} = \frac{\text{expected number of transitions from state } i \text{ to } j}{\text{Expected number of transitions from state } i}$$

$$b_{ijk} = \frac{\text{expected \# of transitions from state } i \text{ to } j \text{ with } k \text{ observed}}{\text{Expected number of transitions from } i \text{ to } j}$$

$$\pi_i = \text{expected frequency in state } i \text{ at time } t=1$$

Baum-Welch algorithm

- Because it is an EM algorithm, it has the same properties:

  1. Guaranteed (fairly rapid) convergence, but only to local maxima, not global maxima

  2. Dependence on initial values. In practice, it is especially important to have good starting points for the output parameters B; estimates of A are fairly robust to initial starting point.

Stepping back a bit...

We have defined what a Hidden Markov Model (HMM) is, and proposed it as a better model for language than an n-gram model (i.e., a standard Markov Model)

We have seen in detail how it is possible to calculate the most probable path of hidden states in an HMM, and the probability of an observation

We have seen in brief how it is possible to figure out (imperfectly) what the most probable set of transition probabilities (A, B, Π) are, given a set of observations

Stepping back a bit...

We have defined what a Hidden Markov Model (HMM) is, and proposed it as a better model for language than an n-gram model (i.e., a standard Markov Model)

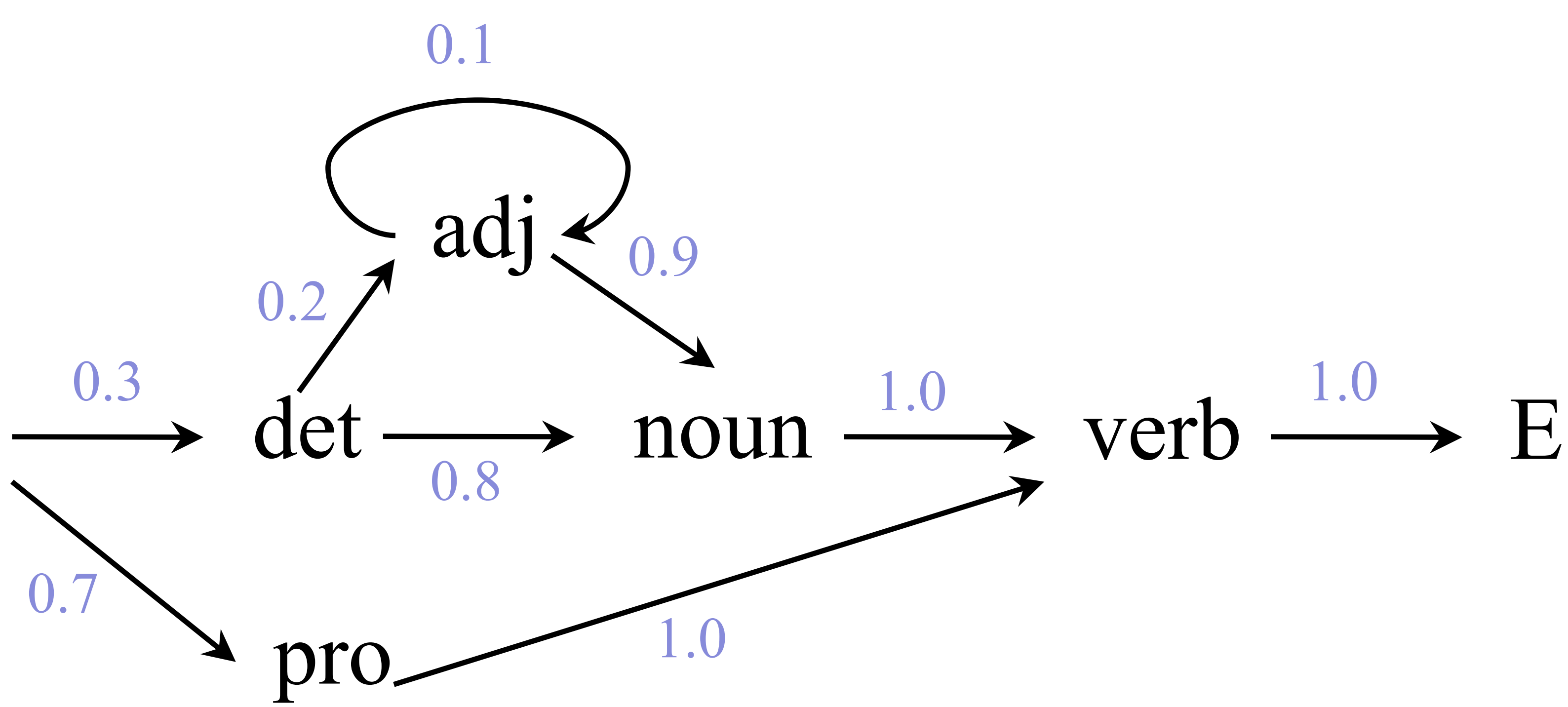
But are HMMs indeed a good model of language?

Not really.

Plan

- Last time: introduction to HMMs
  - Limitations of n-grams applied to language
  - Basics of HMMs
- ➡ Today: finishing HMMs, and more complex structures
  - Determining the likelihood of a given observation
  - Calculating the most likely state sequence
  - Finding the best HMM for given data
  - ➡ More complex models of language

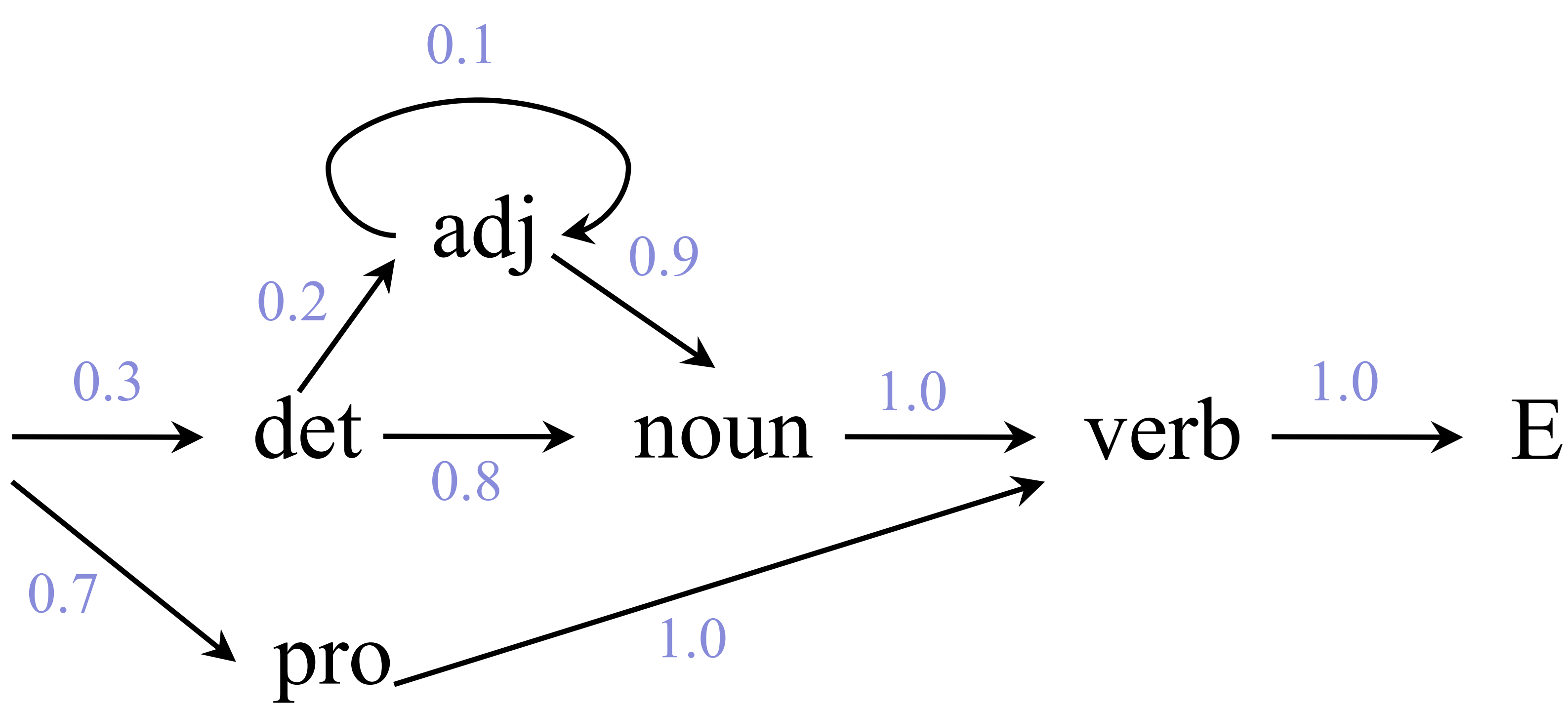
Still have a parameter explosion problem



(0.5) verb → eats
(0.5) verb → runs
(0.3) pro → he
(0.3) pro → she
(0.4) pro → it
(0.7) det → the
(0.3) det → a
(0.4) noun → boy
(0.4) noun → dog
(0.2) noun → tiger
(1.0) adj → happy

Still have a parameter explosion problem

Suppose you want to make it able to
produce: The students write



(0.5) verb → eats
(0.3) verb → runs
(0.2) verb → write
(0.3) pro → he
(0.3) pro → she
(0.4) pro → it
(0.7) det → the
(0.3) det → a
(0.4) noun → boy
(0.2) noun → dog
(0.2) noun → tiger
(0.2) noun → students
(1.0) adj → happy

