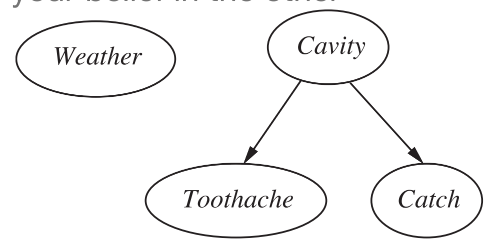
Introduction to Artificial Intelligence
COMP 3501 / COMP 4704-4
Lecture 12: Bayesian Networks

Prof. Nathan Sturtevant

Bayesian Network Semantics

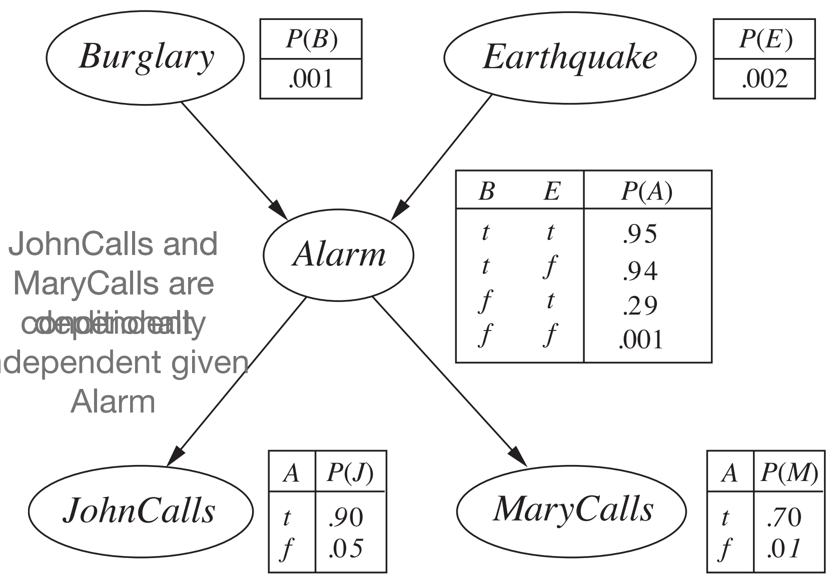
- Bayesian networks are representing independence between variables
  - Two variables are **dependent** if knowing the value of one influences your belief of the value of another
  - Two variables are **independent** if knowing the value of one does not change your belief in the other



Bayesian Network Semantics

- Two variables are **conditionally dependent** if some provided knowledge means that knowing the value of one variable would change your belief in the other
- Two variables are **conditionally independent** if some provided knowledge means that two variables are now independent

Burglary and Earthquake are
conditionally independent dependent given Alarm



P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

JohnCalls and
MaryCalls are
conditionally dependent
independent given
Alarm

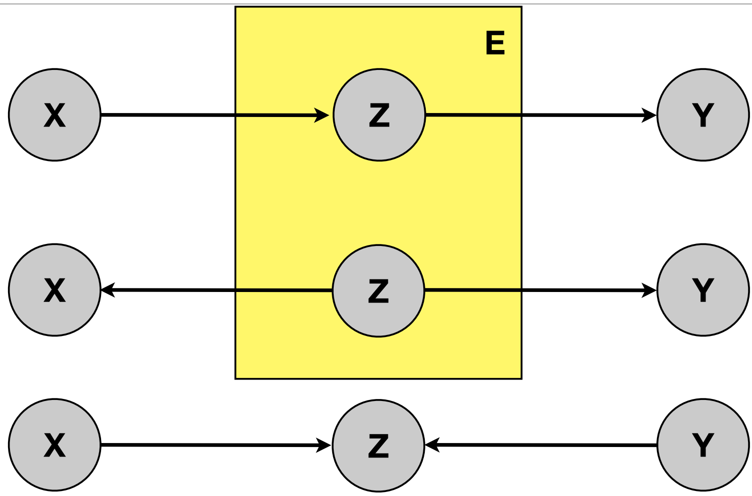
Conditional independence

- Previous versions of the book gave better details on dependence/independence
  - A node is conditionally independent of non-descendants given its parents
  - A node is conditionally independent of all other nodes given its neighbors

