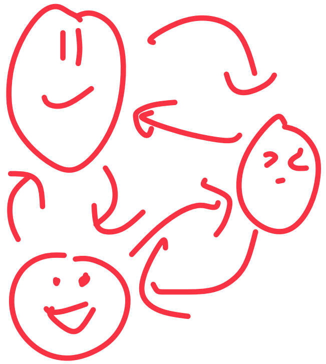
String Algorithms and Data Structures
Markov Chains & HMMs

CS 199-225
Brad Solomon

December 2, 2024



University of Illinois Urbana-Champaign
Department of Computer Science

Please fill out ICES Evaluations

Feedback is important for the development of the class

If not enough people fill it out, doesn’t actually get recorded

Learning Objectives

Introduce State Diagrams and Markov Chains

Identify how Markov chains can be used to:

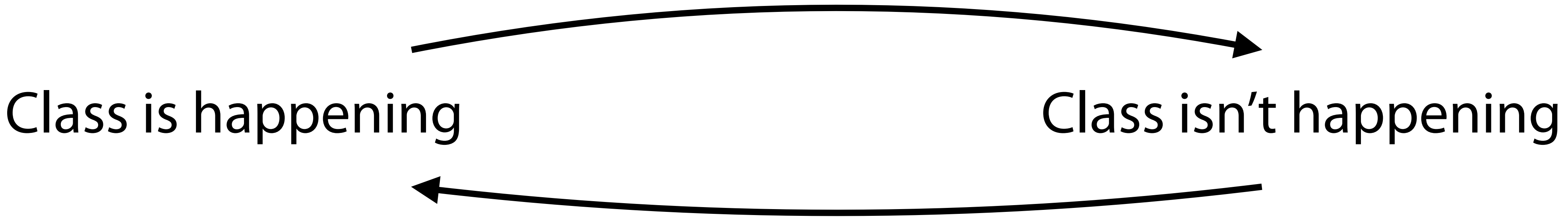
- Estimate probabilities of sequences
- Identify more probable labels
- Predict future states

Define and determine stationary states

Introduce Hidden Markov Models

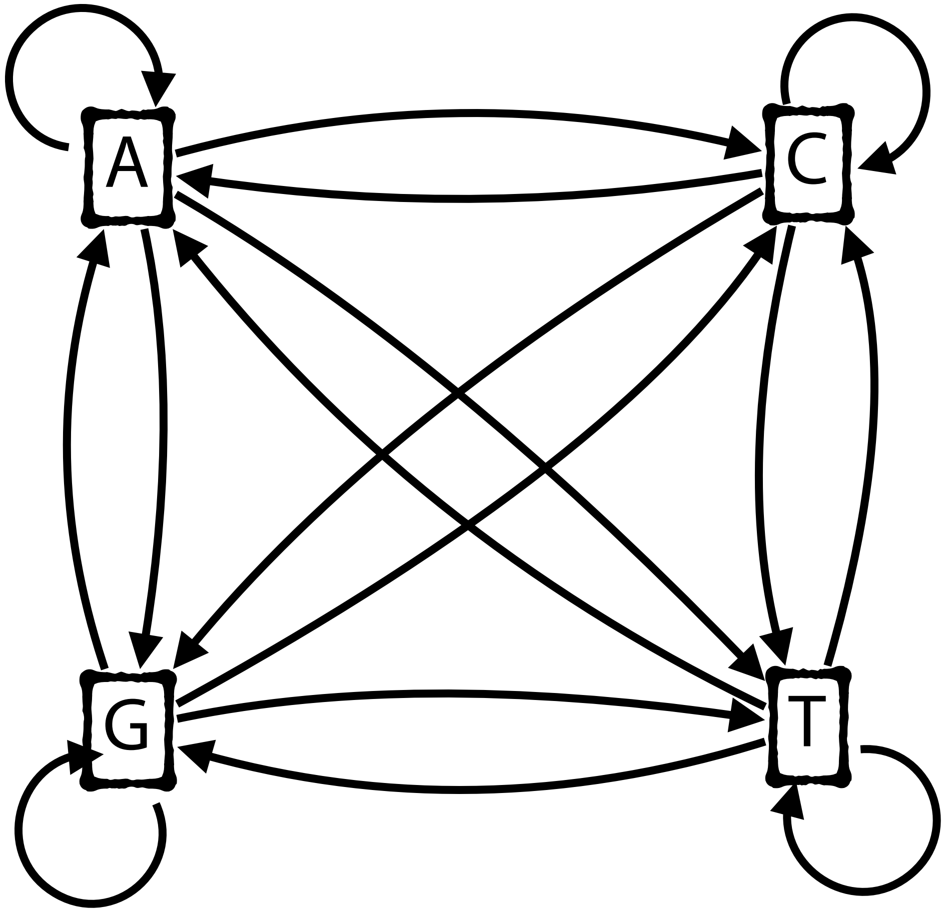
Modeling events with State Diagrams

A **state diagram** is a (usually weighted) directed graph where nodes are states and edges are transitions between them



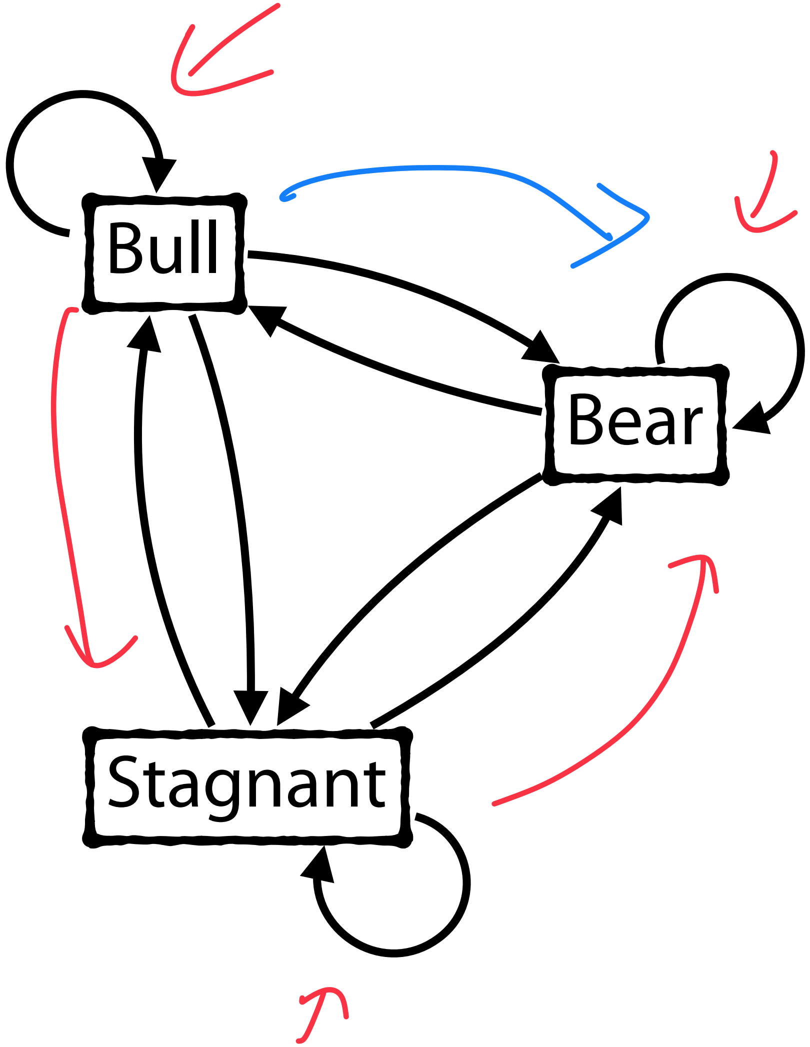
These diagrams are very useful in modeling many real world scenarios!

