Hidden Markov Models

Ben Langmead

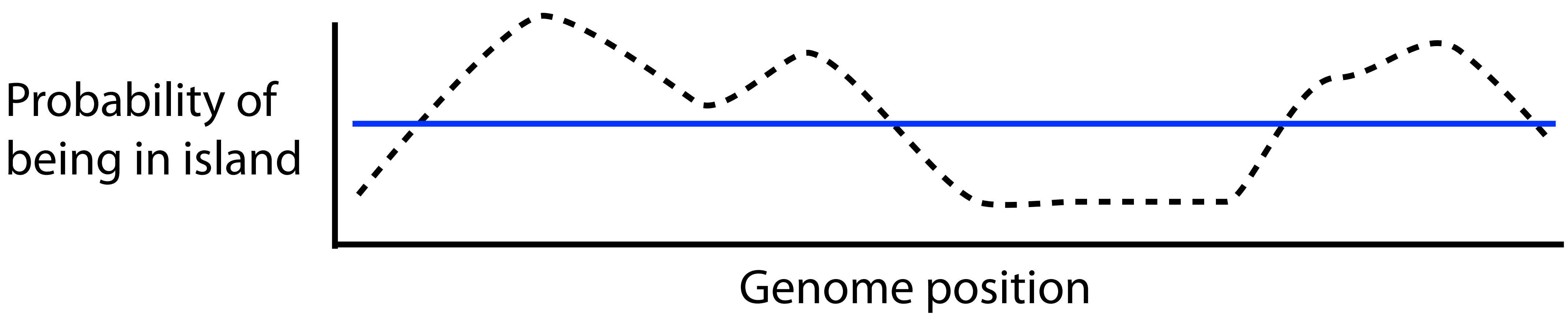
JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Department of Computer Science

Sequence models

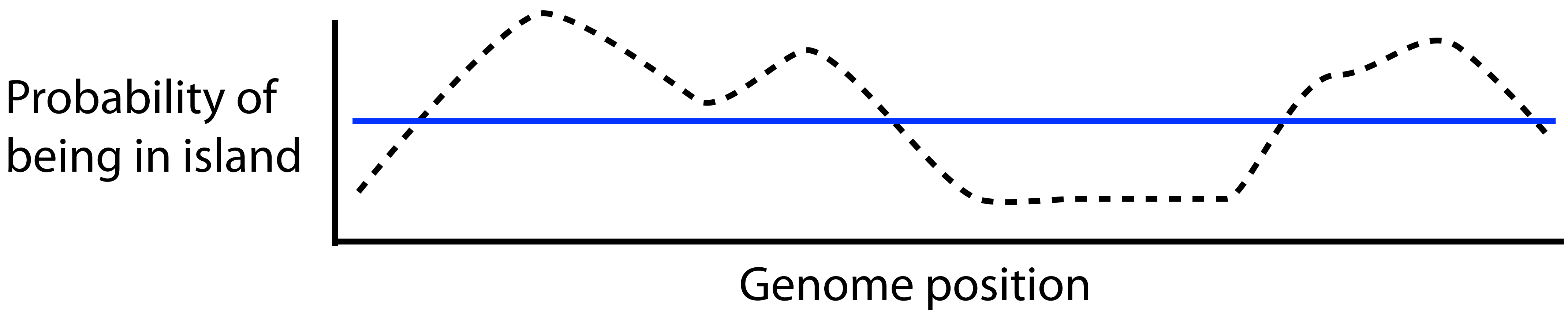
Can Markov chains find CpG islands in a “sea” of genome?

MC assigns a score to a string; doesn’t naturally give a “running” score across a long sequence



But we can adapt it using a *sliding window*

Sequence models



Choice of *k* requires assumption about island lengths

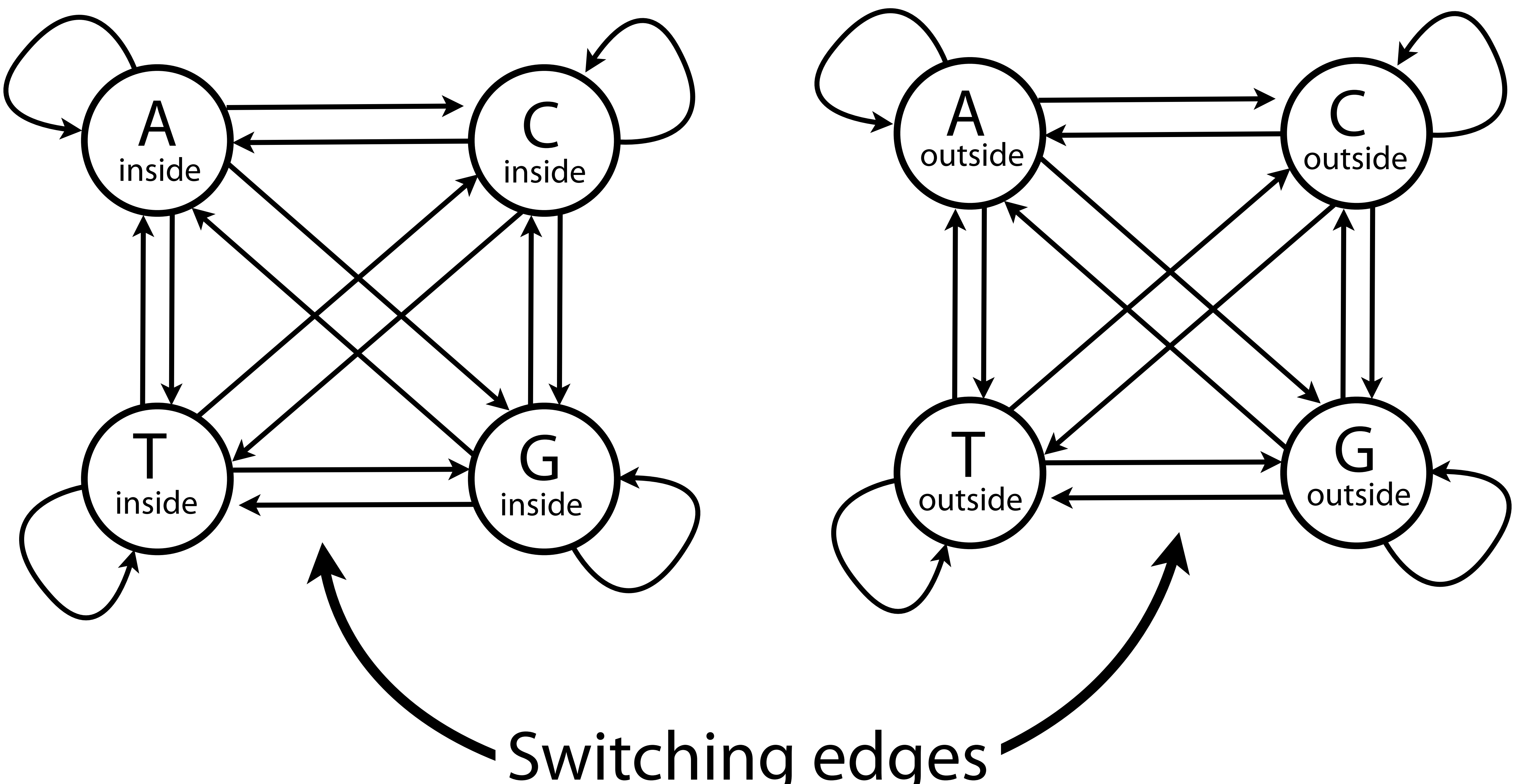
If *k* is too large, we miss small islands

If *k* is too small, we see many small islands

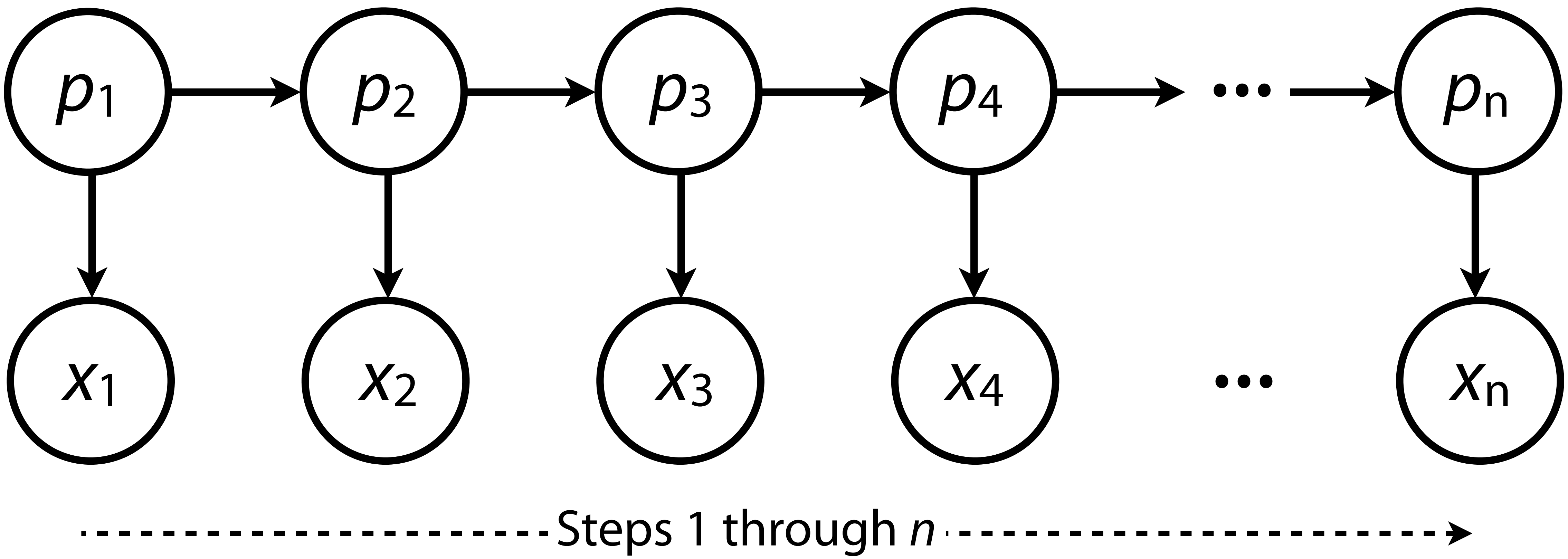
We'd like a method that *switches between Markov chains* when entering or exiting a CpG island

Sequence models

Something like this:



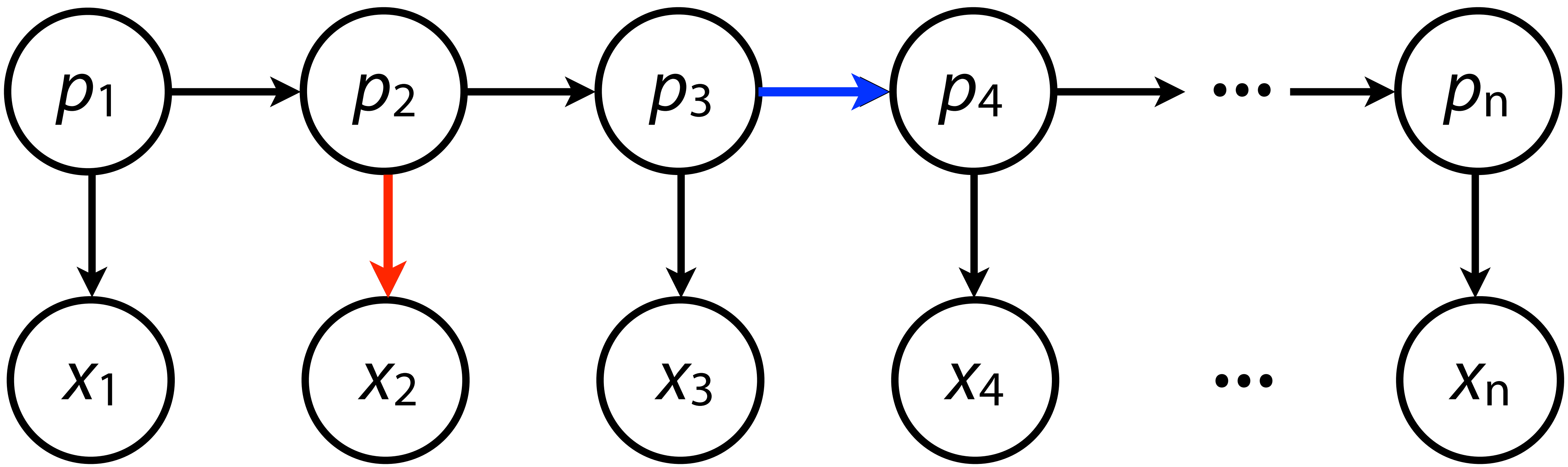
Hidden Markov Model



$p = \{ p_1, p_2, ..., p_n \}$ is a sequence of *states* (AKA a *path*). Each p_i takes a value from set Q. We ***do not*** observe p.

$x = \{ x_1, x_2, ..., x_n \}$ is a sequence of *emissions*. Each x_i takes a value from set Σ. We ***do*** observe x.

Hidden Markov Model



Edges convey conditional independence

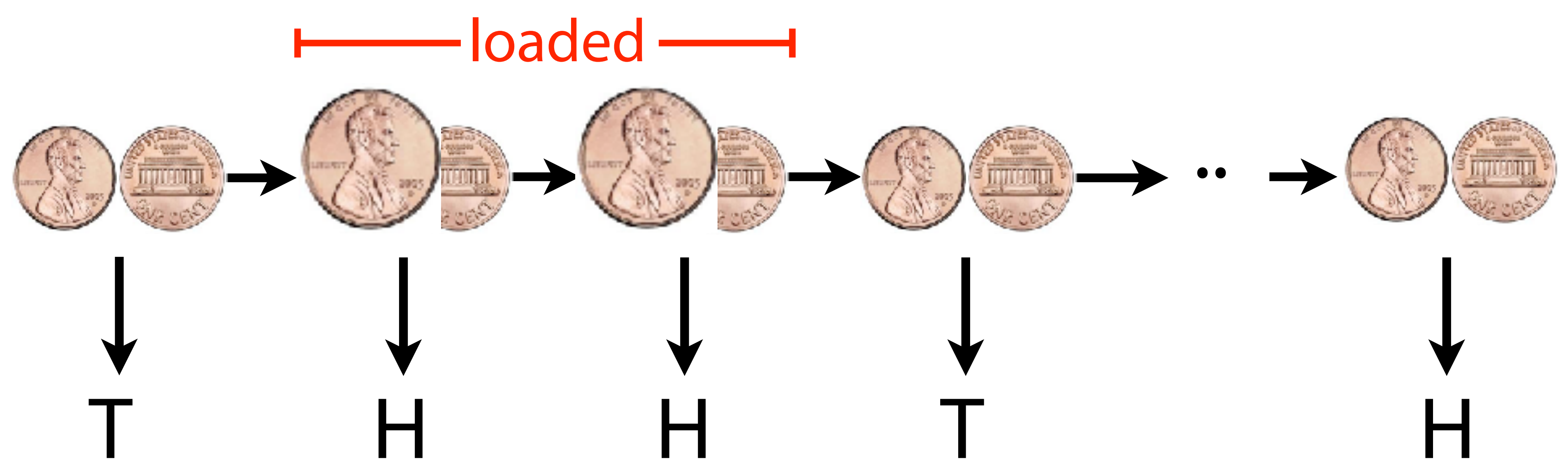
x_2 is conditionally independent of everything else given p_2

p_4 is conditionally independent of everything else given p_3

Hidden Markov Model

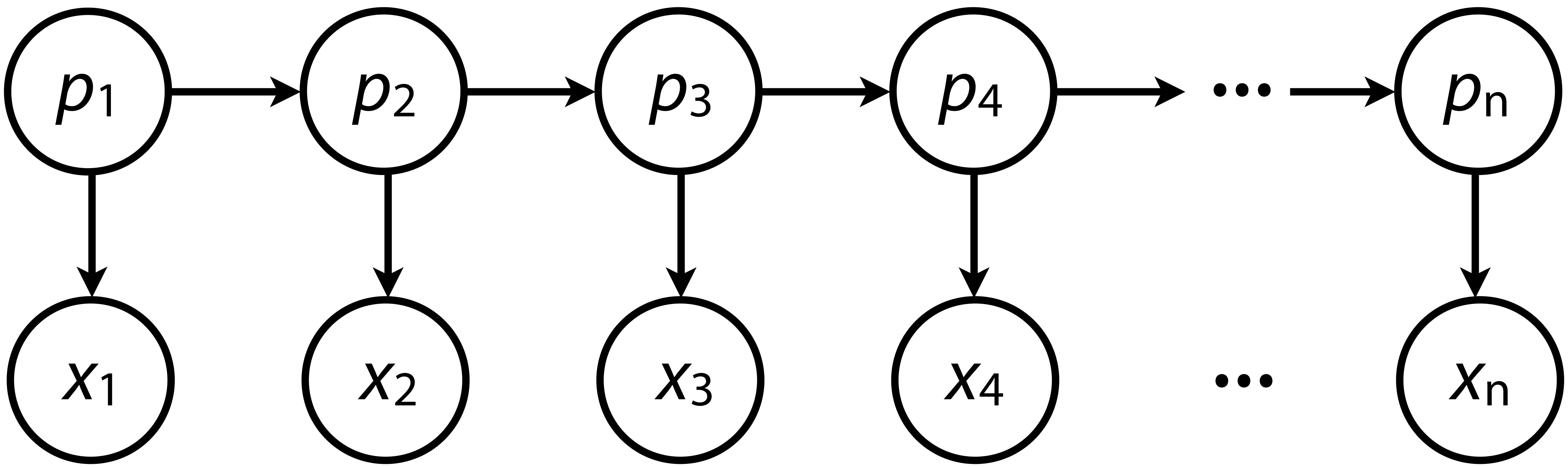
Example: occasionally dishonest casino

Dealer repeatedly flips a coin. Sometimes the coin is *fair*, with P(heads) = 0.5, sometimes it's *loaded*, with P(heads) = 0.8. Between each flip, dealer switches coins (invisibly) with prob. 0.4.



Emissions are heads/tails, states are loaded/fair

Hidden Markov Model

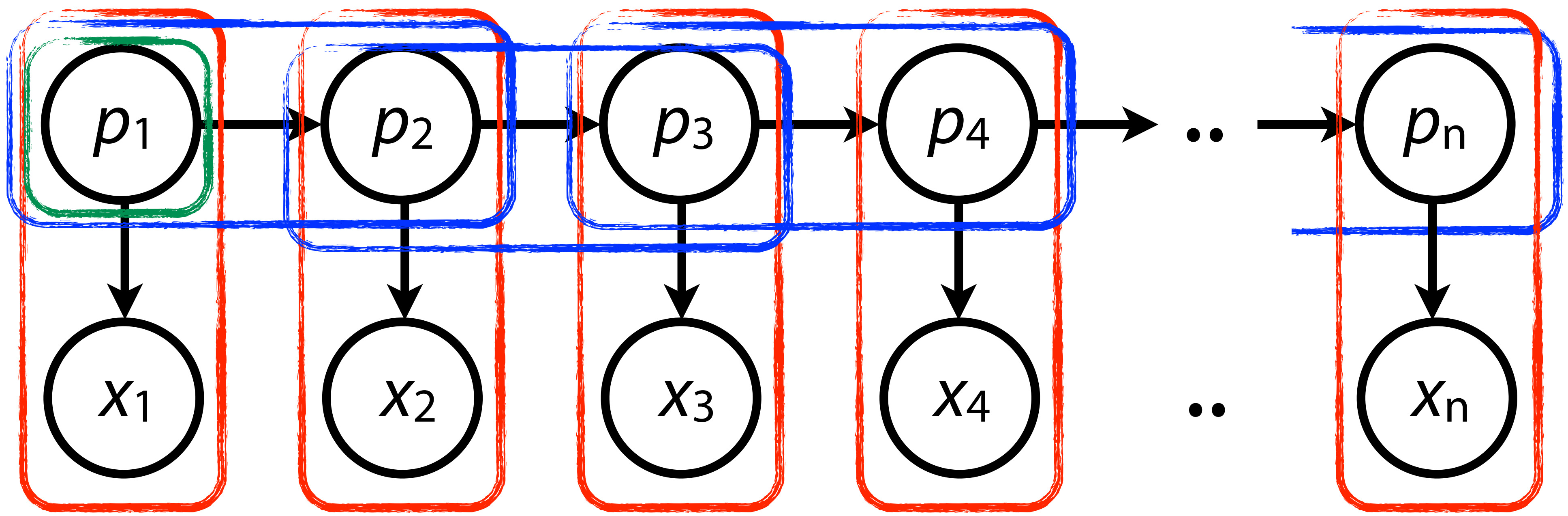


Joint probability of a given ***p***, ***x*** is easy to calculate

- Repeated applications of multiplication rule
- Simplification using Markov assumptions (implied by edges above)

Product of conditional probabilities (1 per edge), times marginal: P(p_1)

Hidden Markov Model



$$P(p_1, p_2, ..., p_n, x_1, x_2, ..., x_n) = \prod_{k=1}^{n} P(x_k \mid p_k) \prod_{k=2}^{n} P(p_k \mid p_{k-1}) \, P(p_1)$$

$|Q| \times |\Sigma|$ emission matrix E encodes $P(x_i \mid p_i)$s — $E[p_i, x_i] = P(x_i \mid p_i)$

$|Q| \times |Q|$ transition matrix A encodes $P(p_i \mid p_{i-1})$s — $A[p_{i-1}, p_i] = P(p_i \mid p_{i-1})$

$|Q|$ array I encodes initial probabilities of each state — $I[p_i] = P(p_1)$

Hidden Markov Model

Dealer repeatedly flips a coin. Coin is sometimes *fair*, with P(heads) = 0.5, sometimes *loaded*, with P(heads) = 0.8. Dealer occasionally switches coins, invisibly to you.