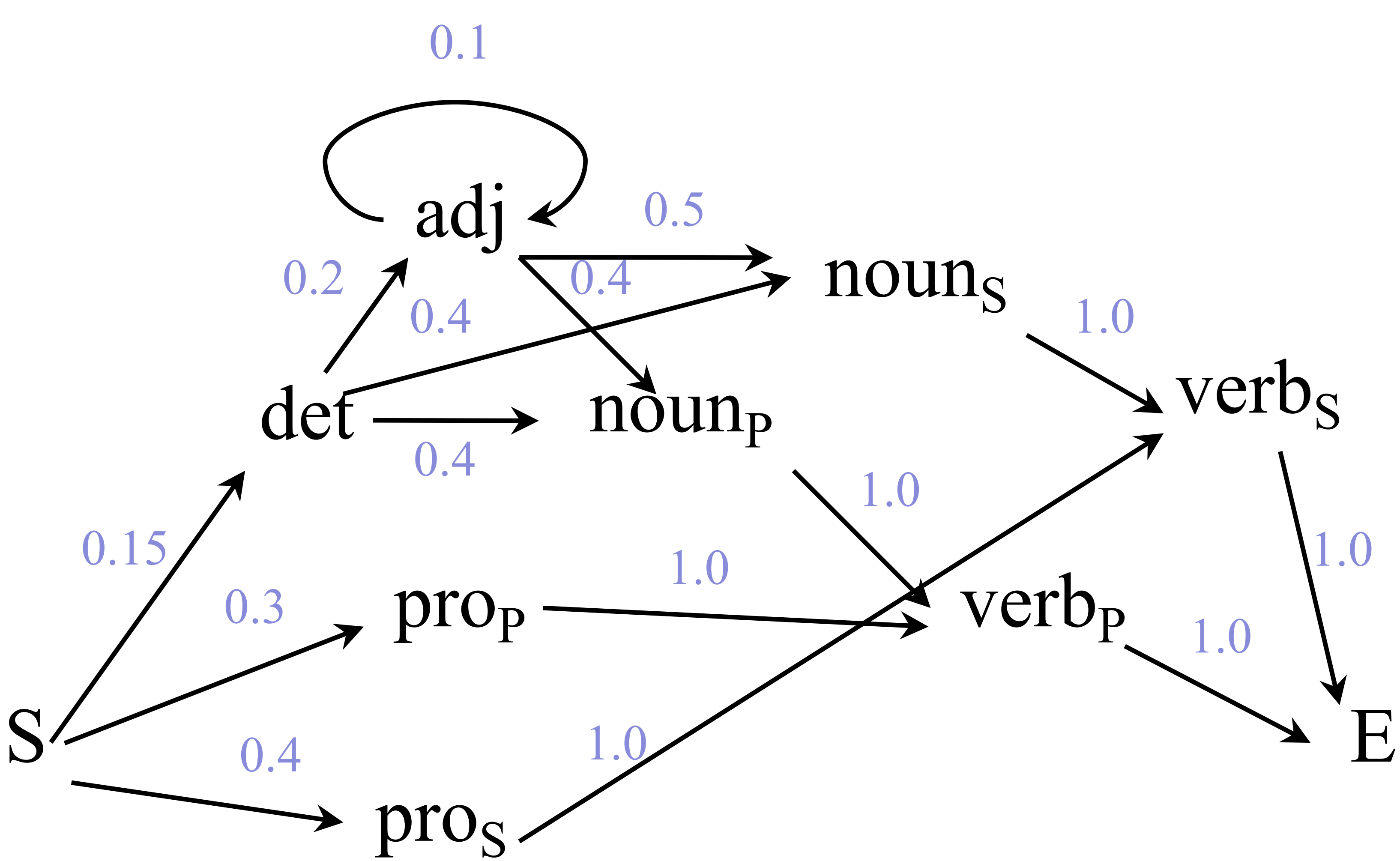
Now it also produces:

The dog write
A students runs
The students eats
He write

Still have a parameter explosion problem

As before, you have to add new states to the model to solve this problem

Still have a parameter explosion problem



(0.5) $verb_S \rightarrow$ eats

(0.5) $verb_S \rightarrow$ runs

(1.0) $verb_P \rightarrow$ write

(0.3) $pro_S \rightarrow$ he

(0.3) $pro_S \rightarrow$ she

(0.4) $pro_S \rightarrow$ it

(0.7) det $\rightarrow$ the

(0.3) det $\rightarrow$ a

(0.4) $noun_S \rightarrow$ boy

(0.4) $noun_S \rightarrow$ dog

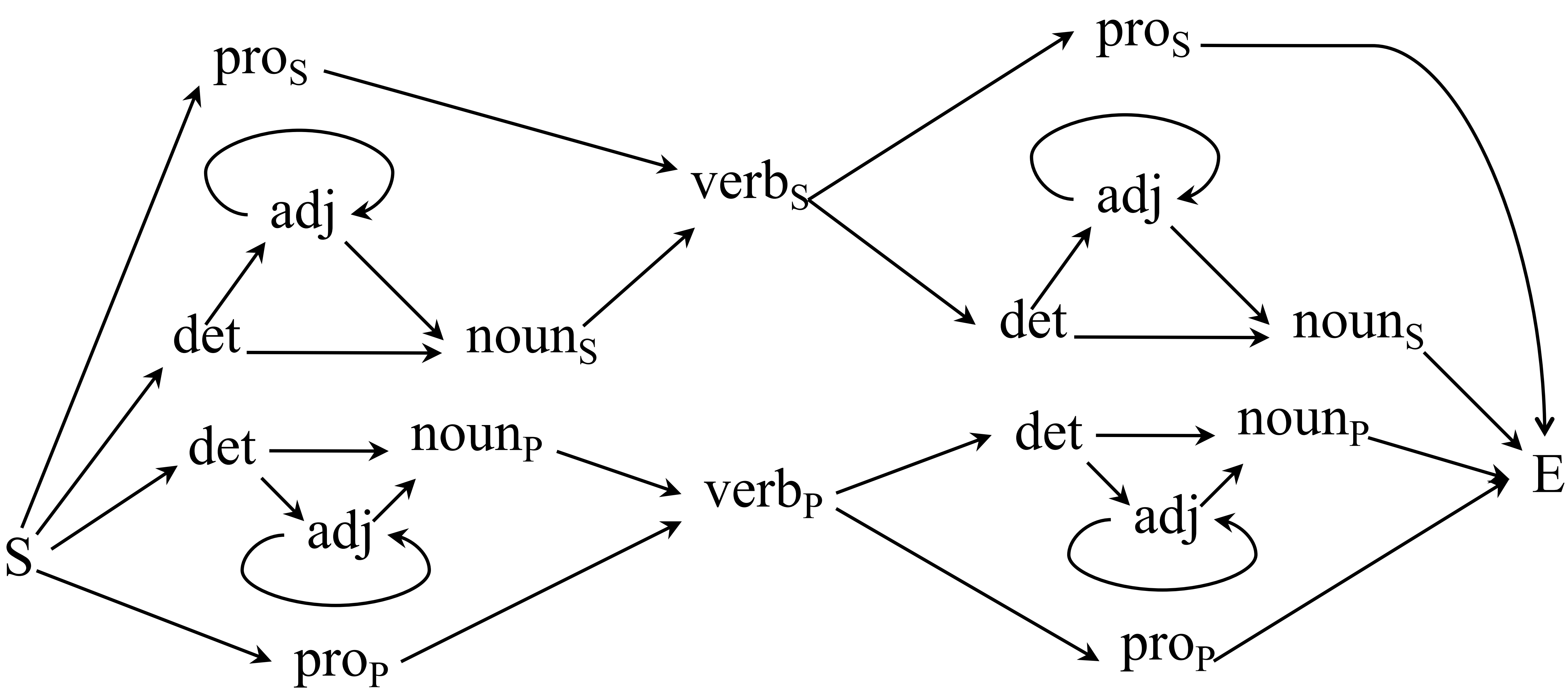
(0.2) $noun_S \rightarrow$ tiger

(1.0) $noun_P \rightarrow$ students

(1.0) adj $\rightarrow$ happy

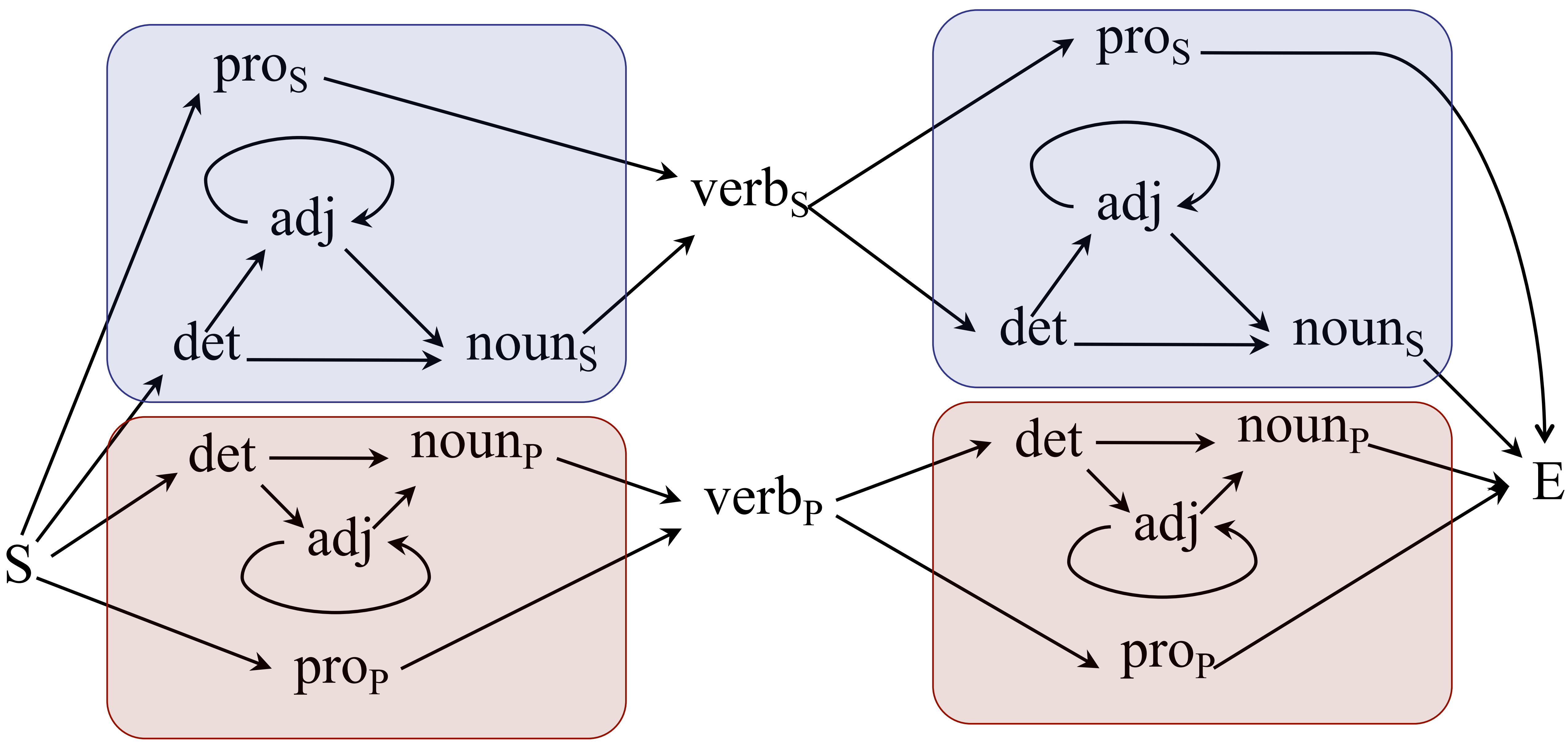
Still have a parameter explosion problem