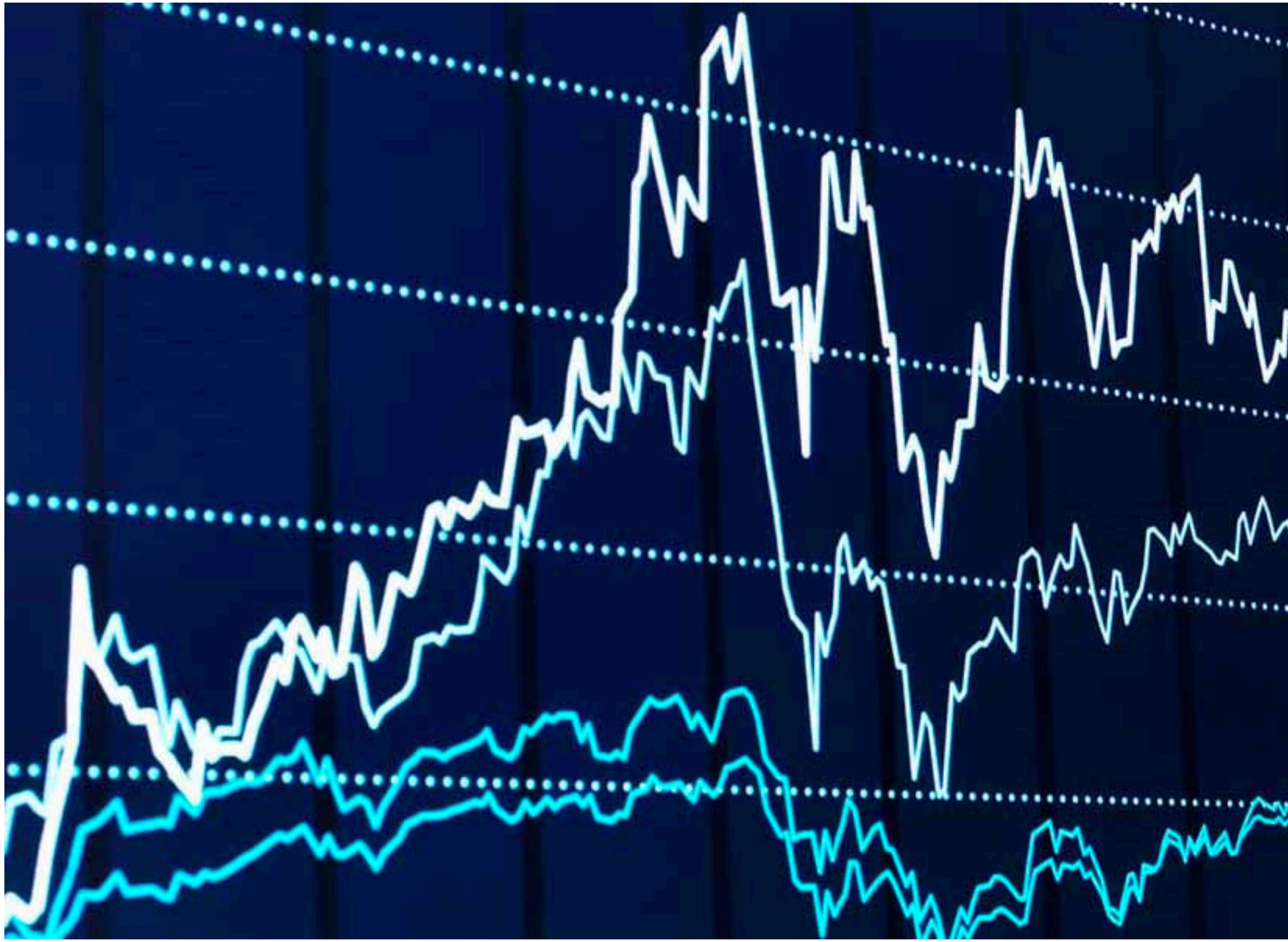
Sequence Modeling in Biology



CATGACGTCGCGGACAACCCAGAATTGTCTTGAGCGATGGTAAGATCTAACCTCACTGC
CTGGGGCTTTACTGATGTCATACCGTCTTGCACGGGGATAGAATGACGGTGCCCGTGTC
ATTTTCTGAAAGTTACAGACTTCGATTAAAAAGATCGGACTGCGCGTGGGCCCGGAGAG
TTTTTCGACGTGTCAAGGACTCAAGGGAATAGTTTGGCGGGAGCGTTACAGCTTCAATT
CGATAAAATTCAACTACTGGTTTCGGCCTAATAGGTCACGTTTTATGTGAAATAGAGGG
CCCTGGGTGTTCTATGATAAGTCCTGCTTTATAACACGGGGCGGTTAGGTTAAATGACT
ATCCAAGCGCCCGCTAATTCTGTTCTGTTAATGTTCATACCAATACTCACATCACATTA
AGCCCAGTCGCAAGGGTCTGCTGCTGTTGTCGACGCCTCATGTTACTCCTGGAATCTAC
GGTTAAGGCGTGTGATCGACGATGCAGGTATACATCGGCTCGGACCTACAGTGGTCGAT
TCGCGGTTCGGCGCGTAGTTGAGTGCGATAACCCAACCGGTGGCAAGTAGCAAGAAGAC
AGACAACCTAACTAATAGTCTCTAACGGGGAATTACCTTTACCAGTCTCATGCCTCCAA
CAATGATATCGCCCACAGAAAGTAGGGTCTCAGGTATCGCATACGCCGCGCCCGGGTCC
GACAGTAGAGAGCTATTGTGTAATTCAGGCTCAGCATTCATCGACCTTTCCTGTTGTGA
TCTCGTCCGTAACGATCTGGGGGGCAAAACCGAATATCCGTATTCTCGTCCTACGGGTC
TGCGCGTGATCGTCAGTTAAGTTAAATTAATTCAGGCTACGGTAAACTTGTAGTGAGCT
ACGGGTTCGCTACAGATGAACTGAATTTATACACGGACAACTCATCGCCCATTTGGGCG
AAAGTGGCAGATTAGGAGTGCTTGATCAGGTTAGCAGGTGGACTGTATCCAACAGCGCA
CCAAAGCGTTGTAGTGGTCTAAGCACCCCTGAACAGTGGCGCCCATCGTTAGCGTAGTA
AGGTGCGACATGGGGCCAGTTAGCCTGCCCTATATCCCTTGCACACGTTCAATAAGAGG
TTTTTAAATTAGGATGCCGACCCCATCATTGGTAACTGTATGTTCATAGATATTTCTTC
AGCTGACACGCAAGGGTCAACAATAATTTCTACTATCACCCCGCTGAACGACTGTCTTT
CTTAGATTCGCGTCCTAACGTAGTGAGGGCCGAGTCATATCATAGATCAGGCATGAGAA
CACACGAGTTGTAAACAACTTGATTGCTATACTGTAGCTACCGCAAGGATCTCCTACAT
ATCTGGATCCGAGTCAGAAATACGAGTTAATGCAAATTTACGTAGACCGGTGAAAACAC
AGACCGTAGTCAGAAGTGTGGCGCGCTATTCGTACCGAACCGGTGGAGTATACAGAATT
AGGAGCTCGGTCCCCAATGCACGCCAAAAAAGGAATAAAGTATTCAAACTGCGCATGGT
CTATTATCCATCCGAACGTTGAACCTACTTCCTCGGCTTATGCTGTCCTCAACAGTATC
CGGCTGTGGATCTTAACGGCCACATTCTTAATTCCGACCGATCACCGATCGCCTTTCCT
ACTAAGTTATCCAGATCAAGGTTTGAACGGACTCGTATGACATGTGTGACTGAACCCGG
CTGTTTCAAGGCCTCTGCTTTGGTATCACTCAATATATTCAGACCAGACAAGTGGCAAA
CTAGGTATTCACGCAACCGTCGTAACATGCACTAAGGATAACTAGCGCCAGGGGGGCAT
AAAGACTACCCTATGGATTCCTTGGAGCGGGGACAATGCAGACCGGTTACGACACAATT
GGTATTATTAGCAAGACAATAAAGGACATTGCACAGAGACTTATTAGAATTCAACAAAC
GTGTTGGGTCGGGCAAGTCCCCGAAGCTCGGCCAAAAGATTCGCCATGGAACCGTCTGG

