Hidden Markov Models

02-251

Based in part on Chapter 11 of Jones & Pevzner, *An Introduction to Bioinformatics Algorithms*

How could you simulate an author?

- Different writers have different styles, sentence structure, word choice, etc.

WRITING ASSIGNMENTS Three major essays (5-10 pages each) will be assigned at intervals over the semester. For each essay, I'll provide a couple of suggested topics, but you can also work up your own (if you do, I recommend that you bounce your topic off me in advance). You'll have at least ten days to do each essay, and it's expected that these papers will represent your very best, most careful and considered work. I will schedule time for optional conferences before each essay is due. If you can make a reasonable case for it, I will permit you to revise either the first or the second major essay; the final grade for that essay will then become the average of the initial grade and the revision grade. (N.B. "Revise" does not mean merely fixing the typos or clunkers that I marked on the first version.)

There will also be 11 unannounced in-class writing assignments (a.k.a. mini-papers) of 1-2 handwritten pages each. Mini-papers, which I will not mark on much,* will be graded either S (= "Satisfactory") or U (= "Unsatisfactory"). In order to qualify for an S, a mini-paper must be written in class at the time it's assigned; there is no way to make up a missed mini-paper even if you have an excused absence on the relevant class day. Nor can mini-papers be revised for credit. At the term's end, a composite grade for all your mini-papers will be calculated as follows: 10 papers graded S will result in the numerical equivalent of an A **; 9 Ss will = A-; 8 Ss will = B+; 7 Ss = B; 6 Ss = B-, and so on and so forth.

COURSE RULES & PROCEDURES

(1) Attendance at each class meeting is required. An absence will be excused only if it's negotiated in advance or there's a medical emergency. If necessary, you will be permitted one free unexcused absence; each additional unexc. abs. will lower your final grade by one whole number.**
(2) You are required to do every last iota of the reading and writing assigned, exactly in the format requested, and it needs to be totally done by the time class starts. There is no such thing as "falling a little behind" in the course reading; either you've done your homework or you haven't. Chronic lack of preparation (which is easy to spot) will lower your final grade by one whole number.
(3) Even in a seminar course, it seems a little silly to *require* participation. Some students who are cripplingly shy, or who can't always formulate their best thoughts and questions in the rapid back-and-forth of a group discussion, are nevertheless good, serious students. On the other hand, as Prof. [redacted] points out *supra,* our class can't really function if there isn't student participation—it will become just me giving a half-assed ad-lib lecture for 90 minutes, which (trust me) will be horrible in all kinds of ways. There is, therefore, a small percentage of the final grade that will concern the quantity and quality of your participation in class discussions. But the truth is that I'm

---

* I'm happy to discuss a graded mini-paper with you at any time, of course.

** See the FYI about numerical grade-conversions on pp. 3-4 of the syllabus.

David Foster Wallace

Is it O.K. to be a Luddite?

Thomas Pynchon

The New York Times Book Review
28 October 1984, pp. 1, 40-41.

As if being 1984 weren't enough, it's also the 25th anniversary this year of C. P. Snow's famous Rede lecture, "The Two Cultures and the Scientific Revolution," notable for its warning that intellectual life in the West was becoming polarized into "literary" and "scientific" factions, each doomed not to understand or appreciate the other. The lecture was originally meant to address such matters as curriculum reform in the age of Sputnik and the role of technology in the development of what would soon be known as the third world. But it was the two-culture formulation that got people's attention. In fact it kicked up an amazing row in its day. To some already simplified points, further reductions were made, provoking certain remarks, name-calling, even intemperate rejoinders, giving the whole affair, though attenuated by the mists of time, a distinctly cranky look.

Today nobody could get away with making such a distinction. Since 1959, we have come to live among flows of data more vast than anything the world has seen. Demystification is the order of our day, all the cats are jumping out of all the bags and even beginning to mingle. We immediately suspect ego insecurity in people who may still try to hide behind the jargon of a specialty or pretend to some data base forever "beyond" the reach of a layman. Anybody with the time, literacy, and access fee can get together with just about any piece of specialized knowledge s/he may need. So, to that extent, the two-cultures quarrel can no longer be sustained. As a visit to any local library or magazine rack will easily confirm, there are now so many more than two cultures that the problem has really become how to find the time to read anything outside one's own specialty.

Markov Chains

- Markov Chains: a way to generate a sequence of **symbols** (i.e. words or nucleotides) randomly.
- Encode "author-specific" features via conditional probability distributions:

$$\Pr(\text{next symbol is } c_{i+1} \mid \text{current symbol is } c_i)$$

"given"

- "First-order Markov chain" because probability of next symbol only depends on the current symbol.
- English example: Pr('u' | 'q') = 0.95

Mark V Shaney

Developed by Bruce Ellis & Rob Pike in the 1980s

>From mvs Fri Nov 16 17:11 EST 1984 remote from alice
It looks like Reagan is going to say? Ummm... Oh yes, I was looking for. I'm so glad I remembered it. Yeah, what I have wondered if I had committed a crime. Don't eat with your assessment of Reagon and Mondale. Up your nose with a guy from a firm that specifically researches the teen-age market. As a friend of mine would say, "It really doesn't matter"... It looks like Reagan is holding back the arms of the American eating public have changed dramatically, and it got pretty boring after about 300 games.

People, having a much larger number of varieties, and are very different from what one can find in Chinatowns across the country (things like pork buns, steamed dumplings, etc.) They can be cheap, being sold for around 30 to 75 cents apiece (depending on size), are generally not greasy, can be adequately explained by stupidity. Singles have felt insecure since we came down from the Conservative world at large. But Chuqui is the way it happened and the prices are VERY reasonable.

Can anyone think of myself as a third sex. Yes, I am expected to have. People often get used to me knowing these things and then a cover is placed over all of them. Along the side of the $$ are spent by (or at least for) the girls. You can't settle the issue. It seems I've forgotten what it is, but I don't. I know about violence against women, and I really doubt they will ever join together into a large number of jokes. It showed Adam, just after being created. He has a modem and an autodial routine. He calls my number 1440 times a day. So I will conclude by saying that I can well understand that she might soon have the time, it makes sense, again, to get the gist of my argument, I was in that (though it's a Republican administration).
_-_-_-_-Mark

Generated from a k-th order Markov chain

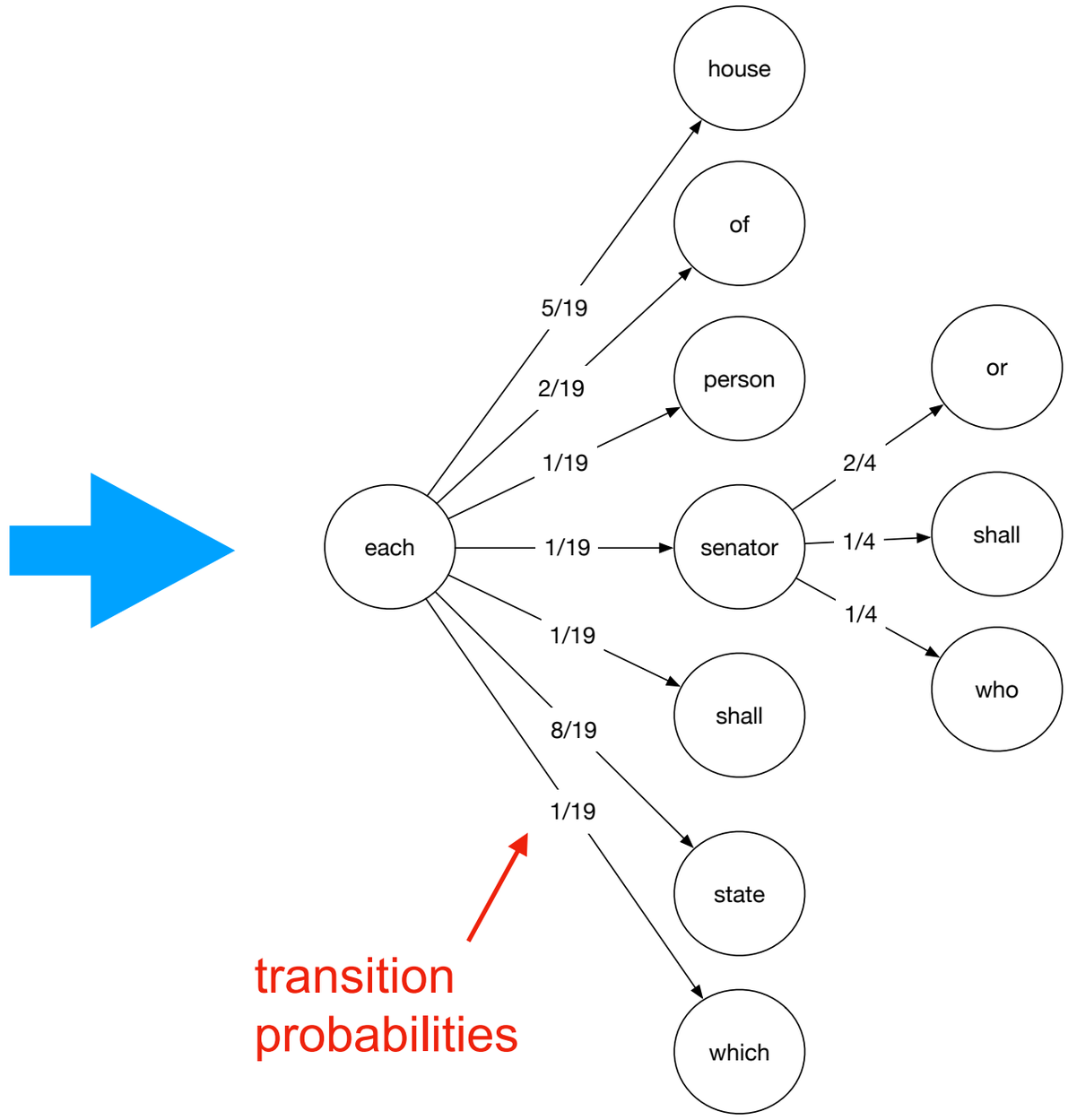
Q: how could you get the probabilities?

