Soft Inference and Posterior Marginals

21 February 2018

The questions we answered so far

- "*What is the best path through this graph*"
- "*What is the state sequence underlying this string*"
- "*Is this string a part of this language*"
- "*How do you compose this string, with this language*"

- Decisive answers to definitive questions
- "Hard" inference

"Soft" questions

- *How probable is it for this language to produce this symbol sequence?*
- *How likely is it that the word "feed" here is a noun and not a verb?*
- *How likely is this segment to be a constituent?*
- *How probable is it that rule X→ YZ has been used in composing this sentence*

- "Confidence"-type answers to questions about

Soft vs. Hard Inference

- Hard inference
  - "Give me a single solution"
  - Viterbi algorithm
  - Maximum spanning tree (Chu-Liu-Edmonds alg.)
- Soft inference
  - Task 1: Compute a distribution over outputs
  - Task 2: Compute functions on distribution
    - **marginal probabilities**, expected values, entropies, divergences

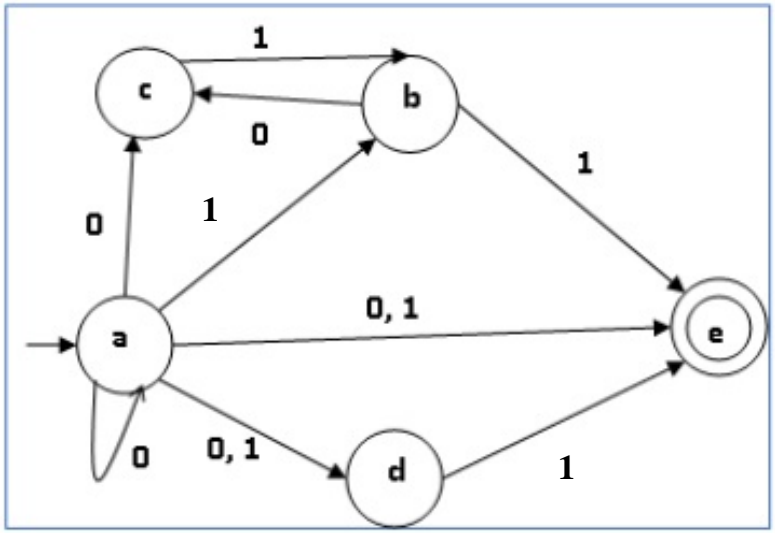
Why Soft Inference?

- Useful applications of posterior distributions
  - **Entropy**: how confused is the model?
  - **Entropy**: how confused is the model of its prediction at time *i*?
  - **Expectations**
    - What is the expected number of words in a translation of this sentence?
    - What is the expected number of times a word ending in –ed was tagged as something other than a verb?
  - **Posterior marginals**: given some input, how likely is it that some (***latent***) event of interest happened?

What we will cover

- Soft inference can be applied to any probabilistically defined model
  - Or weighted model in general
- We will specifically look at soft inference in
  - Regular grammars
    - FSGs / PFSGs
  - Context free grammars
    - HMMs / CFGs / PCFGs

Inference in Regular Languages

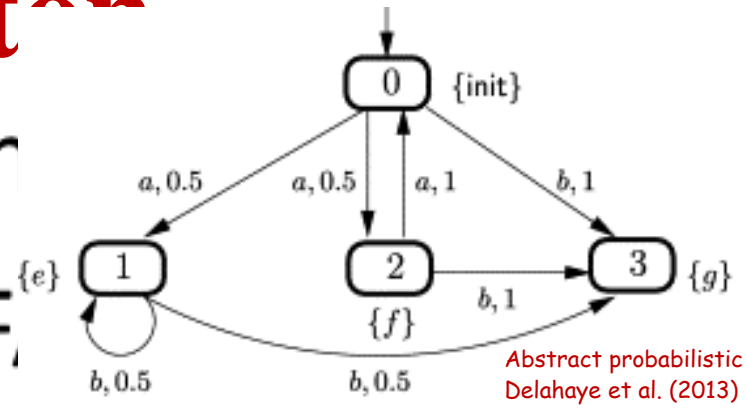


- Regular languages can be recognized by a DFA or an NDFA
  - Question answered: "Does this string belong to this language"

- Can we answer: *Is the state "b" visited in*

The probabilistic (finite) automat

- Probabilistic exter
- Conventional NDF



Abstract probabilistic automata
Delahaye et al. (2013)

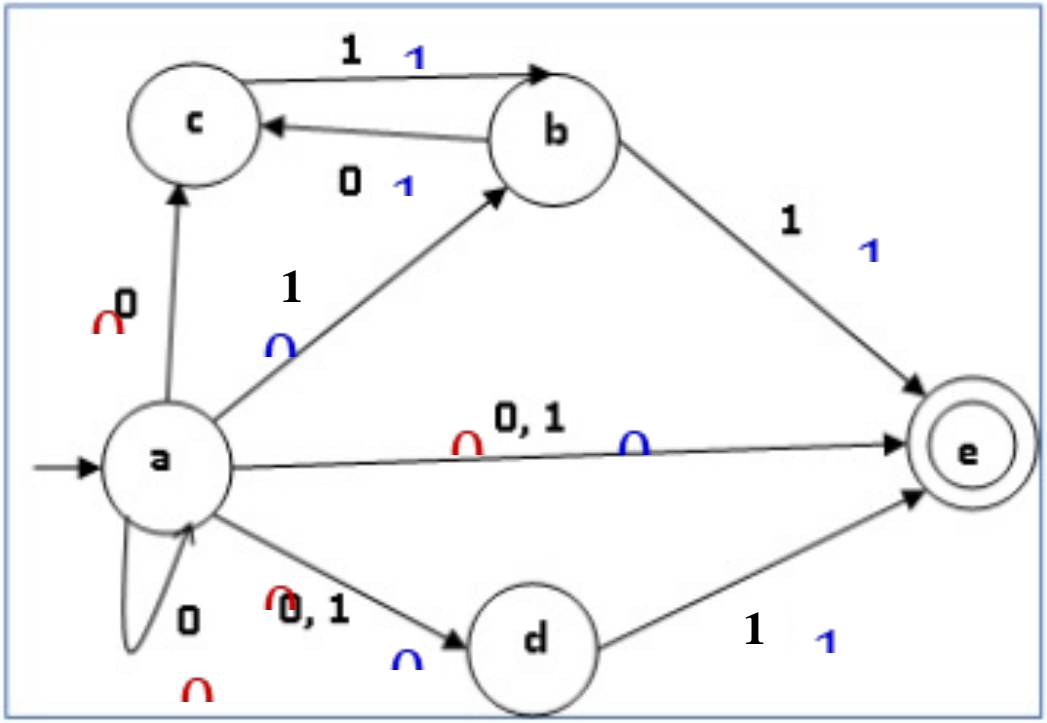
s_i –

s_i –

– State s_i can transition to both s_j

- PFA rules:

Inference in Regular Languages



aacbe

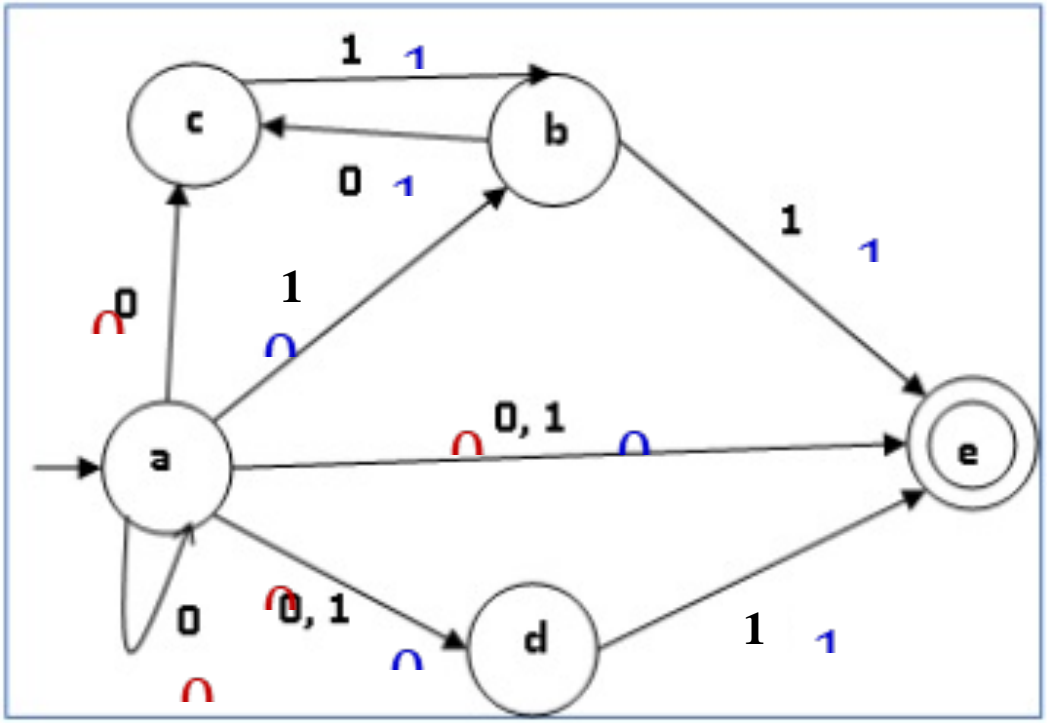
aaabe

aaade

- *What is the probability that the state "b" is visited in recognizing "00011"*

- Can now view the recognition as a random walk

Inference in a PFA



aacbe P = 0.004

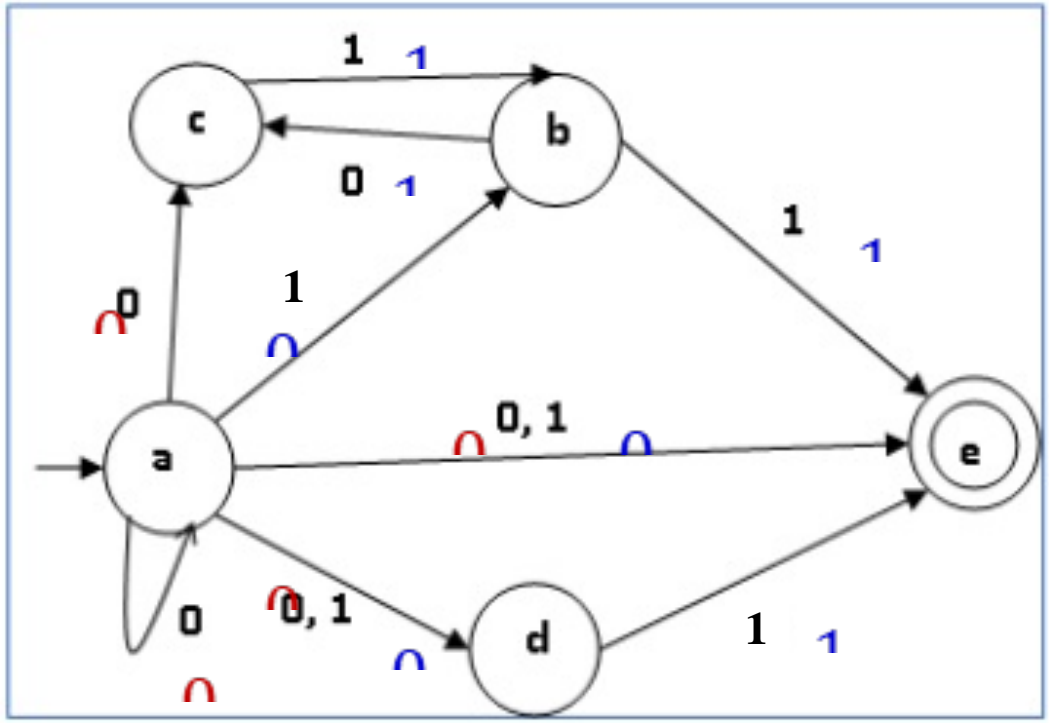
aaabe P = 0.0032

aaade P = 0.0008

- *What is the probability that the state "b" is visited in recognizing "00011"*

- Can now view the recognition as a random walk

Inference in a PFA



aacbe P = 0.004

aaabe P = 0.0032

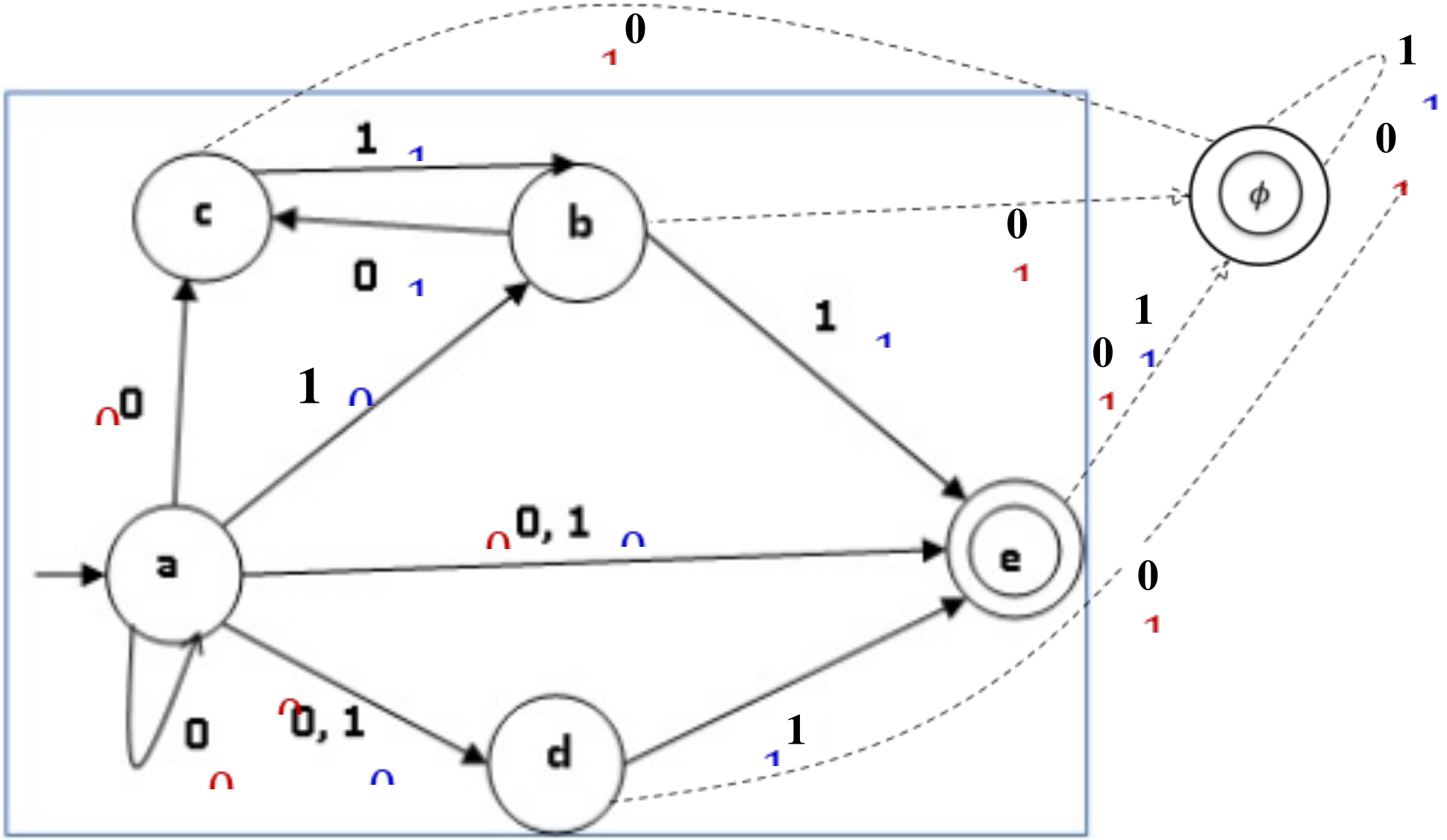
aaade P = 0.0008

Why don't these sum to 1.0?
Hint: figure is incomplete, but it doesn't affect our computation

- *What is the probability that the state "b" is visited in recognizing "00011"*

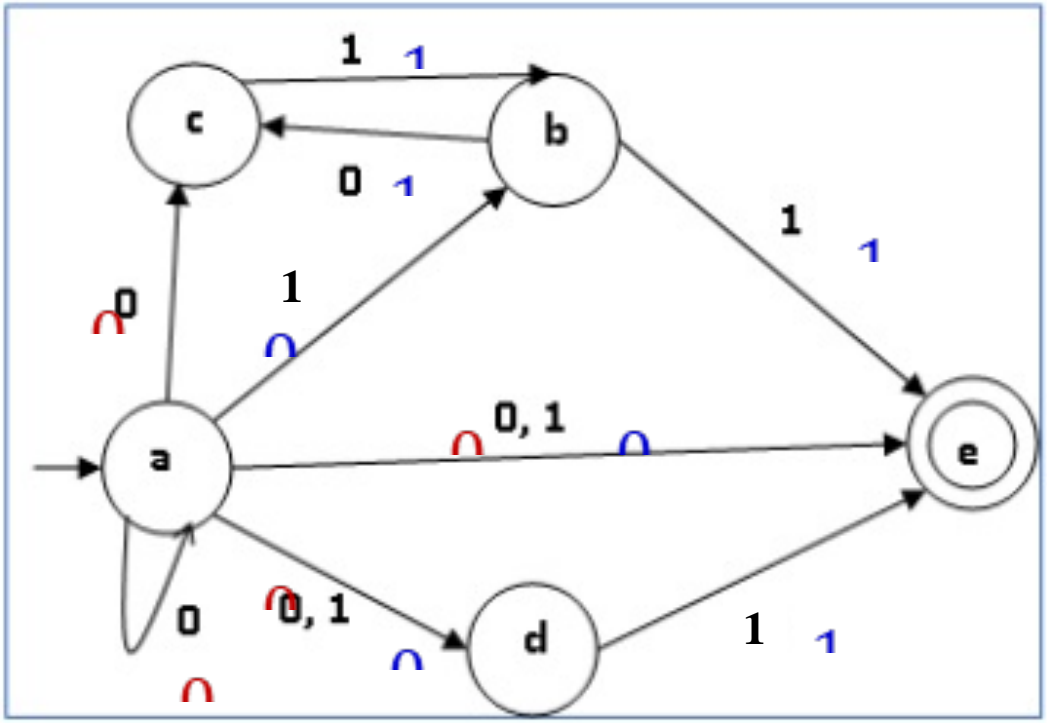
- Can now view the recognition as a random walk

Inference in a PFA



- We aren't interested in st

Inference in a PFA



aacbe P = 0.004

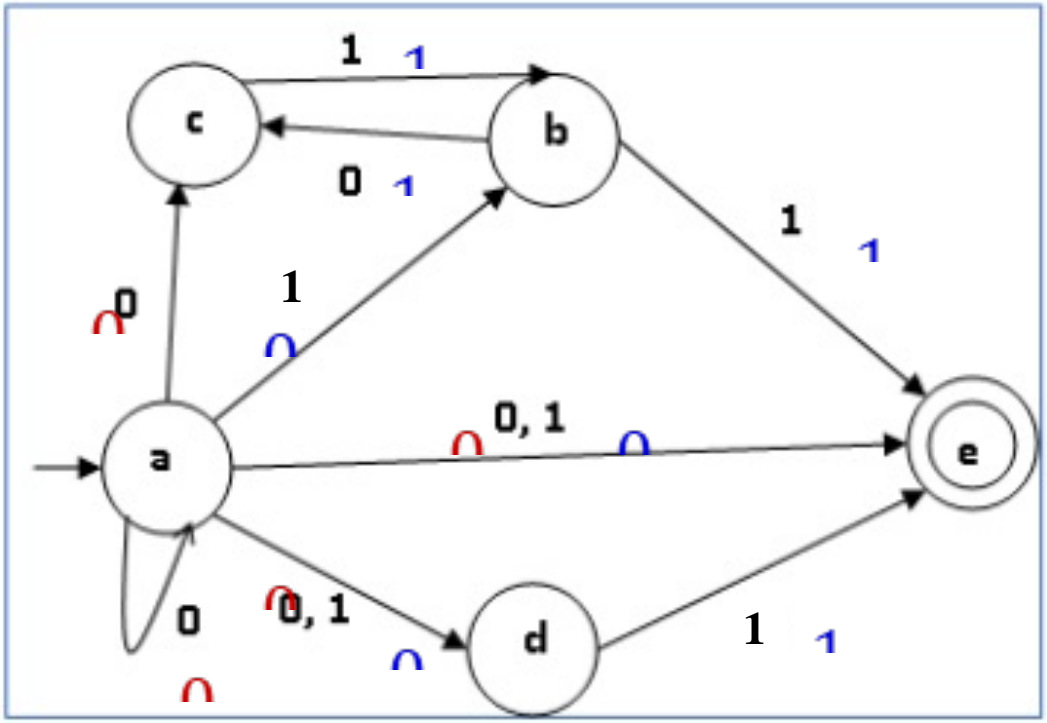
aaabe P = 0.0032

aaade P = 0.0008

P(visiting b) = 0.9

- *What is the probability that the state "b" is visited in recognizing "00011"*
  - *Given that the final state was e!*

Inference in a PFA



aacbe P = 0.004

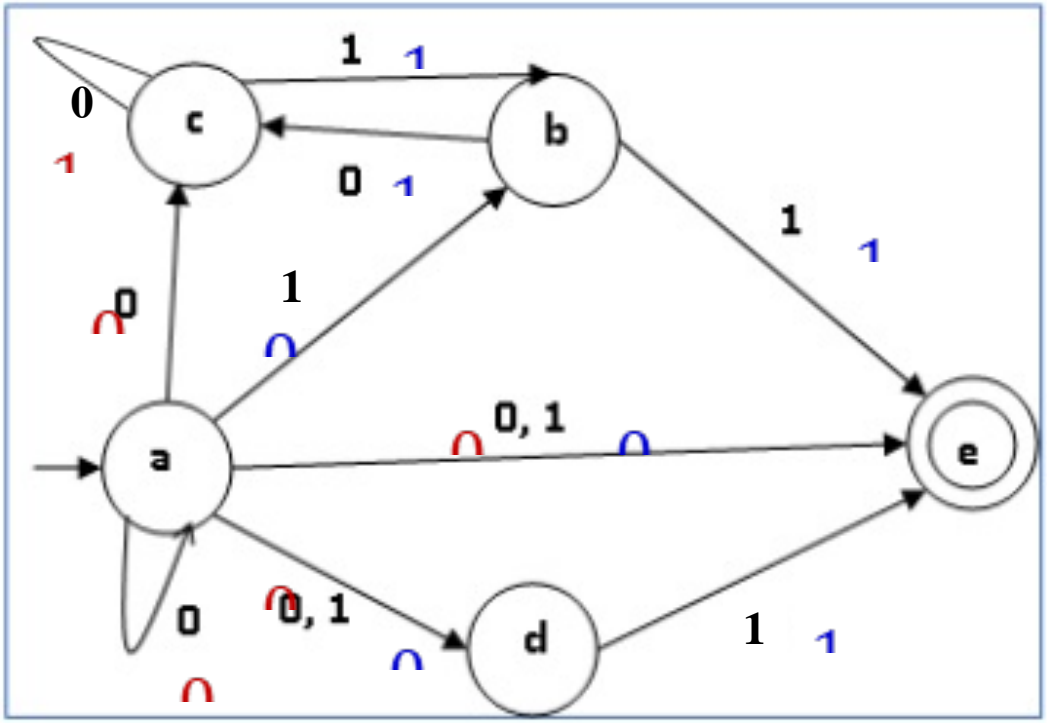
aaabe P = 0.0032

aaade P = 0.0008

P(visiting bl00011) = 0.9

- Note that we really need the probabilistic framework to make this statement
  - The PFA is actually a probability distribution over strings!!

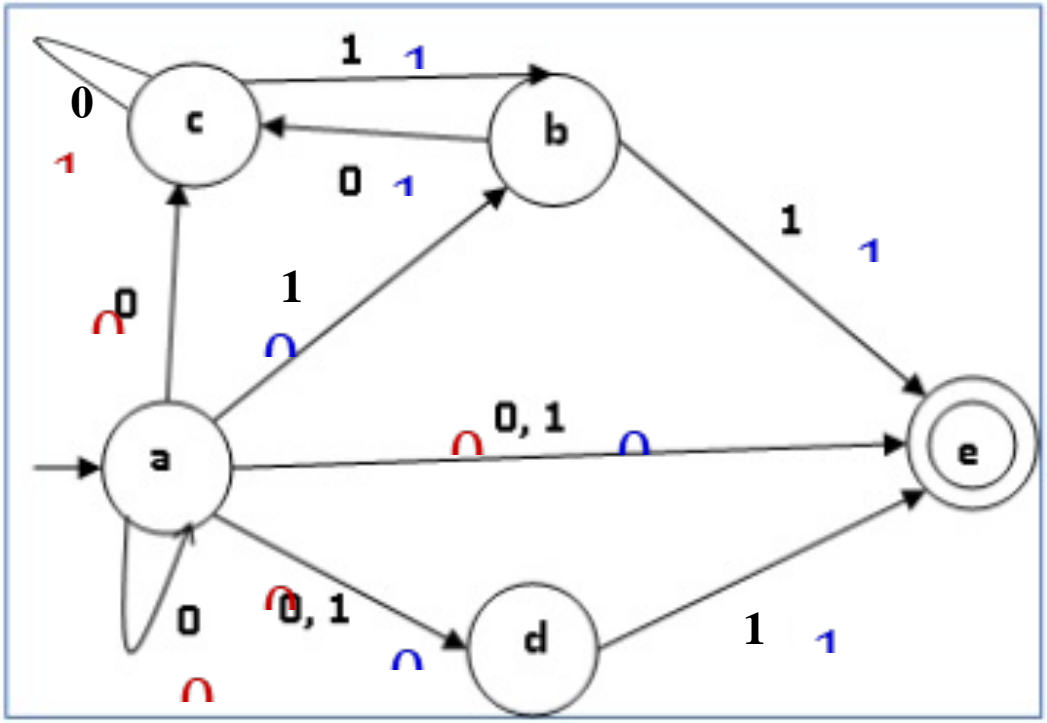
Inference in a PFA

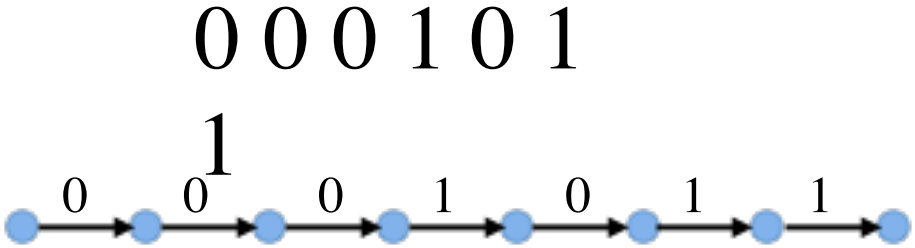


0 0 0 1 0 1
1

- Plot all the state sequences (ending in “e”) that can “consume” the symbol sequence to the right

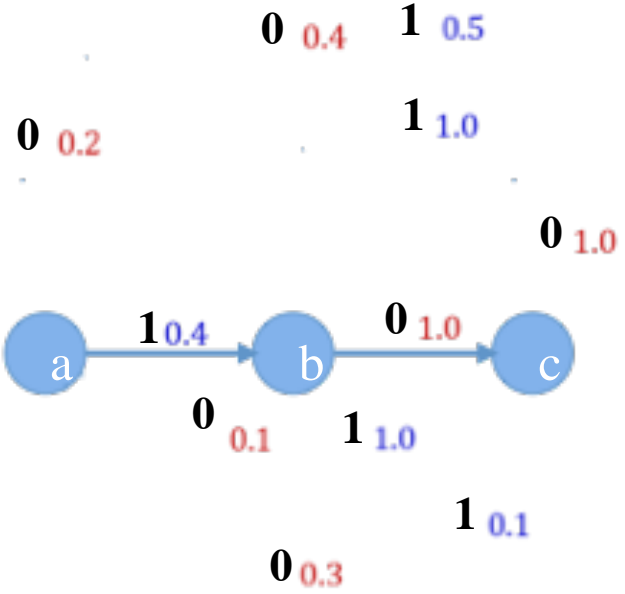
Inference in a PFA

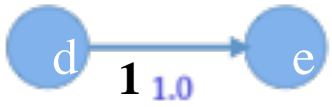


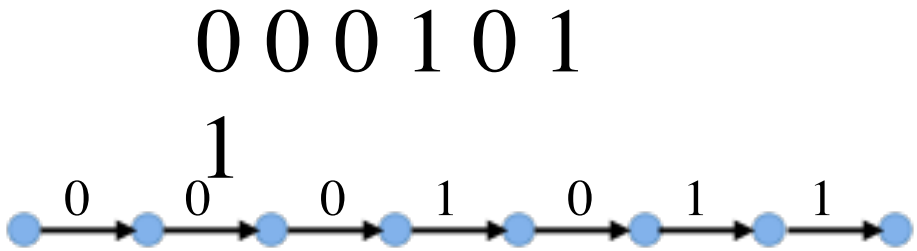


- Convert the string to an FSA
  - Note that the symbols appear on the *edges*
  - This is a *DFA* because the observed string is definitive

Inference in a PFA

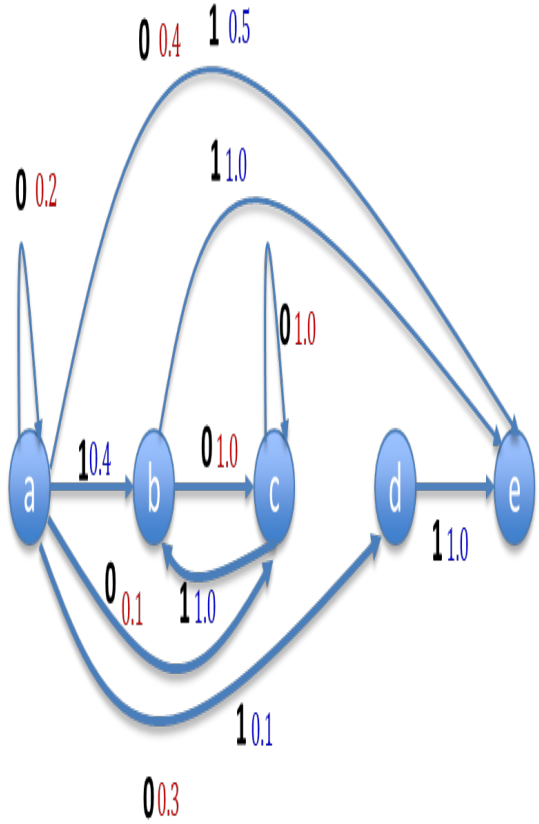


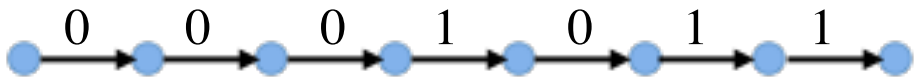




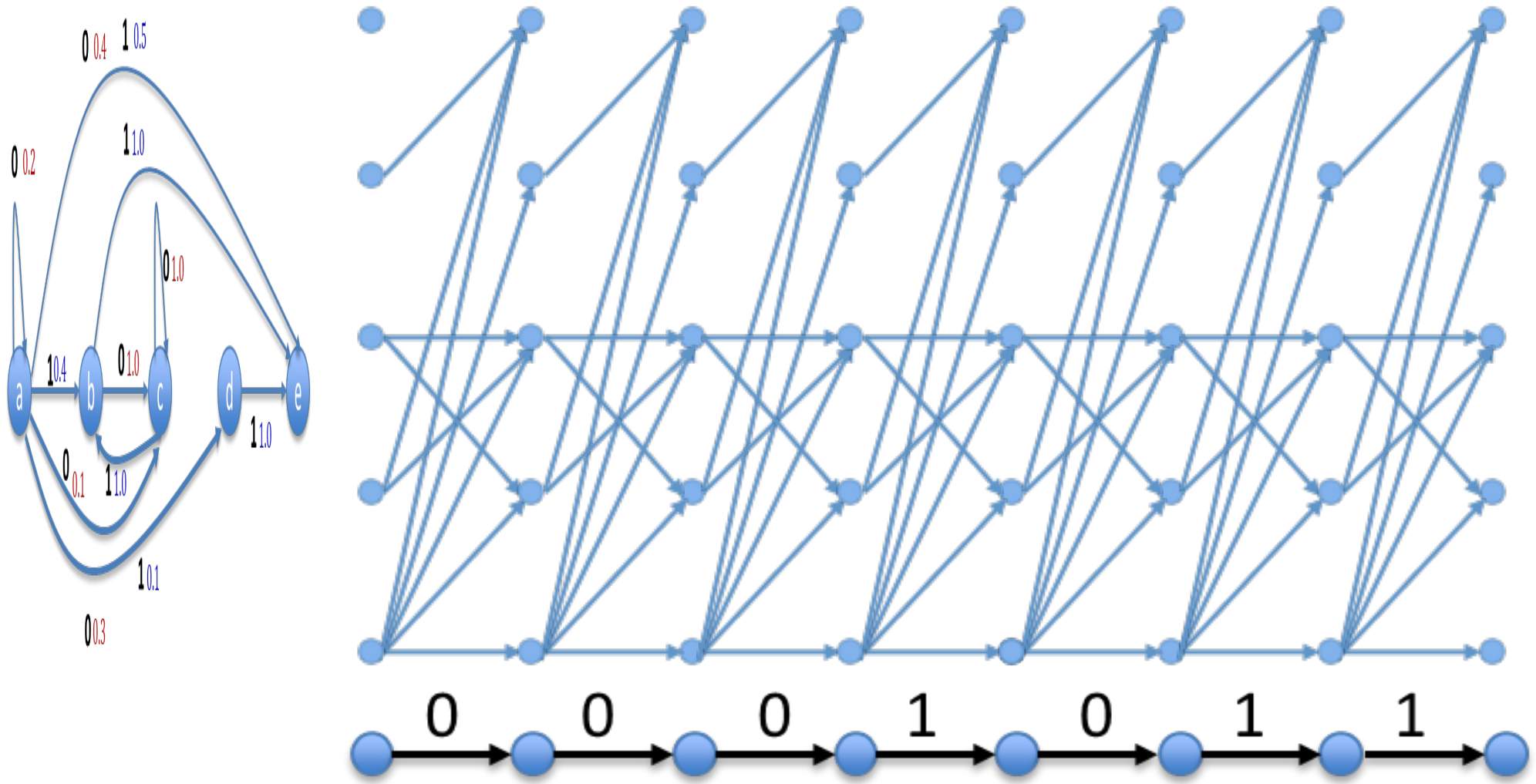
- Redrawing it linearly for illustration..