After each flip, dealer switches coins with probability 0.4

A:

	F	L
F	0.6	0.4
L	0.4	0.6

E:

	H	T
F	0.5	0.5
L	0.8	0.2

$|Q| \times |\Sigma|$ emission matrix E encodes P($x_i \mid p_i$)s $\quad E[p_i, x_i] = P(x_i \mid p_i)$

$|Q| \times |Q|$ transition matrix A encodes P($p_i \mid p_{i-1}$)s $\quad A[p_{i-1}, p_i] = P(p_i \mid p_{i-1})$

Hidden Markov Model

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Given *A* & *E* (right), what is the joint probability of ***p*** & ***x***?

p	**F**	**F**	**F**	**L**	**L**	**L**	**F**	**F**	**F**	**F**	**F**
x	**T**	**H**	**T**	**H**	**H**	**H**	**T**	**H**	**T**	**T**	**H**
$P(x_i \mid p_i)$	0.5	0.5	0.5	0.8	0.8	0.8	0.5	0.5	0.5	0.5	0.5
$P(p_i \mid p_{i-1})$	-	0.6	0.6	0.4	0.6	0.6	0.4	0.6	0.6	0.6	0.6

If $P(p_1 = F) = 0.5$, then joint probability $= 0.5^9\, 0.8^3\, 0.6^8\, 0.4^2 = 0.0000026874$

Hidden Markov Model

Given flip outcomes (heads or tails) and the conditional & marginal probabilities, when was the dealer using the loaded coin?

There are many possible *p*s, but one of them is *p**, the *most likely* given the emissions.

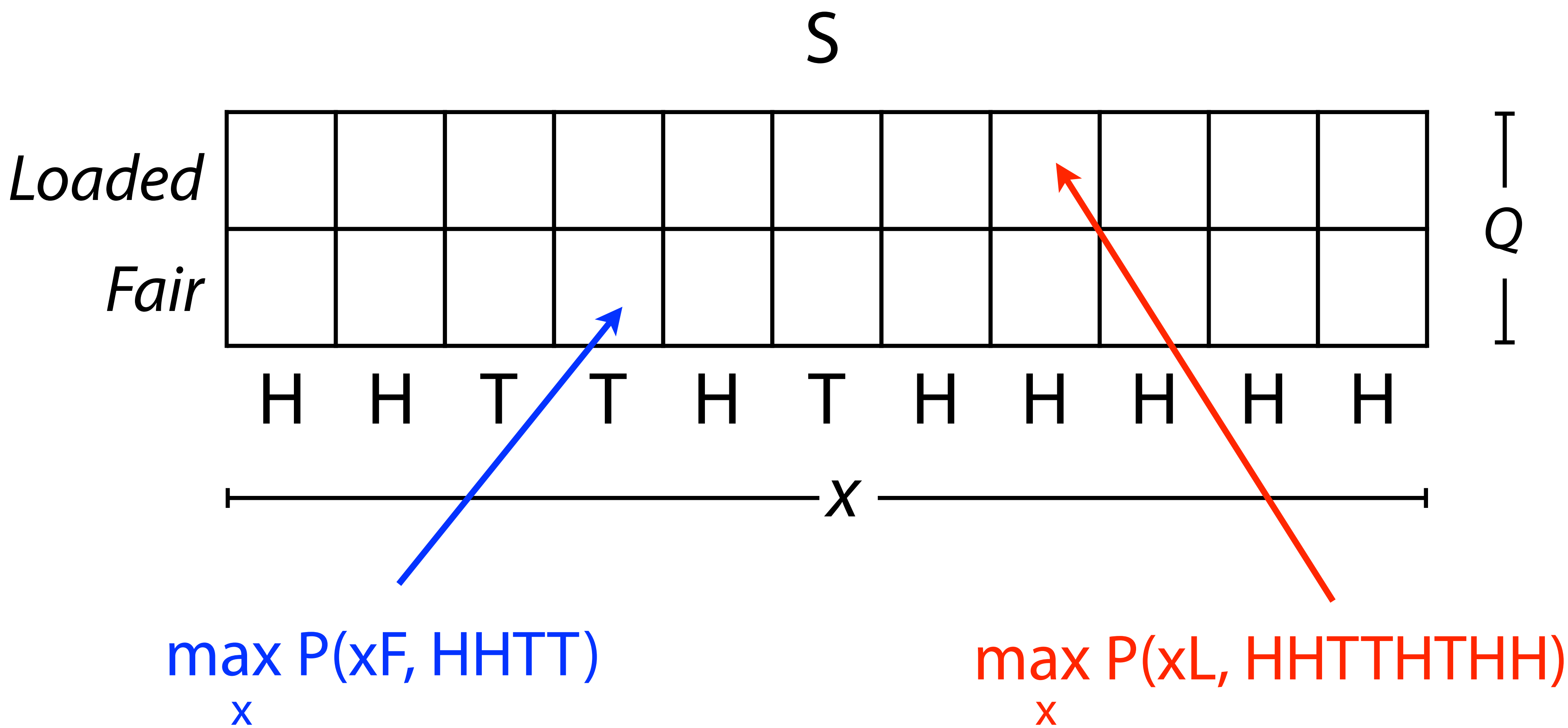
$$p^* = \underset{p}{\operatorname{argmax}}\, \mathrm{P}(\, p \mid x\,) \;=\; \underset{p}{\operatorname{argmax}}\, \mathrm{P}(\, p, x\,)$$

Finding *p** given x and using the Markov assumption is often called *decoding*. *Viterbi* is a common decoding algorithm.

Andrew Viterbi

Hidden Markov Model: Viterbi algorithm

Fill in a dynamic programming matrix *S*:



$S_{k,i}$ = *greatest joint probability* of observing the length-*i* prefix of *x* and any sequence of states ending in state *k*

Hidden Markov Model: Viterbi algorithm

Say x_i is Heads

$$s_{\text{Fair},\,i} = \underbrace{P(\text{Heads} \mid \text{Fair})}_{\text{Emission prob}} \cdot \max_{k \in \{\text{Fair, Loaded}\}} \{ s_{k,i-1} \cdot \underbrace{P(\text{Fair} \mid k)}_{\text{Transition prob}} \}$$

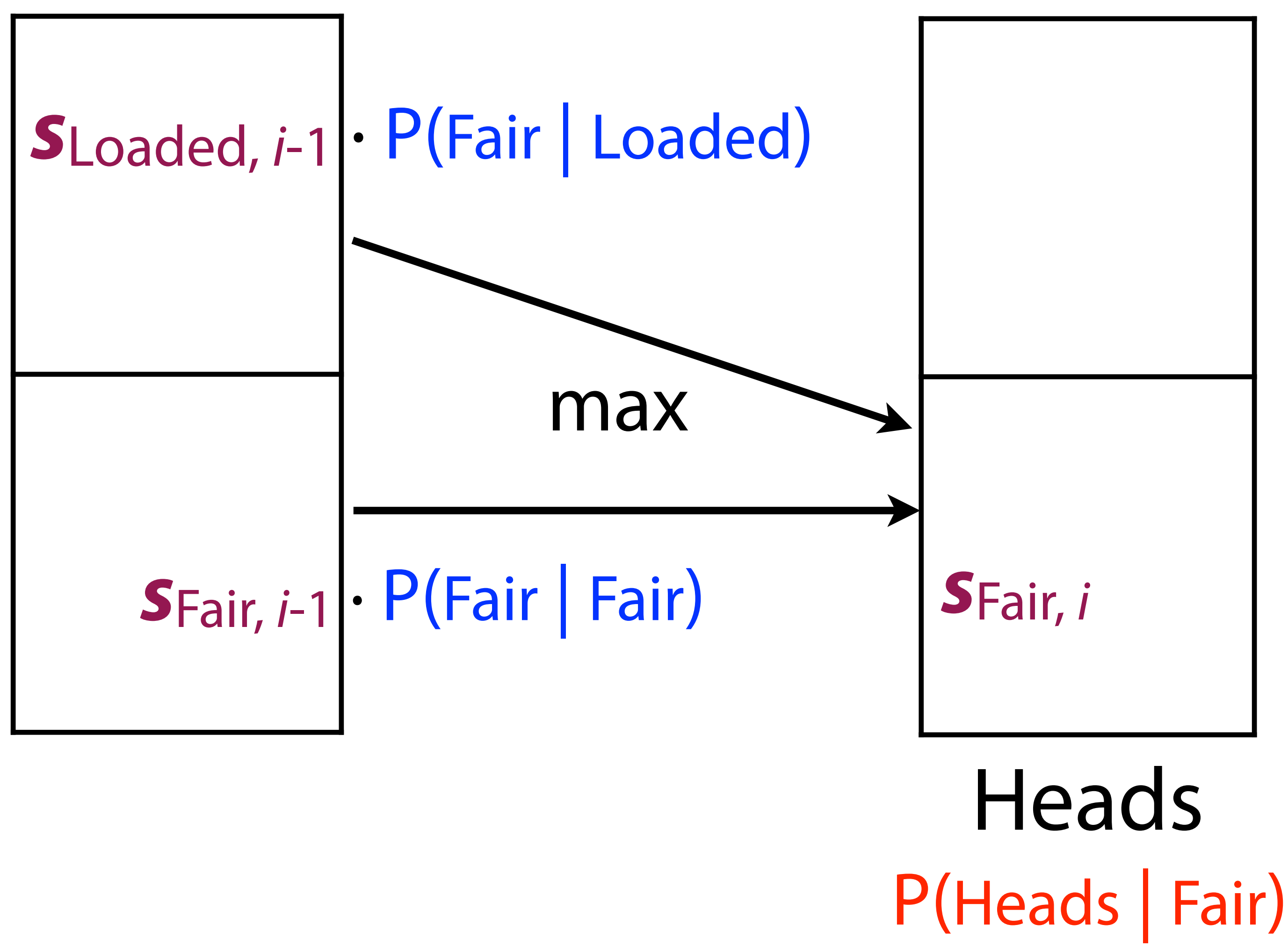
Hidden Markov Model: Viterbi algorithm

Say x_i is Heads

$$\boxed{s_{\text{Fair},\,i}} = \boxed{P(\text{Heads} \mid \text{Fair})} \cdot \boxed{\max_{k \in \{\text{Fair, Loaded}\}} \{ s_{k,i-1} \cdot P(\text{Fair} \mid k) \}}$$