- "I spent an interesting evening recently with a grain of salt."
- "I hope that there are sour apples in every bushel."

(Wikipedia, 2011)

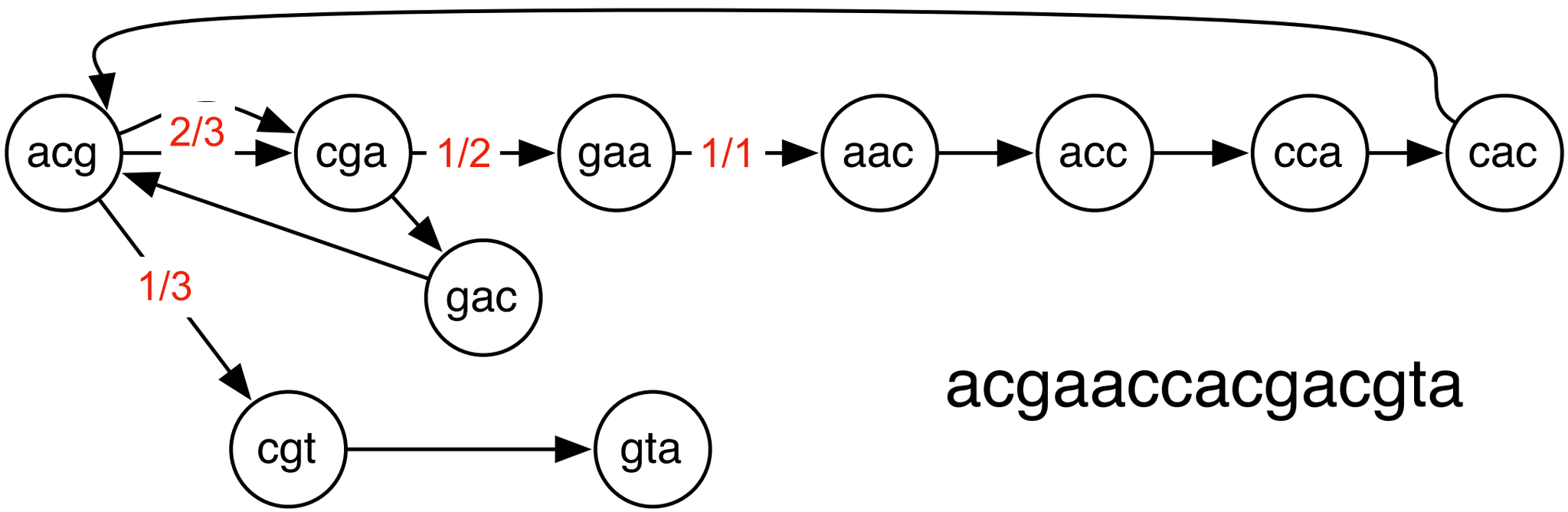
Markov Chains as Graphs

```
each house 5
each of 2
each person 1
each senator 1
each shall 1
each state 8
each which 1
…
senator or 2
senator shall 1
senator who 1
```




```
awk '{print $0}' RS=' ' uscon.txt | tr -d '[:punct:]' | tr '[:upper:]' '[:lower:]' | sed '/^ *$/d' | awk '{T[prev,$0]++;
prev=$0} END {for ( a in T ) {print a,T[a]}}' SUBSEP=' ' | sort -k1
```

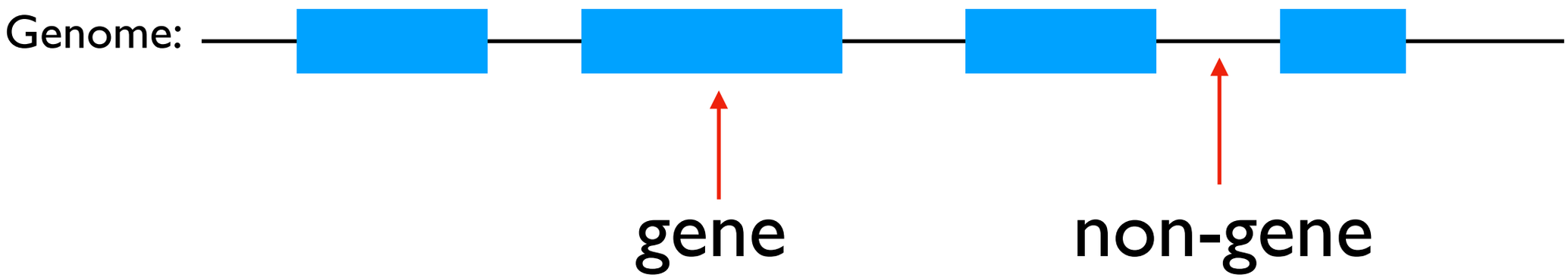
De Bruijn Graphs As Markov Chains



- De Bruijn graph of order k → k-th order Markov chain

Hidden Markov Models

- Often the generated sequence depends on something other than the previous symbols.
- E.g. Bacterial genomes:



- Genes and non-genes have different proportions of A,C,G,T (genes often have more Gs and Cs than non-genes)
- Need some way to encode whether you are in a gene or not.

Example: Checking a Casino

Fair coin:
Pr(Heads) = 0.5

Biased coin:
Pr(Heads) = 0.75

?

Suppose either a fair or biased coin was used to generate a sequence of heads & tails. But we don't know which type of coin was actual used.

Heads/Tails: ↑ ↑ ↓ ↓ ↓ ↓ ↑ ↑ ↑ ↑ ↓ ↑ ↓ ↑ ↓ ↑ ↓ ↑

Generated character depends on which coin is used.

How could we guess which coin was more likely?

Compute the Probability of the Observed Sequence

Fair coin: Pr(Heads) = 0.5
Biased coin: Pr(Heads) = 0.75

x = ↑ ↑ ↓ ↓ ↓ ↓ ↑

$$\Pr(x \mid \text{Fair}) = 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 = 0.5^7 = 0.0078125$$

$$\Pr(x \mid \text{Biased}) = 0.75 \times 0.75 \times 0.25 \times 0.25 \times 0.25 \times 0.25 \times 0.75 = 0.001647949$$

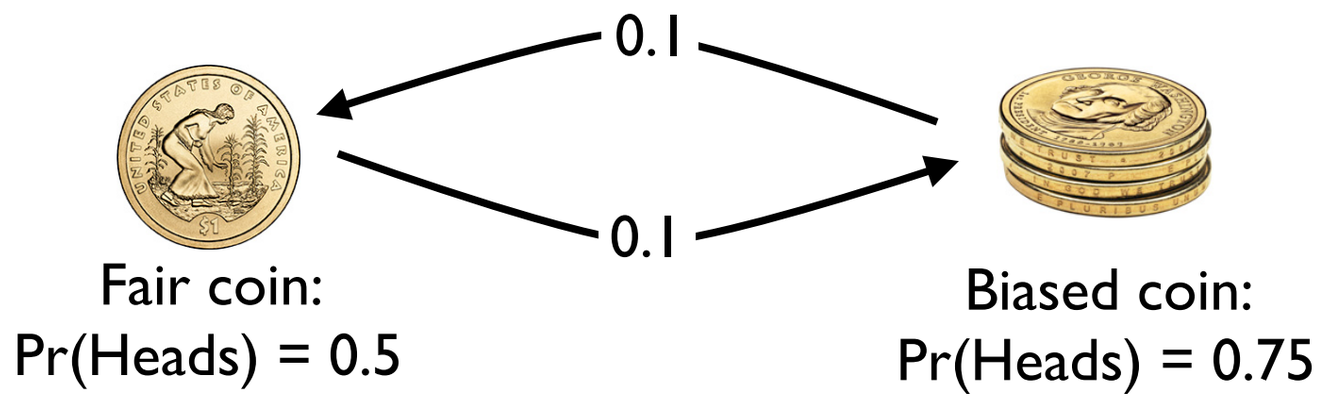
The *log-odds score*:

$$\log_2 \frac{\Pr(x \mid \text{Fair})}{\Pr(x \mid \text{Biased})} = \log_2 \frac{0.0078}{0.0016} = 2.245$$

← > 0. Hence "Fair" is a better guess.