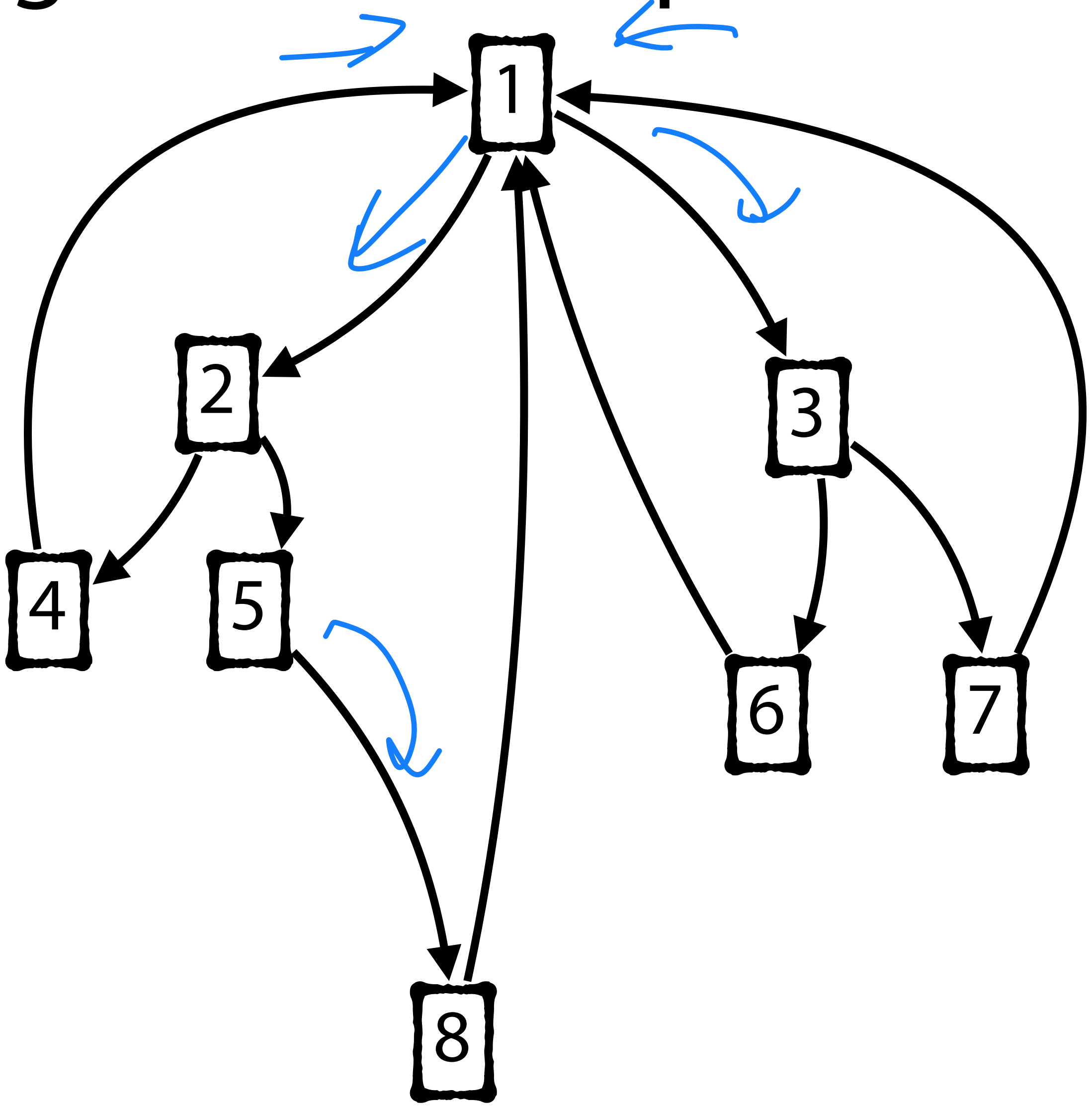
Market Trends in Economics





PageRank in Graphs

Total weight 1



Equilibrium State

1: 4/13

2: 2/13

3: 2/13

4: 1/13

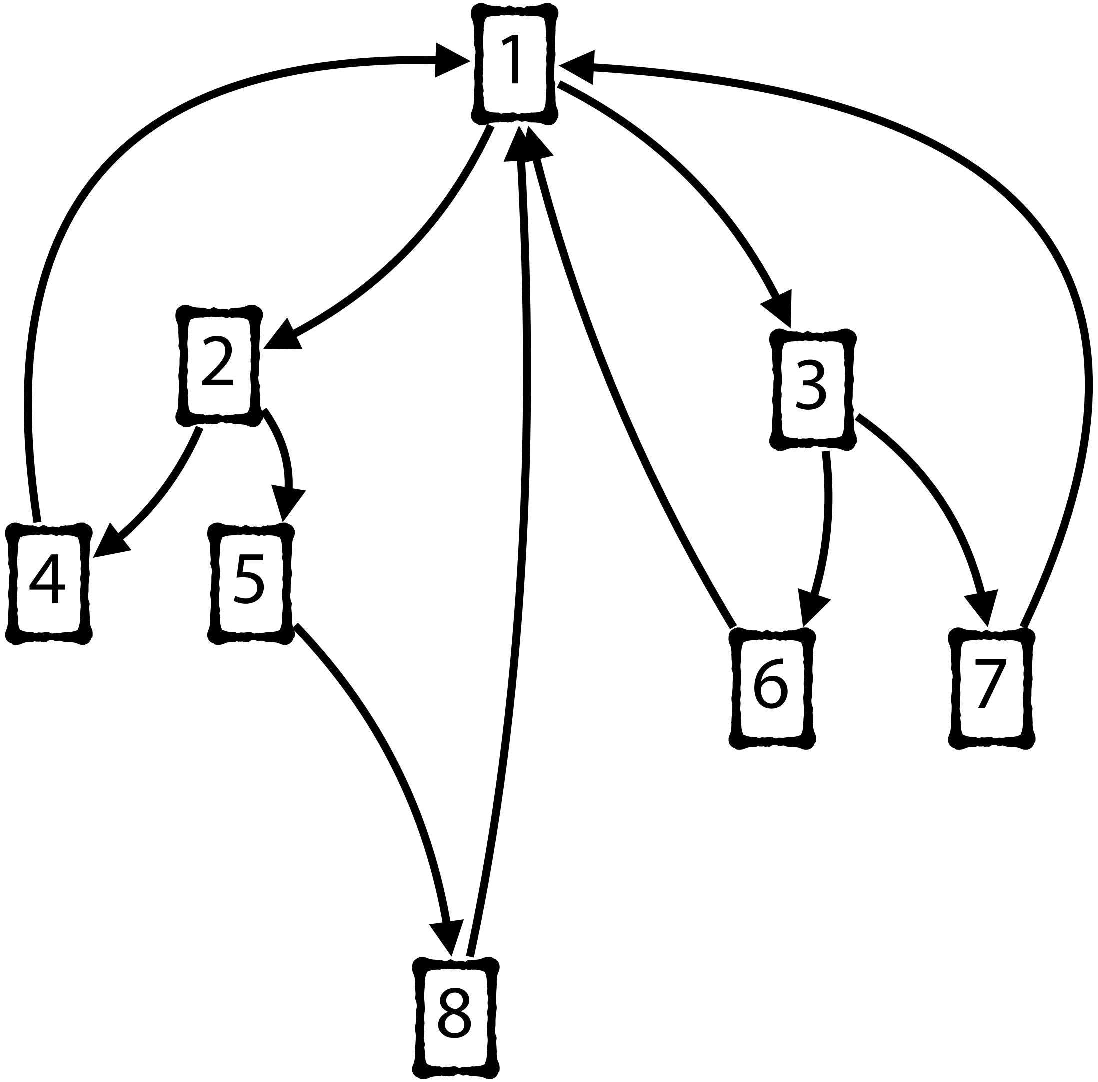
5: 1/13

6: 1/13

7: 1/13

8: 1/13

PageRank in Graphs



Equilibrium State

1: 4/13

2: 2/13

3: 2/13

4: 1/13

5: 1/13

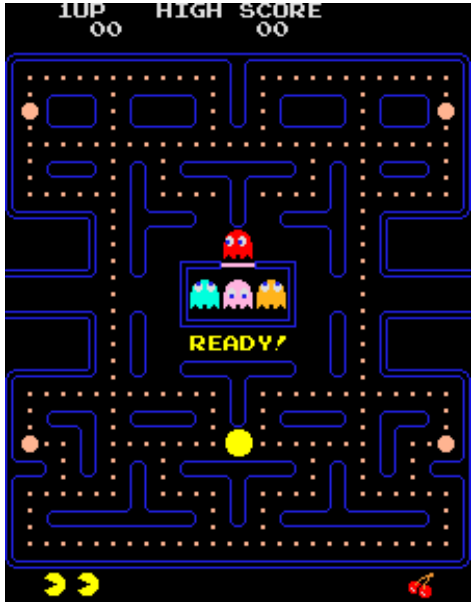
6: 1/13

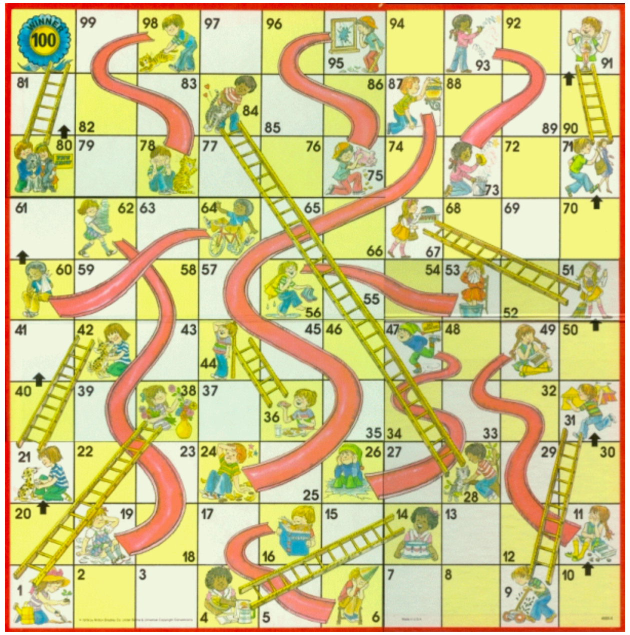
7: 1/13

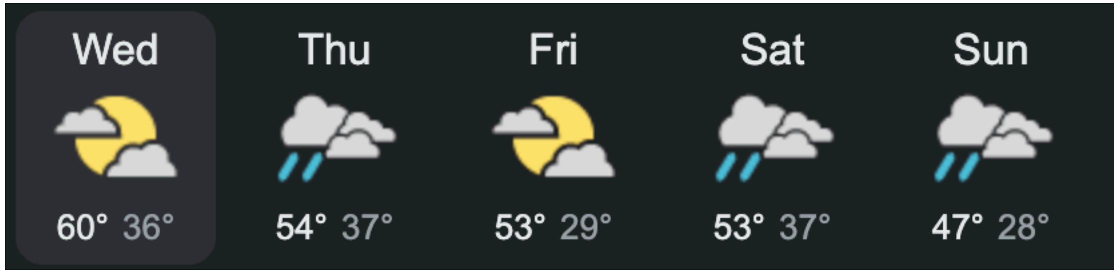
8: 1/13

Markov Assumption

The probability of the next state depends only on our current state

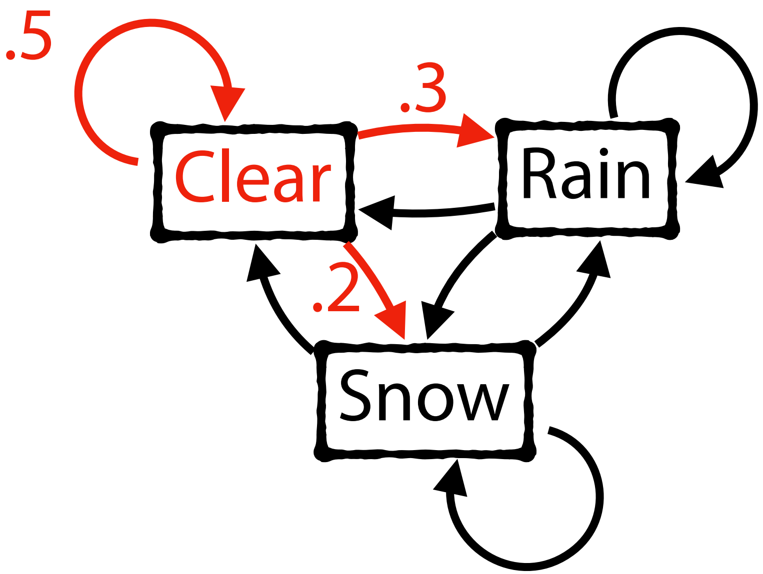






Markov Chain

A **finite Markov Chain** has a set of states S and a finite matrix M

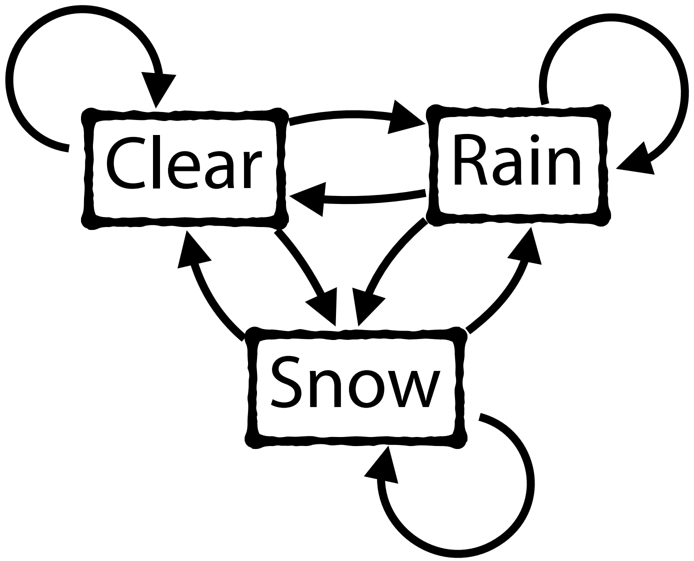


$$S = \{Clear, Rain, Snow\}$$

$$M = \begin{array}{cc} & \begin{array}{ccc} C & R & S \end{array} \\ & \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix} \begin{array}{c} C \\ R \\ S \end{array} \\ & \begin{array}{ccc} & T & \end{array} \end{array}$$

Markov Chain

Given a Markov Chain and an initial state, all subsequent states can be represented either as **a series of random states** or a transition probability.



$$M = \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix}$$

$X_0 = \text{Clear}$

$X_1 = \text{Clear}$

$X_2 = \text{Snow}$

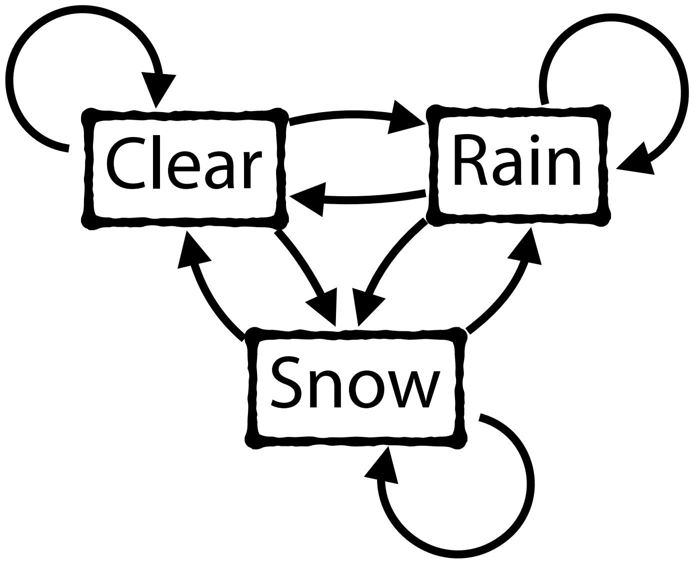
$X_3 = \text{Snow}$

$X_4 = \text{Snow}$

$X_5 = \text{Rain}$

Markov Chain

Given a Markov Chain and an initial state, all subsequent states can be represented either as a series of random states or a **transition probability.**



$$M = \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix}$$

$$M_0 = (.4 \quad .3 \quad .3)$$

$$M_1 = (.41 \quad .27 \quad .32)$$

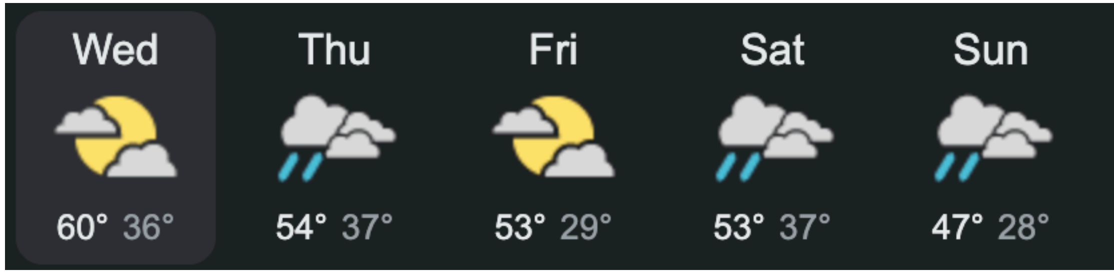
$$M_2 = (.404 \quad .263 \quad .333)$$

$$M_3 = (.401 \quad .259 \quad .340)$$

Markov Assumption

Probability of state x_k depends only on previous state x_{k-1}

Ex: Let $x = \{C, R, C, R, R\}$



$$P(x) = P(x_k, x_{k-1}, \ldots x_1)$$

$$= P(x_k \,|\, x_{k-1}, \ldots x_1) P(x_{k-1}, \ldots x_1)$$

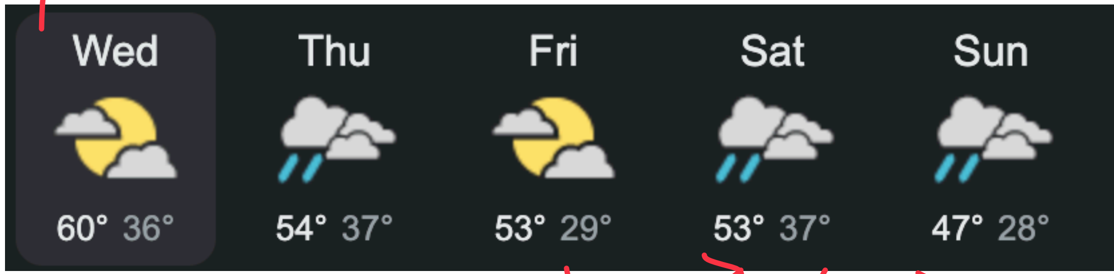
$$= P(x_k \,|\, x_{k-1}, \ldots x_1) P(x_{k-1} \,|\, x_{k-2}, \ldots x_1) \ldots P(x_2 \,|\, x_1) P(x_1)$$

$$P(x) \approx$$

Markov Assumption

Probability of state x_k depends only on previous state x_{k-1}

Ex: Let $x = \{C, R, C, R, R\}$



$$P(x) = P(x_k, x_{k-1}, \dots x_1)$$

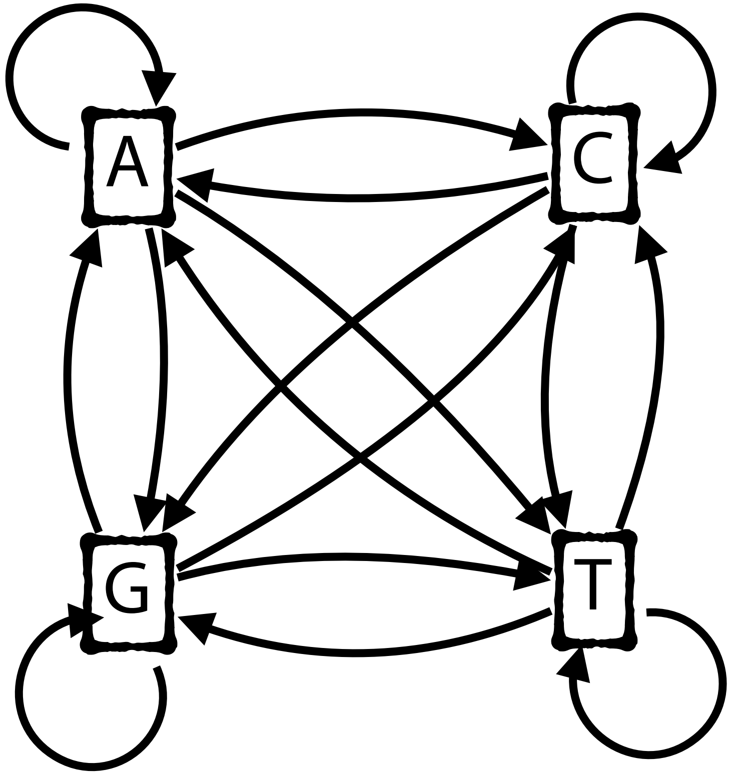
$$= P(x_k \mid x_{k-1}, \dots x_1) P(x_{k-1}, \dots x_1)$$

$$= P(x_k \mid x_{k-1}, \dots x_1) P(x_{k-1} \mid x_{k-2}, \dots x_1) \dots P(x_2 \mid x_1) P(x_1)$$

$$P(x) \approx P(x_k \mid x_{k-1}) P(x_{k-1} \mid x_{k-2}) \dots P(x_2 \mid x_1) P(x_1)$$

Markov Chain in Sequencing

Given a set of sequences, we can construct a model of transitions



$P(A \mid A)$ = # times AA occurs / # times AX occurs

Prev Curr

$P(C \mid A)$ = # times AC occurs / # times AX occurs

$P(G \mid A)$ = # times AG occurs / # times AX occurs

$P(T \mid A)$ = # times AT occurs / # times AX occurs

$P(A \mid C)$ = # times CA occurs / # times CX occurs

(etc)

where X is any base

Example by Ben Langmead

Markov Chain in Sequencing

Given a set of sequences, we can construct a model of transitions

X_{i-1}

```
>>> ins_conds, _ = markov_chain_from_dinucs(samp)
>>> print(ins_conds)
A [[ 0.19152248,  0.27252589,  0.39998803,  0.1359636 ],
C  [ 0.18921984,  0.35832388,  0.25467081,  0.19778547],
G  [ 0.17322219,  0.33142737,  0.35571338,  0.13963706],
T  [ 0.09509721,  0.33836493,  0.37567927,  0.19085859]]
          A            C            G            T
```

X_i

P(T | G)

Example by Ben Langmead

Markov Chain in Sequencing

```
>>> ins_conds, _ = markov_chain_from_dinucs(samp)
>>> print(ins_conds)
[[ 0.19152248,  0.27252589,  0.39998803,  0.1359636 ],
 [ 0.18921984,  0.35832388,  0.25467081,  0.19778547],
 [ 0.17322219,  0.33142737,  0.35571338,  0.13963706],
 [ 0.09509721,  0.33836493,  0.37567927,  0.19085859]]
```

