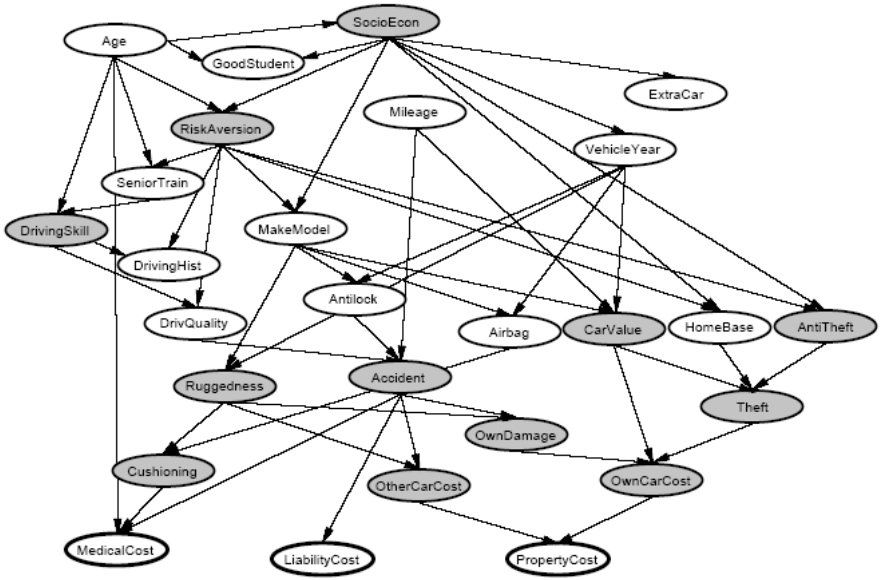
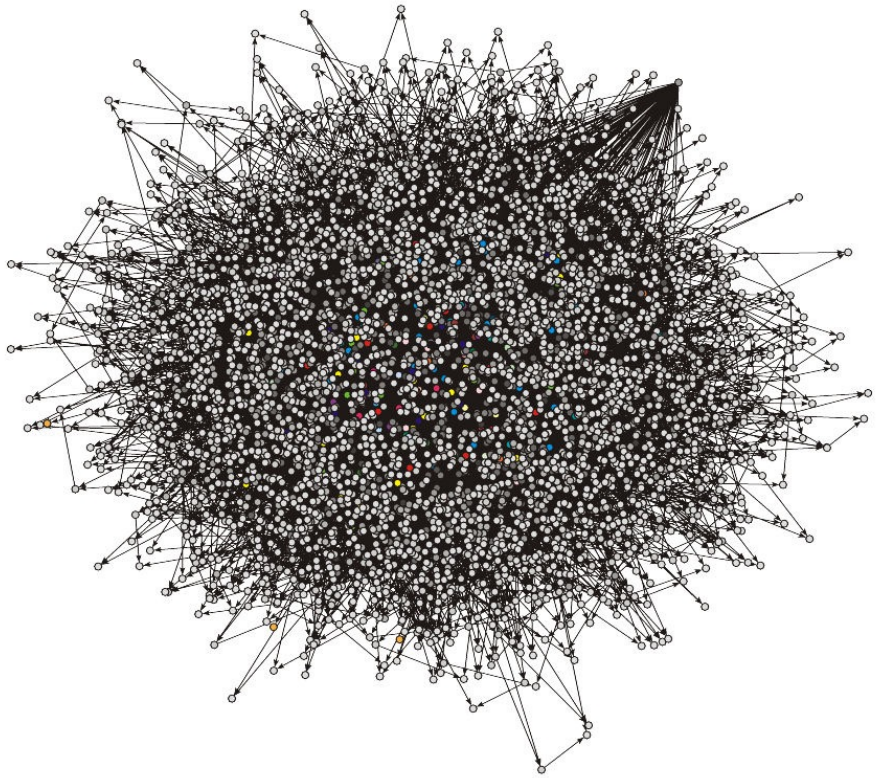
MCMC

Announcements

- A2 coming soon (sign up for teams)
- Midterm exam 10/26 6:30pm-7:45pm
  - Mostly multiple choice
  - Review questions posted
  - Online using Canvas
- Final exam
  - Friday 12/16 7:40pm-9:40pm
  - Online using Canvas

What about complicated Bayes Nets?





http://bi.snu.ac.kr/NRL/HBN/hbn.htm

Making inference tractable

- In practice, making inference tractable is a key challenge in applying graphical models to applications
- Typically, the options are:
  - **Exact inference** with **arbitrary conditional probability distributions**, but with a **simplified graphical structure**
  - **Exact inference** with **arbitrary graphical structure**, but **restricted conditional probability distributions**
  - **Arbitrary graphical structure** and **arbitrary conditional probability distributions,** but **approximate inference**