What if the casino switches coins mid-sequence?

Fair coin: Pr(Heads) = 0.5
Biased coin: Pr(Heads) = 0.75
Probability of switching coins = 0.1

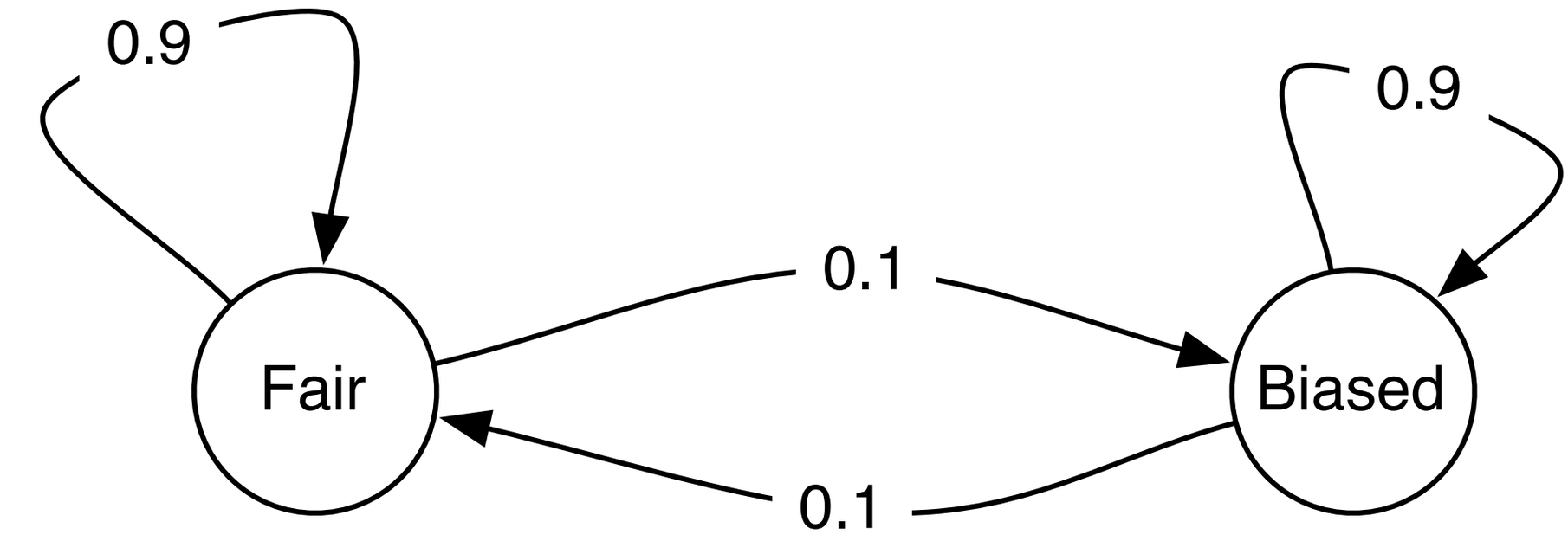


How can we compute the probability of the entire sequence?

How could we guess which coin was more likely **at each position**?

Hidden Markov Model (HMM)

Fair coin: Pr(Heads) = 0.5
Biased coin: Pr(Heads) = 0.75
Probability of switching coins = 0.1



Fair	0.5	0.5
	H	T

Biased	0.75	0.25
	H	T

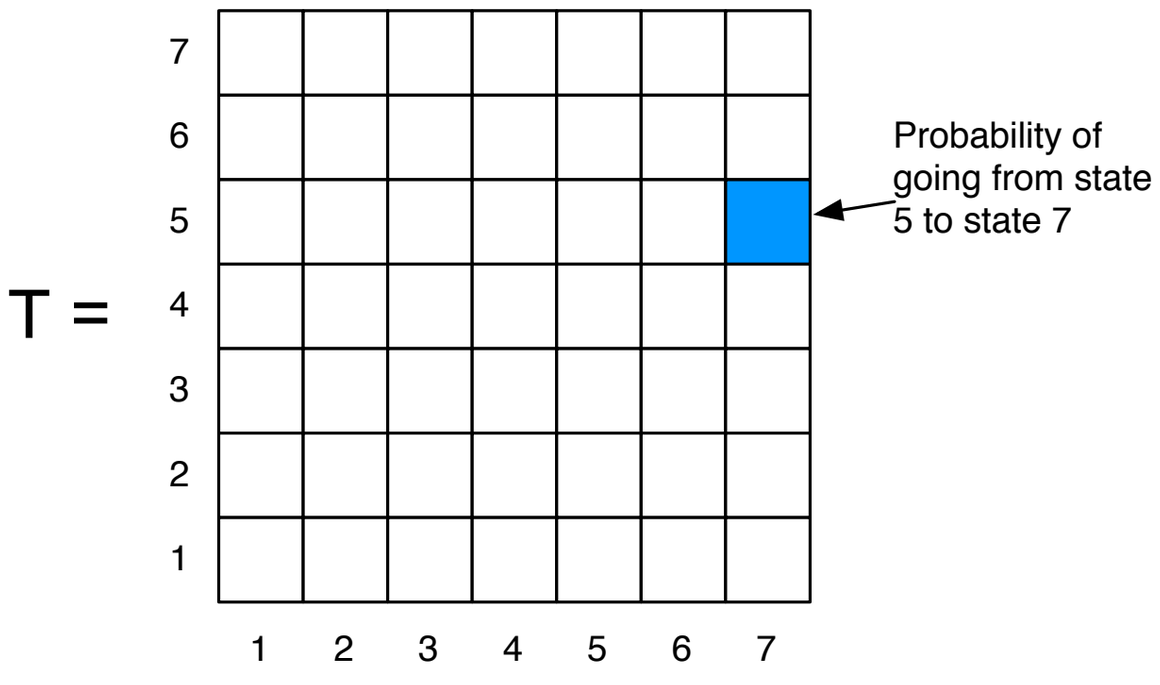
Formal Definition of a HMM

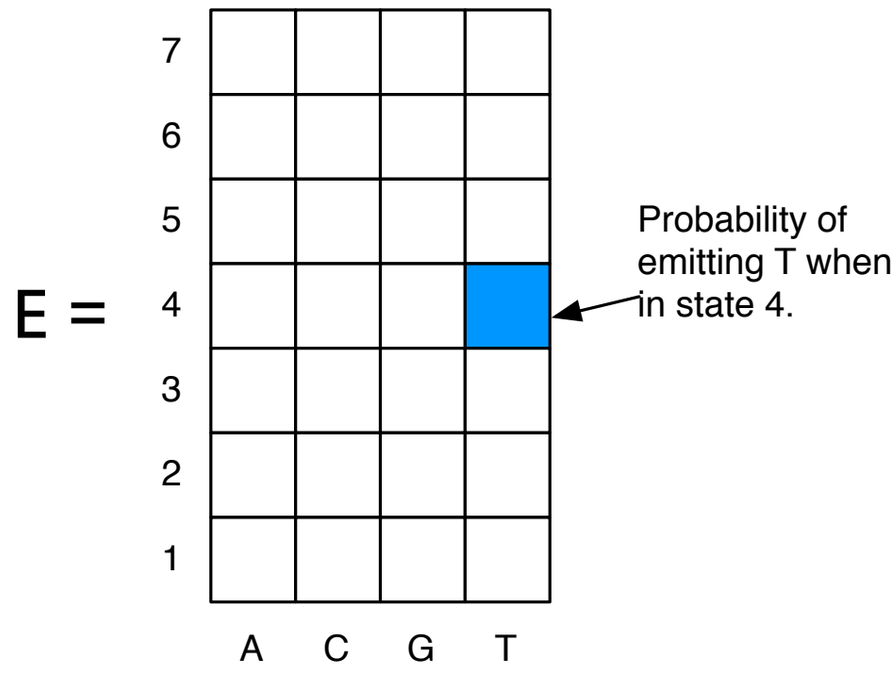
∑ = alphabet of symbols.

Q = set of states.