Rows (X_{i-1}): A, C, G, T

Columns (X_i): A, C, G, T

$x = \text{GATC}$

$$P(x) = P(x_4 \mid x_3)\, P(x_3 \mid x_2)\, P(x_2 \mid x_1)\, P(x_1)$$

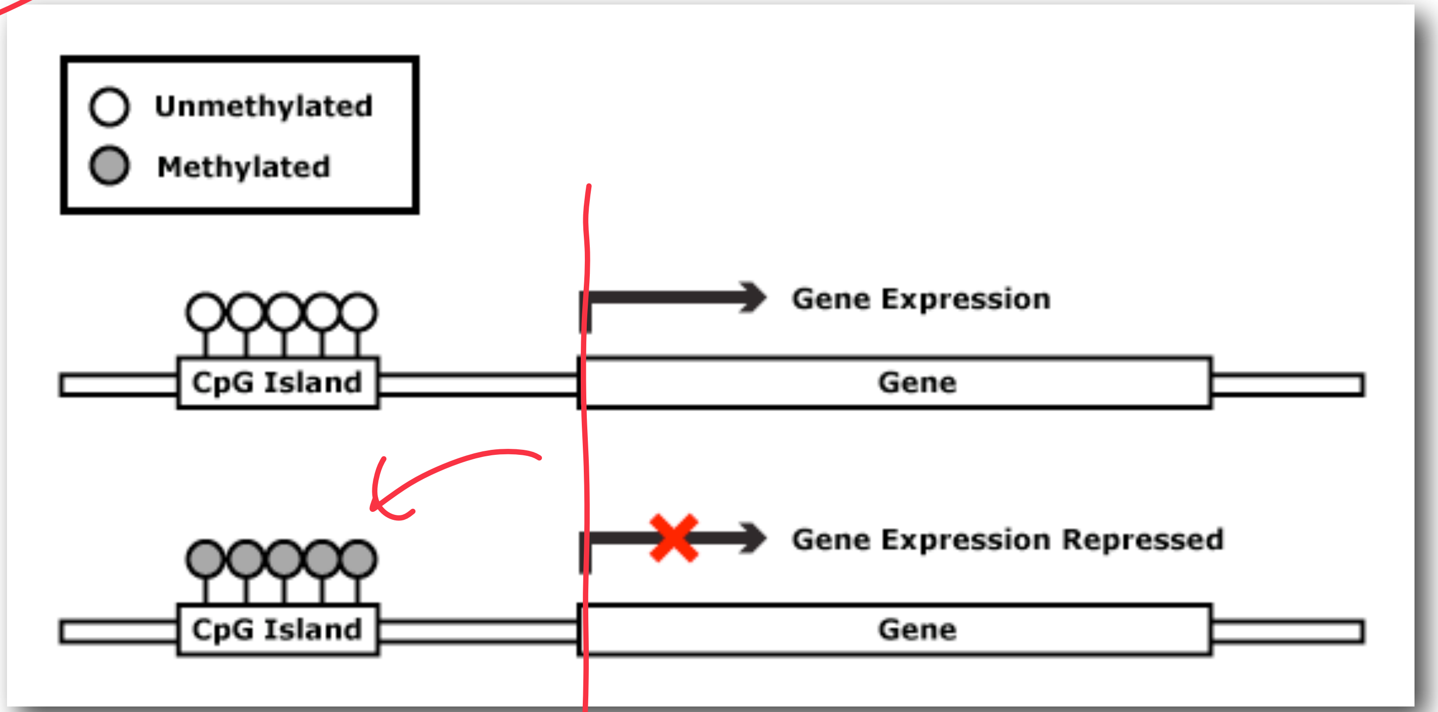
$$P(x) = P(C \mid T)\, P(T \mid A)\, P(A \mid G)\, P(G) = 0.33836493 * 0.1359636 * 0.17322219 * 0.25 = 0.001992$$

Example by Ben Langmead

Markov Chain in Sequencing

We can use this same approach to predict a *label* in our sequences as well

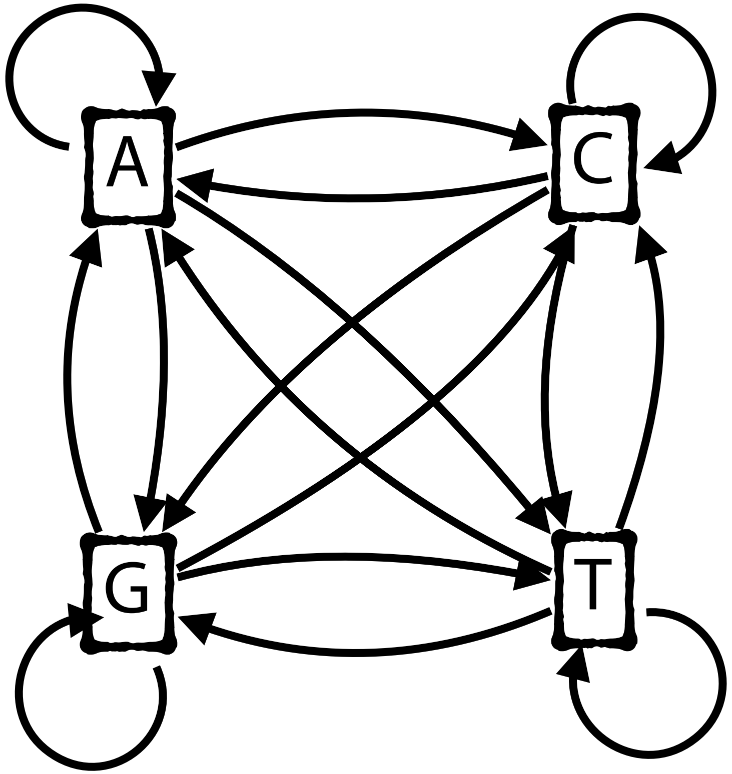
CpG island: part of the genome where CG occurs particularly frequently



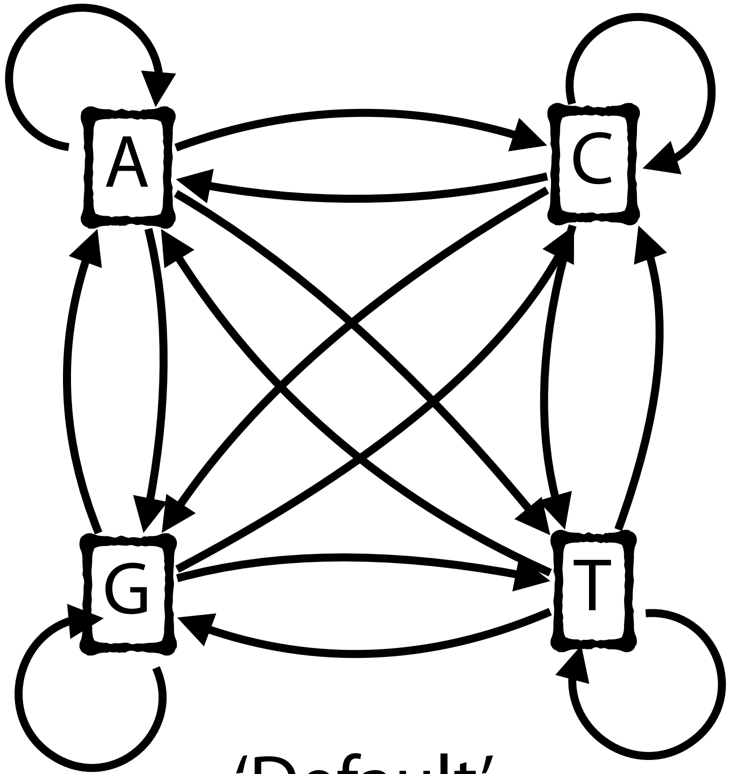
Example by Ben Langmead

Markov Chain in Sequencing

To predict a *label* of a sequencing region, make a Markov chain for both!



CpG Island

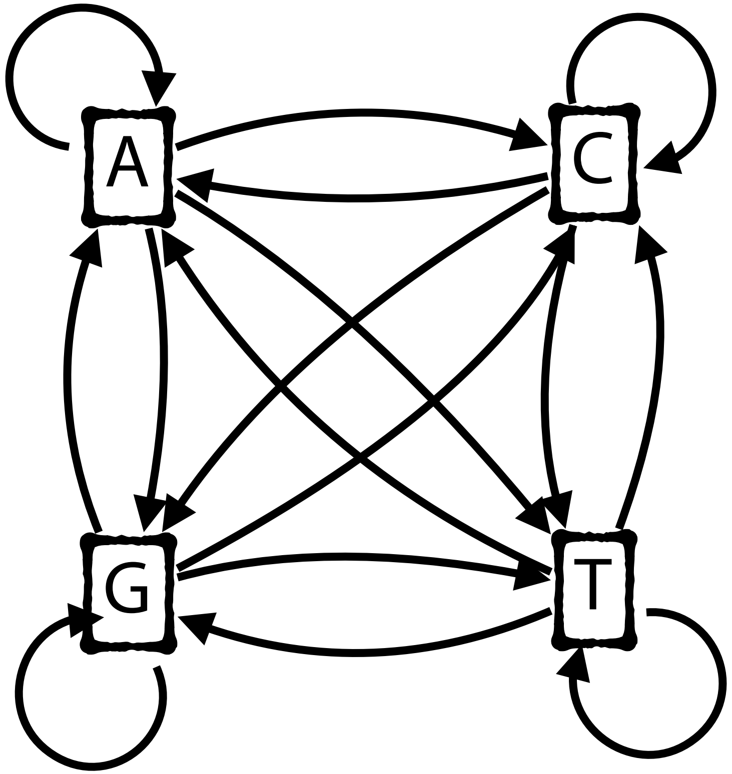


'Default'

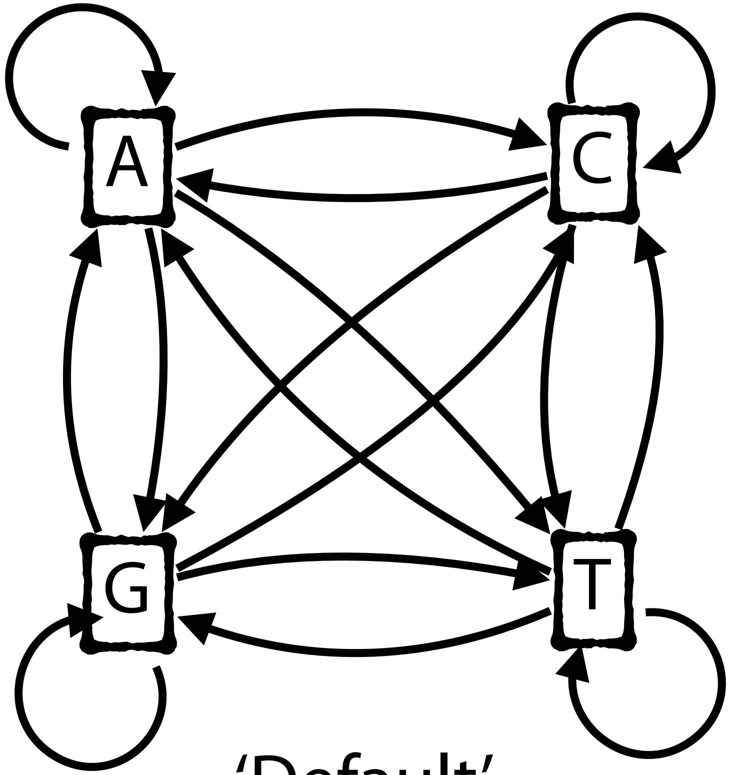
Example by Ben Langmead

Markov Chain in Sequencing

To predict a *label* of a sequencing region, make a Markov chain for both!



CpG Island



'Default'

Use *ratio:* $\dfrac{P(\,x\,)\text{ from CpG model}}{P(\,x\,)\text{ from Default model}}$

Example by Ben Langmead

Markov Chain in Sequencing

To predict a *label* of a sequencing region, make a Markov chain for both!

Take log, get a *log ratio*: $$S(x) = \log \frac{P(x) \text{ inside CpG}}{P(x) \text{ outside CpG}}$$

$$\log P(x) \approx \log [\, P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, P(x_1)\,]$$

$$= \log P(x_k \mid x_{k-1}) + \log P(x_{k-1} \mid x_{k-2}) + \ldots$$

product becomes **sum**

$$= \sum_{i=2}^{k} \log P(x_i \mid x_{i-1}) + \log P(x_1)$$

If inside more probable than outside, fraction is > 1, log ratio is > 0.
Otherwise, fraction is ≤ 1 and log ratio is ≤ 0.

Markov Chain in Sequencing

To predict a *label* of a sequencing region, make a Markov chain for both!