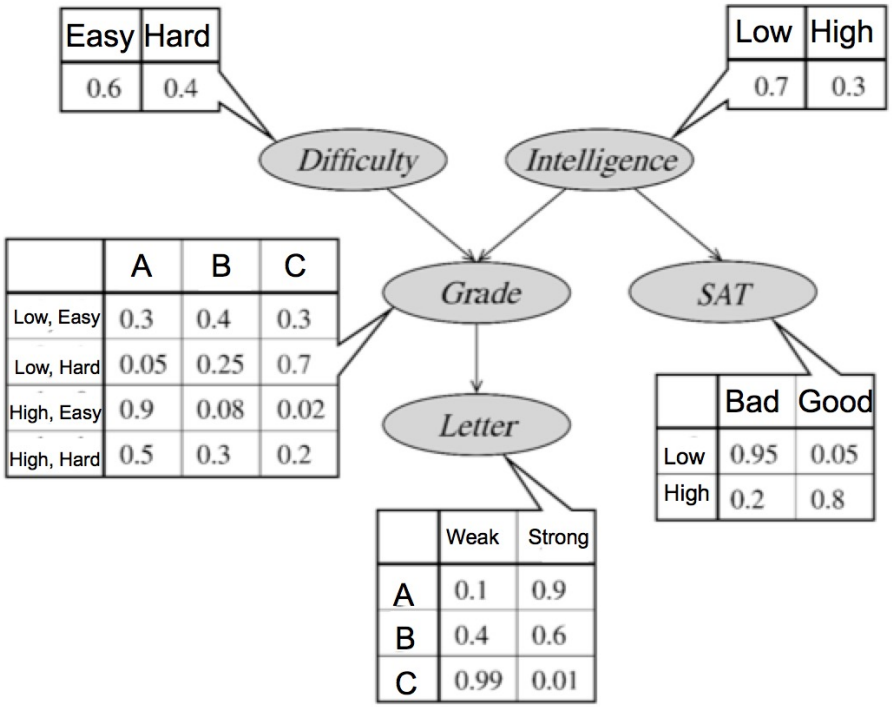
Particle-based techniques

- A *particle* is an assignment of values to (some) variables of a graphical model

- Basic idea: Sets of particles can be used to approximate a distribution
  - E.g. Many samples from a distribution can be a good representation of original distribution

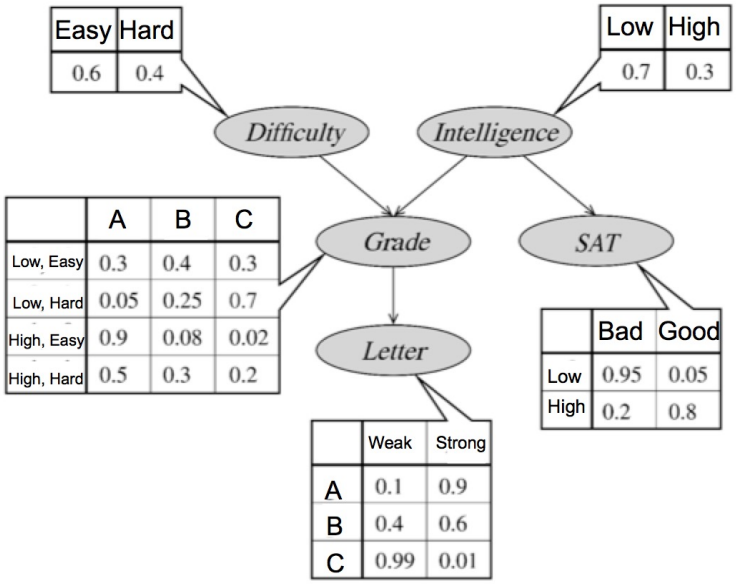
Forward sampling

- We can sample particles using the simple *Forward sampling* algorithm
  - Sample values from priors at root nodes
  - For a node X for which values have been sampled for all parents, sample from P(X | Parents(X))



Some sampled particles

Hard	Low	C	Bad	Strong
Hard	High	A	Good	Strong
Easy	High	A	Good	Strong
Hard	Low	C	Bad	Strong
Easy	Low	C	Bad	Strong
Easy	Low	C	Bad	Strong
Easy	Low	B	Bad	Strong
Hard	High	A	Bad	Strong
Easy	Low	B	Bad	Weak
Easy	Low	A	Bad	Strong



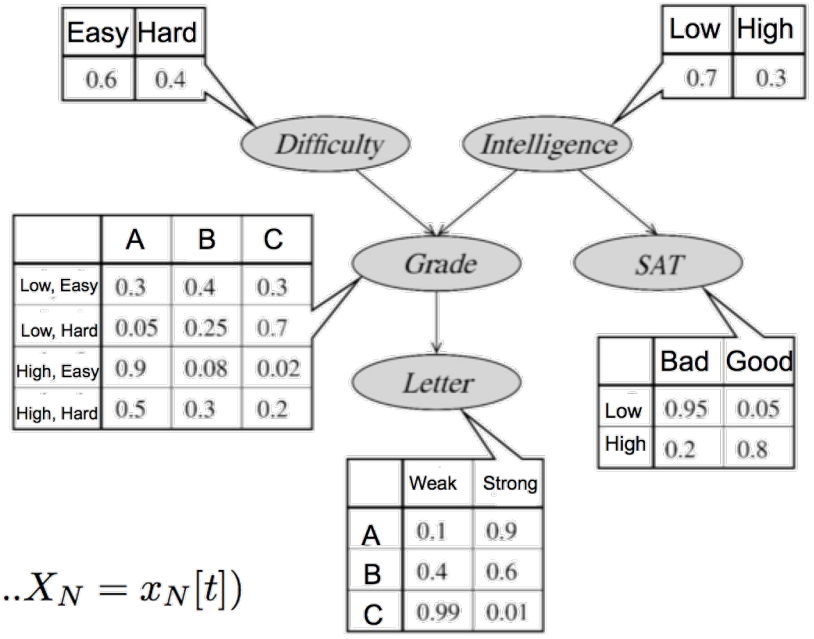
- What is P(G=A)?
- What is P(L=Weak)?
- What is P(S=Good | Grade = A)?
- What is P(L=Strong | Grade = C)?