Inference in a PFA

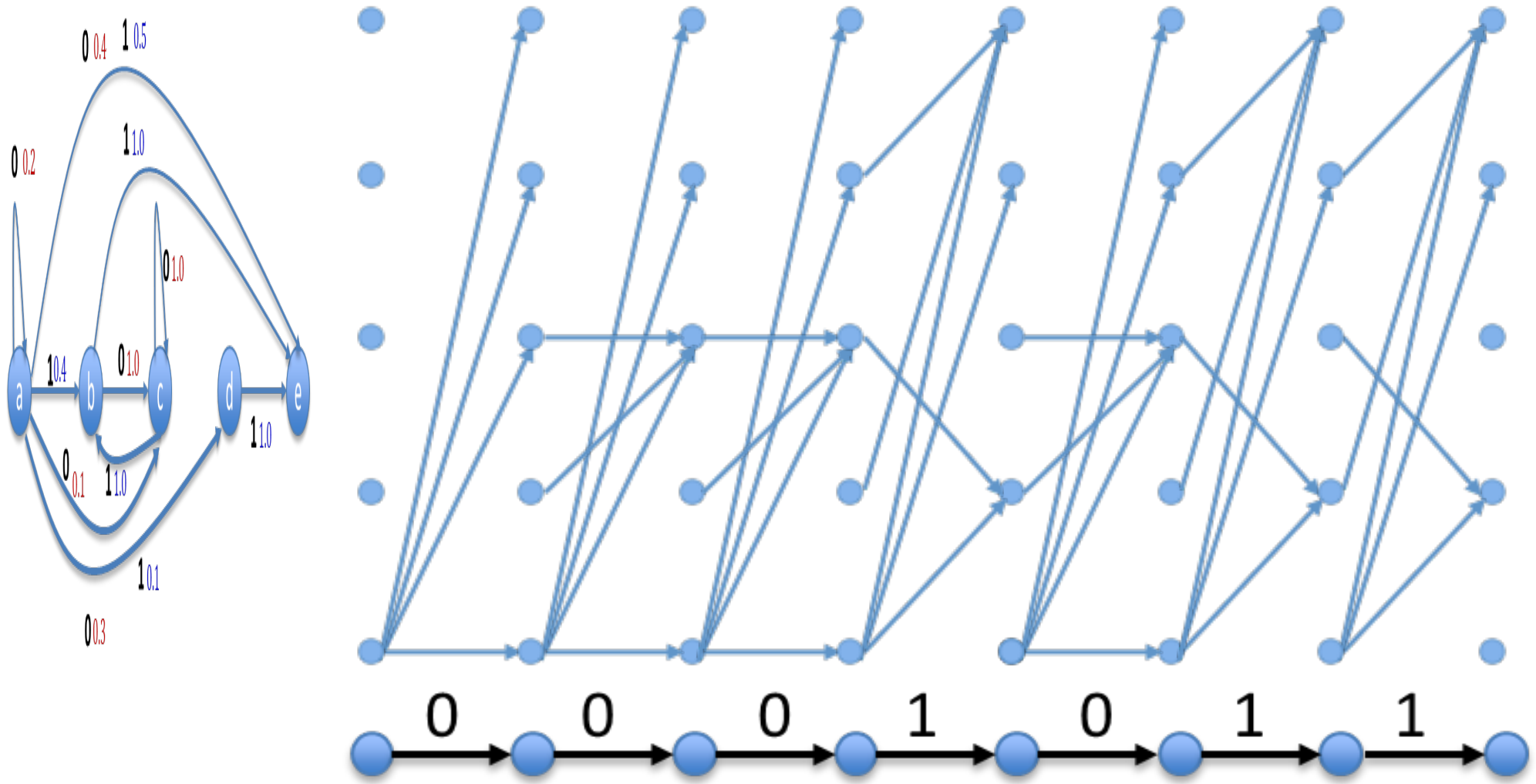




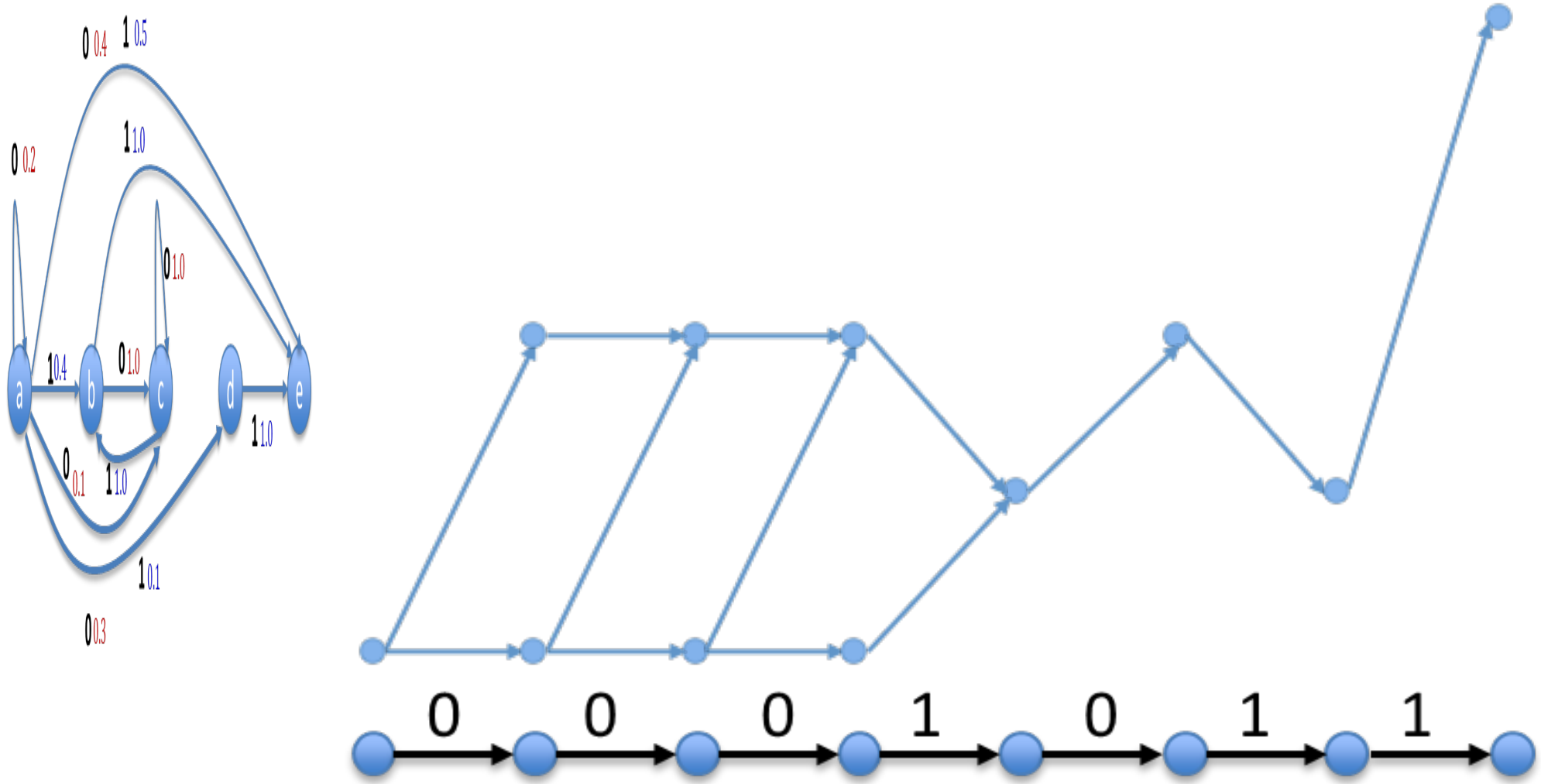
- Redrawing it linearly (and rotating it) for illustration..



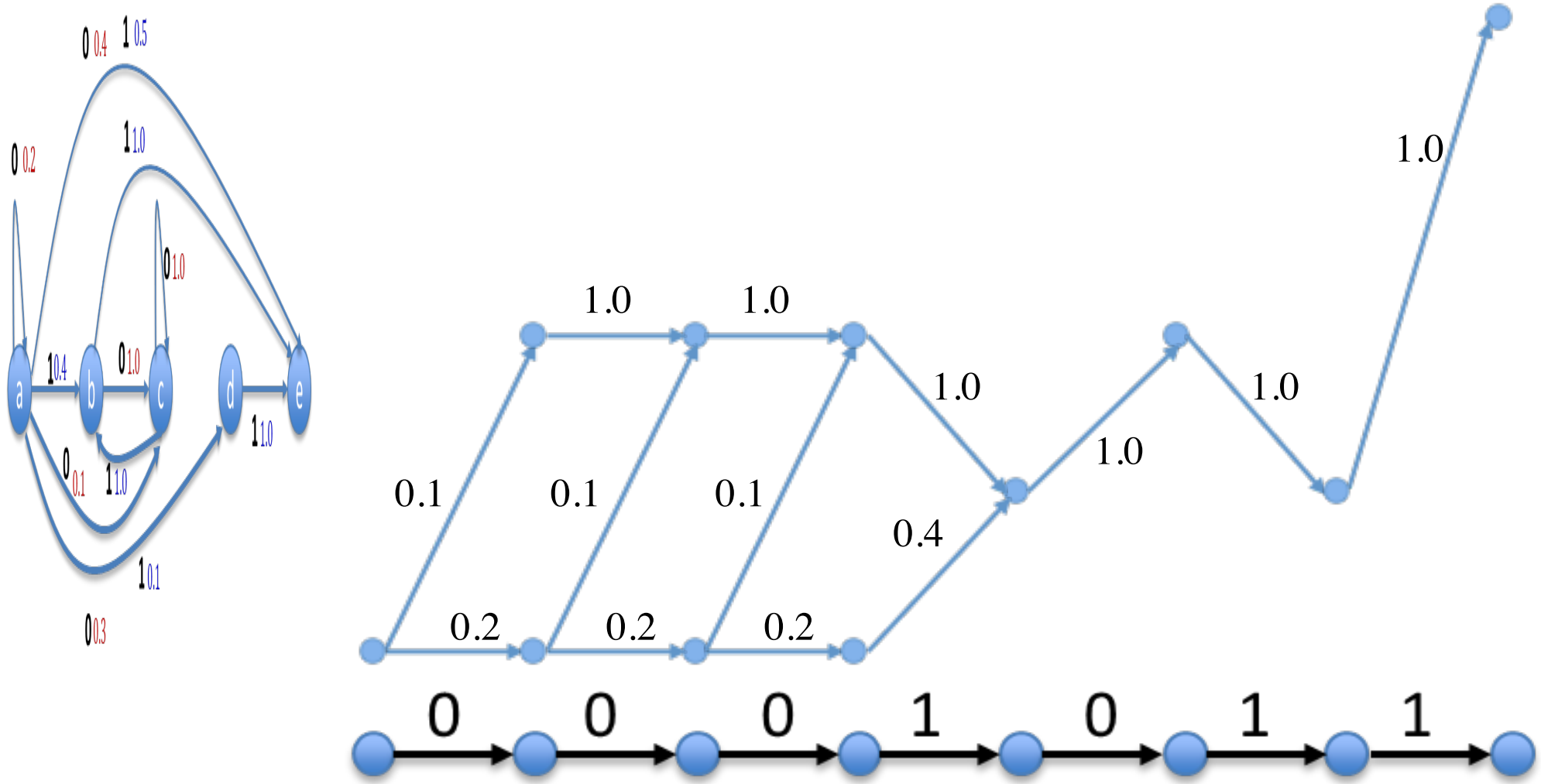
- This graph shows all paths that can consume *any* sequence of seven symbols



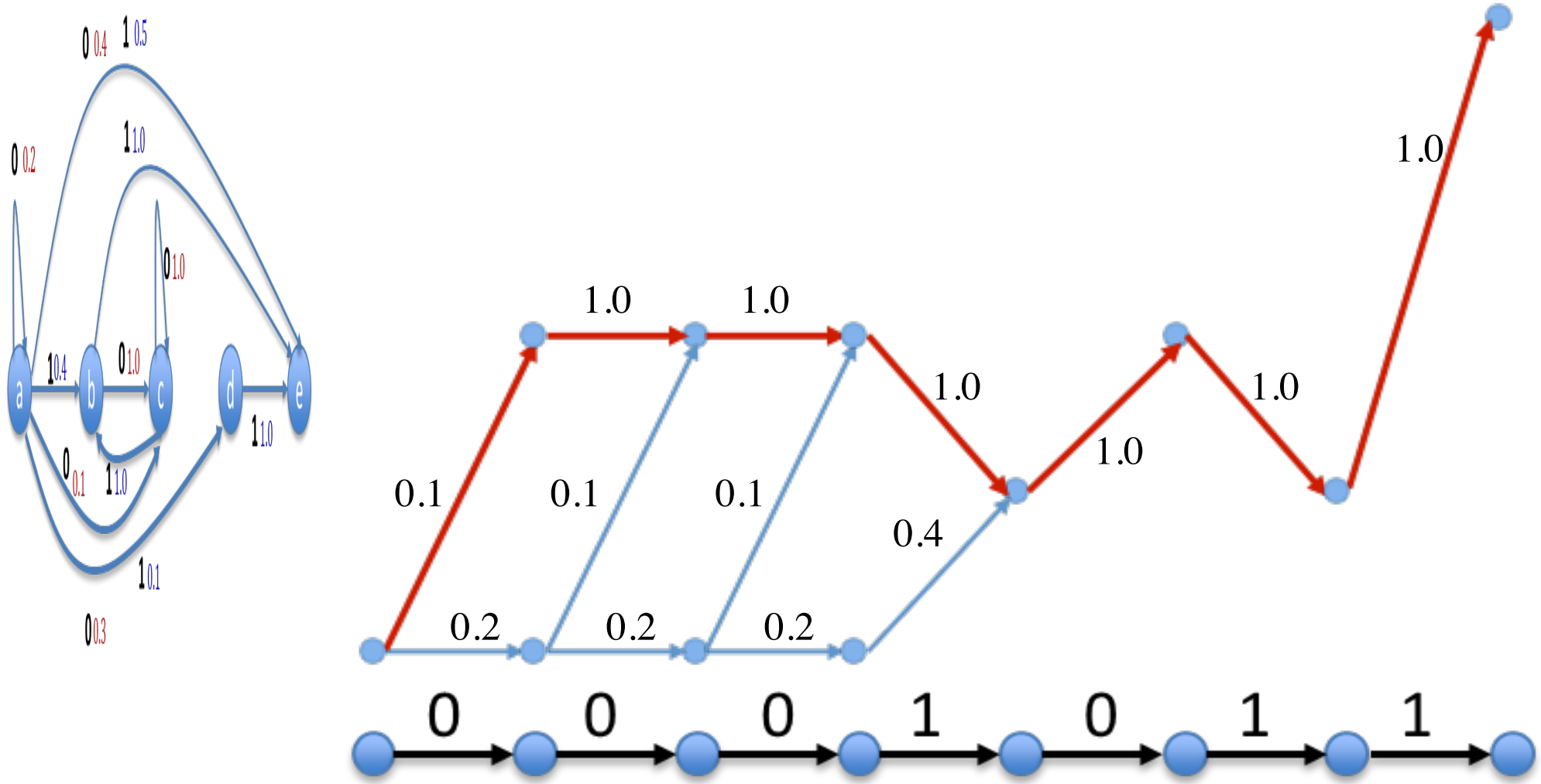
- Cleanup: Eliminate all nodes without incoming edges, and all nodes (except in the last column)



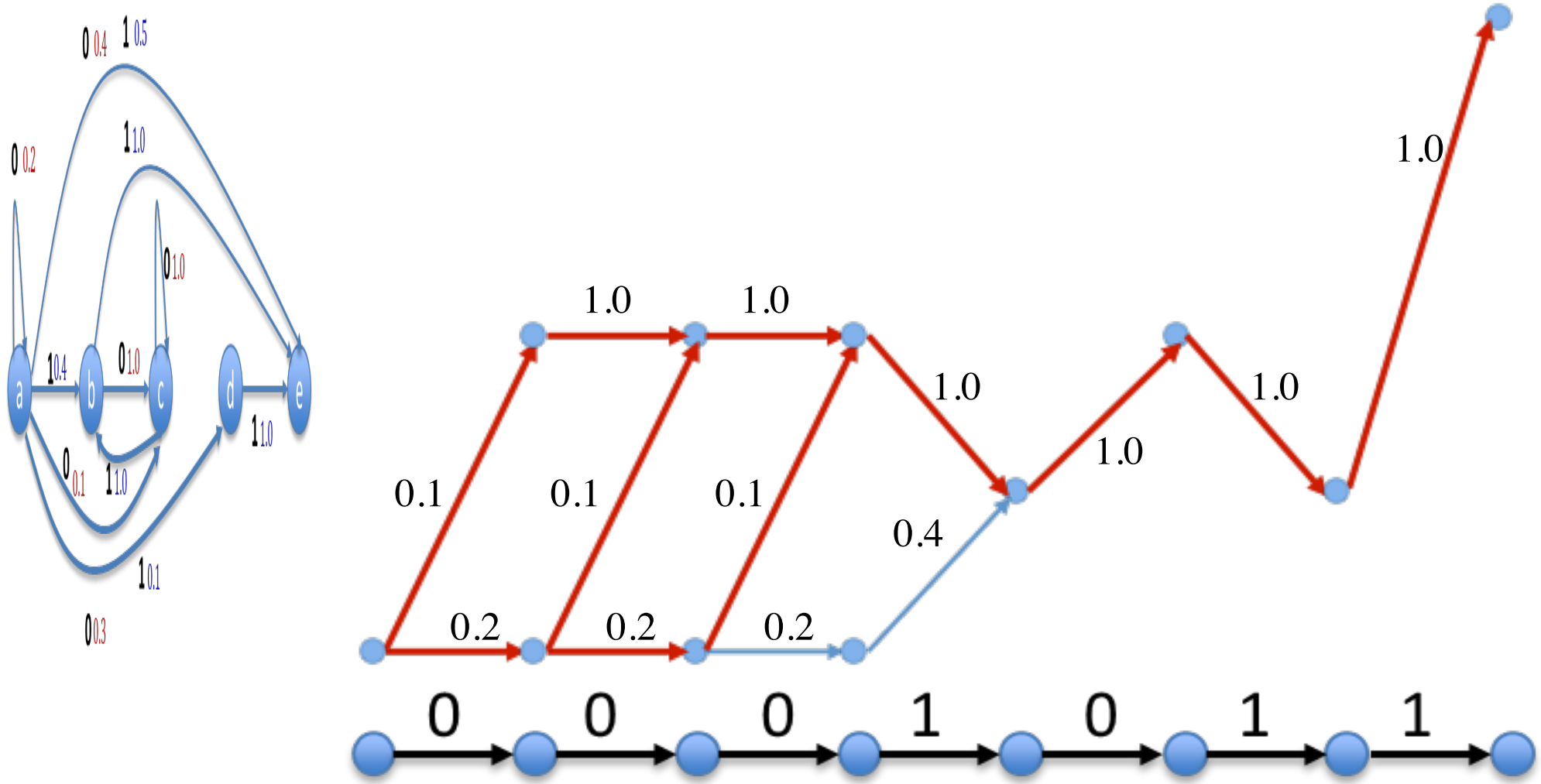
- The complete set of all paths that can absorb the observed sequence



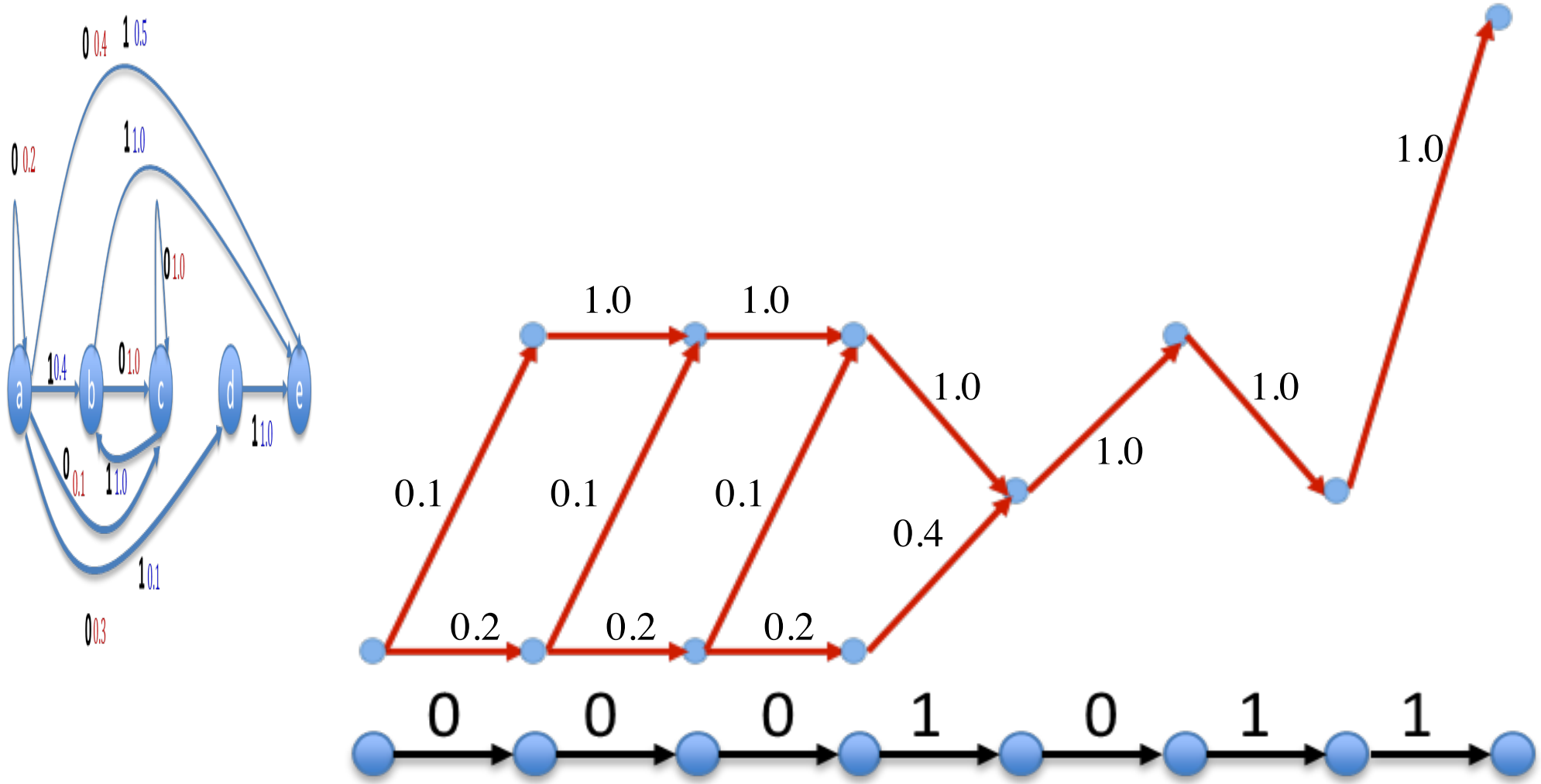
- The complete set of all paths that can absorb the observed sequence



- The probability of any given state sequence is the product of the probabilities on all the edges



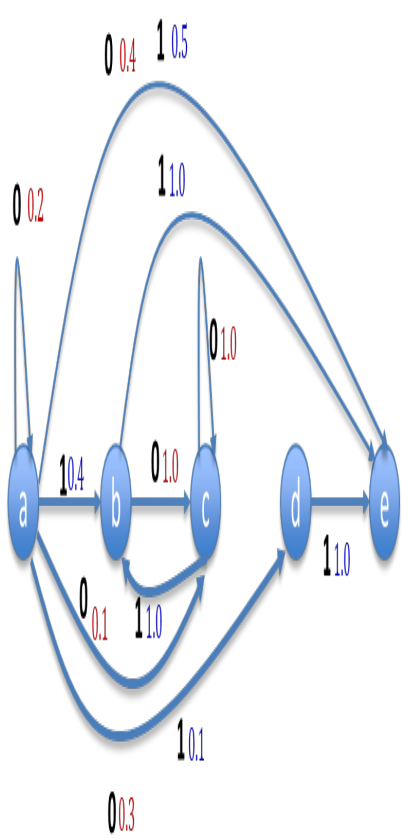
- The total probability of visiting “c” is the total probability of all paths that go through “c” and

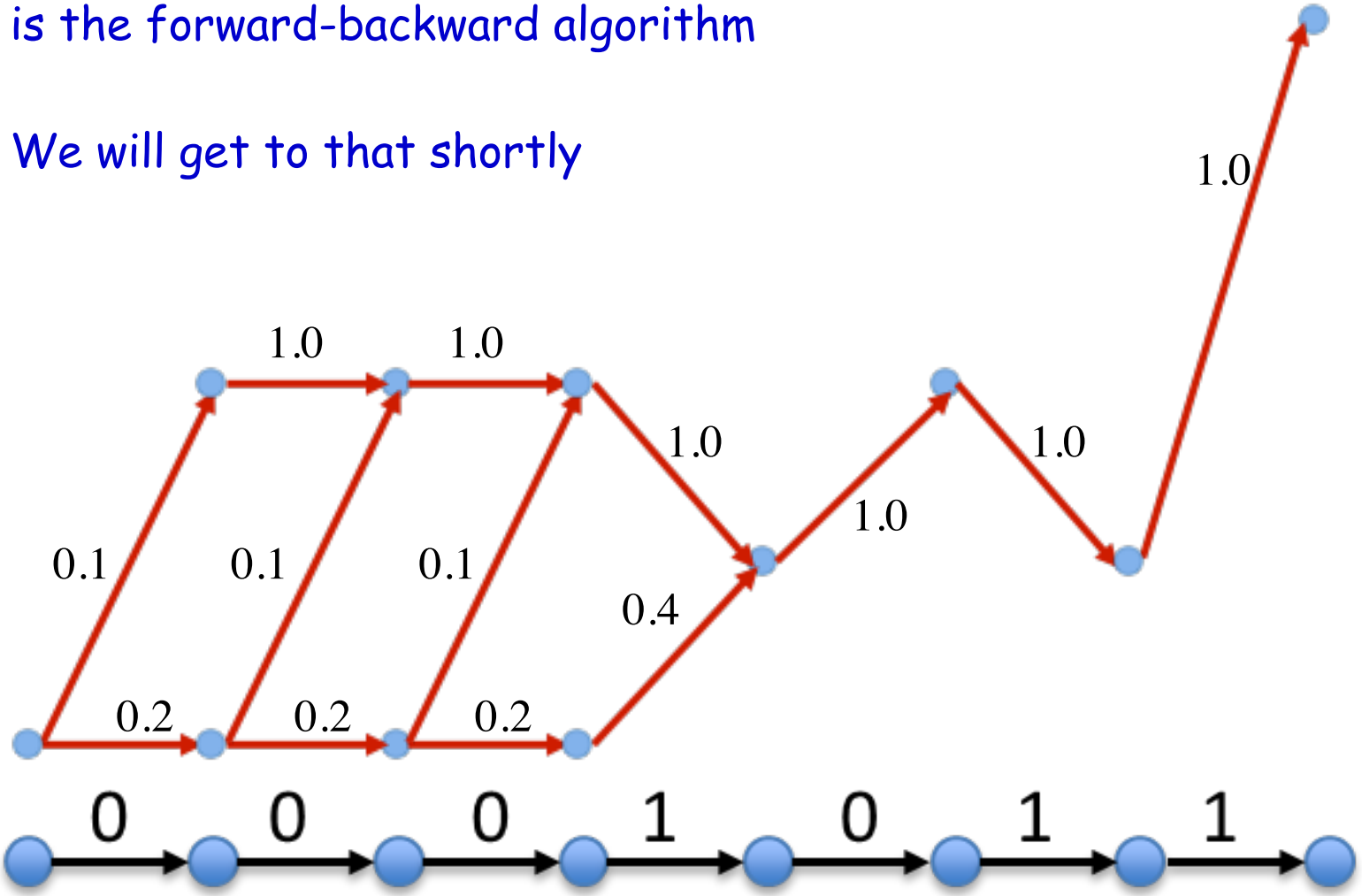


- The total probability of all paths that get to the final state “e” is the probability of the entire

The actual algorithm to compute these probabilities is the forward-backward algorithm

We will get to that shortly



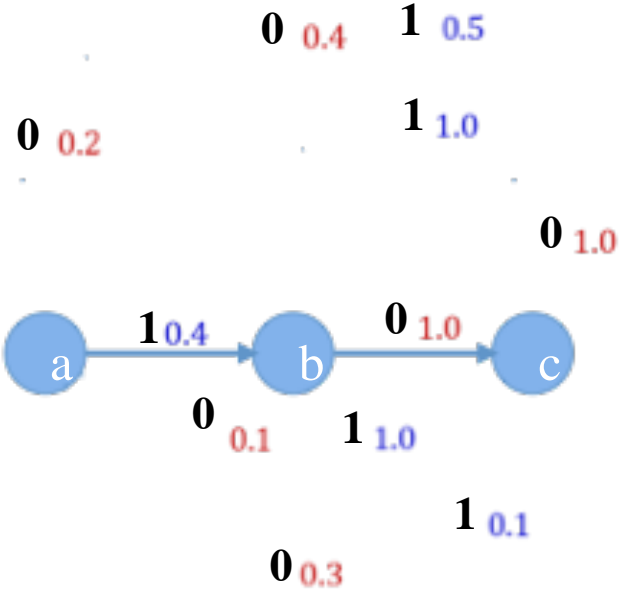


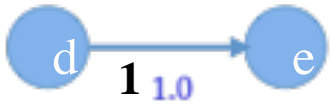
- The total probability of all paths that get to the final state “e” is the probability of the entire

Composition and computation

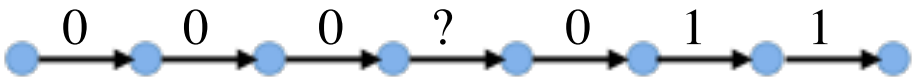
- Composition and computation can be done dynamically as one processes the input string
- Alternately, one may use any of the FSA composition algorithms in the literature (and tools available on the web)
  - These can be highly efficient

Dealing with uncertainty..



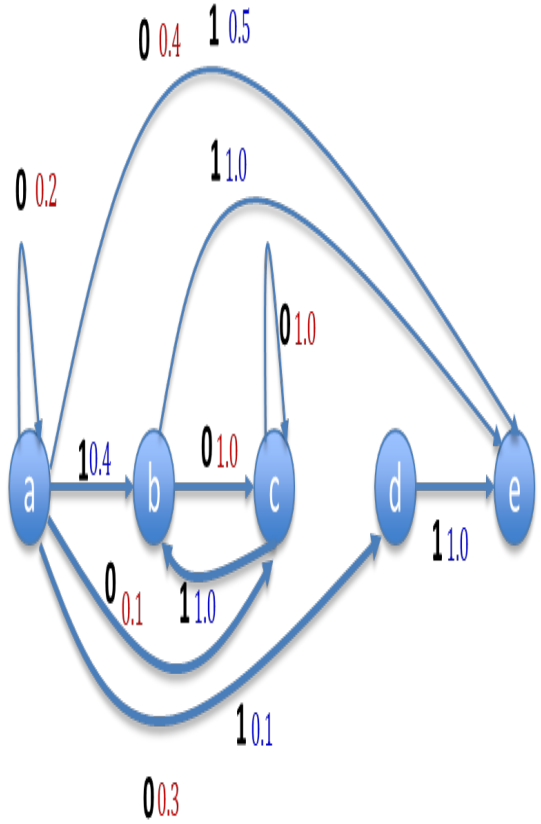


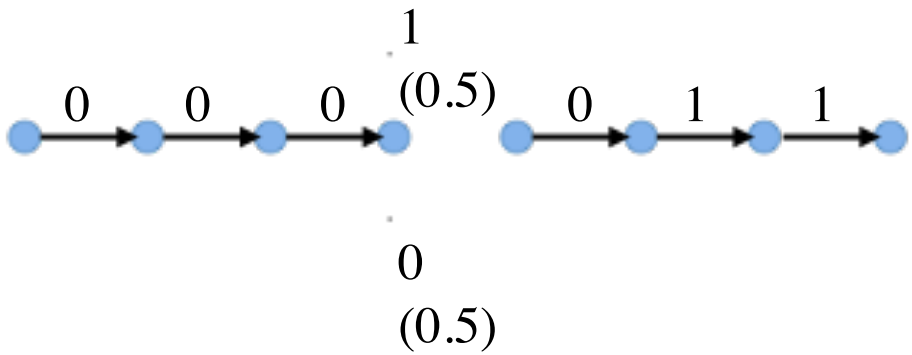
0 0 0 ? 0 1 1



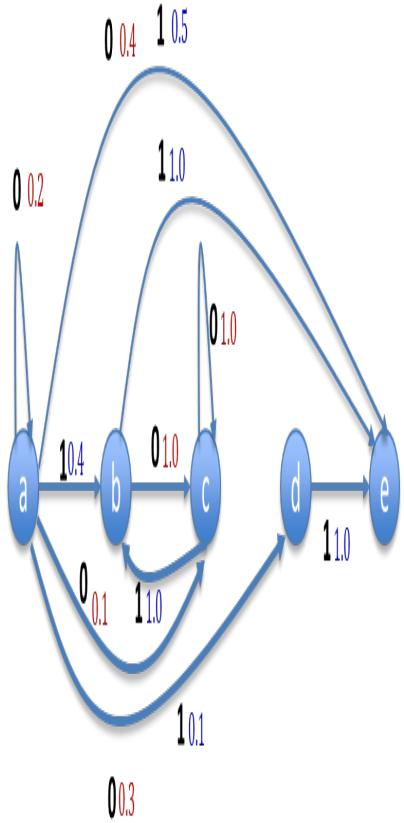
- Easily adapted to deal with uncertainty..

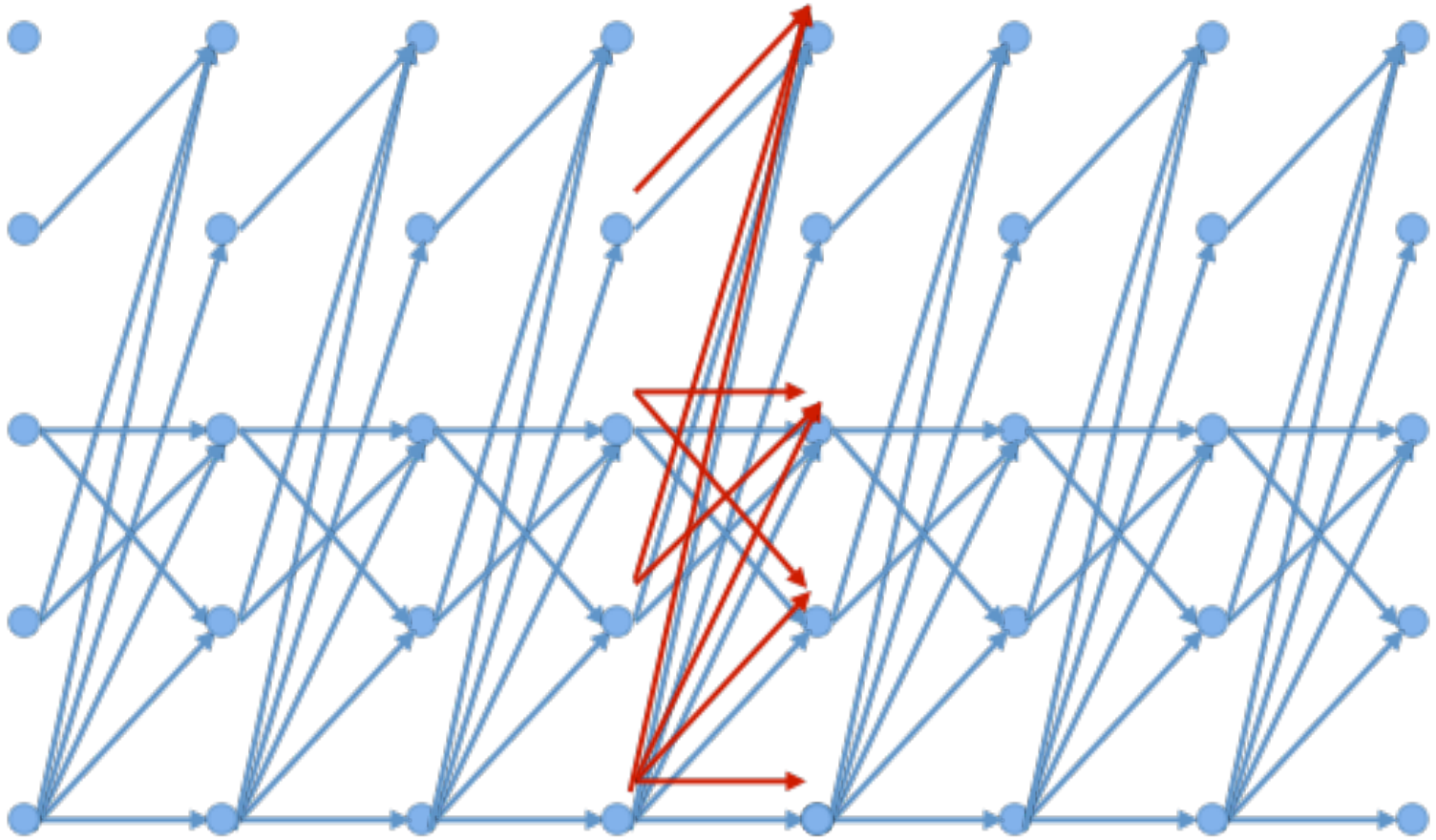
Inference in a PFA





- Uncertainty is reflected in the input string
- The rest of the process remains largely





1 (0.5)

0 (0.5)

Moving on: Generative models

- Hidden Markov Models
  - "Stochastic functions of Markov Chains"

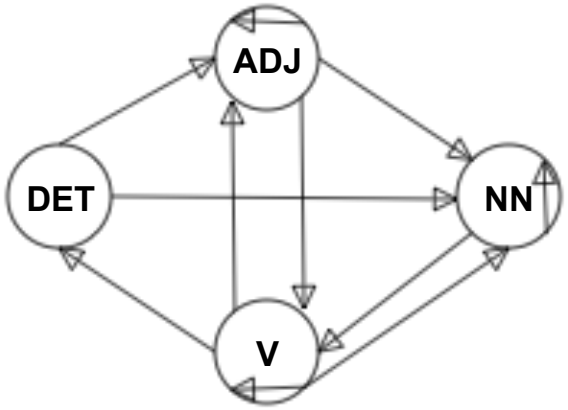
- E.g. a finite-state automaton over tags, that can generate word sequences

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.2	0.1	0.4	0.1
V	0.0	0.1	0.1	0.2



Emission Probabilities:

DET		ADJ		NN		V	
the	0.7	green	0.1	book	0.3	might	0.2
a	0.3	big	0.4	plants	0.2	watch	0.3
		old	0.4	people	0.2	watches	0.2
		might	0.1	person	0.1	loves	0.1
				John	0.1	reads	0.19
				watch	0.1	books	0.01

Examples:

John	might	watch
NN	V	V

the	old	person	loves	big	books
DET	ADJ	NN	V	ADJ	NN

String Marginals

- Inference question for HMMs
  - What is the probability of a string **w**?
    Answer: generate all possible tag sequences and explicitly *marginalize*

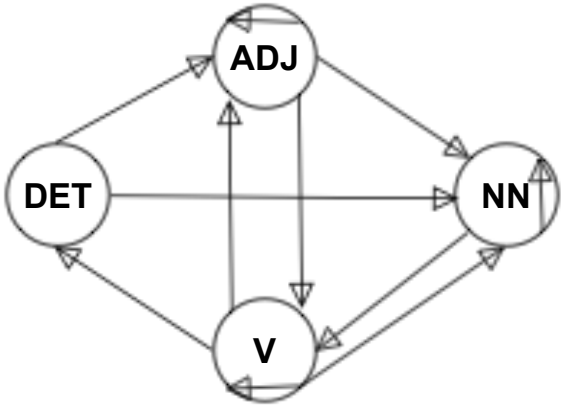
$$O(|\Omega|^{|\mathbf{w}|}) \quad \text{time}$$

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.2	0.1	0.1	0.1
V	0.1	0.0	0.1	0.1



Emission Probabilities:

DET		ADJ		NN		V	
the	0.7	green	0.1	book	0.3	might	0.2
a	0.3	big	0.4	plants	0.2	watch	0.3
		old	0.4	people	0.2	watches	0.2
		might	0.1	person	0.1	loves	0.1
				John	0.1	reads	0.19
				watch	0.1	books	0.01

Examples:

John might watch

NN V V

the old person loves big books

John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$
DET	DET	DET	0.0	ADJ	DET	DET	0.0	NN	DET	DET	0.0	V	DET	DET	0.0
DET	DET	ADJ	0.0	ADJ	DET	ADJ	0.0	NN	DET	ADJ	0.0	V	DET	ADJ	0.0
DET	DET	NN	0.0	ADJ	DET	NN	0.0	NN	DET	NN	0.0	V	DET	NN	0.0
DET	DET	V	0.0	ADJ	DET	V	0.0	NN	DET	V	0.0	V	DET	V	0.0
DET	ADJ	DET	0.0	ADJ	ADJ	DET	0.0	NN	ADJ	DET	0.0	V	ADJ	DET	0.0
DET	ADJ	ADJ	0.0	ADJ	ADJ	ADJ	0.0	NN	ADJ	ADJ	0.0	V	ADJ	ADJ	0.0
DET	ADJ	NN	0.0	ADJ	ADJ	NN	0.0	**NN**	**ADJ**	**NN**	**0.0000042**	V	ADJ	NN	0.0
DET	ADJ	V	0.0	ADJ	ADJ	V	0.0	**NN**	**ADJ**	**V**	**0.0000009**	V	ADJ	V	0.0
DET	NN	DET	0.0	ADJ	NN	DET	0.0	NN	NN	DET	0.0	V	NN	DET	0.0
DET	NN	ADJ	0.0	ADJ	NN	ADJ	0.0	NN	NN	ADJ	0.0	V	NN	ADJ	0.0
DET	NN	NN	0.0	ADJ	NN	NN	0.0	NN	NN	NN	0.0	V	NN	NN	0.0
DET	NN	V	0.0	ADJ	NN	V	0.0	NN	NN	V	0.0	V	NN	V	0.0
								NN	V	DE	0.0	V	V	DE	0.0

John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$
DET	DET	DET	0.0	ADJ	DET	DET	0.0	NN	DET	DET	0.0	V	DET	DET	0.0
DET	DET	ADJ	0.0	ADJ	DET	ADJ	0.0	NN	DET	ADJ	0.0	V	DET	ADJ	0.0
DET	DET	NN	0.0	ADJ	DET	NN	0.0	NN	DET	NN	0.0	V	DET	NN	0.0
DET	DET	V	0.0	ADJ	DET	V	0.0	NN	DET	V	0.0	V	DET	V	0.0
DET	ADJ	DET	0.0	ADJ	ADJ	DET	0.0	NN	ADJ	DET	0.0	V	ADJ	DET	0.0
DET	ADJ	ADJ	0.0	ADJ	ADJ	ADJ	0.0	NN	ADJ	ADJ	0.0	V	ADJ	ADJ	0.0
DET	ADJ	NN	0.0	ADJ	ADJ	NN	0.0	**NN**	**ADJ**	**NN**	**0.0000042**	V	ADJ	NN	0.0
DET	ADJ	V	0.0	ADJ	ADJ	V	0.0	**NN**	**ADJ**	**V**	**0.0000009**	V	ADJ	V	0.0
DET	NN	DET	0.0	ADJ	NN	DET	0.0	NN	NN	DET	0.0	V	NN	DET	0.0
DET	NN	ADJ	0.0	ADJ	NN	ADJ	0.0	NN	NN	ADJ	0.0	V	NN	ADJ	0.0
DET	NN	NN	0.0	ADJ	NN	NN	0.0	NN	NN	NN	0.0	V	NN	NN	0.0
DET	NN	V	0.0	ADJ	NN	V	0.0	NN	NN	V	0.0	V	NN	V	0.0
								NN	V	DET	0.0	V	V	DET	0.0

$p = 0.0000219$

John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$	John	might	watch	$\Pr(x, y)$
DET	DET	DET	0.0	ADJ	DET	DET	0.0	NN	DET	DET	0.0	V	DET	DET	0.0
DET	DET	ADJ	0.0	ADJ	DET	ADJ	0.0	NN	DET	ADJ	0.0	V	DET	ADJ	0.0
DET	DET	NN	0.0	ADJ	DET	NN	0.0	NN	DET	NN	0.0	V	DET	NN	0.0
DET	DET	V	0.0	ADJ	DET	V	0.0	NN	DET	V	0.0	V	DET	V	0.0
DET	ADJ	DET	0.0	ADJ	ADJ	DET	0.0	NN	ADJ	DET	0.0	V	ADJ	DET	0.0
DET	ADJ	ADJ	0.0	ADJ	ADJ	ADJ	0.0	NN	ADJ	ADJ	0.0	V	ADJ	ADJ	0.0
DET	ADJ	NN	0.0	ADJ	ADJ	NN	0.0	**NN**	**ADJ**	**NN**	**0.0000042**	V	ADJ	NN	0.0
DET	ADJ	V	0.0	ADJ	ADJ	V	0.0	**NN**	**ADJ**	**V**	**0.0000009**	V	ADJ	V	0.0
DET	NN	DET	0.0	ADJ	NN	DET	0.0	NN	NN	DET	0.0	V	NN	DET	0.0
DET	NN	ADJ	0.0	ADJ	NN	ADJ	0.0	NN	NN	ADJ	0.0	V	NN	ADJ	0.0
DET	NN	NN	0.0	ADJ	NN	NN	0.0	NN	NN	NN	0.0	V	NN	NN	0.0
DET	NN	V	0.0	ADJ	NN	V	0.0	NN	NN	V	0.0	V	NN	V	0.0
NN	V	DET	0.0	V	V	DET	0.0								

Exponential computation, if done naively. $p = 0.0000219$

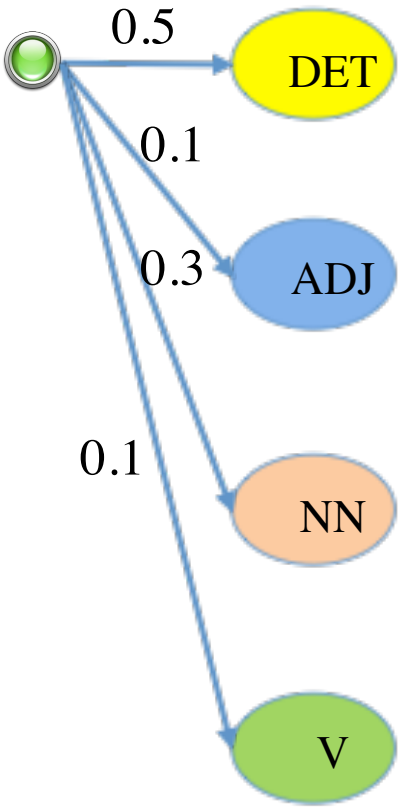
A different perspective

- “Graphical” view of the generative process..