Take log, get a *log ratio*: $$S(x) = \log \frac{P(x) \text{ inside CpG}}{P(x) \text{ outside CpG}}$$

$$\log P(x) \approx \log [\, P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, P(x_1)\,]$$

$$= \log P(x_k \mid x_{k-1}) + \log P(x_{k-1} \mid x_{k-2}) + \ldots$$

product becomes **sum**

$$= \sum_{i=2}^{k} \log P(x_i \mid x_{i-1}) + \log P(x_1)$$

If inside more probable than outside, fraction is > 1, log ratio is > 0.
Otherwise, fraction is ≤ 1 and log ratio is ≤ 0.

```
>>> cpg_conds, _ = markov_chain_from_dinucs(samp_cpg)
>>> print(cpg_conds)
[[ 0.19152248,  0.27252589,  0.39998803,  0.1359636 ],
 [ 0.18921984,  0.35832388,  0.25467081,  0.19778547],
 [ 0.17322219,  0.33142737,  0.35571338,  0.13963706],
 [ 0.09509721,  0.33836493,  0.37567927,  0.19085859]]
>>> default_conds, _ = markov_chain_from_dinucs(samp_def)
>>> print(default_conds)
[[ 0.33804066,  0.17971034,  0.23104207,  0.25120694],
 [ 0.37777025,  0.25612117,  0.03987225,  0.32623633],
 [ 0.30257815,  0.20326794,  0.24910719,  0.24504672],
 [ 0.21790184,  0.20942905,  0.2642385 ,  0.3084306 ]]
>>> print(np.log2(cpg_conds) - np.log2(def_conds))
[[-0.87536356,  0.59419041,  0.81181564, -0.85527103],
 [-0.98532149,  0.49570561,  2.64256972, -0.7126391 ],
 [-0.79486196,  0.68874785,  0.51821792, -0.79549511],
 [-1.22085697,  0.73036913,  0.48119354, -0.69736839]]
```

CpG: rows A, C, G, T

Default: rows A, C, G, T

Log ratio: rows A, C, G, T

Columns: A C G T

Markov Chain in Sequencing

x_{i-1}

```
>>> print(np.log2(cpg_conds) - np.log2(def_conds))
A [[-0.87536356,  0.59419041,  0.81181564, -0.85527103],
C  [-0.98532149,  0.49570561,  2.64256972, -0.7126391 ],
G  [-0.79486196,  0.68874785,  0.51821792, -0.79549511],
T  [-1.22085697,  0.73036913,  0.48119354, -0.69736839]]
          A            C            G            T
```

x_i

Likely "default"

$x = \text{GATC}$

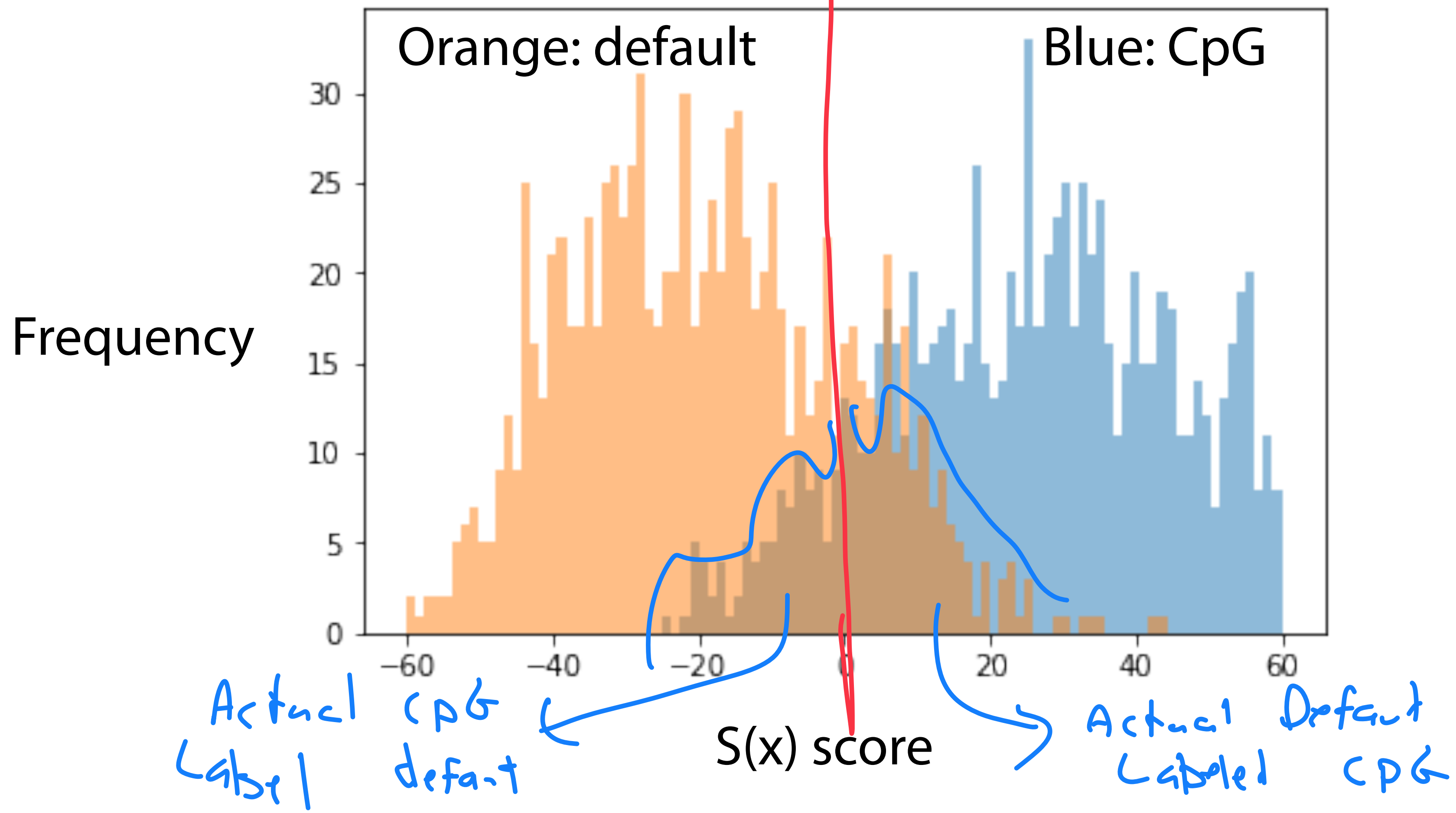
$$P(x) = P(x_4 \mid x_3)\, P(x_3 \mid x_2)\, P(x_2 \mid x_1)\, P(x_1)$$

$$P(x) = P(C \mid T)\, P(T \mid A)\, P(A \mid G)\, P(G) \quad = \quad 0.73036913 + (-0.85527103) + (-0.79486196) = -0.919763$$

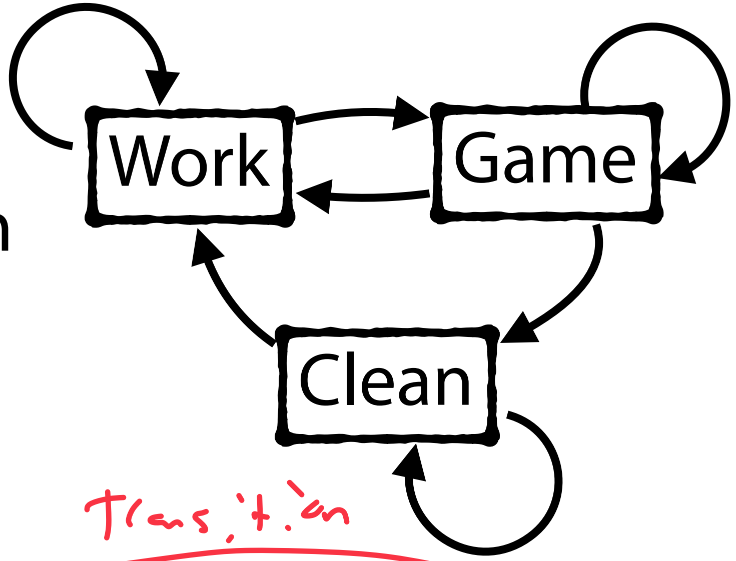
Example by Ben Langmead

Markov Chain in Sequencing

Drew 1,000 100-mers from inside CpG islands and another 1,000 from outside, and calculated S(x) for all



Markov Chain Matrix



If I'm working at time 0, what is probability that I'm working at time t?

start

end

Claim: $Pr(X_t = v \mid \boxed{X_0 = u}) = M^t[u, v]$

work　　work

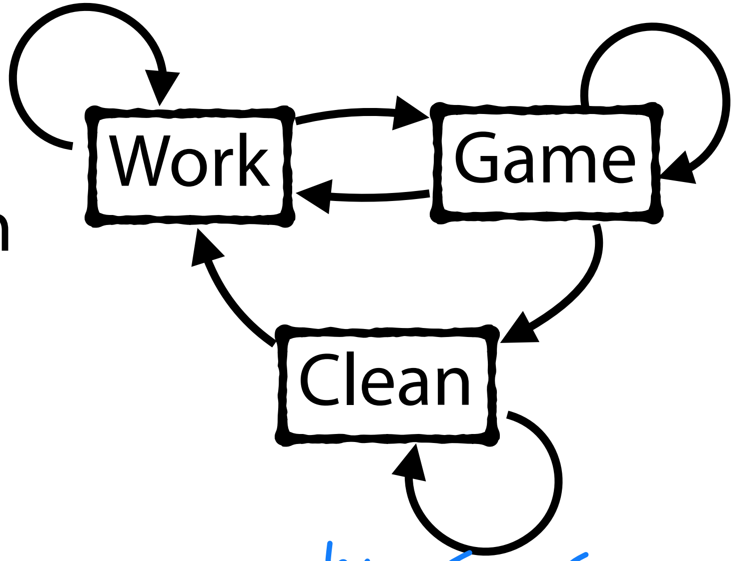
transition

$$M = \begin{pmatrix} .4 & .6 & 0 \\ .1 & .6 & .3 \\ .5 & 0 & .5 \end{pmatrix}$$

Base Case:

T=1: $Pr(X_1 = w \mid X_0 = w) = M^1[w, w]$

Markov Chain Matrix



If I'm working at time 0, what is probability that I'm working at time t?

Claim: $Pr(X_t = v \mid X_0 = u) = M^t[u, v]$

Base Case:

T=2: $Pr(X_2 = w \mid X_0 = w)$ 0.4

$= Pr(X_2 = w \mid X_1 = w) \cdot Pr(X_1 = w \mid X_0 = w)$ 0.4   0.16

$+ Pr(X_2 = w \mid X_1 = G) \cdot Pr(X_1 = G \mid X_0 = w)$ — 0.6   0.1   0.06

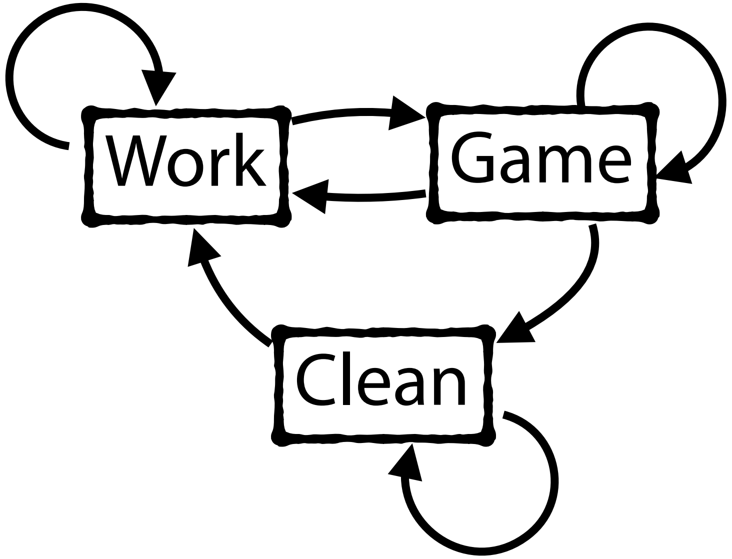
$+ Pr(X_2 = w \mid X_1 = C) \cdot Pr(X_1 = C \mid X_0 = w)$ 1   0

0.22

$$M = \begin{array}{c c} \begin{array}{ccc} w & G & C \end{array} & \\ \begin{pmatrix} .4 & .6 & 0 \\ .1 & .6 & .3 \\ .5 & 0 & .5 \end{pmatrix} & \begin{array}{c} w \\ G \\ C \end{array} \end{array}$$

$$M^2 = \begin{pmatrix} .22 & .6 & .18 \\ .25 & .42 & .33 \\ .45 & 0.3 & .25 \end{pmatrix}$$

Markov Chain Matrix



Claim: $Pr(X_t = v \mid X_0 = u) = M^t[u, v]$

Induction:

Assume $Pr(X_{t-1} = v \mid X_0 = u) = M^{t-1}[u, v]$.

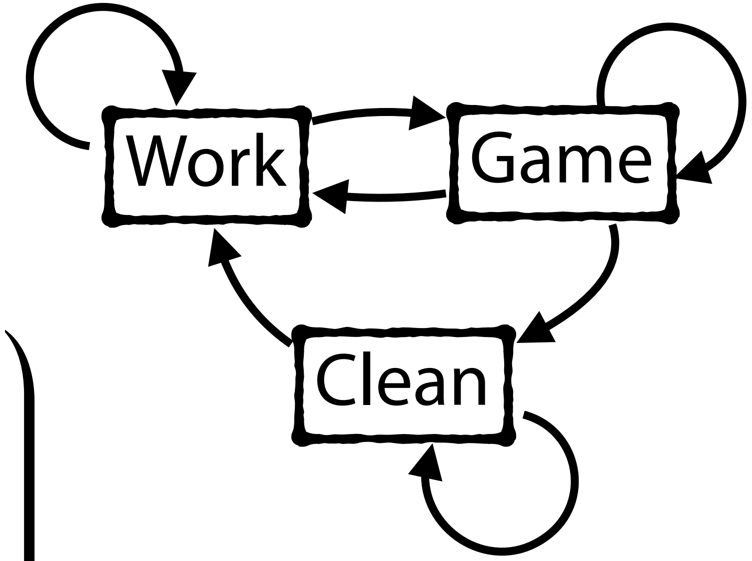
Show holds for $Pr(X_t = w \mid X_0 = u) = M^t[u, w]$

$$M = \begin{pmatrix} .4 & .6 & 0 \\ .1 & .6 & .3 \\ .5 & 0 & .5 \end{pmatrix}$$

By Markov Assumption — trivial!