Markov Chain Monte Carlo (MCMC)

- General class of techniques that produce a *sequence* of samples
- Main idea: Save effort by using information from *past samples* in producing *future samples*
  - Initial samples are from a *proposal (approximate) distribution* Q
  - Subsequent sampling is biased towards P
  - Eventually the samples are drawn from a distribution that is closer and closer to P

Special case of MCMC: Gibbs sampling

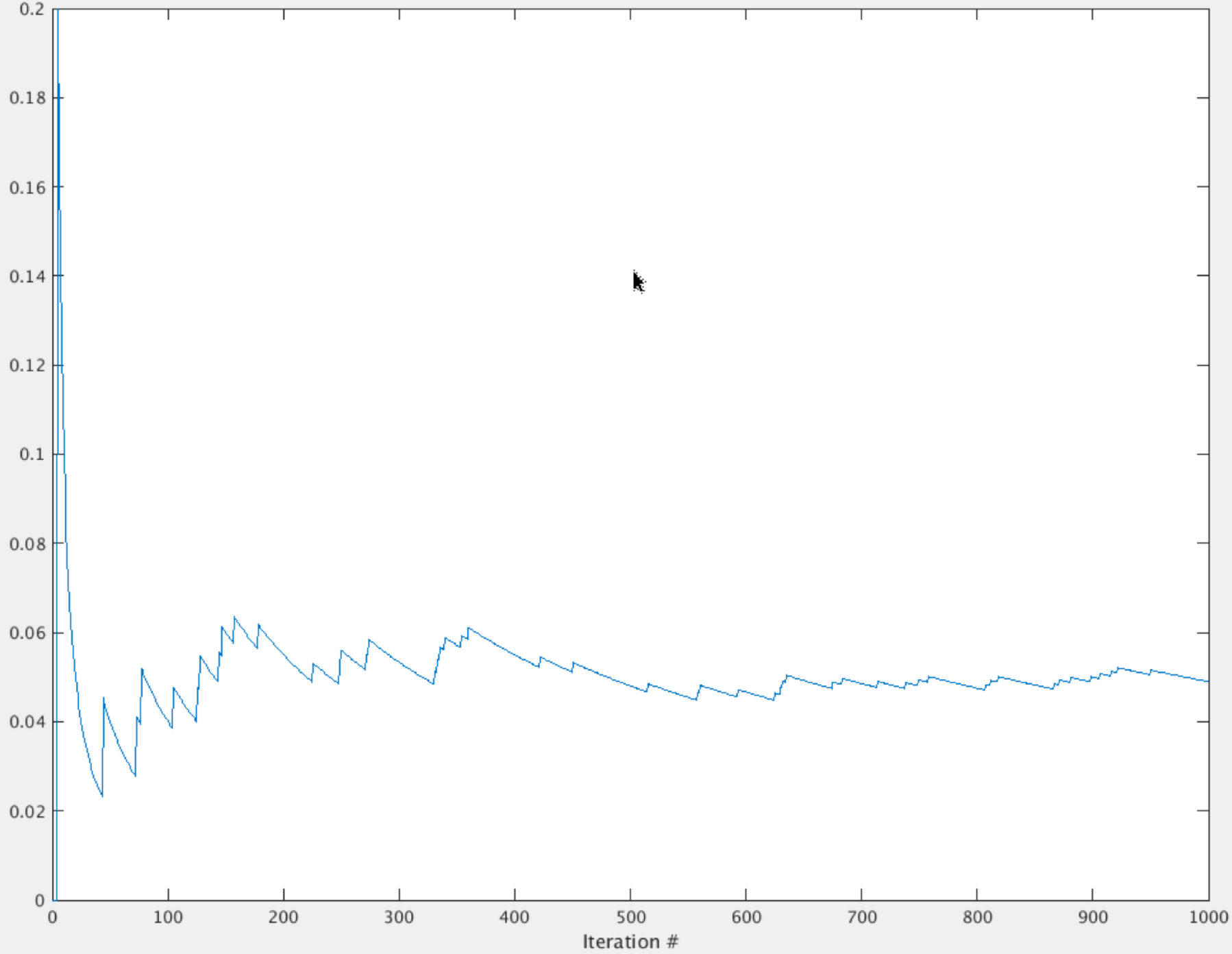
- Generate initial sample *x[0]*
- For each sample *t=1…T*
  - Let *x[t] = x[t-1]*
  - For each unobserved variable X_i,
    - Sample a value for X_i given values for all other variables in *x[t];* i.e. sample from:



$$P(X_i|X_1 = x_1[t], \ldots X_{i-1} = x_{i-1}[t], X_{i+1} = x_{i+1}[t], \ldots X_N = x_N[t])$$
$$= P(X_i|\mathbf{X_{-i}} = \mathbf{x_{-i}}[t])$$

where $\mathbf{X_{-i}} = \mathbf{X} - \{X_i\}$

- Put this sampled value in $x_i[t]$

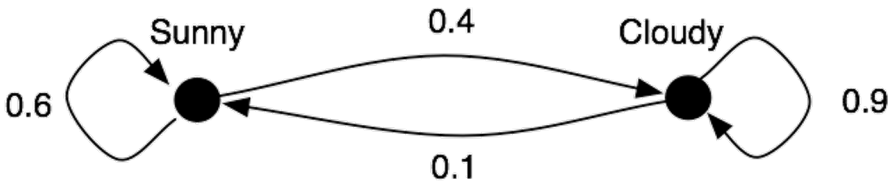

0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
100
200
300
400
500
600
700
800
900
1000
Iteration #

Properties of Gibbs sampling

- Gibbs can be applied to any Bayes networks

- Gibbs sampling will converge to sampling from the correct distribution, ***eventually***
  - But may require a long time to converge
  - Why does this happen?