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



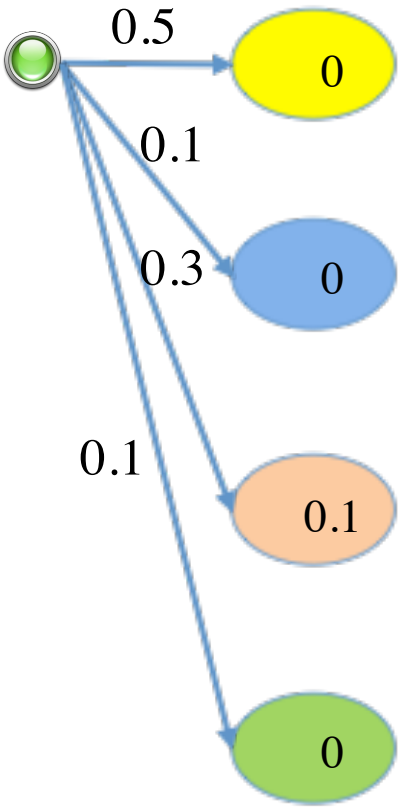
Initial Probabilities:

DET ADJ NN V

JOHN MIGHT
WATCH

0.5

Emission Probabilities:



DET	
the	0.7
a	0.3

ADJ	
green	0.1
big	0.4
old	0.4
might	0.1

NN	
book	0.3
plants	0.2
people	0.2
person	0.1
John	0.1
watch	0.1

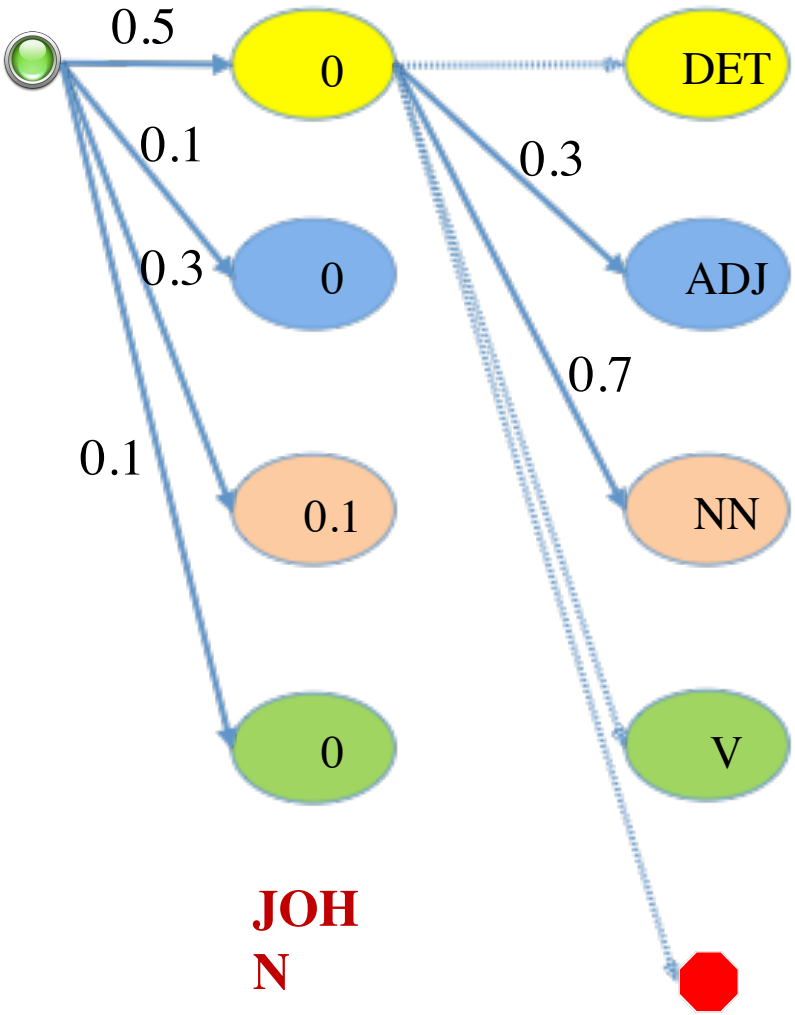
V	
might	0.2
watch	0.3
watches	0.2
loves	0.1
reads	0.19
books	0.01

JOHN

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



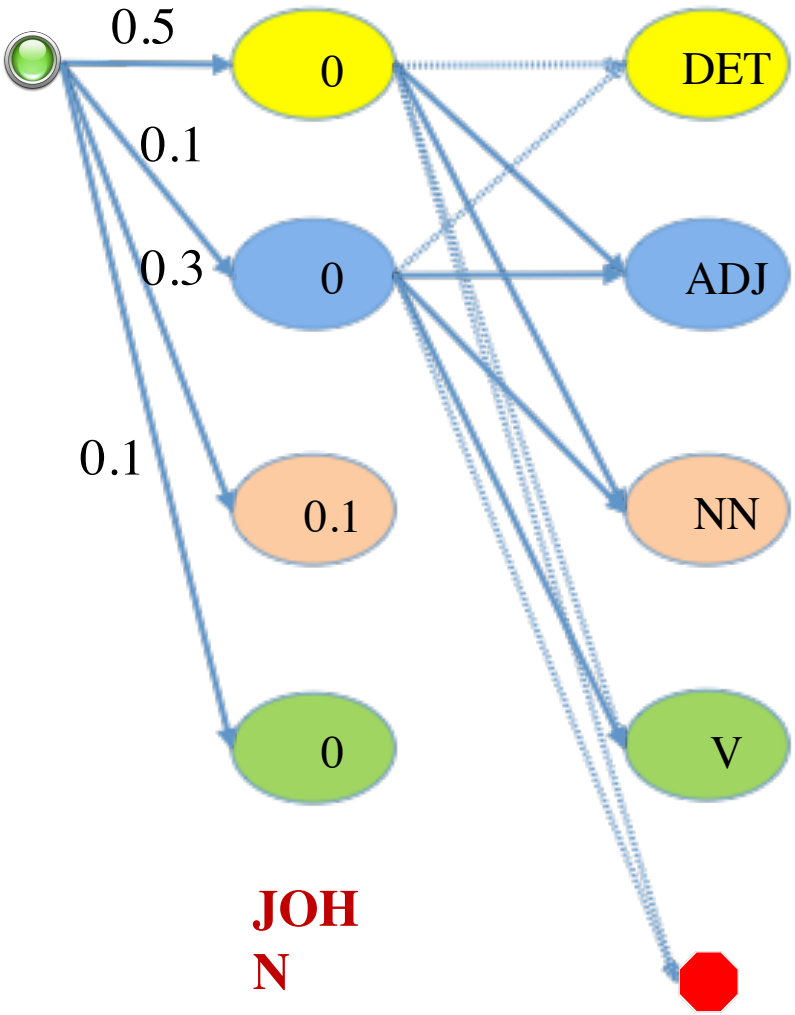
Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



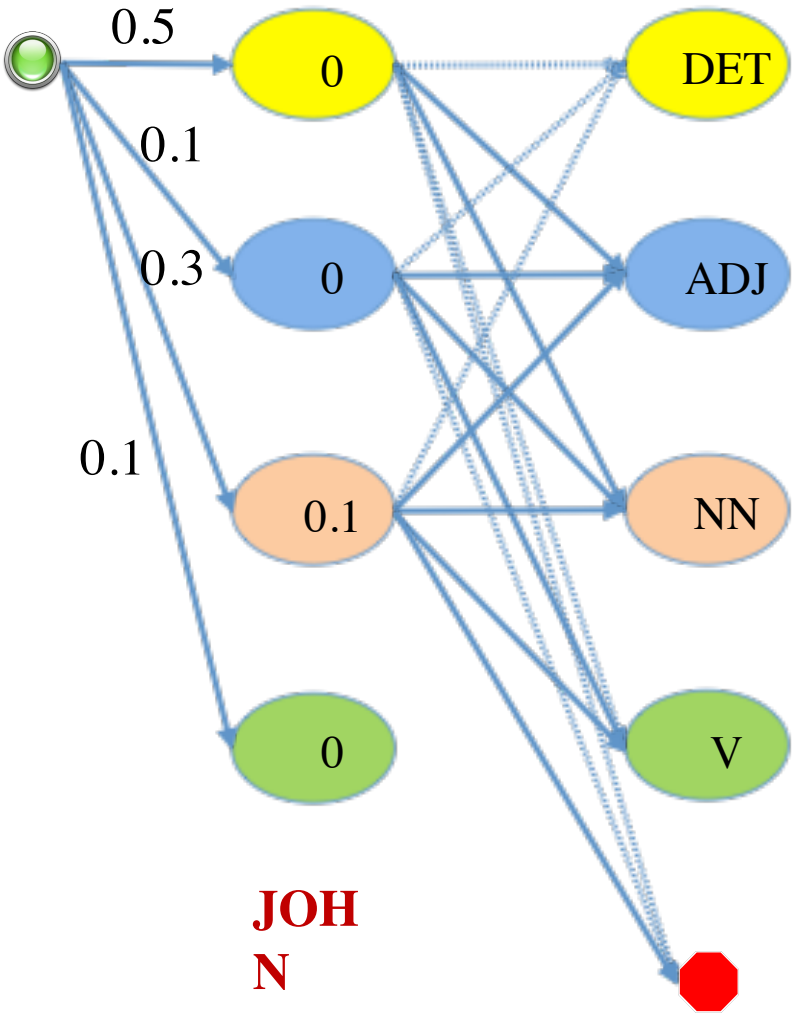
Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



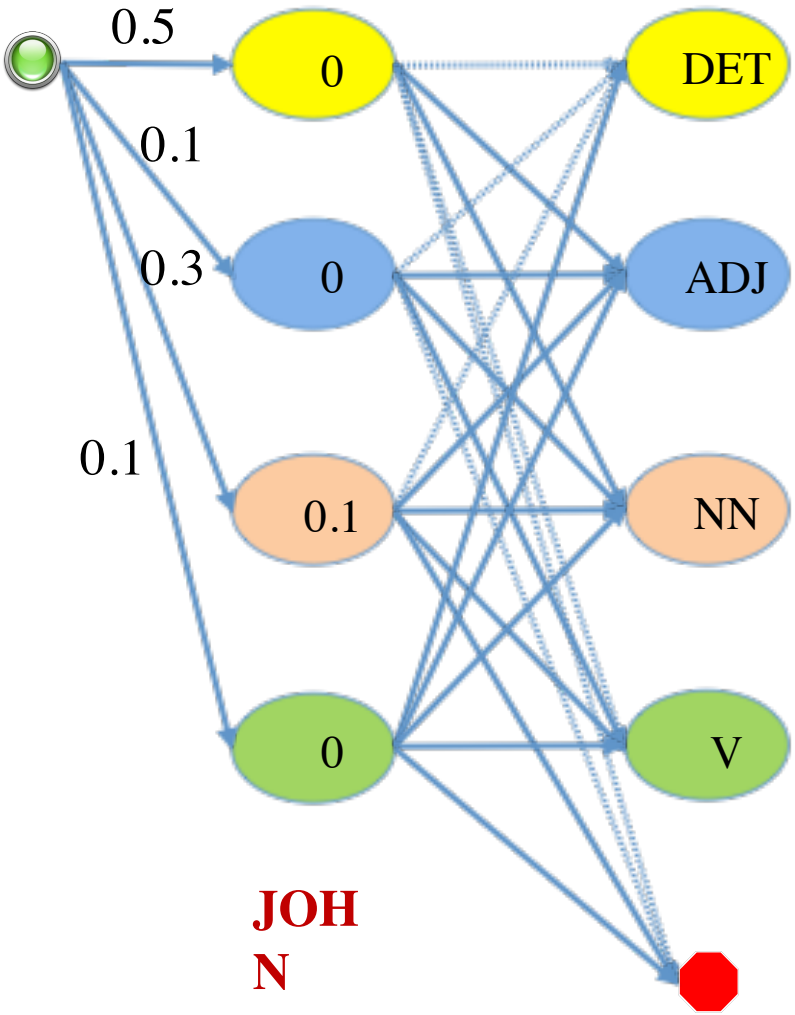
Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



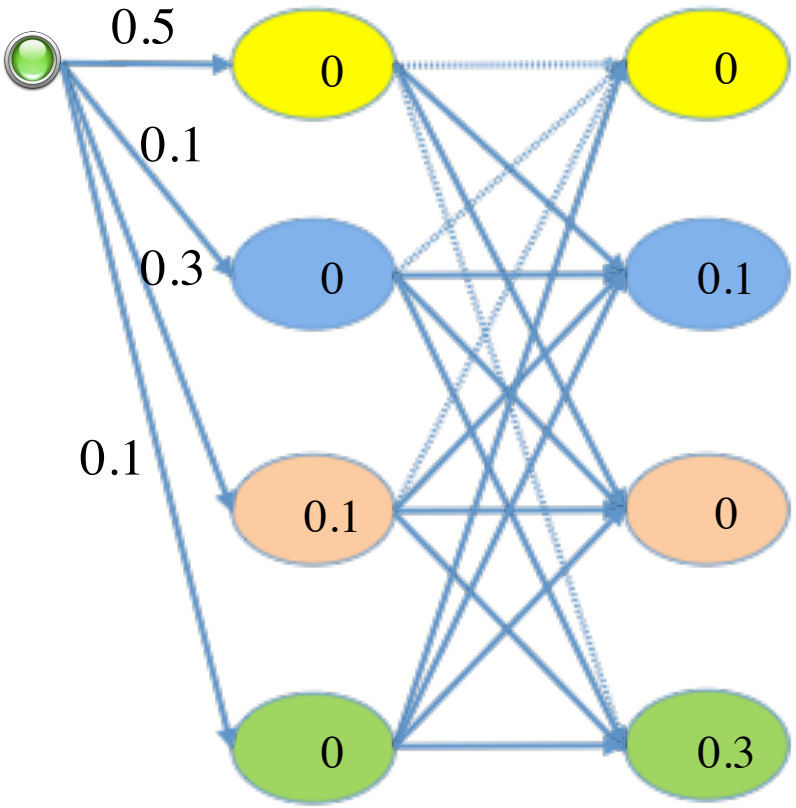
Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

JOHN MIGHT
WATCH



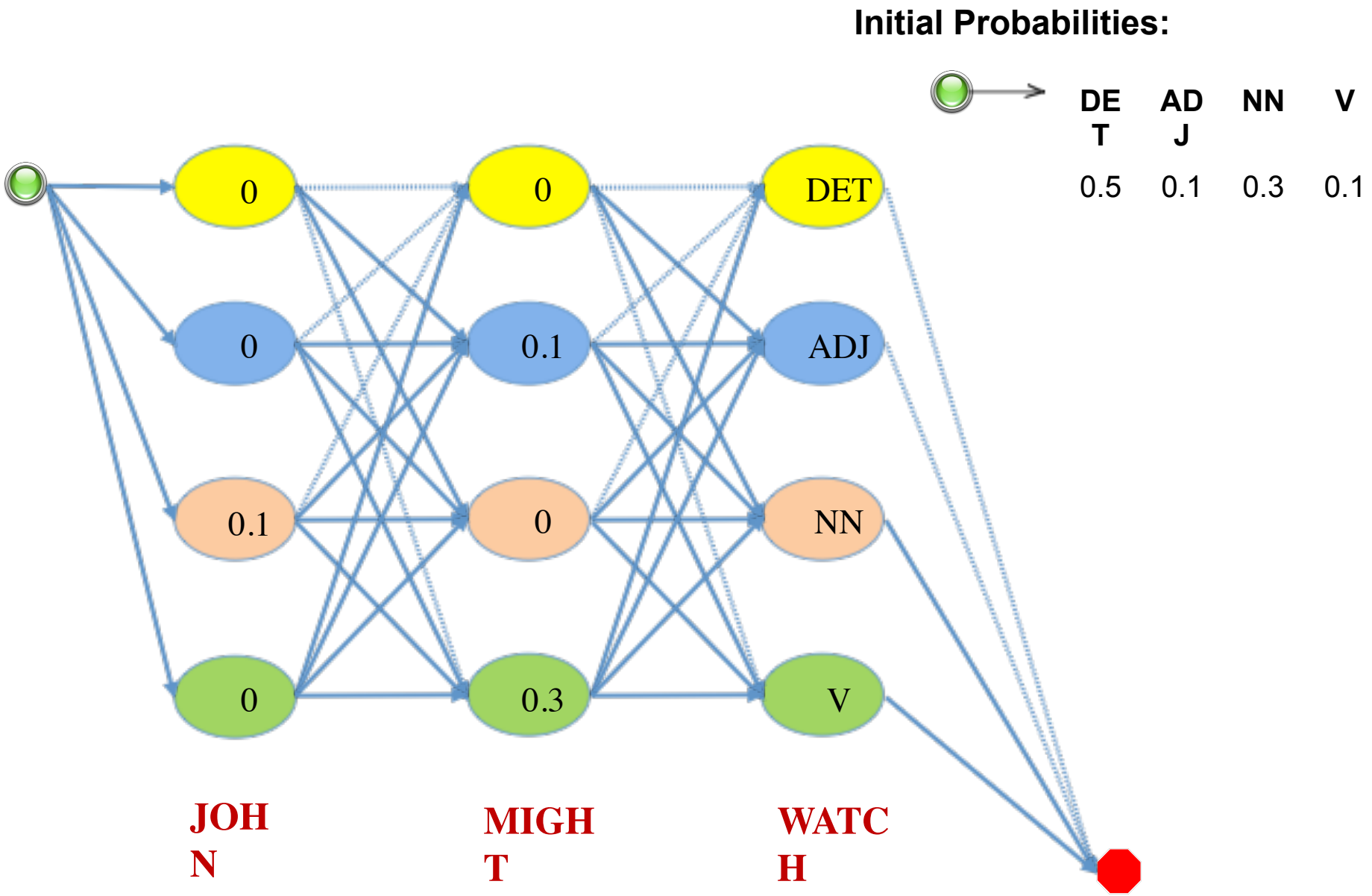
JOH
N

MIGH
T

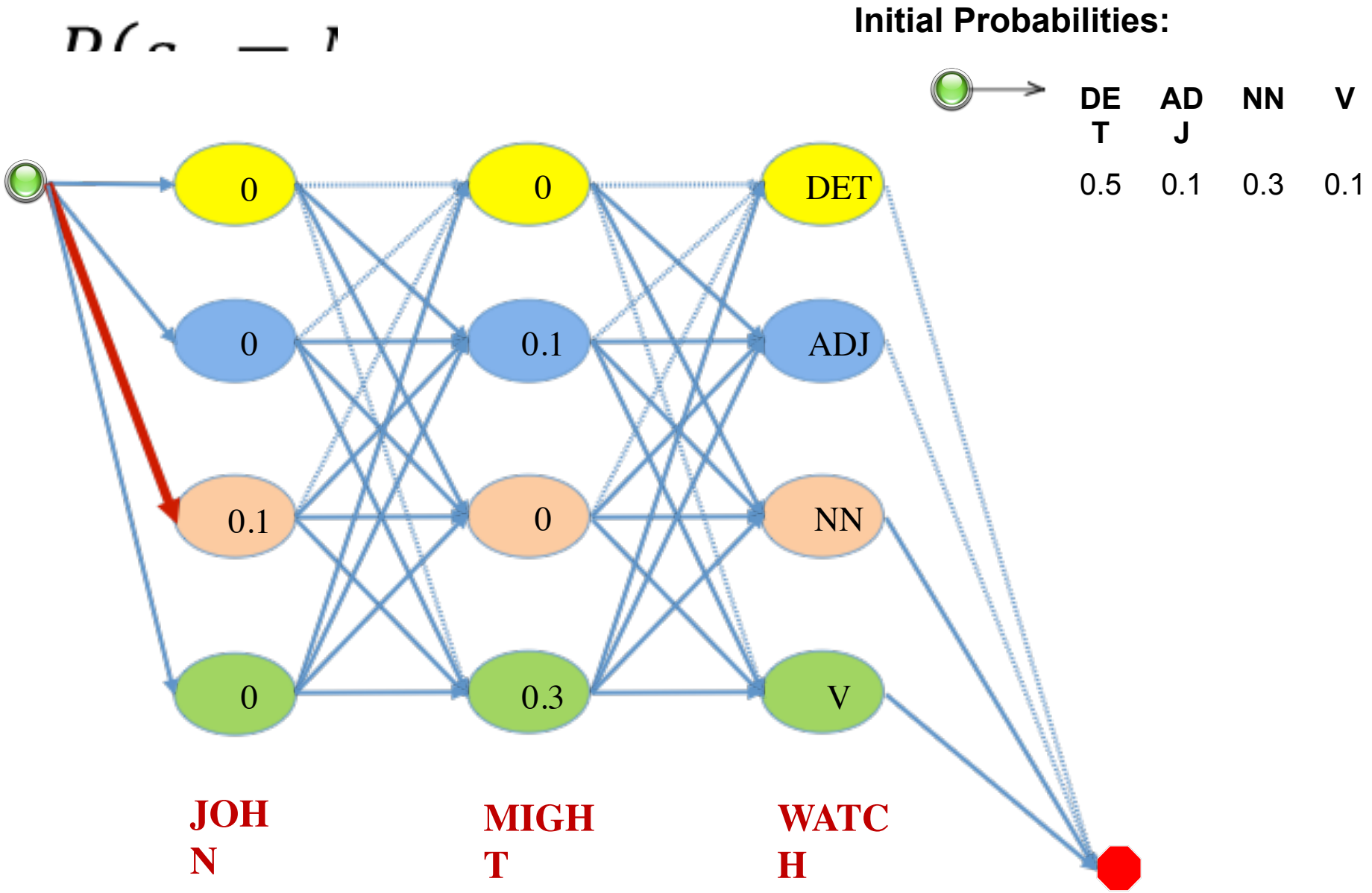
ADJ	
green	0.1
big	0.4
old	0.4
might	0.1

V	
might	0.2
watch	0.3
watches	0.2
loves	0.1
reads	0.19
books	0.01

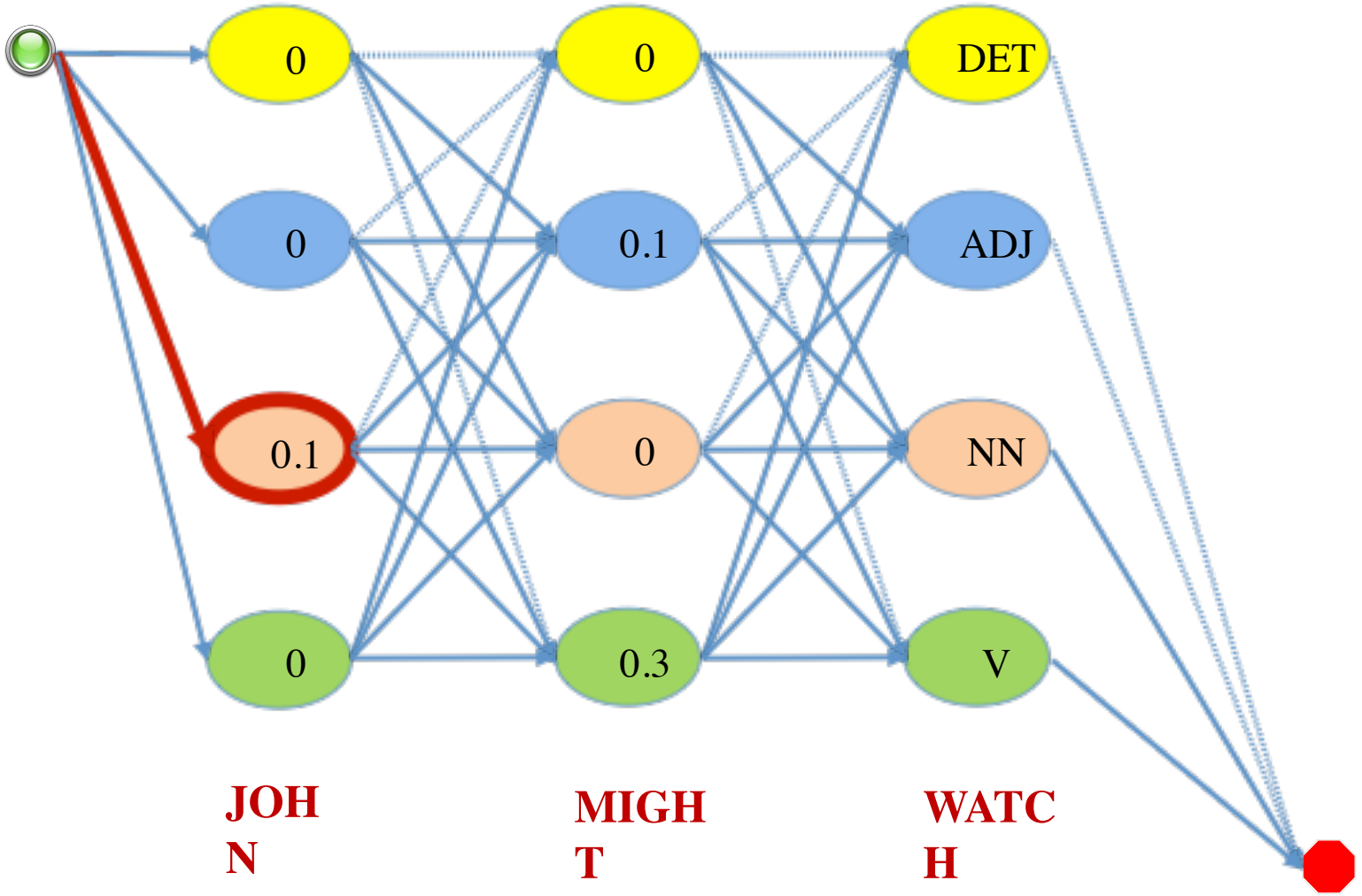
This is the “trellis” that shows all possible ways of generating the word sequence



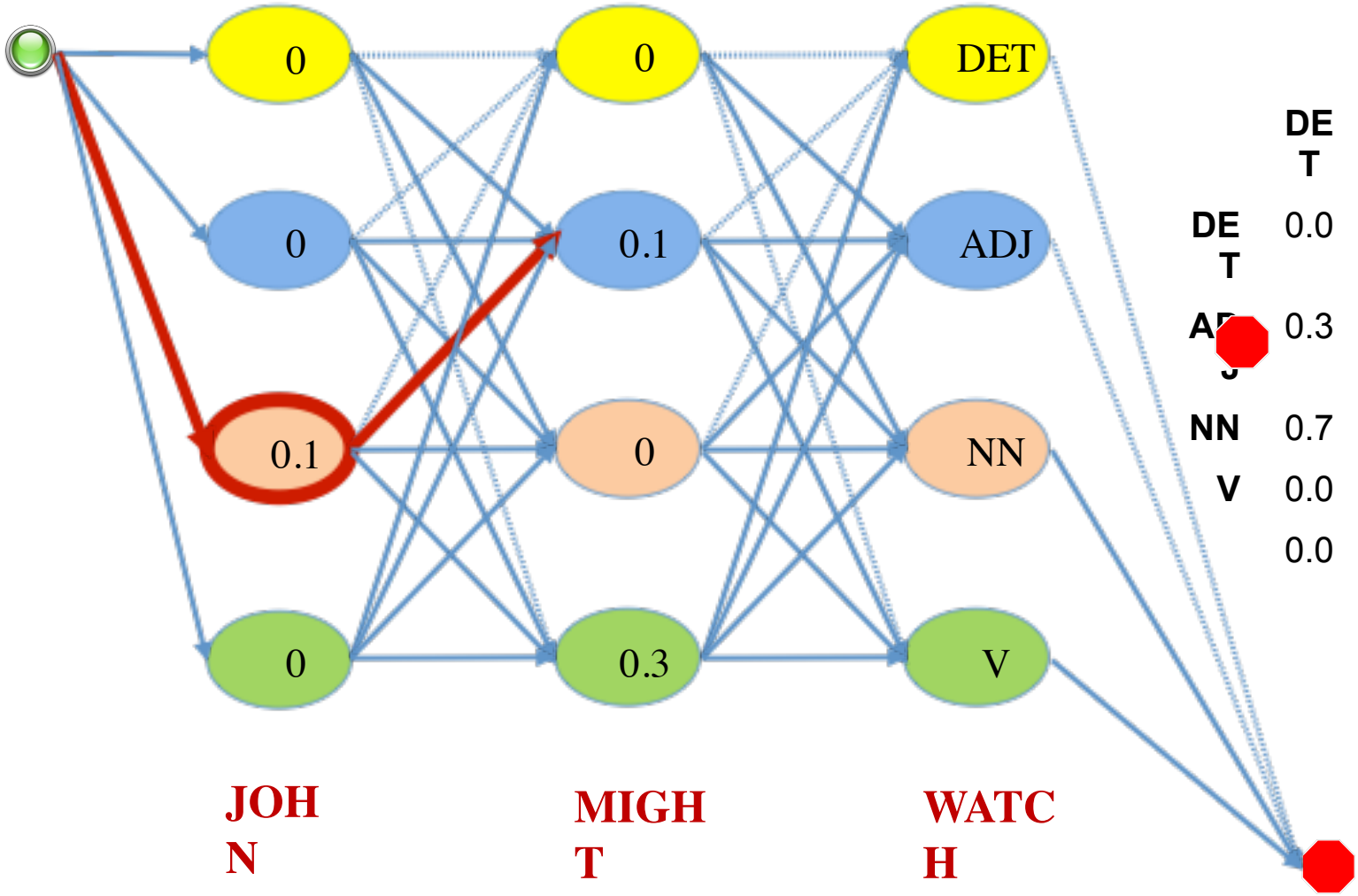
This is the “trellis” that shows all possible ways of generating the word sequence



This is the “trellis” that shows all possible ways of generating the word sequence

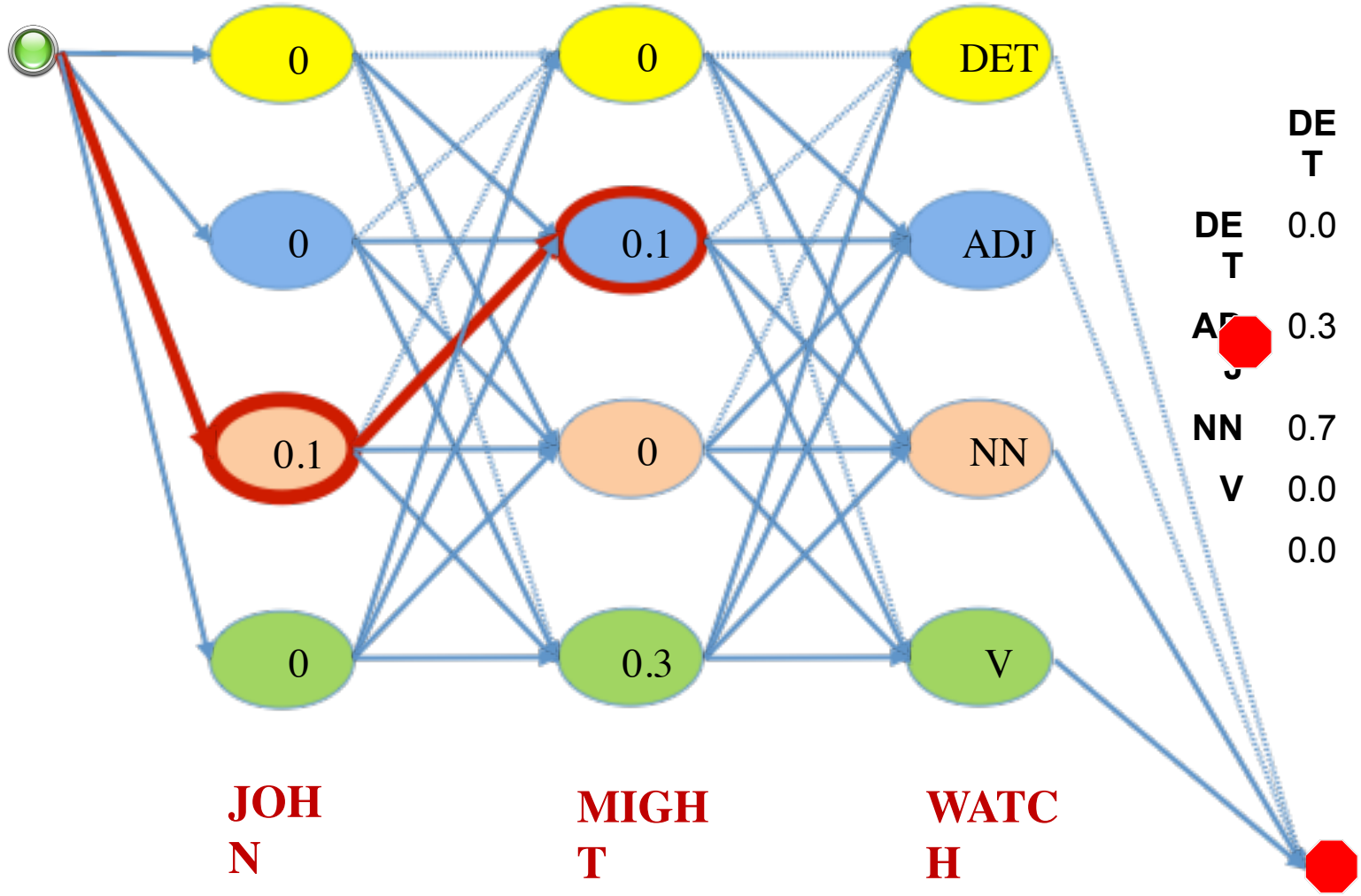


This is the “trellis” that shows all possible ways of generating the word sequence