It's made worse if we make the grammar even more complicated



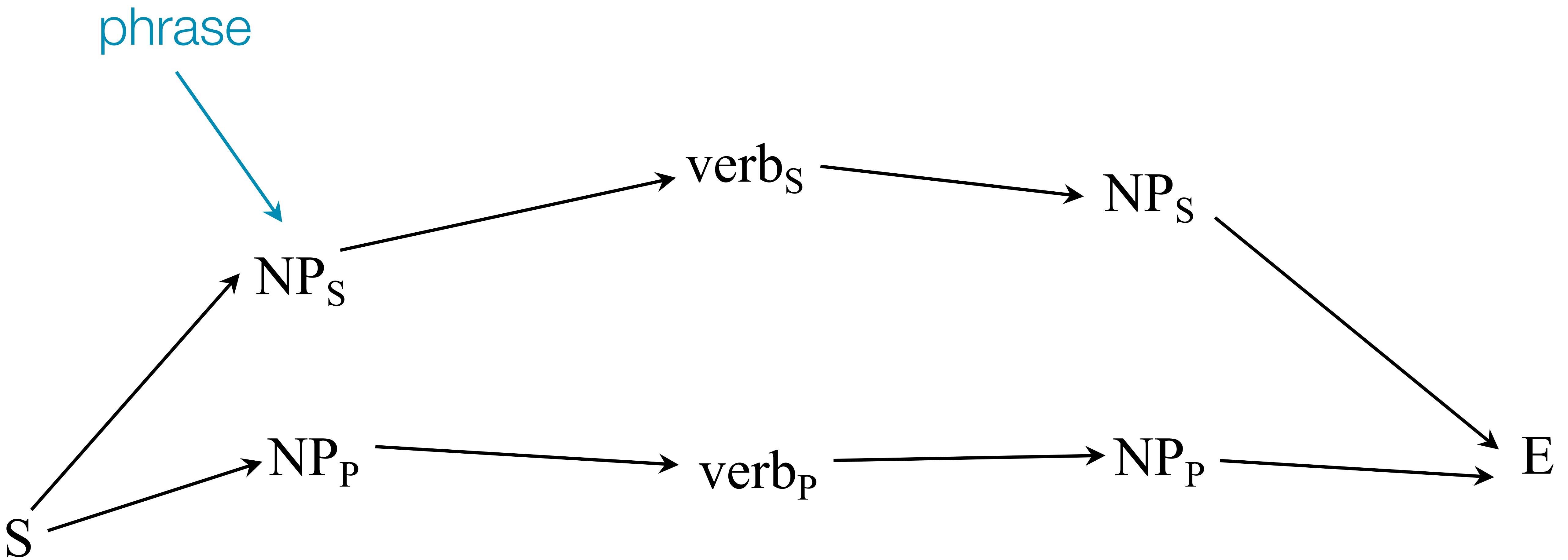
Still have a parameter explosion problem

However, you might notice a regularity in this grammmar



Still have a parameter explosion problem

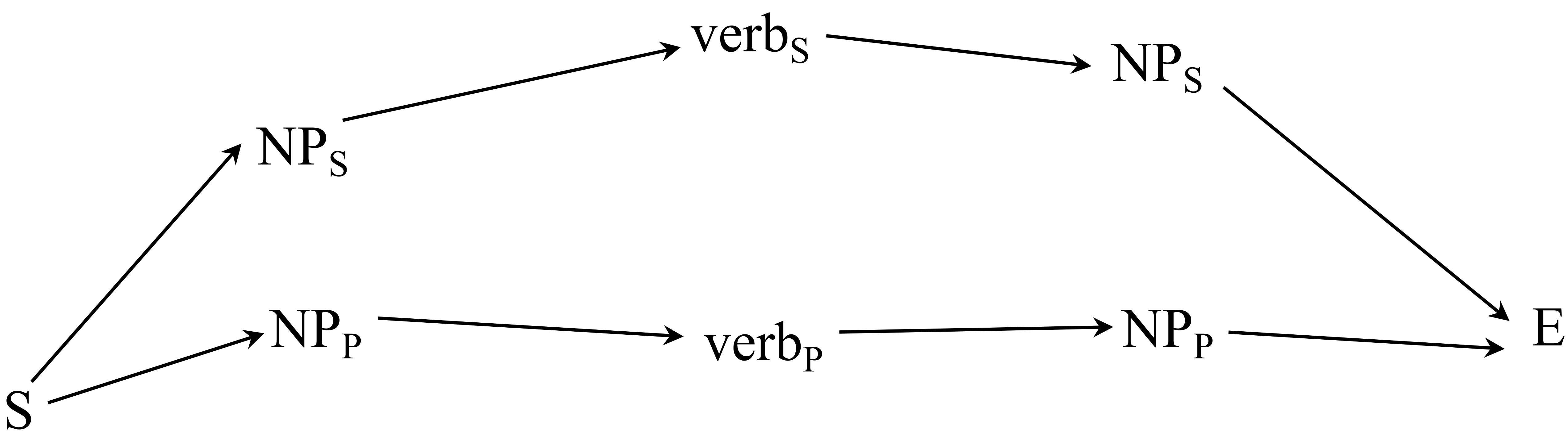
However, you might notice a regularity in this grammmar



Still have a parameter explosion problem

Need to specify what each phrase means

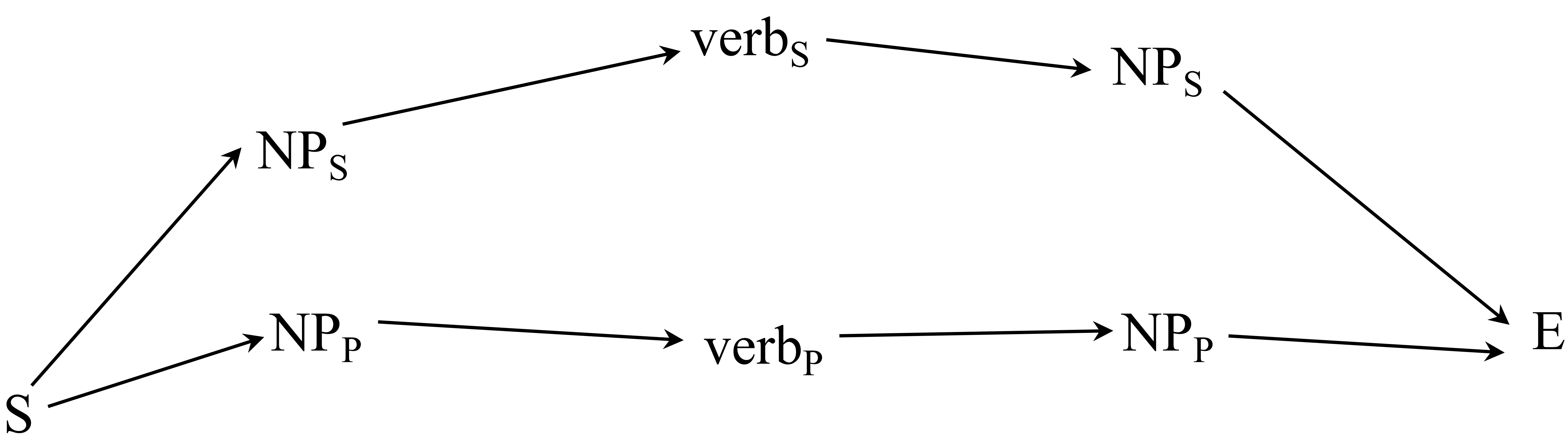
$NP_S \rightarrow pro_S$
$NP_S \rightarrow det\ noun_S$
$NP_P \rightarrow pro_P$
$NP_P \rightarrow det\ noun_P$



Still have a parameter explosion problem

A grammar like this, which is formed by "clustering" states of an HMM, has **phrase structure**

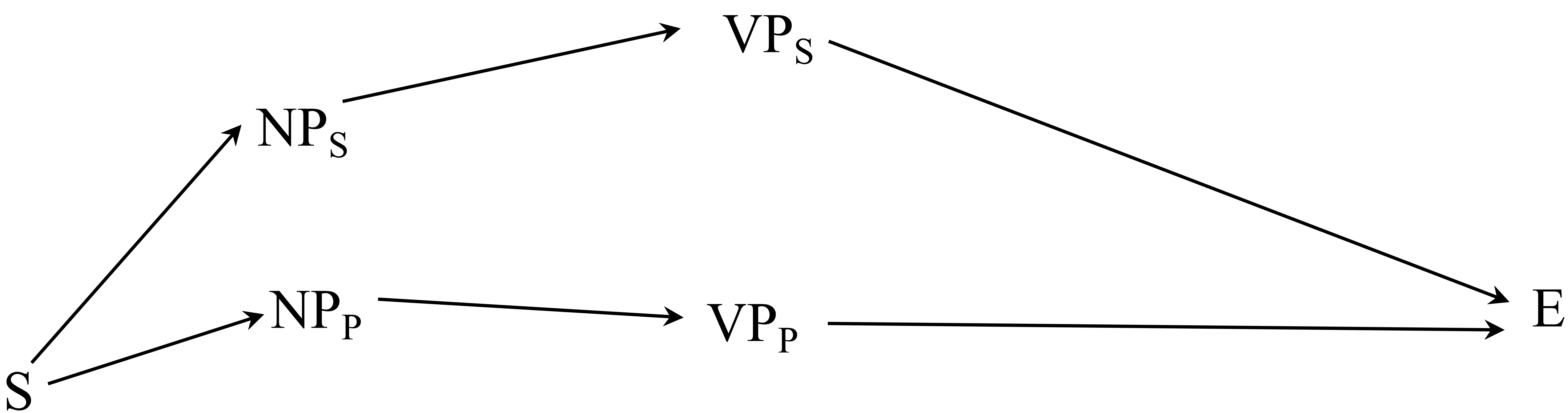
$NP_S \rightarrow pro_S$
$NP_S \rightarrow det\ noun_S$
$NP_P \rightarrow pro_P$
$NP_P \rightarrow det\ noun_P$



Still have a parameter explosion problem

We can even form phrases of other phrases - if we do that, we say the grammar has **hierarchical phrase structure**

$VP_S \rightarrow verb_S\ NP_S$
$VP_P \rightarrow verb_P\ NP_P$
$NP_S \rightarrow pro_S$
$NP_S \rightarrow det\ noun_S$
$NP_P \rightarrow pro_P$
$NP_P \rightarrow det\ noun_P$



This is a context-free grammar (CFG)

Context-free grammars can be formed from an HMM by clustering states into phrases. They have hierarchical phrase structure: a key feature of language

$S \rightarrow NP_S \; VP_S$

$S \rightarrow NP_P \; VP_P$

$VP_S \rightarrow verb_S \; NP_S$

$VP_P \rightarrow verb_P \; NP_P$

$NP_S \rightarrow pro_S$

$NP_S \rightarrow det \; noun_S$

$NP_P \rightarrow pro_P$

$NP_P \rightarrow det \; noun_P$

Formal definition of CFG (or PCFG)

A PCFG G with a vocabulary V and n nonterminals consists of:

- A set of terminals, $\{w^k\}, k = 1, \ldots, V$ — {a, the, boy, tiger, dog, school, it, she, her, him, he…}
- A set of nonterminals, $\{N^i\}, i = 1, \ldots, n$ — $\{S, VP_S, VP_P, NP_S, NP_P, PRO_S, D, N_S, PRO_P, N_P, V_S, A, V_P\}$
- A designated start symbol, N^1 — S
- A set of rules, $\{N^i \to \zeta^j\}$, where ζ^j is a sequence of terminals and nonterminals
- A corresponding set of probabilities on rules such that $\forall i \sum_j P(N^i \to \zeta^j) = 1$

(0.5) $S \rightarrow NP_S\ VP_S$
(0.5) $S \rightarrow NP_P\ VP_P$
(1.0) $VP_S \rightarrow V_S\ NP_S$
(1.0) $VP_P \rightarrow V_P\ NP_P$
(0.7) $NP_S \rightarrow PRO_S$
(0.3) $NP_S \rightarrow D\ N_S$
(0.7) $NP_P \rightarrow PRO_P$
(0.3) $NP_P \rightarrow D\ N_P$

(0.3) $PRO_S \rightarrow$ he
(0.3) $PRO_S \rightarrow$ she
(0.4) $PRO_S \rightarrow$ it
(0.3) $PRO_P \rightarrow$ her
(0.3) $PRO_P \rightarrow$ him
(0.4) $PRO_P \rightarrow$ it
(0.3) $D \rightarrow$ a
(0.7) $D \rightarrow$ the
(0.1) $N_S \rightarrow A\ N_S$
(0.1) $N_P \rightarrow A\ N_P$