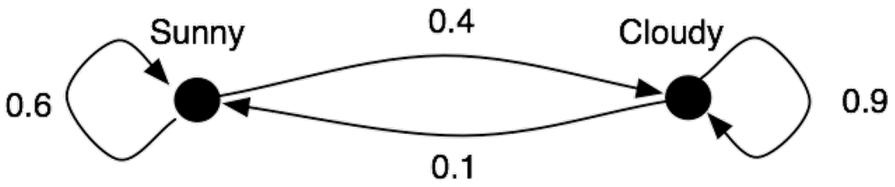
Markov chains



- Suppose there's an 80% chance of sun on day 0. What is the probability of sun on day 3?

$$
\begin{aligned}
P(Q_3 = \text{☀}) &= P(Q_3 = \text{☀} | Q_2 = \text{☀}) P(Q_2 = \text{☀}) + P(Q_3 = \text{☀} | Q_2 = \text{🌧}) P(Q_2 = \text{🌧}) \\
&= 0.6 P(Q_2 = \text{☀}) + 0.1 P(Q_2 = \text{🌧}) \\
&= 0.6(0.6 P(Q_1 = \text{☀}) + 0.1 P(Q_1 = \text{🌧})) + 0.1(0.4 P(Q_1 = \text{☀}) + 0.9 P(Q_1 = \text{🌧})) \\
&= 0.6(0.6(0.6 P(Q_0 = \text{☀}) + 0.1 P(Q_0 = \text{🌧})) + 0.1(0.4 P(Q_0 = \text{☀}) + 0.9 P(Q_0 = \text{🌧}))) \\
&+ 0.1(0.4(0.4 P(Q_0 = \text{☀}) + 0.1 P(Q_0 = \text{🌧})) + 0.9(0.4 P(Q_0 = \text{☀}) + 0.9 P(Q_0 = \text{🌧}))) \\
&= 0.6(0.6(0.6(0.8) + 0.1(0.2)) + 0.1(0.4(0.8) + 0.9(0.2))) \\
&+ 0.1(0.4(0.6(0.8) + 0.1(0.2)) + 0.9(0.4(0.8) + 0.9(0.2))) \\
&= 0.275
\end{aligned}
$$

Markov chains



- Suppose there's an 80% chance of sun on day 0. What is the probability of sun on day 3?

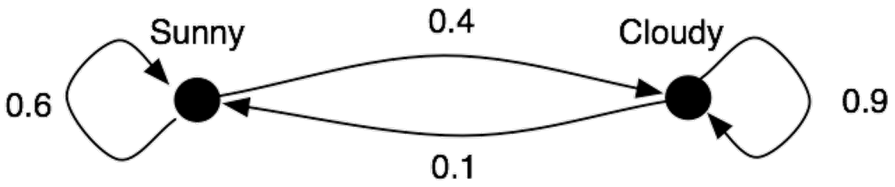
$$B = \begin{bmatrix} 0.6 & 0.4 \\ 0.1 & 0.9 \end{bmatrix} \qquad w = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}$$