	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

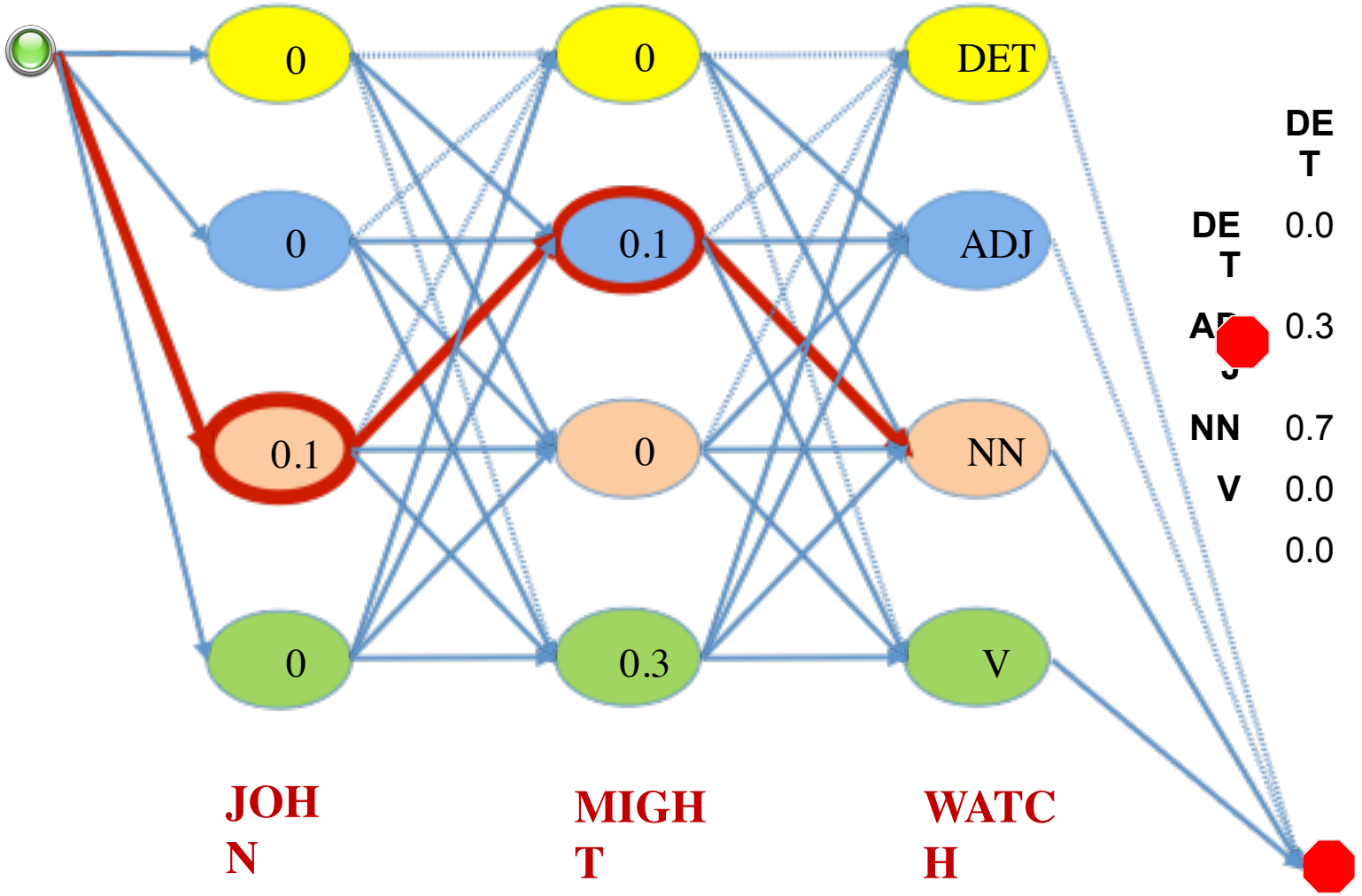
This is the “trellis” that shows all possible ways of generating the word sequence



	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

This is the "trellis" that shows all possible ways of generating the word sequence

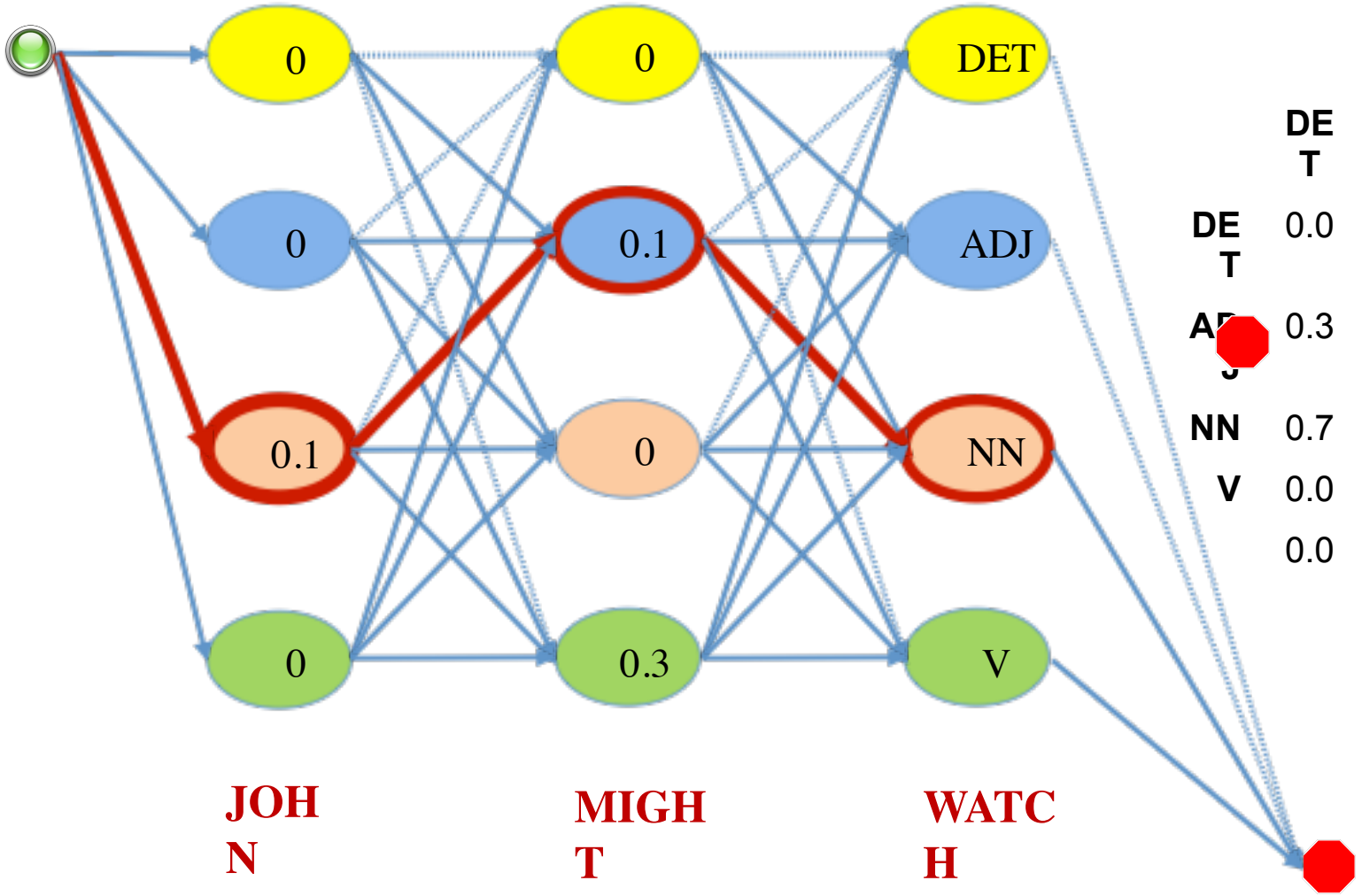
$$P(s_1 = NN, x_1 = JOHN, s_2 = ADJ, :$$



	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

This is the "trellis" that shows all possible ways of generating the word sequence

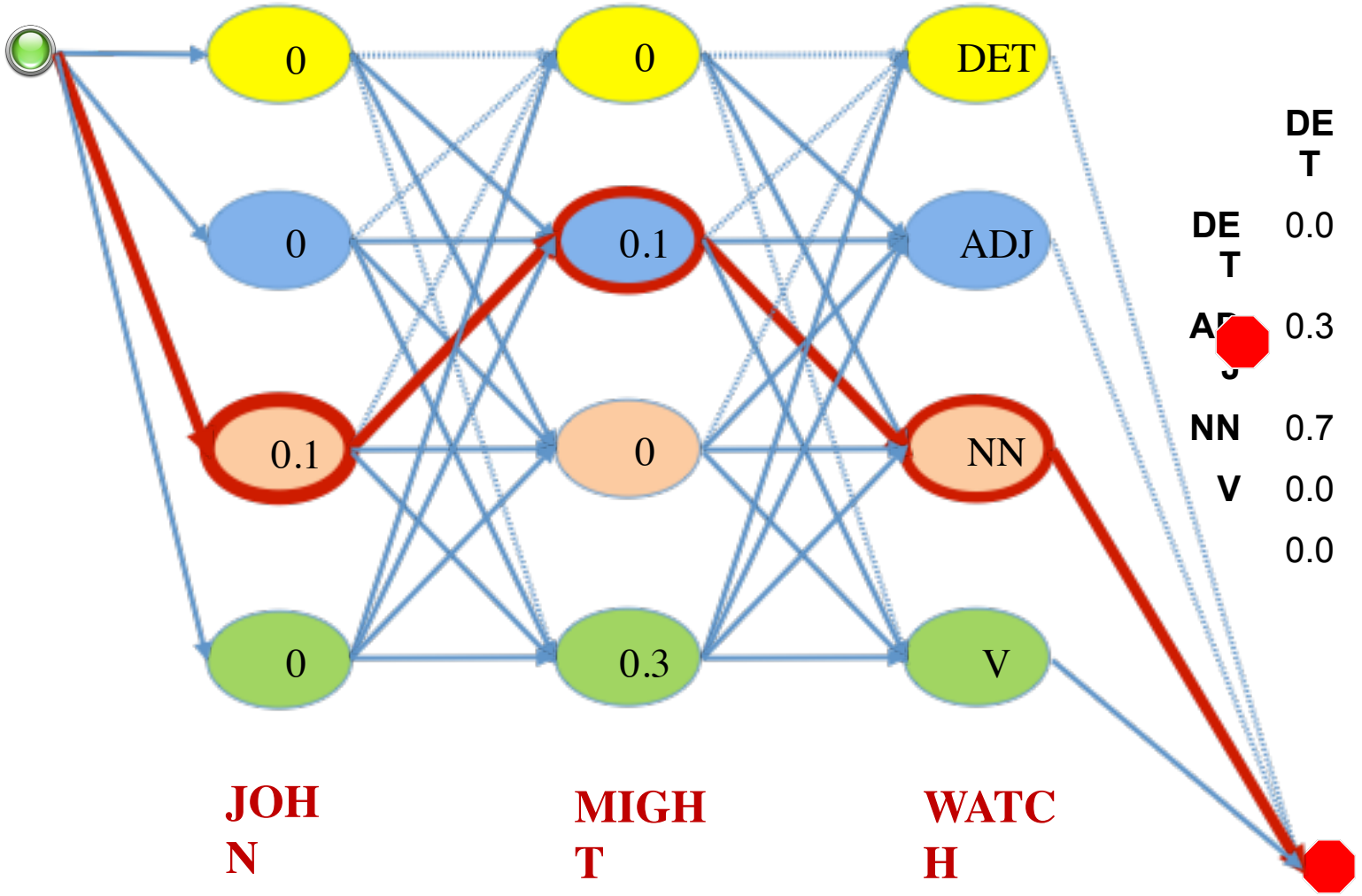
$$P(s_1 = NN, x_1 = JOHN, s_2 = ADJ, x$$



	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

This is the "trellis" that shows all possible ways of generating the word sequence

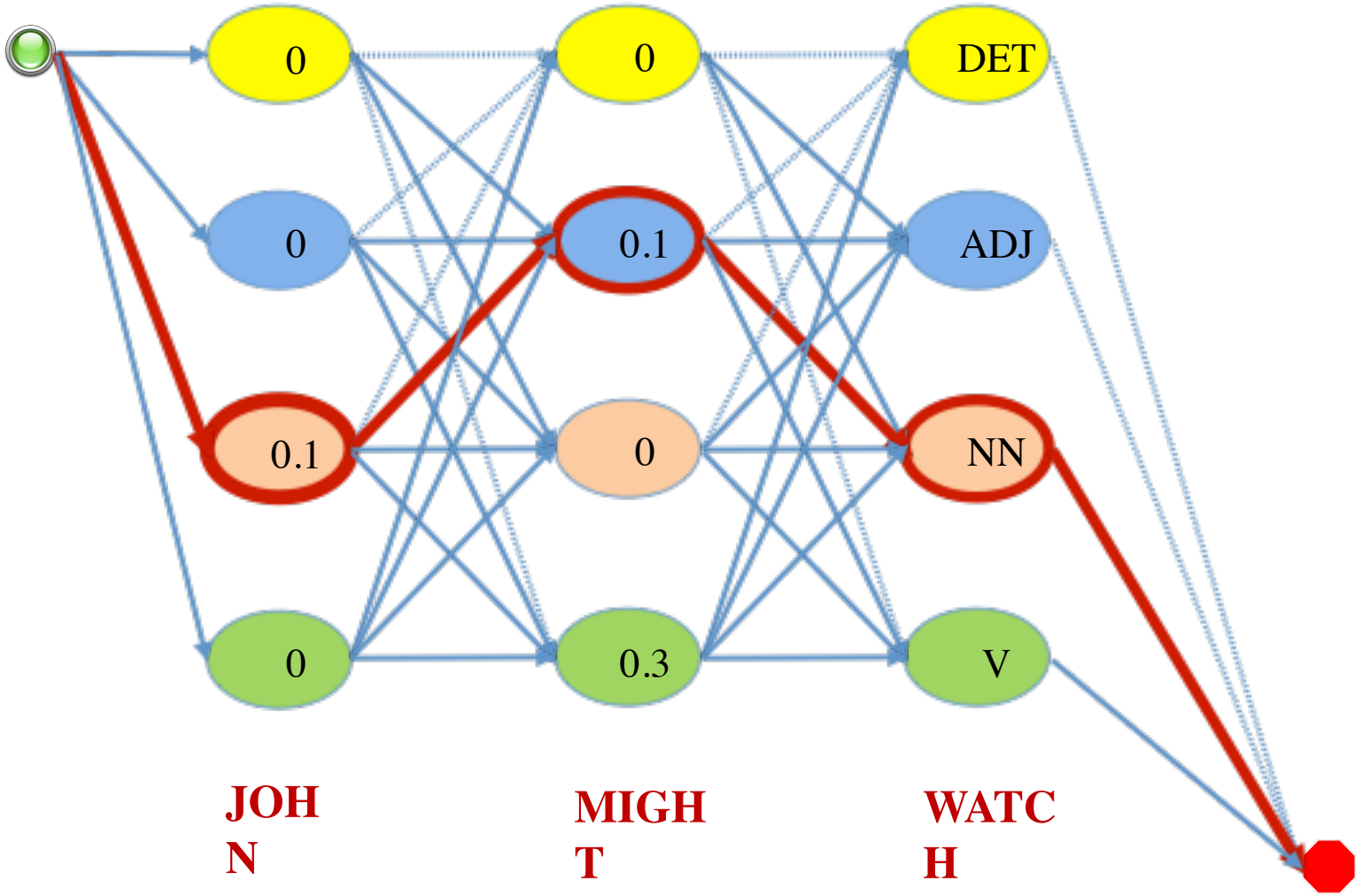
$$P(s_1 = NN, x_1 = JOHN, s_2 = ADJ, x_2 = I$$



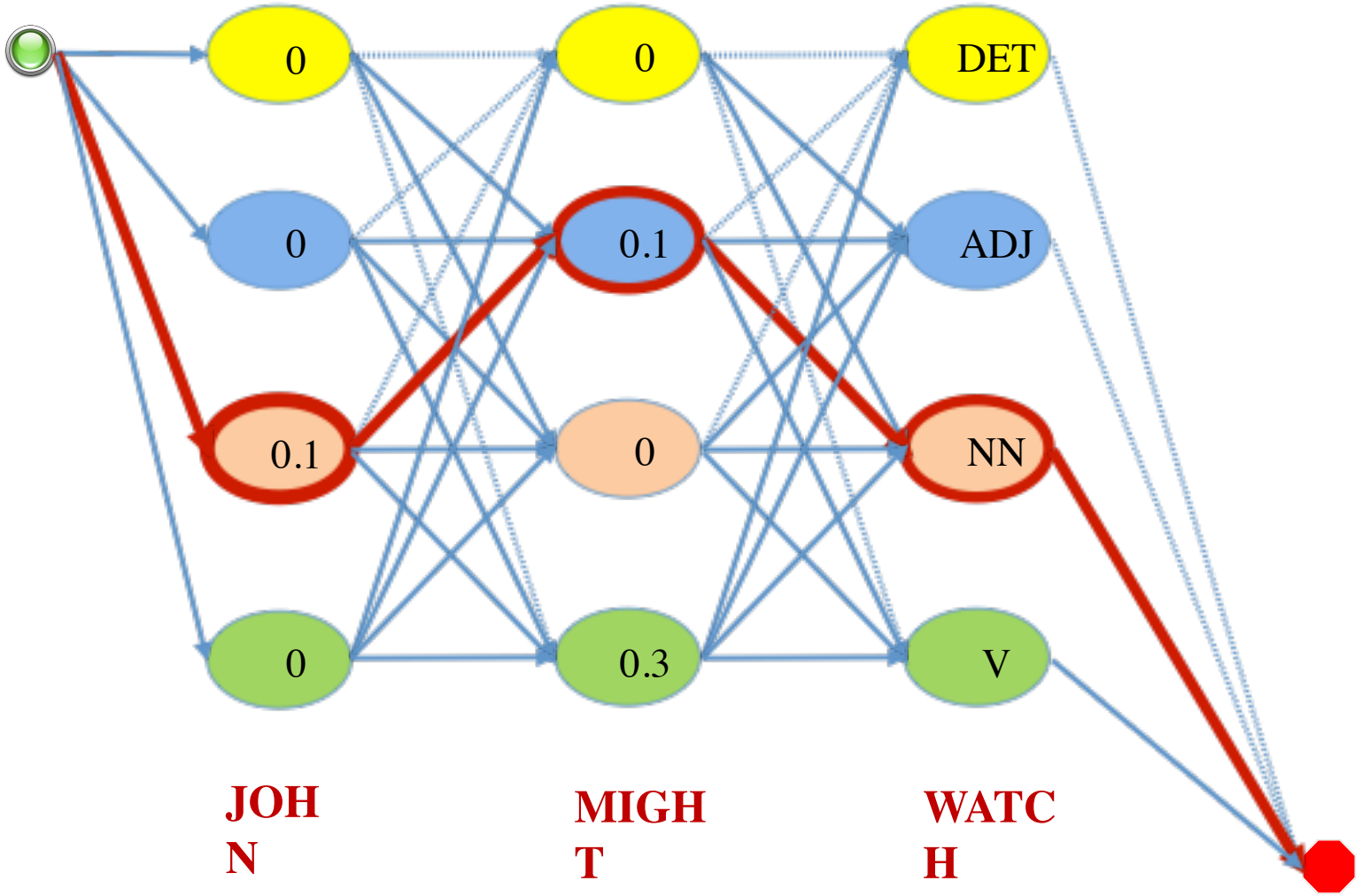
	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.7	0.3	0.2
V	0.0	0.1	0.4	0.1
	0.0	0.0	0.2	0.1

This is the "trellis" that shows all possible ways of generating the word sequence

$$P(s_1 = NN, x_1 = JOHN, s_2 = ADJ, x_2 = I$$

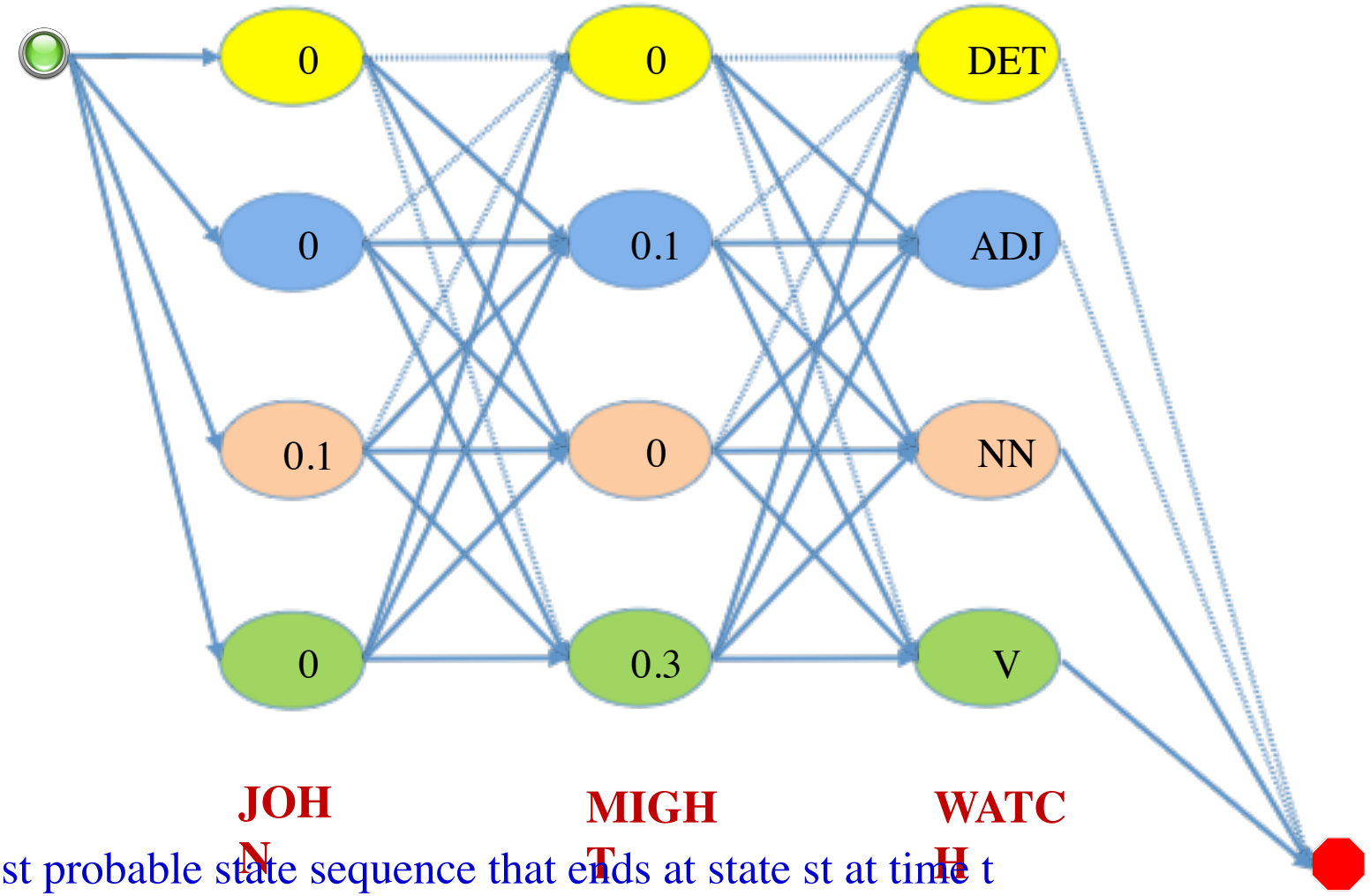


Viterbi : Find the best path (most probable)



Viterbi : Find the best path (most probable)

DP Argument: For a Markov process, the best N-length path to any state *must* be an extension of a best N-1 length path to some state