(0.3) $N_S \rightarrow$ boy
(0.3) $N_S \rightarrow$ tiger
(0.3) $N_S \rightarrow$ dog
(0.2) $N_P \rightarrow$ boy
(0.2) $N_P \rightarrow$ tiger
(0.2) $N_P \rightarrow$ dog
(0.15) $N_P \rightarrow$ school
(0.15) $N_P \rightarrow$ house
(0.5) $V_S \rightarrow$ sees
(0.5) $V_S \rightarrow$ cleans
(0.5) $V_P \rightarrow$ see
(0.5) $V_P \rightarrow$ clean
(1.0) $A \rightarrow$ happy

Example sentences

The boys clean the house

She cleans it

He sees the dog

The happy dogs see a tiger

(0.5) $S \rightarrow NP_S\ VP_S$
(0.5) $S \rightarrow NP_P\ VP_P$
(1.0) $VP_S \rightarrow V_S\ NP_S$
(1.0) $VP_P \rightarrow V_P\ NP_P$
(0.7) $NP_S \rightarrow PRO_S$
(0.3) $NP_S \rightarrow D\ N_S$
(0.7) $NP_P \rightarrow PRO_P$
(0.3) $NP_P \rightarrow D\ N_P$

(0.3) $PRO_S \rightarrow$ he
(0.3) $PRO_S \rightarrow$ she
(0.4) $PRO_S \rightarrow$ it
(0.3) $PRO_P \rightarrow$ her
(0.3) $PRO_P \rightarrow$ him
(0.4) $PRO_P \rightarrow$ it
(0.3) $D \rightarrow$ a
(0.7) $D \rightarrow$ the
(0.1) $N_S \rightarrow A\ N_S$
(0.1) $N_P \rightarrow A\ N_P$

(0.3) $N_S \rightarrow$ boy
(0.3) $N_S \rightarrow$ tiger
(0.3) $N_S \rightarrow$ dog
(0.2) $N_P \rightarrow$ boy
(0.2) $N_P \rightarrow$ tiger
(0.2) $N_P \rightarrow$ dog
(0.15) $N_P \rightarrow$ school
(0.15) $N_P \rightarrow$ house
(0.5) $V_S \rightarrow$ sees
(0.5) $V_S \rightarrow$ cleans
(0.5) $V_P \rightarrow$ see
(0.5) $V_P \rightarrow$ clean
(1.0) $A \rightarrow$ happy

Example sentences

The boys clean the house

She cleans it

The boys see

She cleans

The happy dogs see a tiger

He sees the dog

It is relatively easy to expand on this and add new types of sentences

(0.5) $S \rightarrow NP_S\ VP_S$
(0.5) $S \rightarrow NP_P\ VP_P$
(0.5) $VP_S \rightarrow V_S\ NP_S$
(0.5) $VP_S \rightarrow V_S$
(0.5) $VP_P \rightarrow V_P\ NP_P$
(0.5) $VP_P \rightarrow V_P$
(0.7) $NP_S \rightarrow PRO_S$
(0.3) $NP_S \rightarrow D\ N_S$
(0.7) $NP_P \rightarrow PRO_P$
(0.3) $NP_P \rightarrow D\ N_P$

(0.3) $PRO_S \rightarrow$ he
(0.3) $PRO_S \rightarrow$ she
(0.4) $PRO_S \rightarrow$ it
(0.3) $PRO_P \rightarrow$ her
(0.3) $PRO_P \rightarrow$ him
(0.4) $PRO_P \rightarrow$ it
(0.3) $D \rightarrow$ a
(0.7) $D \rightarrow$ the
(0.1) $N_S \rightarrow A\ N_S$
(0.1) $N_P \rightarrow A\ N_P$

(0.3) $N_S \rightarrow$ boy
(0.3) $N_S \rightarrow$ tiger
(0.3) $N_S \rightarrow$ dog
(0.2) $N_P \rightarrow$ boy
(0.2) $N_P \rightarrow$ tiger
(0.2) $N_P \rightarrow$ dog
(0.15) $N_P \rightarrow$ school
(0.15) $N_P \rightarrow$ house
(0.5) $V_S \rightarrow$ sees
(0.5) $V_S \rightarrow$ cleans
(0.5) $V_P \rightarrow$ see
(0.5) $V_P \rightarrow$ clean
(1.0) $A \rightarrow$ happy

Example sentences

The boys behind the school clean the house.

He sees the dog with it

She sees the dog behind a tiger

(0.5) $S \rightarrow NP_S\ VP_S$
(0.5) $S \rightarrow NP_P\ VP_P$
(0.4) $VP_S \rightarrow V_S\ NP_S$
(0.2) $VP_S \rightarrow V_S\ NP_S\ PP$
(0.4) $VP_S \rightarrow V_S$
(0.4) $VP_P \rightarrow V_P\ NP_P$
(0.2) $VP_P \rightarrow V_P\ NP_P\ PP$
(0.4) $VP_P \rightarrow V_P$
(0.6) $NP_S \rightarrow PRO_S$
(0.3) $NP_S \rightarrow D\ N_S$
(0.6) $NP_P \rightarrow PRO_P$
(0.3) $NP_P \rightarrow D\ N_P$
(0.1) $NP_S \rightarrow NP_S\ PP$
(0.1) $NP_P \rightarrow NP_P\ PP$

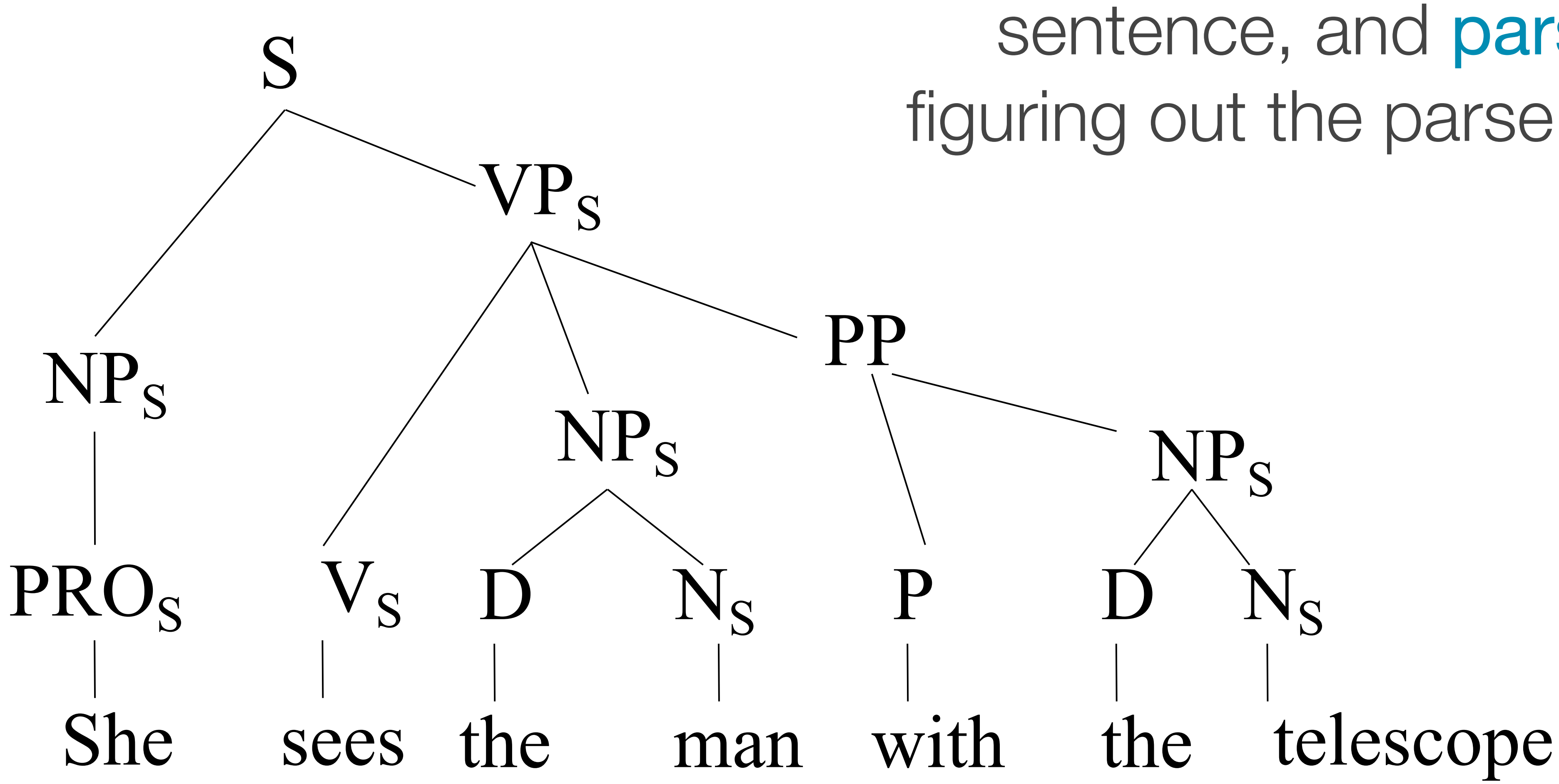
(0.5) $PP \rightarrow P\ NP_S$
(0.5) $PP \rightarrow P\ NP_P$
(0.3) $PRO_S \rightarrow$ he
(0.3) $PRO_S \rightarrow$ she
(0.4) $PRO_S \rightarrow$ it
(0.3) $PRO_P \rightarrow$ her
(0.3) $PRO_P \rightarrow$ him
(0.4) $PRO_P \rightarrow$ it
(0.3) $D \rightarrow$ a
(0.7) $D \rightarrow$ the
(0.1) $N_S \rightarrow A\ N_S$
(0.1) $N_P \rightarrow A\ N_P$
(0.5) $P \rightarrow$ with
(0.5) $P \rightarrow$ behind

(0.3) $N_S \rightarrow$ boy
(0.3) $N_S \rightarrow$ tiger
(0.3) $N_S \rightarrow$ dog
(0.2) $N_P \rightarrow$ boy
(0.2) $N_P \rightarrow$ tiger
(0.2) $N_P \rightarrow$ dog
(0.15) $N_P \rightarrow$ school
(0.15) $N_P \rightarrow$ house
(0.5) $V_S \rightarrow$ sees
(0.5) $V_S \rightarrow$ cleans
(0.5) $V_P \rightarrow$ see
(0.5) $V_P \rightarrow$ clean
(1.0) $A \rightarrow$ happy