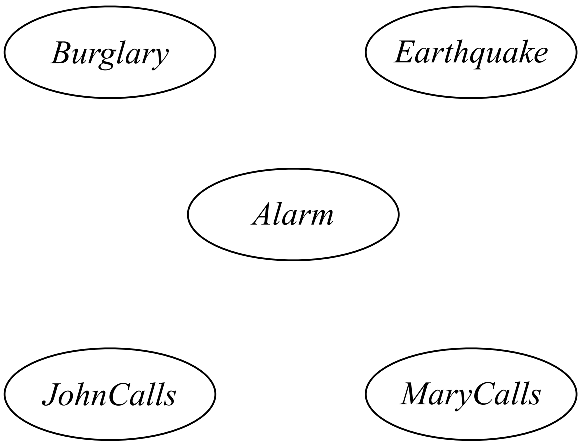
D-separation

- Given a network, how can we easily know if two variables are independent or dependent?
  - direction-dependent separation (d-separation)
- To tell if X and Y are independent:
  - Look at every undirected path between X and Y
  - Check to see if every path is blocked (given E)
- Use the same information to determine whether links need to be added to a network when adding nodes

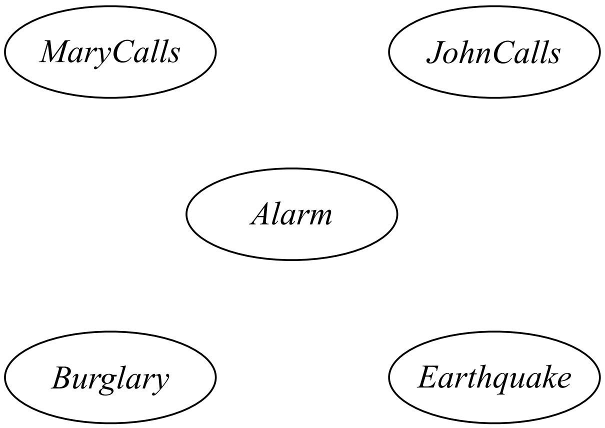
D-separation cases (independence)