The most probable state sequence that ends at state st at time t

$$max_{s_1 s_2, \ldots, s_{t-1}} P(s_1, x_1, \ldots, s_t, x_t) =$$

The Viterbi Algorithm

Probability of the most likely state seq

$$max_{s_1 s_2, \ldots, s_{t-1}}$$

Probability of the most likely state seq

- The probabilities are decomposed in a manner suited to DP

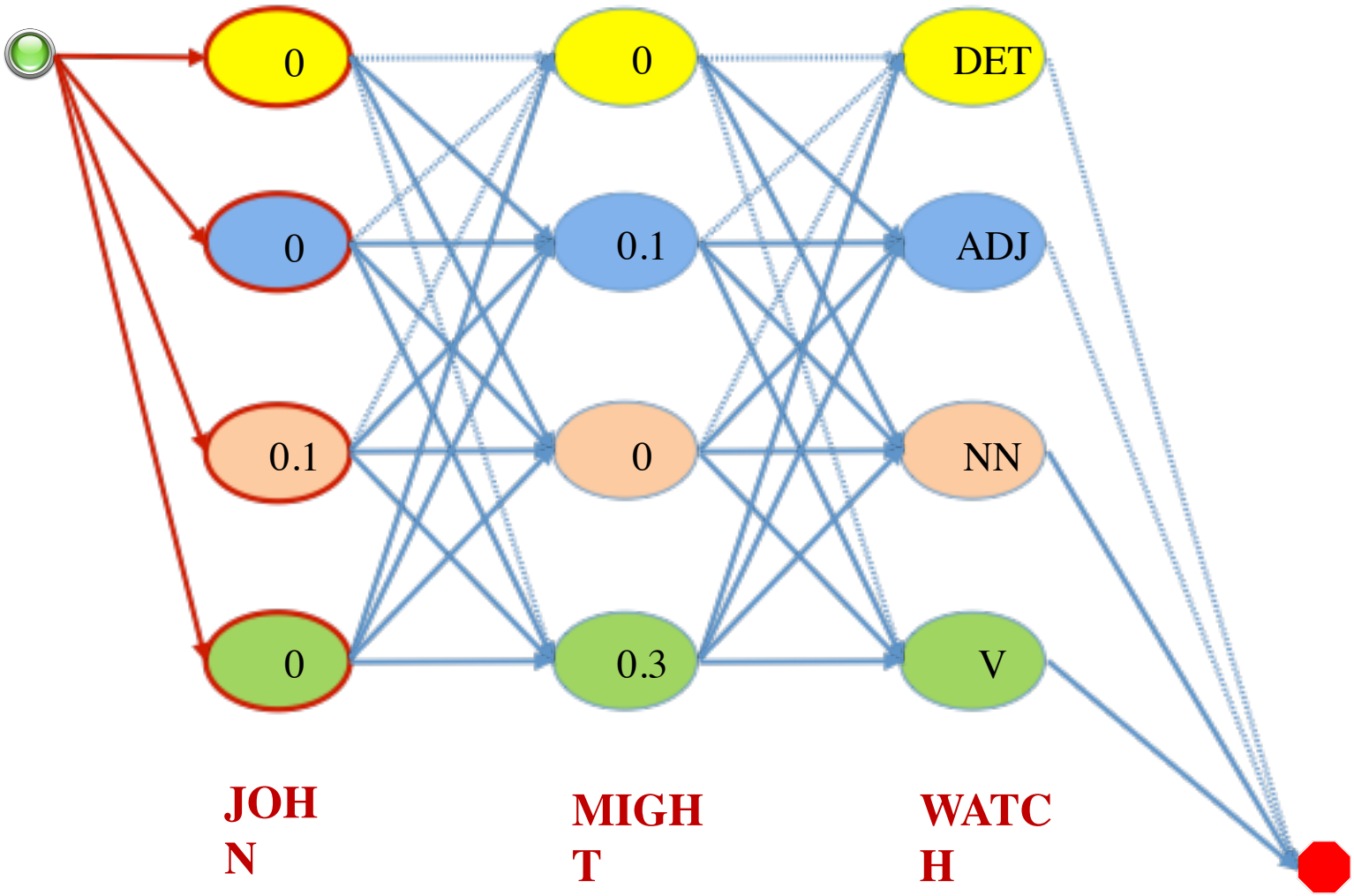
Viterbi

$max_{s_1 s_2,\ldots,s_{t-1}}$

- Let $max_{s_1 s_2,\ldots,s_{t-1}} P(s$
- for $t = 1{:}T$

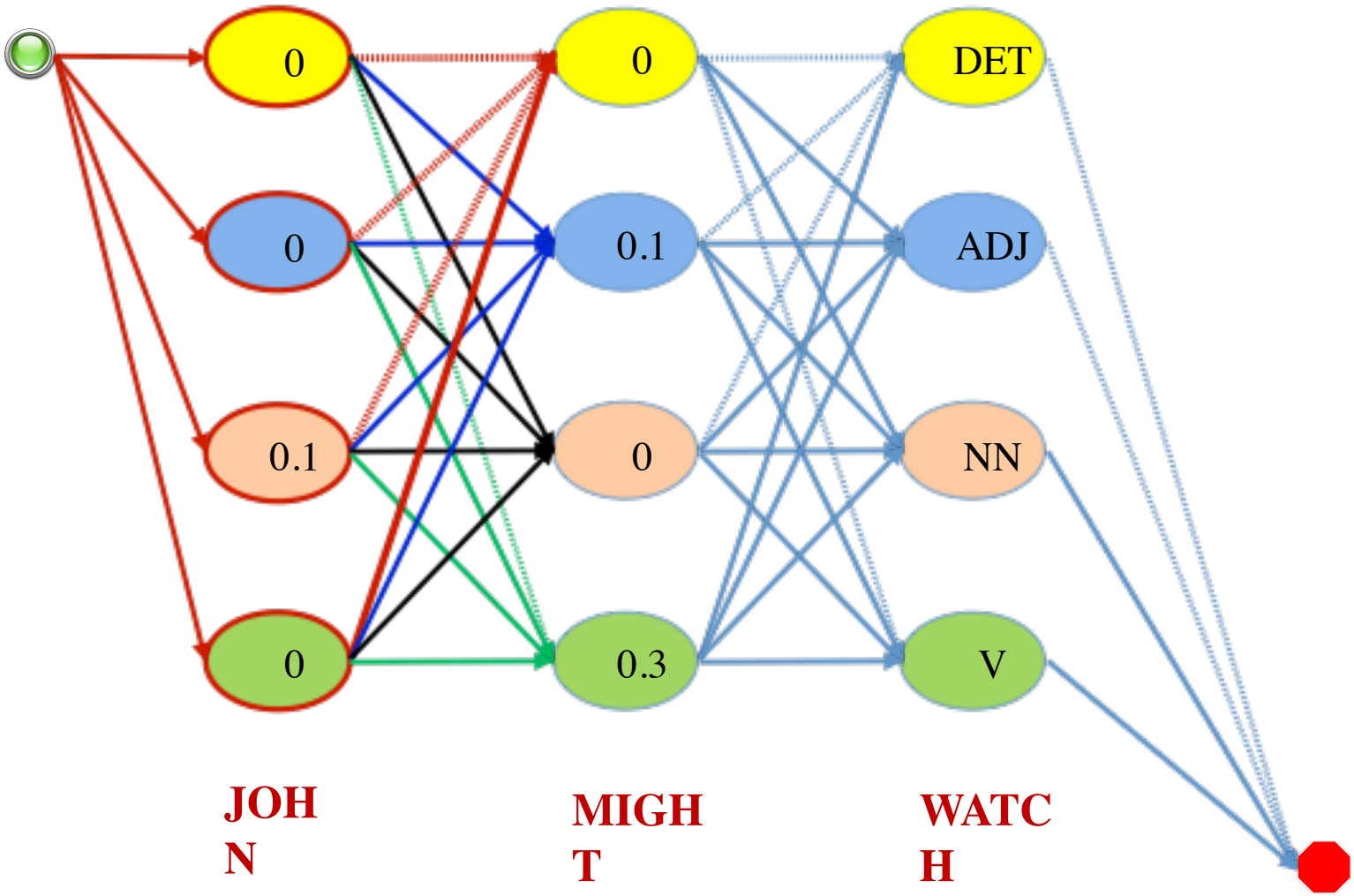
$h(s, t) = \text{argmax}$

Viterbi Algorithm



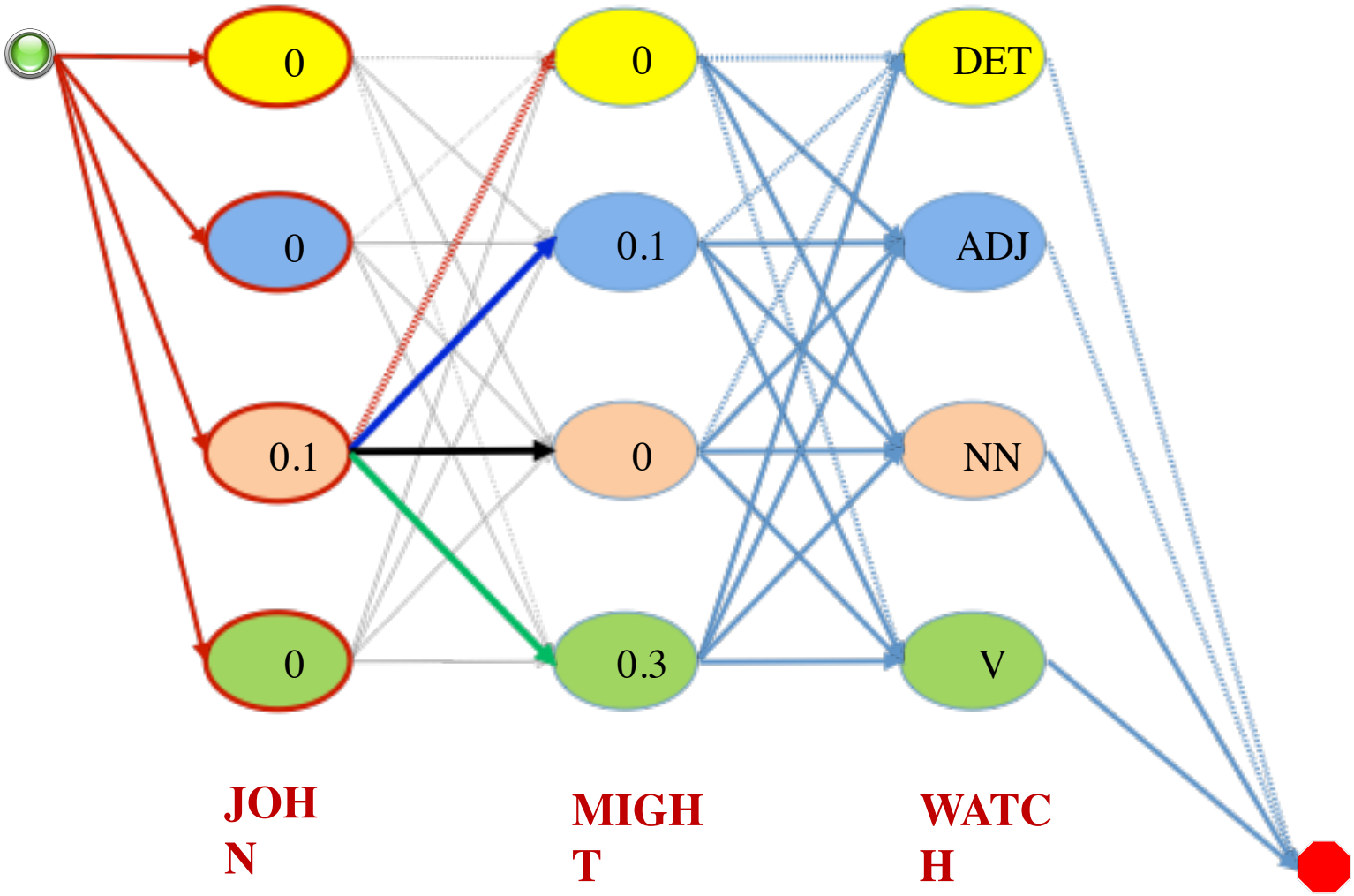
Viterbi Algorithm

$$h(s,t) = \operatorname{argmax} F$$



Viterbi Algorithm

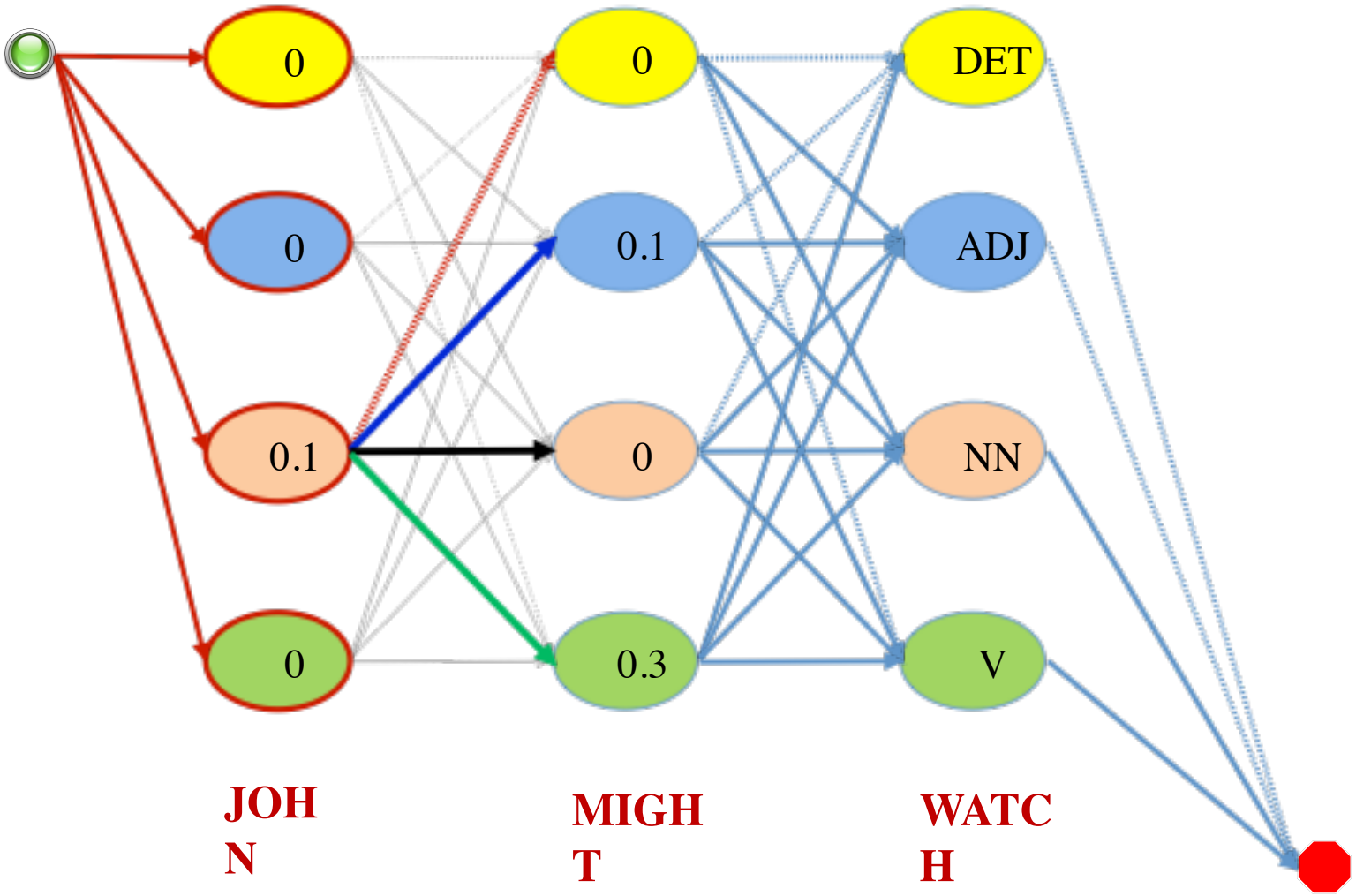
$$h(s,t) = \text{argmax}\, F$$



Viterbi Algorithm

$$h(s,t) = \operatorname{argmax} P$$

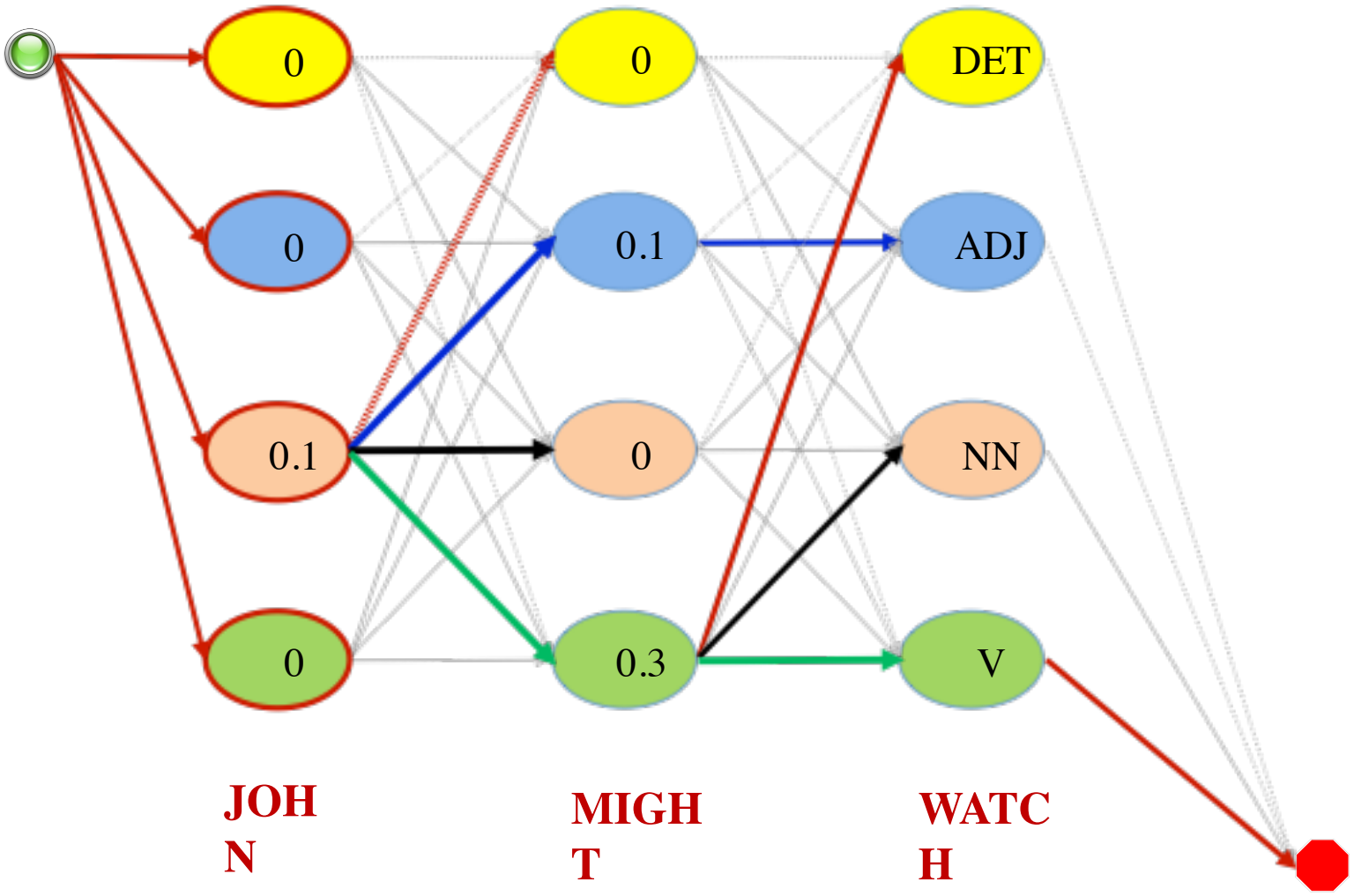
$$P(s,t) = P(h(s,t)) \cdot$$



Viterbi Algorithm

$$h(s,t) = \text{argmax}\, P$$

$$P(s,t) = P(h(s,t)) +$$



String Marginals

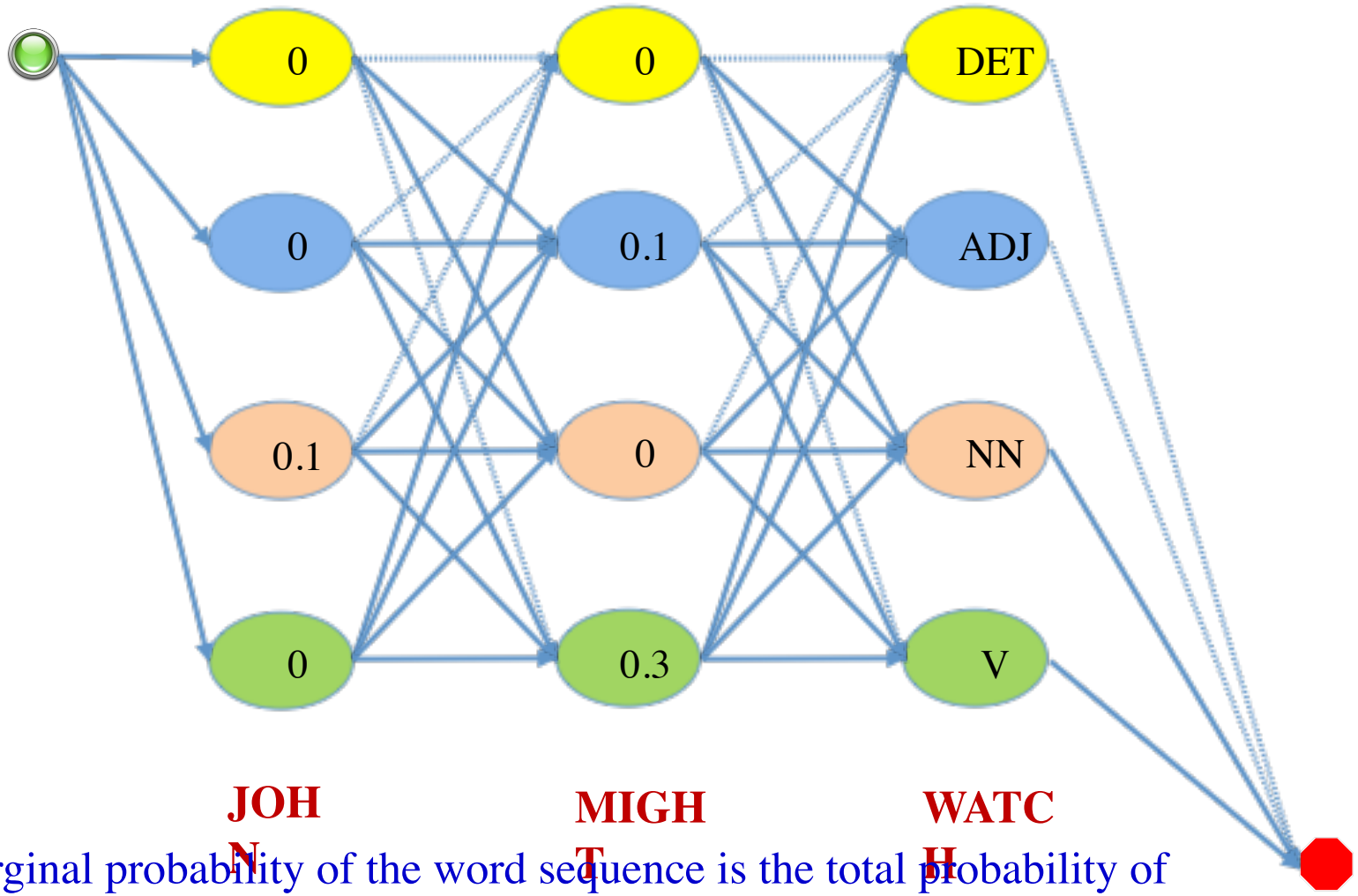
- Inference question for HMMs
  - What is the probability of a string **w**?
    Answer: generate all possible tag sequences and explicitly *marginalize*

$$O(|\Omega|^{|\mathbf{w}|}) \quad \text{time}$$

Can we do this efficiently?

Viterbi : Find the best path (most probable)

String Marginals



The marginal probability of the word sequence is the total probability of all paths through the trellis

$$P(x_1, \ldots, x_t) = \sum$$

Forward Algorithm

- How to compute: use the **forward algorithm**
  - Analogous to Viterbi
    - Instead of computing a **max** of inputs at each node, use **addition**
  - Same run-time, same space requirements

$$O(|\Omega|^2 \times |\mathbf{w}|) \quad \text{time}$$

$$O(|\Omega|) \quad \text{space}$$

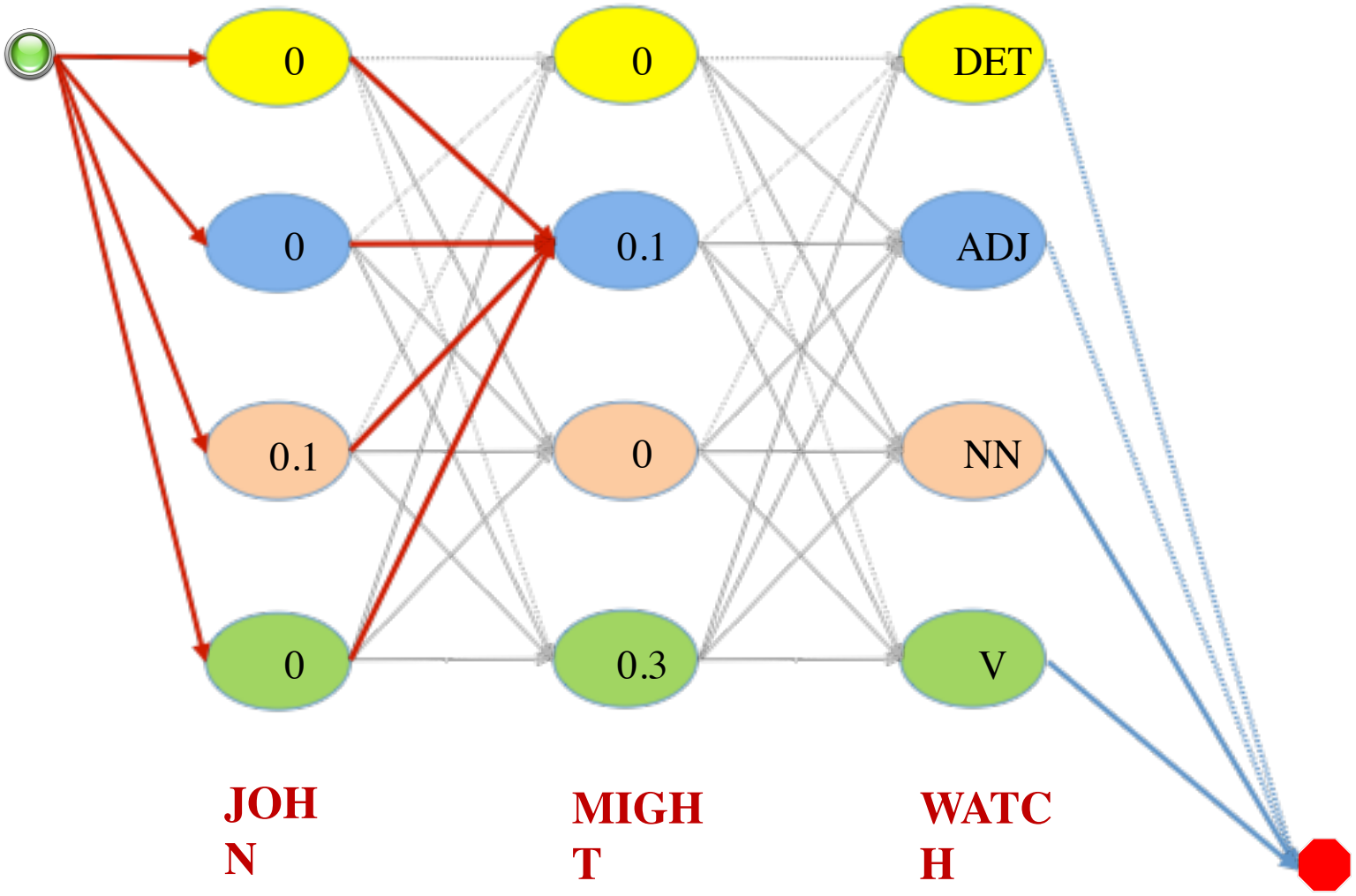
Define

- $\alpha_t(s) = P($

- The probability of ge
  the process is in stat

Viterbi : Find the best path (most probable)

String Marginals



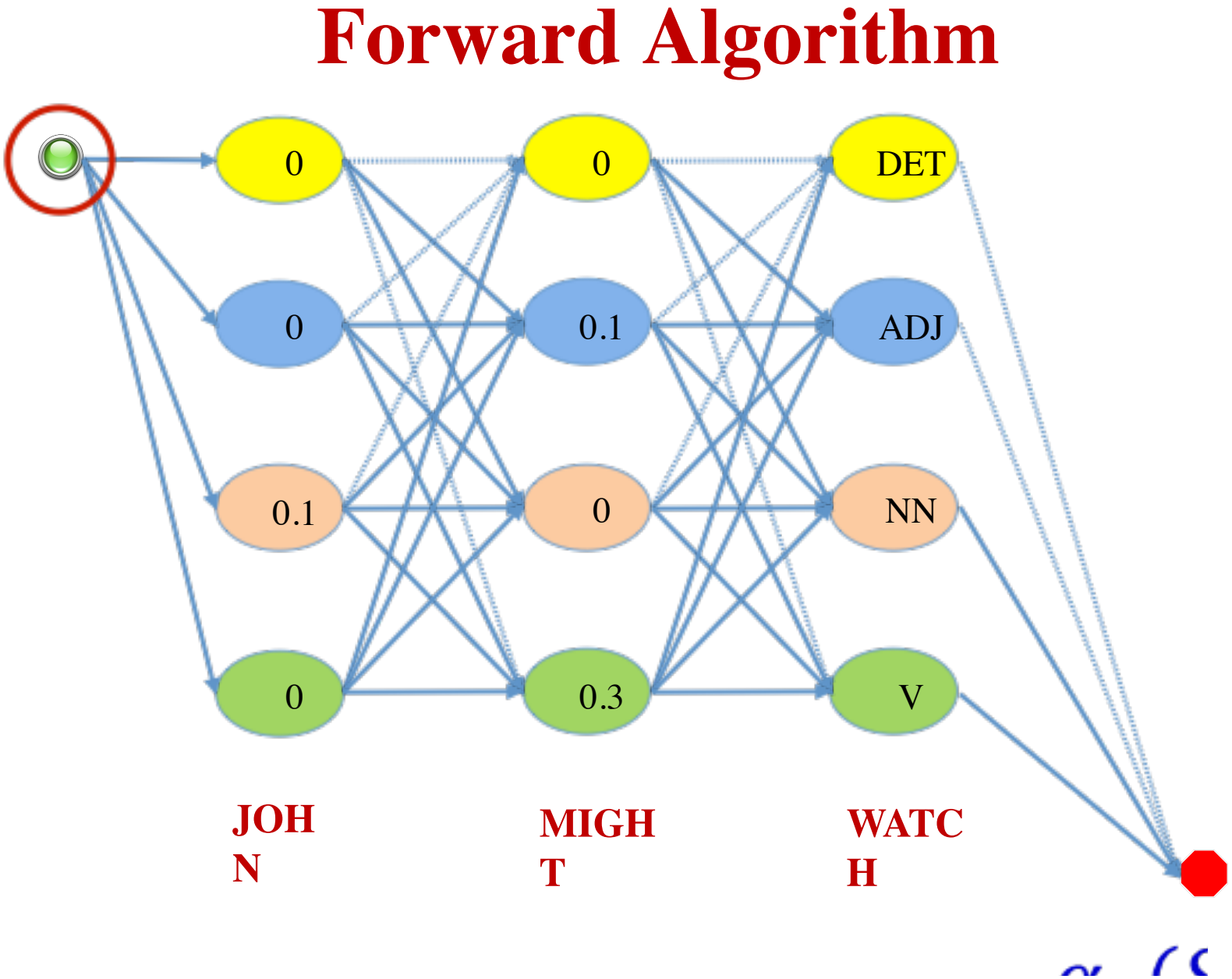
$\alpha_2(ADJ)$ is the probability of producing JOH

Forward Algorithm Recurrence

$$\alpha_0(\text{START}) = 1$$

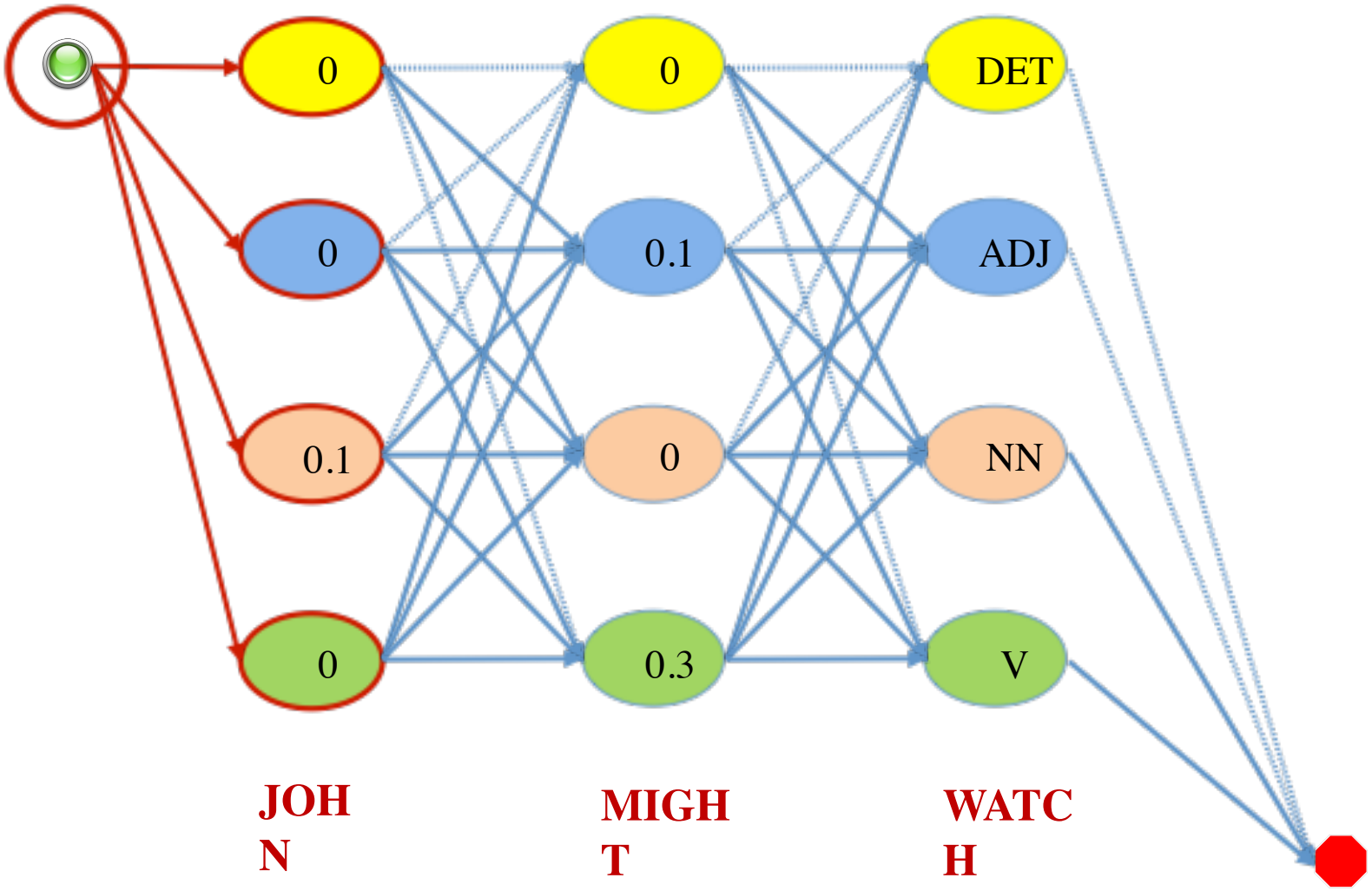
$$\alpha_t(r) = \sum \alpha_{t-1}(q$$

Viterbi : Find the best path (most probable)



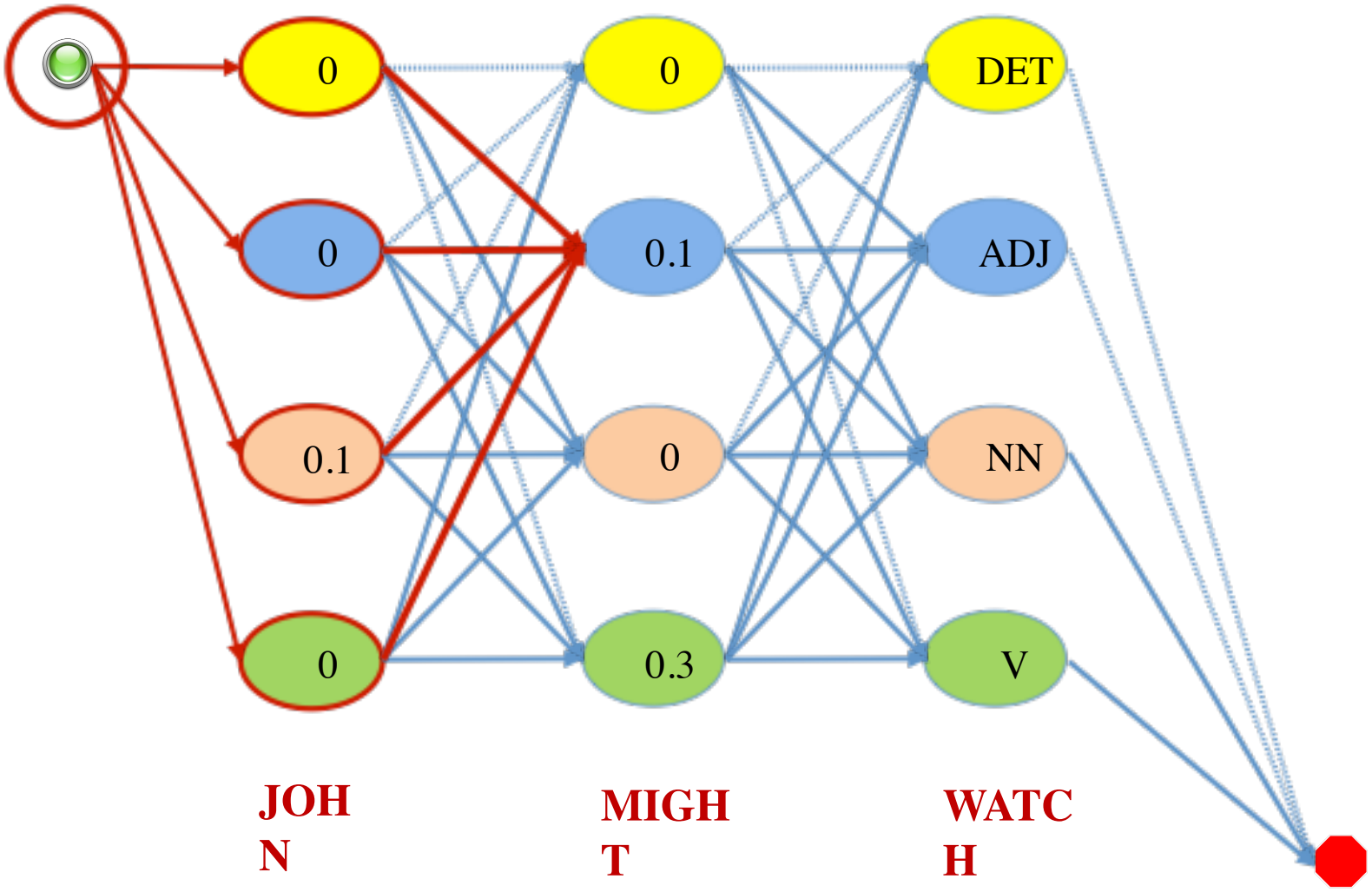
Viterbi : Find the best path (most probable)

Forward Algorithm



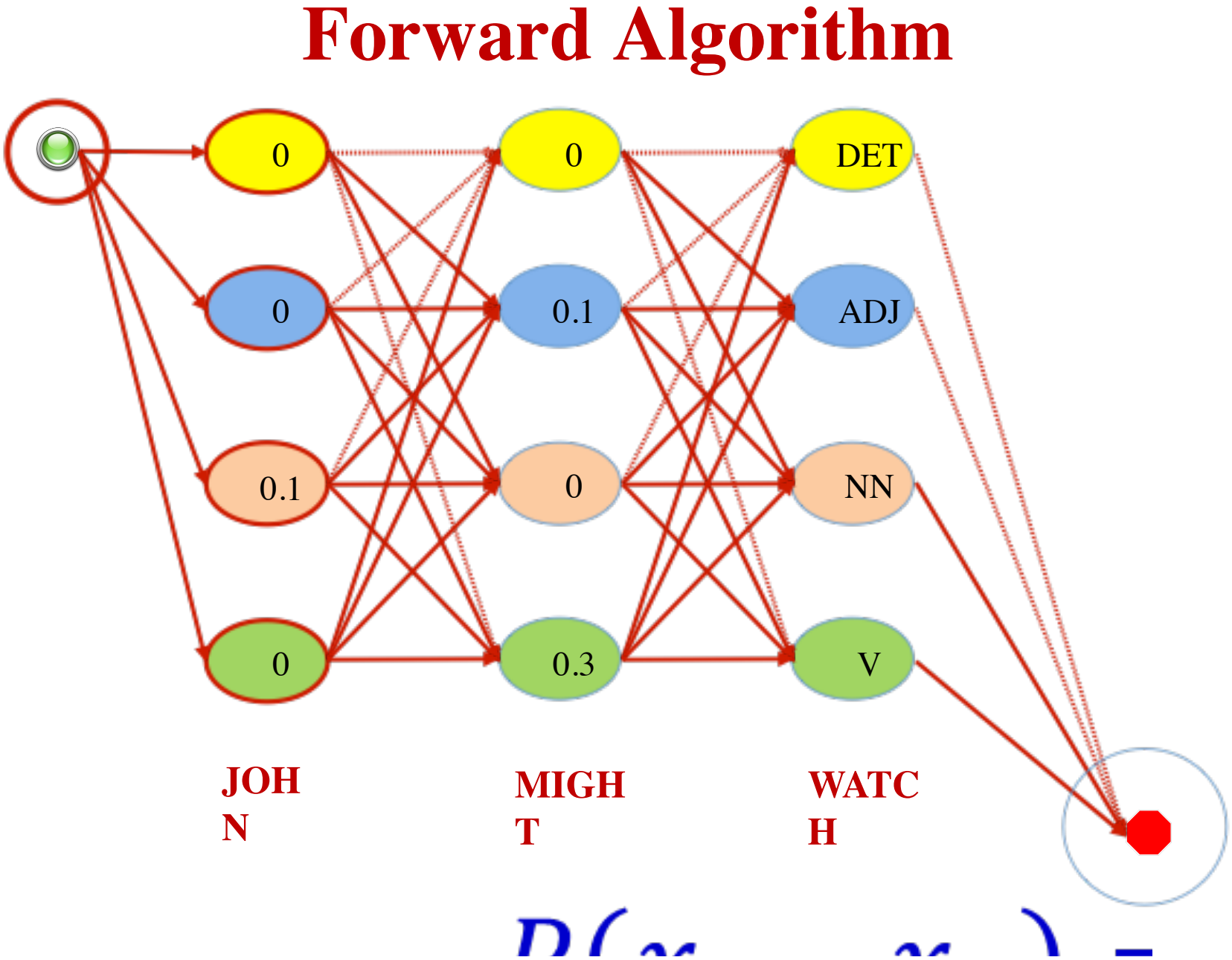
Viterbi : Find the best path (most probable)

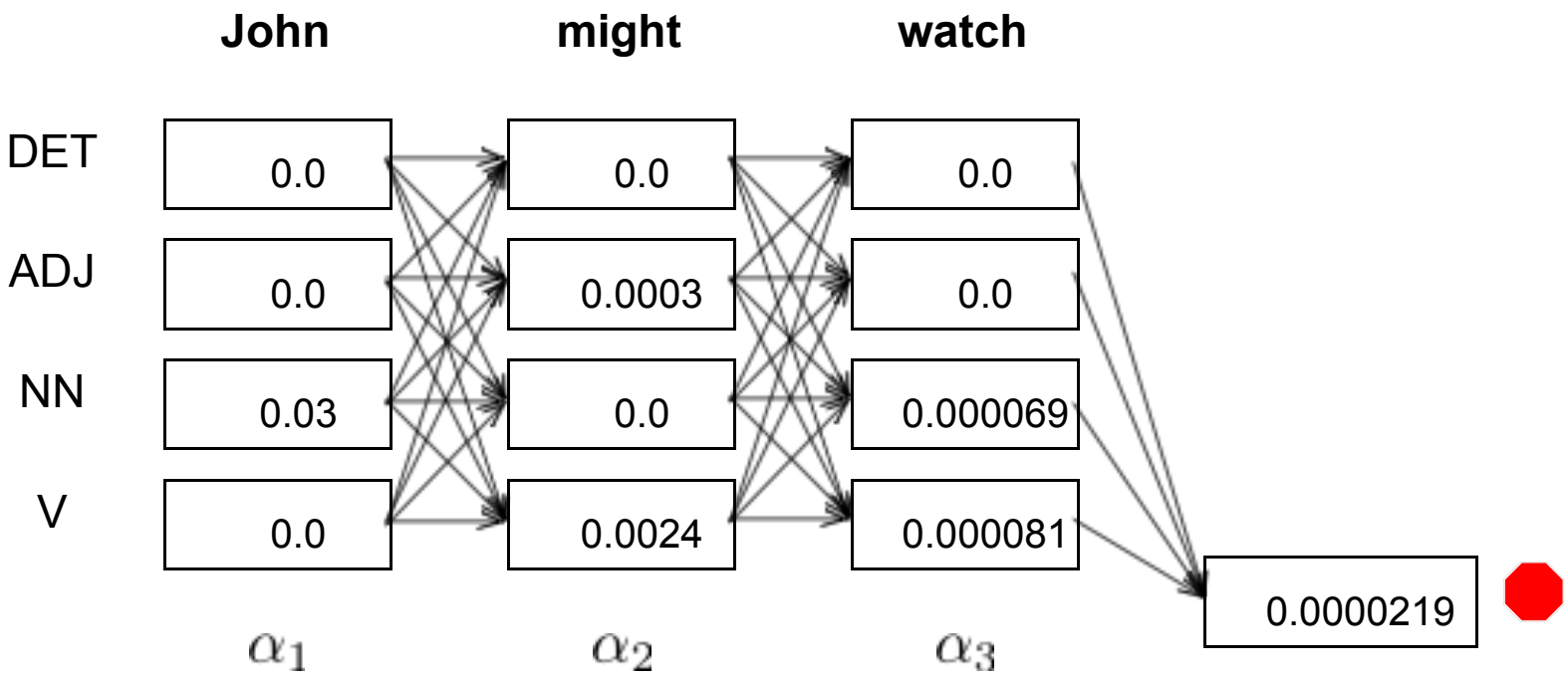
Forward Algorithm



$$\alpha_t(x) = \sum \alpha_{t-1}(\cdot$$

Viterbi : Find the best path (most probable)




John
might
watch
DET
ADJ
NN
V
0.0
0.0
0.0
0.0
0.0
0.0003
0.03
0.0
0.000069
0.0
0.0024
0.000081
0.0000219
α_1
α_2
α_3

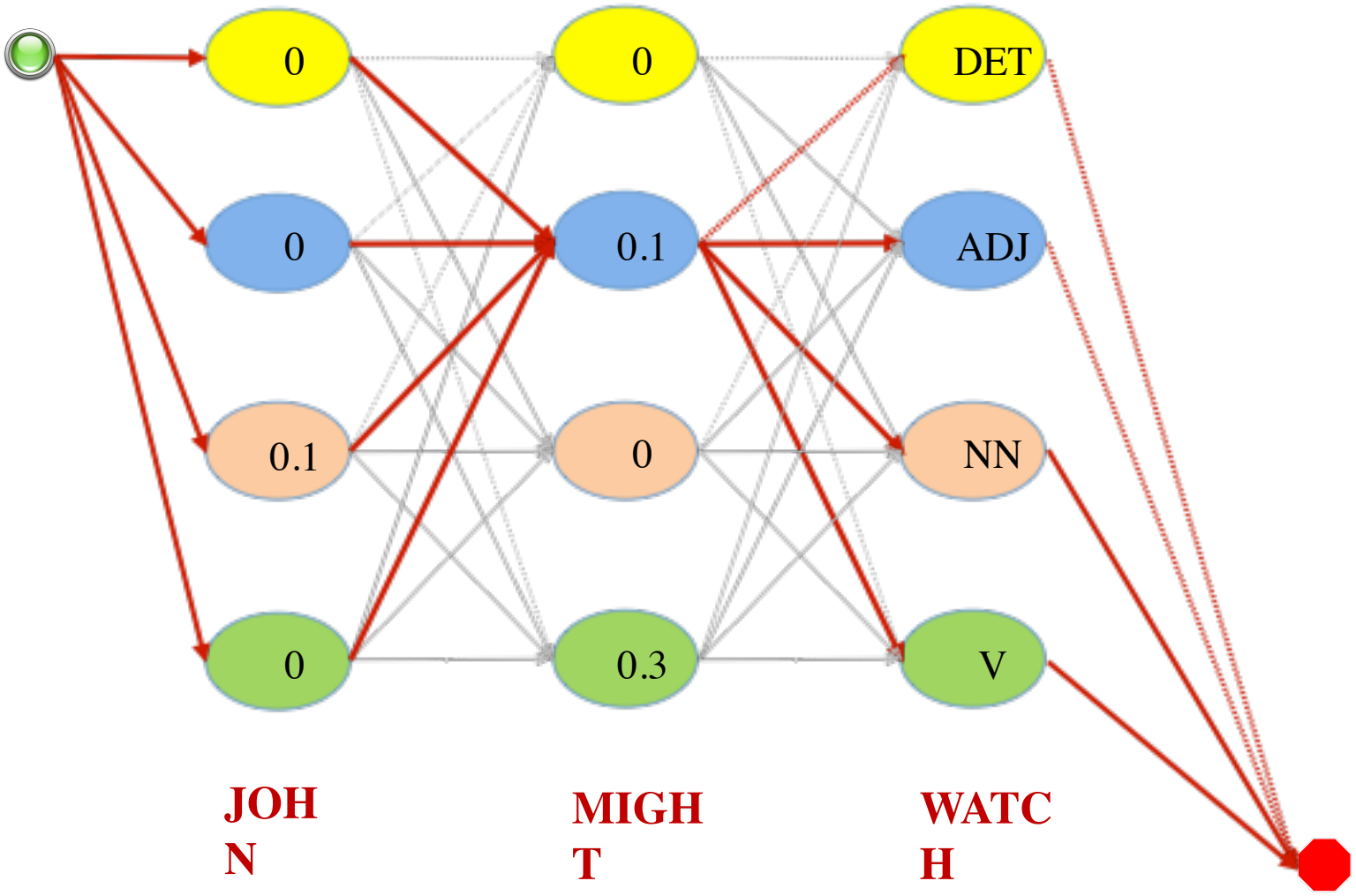
$$p = 0.0000219$$

Posterior Marginals

- Marginal inference question for HMMs
  - *Posterior Marginal*: Given **x**, what is the probability of being in a state *q* at time *t*?
  - *Marginal*: What is the probability of ***x*** *and* being in state *q* at time *t*?

$$p(s_t = q | x_1, \ldots, x_T) \propto p(x$$

Posterior Marginals



The marginal of ADJ at t=2 is the total probability of all paths that go through ADJ at t=2

Posterior Marginals

- Marginal inference question for HMMs
  - State: Given **x**, what is the probability of being in a state *q* at time *t*?

$$p(s_t = q | x_1, \ldots, x_T) \propto p(x$$

  - Transition: Given **x**, what is the probability of transitioning from state *q* to *r* at time *t*?