The same logic (and math) for finding T=2 applies here

Markov Chain Matrix



What happens as $t \to \infty$?

$$M = \begin{pmatrix} .4 & .6 & 0 \\ .1 & .6 & .3 \\ .5 & 0 & .5 \end{pmatrix} \qquad M^3 = \begin{pmatrix} .238 & .492 & .270 \\ .307 & .402 & .291 \\ .335 & .450 & .215 \end{pmatrix}$$

$$M^{10} = \begin{pmatrix} .2940 & .4413 & .2648 \\ .2942 & .4411 & .2648 \\ .2942 & .4413 & .2648 \end{pmatrix}$$

$$M^{60} = \begin{pmatrix} .2941 & .4412 & .2647 \\ .2941 & .4412 & .2647 \\ .2941 & .4412 & .2647 \end{pmatrix}$$

Markov Chain Stationary Distribution

A probability vector π is called a **stationary distribution** for a Markov Chain if it satisfies the stationary equation: $\pi = \pi M$

$$
\begin{array}{cc}
 & \text{end} \\
 & \begin{array}{ccc} W & G & C \end{array} \\
\text{start} & \\
\begin{array}{c} W \\ G \\ C \end{array} & M = \begin{pmatrix} .4 & .6 & 0 \\ .1 & .6 & .3 \\ .5 & 0 & .5 \end{pmatrix}
\end{array}
$$

$$\pi[W] = .4\pi[W] + .1\pi[G] + .5\pi[C]$$

$$\pi[\overset{G}{\cancel{S}}] = .6\pi[W] + .6\pi[G] + \cancel{0\pi[C]}$$

$$\pi[\overset{C}{\cancel{Z}}] = \cancel{0\pi[W]} + .3\pi[G] + .5\pi[C]$$

$.6W = .1G + .5C$

$.4G = .6W$

$.5C = .3G$

$W + G + C = 1$

$\rightarrow$ $W = \frac{10}{34}$

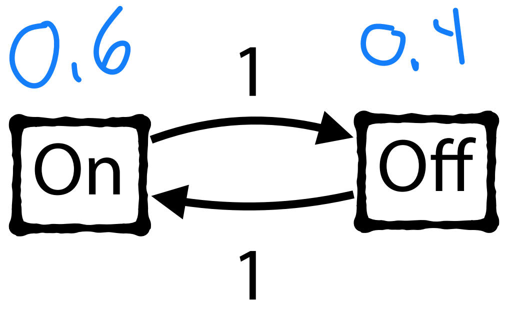
$G = \frac{15}{34}$

$C = 9/34$

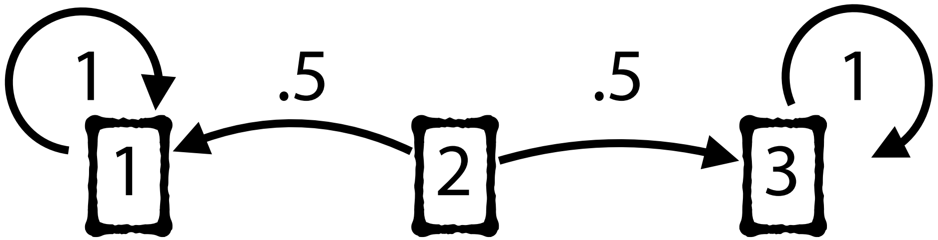
$\rightarrow$ $[.294, .441, .265]$

Markov Chain Stationary Distribution

Stationary distributions can be calculated using the system of equation (and that all probabilities sum to 1). **But not every Markov Chain has a steady state (and some have infinitely many)!**



If on/off = 0.5



∞ # of steady states

Markov Chain Monte Carlo

There are ways to prove whether a Markov Chain has a stationary distribution, but several algorithms exist that approximate!

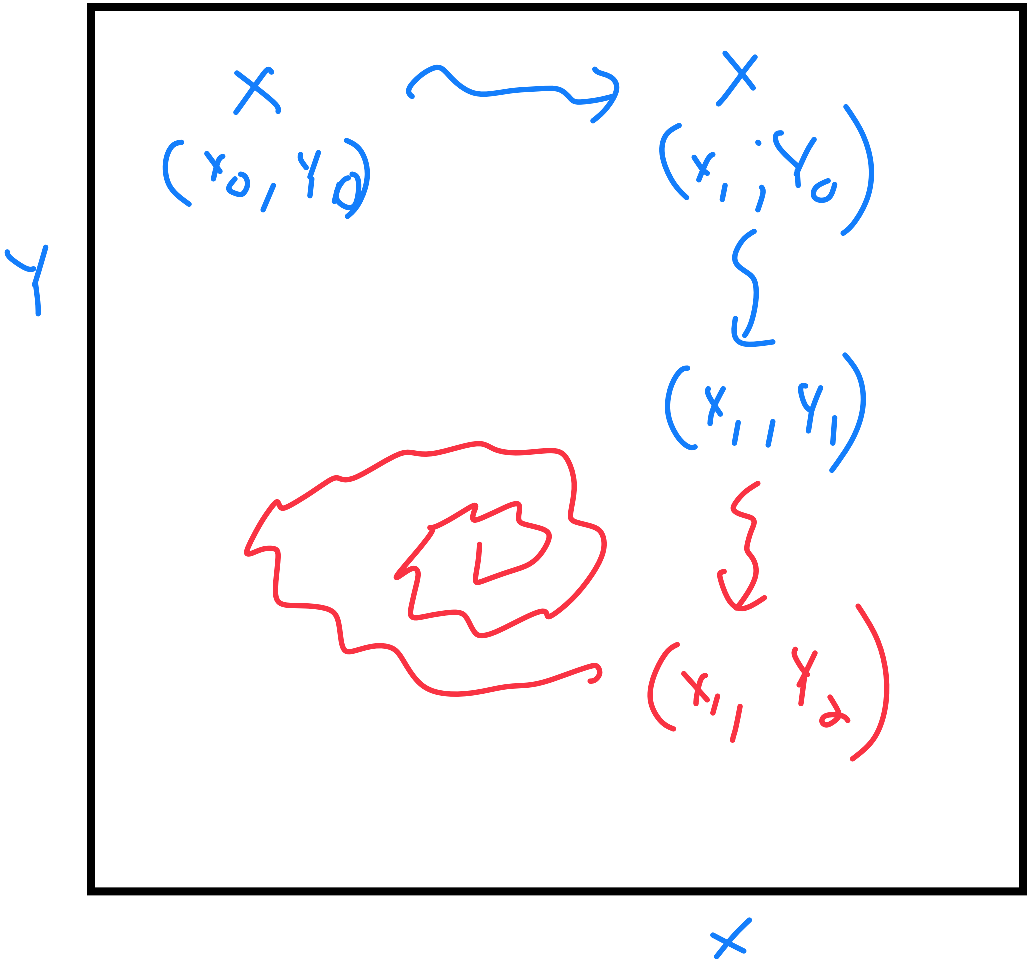
Gibbs Sampling:

Randomly assign values to a probability vector $\pi_{t=0} = (\theta_0, \theta_1, \ldots, \theta_{d-1})$.

Compute π_{t+1} for each i, $0 \leq i < d$:

Update value θ_i based on $(\theta_0, \ldots, \theta_{i-1})_{t+1}, (\theta_{i+1}, \ldots, \theta_{d-1})_t$

Repeat for different ordering of i



Markov Chain Monte Carlo

A single step of a 3D Gibbs Sampling:

Given $\pi_t = (X_t,\ Y_t,\ Z_t)$

Compute π_{t+1} by updating each value one at a time:

$$X_{t+1} = M[X,X]X_t + M[Y,X]Y_t + M[Z,X] * Z_t$$

$$Y_{t+1} = M[X,Y]\boxed{X_{t+1}} + M[Y,Y]Y_t + M[Z,Y] * Z_t$$

$$Z_{t+1} = M[X,Z]\boxed{X_{t+1}} + M[Y,Z]\boxed{Y_{t+1}} + M[Z,Z] * Z_t$$

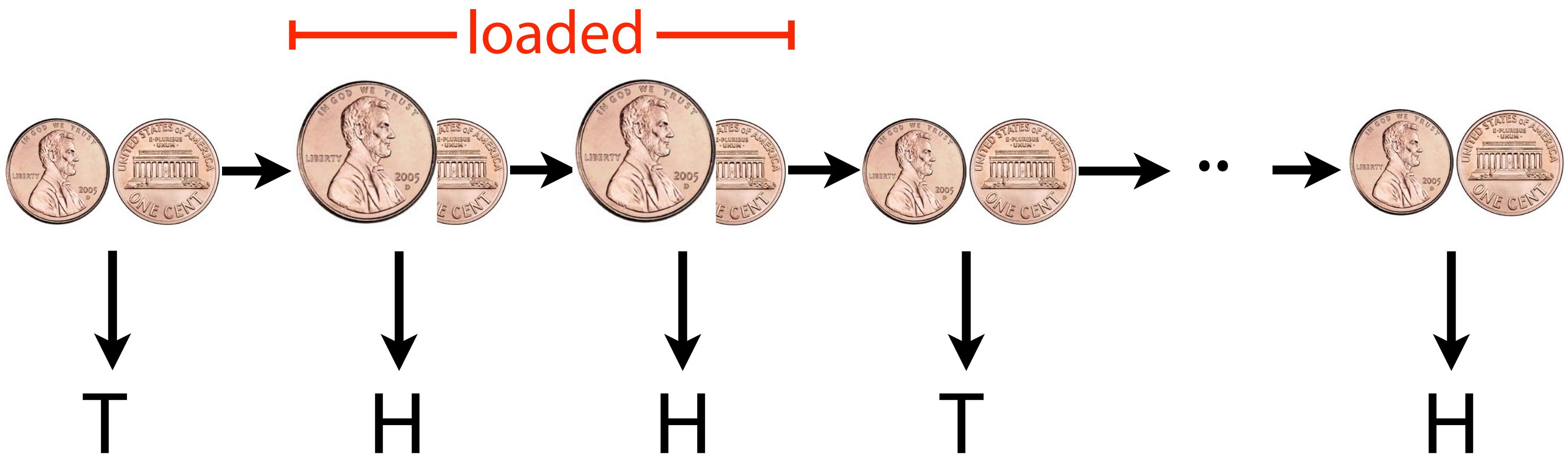
Now have $\pi_{t+1} = (X_{t+1},\ Y_{t+1},\ Z_{t+1})$

Hidden Markov Models

In the real world, we often don't know the underlying markov chain!

Instead, we have observations that can be used to predict our current state.

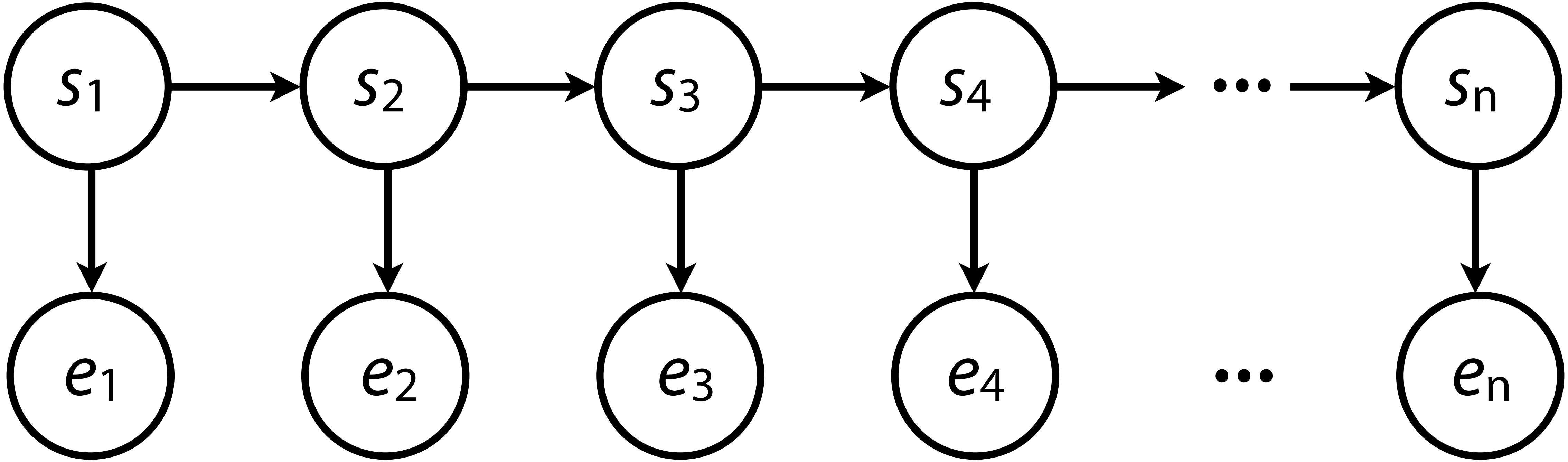
Ex: Repeated coin flips but *sometimes* I cheat and use a fixed coin.