$$(B^T)^3 w = \begin{bmatrix} 0.275 \\ 0.725 \end{bmatrix}$$

$P(X_3 = \text{sun})$

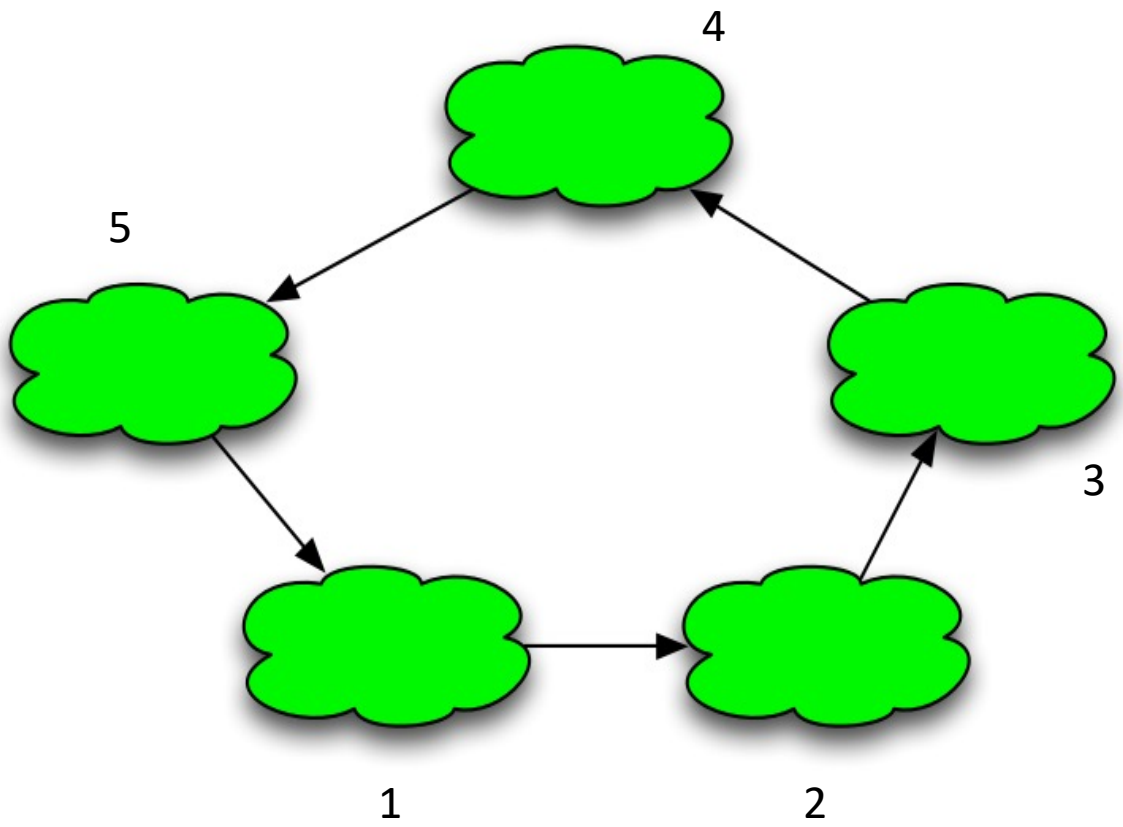
$P(X_3 = \text{cloudy})$

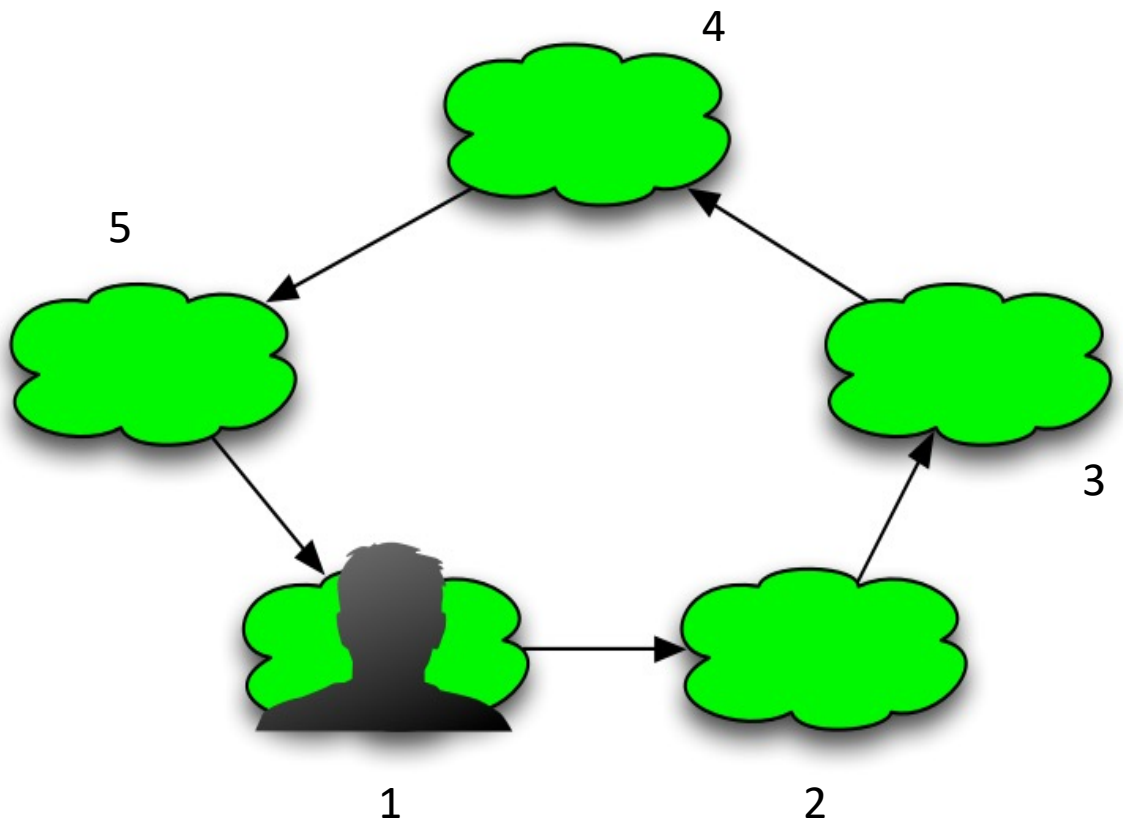
Stationary distributions



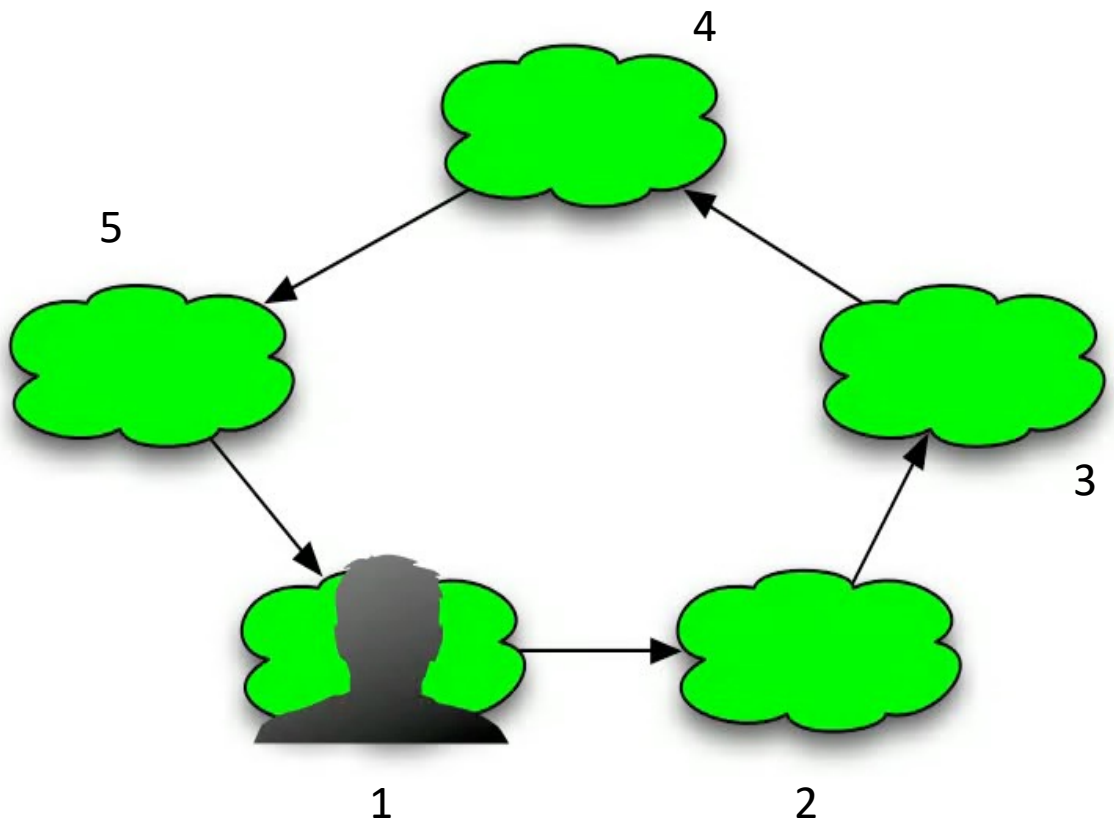
- For an *ergodic* chain, a *stationary distribution* exists
  - ergodic: all states are recurrent and aperiodic