Context free grammars

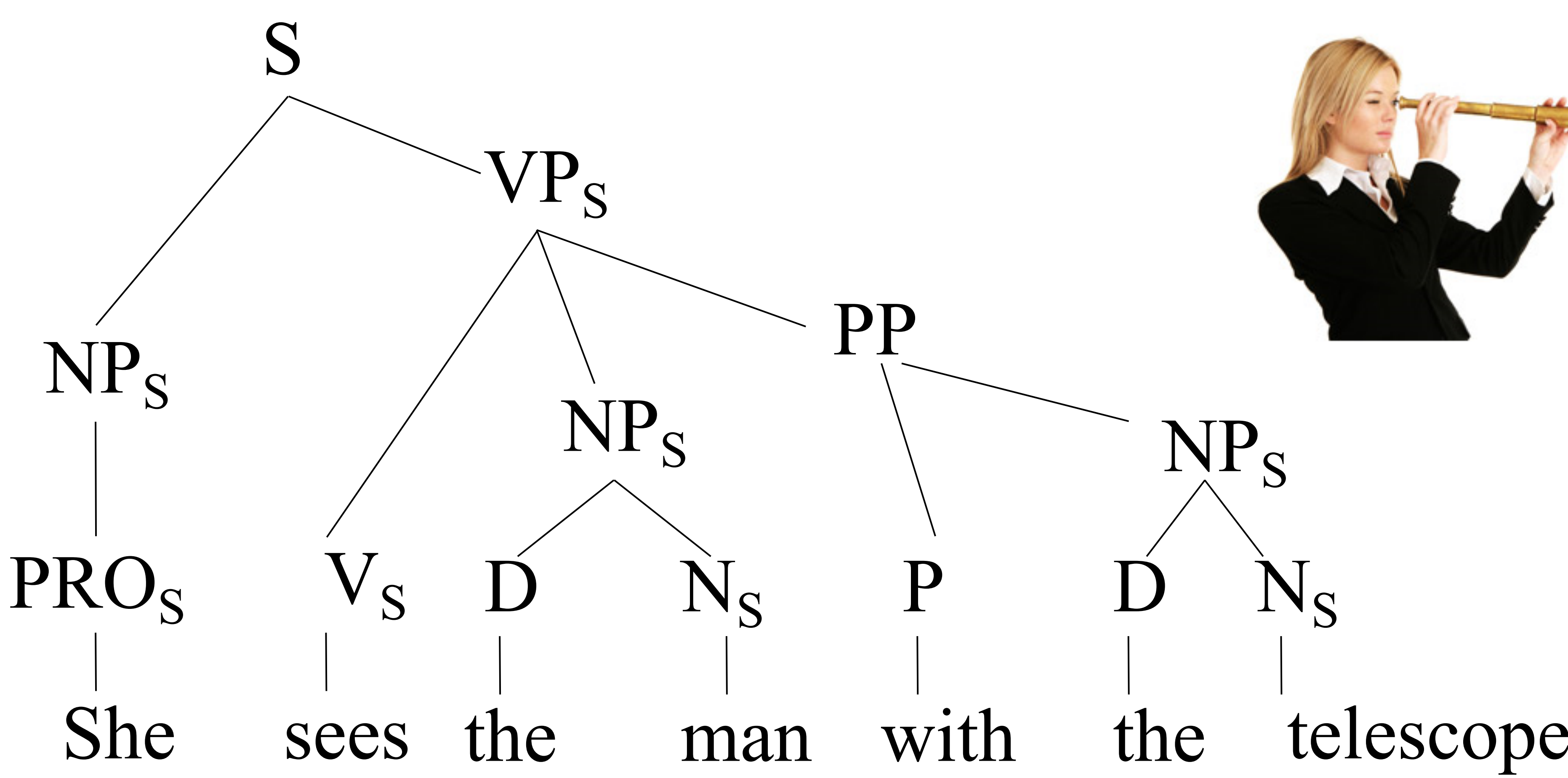
Yield sentences with hierarchical phrase structure, in which phrases can be nestled hierarchically within one another.

This is known as a parse tree for that sentence, and parsing is the act of figuring out the parse trees for a given sentence.



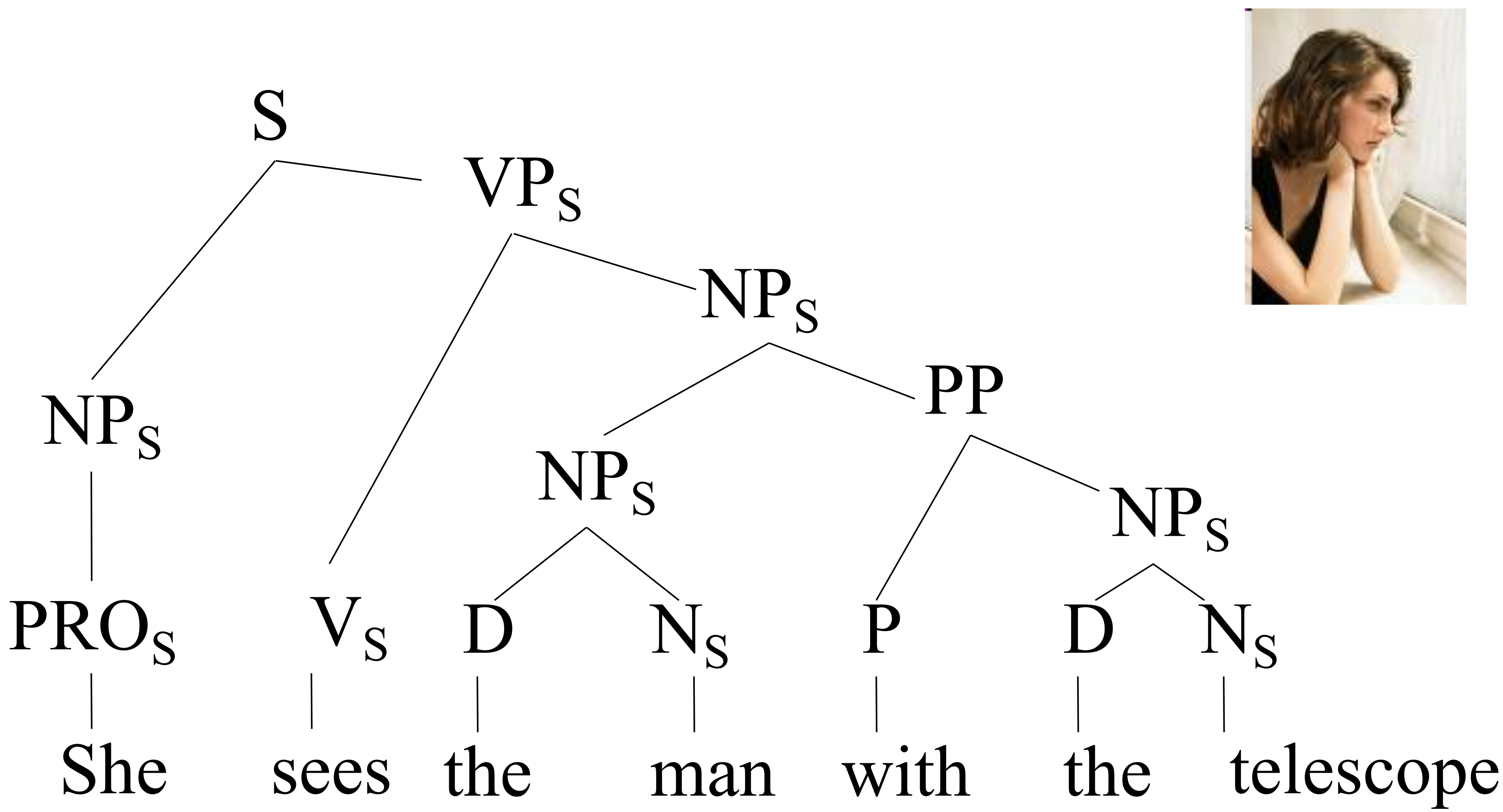
Context free grammars

Many sentences are ambiguous - they have multiple possible parse trees



Context free grammars

Many sentences are ambiguous - they have multiple possible parse trees



Context free grammars

This can often be a source of unintentional humour

```
Don't let worry kill you — let the church help.

Ingres enjoyed painting his models nude.

Visiting relatives can be boring.
```

```
Iraqi head seeks arms

Grandmother of eight makes hole in one

Two sisters reunite after eighteen years at
checkout counter

Dr. Ruth to talk about sex with newspaper editors
```

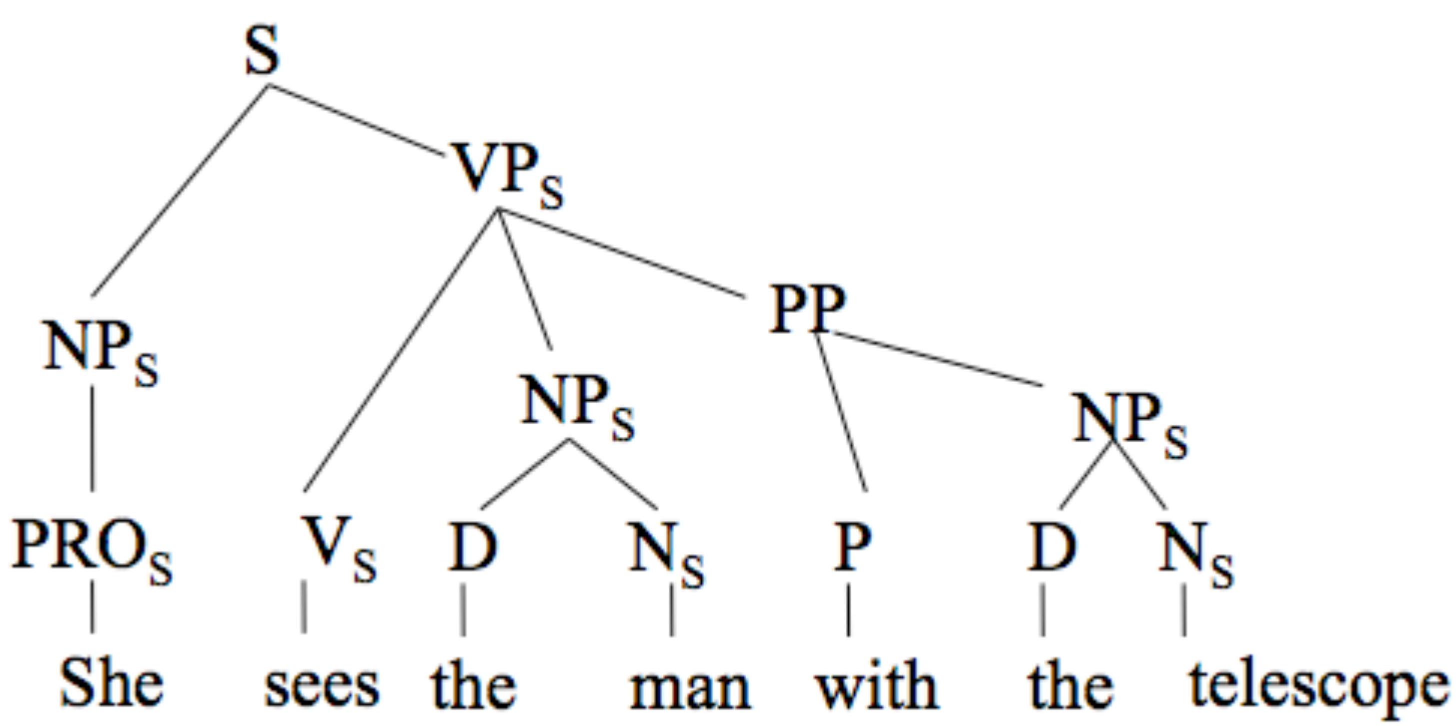
Context free grammars

These sorts of misunderstandings are one of the pieces of evidence suggesting that the underlying parse trees are psychologically "real"

Using context-free grammars

The probability of a parse is the probability of each of the rules used to generate that parse