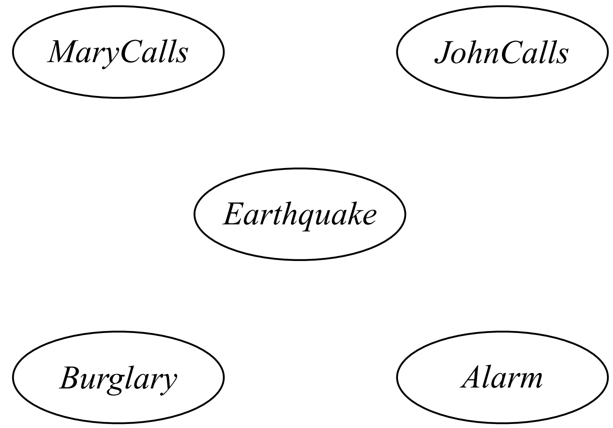
Example 1

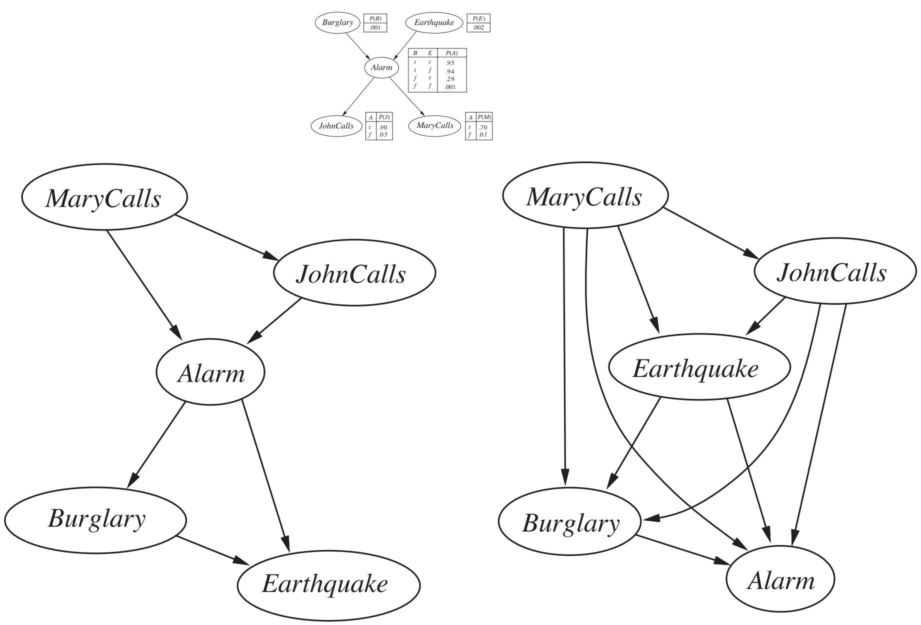


Example 2



Example 3

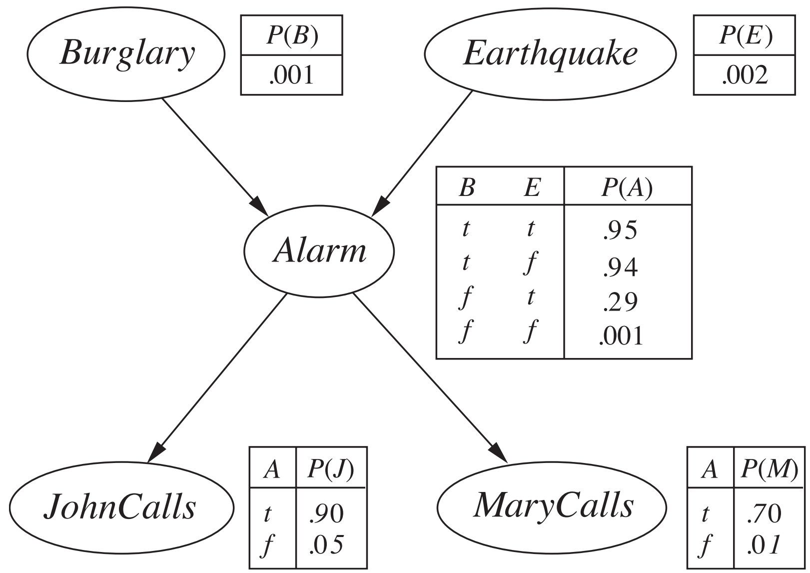




(a) (b)

Exact Inference in Bayesian Networks

- Inferences allows us to ask any question about the variables in the network
- Example: What is P(B=true | j=true ∧ m=true)
  - Hidden variables are alarm & earthquake
  - Enumerate all values for alarm, earthquake, burglary
  - Use ratios to find probabilities



Example

- P(B | j=true ∧ m=true)

b	e	a	j	m	Prob
t	t	t	t	t	1.197E-07
t	t	f	t	t	5.000E-11
t	f	t	t	t	5.910E-04
t	f	f	t	t	2.994E-08
f	t	t	t	t	3.650E-04
f	t	f	t	t	6.094E-07
f	f	t	t	t	6.281E-04
f	f	f	t	t	4.980E-04

Approximate Inference in Bayesian Networks

- Monte-Carlo approach
  - Starting from the top of the tree:
    - Randomly sample unknown variables according to provided distributions
    - Measure if query is true/false
- The ratio of the query being true/false will approach the actual ratio

Direct Sampling

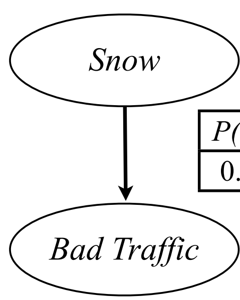
- Consider the following (somewhat silly) example
  - Estimate the distribution of a coin coming up heads
    - Given that it is a fair coin (0.5 heads; 0.5 tails)
- Simulate coin and measure the result

```
nathanst% ./a.out 10
4 of 10 trials are heads (0.400)
nathanst% ./a.out 100
45 of 100 trials are heads (0.450)
nathanst% ./a.out 1000
503 of 1000 trials are heads (0.503)
nathanst% ./a.out 10000
4990 of 10000 trials are heads (0.499)
nathanst% ./a.out 100000
49827 of 100000 trials are heads (0.498)
nathanst% ./a.out 1000000
499324 of 1000000 trials are heads (0.499)
nathanst% ./a.out 10000000
5001293 of 10000000 trials are heads (0.500)
nathanst% ./a.out 100000000
50001419 of 100000000 trials are heads (0.500)
nathanst% ./a.out 1000000000
500024676 of 1000000000 trials are heads (0.500)
```

```
500007582 of 1000000000 trials are heads (0.500)
500024924 of 1000000000 trials are heads (0.500)
500028651 of 1000000000 trials are heads (0.500)
499953457 of 1000000000 trials are heads (0.500)
500031434 of 1000000000 trials are heads (0.500)
500015282 of 1000000000 trials are heads (0.500)
499975931 of 1000000000 trials are heads (0.500)
500003028 of 1000000000 trials are heads (0.500)
500008571 of 1000000000 trials are heads (0.500)
499999580 of 1000000000 trials are heads (0.500)
```