$$p(s_t = q, s_{t+1} = r | x_1, \ldots, x_T)$$

Posterior Marginals

- Marginal inference question
  - State: What is the probabilit

$$p(x_1, \ldots, x_T, s_t = q) = p(x_1, ..$$

  - Transition: What is the proba

Posterior Marginals

- Marginal inference question
  - State: What is the probabilit

$$p(x_1, \ldots, x_T, s_t = q) = p(x_1, \ldots$$

  - Transition: What is the proba

Posterior Marginals

- Marginal inference question
  - State: What is the probabilit

$$p(x_1, \ldots, x_T, s_t = q) = p(x_1, ..$$

  - Transition: What is the prob

The Backward Probability

- $$P(x_{t+1}, \dots, x_T | s_t = q) \equiv \sum_s$$

$$P(x_{t+1}, \dots, x_T, s$$

$$\eta(q \to s)\gamma(s \downarrow x_{t+1})$$

Backward Algorithm

$P(x_{\cdot} \ldots x_{-}|s_{\cdot} = q) = \sum n(q \rightarrow$

- Define $\beta_t(q) = P(x_t$
- Recursion

Backward Algorithm

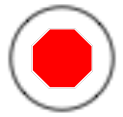
- Start at the goal node(s) and work **backwards** through the trellis

Backward Recurrence

$$\beta_{|\mathbf{x}|+1}(\text{STOP}) = 1$$

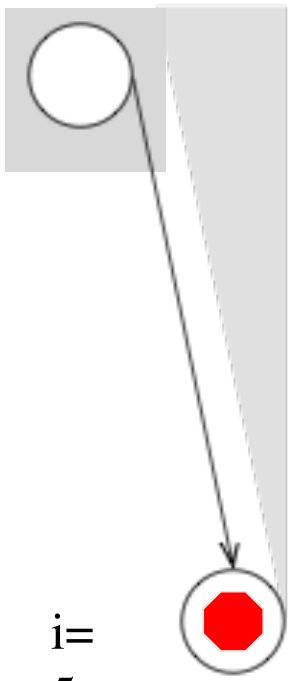
$$\beta_i(q) = \sum_{q'} p(q \rightarrow q'

Backward Chart

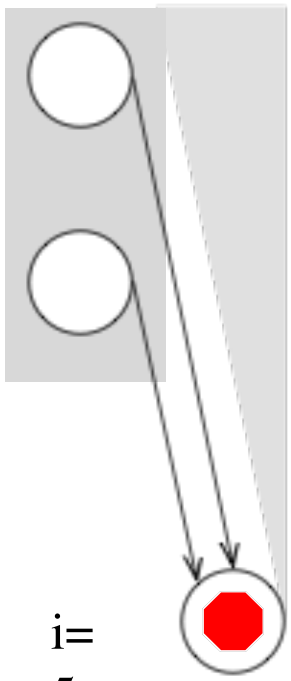


$$\beta_{|\mathbf{x}|+1}(\text{STOP}) = 1$$

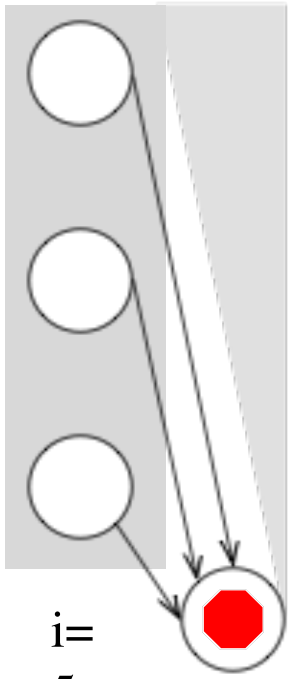
Backward Chart



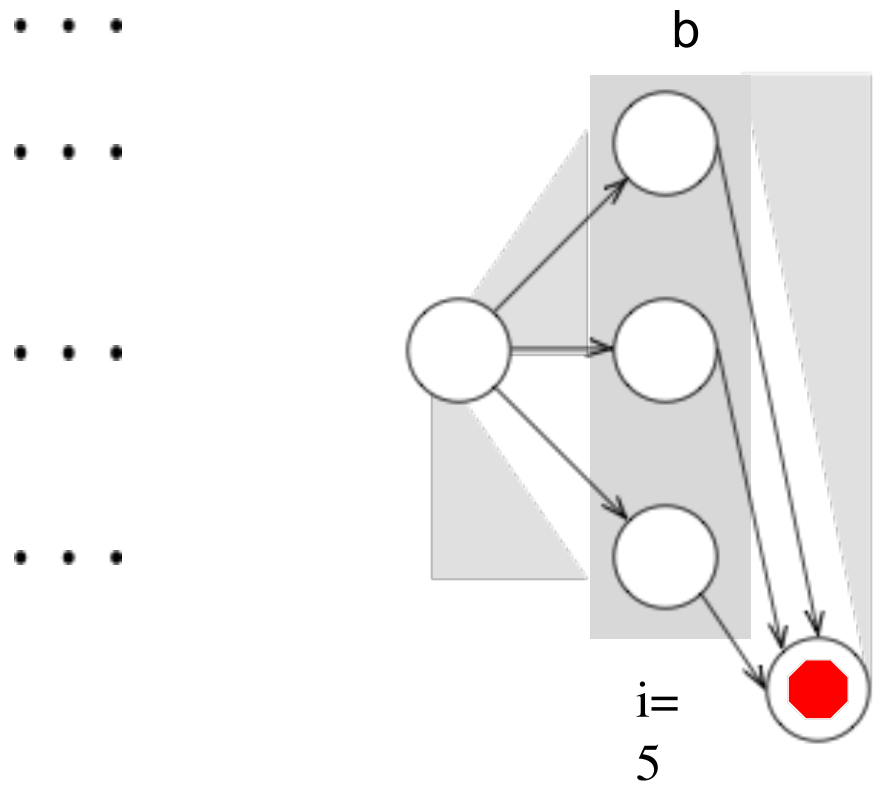
Backward Chart



Backward Chart

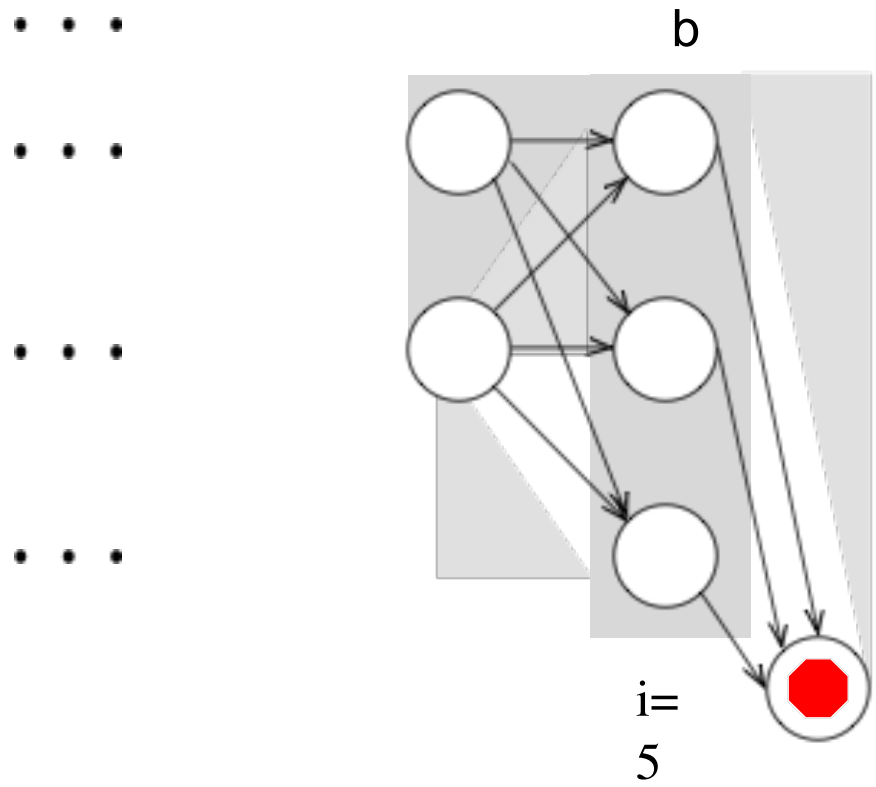


Backward Chart



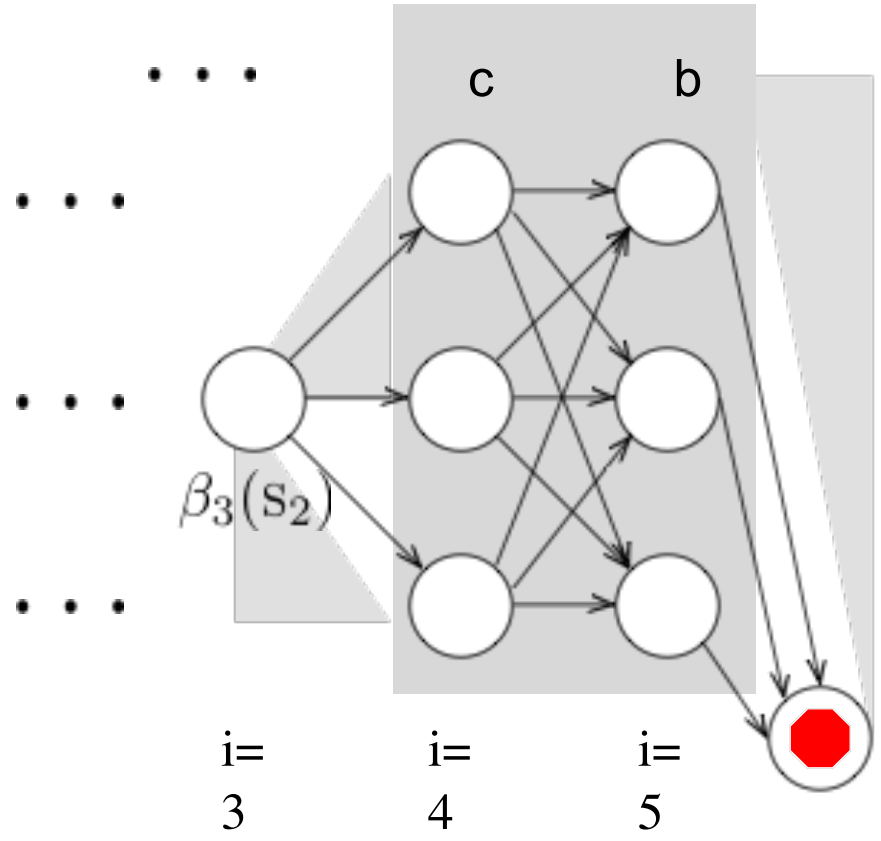
$$R_i(a) = \sum n(a \rightarrow$$

Backward Chart



Backward Chart

$$\beta_t(q) = p(x_{t+1}, \ldots, x_{|\mathbf{x}|} \mid y_t = q)$$



$$\beta_t(q) = \sum_{} p(q \rightarrow$$

Forward-Backward

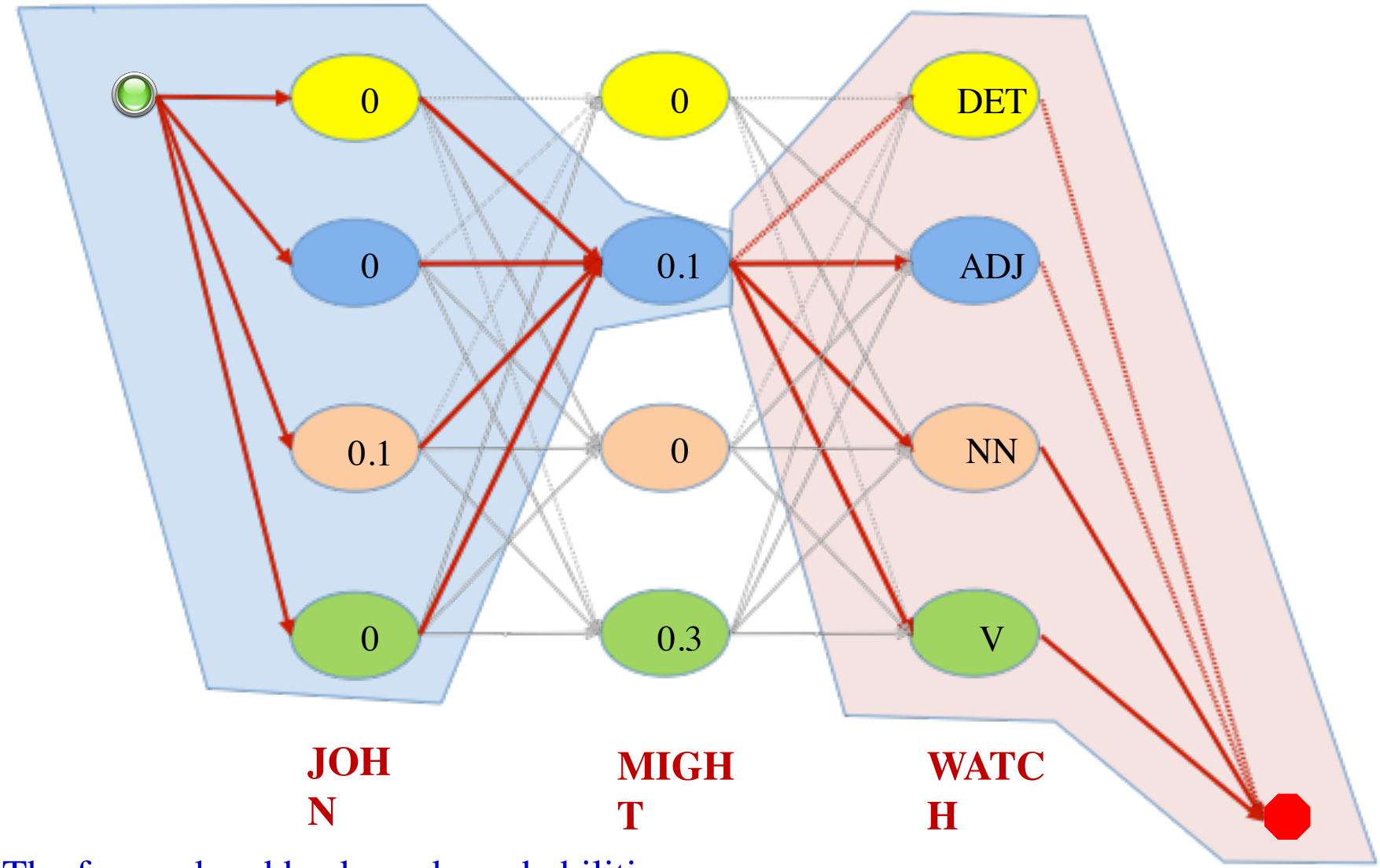
- Compute forward chart

$$\alpha_t(q) = p(\text{START}, x_1, \ldots, x_t, y_t = q)$$

- Compute backward chart

$$\beta_t(q) = p(x_{t+1}, \ldots, x_{|\mathbf{x}|}, \text{STOP} \mid y_t = q)$$

Forward Backward



The forward and backwards probabilities

Forward-Backward

- Compute forward chart

$$\alpha_t(q) = p(\text{START}, x_1, \ldots, x_t, y_t = q)$$

- Compute backward chart

$$\beta_t(q) = p(x_{t+1}, \ldots, x_{|\mathbf{x}|}, \text{STOP} \mid y_t = q)$$

$$\alpha_t(q) \times \beta_t(q)$$

$$p(\mathbf{x}, y_t = q) = \alpha_t(q) \times \beta_t(q)$$

Edge probability

- What is the probability that **x** was generated and $q \rightarrow r$ happened at time t?

$$p(x_1, \ldots, x_T, s_t$$

$$p(x_1, \ldots, x_t, s$$

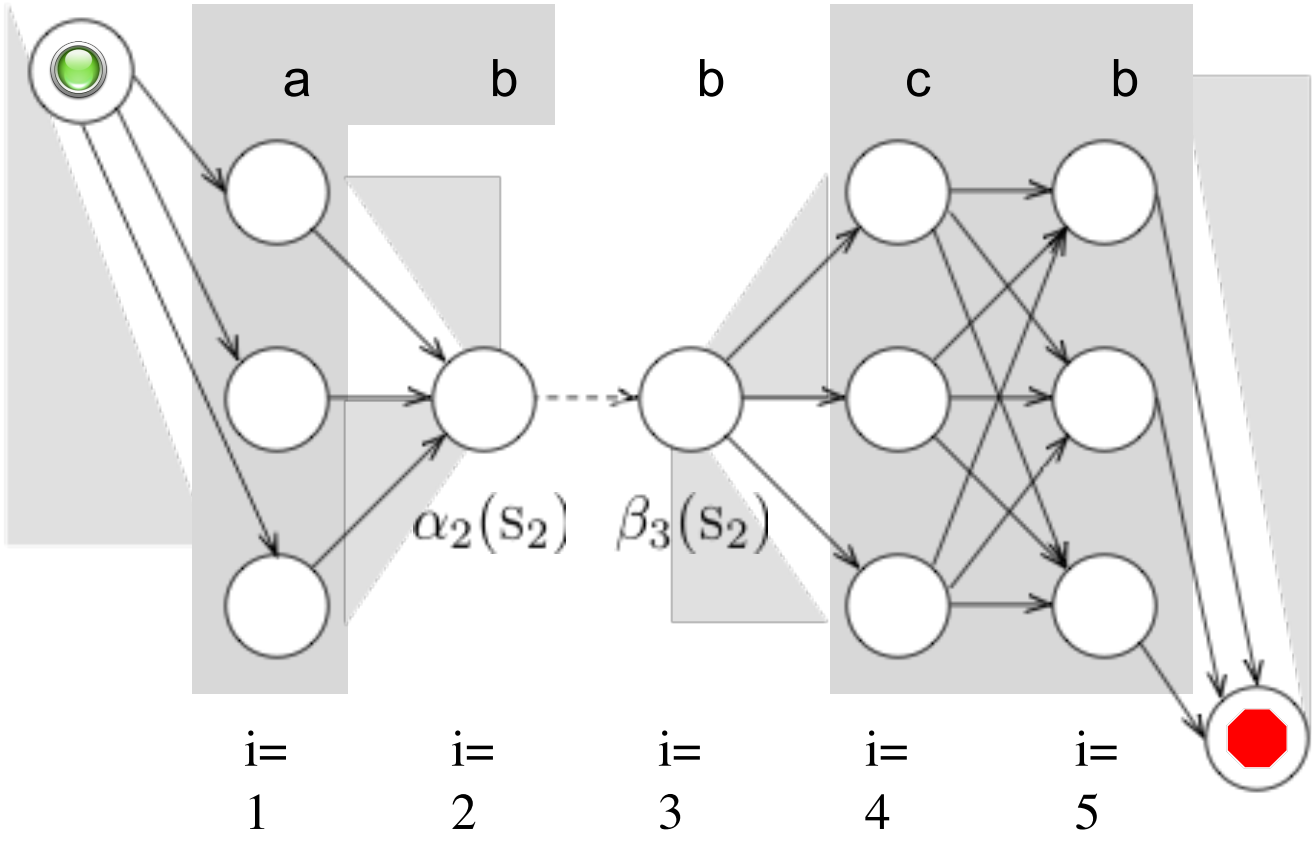
Edge probability

- What is the probability that **x** was generated and $q \rightarrow r$ happened at time t?

$$p(x_1, \ldots, x_T, s_t$$

$$p(x_1, \ldots, x_t, s$$

Forward-Backward



Actual Marginals

- Posterior Marginal

$p(x_1,$

- Edge Marginal

α

What we've done

- Able to answer questions about marginal distributions of components of finite-state models of language
  - Finite state grammars
  - HMMs

- E.g.
  - How probable is state q at time t, given $\mathbf{x}$
    - E.g. how likely is it that the second word in "John might watch" is an adjective
  - How probable is the state transition $q \rightarrow r$ at time t

A different problem

- We've answered the following questions:
  - How probable is state q at time t, given $\mathbf{x}$
  - How probable is the state transition $q \rightarrow r$ at time t, given $\mathbf{x}$

- More generic question:
  - How probable is it that the process visited state q, given $\mathbf{x}$
    - Is there an adjective in "John might watch"?

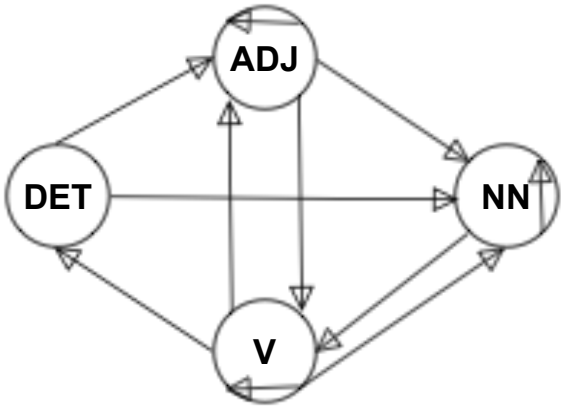
HMMs are PCFGs too

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.0	0.2	0.1
V	0.0	0.1	0.4	0.1



Emission Probabilities:

DET		ADJ		NN		V	
the	0.7	green	0.1	book	0.3	might	0.2
a	0.3	big	0.4	plants	0.2	watch	0.3
		old	0.4	people	0.2	watches	0.2
		might	0.1	person	0.1	loves	0.1
				John	0.1	reads	0.19
				watch	0.1	books	0.01

EXERCISE: Convert this HMM to a PCFG

HMM→PCFG

- Split a state into State emission NT

$$S \rightarrow Q_i \{Q_i = ADJ, NN$$

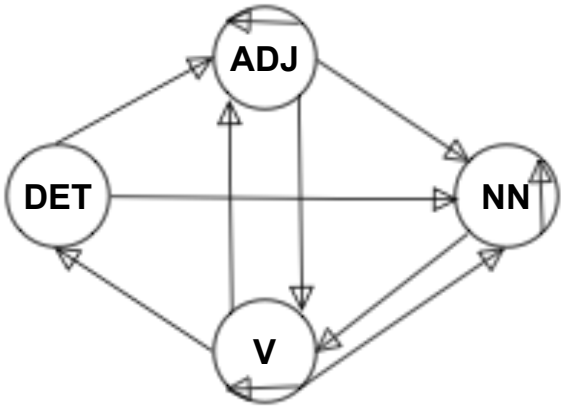
HMMs can be cast as PFAs too

Initial Probabilities:

DET	ADJ	NN	V
0.5	0.1	0.3	0.1

Transition Probabilities:

	DET	ADJ	NN	V
DET	0.0	0.0	0.0	0.5
ADJ	0.3	0.2	0.1	0.1
NN	0.7	0.0	0.2	0.1
V	0.0	0.1	0.4	0.1



Emission Probabilities:

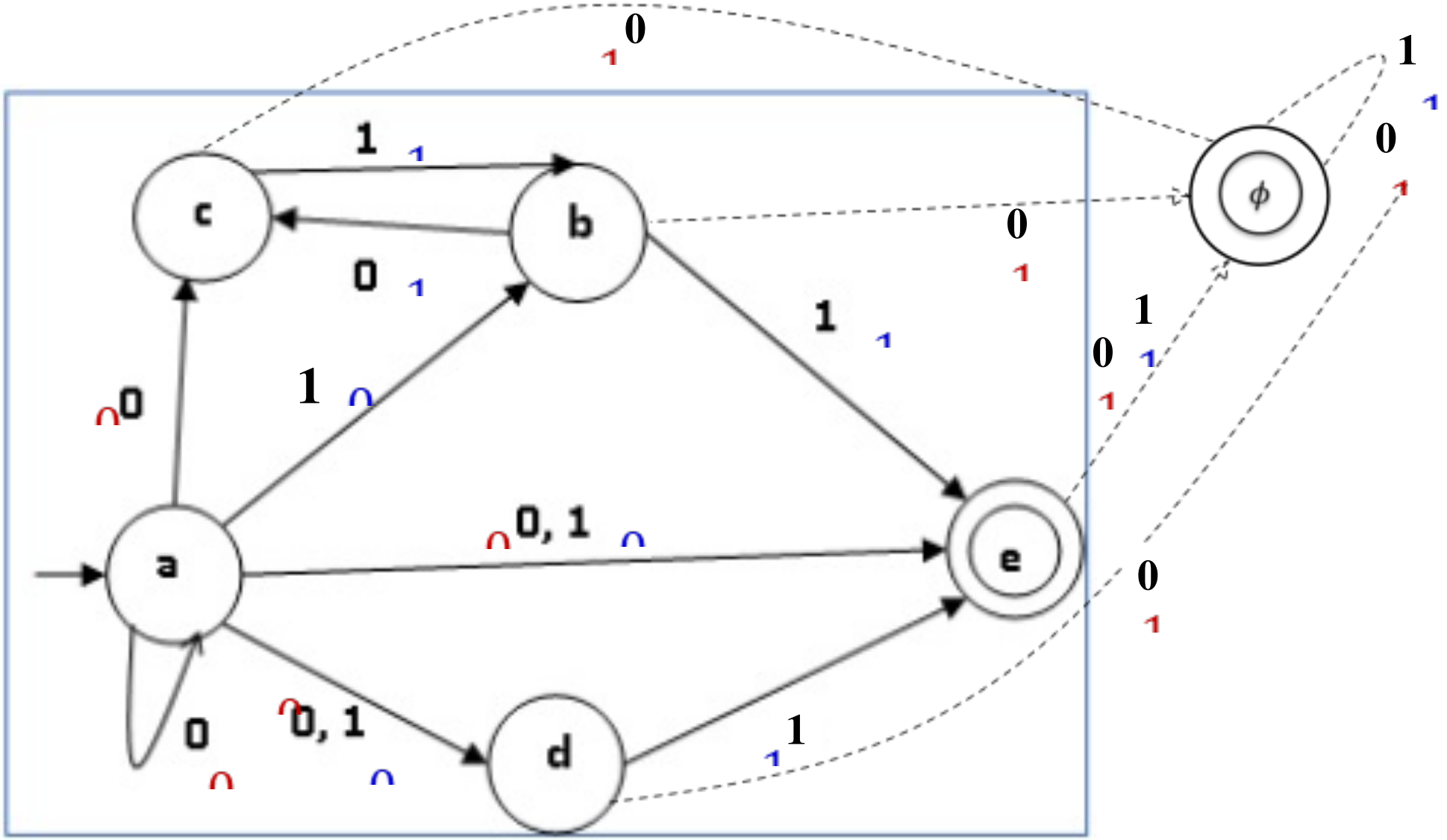
DET		ADJ		NN		V	
the	0.7	green	0.1	book	0.3	might	0.2
a	0.3	big	0.4	plants	0.2	watch	0.3
		old	0.4	people	0.2	watches	0.2
		might	0.1	person	0.1	loves	0.1
				John	0.1	reads	0.19
				watch	0.1	books	0.01

EXERCISE: Convert this HMM to a PFA

HMM→PFA

- First, recall an earlier grammar

Inference in a PFA



- We aren't interested in state se

HMM→PFA: Back to our grammar

- Let $Q = \{ADJ, NN,$
- Let $W = \{w_1, w_2, ..$
  - ■ is a termination sy

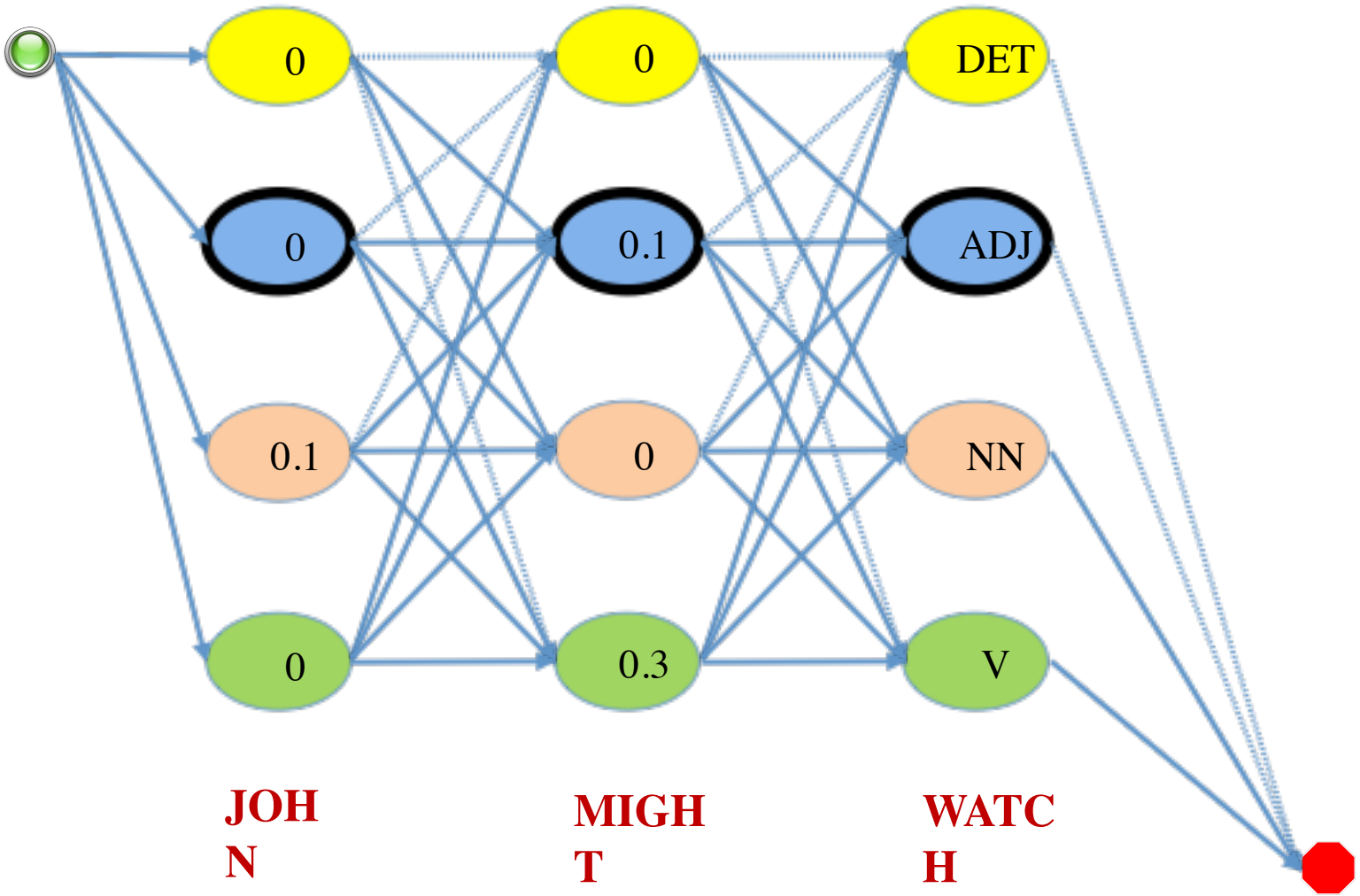
    sentence

A different problem

- We've answered the following questions:
  - How probable is state q at time t, given **x**
  - How probable is the state transition $q \rightarrow r$ at time t, given **x**

- More generic question:
  - How probable is it that the process visited state q, given **x**
    - E.g. does the sentence have an adjective

Visiting a state

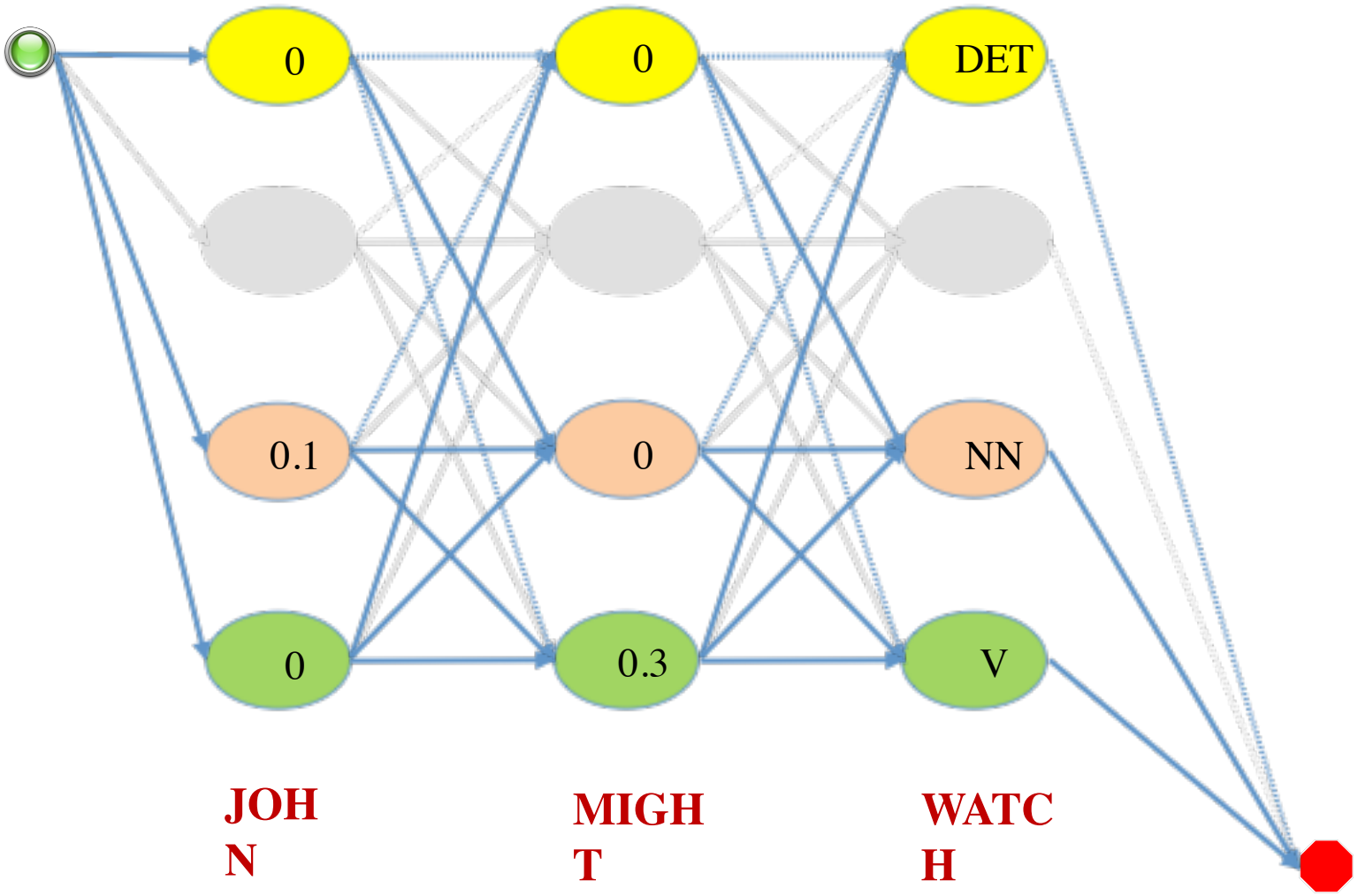


- $P(state\ s = visited, \mathbf{x})$ is the p
  - E.g. "What is the probability that
  - This is the total probability of the

Derivation by ablation

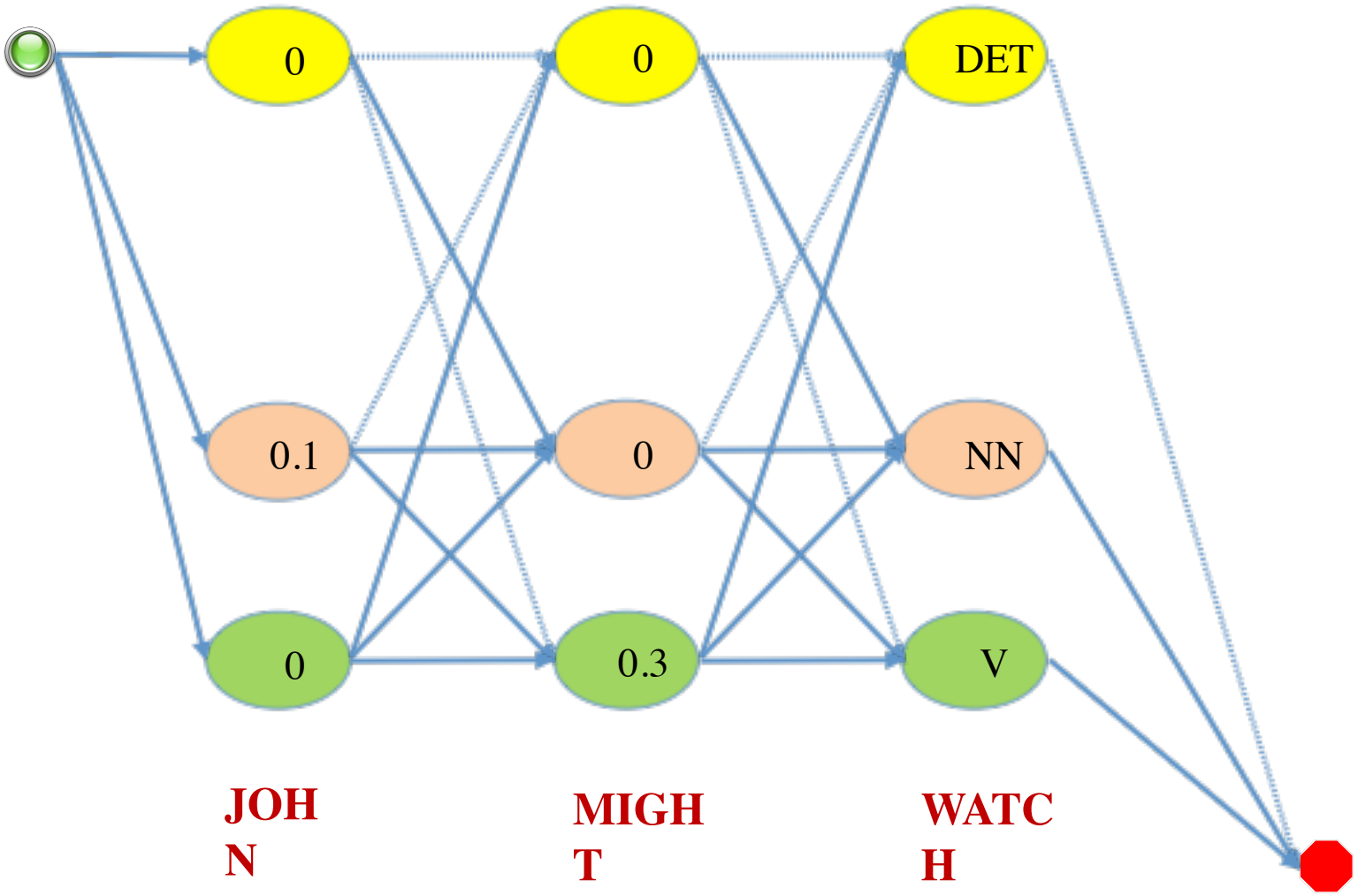
- $p(\mathbf{x}) = p(s, \mathbf{x}) + p(\bar{s}$
  - s = state s is visited at
  - $\bar{s}$ = state s is never vis