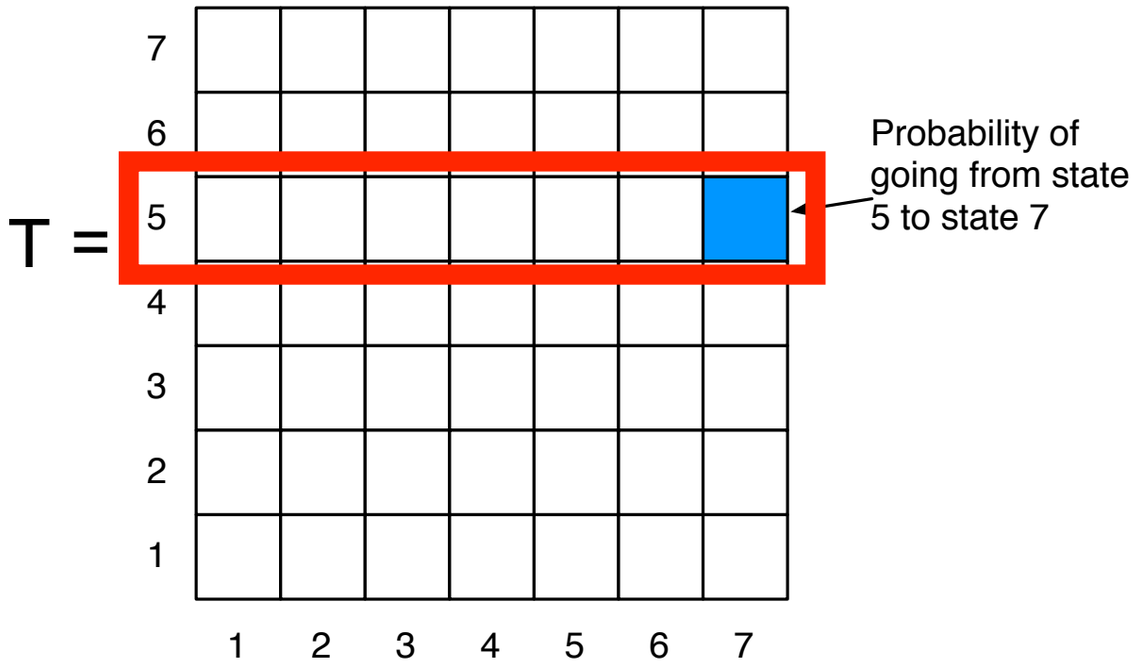
T = an |Q| x |Q| matrix where entry (k,l) is the probability of moving from state k to state l.

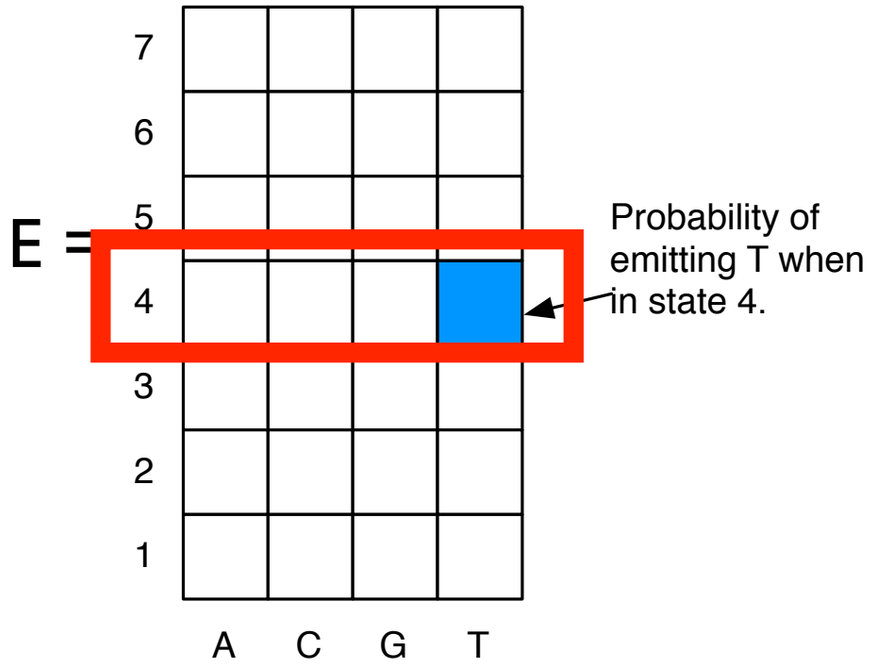
E = a |Q| x |∑| matrix, where entry (k,b) is the probability of emitting b when in state k.





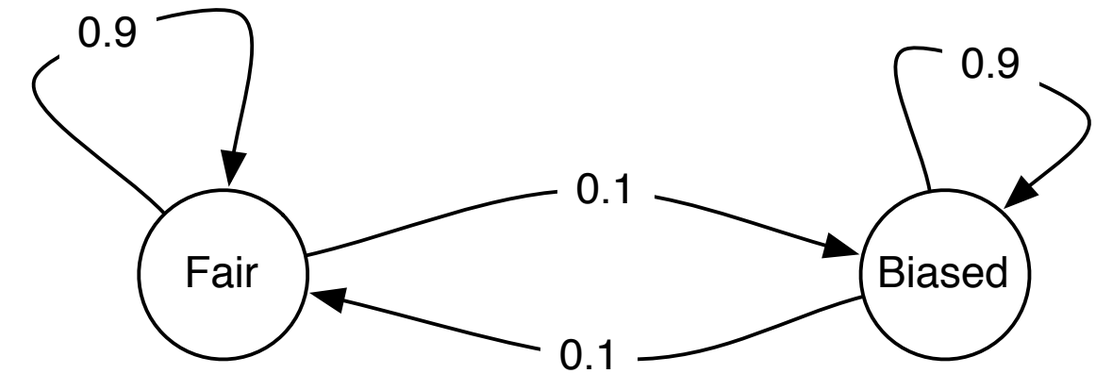
Constraints on A and E





Sum of the # in each row must be 1.

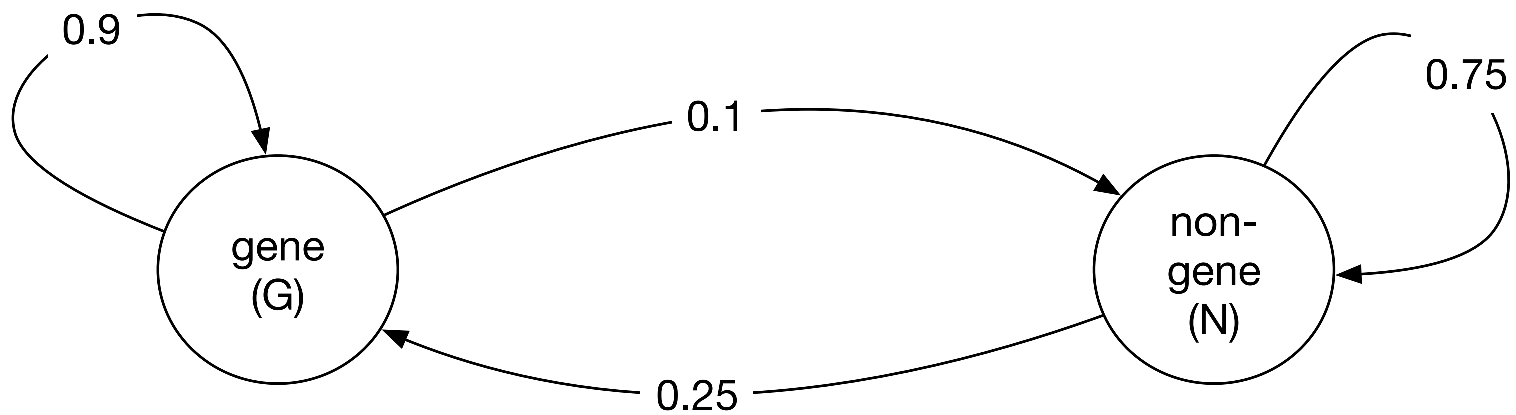
Computing Probabilities Given a Path



	H	T
Fair	0.5	0.5
Biased	0.75	0.25

$x =$	↓	↑	↓	↑	↑	↑	↓	↑	↑	↓
$\pi =$	F	F	F	B	B	B	B	F	F	F
$Pr(x_i \mid \pi_i) =$	0.5	0.5	0.5	0.75	0.75	0.75	0.25	0.5	0.5	0.5
$Pr(\pi_i \rightarrow \pi_{i+1}) =$	0.1	0.9	0.9	0.1	0.9	0.9	0.9	0.1	0.1	0.1

HMMs for Gene Finding



(simple HMM, won't work well in practice)

gene (G)	non-gene (N)
Pr[A] = 0.15	Pr[A] = 0.25
Pr[T] = 0.15	Pr[T] = 0.25
Pr[G] = 0.35	Pr[G] = 0.25
Pr[C] = 0.35	Pr[C] = 0.25

(made up probabilities)

(observed) Genome:

HMM path (hidden): NNNNGGGGGGGGNNNNNGGGGGGGGGGGGNNNNNGGGGGGGGNNNNNNGGGGGGNNNNNN

Predicted Gene Locations

Q: How can we find a high probability path to explain the observed sequence?