Hidden Markov Models

Unobserved States

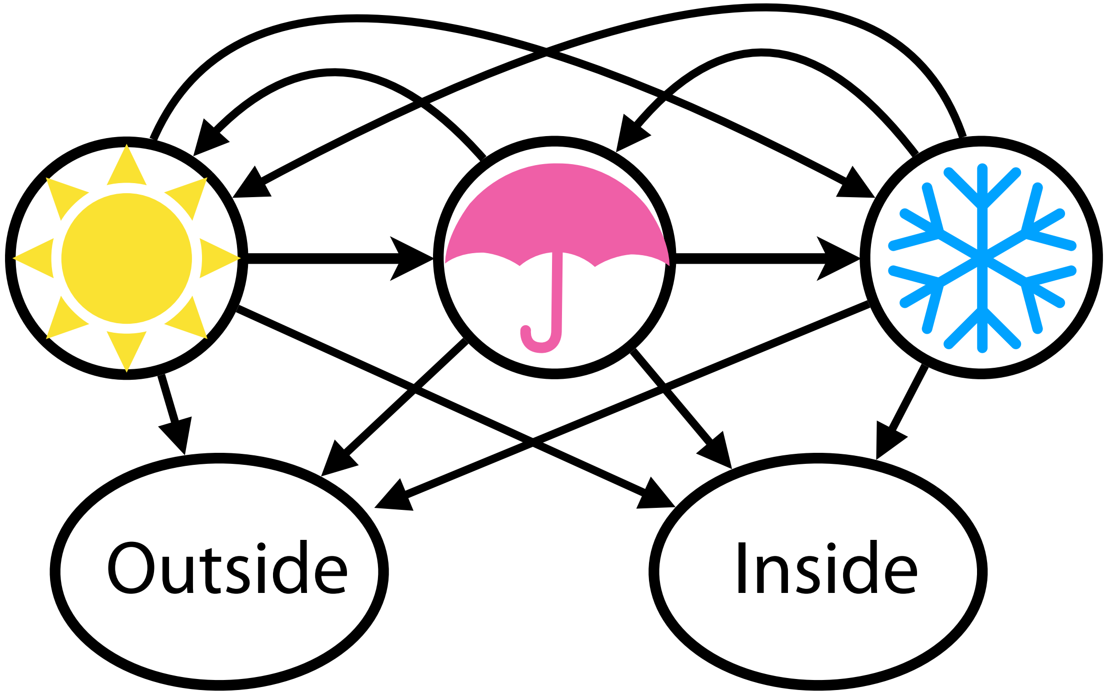
fair coin
Loaded coin



Observed Emissions

Coin heads / tails

Hidden Markov Models

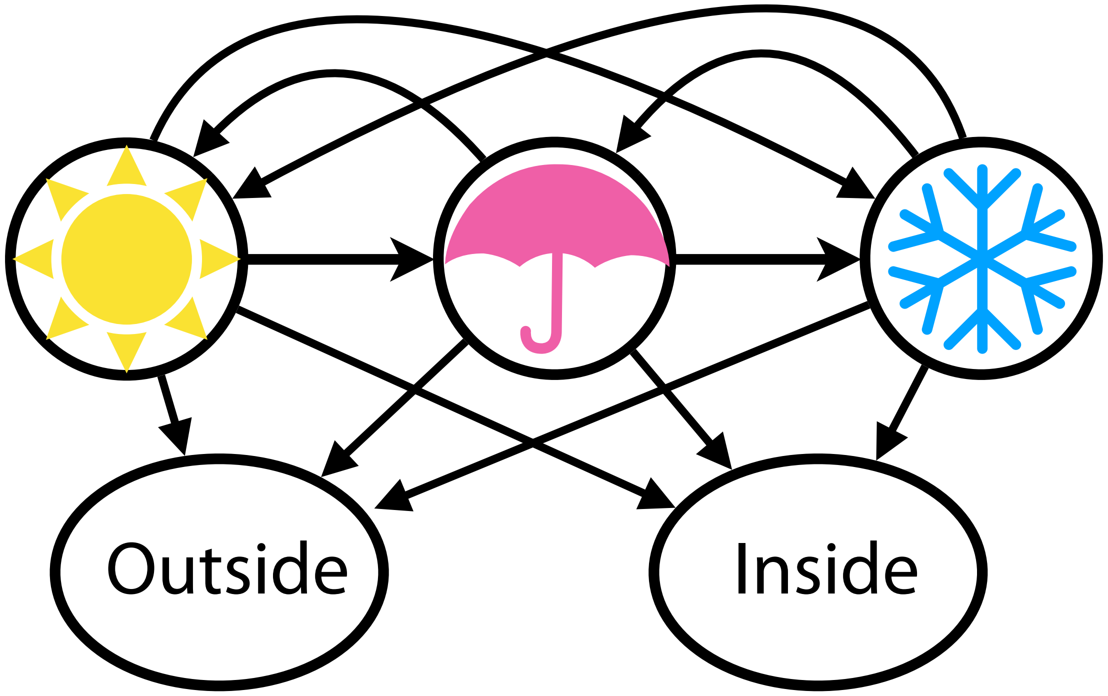


$$M = \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix} \quad E = \begin{pmatrix} .8 & .2 \\ .3 & .7 \\ .5 & .5 \end{pmatrix}$$

Pr({O, I, O} | {C, R, S})?

Pr({O, I, O}, {C, R, S} | P(T_0 = C) = 0.4)?

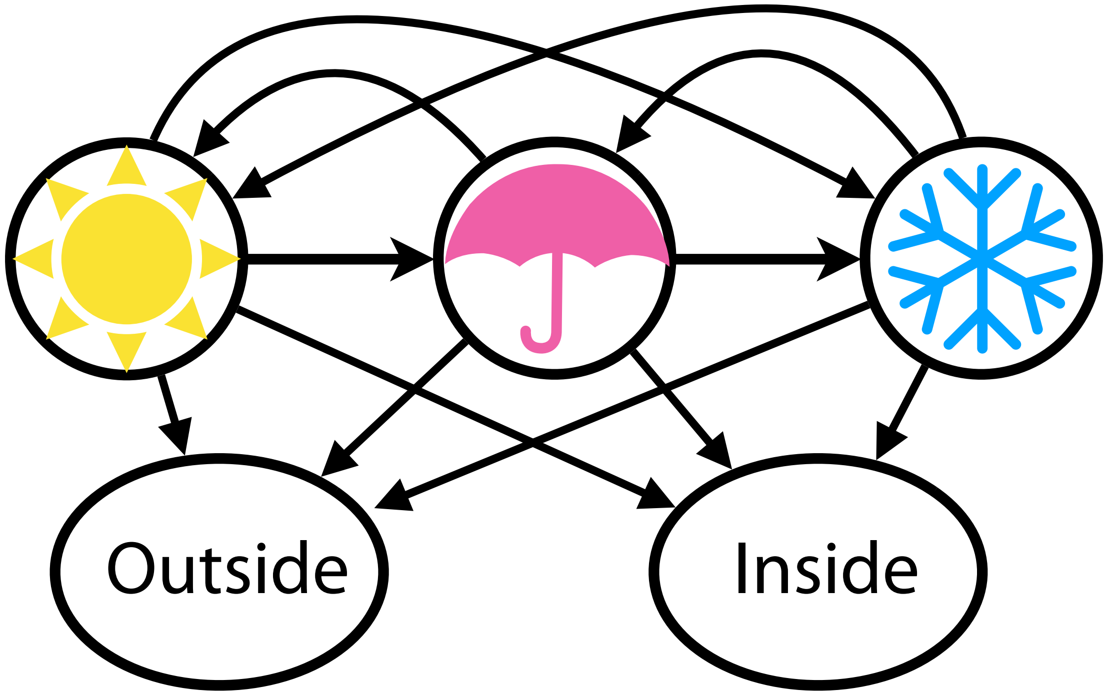
Hidden Markov Models



$$M = \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix} \quad E = \begin{pmatrix} .8 & .2 \\ .3 & .7 \\ .5 & .5 \end{pmatrix}$$

Pr({O, I, O})?

Hidden Markov Models



$$M = \begin{pmatrix} .5 & .3 & .2 \\ .5 & .4 & .1 \\ .2 & .1 & .7 \end{pmatrix} \quad E = \begin{pmatrix} .8 & .2 \\ .3 & .7 \\ .5 & .5 \end{pmatrix}$$

If I go outside for three days, what was the most likely weather?

exact O, O, O | C C C

calc O, O, O | C C R

vs

Clear
Rain
Snow

Viterbi

O O O

Viterbi Algorithm

We can brute force all possible combinations…

… or we can use the Markov Assumption with Dynamic Programming

States

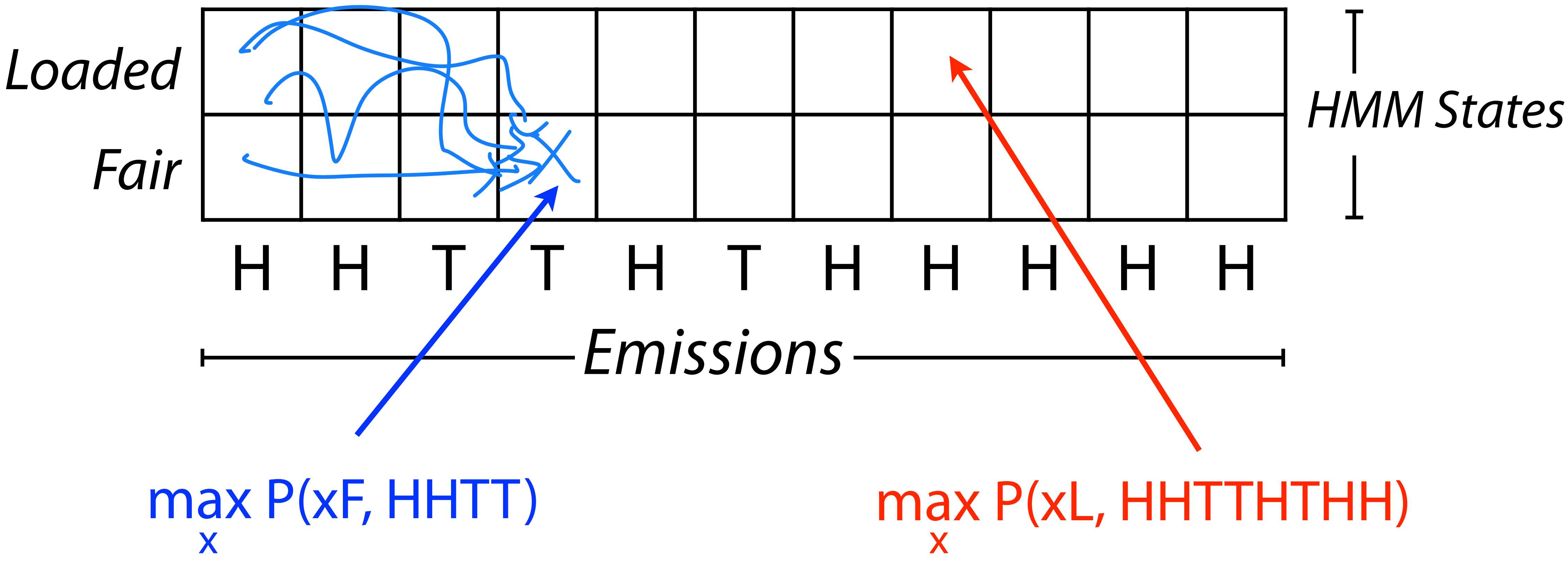
$$M = \begin{array}{c} \text{Load} \\ \text{Fair} \end{array} \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix}$$

Emission

$$E = \begin{array}{cc} & \begin{array}{cc} H & T \end{array} \\ \begin{array}{c} L \\ F \end{array} & \begin{pmatrix} .8 & .2 \\ .5 & .5 \end{pmatrix} \end{array}$$