  - stationary distribution: for large *t*, the probability of being in state *i* at time *t* depends *only* on the transition probabilities
  - the stationary distribution π is the vector satisfying

$$B^T \pi = \pi$$


4
5
3
1
2

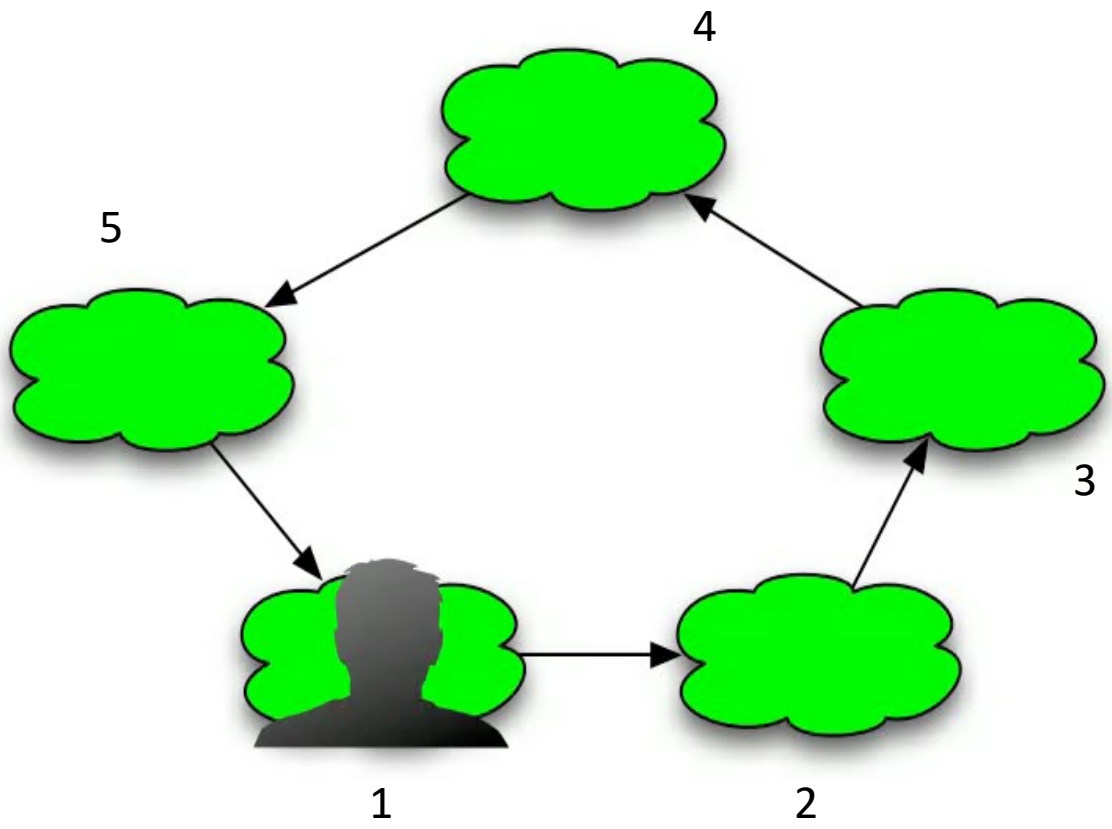

4
5
3
1
2

Q: At any moment in time, what's the probability that the frog is on pad 1?