The Decoding Problem

Given x and π, we can compute:

- Pr(x | π): product of $Pr(x_i \mid \pi_i)$
- Pr(π): product of $Pr(\pi_i \rightarrow \pi_{i+1})$
- Pr(x, π): product of all the $Pr(x_i \mid \pi_i)$ and $Pr(\pi_i \rightarrow \pi_{i+1})$

$$\Pr(x, \pi) = \Pr(\pi_0 \rightarrow \pi_1) \prod_{i=1}^{n} \Pr(x_i \mid \pi_i) \Pr(\pi_i \rightarrow \pi_{i+1})$$

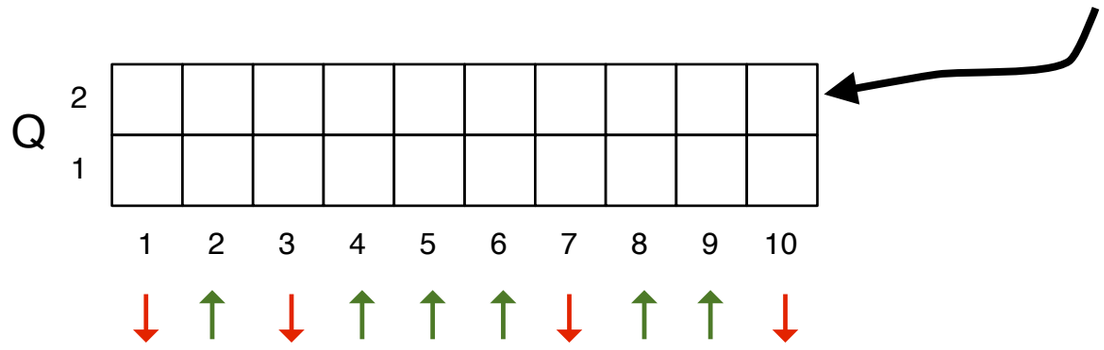
But they are "hidden" Markov models because π is unknown.

Decoding Problem: Given a sequence $x_1x_2x_3...x_n$ generated by an HMM (∑, Q, A, E), find a path π that maximizes Pr(x, π).

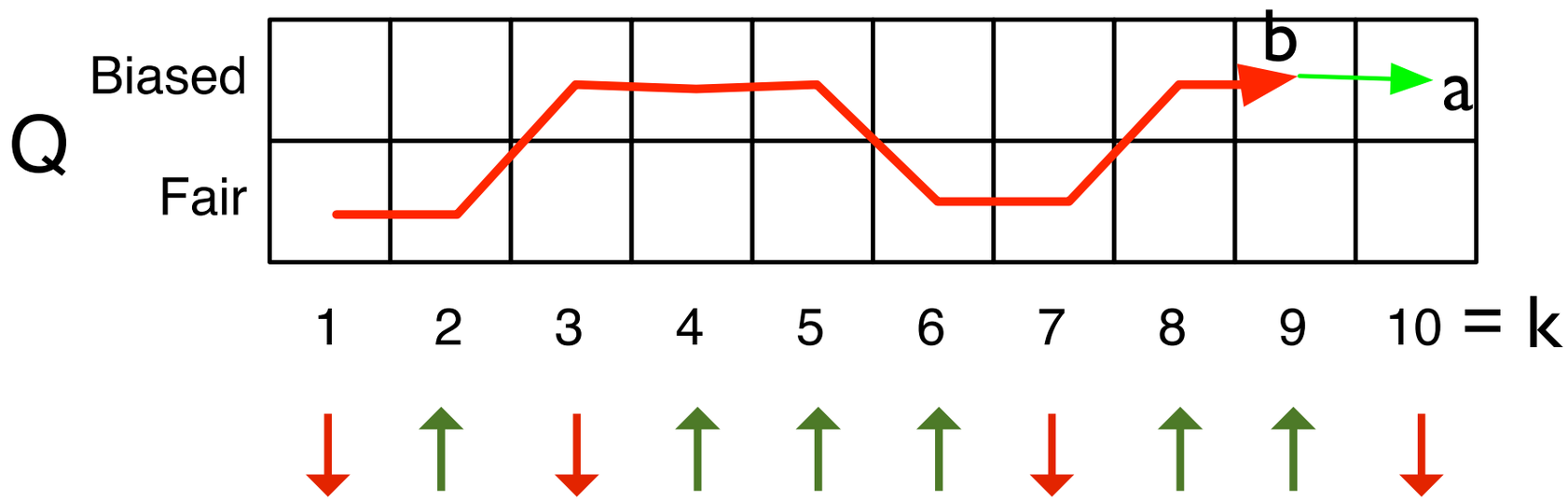
Viterbi Algorithm

The Viterbi Algorithm to Find Best Path

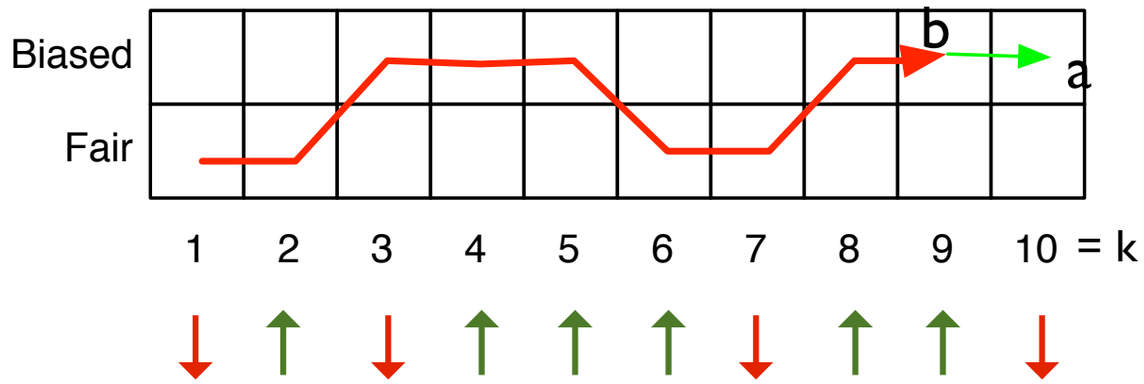
A[a, k] := the probability of the **best** path for $x_1 \ldots x_k$ that ends at state a.



A[a, k] = the cost of the path for $x_1 \ldots x_{k-1}$ that goes to some state *b* times cost of a transition from b to i, and then to output x_k from state a.



Viterbi DP Recurrence



$$A[a,k] = \max_{b \in Q} \{ \underbrace{A[b,k-1]}_{} \times \underbrace{\Pr(b \to a)}_{} \times \underbrace{\Pr(x_k \mid \pi_k = a)}_{} \}$$

Over all possible previous states.

Best path for $x_1..x_k$ ending in state b

Probability of transitioning from state b to state a

Probability of outputting x_k given that the kth state is a.