Direct sampling, version 2

- Now consider a simple network
- Simple to estimate probability of bad traffic through sampling & directly
  - $P(S) \cdot P(T \mid S) + P(\neg S) \cdot P(T \mid \neg S)$
  - $0.2 \cdot 0.9 + 0.8 \cdot 0.5 = 0.58$



$P(S)$
0.2

S	$P(T)$
t	0.9
f	0.5


```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
  srandom(time(0));
  int cnt = 10;
  if (argc > 1)
      cnt = atoi(argv[1]);

  int traffic = 0;
  for (int x = 0; x < cnt; x++)
  {
      bool snow = false;
      if ((random()%10) <= 1)
         snow = true;
      if ((snow) && ((random()%10) != 3))
         traffic++;
      if ((!snow) && ((random()%2) != 0))
         traffic++;
  }

  printf("%d of %d trials have bad traffic (%1.3f)\n",
         traffic, cnt, (float)traffic/cnt);
}
```

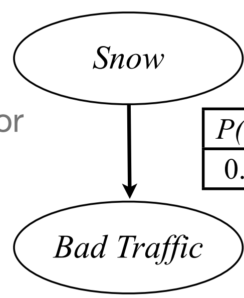
nathanst% ./a.out 10
3 of 10 trials have bad traffic (0.300)
nathanst% ./a.out 100
54 of 100 trials have bad traffic (0.540)
nathanst% ./a.out 1000
592 of 1000 trials have bad traffic (0.592)
nathanst% ./a.out 10000
5818 of 10000 trials have bad traffic (0.582)
nathanst% ./a.out 100000
58309 of 100000 trials have bad traffic (0.583)
nathanst% ./a.out 1000000
579689 of 1000000 trials have bad traffic (0.580)
nathanst% ./a.out 10000000
5800969 of 10000000 trials have bad traffic (0.580)
nathanst% ./a.out 100000000
57994161 of 100000000 trials have bad traffic (0.580)
nathanst% ./a.out 1000000000
579973377 of 1000000000 trials have bad traffic (0.580)

Limitations

- This works so far because we are just sampling probabilities in the network
- What happens if we want to introduce evidence?
  - Depends on where the evidence is introduced

Introducing evidence

- Suppose I want the probability of bad traffic given snow?
  - Like our coin flip, we can compute or sample this directly
  - P(T | S) = 0.9
- In general, this only works if our evidence has no parents

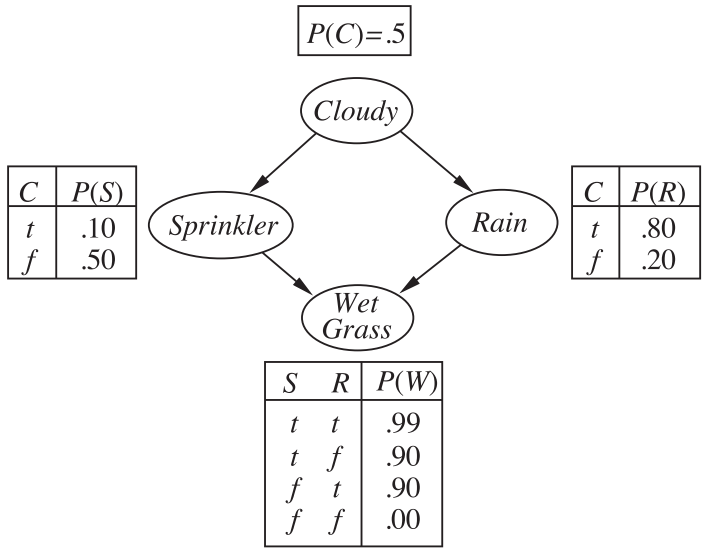


P(S)
0.2

S	P(T)
t	0.9
f	0.5

Rejection Sampling

- To sample a network with evidence:
  - Sample the network as before
  - Throw out cases where evidence isn't true
  - Measure probabilities in resulting network



What is P(Rain|Sprinkler = *true*)?