Emission prob

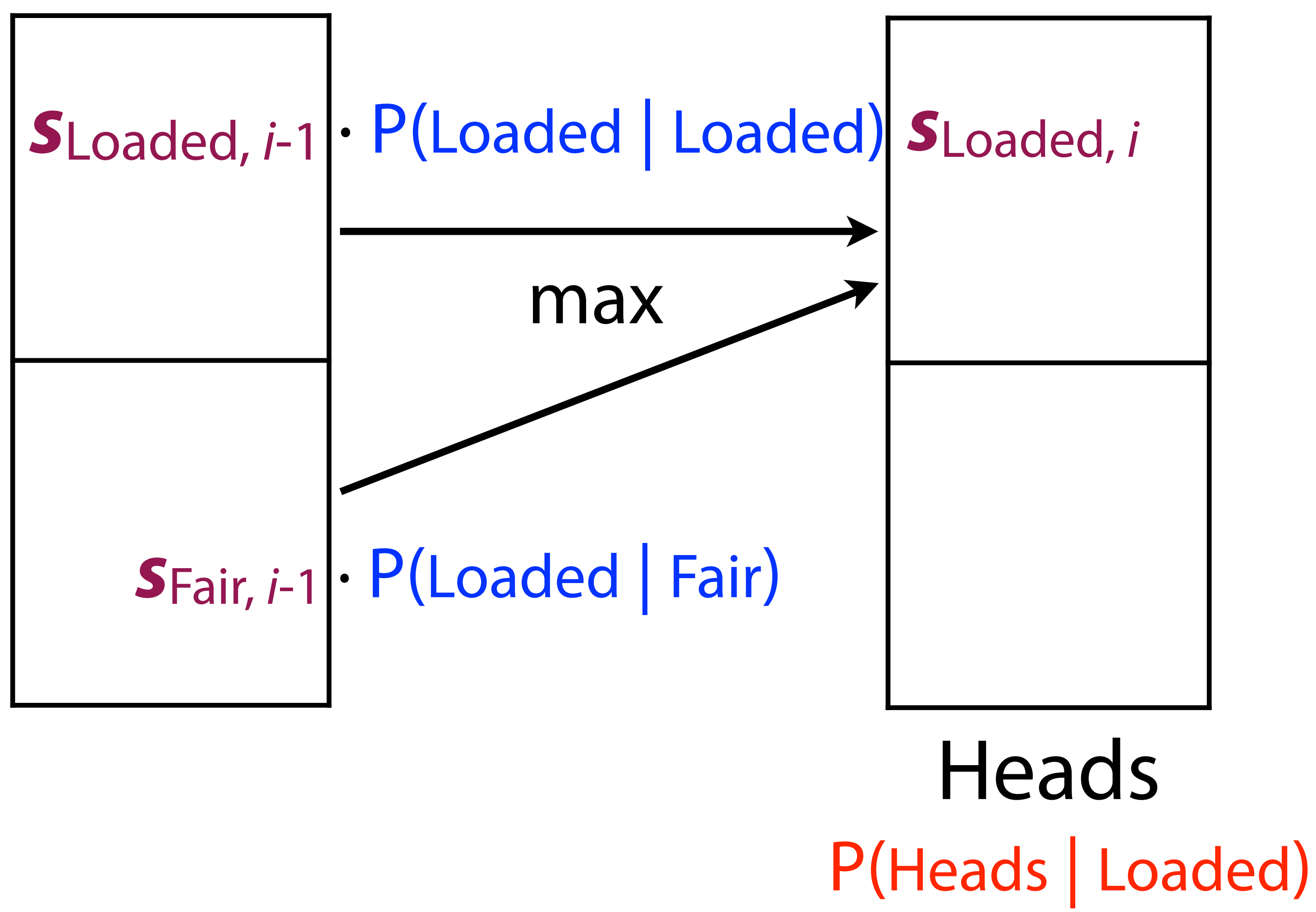
Transition prob



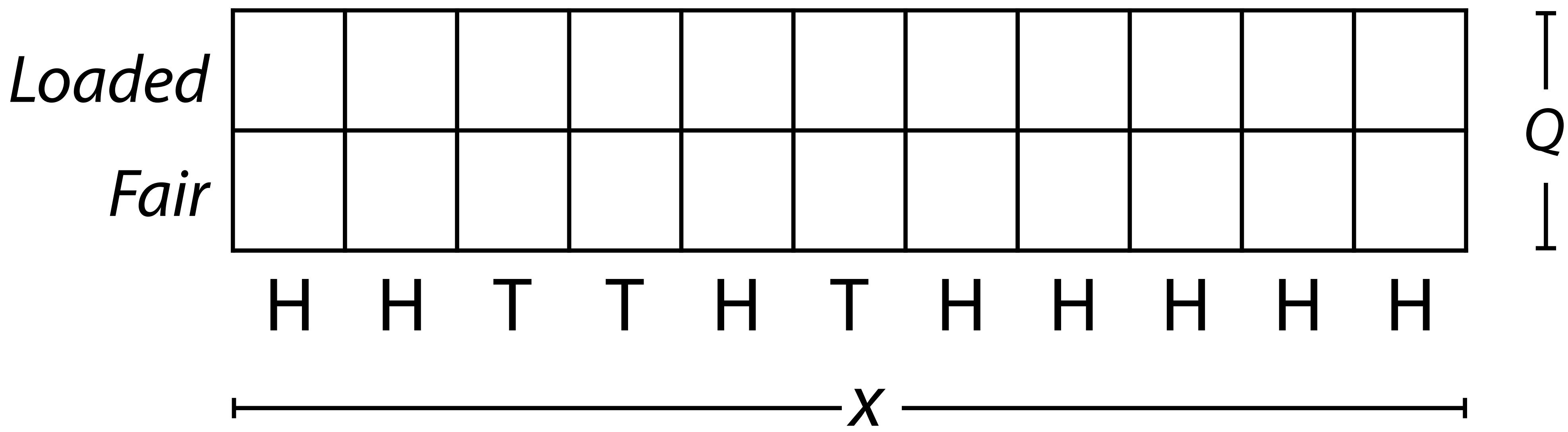
Hidden Markov Model: Viterbi algorithm

Say x_i is Heads

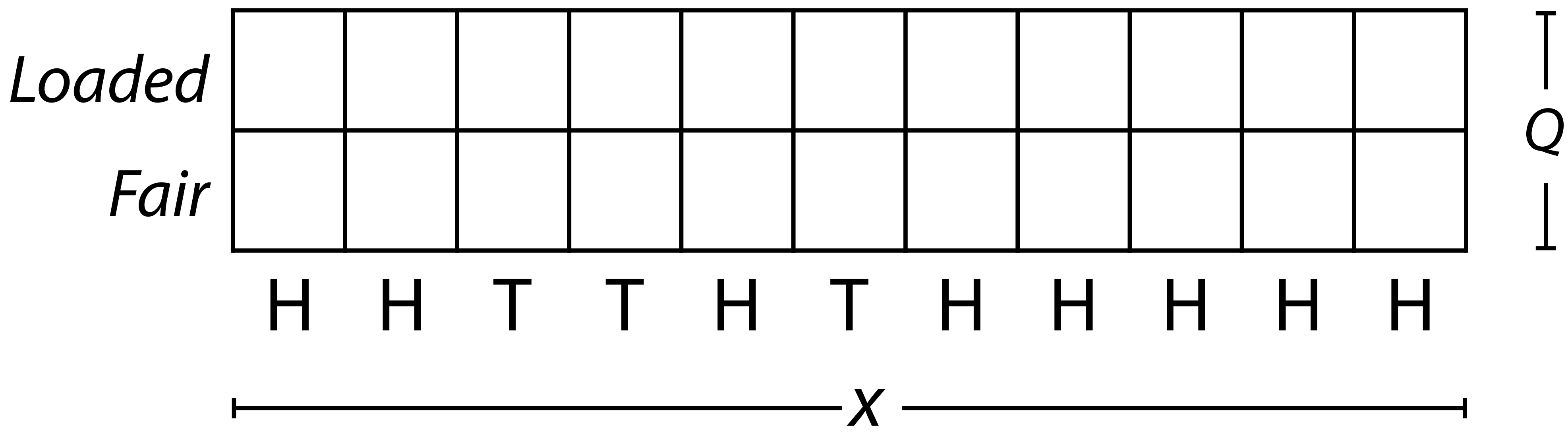
$$\boxed{s_{\text{Fair},\,i}} = \underbrace{P(\text{Heads} \mid \text{Loaded})}_{\text{Emission prob}} \cdot \max_{k \in \{\text{Fair, Loaded}\}} \{ s_{k,i-1} \cdot \underbrace{P(\text{Loaded} \mid k)}_{\text{Transition prob}} \}$$



Hidden Markov Model: Viterbi algorithm

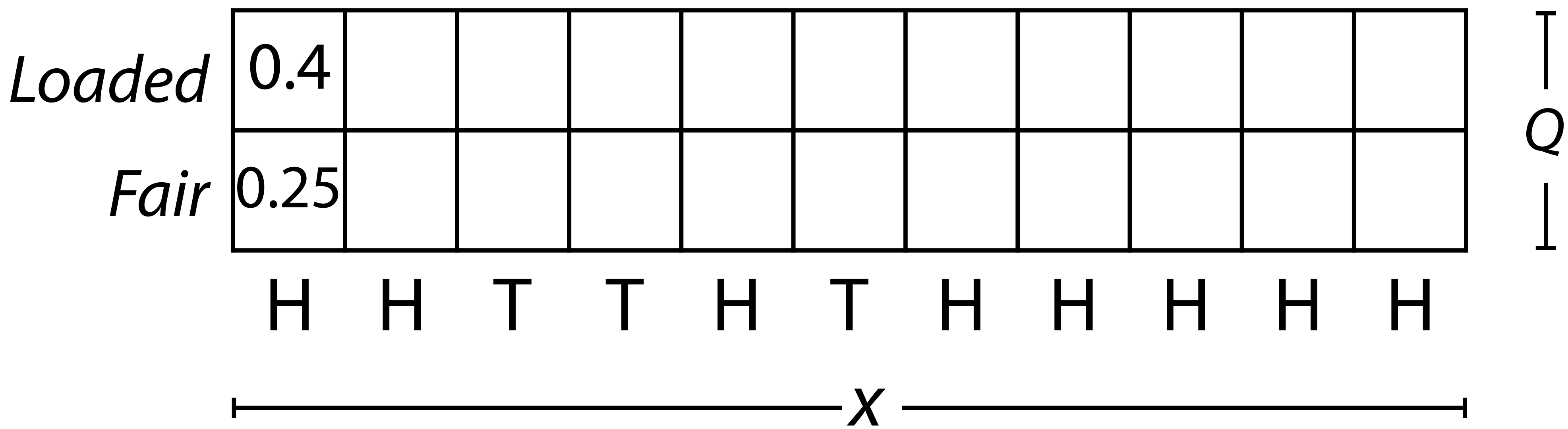


Hidden Markov Model: Viterbi algorithm



Dealer repeatedly flips a coin. Sometimes the coin is *fair*, with P(heads) = 0.5, sometimes it's *loaded*, with P(heads) = 0.8. Between each flip, dealer switches coins (invisibly) with prob. 0.4.