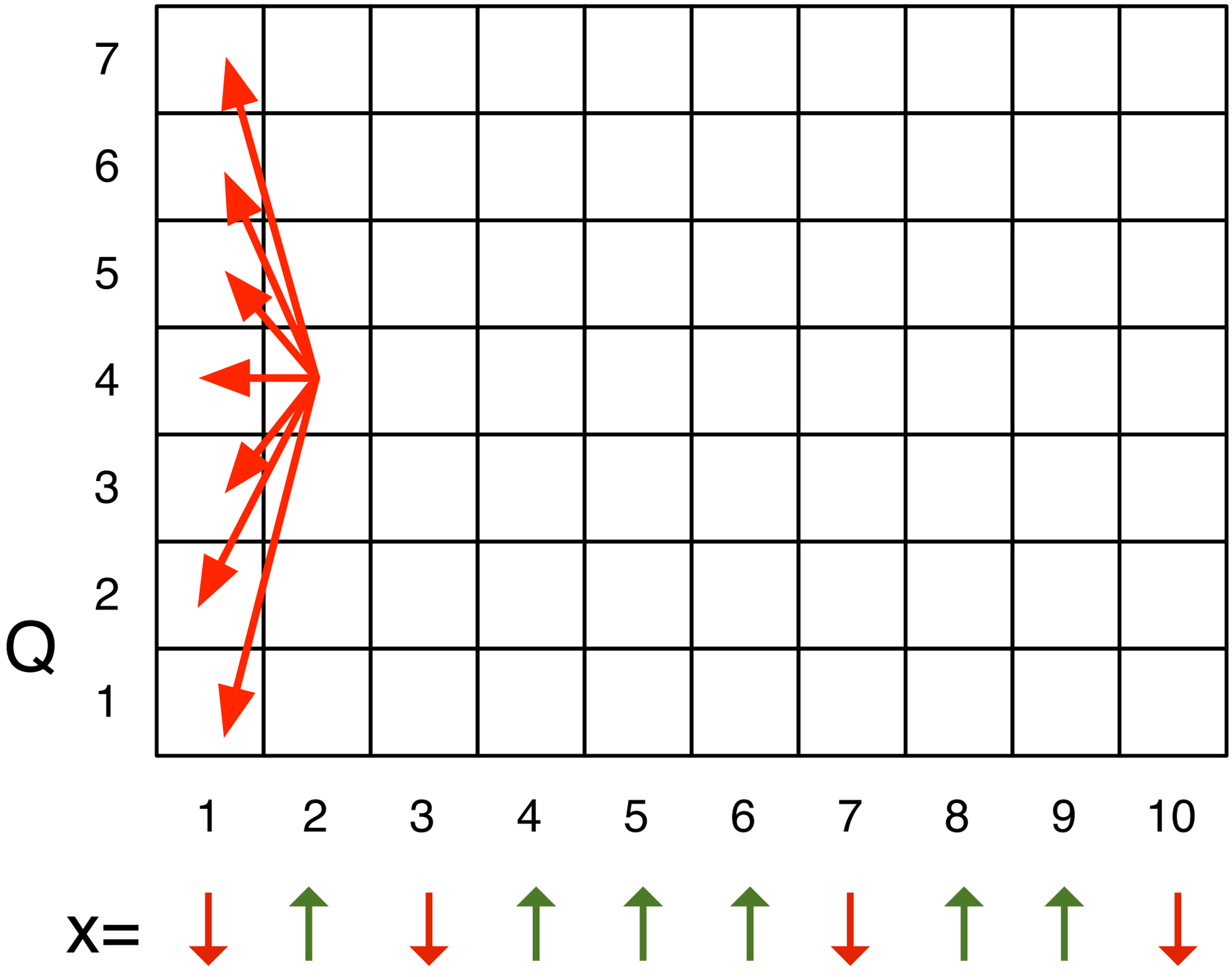
Base case:

$$A[a,1] = \underbrace{\Pr(\pi_1 = a)}_{} \times \underbrace{\Pr(x_1 \mid \pi_1 = a)}_{}$$

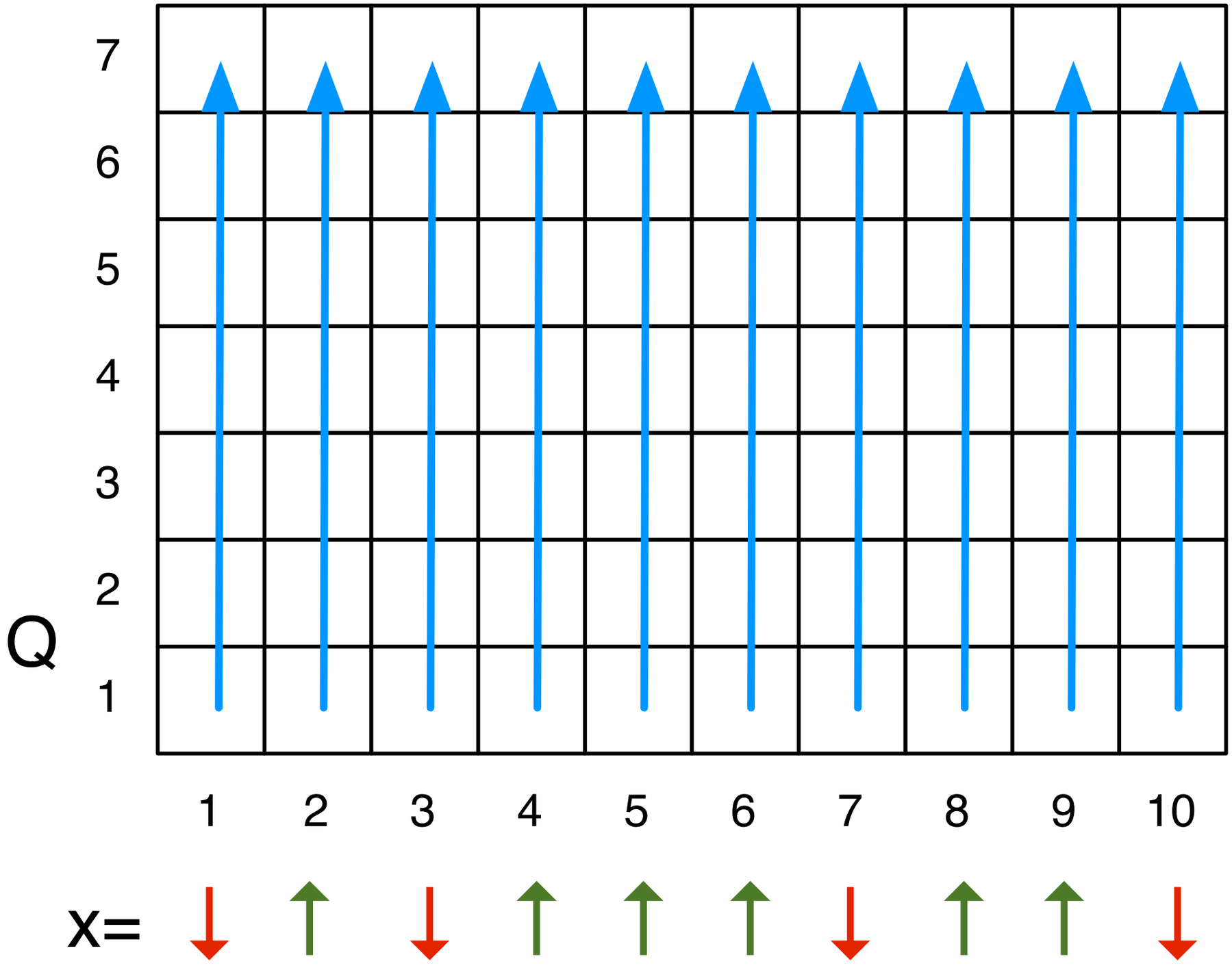
Probability that the first state is a

Probability of emitting x_1 given the first state is a.

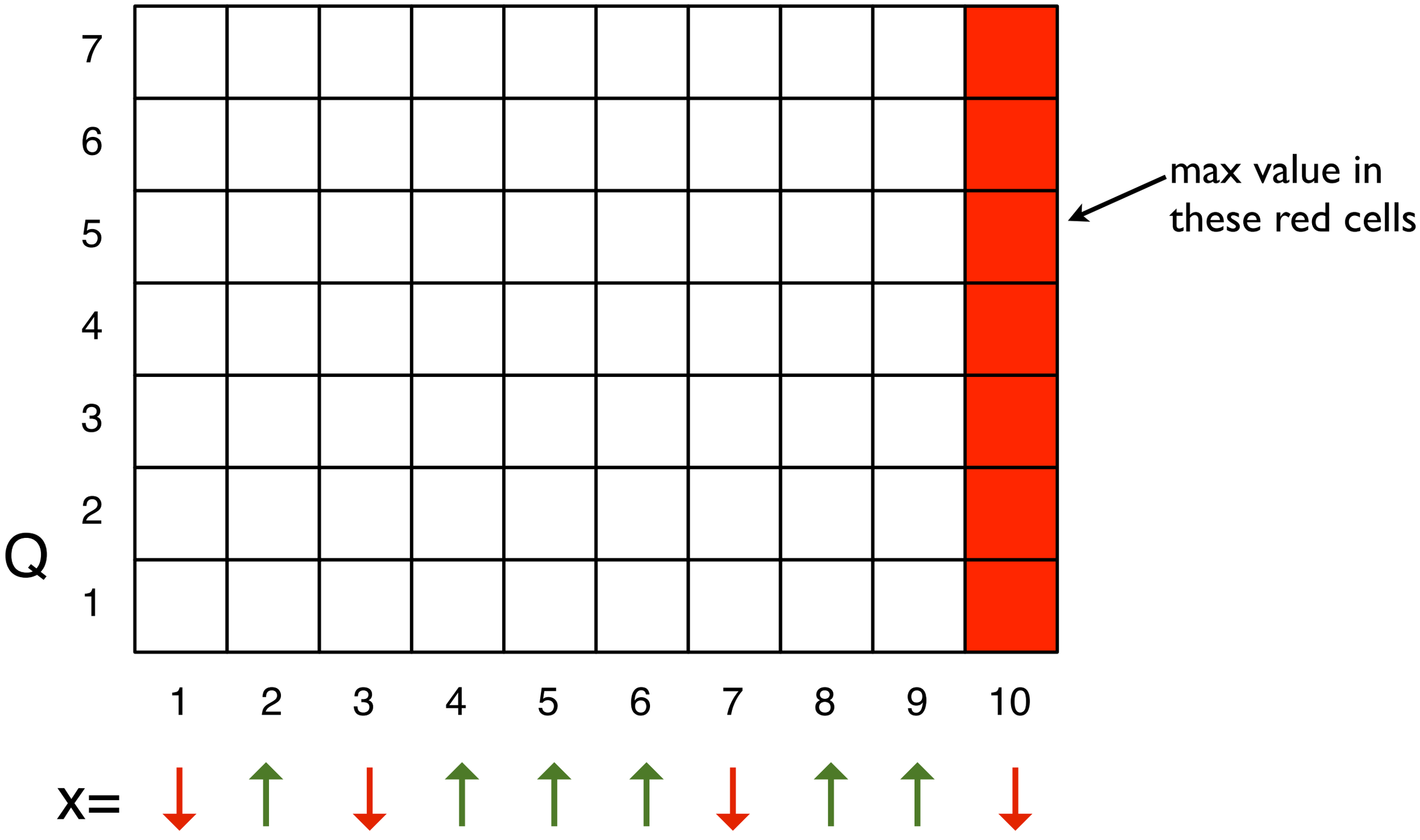
Which Cells Do We Depend On?