0.5*0.6*0.3*0.2*0.5*0.3*0.7*0.3*0.5*0.5*0.3*0.7*0.2 = 5.95e-6

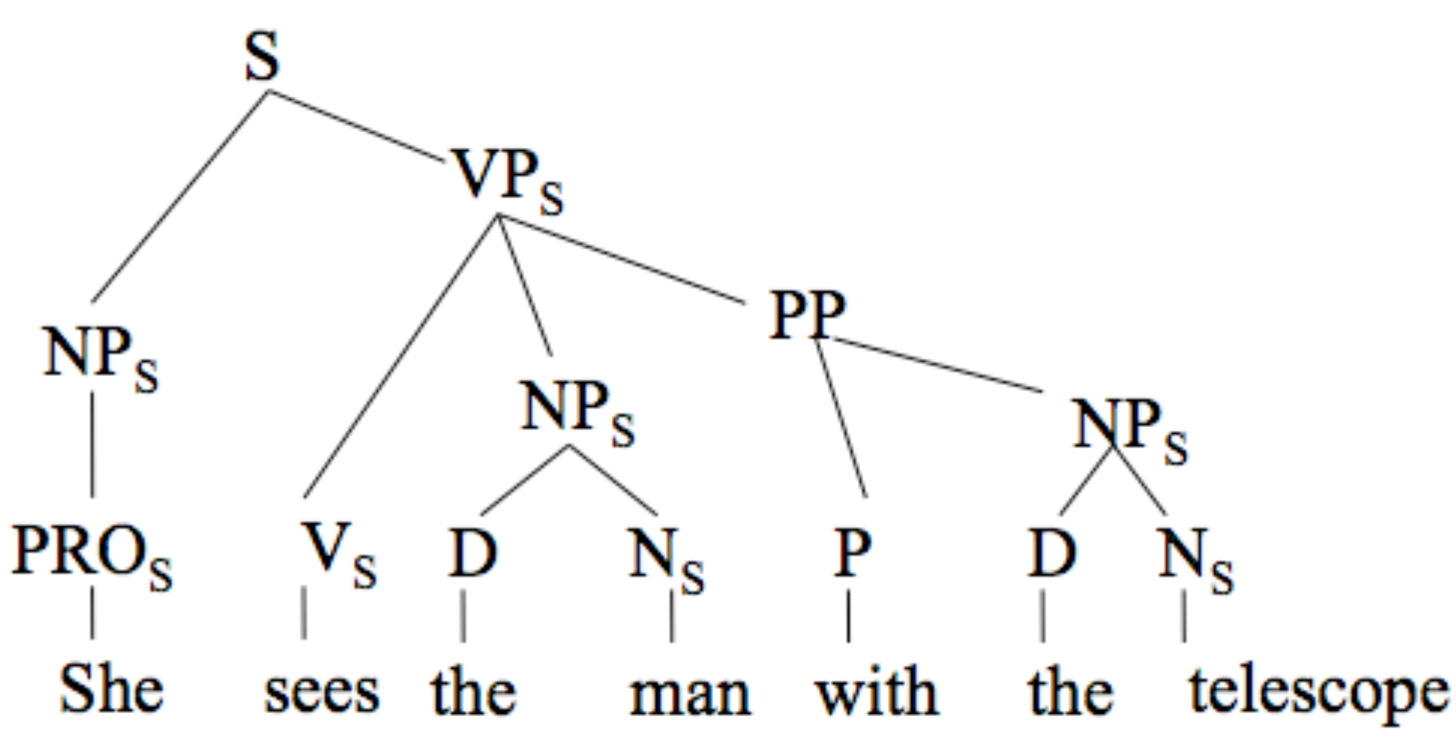


(0.5) $S \rightarrow NP_S\ VP_S$
(0.6) $NP_S \rightarrow PRO_S$
(0.3) $PRO_S \rightarrow$ she
(0.2) $VP_S \rightarrow V_S\ NP_S\ PP$
(0.5) $V_S \rightarrow$ sees
(0.3) $NP_S \rightarrow D\ N_S$
(0.7) $D \rightarrow$ the
(0.3) $N_S \rightarrow$ man
(0.5) $PP \rightarrow P\ NP_S$
(0.5) $P \rightarrow$ with
(0.7) $D \rightarrow$ the
(0.2) $N_S \rightarrow$ telescope

Using context-free grammars

If the sentence is ambiguous, you need to add the probabilities of each of the possible parses

0.5*0.6*0.3*0.2*0.5*0.3*0.7*0.3*0.5*0.5*0.3*0.7*0.2 +
0.5*0.6*0.3*0.4*0.5*0.3*0.3*0.7*0.3*0.5*0.5*0.3*0.7*0.2 = 9.52e-6

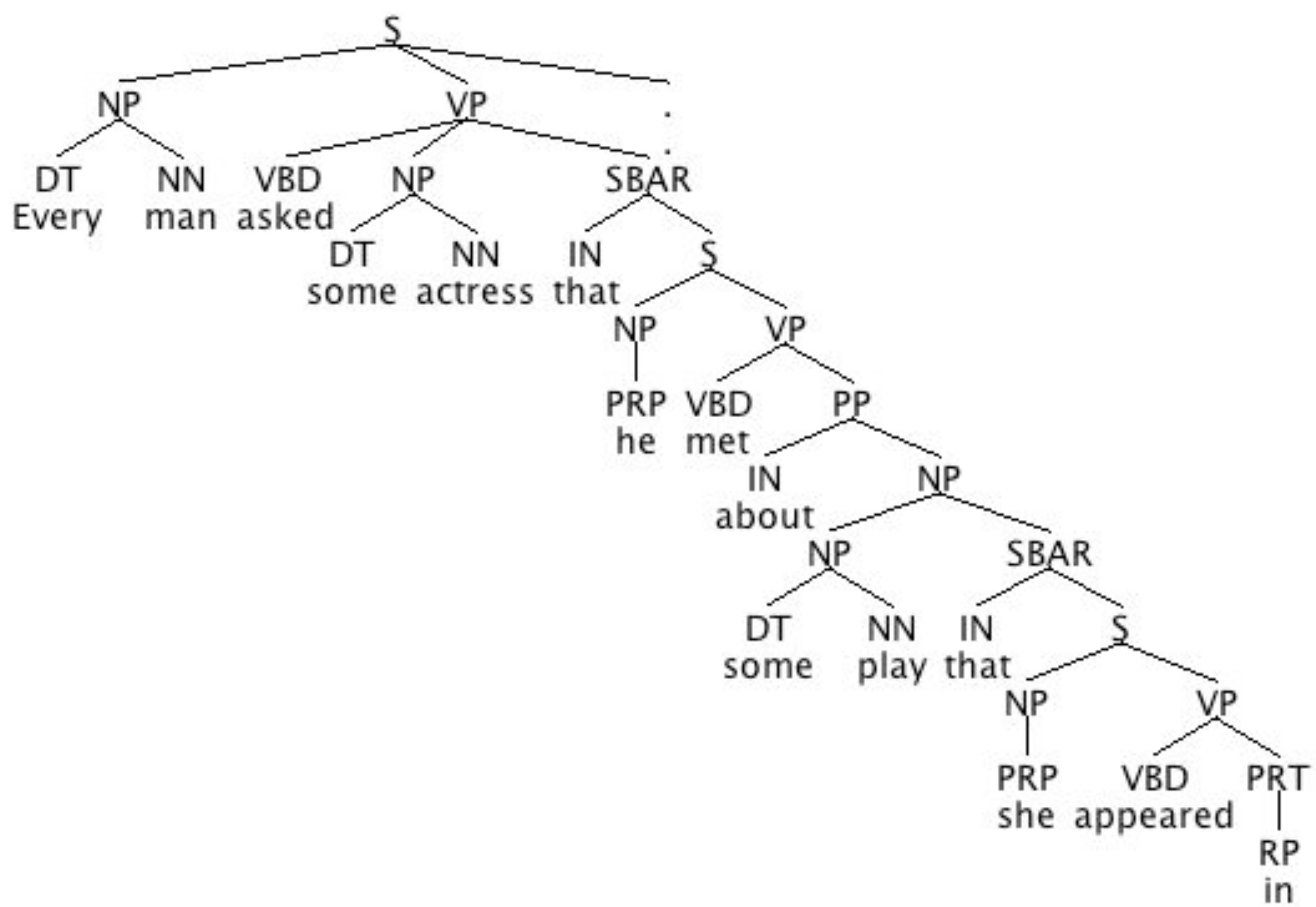


(0.5) S → NP_S VP_S
(0.6) NP_S → PRO_S
(0.3) PRO_S → she
(0.2) VP_S → V_S NP_S PP
(0.5) V_S → sees
(0.3) NP_S → D N_S
(0.7) D → the
(0.3) N_S → man
(0.5) PP → P NP_S
(0.5) P → with
(0.7) D → the
(0.2) N_S → telescope

Using context-free grammars

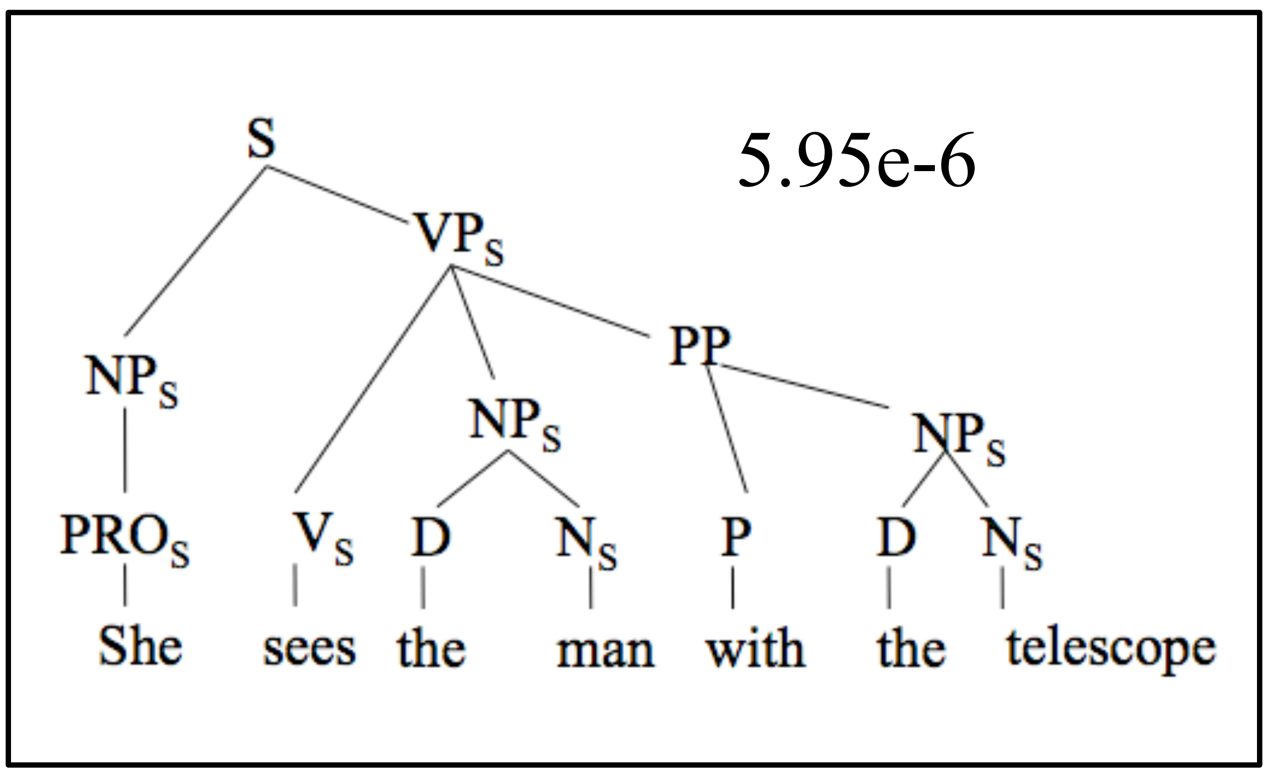
More massive grammar = more ambiguous sentences.