nathanst% ./a.out 10
3 of 10 trials are valid. 1 of 3 with sprinkler have rain (0.333)
nathanst% ./a.out 100
34 of 100 trials are valid. 11 of 34 with sprinkler have rain (0.324)
nathanst% ./a.out 1000
285 of 1000 trials are valid. 87 of 285 with sprinkler have rain (0.305)
nathanst% ./a.out 10000
3010 of 10000 trials are valid. 881 of 3010 with sprinkler have rain (0.293)
nathanst% ./a.out 100000
29933 of 100000 trials are valid. 8897 of 29933 with sprinkler have rain (0.297)
nathanst% ./a.out 1000000
301476 of 1000000 trials are valid. 90165 of 301476 with sprinkler have rain (0.299)
nathanst% ./a.out 10000000
3006959 of 10000000 trials are valid. 901158 of 3006959 with sprinkler have rain (0.300)
nathanst% ./a.out 100000000
30078397 of 100000000 trials are valid. 9002019 of 30078397 with sprinkler have rain (0.299)
nathanst% ./a.out 1000000000
300782965 of 1000000000 trials are valid. 90022215 of 300782965 with sprinkler have rain (0.299)

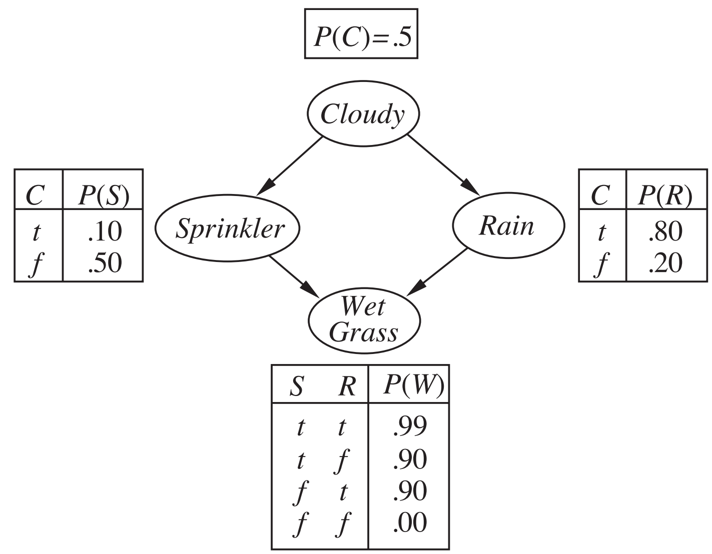
```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
   srandom(time(0));
   int cnt = 10;
   if (argc > 1)
        cnt = atoi(argv[1]);

   int valid = 0;
   int rainCnt = 0;
   for (int x = 0; x < cnt; x++)
   {
        bool cloudy = false;
        if ((random()%2) == 1)
            cloudy = true;
        bool sprinkler = false;
        if ((cloudy) && ((random()%10) == 3))
            sprinkler = true;
        if ((!cloudy) && ((random()%2) == 0))
            sprinkler = true;

        bool rain = false;
        if ((cloudy) && ((random()%10) >= 2))
            rain = true;
        if ((!cloudy) && ((random()%10) < 2))
            rain = true;

        bool wet = false;
        if (cloudy&&rain&& (random()%100) < 99)
            wet = true;
        if (cloudy&&!rain&& (random()%100) < 90)
            wet = true;
        if (!cloudy&&rain&& (random()%100) < 90)
            wet = true;

        if (sprinkler)
        {
            valid++;
            if (rain)
                rainCnt++;
        }
   }

   printf("%d of %d trials are valid. %d of %d with sprinkler have rain (%1.3f)\n", valid, cnt, rainCnt, valid, (float)rainCnt/valid);
}
```

Drawbacks of this approach?

- What if the number of variables we need to sample grows?
- What if the likelihood of the data shrinks?

Likelihood weighting

- How can we measure uncommon events?
  - Fix evidence in bayesian network
  - Sample all other variables
  - Weight sample according to likelihood of sample given the evidence



C	P(S)
t	.10
f	.50

C	P(R)
t	.80
f	.20

S	R	P(W)
t	t	.99
t	f	.90
f	t	.90
f	f	.00

What is P(Rain|Sprinkler = *true*)?