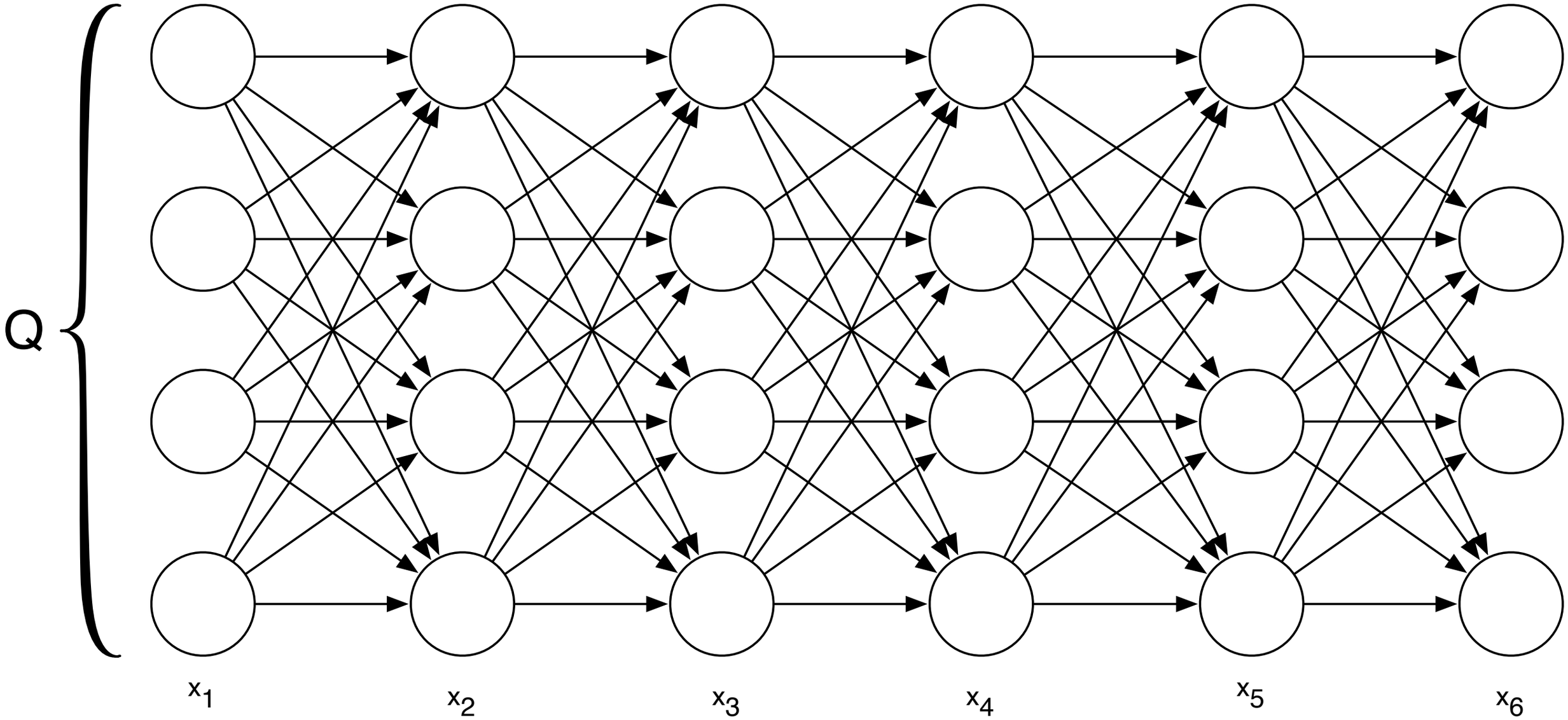
Order to Fill in the Matrix:



Where’s the answer?



Graph View of Viterbi



Viterbi is finding "shortest" path in this graph.

Running Time

- # of subproblems = O(n|Q|), where n is the length of the sequence.
- Time to solve a subproblem = O(|Q|)
- Total running time: $O(n|Q|^2)$

Using Logs

Typically, we take the log of the probabilities to avoid multiplying a lot of terms:

$$\begin{aligned}\log(A[a,k]) &= \max_{b \in Q} \{\log(A[b,k-1] \times \Pr(b \rightarrow a) \times \Pr(x_k \mid \pi_k = a))\} \\ &= \max_{b \in Q} \{\log(A[b,k-1]) + \log(\Pr(b \rightarrow a)) + \log(\Pr(x_k \mid \pi_k = a))\}\end{aligned}$$

Remember: $\log(ab) = \log(a) + \log(b)$

Why do we want to avoid multiplying lots of terms?

Multiplying leads to very small numbers:
0.1 x 0.1 x 0.1 x 0.1 x 0.1 = 0.00001
This can lead to underflow.
Taking logs and adding keeps numbers bigger.

Other Things to Compute from HMMs

Some probabilities we are interested in

What is the probability of observing a string *x* under the assumed HMM?

$$\Pr(x) = \sum_{\pi} \Pr(x, \pi)$$

What is the probability of observing *x* using a path where the i[th] state is *a*?

$$\Pr(x, \pi_i = a) = \sum_{\pi : \pi_i = a} \Pr(x, \pi)$$

What is the probability that the i[th] state is *a*?

$$\Pr(\pi_i = a | x) = \frac{\Pr(x, \pi_i = a)}{\Pr(x)}$$

The Forward Algorithm

How do we compute this:

$$\Pr(x, \pi_k = a) = \Pr(x_1, \ldots, x_i, \pi_i = a) \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = a)$$

Recall the recurrence to compute **best** path for $x_1 \ldots x_k$ that ends at state a:

$$A[a, k] = \max_{b \in Q} \{A[b, k-1] \times \Pr(b \to a) \times \Pr(x_k \mid \pi_k = a)\}$$