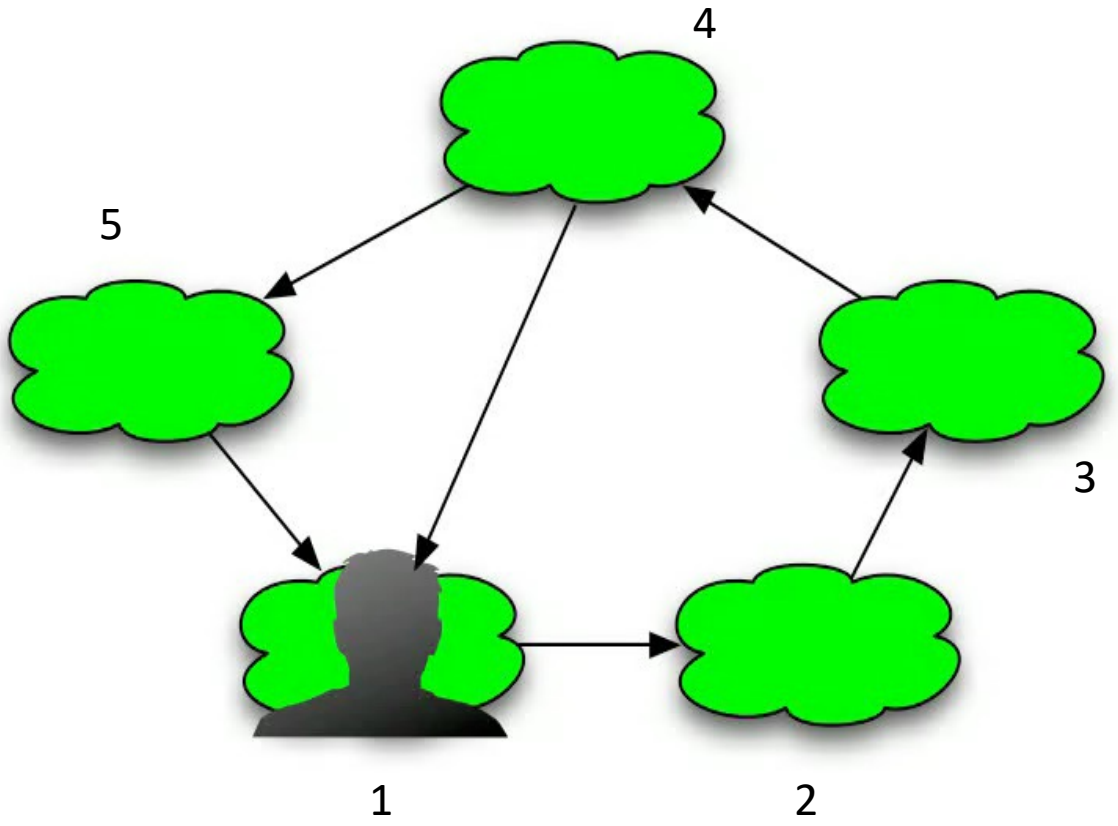
Q: At any moment in time, what's the probability that the frog is on pad 1?

A: P(Pad=1) = 1/5. Same for 2, 3, 4, 5.



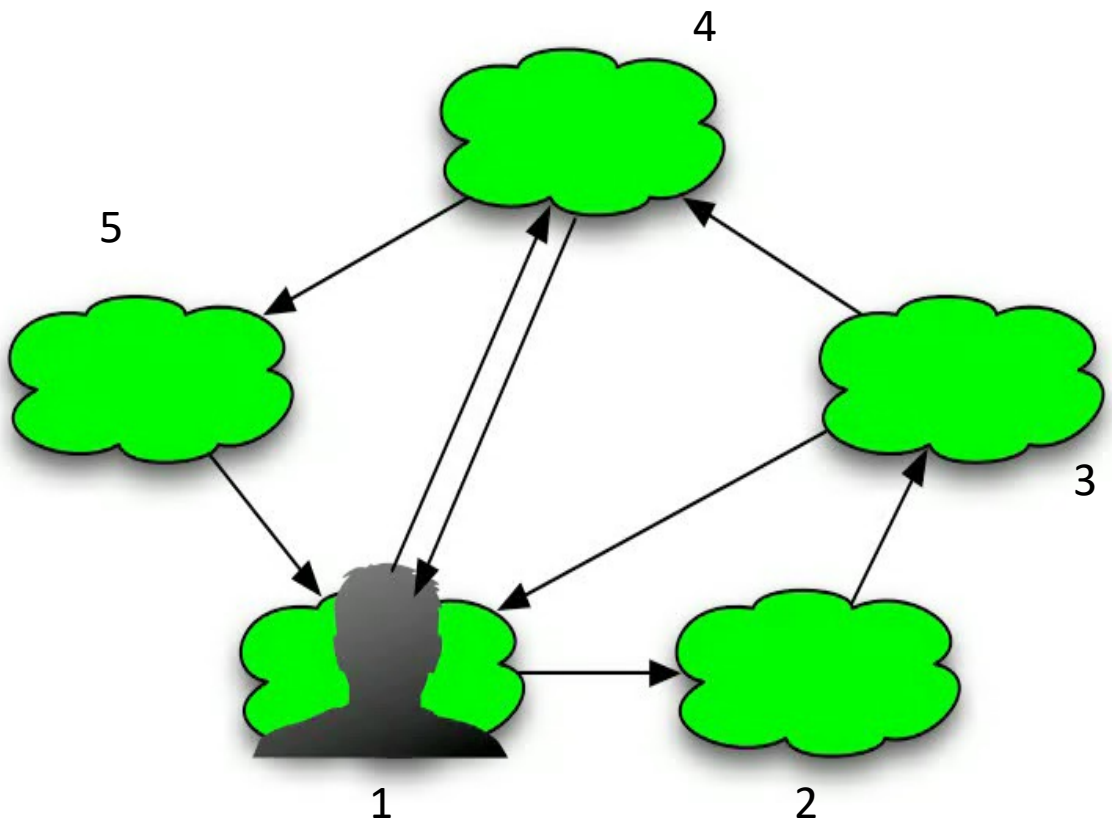
Q: At any moment in time, what's the probability that the frog is on pad 1?