Likelihood weighting

- Sample from the top to bottom of the network
  - Never sample child until parent is sampled
  - eg sample Cloudy, Sprinkler, Rain, Wet Grass
- Cloudy & Rain are sampled according to distributions
  - Sprinkler is fixed at true, but weighted according to P(S=true|C)
  - Wet grass is sampled


```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
   srandom(time(0));
   int cnt = 10;
   if (argc > 1)
        cnt = atoi(argv[1]);

   double rainCnt = 0;
   double rainNotCnt = 0;
   double weight = 1;
   for (int x = 0; x < cnt; x++)
   {
        bool cloudy = false;
        if ((random()%2) == 1)
            cloudy = true;
        bool sprinkler = true;
        if (cloudy) weight = 0.1;
        if (!cloudy) weight = 0.5;

        bool rain = false;
        if ((cloudy) && ((random()%10) >= 2))
            rain = true;
        if ((!cloudy) && ((random()%10) < 2))
            rain = true;
        bool wet = false;
        if (cloudy&&rain&& (random()%100) < 99)
            wet = true;
        if (cloudy&&!rain&& (random()%100) < 90)
            wet = true;
        if (!cloudy&&rain&& (random()%100) < 90)
            wet = true;

        if (rain) rainCnt += weight;
        if (!rain) rainNotCnt += weight;
   }

   printf("%1.3f weight on rain; %1.3f on no rain.\n(%1.3f normalized chance of rain | sprinkler = true)\n",
          rainCnt, rainNotCnt, rainCnt/(rainCnt+rainNotCnt));
}
```

```
nathanst% ./a.out 10
1.000 weight on rain; 1.200 on no rain.
(0.455 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 100
10.000 weight on rain; 18.800 on no rain.
(0.347 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 1000
96.200 weight on rain; 204.600 on no rain.
(0.320 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 10000
894.600 weight on rain; 2127.800 on no rain.
(0.296 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 100000
9043.100 weight on rain; 21028.900 on no rain.
(0.301 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 1000000
90097.000 weight on rain; 209879.000 on no rain.
(0.300 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 10000000
900268.800 weight on rain; 2100807.600 on no rain.
(0.300 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 100000000
9001197.493 weight on rain; 20999451.701 on no rain.
(0.300 normalized chance of rain | sprinkler = true)
nathanst% ./a.out 1000000000
90001623.707 weight on rain; 210021675.228 on no rain.
(0.300 normalized chance of rain | sprinkler = true)
```

Likelihood weighting

- Likelihood weighting uses information from every sample generated
- But:
  - If there is a lot of evidence, the weights will be small (and thus less accurate)
  - If the evidence occurs late in the inference, it may be inconsistent with the rest of the network
    - eg allow Sprinkler & Rain to be false with evidence that the grass is wet [resulting in 0 weight]

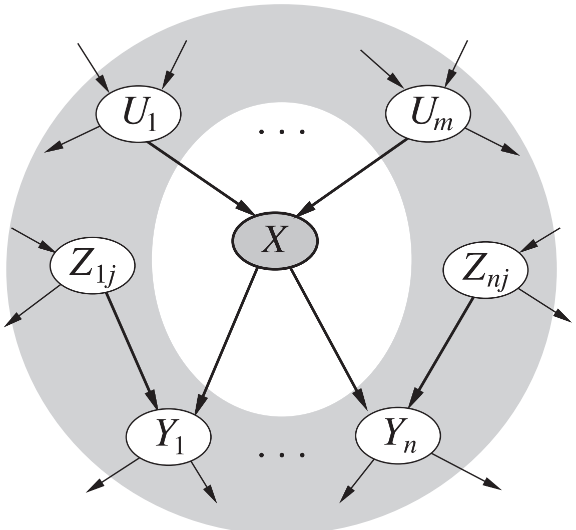
Local Search for inference

- Markov-Chain Monte-Carlo sampling
  - Build a complete state of the world
  - Re-sample each non-evidence variable according to probability distribution
  - Count how often query is true

Probability distributions

- How do we compute the chance of a given variable given everything else in the network?
- Markov Blanket: A variable is independent of the rest of the networking given its parents, children, and children's parents

Markov Blanket



Sampling a variable

- Sample each variable from:

$$P(x_i'|mb(X_i)) = \alpha P(x_i'|parents(X_i)) \prod_{Y_j \in Children(X_i)} P(y_j|parents(Y_j))$$

- Why is this correct?
- How would this work for our network?