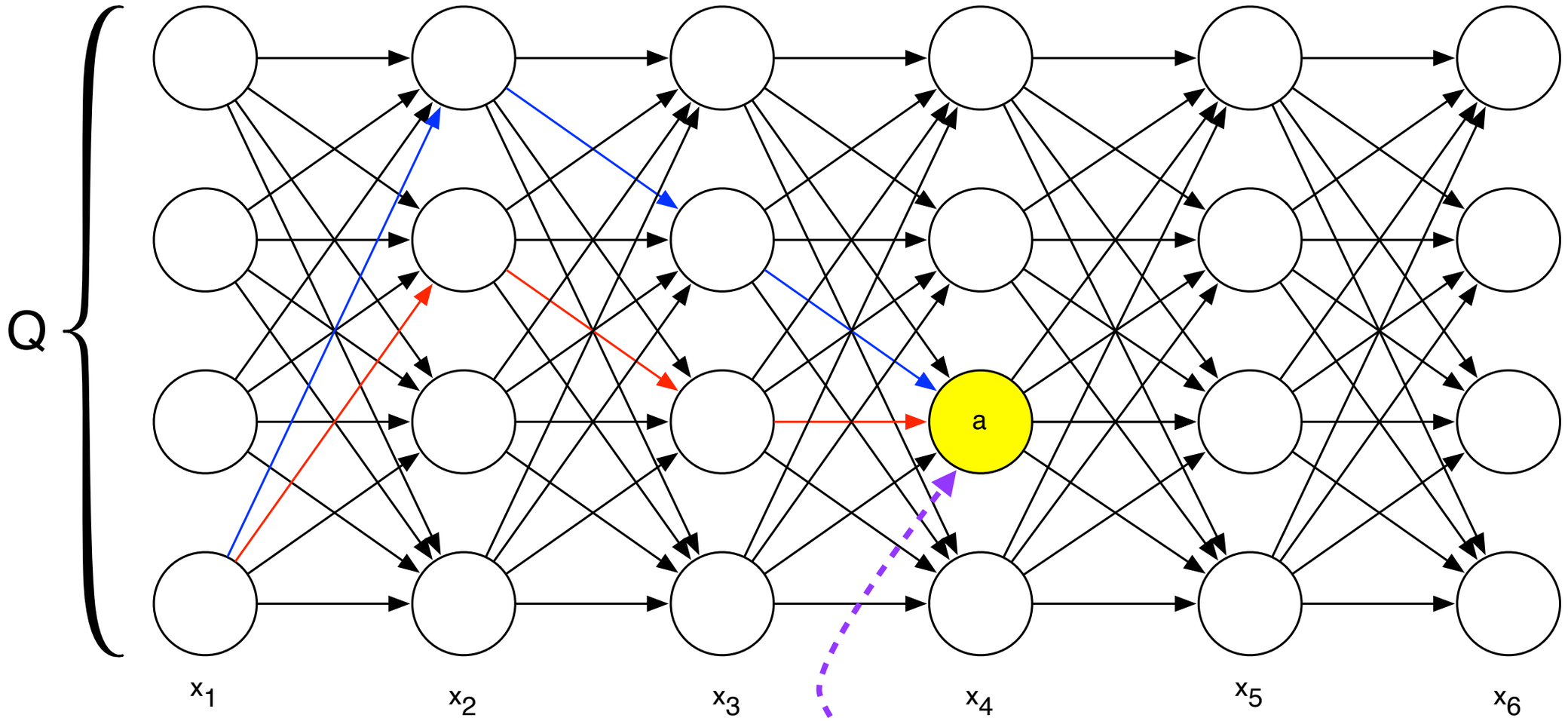
We can compute the probability of emitting $x_1, \ldots, x_k$ using **some** path that ends in *a*:

$$F[a, k] = \sum_{b \in Q} F[b, k-1] \times \Pr(b \to a) \times \Pr(x_k \mid \pi_k = a)$$

The Forward Algorithm

Computes the total probability of all the paths of length *k* ending in state *a*.

Still need to compute the probability of paths leaving *a* and going to the end.



F[a,4]

The Backward Algorithm

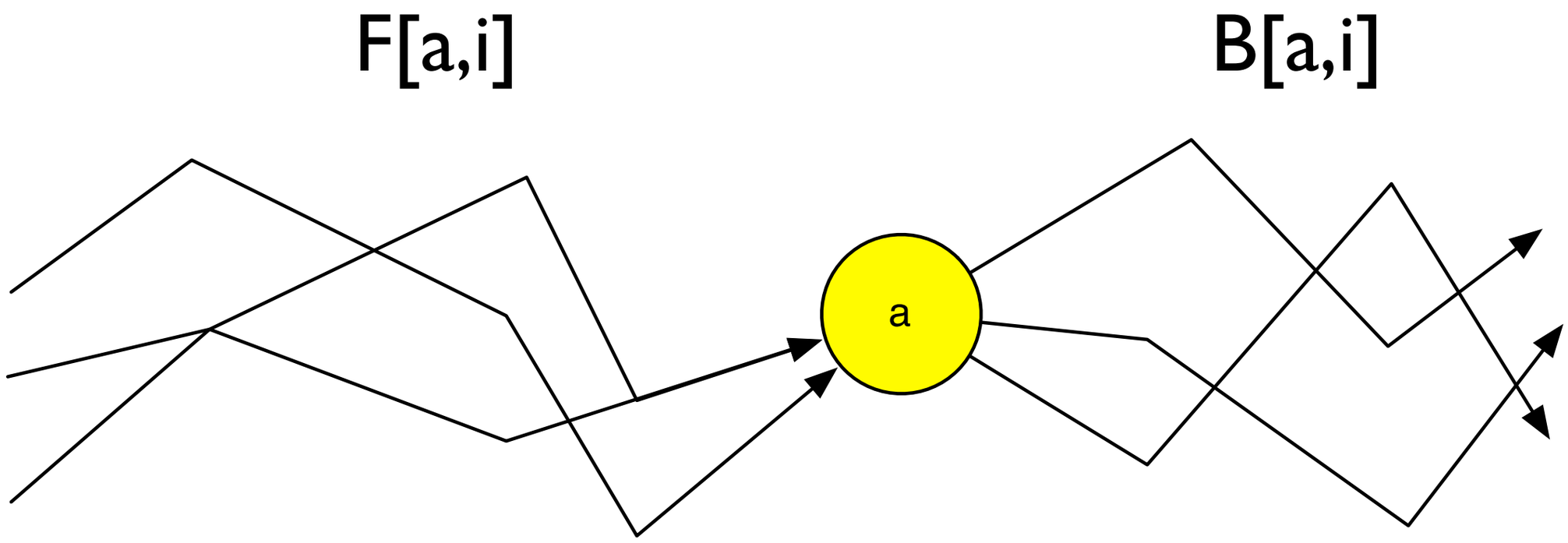
The same idea as the forward algorithm, we just start from the *end* of the input string and work towards the beginning:

B[a,k] = "the probability of generating string $x_{k+1},...,x_n$ starting from state a"

$$B[a,k] = \sum_{b \in Q} \underbrace{B[b,k+1]}_{\text{Prob for } x_{k+1}..x_n \text{ starting in state } b} \times \underbrace{\Pr(a \to b)}_{\text{Probability going from state } a \text{ to } b} \times \underbrace{\Pr(x_{k+1} \mid \pi_{k+1} = b)}_{\text{Probability of emitting } x_{k+1} \text{ given that the next state is b.}}$$

The Forward-Backward Algorithm

$$\Pr(\pi_i = a \mid x) = \frac{\Pr(x, \pi_i = k)}{\Pr(x)} = \frac{F[a,i] \cdot B[a,i]}{\Pr(x)}$$



Learning HMM probabilities

Estimating HMM Parameters

$$(\mathbf{x}^{(1)},\boldsymbol{\pi}^{(1)}) = \begin{matrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & x_4^{(1)} & x_5^{(1)} & \dots & x_n^{(1)} \\ \pi_1^{(1)} & \pi_2^{(1)} & \pi_3^{(1)} & \pi_4^{(1)} & \pi_5^{(1)} & \dots & \pi_n^{(1)} \end{matrix}$$

$$(\mathbf{x}^{(2)},\boldsymbol{\pi}^{(2)}) = \begin{matrix} x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & x_4^{(2)} & x_5^{(2)} & \dots & x_n^{(2)} \\ \pi_1^{(2)} & \pi_2^{(2)} & \pi_3^{(2)} & \pi_4^{(2)} & \pi_5^{(2)} & \dots & \pi_n^{(2)} \end{matrix}$$

Training examples where outputs and paths are known.

# of times transition $a \to b$ is observed.

$$\Pr(a \to b) = \frac{A_{ab}}{\sum_{q \in Q} A_{aq}}$$

# of times x was observed to be output from state a.

$$\Pr(x \mid a) = \frac{E_{xa}}{\sum_{x \in \Sigma} E_{xq}}$$

Pseudocounts

of times transition $a \to b$ is observed.

$$\Pr(a \to b) = \frac{A_{ab}}{\sum_{q \in Q} A_{aq}}$$

# of times x was observed to be output from state a.

$$\Pr(x \mid a) = \frac{E_{xa}}{\sum_{x \in \Sigma} E_{xq}}$$

What if a transition or emission is never observed in the training data?

⇒ 0 probability

Meaning that if we observe an example with that transition or emission in the real world, we will give it 0 probability.

But it's unlikely that our training set will be large enough to observe *every* possible transition.

Hence: we take A_{ab} = (#times a → b was observed) + 1 ← "pseudocount"

Similarly for E_{xa}.

Viterbi Training

- **Problem**: typically, in the real would we only have examples of the output x, and we don't know the paths π.

Viterbi Training Algorithm:

1. Choose a random set of parameters.
2. Repeat:
    1. Find the best paths.
    2. Use those paths to estimate new parameters.

This is a local search algorithm.

The Baum-Welch algorithm is a modification of this that is often used in practice.

It is similar, but doesn't commit to a single best path for each example.

Recap

- Hidden Markov Model (HMM) model the generation of strings.
- They are governed by a string alphabet (∑), a set of states (Q), a set of transition probabilities A, and a set of emission probabilities for each state (E).
- Given a string and an HMM, we can compute:
  - The most probable path the HMM took to generate the string (Viterbi).
  - The probability that the HMM was in a particular state at a given step (forward-backward algorithm).
- Algorithms are based on dynamic programming.
- *Finding* good parameters is a much harder problem.
  The Baum-Welch algorithm is an oft-used heuristic algorithm.