Not visiting a state



- The portion of the trellis v

Not visiting a state



- The portion of the trellis where
  - This is complete; there are no c

Derivation by ablation

- $p(\mathbf{x}) = p(s, \mathbf{x}) + p(\bar{s}$
  - s = state s i
  - $\bar{s}$ = state s is never vis

Computed by the forward algorithm on the *reduced* trellis

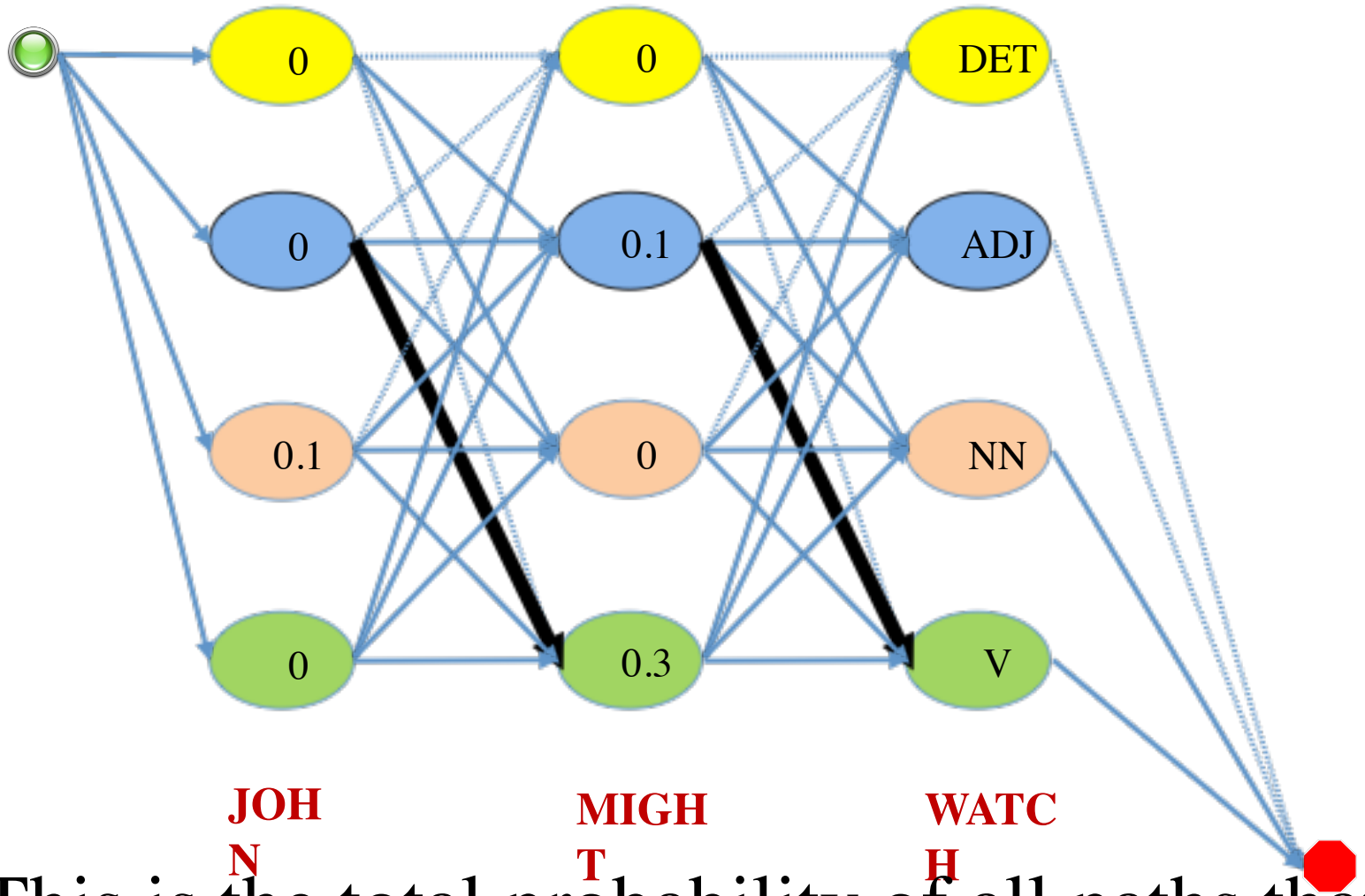
Computed by the forward algorithm on the complete trellis

A different problem

- We've answered the following questions:
  - How probable is state q at time t, given **x**
  - How probable is the state transition $q \rightarrow r$ at time t, given **x**

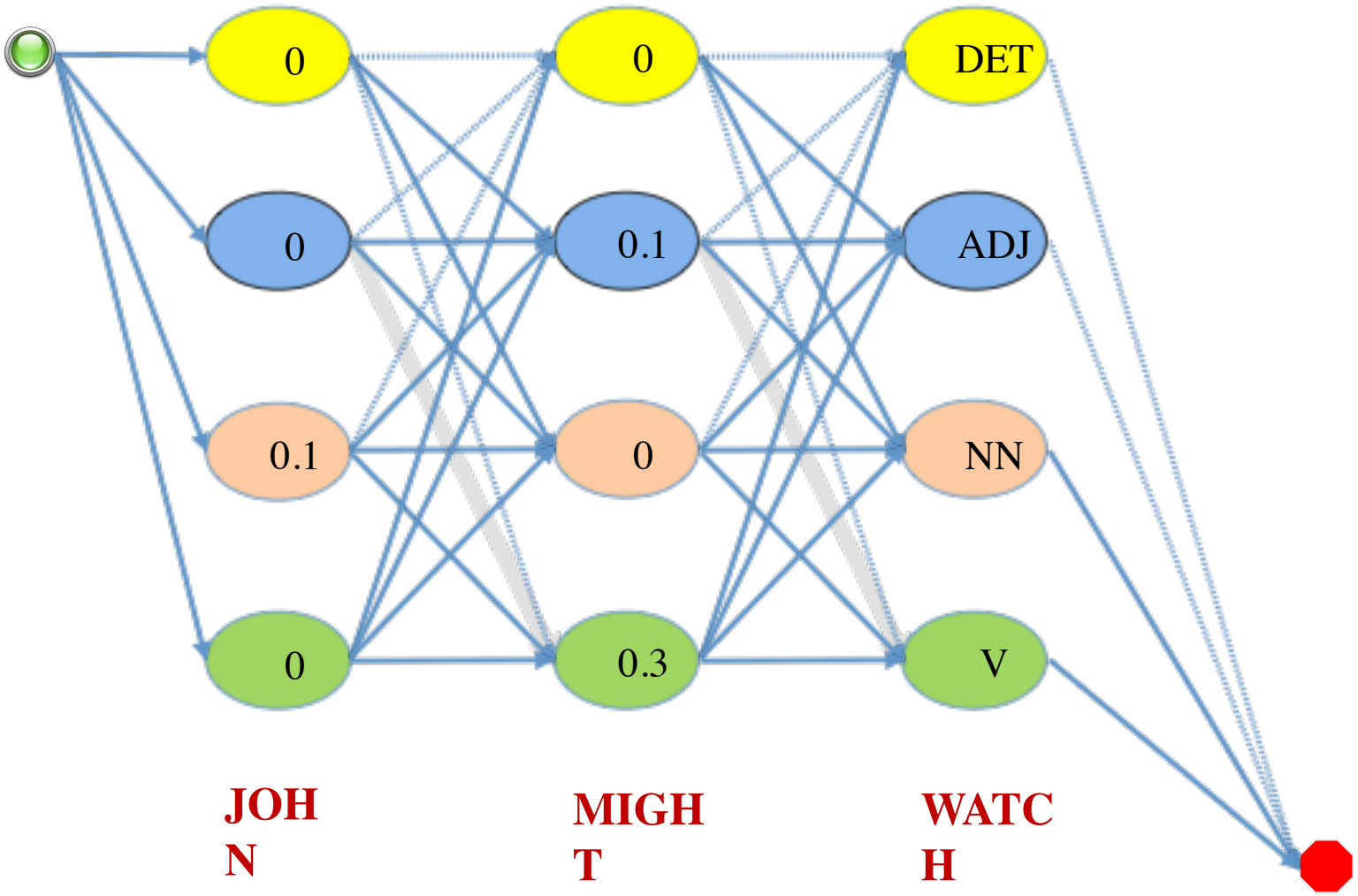
- More generic question:
  - How probable is it that the process visited state q, given **x**
  - How probable is it that the transition $q \rightarrow r$ occurred, given **x**
    - E.g. Is an adjective followed by a verb in this sentence?

Visiting a transition



- This is the total probability of all paths that use the shown transition
  - No possible to isolate this portion of the trellis

Not visiting a transition

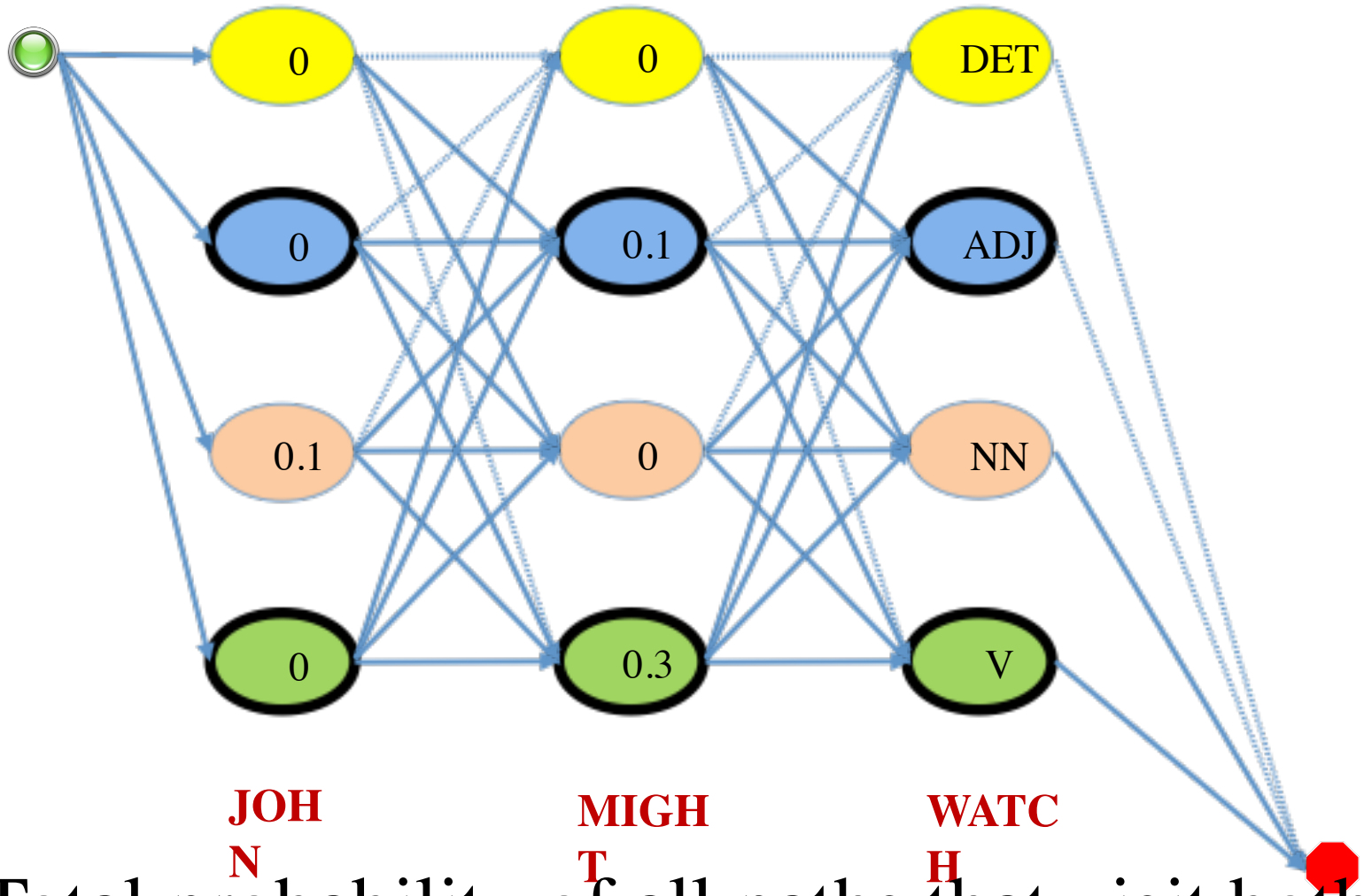


- Delete the transition a

More complex inferences

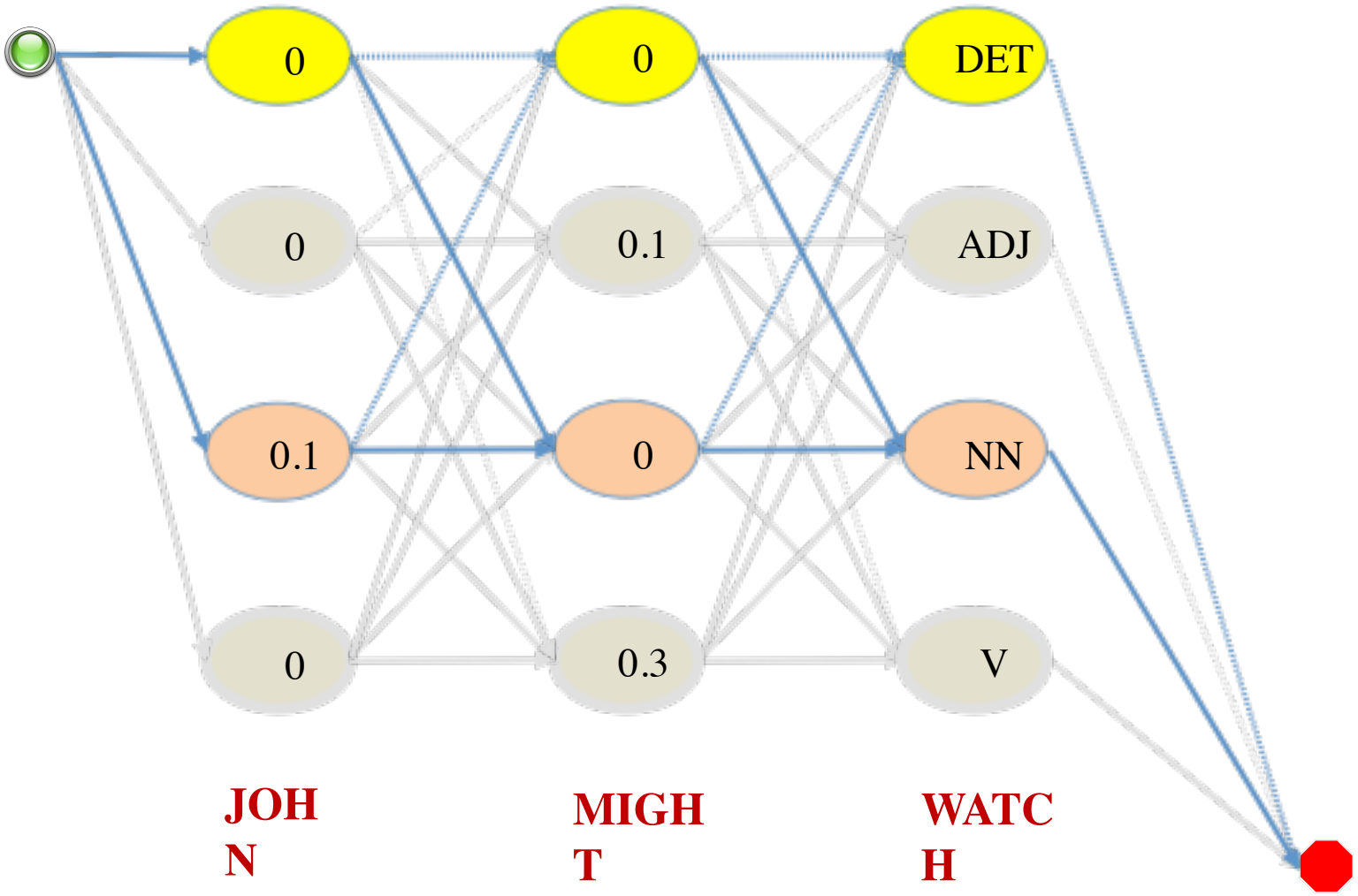
- What is the probability that *both* state *s* AND state *r* were visited?
  - What is the probability that "John might watch" includes both a verb and an adjective?

Visiting multiple states



- Total probability of all paths that visit both the blue and green states
  - Again, not possible to isolate the corresponding

NOT visiting multiple states



- All paths that visit ne

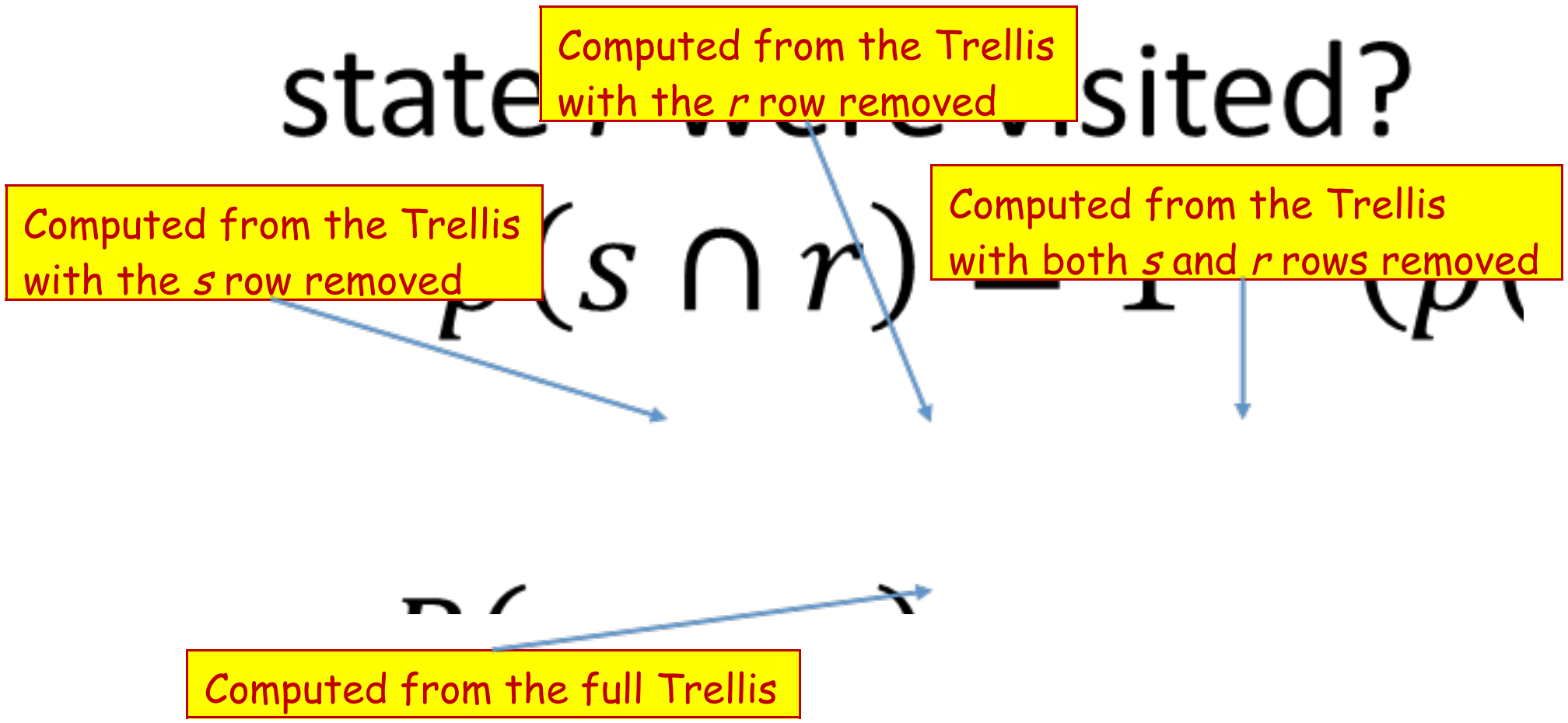
More complex inferences

- What is the probabilit
  state *r* were visited?

$$p(s \cap r) = 1 - (p($$

More complex inferences

- What is the probabilit
  state / were visited?



More complex inferences

- Other more complex inferences can be similarly obtained

- Becomes increasingly more computationally expensive as the order of the inference increases

Higher-level grammars

- We have derived probabilistic inferences from PFAs and HMMs

- More generally we want similar inferences from CFGs and PCFGs
  - Given word sequence $\boldsymbol{w}$, what is the probability of having a constituent of type Z from i to j?
    - '*A person who trusts no one can't be trusted*' : what is the probability that the "A person who trusts no one" is a noun phrase?

Generalizing Forward-Backward

- Inference in HMMs was performed using the forward-backward algorithm
  - Recall that HMMs are instances of PCFGs

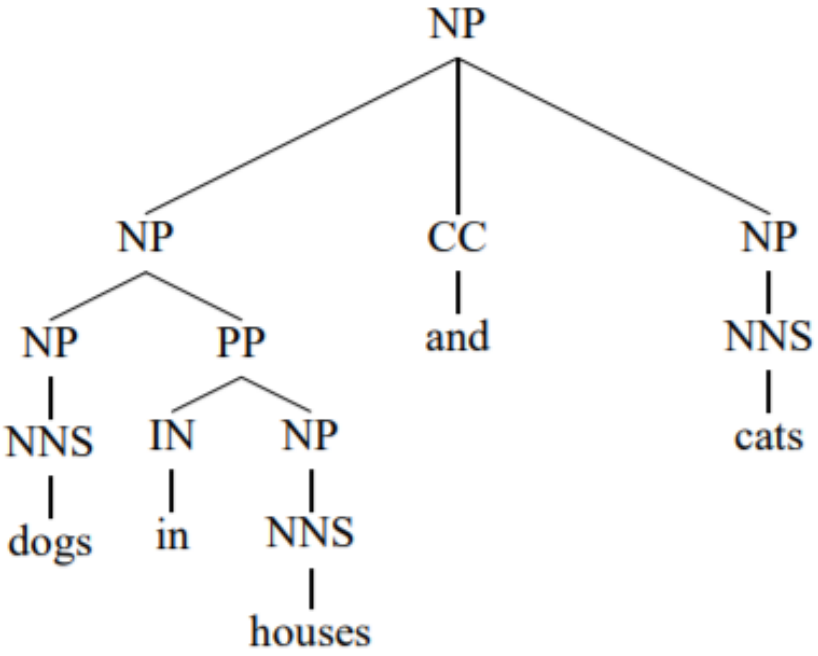
- For more general PCFGs we will use the inside-outside algorithm
  - A generalization of the forward backward algorithm
  - Builds upon the CKY algorithm

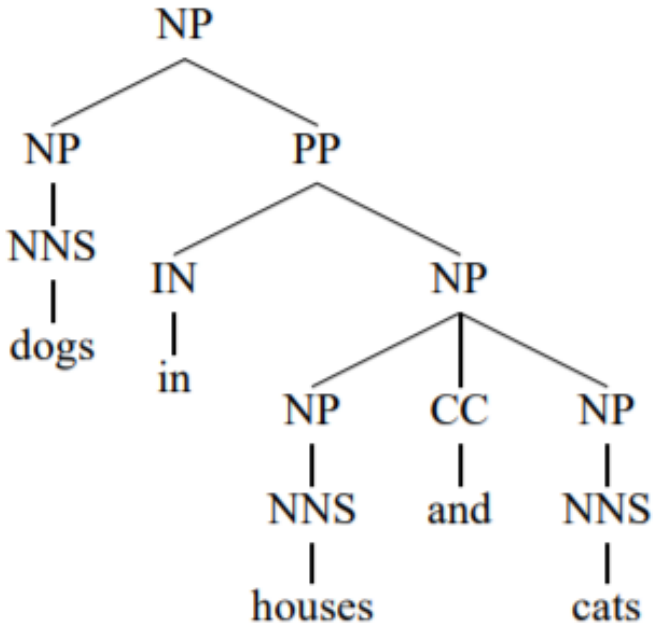
Inside/Outside Algorithm

Have you seen this man somewhere?

- "Trainable grammars for speech recognition," J. K. Baker, 1979

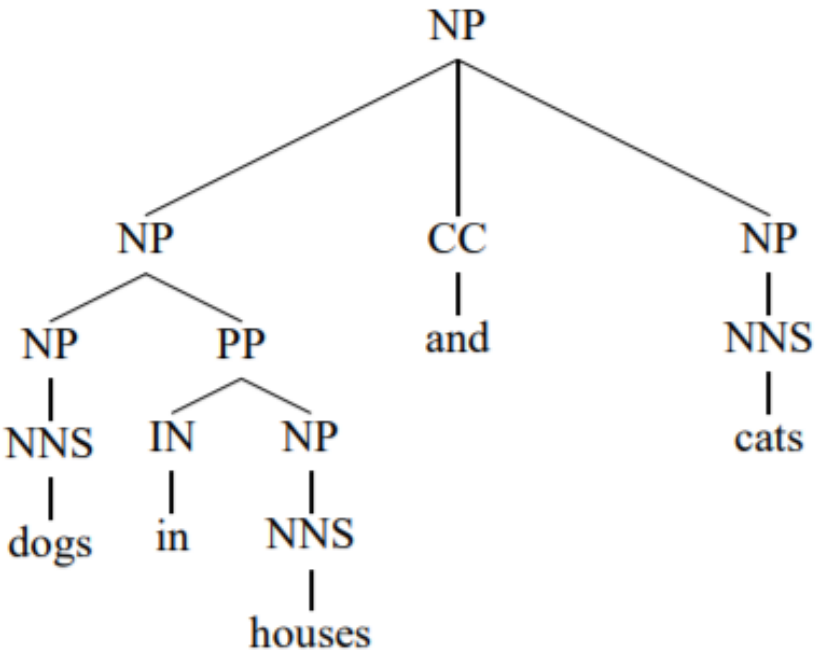
Inferences we would like to make..

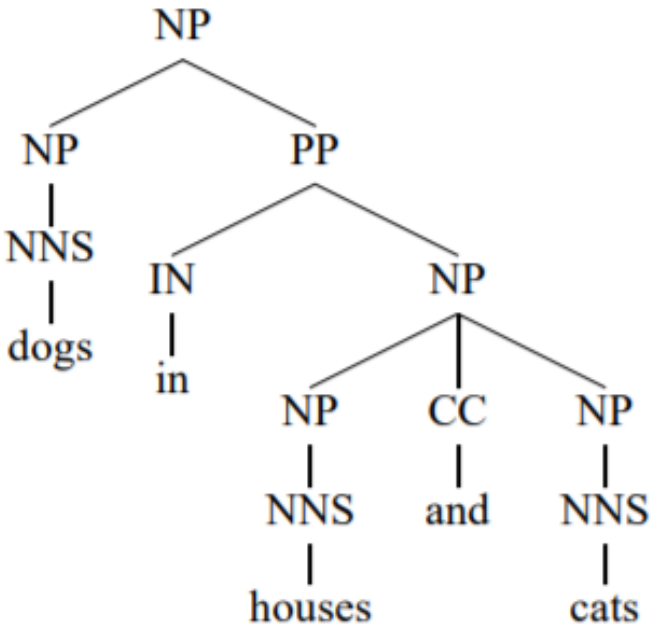




- What is the probability of "dogs in houses and cats"?
- What is the probability that "houses and cats" is a clause by itself?

Inferences we would like to make..





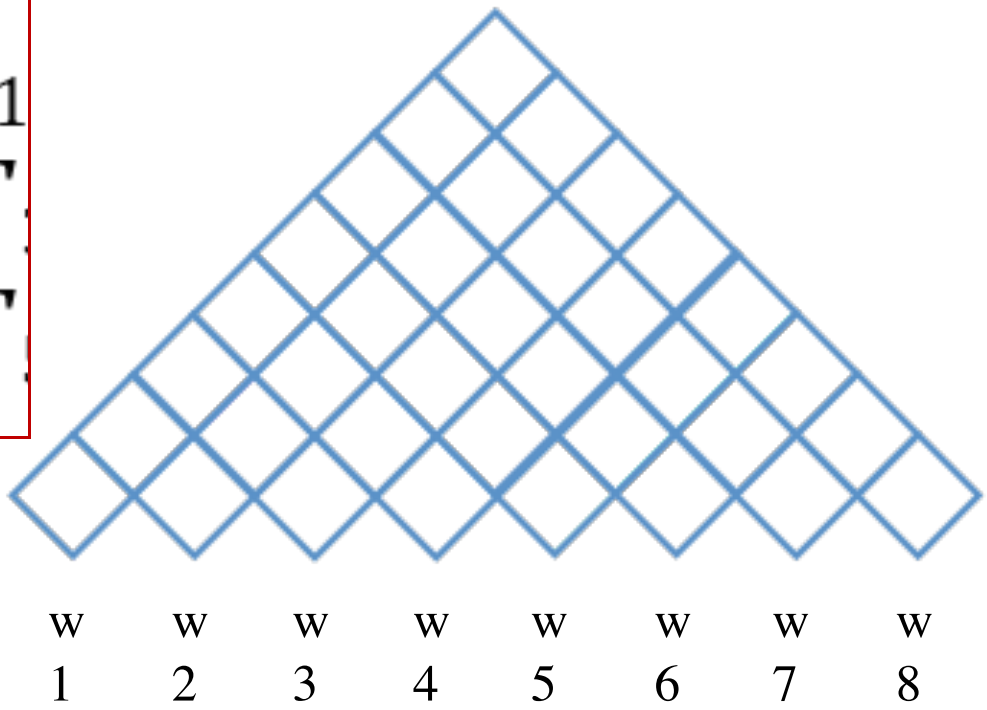
- Which of the probability of "dogs in houses and cats"
  - P("dogs in houses and cats")
- What is the probability that "houses and cats" is a clause by itself?

Recall the CKY algorithm

$R_0: \quad S \rightarrow NT_1$

$R_1: NT_1 \rightarrow NT$

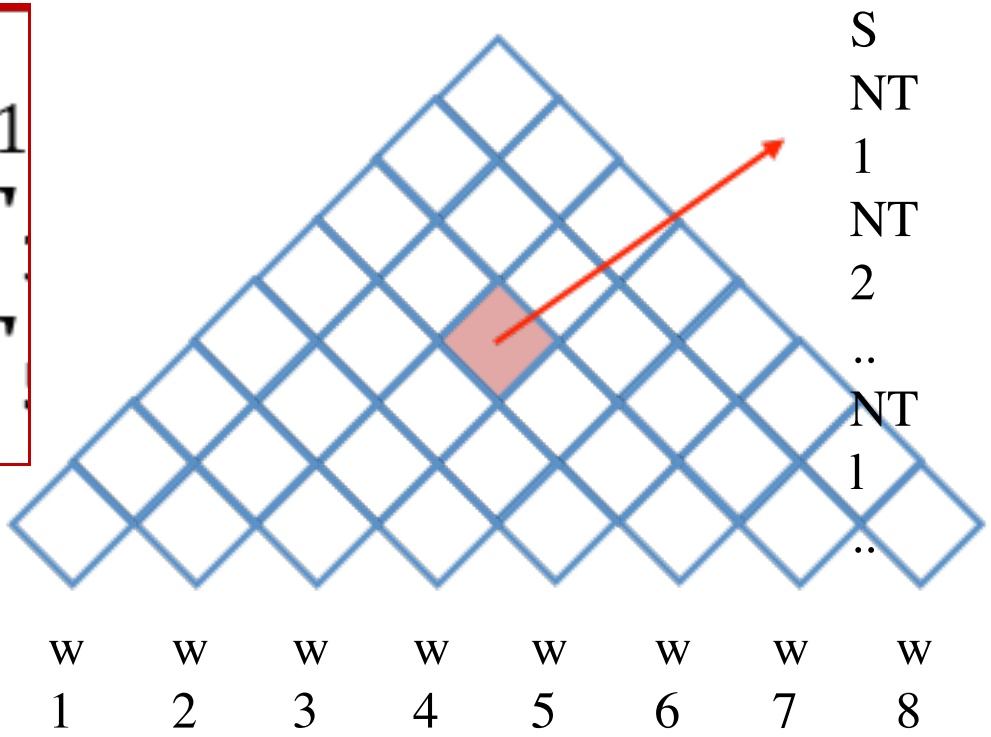
$R_2: NT_2 \rightarrow NT$



- Given: A PCFG in CNF, and a word sequence
- Build a skeleton that can hold every possible tree

Recall the CKY algorithm

$$R_0: \quad S \rightarrow NT_1$$
$$R_1: NT_1 \rightarrow NT$$
$$R_2: NT_2 \rightarrow NT$$



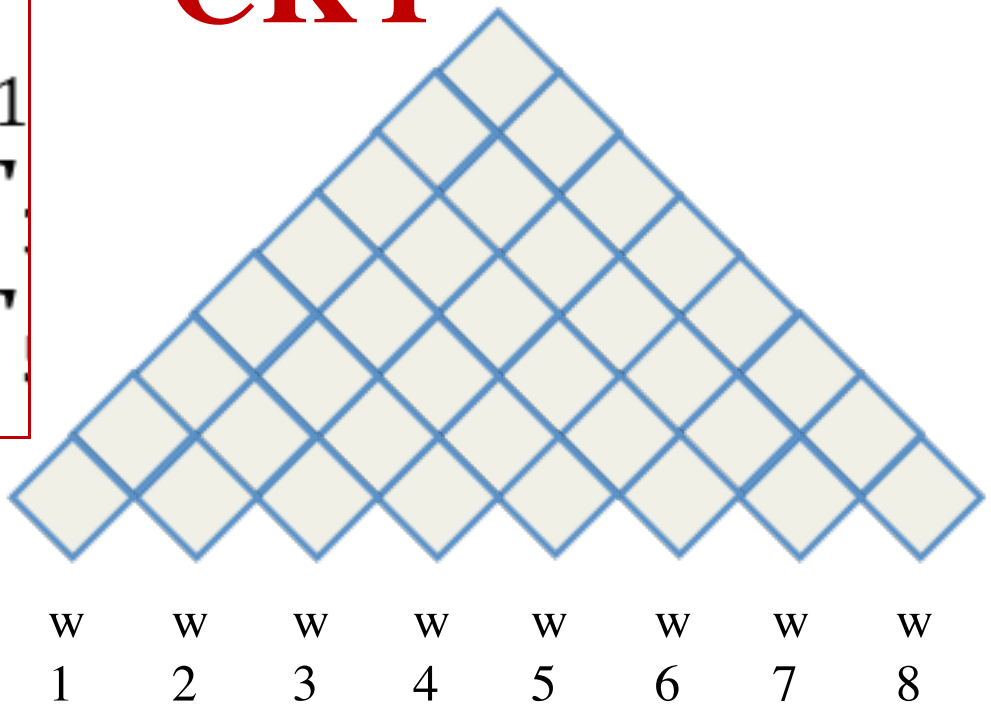
- *Each* box in the grid (potentially) holds *every* non-terminal

Probability computation using CKY

$R_0: \quad S \rightarrow NT_1$

$R_1: NT_1 \rightarrow NT$

$R_2: NT_2 \rightarrow NT$



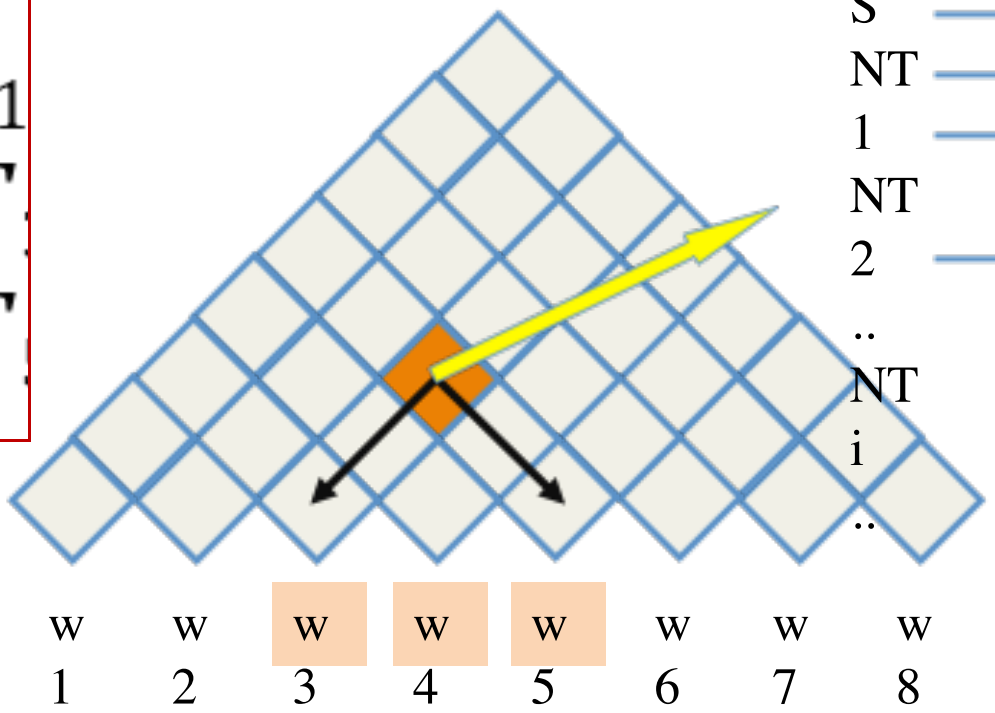
-
  - What we desire to comp
    - $P(w_1, \ldots, w_N)$: Probabili