Hidden Markov Model: Viterbi algorithm

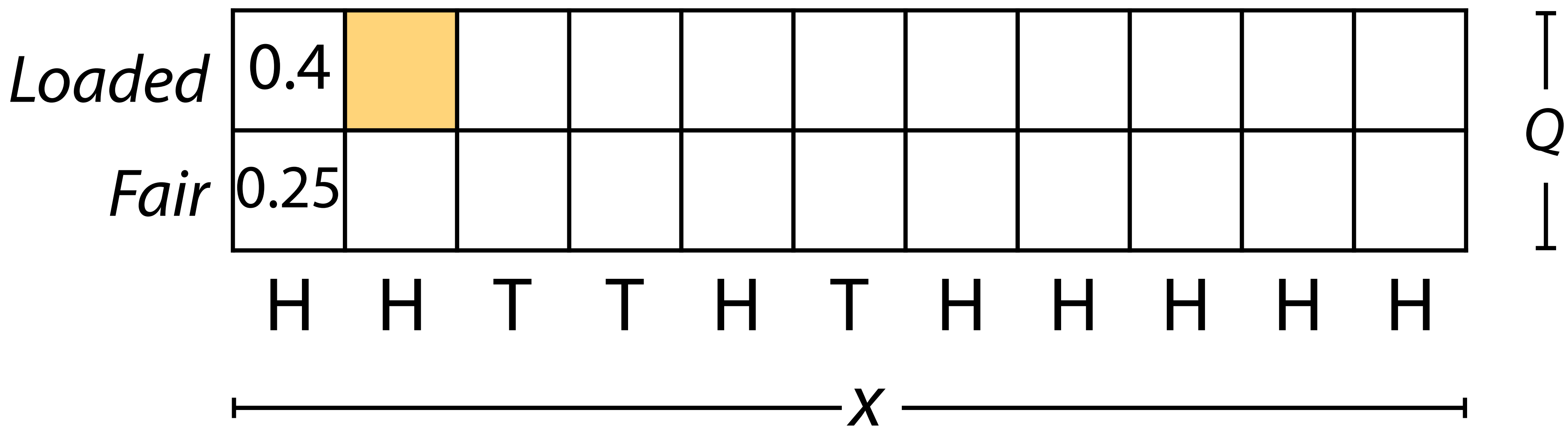


Assume we start with Fair/Loaded with equal probability

$$S_{L,0} = P(H \mid L) \cdot 0.5 \qquad S_{F,0} = P(H \mid F) \cdot 0.5$$

$$= 0.8 \cdot 0.5 \qquad = 0.5 \cdot 0.5$$

Hidden Markov Model: Viterbi algorithm



$S_{L,1} =$

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Hidden Markov Model: Viterbi algorithm

	H	H	T	T	H	T	H	H	H	H	H
Loaded	0.4										
Fair	0.25										

Q

X

$$S_{L,1} = P(H \mid L) \cdot \max \begin{cases} 0.4 \cdot P(L \mid L) \\ 0.25 \cdot P(L \mid F) \end{cases}$$

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Hidden Markov Model: Viterbi algorithm

Loaded	0.4	0.19									
Fair	0.25										
	H	H	T	T	H	T	H	H	H	H	H

Q

x

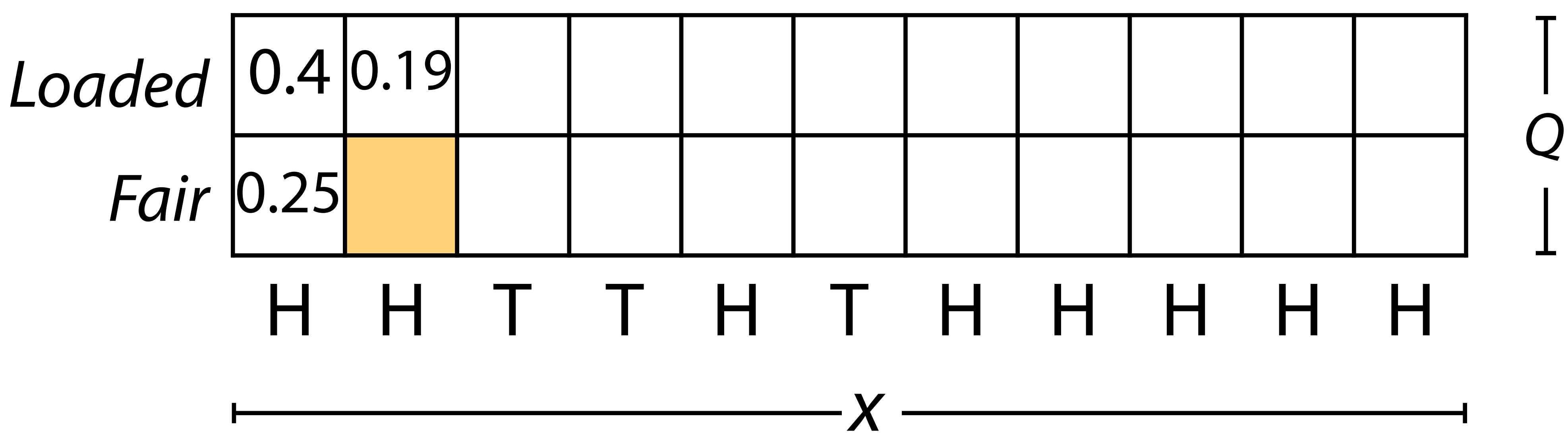
$$S_{L,1} = 0.8 \cdot \max\begin{cases} 0.4 \cdot 0.6 \\ 0.25 \cdot 0.4 \end{cases}$$

$$= 0.8 \cdot 0.4 \cdot 0.6 \approx 0.19$$

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Hidden Markov Model: Viterbi algorithm



$$S_{F,1} = P(H \mid F) \cdot \max \begin{cases} 0.4 \cdot P(F \mid L) \\ 0.25 \cdot P(F \mid F) \end{cases}$$

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Hidden Markov Model: Viterbi algorithm

Loaded	0.4	0.19									
Fair	0.25	0.08									
	H	H	T	T	H	T	H	H	H	H	H

Q

x

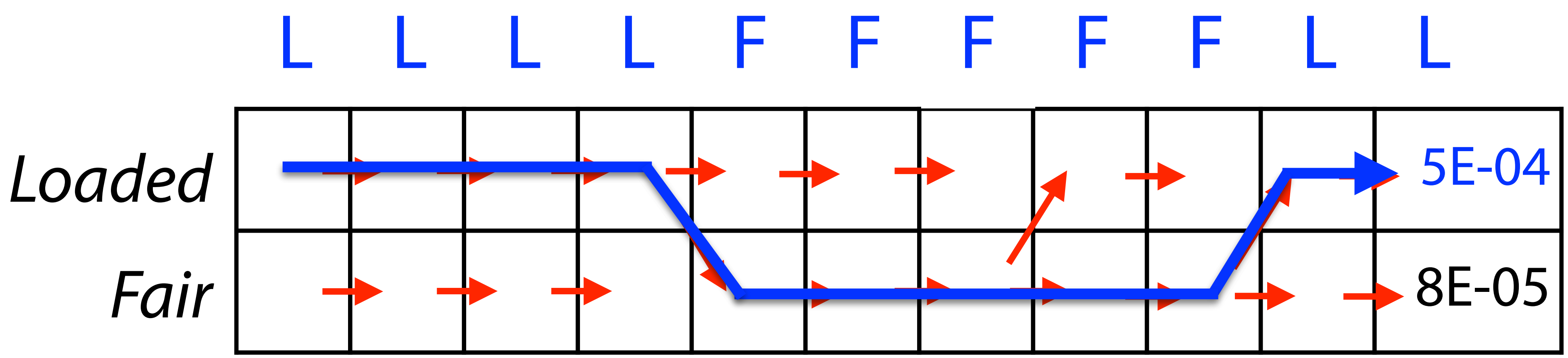
$$S_{F,1} = 0.5 \cdot \max \begin{cases} 0.4 \cdot 0.4 \\ 0.25 \cdot 0.6 \end{cases}$$

$$= 0.5 \cdot 0.4 \cdot 0.4 = 0.08$$

A	F	L
F	0.6	0.4
L	0.4	0.6

E	H	T
F	0.5	0.5
L	0.8	0.2

Hidden Markov Model: Viterbi algorithm



Arrow corresponds to term that "wins" the maximum

Traceback:

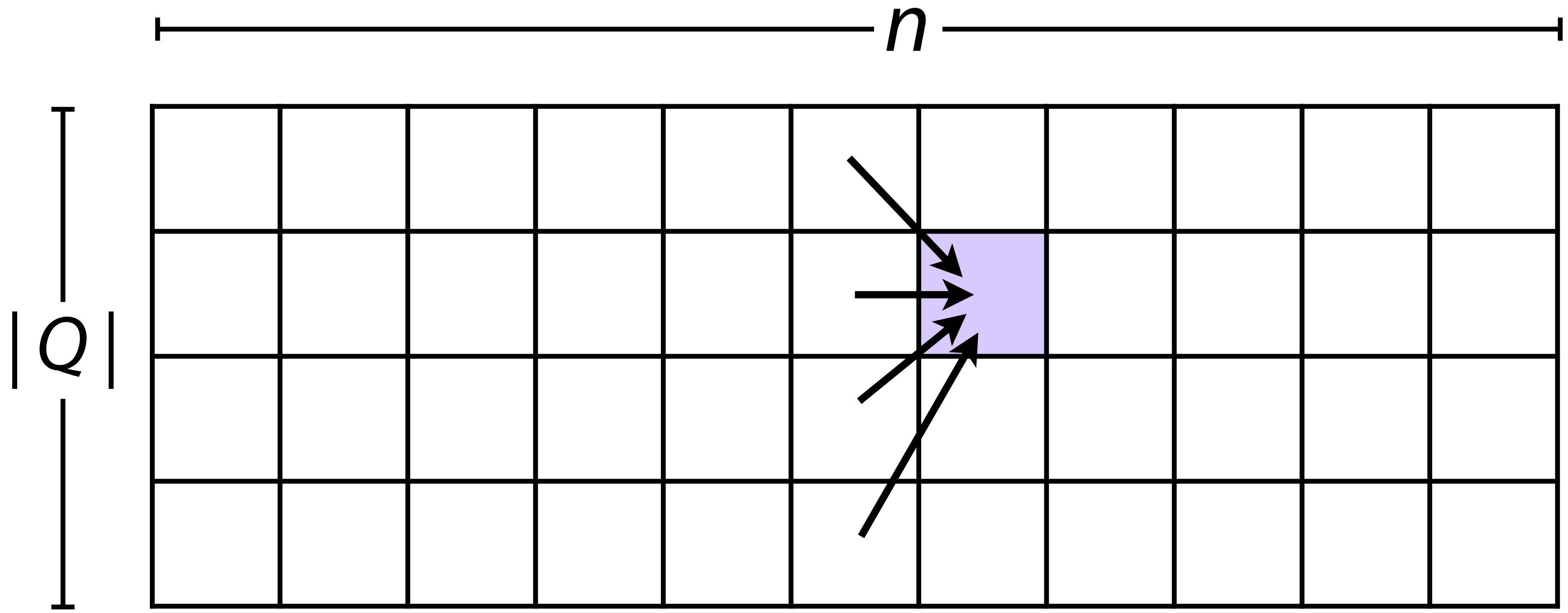
Start from greatest score in final column