A: P(Pad=1) = ?

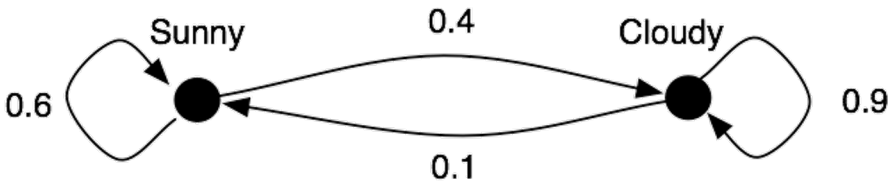


Q: At any moment in time, what's the probability that the frog is on pad 1?

A: P(Pad=1) = ?



Stationary distributions

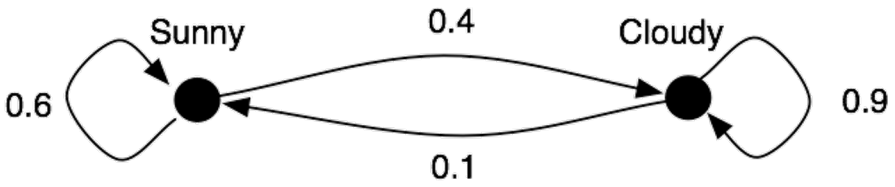


- For an *ergodic* chain, a *stationary distribution* exists
  - ergodic: all states are recurrent and aperiodic
  - stationary distribution: for large *t*, the probability of being in state *i* at time *t* depends *only* on the transition probabilities
  - the stationary distribution π is the vector satisfying

$$B^T \pi = \pi$$

How do we compute π?

Stationary distribution of Markov chain



- What is the stationary distribution of this chain?

```
>> % e.g. in Matlab:

>> [v d]=eigs([0.6 0.4; 0.1 0.9]',1)

v =
  -0.24253562503633
  -0.97014250014533

d =
     1

>> v/sum(v)

ans =

   0.20000000000000
   0.80000000000000
```

$$\pi = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}$$

Monte Carlo Methods

- Monte Carlo techniques involve random sampling for applications like numerical integration and optimization
  - Originally invented for nuclear physics, now widely used across a wide range of domains

- Useful when it's difficult to measure some quantity but it's easy to generate samples from a distribution related to that quantity

Back to Markov Chain Monte Carlo (MCMC)

- Recall we want to estimate some distribution P(X)
  - But inference is too hard to compute it directly
- Basic idea: Construct a Markov Chain whose *stationary distribution* is exactly P(X)
  - Then take random walks on the Markov Chain
  - If we walk long enough, sampling from the Markov Chain is exactly equivalent to sampling from P(X)

Next class

- More MCMC and statistical learning

MCMC

- Each state in the Markov Chain is one possible assignment of values to all variables

- In Gibb's Sampling, we chose the transition probabilities such that:
  - a stationary distribution exists and,
  - the stationary distribution is exactly the posterior probability distribution we want to sample from

Why does Gibbs work?

$$\mathcal{T}_i((\mathbf{x}_{-i}, x_i) \to (\mathbf{x}_{-i}, x_i')) = P(X_i = x_i' | \mathbf{X}_{-i} = \mathbf{x}_{-i})$$

- To prove that Gibbs sampling works and is practical, we need to show that:

1. A stationary distribution for this Markov Chain exists (under some assumptions)
2. The stationary distribution of the Markov Chain is the posterior distribution of the Markov network
3. It's possible to sample from $P(X_i = x_i' | \mathbf{X}_{-i} = \mathbf{x}_{-i})$ efficiently

MCMC in practice

- Might take a long time for samples to be close to the stationary distribution – to "mix"
  - The "burn-in" or "warm-up" time
- The burn-in time depends on the structure of the chain and the transition probabilities
  - When might the burn-in time be particularly long?
- It's possible to compute bounds on the burn-in time, by spectral analysis of the transition matrix
  - I.e. computing eigenvalues and eigenvectors of the Markov Chains' transition matrix
  - But this is completely unhelpful in practice – why?

Practical solutions

- Construct a small number of identical Markov chains
  - Take random walks on each of the chains for a large number of time steps, starting from different initial states
  - Run the chains until the samples seem to be coming from the same distribution across all (or most) of the chains
  - Now use each of the chains to generate (estimated) samples from the posterior distribution