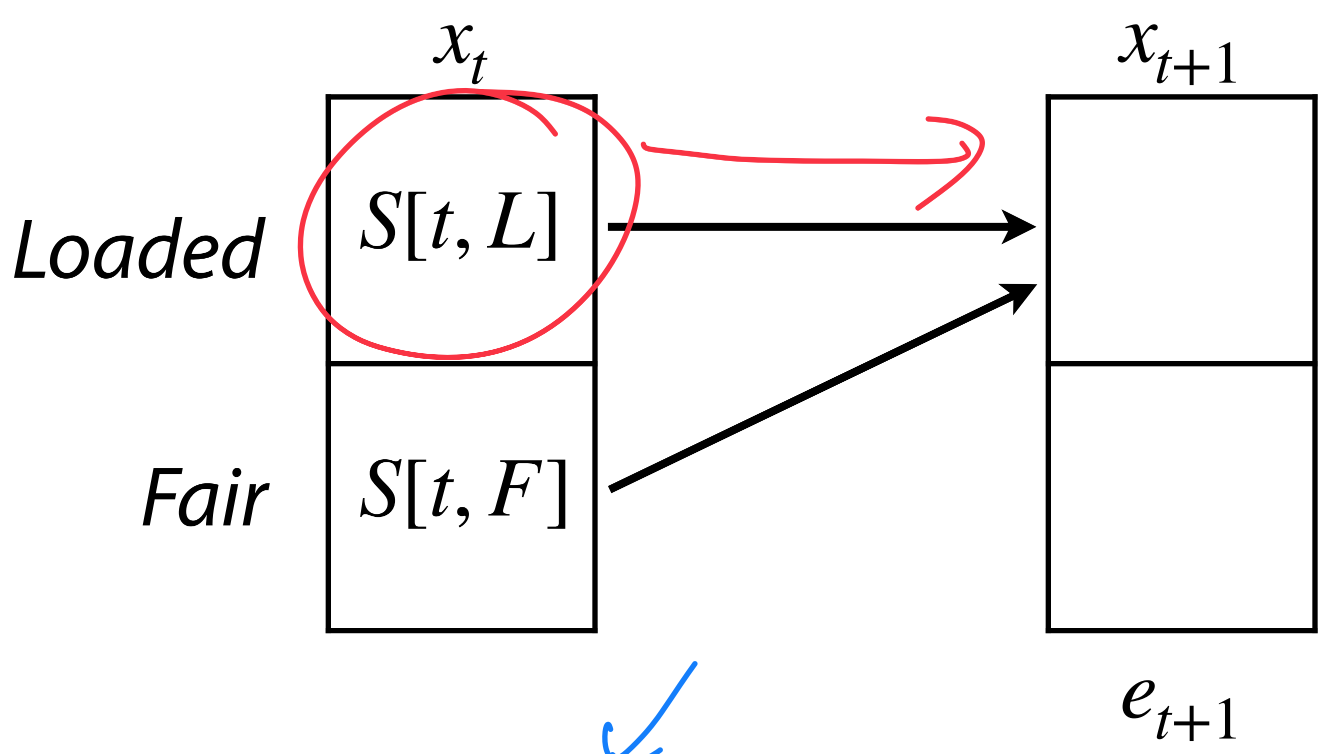
Example by Ben Langmead

Viterbi Algorithm



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

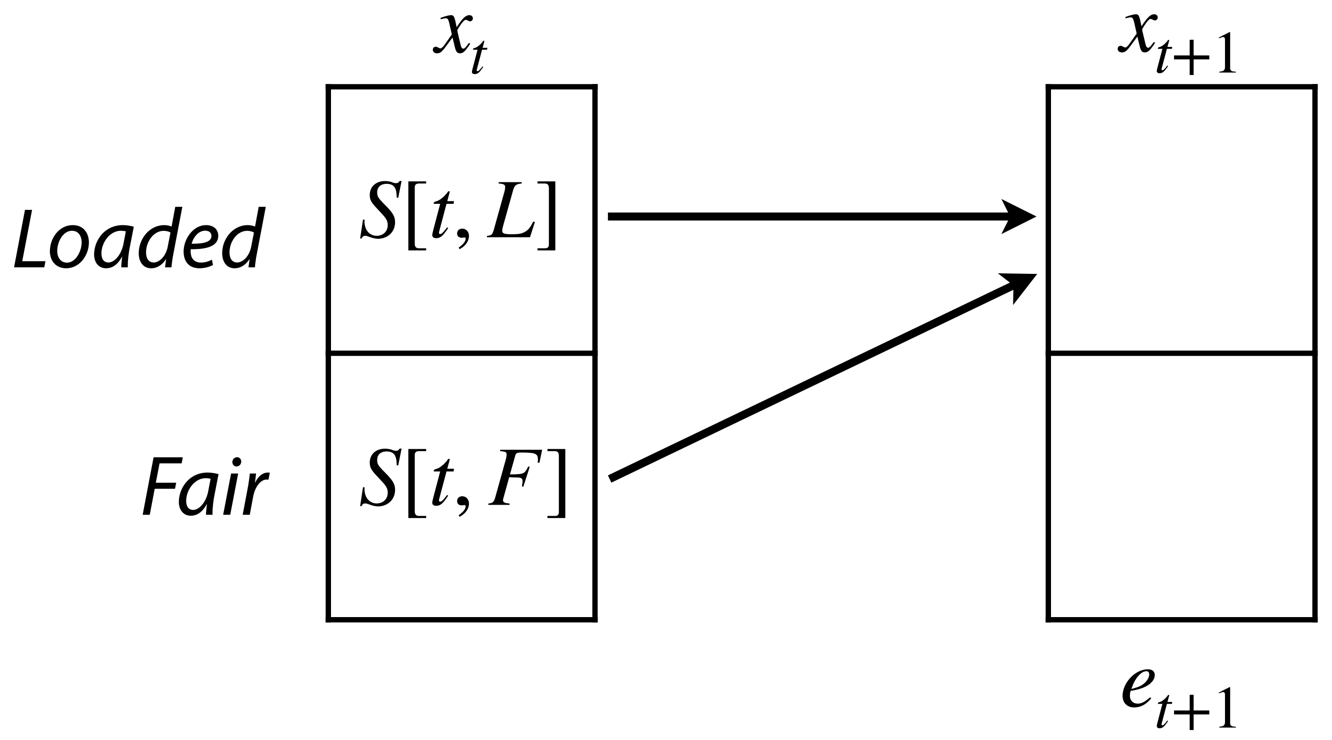
Viterbi Algorithm



$$S[t+1, L] = s[t, L] * M[L|L] \cdot E[e_{t+1}|L]$$

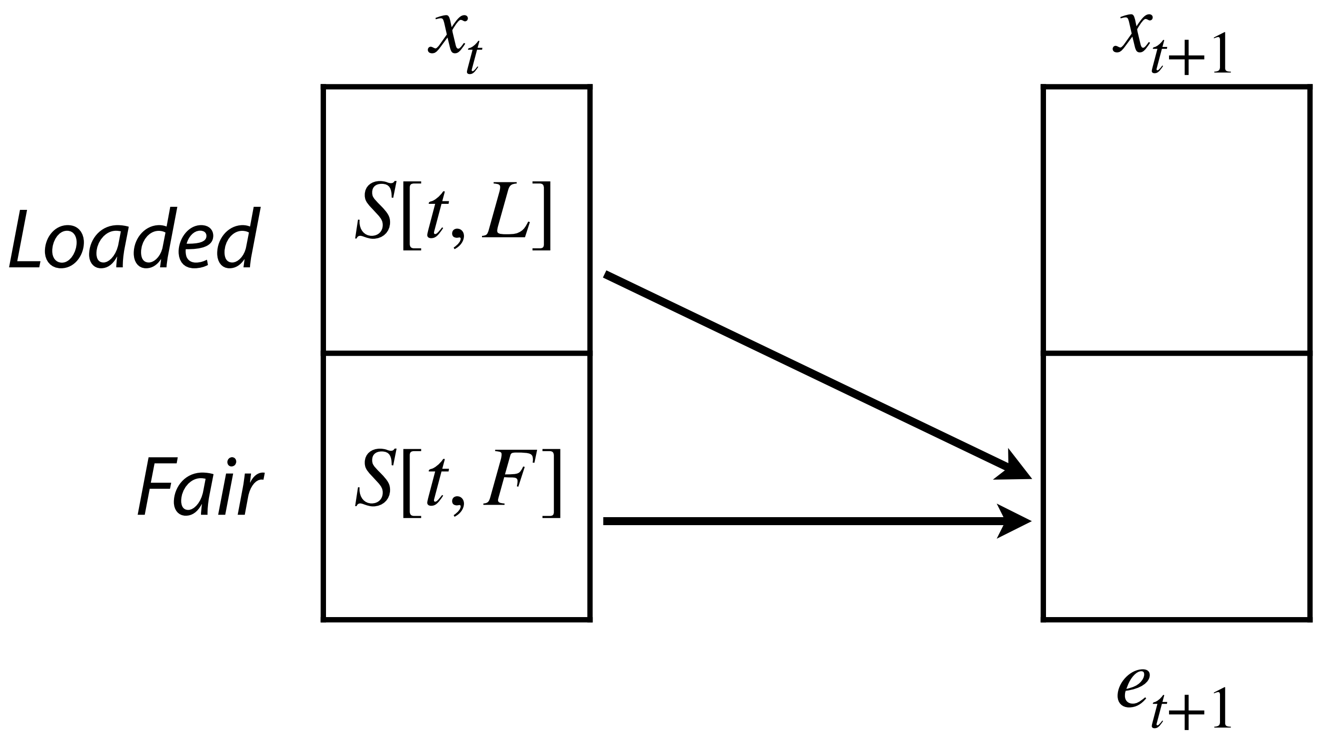
$$s[t, F] * M[L|F]$$

Viterbi Algorithm



$$S[t+1,\ L] = \ max \begin{cases} S[t,\ L] * M[L \mid L] * E[e_{t+1} \mid L] \\ S[t,\ F] * M[L \mid F] * E[e_{t+1} \mid L] \end{cases}$$

Viterbi Algorithm



$$S[t + 1,\ F] =$$

Viterbi Algorithm

Loaded											
Fair											
	H	H	T	T	H	T	H	H	H	H	H

$$M = \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix}$$

$$E = \begin{pmatrix} .8 & .2 \\ .5 & .5 \end{pmatrix}$$

Assume we start with Fair/Loaded with equal probability

$$S[0, L] = 0.5 \cdot \boldsymbol{E(H \mid L)} \qquad S[0, F] = 0.5 \cdot \boldsymbol{E(H \mid F)}$$

$$= 0.5 \cdot \boldsymbol{0.8} \qquad\qquad = 0.5 \cdot \boldsymbol{0.5}$$

Viterbi Algorithm

$$M = \begin{array}{cc} & \begin{array}{cc} L & F \end{array} \\ \begin{array}{c} L \\ F \end{array} & \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix} \end{array}$$

$$E = \begin{array}{cc} & \begin{array}{cc} H & T \end{array} \\ \begin{array}{c} L \\ F \end{array} & \begin{pmatrix} .8 & .2 \\ .5 & .5 \end{pmatrix} \end{array}$$

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

$$S[1, L] = \text{Max}\left(\begin{array}{l} 0.4 \cdot M[L|L] \cdot E[H|L] = 0.4 \cdot 0.6 \cdot 0.8 \\ 0.25 \cdot M[L|F] \cdot E[H|L] = 0.25 \cdot 0.4 \cdot 0.8 \end{array} \right.$$

Viterbi Algorithm

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

$$M = \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix}$$

$$E = \begin{pmatrix} .8 & .2 \\ .5 & .5 \end{pmatrix}$$

c m[F|L] e

$S[1, F] =$ 0.4 · 0.4 · 0.5 = 0.08

vs

0.25 · 0.6 · 0.5 = 0.075

Viterbi Algorithm

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

$$M = \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix}$$

$$E = \begin{pmatrix} .8 & .2 \\ .5 & .5 \end{pmatrix}$$

Viterbi Algorithm

These get small very fast— use log_2 scaling

-1.32	-2.38	-5.44	-8.35	-8.08	-11.1	-11.6	-12.6	-13.7	-14.7	-15.8
-2	-3.64	-4.7	-6.4	-8.2	-9.9	-11.7	-13.4	-14.9	-16	-17
H	H	T	T	H	T	H	H	H	H	H

Traceback: Same as edit distance!

Start from largest value and remember 'where I came from'

Viterbi Algorithm

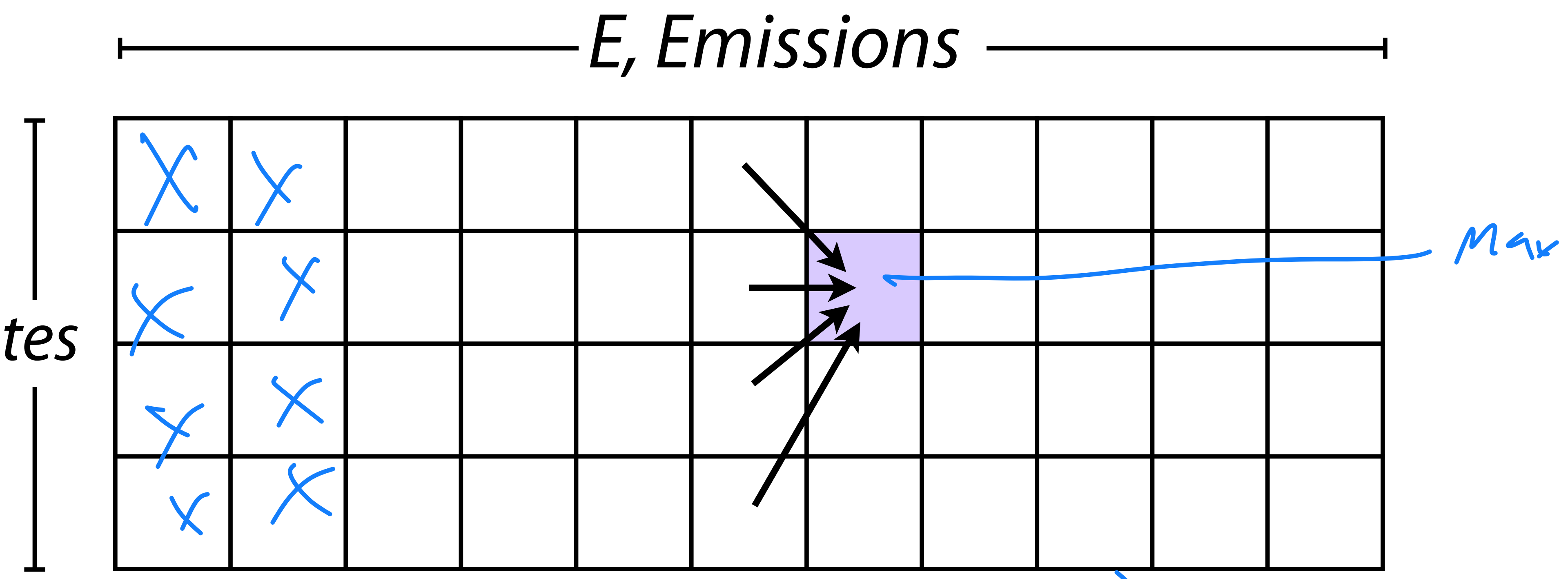
These get small — now log_2 scaled

-1.32	-2.38	-5.44	-8.35	-8.08	-11.1	-11.6	-12.6	-13.7	-14.7	-15.8
-2	-3.64	-4.7	-6.4	-8.2	-9.9	-11.7	-13.4	-14.9	-16	-17
H	H	T	T	H	T	H	H	H	H	H

Traceback: Same as edit distance!

Start from largest value and remember 'where I came from'

Viterbi Algorithm



What is running time?

$$\underline{|S|} * (|S| \cdot |E|)$$