Keep asking "how did I get here?" (which predecessor state "won" the maximum) until we reach 1st column

http://bit.ly/CG_HMM

Hidden Markov Model: Viterbi algorithm

How much work is this? Q = set of states, n = length of emission string



\# $s_{k,i}$ values to calculate = $n \cdot |Q|$, each involves max over $|Q|$ products

$O(n \cdot |Q|^2)$

Matrix A has $|Q|^2$ elements, E has $|Q||\Sigma|$ elements, I has $|Q|$ elements

Hidden Markov Model: Viterbi algorithm

```
>>> hmm = HMM({"FF":0.6, "FL":0.4, "LF":0.4, "LL":0.6},
...           {"FH":0.5, "FT":0.5, "LH":0.8, "LT":0.2},
...           {"F":0.5, "L":0.5})
>>> prob, _ = hmm.viterbi("THTHHHTHTTH")
>>> print prob
2.86654464e-06
>>> prob, _ = hmm.viterbi("THTHHHTHTTH" * 100)
>>> print prob
0.0
```

Repeat string 100 times

Occassionally dishonest casino setup

What happened? Underflow!

http://bit.ly/CG_HMM

Hidden Markov Model: Viterbi algorithm

```
>>> hmm = HMM({"FF":0.6, "FL":0.4, "LF":0.4, "LL":0.6},
...           {"FH":0.5, "FT":0.5, "LH":0.8, "LT":0.2},
...           {"F":0.5, "L":0.5})
>>> prob, _ = hmm.viterbi("THTHHHTHTTH")
>>> print prob
2.86654464e-06
>>> prob, _ = hmm.viterbi("THTHHHTHTTH" * 100)
>>> print prob
0.0
>>> logprob, _ = hmm.viterbiL("THTHHHTHTTH" * 100)
>>> print logprob
-1824.4030071946879
```

Repeat string 100 times

log-space Viterbi

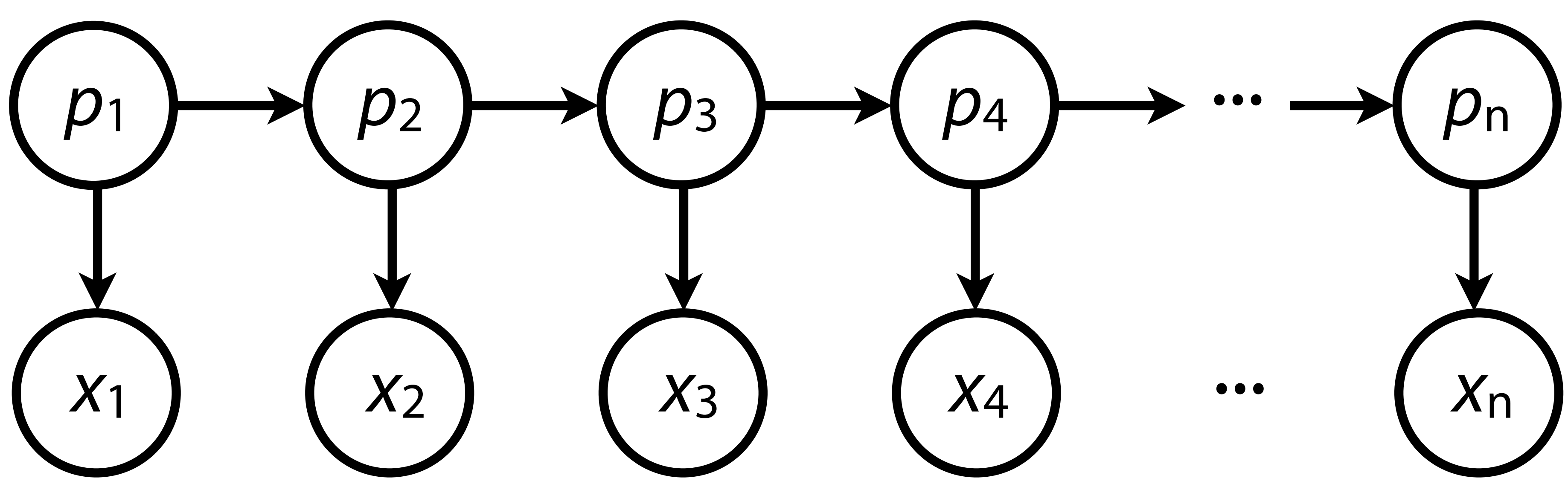
Solution: switch to *log probabilities*. Multiplies become adds.

http://bit.ly/CG_HMM

Hidden Markov Model

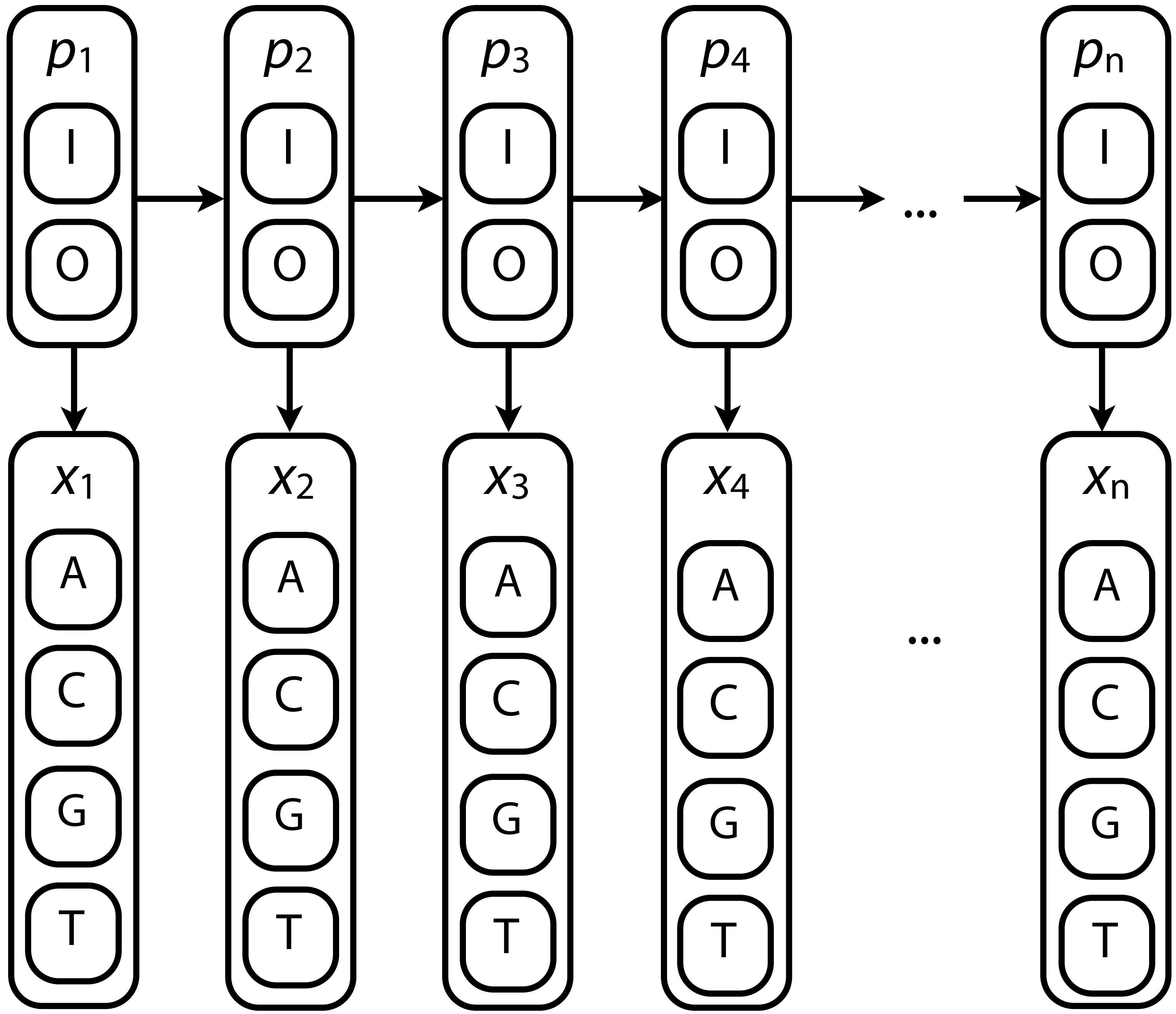
We know what an HMM is, how to calculate joint probability, and how to find the most likely path given an emission (Viterbi)

Can we design an HMM for finding CpG islands?



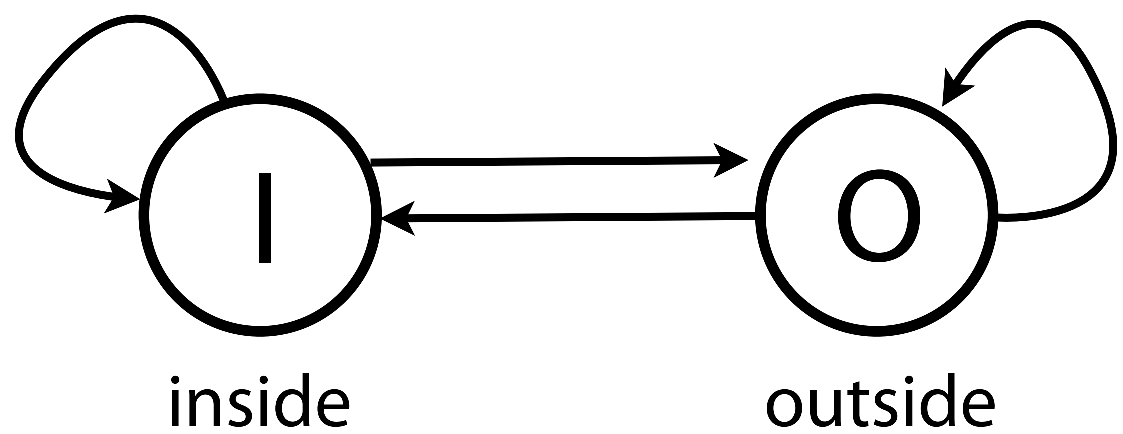
Hidden Markov Model

Idea 1: Q = { inside, outside }, Σ = { A, C, G, T }



Hidden Markov Model

Idea 1: $Q = \{ \text{inside, outside} \}$, $\Sigma = \{ A, C, G, T \}$



A	I	O
I		
O		

Transition matrix

Fraction of Is followed by Os

E	A	C	G	T
I				
O				

Emission matrix

Estimate as fraction of nucleotides inside islands that are C

Hidden Markov Model

Example 1 using HMM idea 1:

A	I	O
I	0.8	0.2
O	0.2	0.8

E	A	C	G	T
I	0.1	0.4	0.4	0.1
O	0.25	0.25	0.25	0.25

I
0.5
0.5

x: ATATATACGCGCGCGCGCGCGATATATATATATA

p: 0000000IIIIIIIIIIIIII0000000000000

(from Viterbi)

http://bit.ly/CG_HMM

Hidden Markov Model

Example 2 using HMM idea 1:

A	I	O
I	0.8	0.2
O	0.2	0.8

E	A	C	G	T
I	0.1	0.4	0.4	0.1
O	0.25	0.25	0.25	0.25

I
0.5
0.5

x: ATATCGCGCGCGATATATCGCGCGCGATATATAT

p: OOOOIIIIIIIIOOOOOOIIIIIIIIOOOOOOOO

(from Viterbi)

http://bit.ly/CG_HMM

Hidden Markov Model

Example 3 using HMM idea 1:

A	I	O
I	0.8	0.2
O	0.2	0.8

E	A	C	G	T
I	0.1	0.4	0.4	0.1
O	0.25	0.25	0.25	0.25

I
0.5
0.5

x: ATATATACCCCCCCCCCCCCCATATATATATATA

p: 0000000IIIIIIIIIIIIII0000000000000

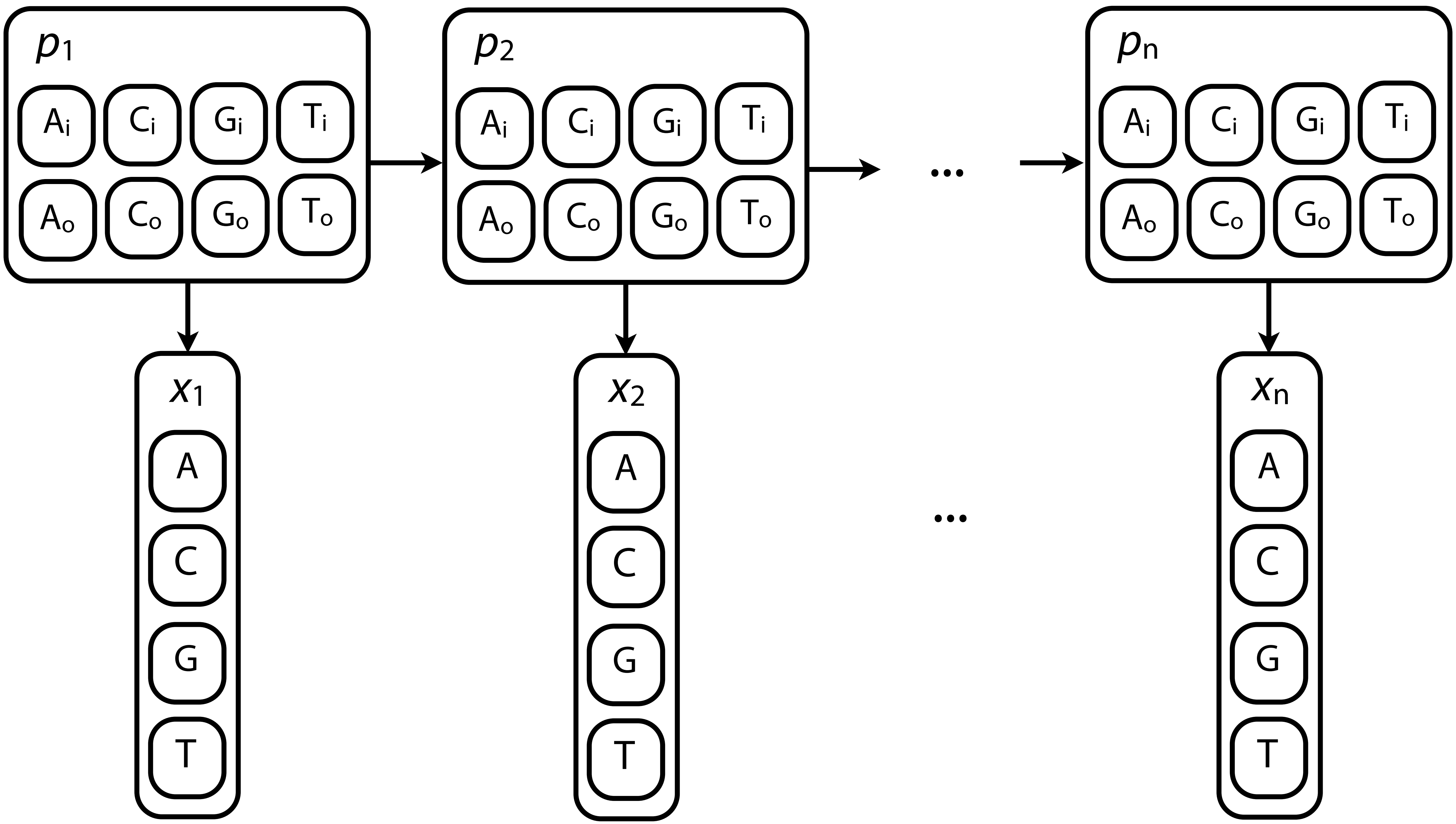
(from Viterbi)

Oops - clearly not a CpG island

http://bit.ly/CG_HMM

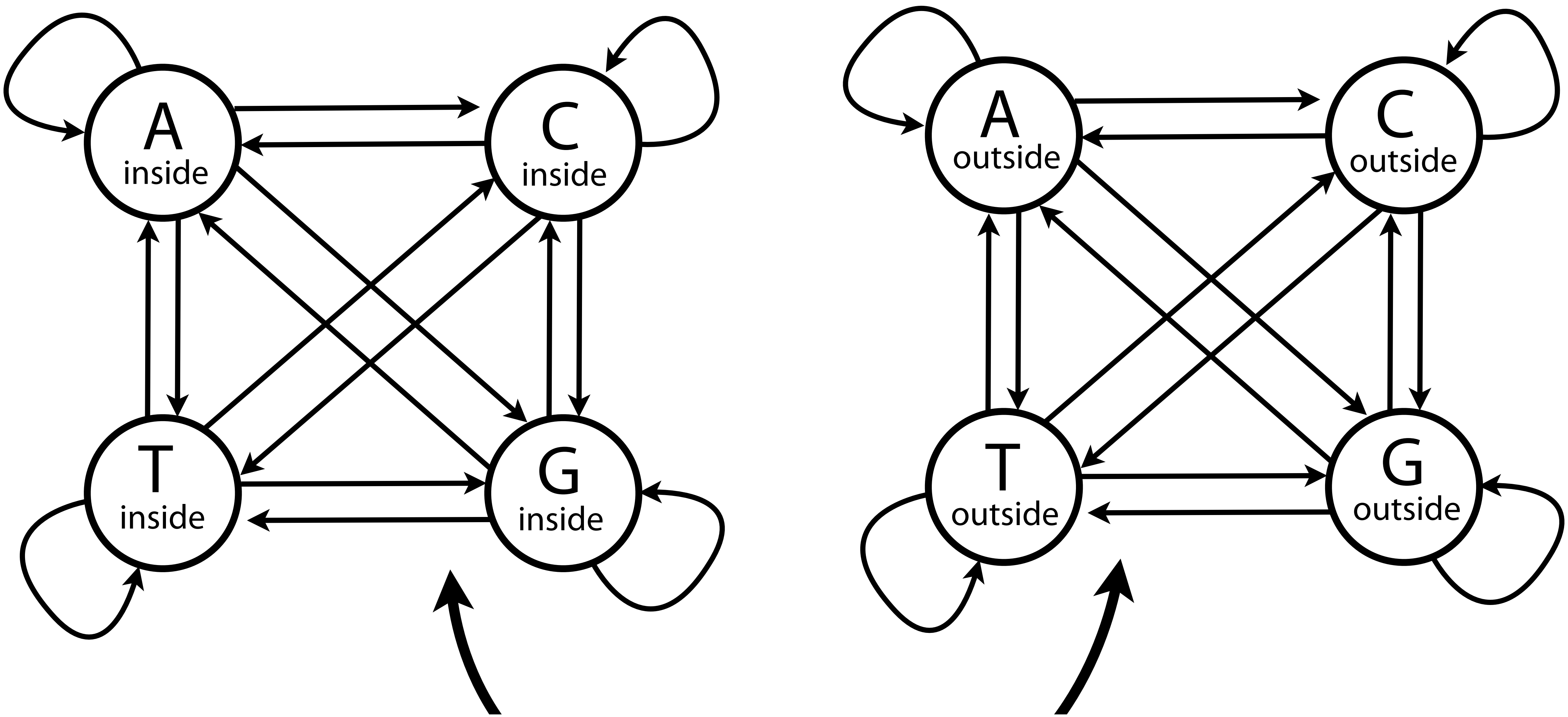
Hidden Markov Model

Idea 2: $Q = \{ A_i, C_i, G_i, T_i, A_o, C_o, G_o, T_o \}, \Sigma = \{ A, C, G, T \}$



Hidden Markov Model

Idea 2: $Q = \{ A_i, C_i, G_i, T_i, A_o, C_o, G_o, T_o \}$, $\Sigma = \{ A, C, G, T \}$



All inside-outside edges

Hidden Markov Model

Idea 2: $Q = \{ A_i, C_i, G_i, T_i, A_o, C_o, G_o, T_o \}$, $\Sigma = \{ A, C, G, T \}$

A	A_i	C_i	G_i	T_i	A_o	C_o	G_o	T_o
A_i								
C_i								
G_i								
T_i								
A_o								
C_o								
G_o								
T_o								

Estimate $P(C_i \mid T_i)$ as # T_i C_i s divided by # T_i s

Transition matrix

E	A	C	G	T
A_i	1	0	0	0
C_i	0	1	0	0
G_i	0	0	1	0
T_i	0	0	0	1
A_o	1	0	0	0
C_o	0	1	0	0
G_o	0	0	1	0
T_o	0	0	0	1

Emission matrix

Hidden Markov Model

Trained transition matrix:

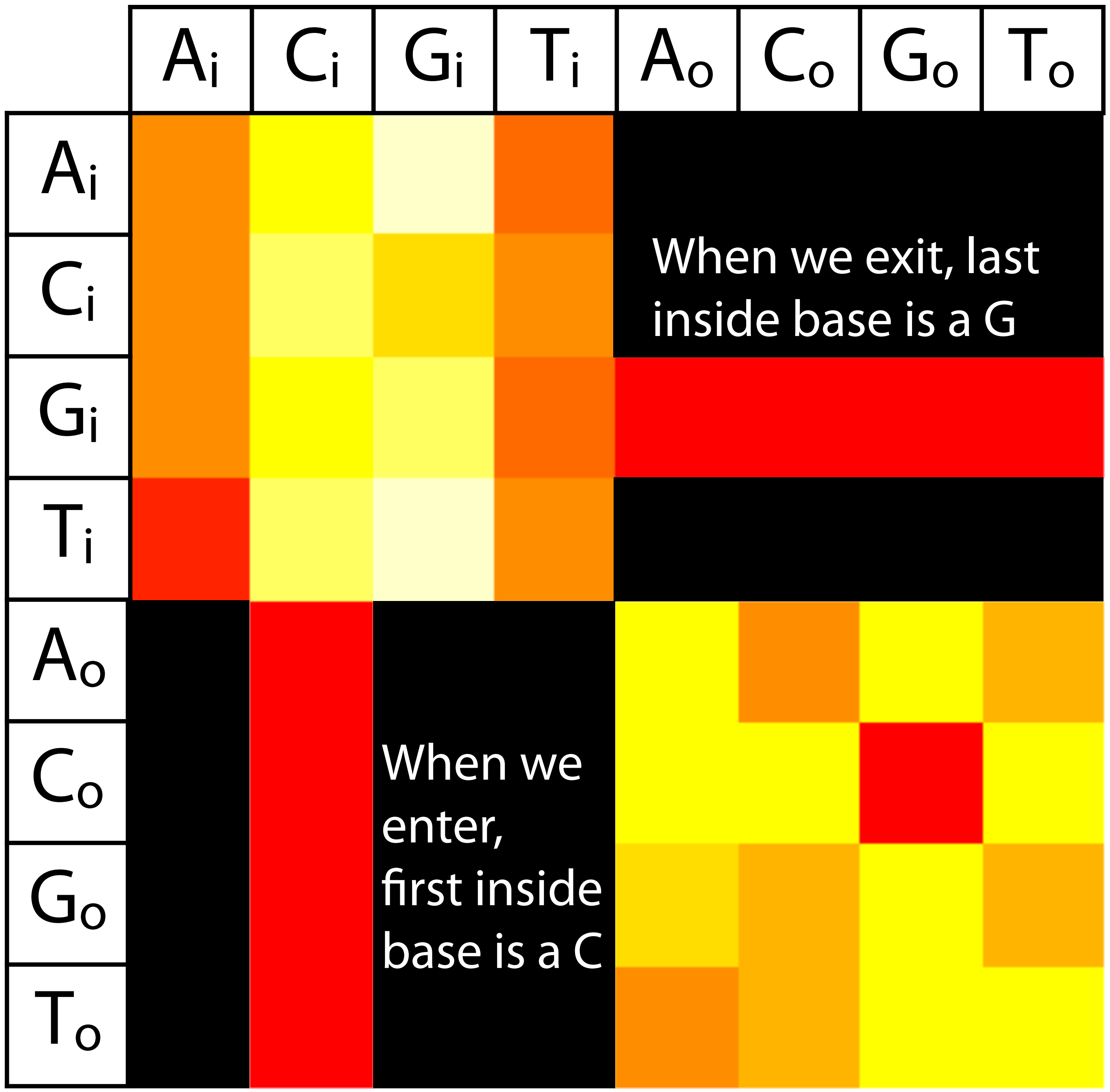
	A	C	G	T	a	c	g	t
A:	0.185441,	0.276405,	0.400914,	0.137241,	0.000000,	0.000000,	0.000000,	0.000000
C:	0.189582,	0.359051,	0.253240,	0.198127,	0.000000,	0.000000,	0.000000,	0.000000
G:	0.171904,	0.328635,	0.354250,	0.139893,	0.000785,	0.001479,	0.001857,	0.001198
T:	0.094102,	0.341636,	0.376867,	0.187395,	0.000000,	0.000000,	0.000000,	0.000000
a:	0.000000,	0.000038,	0.000000,	0.000000,	0.294811,	0.194641,	0.286965,	0.223544
c:	0.000000,	0.000076,	0.000000,	0.000000,	0.326810,	0.294085,	0.061724,	0.317305
g:	0.000000,	0.000057,	0.000000,	0.000000,	0.257128,	0.233486,	0.294244,	0.215085
t:	0.000000,	0.000031,	0.000000,	0.000000,	0.179562,	0.232469,	0.294630,	0.293308

Uppercase = inside, lowercase = outside

Hidden Markov Model

Trained transition matrix A:

Once inside, we're likely to stay inside for a while



Red: low probability

Yellow: high probability

Black: probability = 0

Same for outside

Hidden Markov Model

Viterbi result; lowercase = *outside*, uppercase = *inside*:

```
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCG
CGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCG
CGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
```

Hidden Markov Model

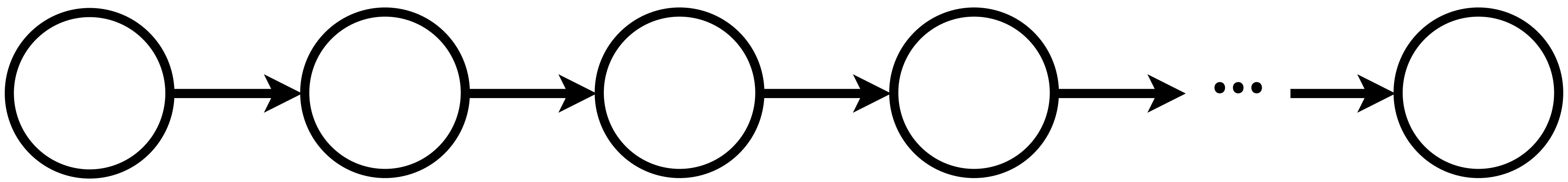
Viterbi result; lowercase = *outside*, uppercase = *inside*:

```
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatgggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
ggggggggggggggggggggggggggggggggggggggggggggatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
atatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatatat
```

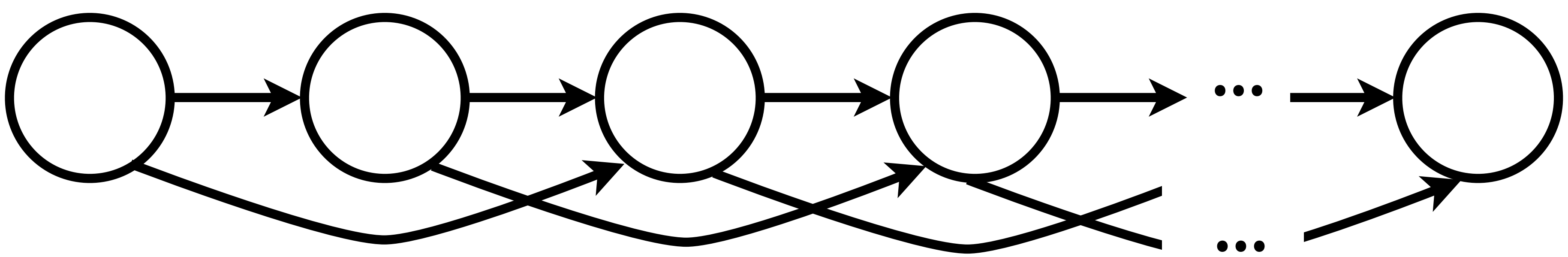
Hidden Markov Model

Many of the Markov chains and HMMs we've discussed are *first order,* but we can also design models of higher orders

First-order Markov chain:

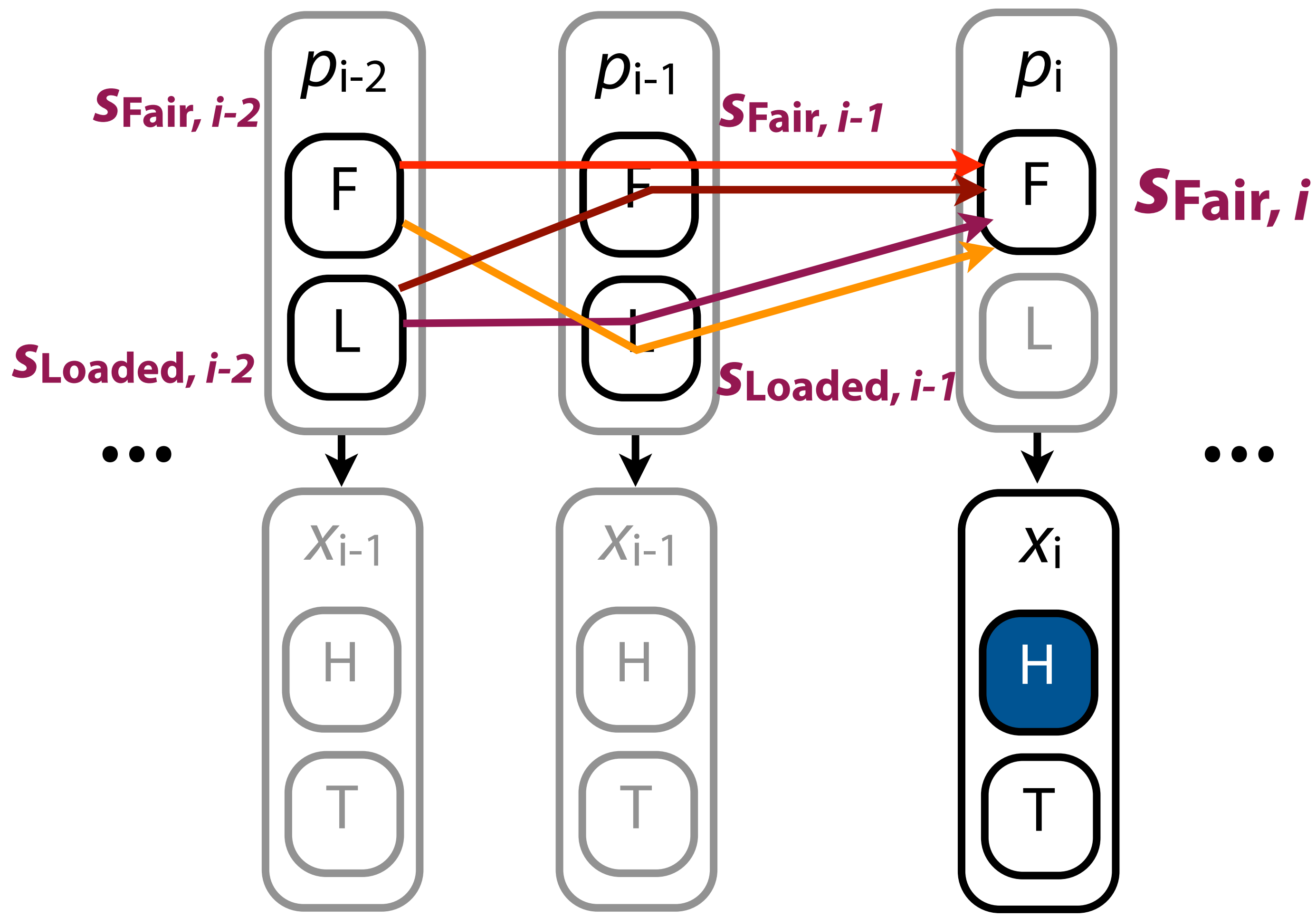


Second-order Markov chain:



Hidden Markov Model

For higher-order HMMs, Viterbi $s_{k,\,i}$ no longer depends on just the previous state assignment



Equivalently, we can expand the state space, as we did for CpG islands.