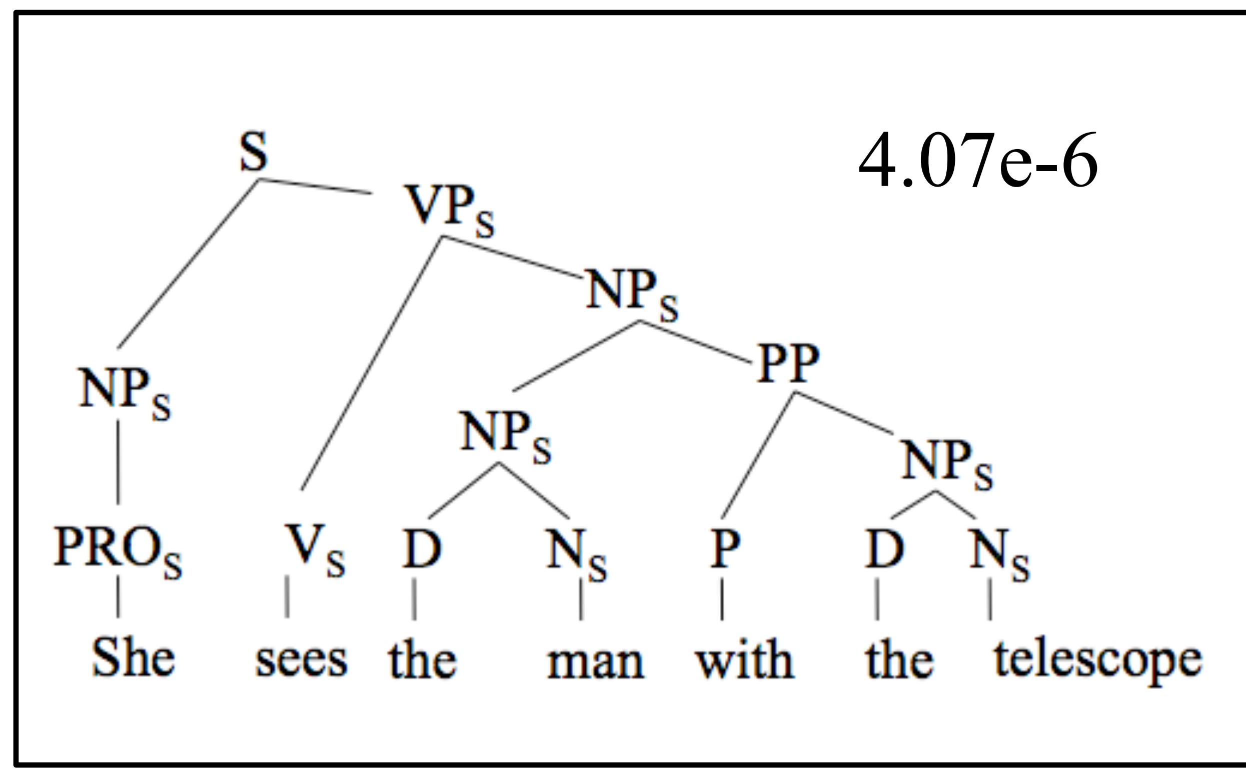
Grammars that are typically used in computational linguistics have many ambiguous parses.



Using context-free grammars

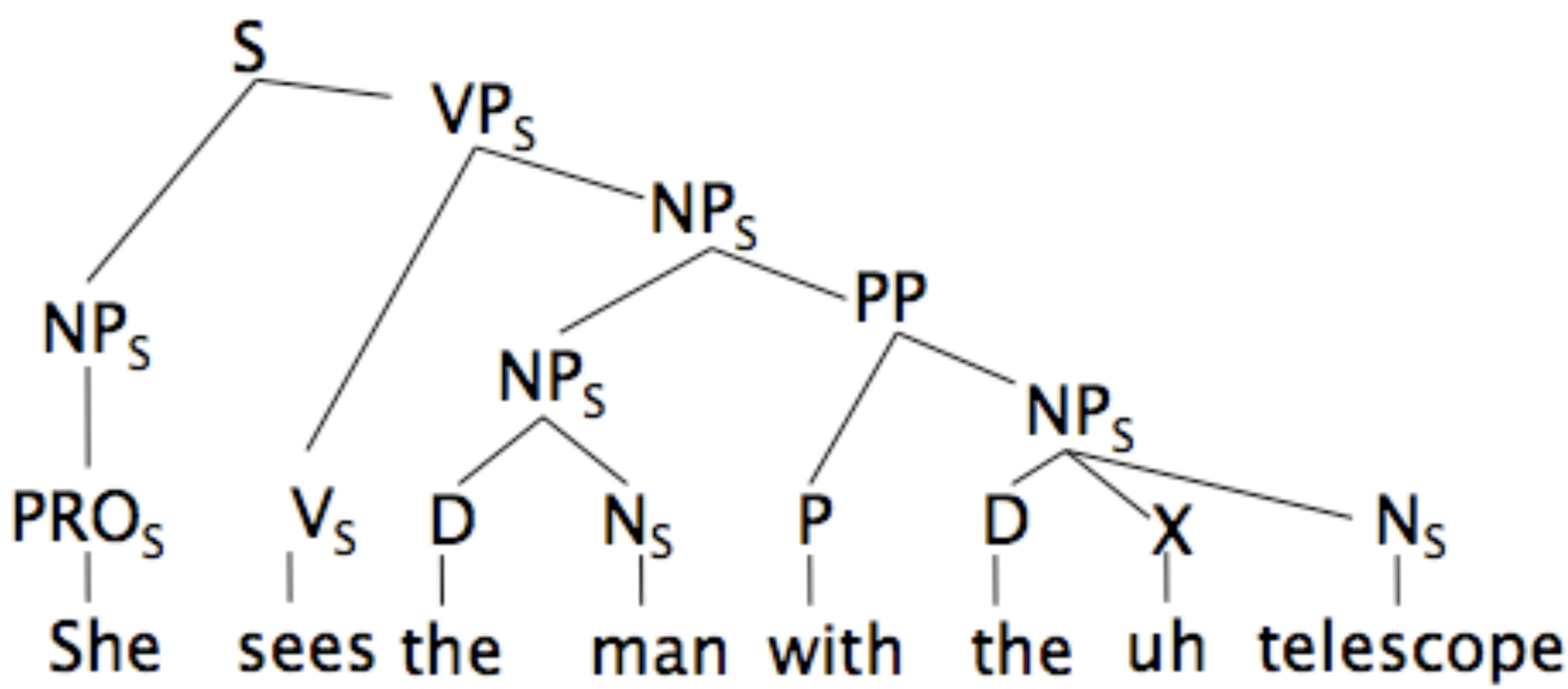
A PCFG gives some idea of the plausibility of different parses; however, this is often not very linguistically accurate, since it doesn't take into account semantics (meaning) or local lexical context





Using context-free grammars

Real language contains a lot of grammatical mistakes; PCFGs can be fairly robust to those, at the price of having many incorrect (but very low-probability) rules.



(0.5) S → NP$_S$ VP$_S$
(0.6) NP$_S$ → PRO$_S$
(0.3) PRO$_S$ → she
(0.4) VP$_S$ → V$_S$ NP$_S$
(0.5) V$_S$ → sees
(0.29) NP$_S$ → D N$_S$
(0.01) NP$_S$ → D X N$_S$
(0.7) D → the
(0.3) N$_S$ → man
(0.5) PP → P NP$_S$
(0.5) P → with
(0.7) D → the
(0.2) N$_S$ → telescope
(1.0) X → uh

Using context-free grammars

CFGs are useful because:

They are tractable, and more realistic models of language than HMMs or n-grams

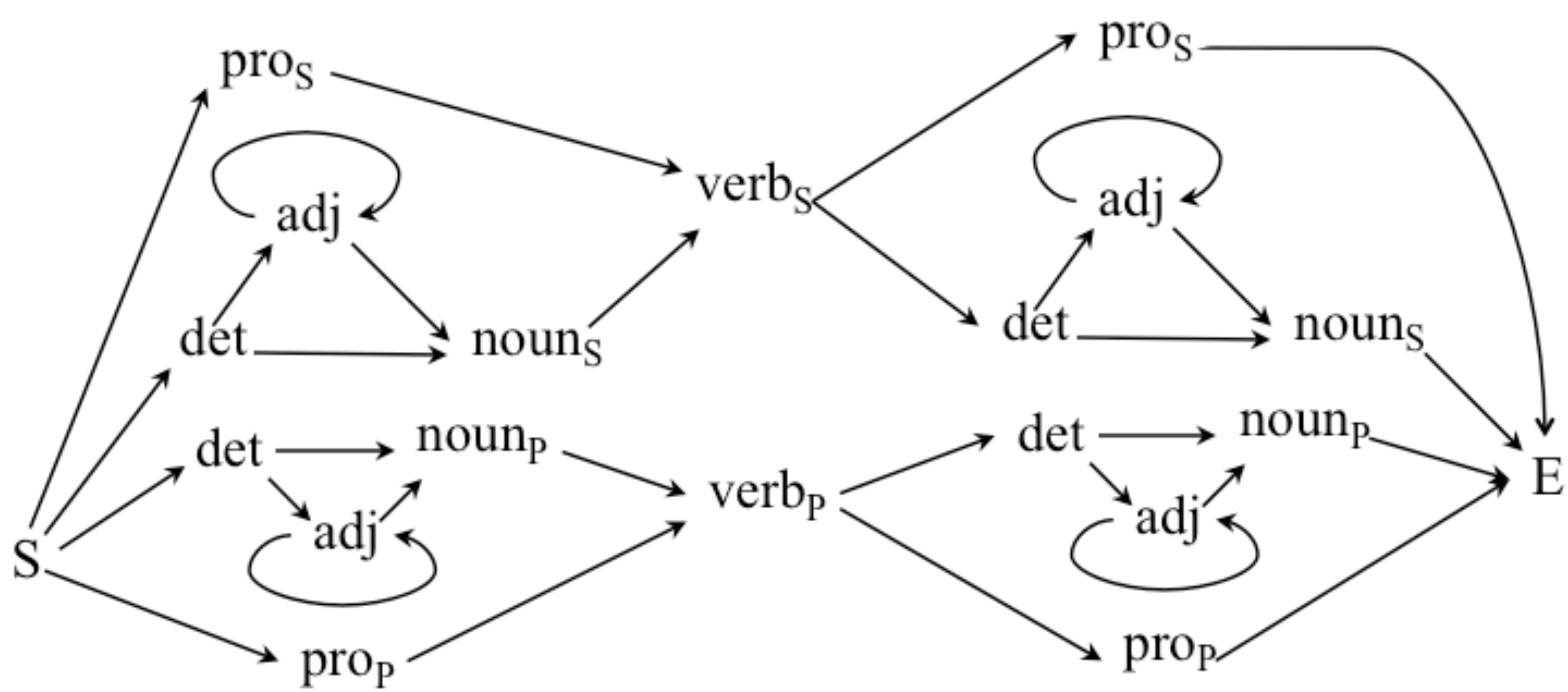
But we still use HMMs and n-grams because:

They are much *more* tractable, and scale better with large vocabularies.