The Inside Algorithm

$R_0: \quad S \rightarrow NT_1$

$R_1: NT_1 \rightarrow NT$

$R_2: NT_2 \rightarrow NT$

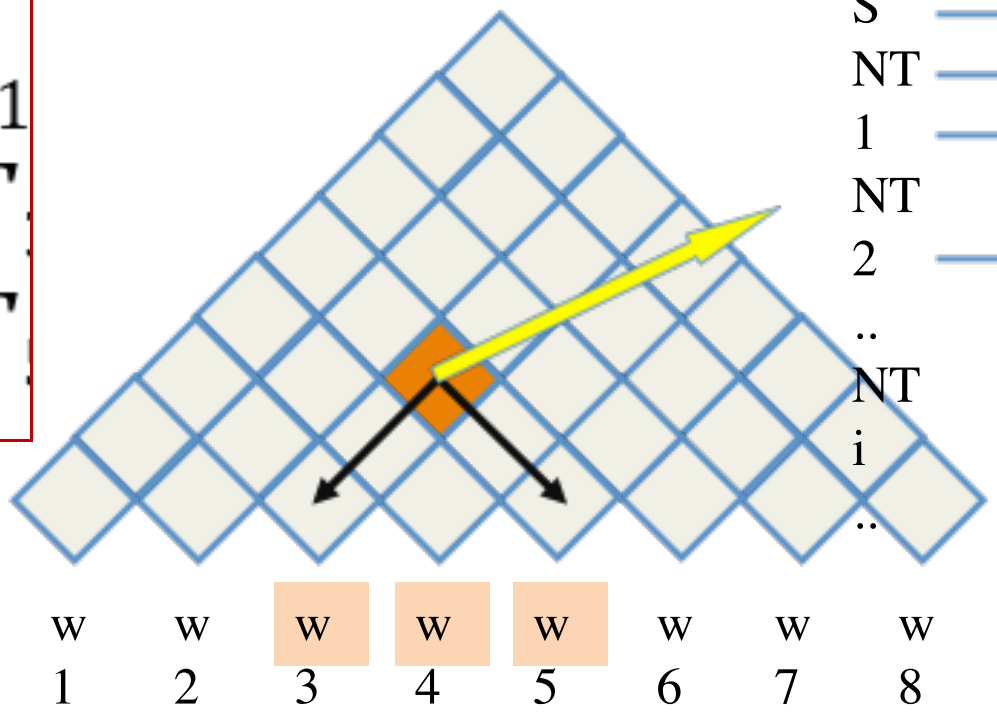


- Let $\alpha(NT, i, j)$ be the p

terminal NT produced

The Inside Algorithm

$$R_0: \quad S \rightarrow NT_1$$
$$R_1: NT_1 \rightarrow NT$$
$$R_2: NT_2 \rightarrow NT$$

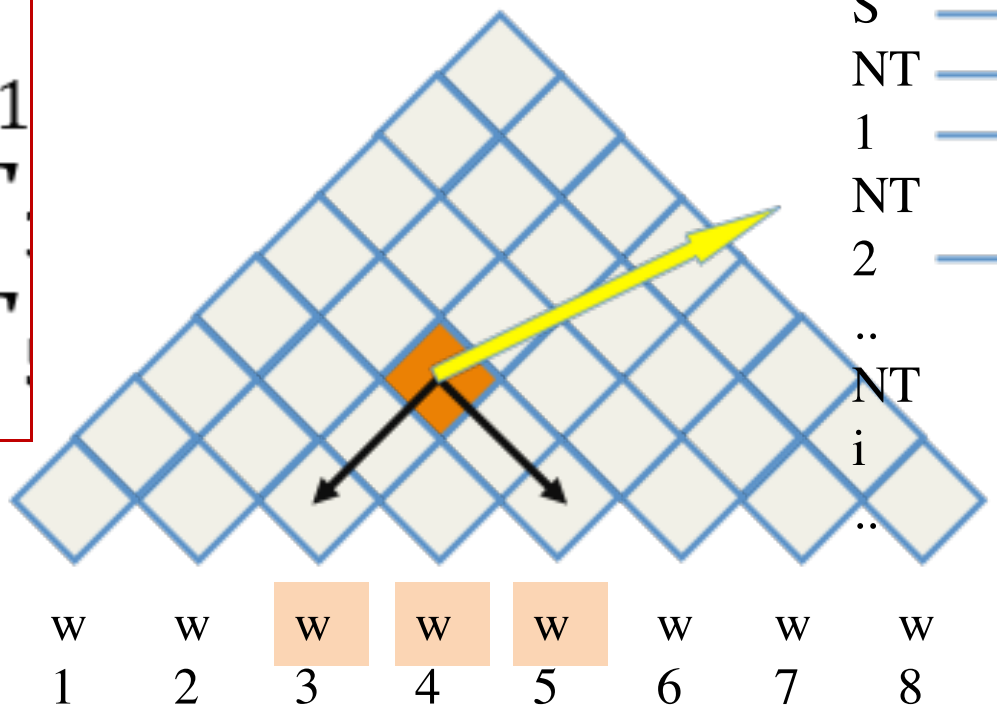


- $\alpha(NT, i, j) =$

- This can be computed

The Inside Algorithm

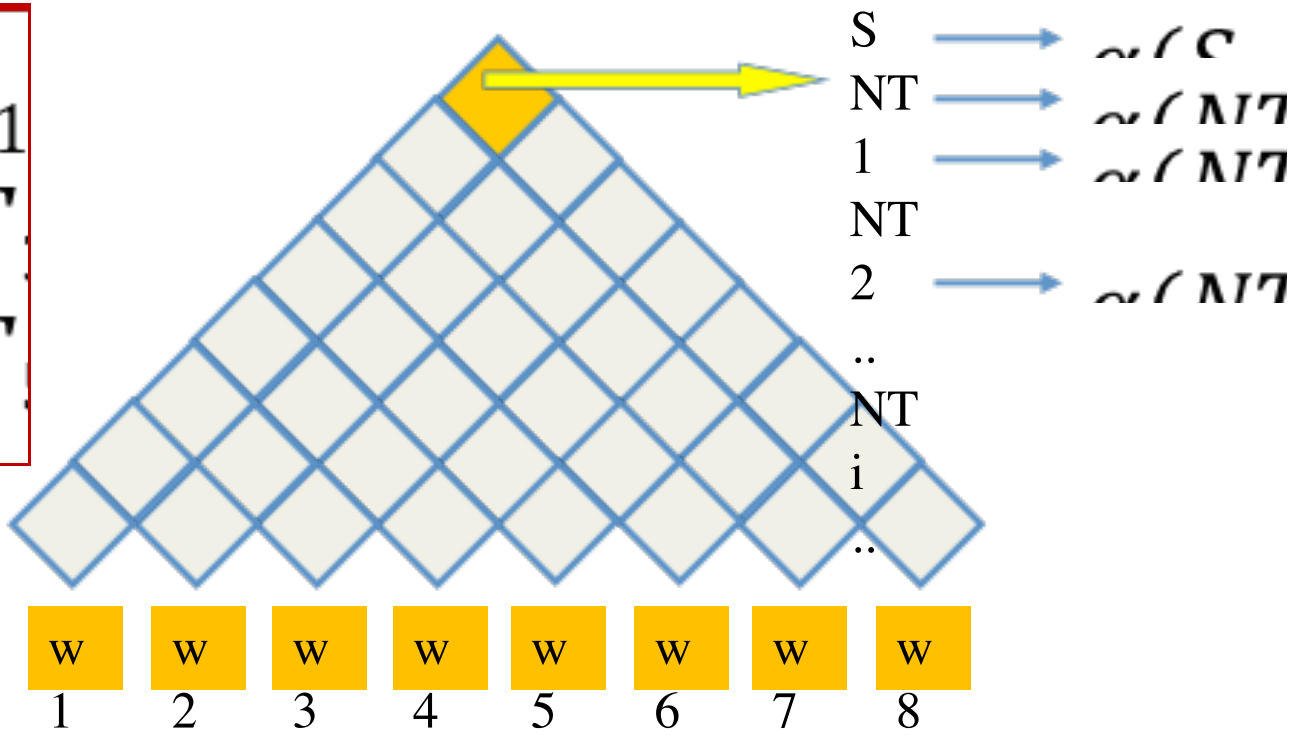
$$R_0: \quad S \rightarrow NT_1$$
$$R_1: NT_1 \rightarrow NT$$
$$R_2: NT_2 \rightarrow NT$$



- $\alpha(NT, i, j) =$
  - The probability that words $i \ldots j$ a
    marginalizing out the NT

The Inside Algorithm

$$R_0: \quad S \rightarrow NT_1$$
$$R_1: NT_1 \rightarrow NT$$
$$R_2: NT_2 \rightarrow NT$$



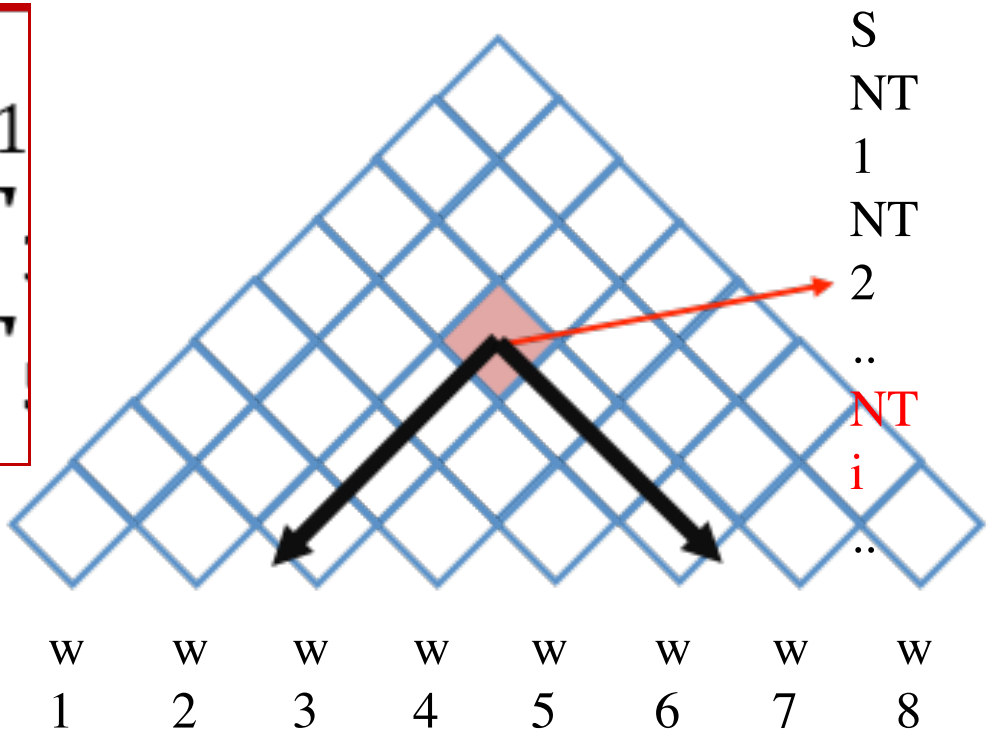
-
  - The total probability of
    is the sum of the alphas

Recall the CKY algorithm

$R_0: \quad S \rightarrow NT_1$

$R_1: NT_1 \rightarrow NT$

$R_2: NT_2 \rightarrow NT$



- What we desire to compute:
  - $P(w_1, \ldots, w_N)$: Probability of
  - $P(NT_i \in c(i,i) | W) \cdot$ Probabi

Conditional vs Joint

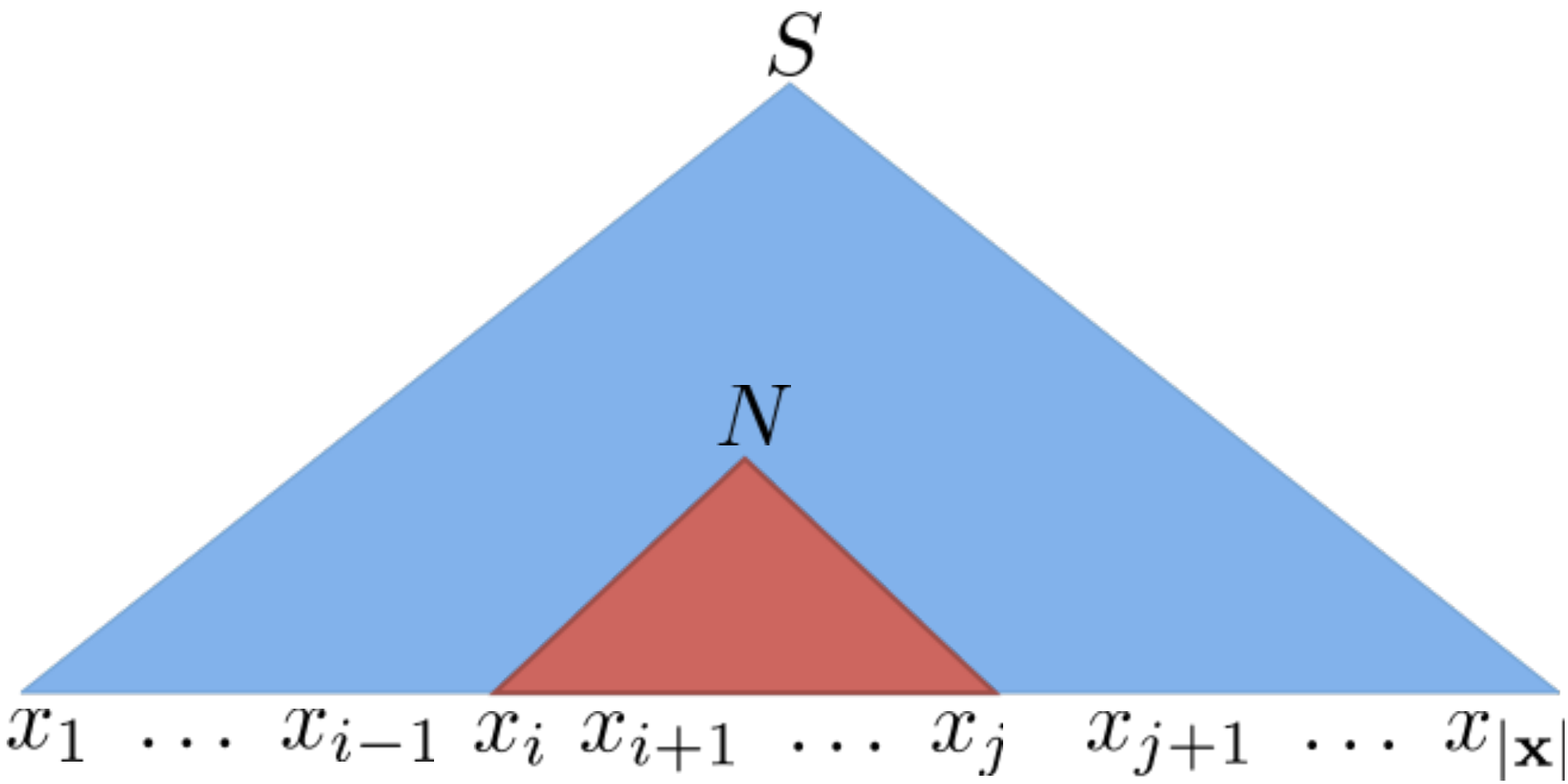
- $P(NT_i \in c(i,j)|W)$ (
- So, we must comput

Posterior Marginals

- Marginal inference question for PCFGs
  - Given **w**, what is the probability of having a constituent of type Z from *i* to *j*?
  - Given **w**, what is the probability of having a constituent of ***any*** type from *i* to *j*?
  - Given **w**, what is the probability of using rule Z -> XY to derive the span from *i* to *j*?

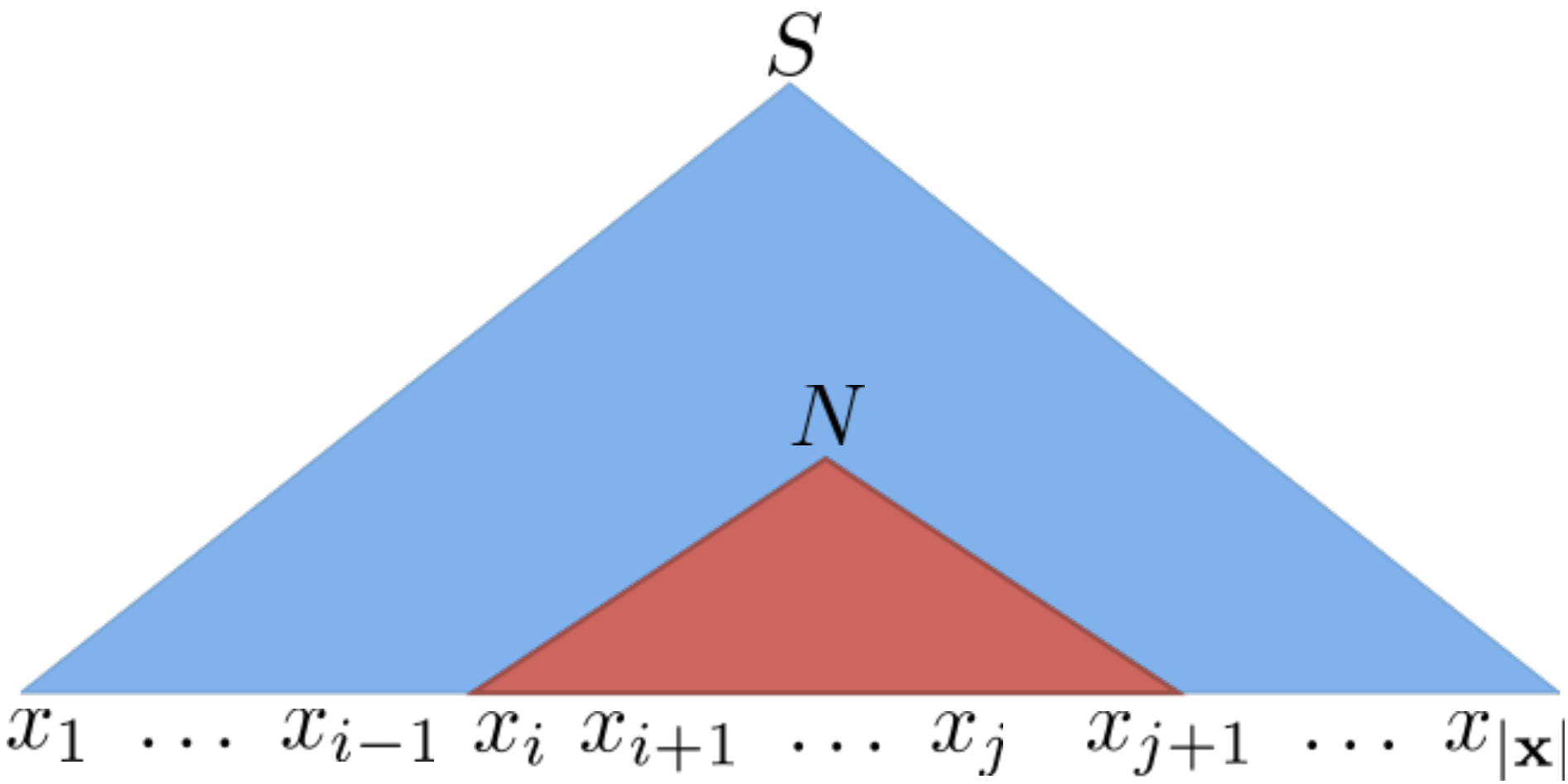
Inside Algorithm

$$\alpha_{[i,j]}(N) = p(x_i, x_{i+1}, \ldots, x_j \mid N; \mathcal{G})$$



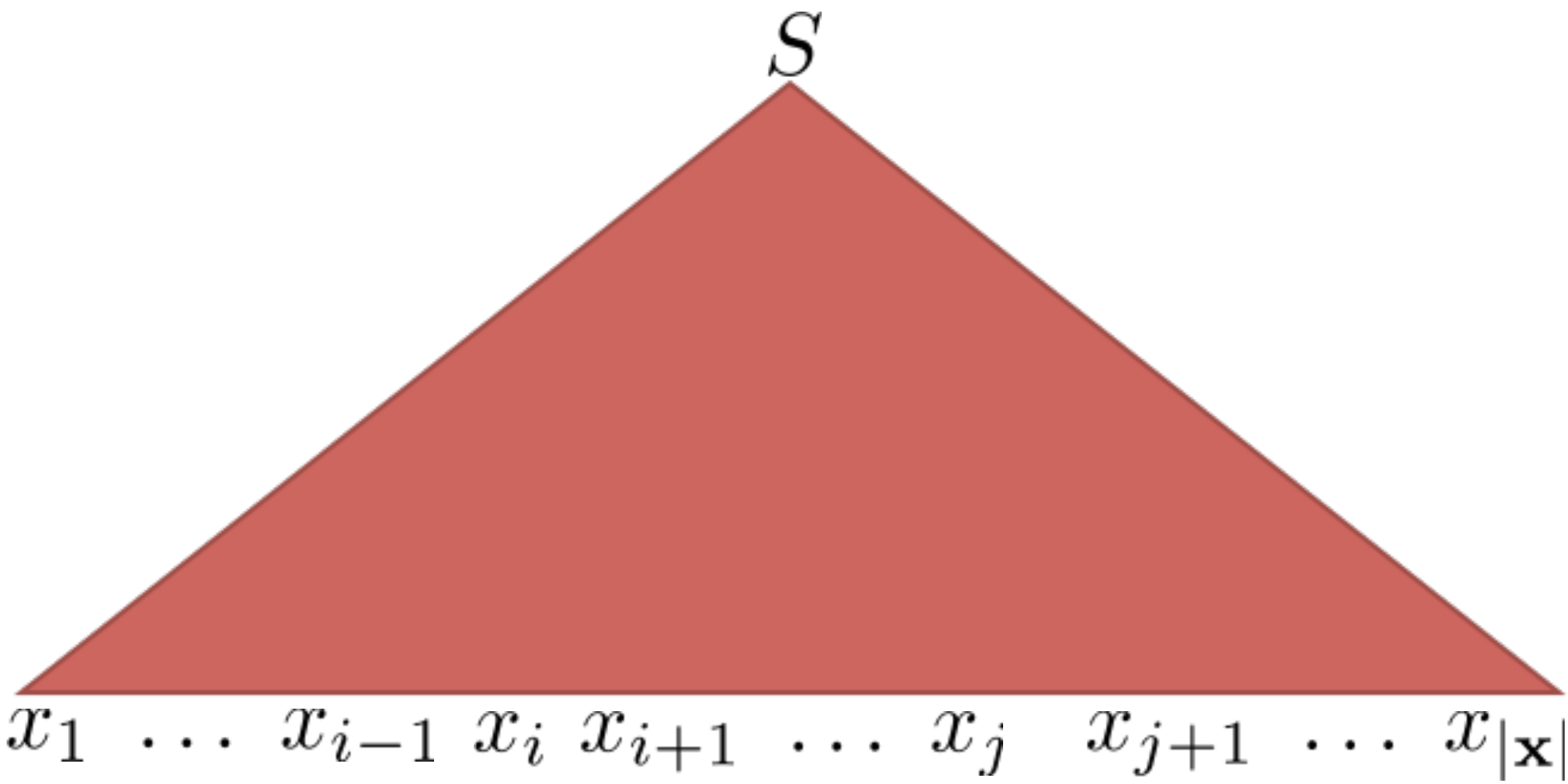
Inside Algorithm

$$\alpha_{[i,j]}(N) = p(x_i, x_{i+1}, \ldots, x_j \mid N; \mathcal{G})$$



Inside Algorithm

$$\alpha_{[i,j]}(N) = p(x_i, x_{i+1}, \ldots, x_j \mid N; \mathcal{G})$$



CKY Inside Algorithm

Base case(s)

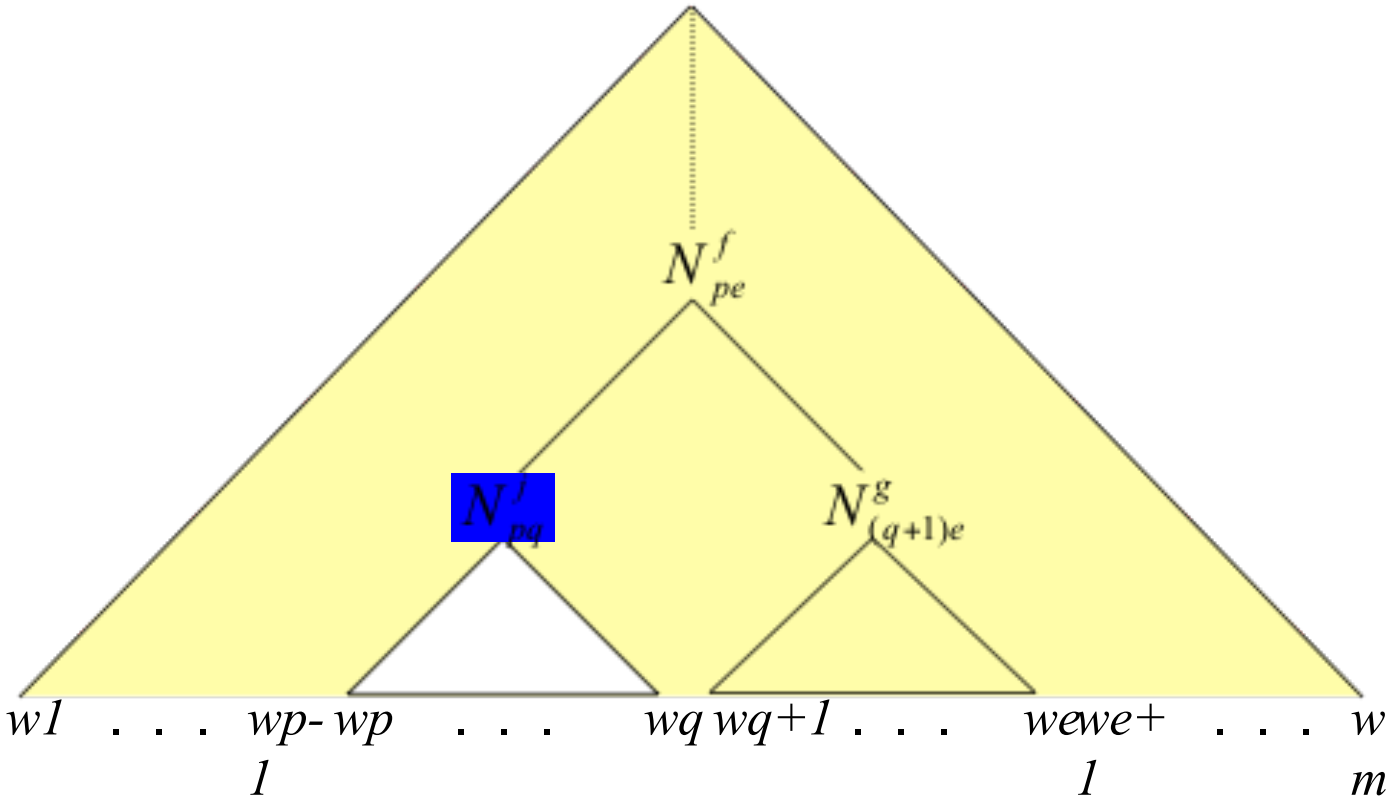
$$\alpha_{[i,i+1]}(Z) = p(Z \rightarrow x_i)$$

Recurrence

$$\alpha_{[i,j]}(Z) = \sum_{k=i+1}^{j-1} \sum_{(Z \rightarrow XY) \in G} \alpha_{[i,k]}(X) \times \alpha_{[k,j]}(Y) \times p(Z \rightarrow XY)$$

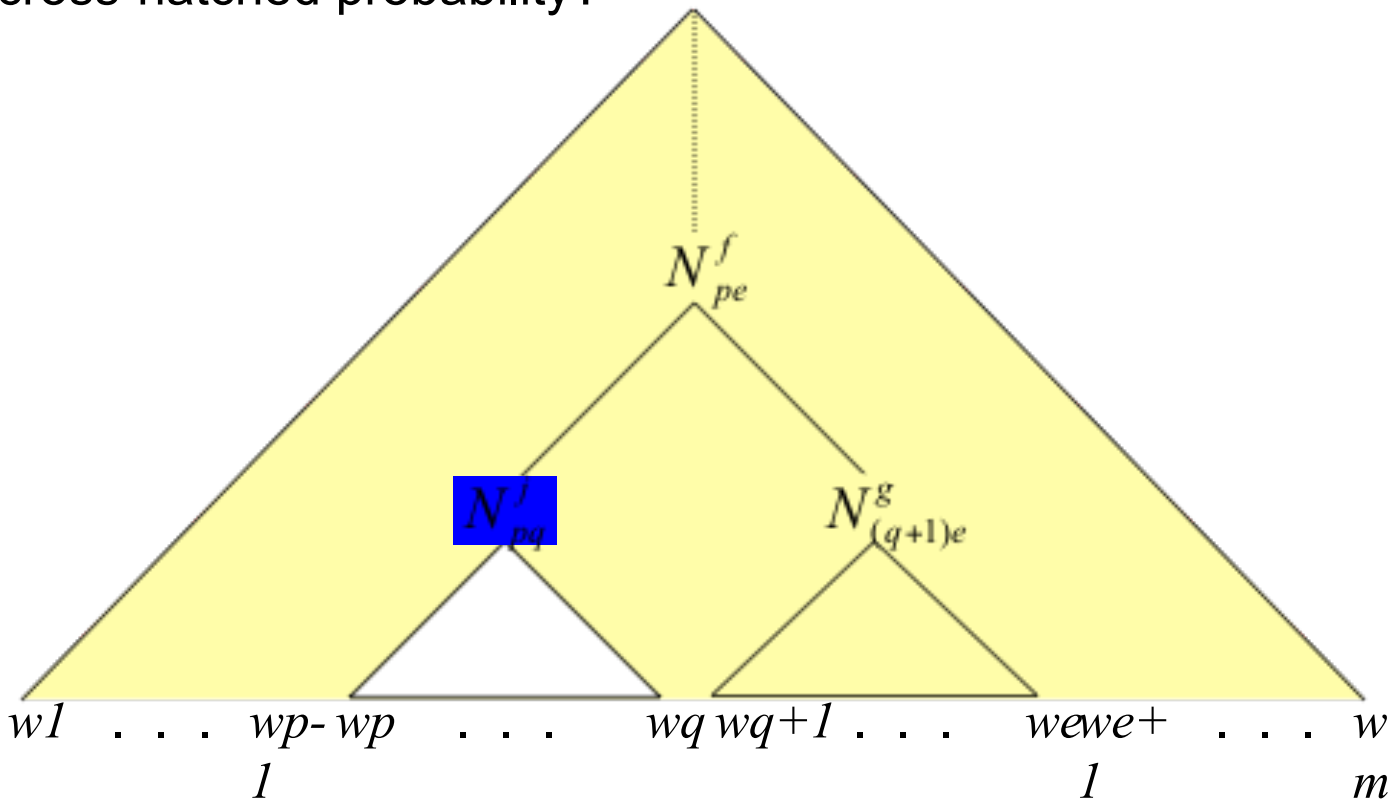
Outside probabilities: decomposing the problem

The shaded area represents the outside probability $\alpha_j(p,q)$ which we need to calculate. How can this be decomposed?



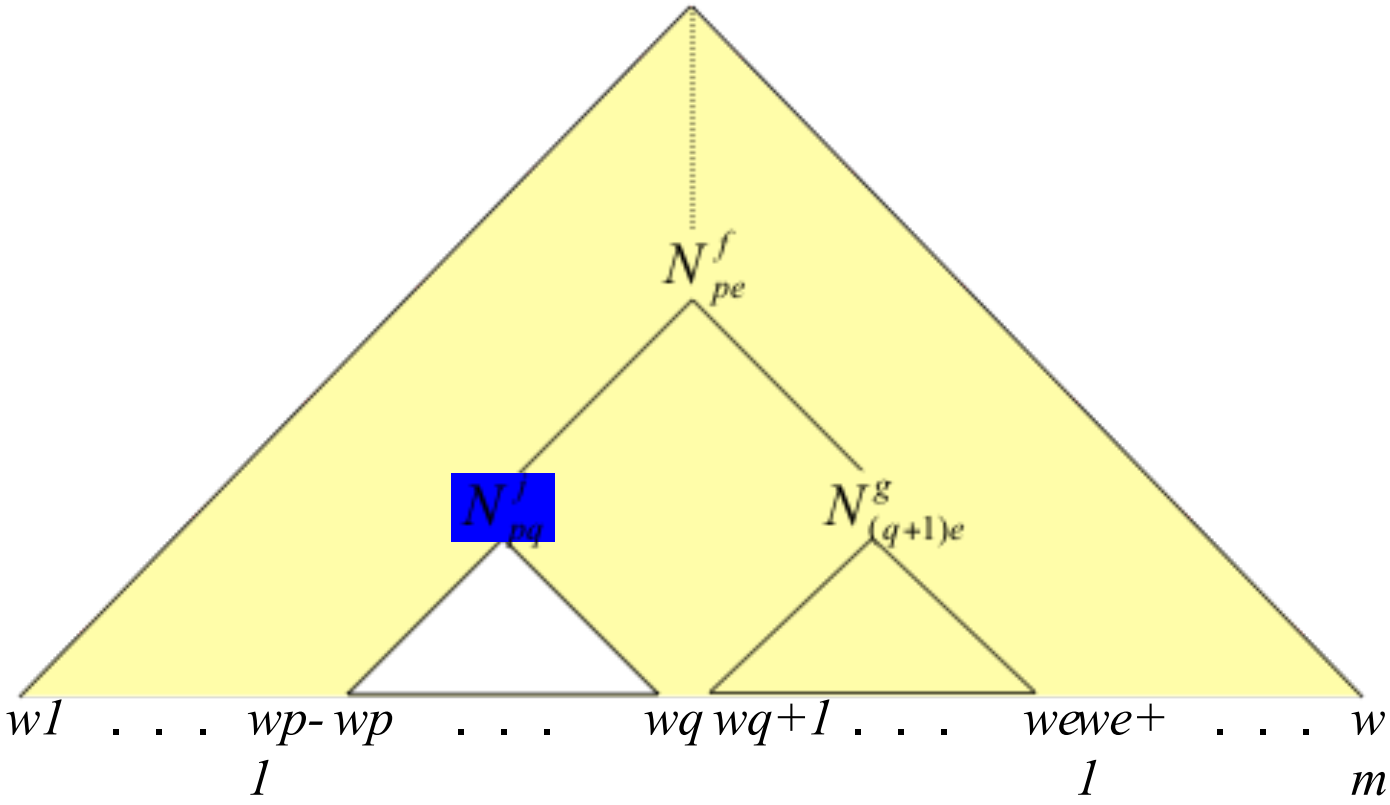
Outside probabilities: decomposing the problem

Step 1: We assume that N^f_{pe} is the parent of N^j_{pq}. Its outside probability, $\alpha_f(p,e)$, (represented by the yellow shading) is available recursively. How do we calculate the cross-hatched probability?



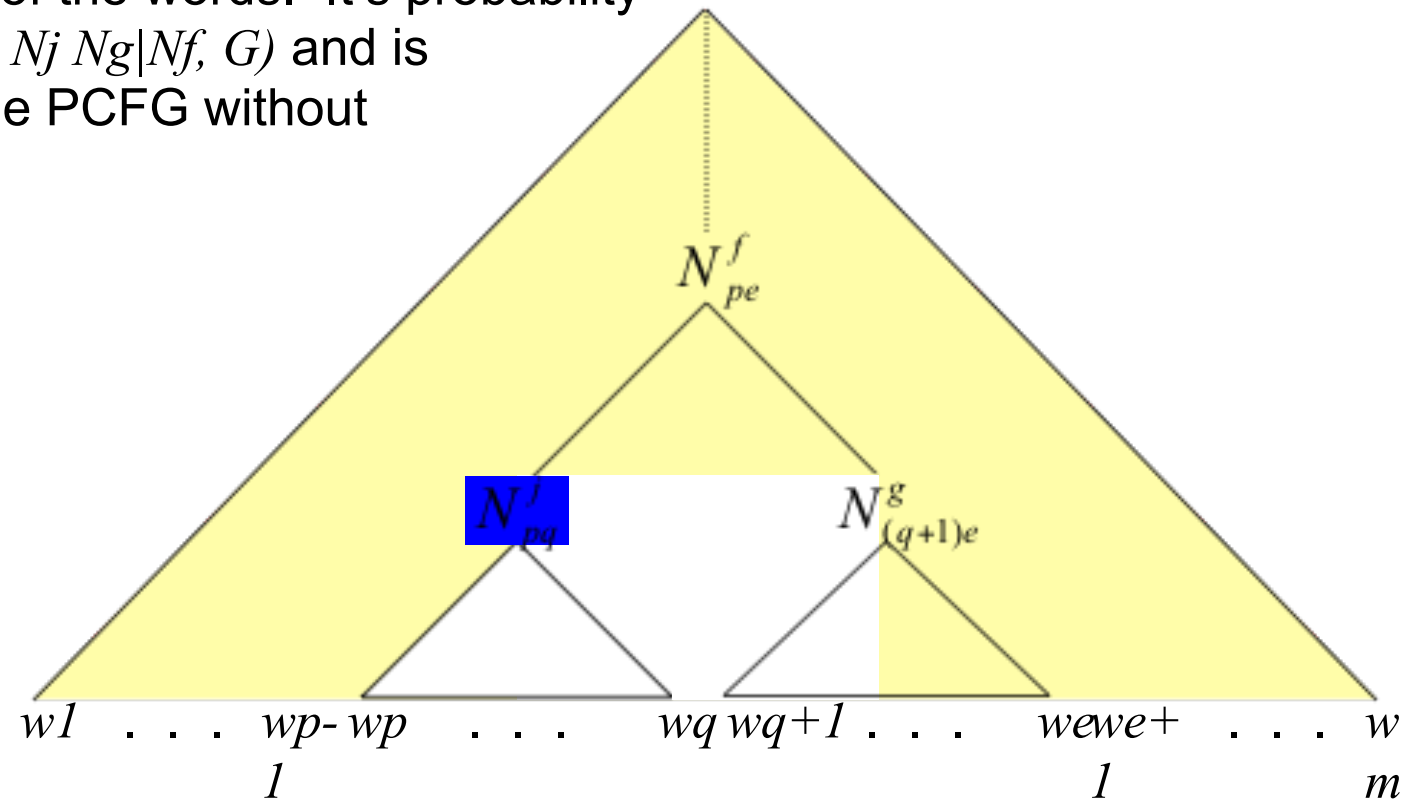
Outside probabilities: decomposing the problem

Step 2: The red shaded area is the inside probability of $N^{g}_{(q+1)e}$, which is available as $\beta_g(q+1,e)$.



Outside probabilities: decomposing the problem

Step 3: The blue shaded part corresponds to the production $Nf \rightarrow Nj\ Ng$, which because of the context-freeness of the grammar, is not dependent on the positions of the words. It's probability is simply $P(Nf \rightarrow Nj\ Ng|Nf, G)$ and is available from the PCFG without calculation.



Inside-Outside

- Inside probabilities are required to compute Outside probabilities
- Inside-Outside works where Forward-Backward does, but not vice-versa
- Implementation considerations
  - Building a hypergraph explicitly simplifies code, but it can be expensive in terms of memory