In practice, most state-of-the-art stuff combines these different techniques to try to take advantage of the best aspects of each

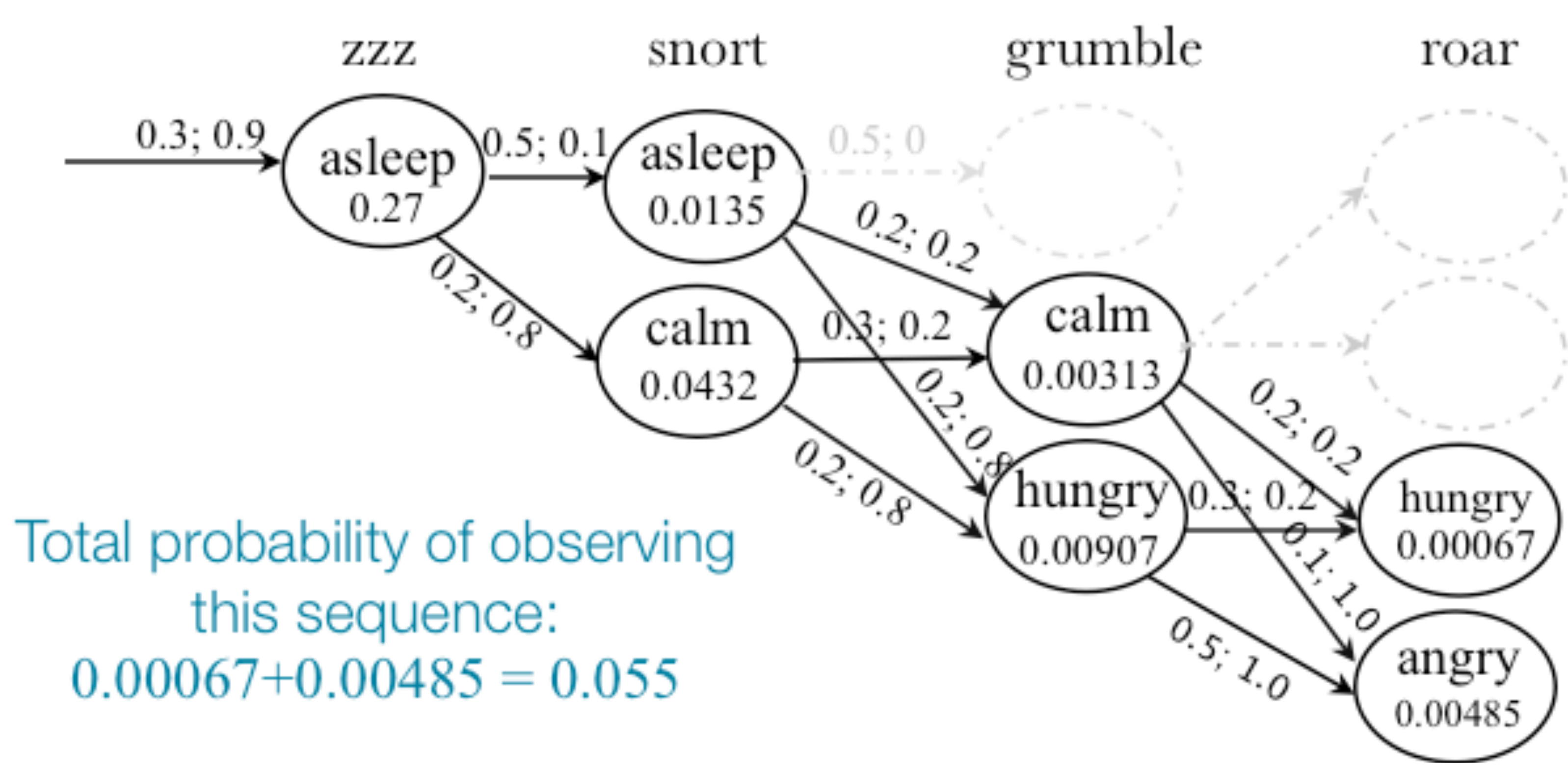
Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems



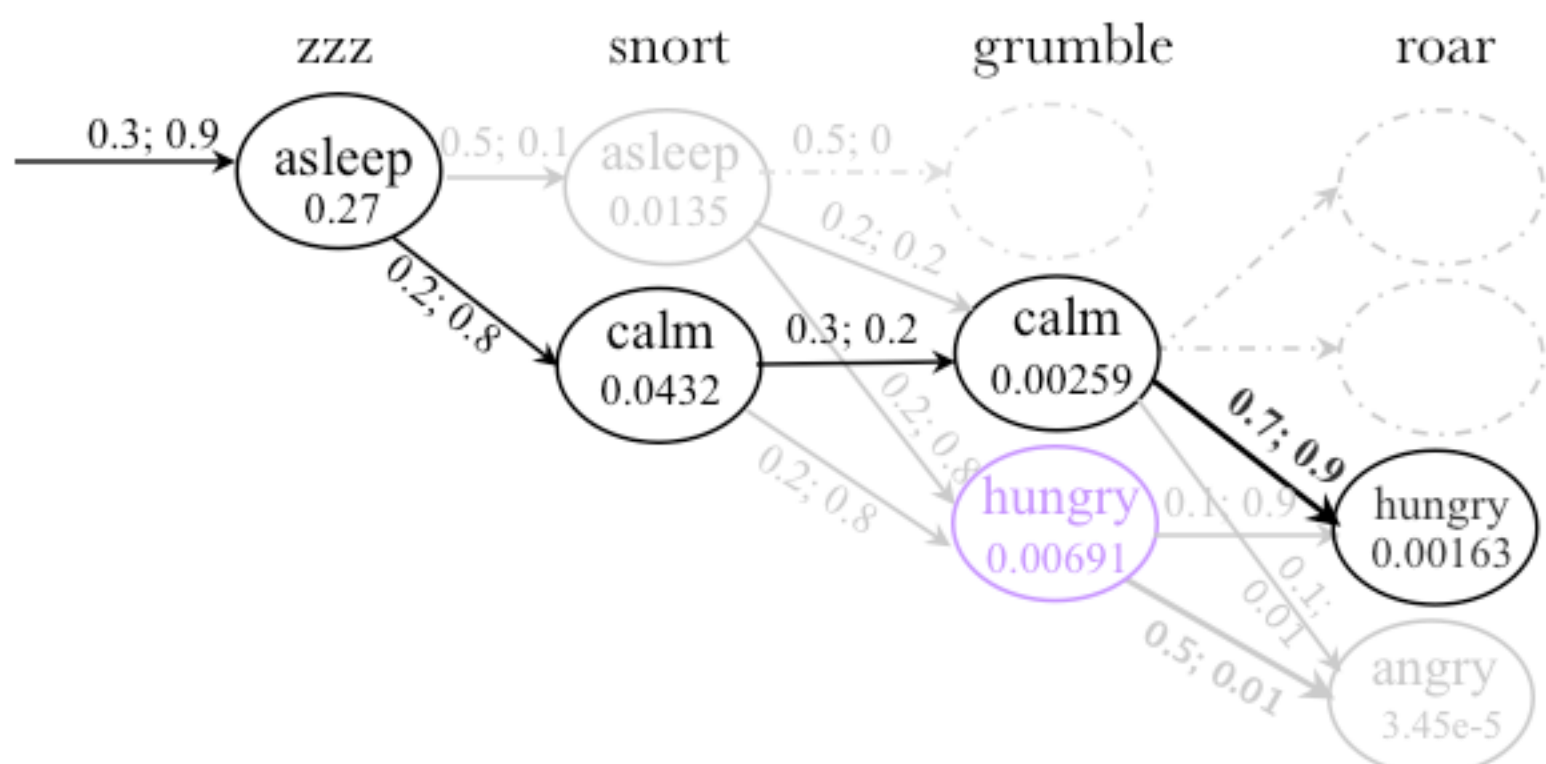
Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems
- Forward algorithm: calculate the probability of an observation



Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems
- Forward algorithm: calculate the probability of an observation
- Viterbi algorithm: calculate the most likely path through an HMM



Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems
- Forward algorithm: calculate the probability of an observation
- Viterbi algorithm: calculate the most likely path through an HMM
- Baum-Welch algorithm: figure out the most likely model given a set of observations

```
Assignment step (E-step):
   Calculate the probability of the observation
sequence given the current model (A, B, Π)

Update step (M-step):

   Recalculate A
   Recalculate B
   Recalculate Π
```

Forward algorithm

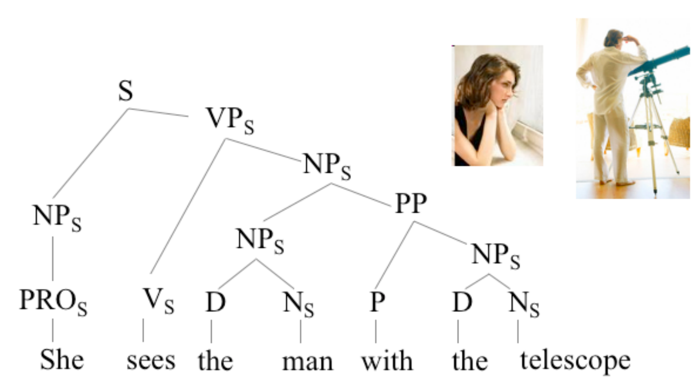
$$a_{ij} = \frac{\text{expected number of transitions from state } i \text{ to } j}{\text{Expected number of transitions from state } i}$$

$$b_{ijk} = \frac{\text{expected \# of transitions from state } i \text{ to } j \text{ with } k \text{ observed}}{\text{Expected number of transitions from } i \text{ to } j}$$

$$\pi_i = \text{expected frequency in state } i \text{ at time } t{=}1$$

Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems
- Forward algorithm: calculate the probability of an observation
- Viterbi algorithm: calculate the most likely path through an HMM
- Baum-Welch algorithm: figure out the most likely model given a set of observations
- Context free grammars: Are a much better model for language because they have hierarchical phrase structure



Summary

- Hidden Markov models: Markov models with hidden states (often corresponding to parts of speech) do better than n-grams, although still have parameter explosion problems
- Forward algorithm: calculate the probability of an observation
- Viterbi algorithm: calculate the most likely path through an HMM
- Baum-Welch algorithm: figure out the most likely model given a set of observations
- Context free grammars: Are a much better model for language because they have hierarchical phrase structure

Starting next time: switching gears again to how people use data. In particular, we'll talk about how the informativeness of data depends on how it was sampled and the structure of the hypotheses (and whether people are aware of this)

Additional references (not required)

HMMs

- Wikipedia entry on CFGs is also pretty good!
- Manning, C., & Schutze, H. (1999). Foundations of statistical natural language processing. Chapter 11.
- Russell, S., & Norvig, P. (1995). Artificial Intelligence: A modern approach. (This one is first edition, but all editions have good resources on grammars). Chapter 22