# Probabilistic Reasoning

Unit # 14



## Structure Learning

- Structure Learning consists of learning the DAG in a Bayesian network from data.
- We need to learn a DAG that satisfies the Markov condition with the probability distribution P that is generating the data.
- *We do not know P, all we know are the data.*

# Score-based Structure Learning

- In score-based structure learning, we assign a score to a DAG based on how well the DAG fits the data.
- Two popular scores are:
  - Bayesian Score
  - Bayesian Information Criterion Score



# Probability of Data

- Suppose that we are about to repeatedly toss a thumbtack (or perform any repeatable experiment with two outcomes).
- Suppose further that we assume exchangeability, and we represent our prior belief concerning the probability of heads using Dirichlet distribution with parameters a and b, where a and b are positive integers and m = a + b.
- Let D be data that consists of s heads and t tails in n trials.
- Then

$$P(\mathrm{D}) = \frac{(m-1)!}{(m+n-1)!} \times \frac{(a+s-1)!(b+t-1)!}{(a-1)!(b-1)!}.$$

# Example 8.2 (Source: Neapolitan)

- Suppose that, before tossing a thumbstack, we assign a=3 and b=5 to model the slight belief that tails is more probable than heads.
- We then toss the thumbstack ten times and obtain four heads and six tails.
- The probability of obtaining these data D is given by

$$P(D) = \frac{(8-1)!}{(8+10-1)!} \times \frac{(3+4-1)!\,(5+6-1)!}{(3-1)!\,(5-1)!}$$



# Example 8.2 (Source: Neapolitan)

- Sometimes the following equation

$$P(\mathrm{D}) = \frac{(m-1)!}{(m+n-1)!} \times \frac{(a+s-1)!(b+t-1)!}{(a-1)!(b-1)!}.$$

- is written as

$$P(\mathrm{D}) = \frac{\Gamma(m)}{\Gamma(m+n)} \times \frac{\Gamma(a+s)\Gamma(b+t)}{\Gamma(a)\Gamma(b)}.$$

- $\Gamma$ denotes the gamma function. When n is an integer $\geq 1$, we have

$$\Gamma(n) = (n-1)!$$

# Learning DAG Models using Bayesian Score

- We can score a DAG model G based on data D by determining how probable the data are given the DAG model. That is, we compute (D|G).
- The formula for this probability is the same as discussed in the previous slides, except there is a term for each probability in the network.



# Learning DAG Models using Bayesian Score (Cont'd)


$$P(\mathsf{D}|\mathbb{G}_1) = \frac{\Gamma(m_{11})}{\Gamma(m_{11}+n_{11})} \times \frac{\Gamma(a_{11}+s_{11})\Gamma(b_{11}+t_{11})}{\Gamma(a_{11})\Gamma(b_{11})} \times \frac{\Gamma(m_{21})}{\Gamma(m_{21}+n_{21})} \times \frac{\Gamma(a_{21}+s_{21})\Gamma(b_{21}+t_{21})}{\Gamma(a_{21})\Gamma(b_{21})} \times \frac{\Gamma(m_{22})}{\Gamma(m_{22}+n_{22})} \times \frac{\Gamma(a_{22}+s_{22})\Gamma(b_{22}+t_{22})}{\Gamma(a_{22})\Gamma(b_{22})}.$$

$$P(\mathsf{D}|\mathbb{G}_2) = \frac{\Gamma(m_{11})}{\Gamma(m_{11}+n_{11})} \times \frac{\Gamma(a_{11}+s_{11})\Gamma(b_{11}+t_{11})}{\Gamma(a_{11})\Gamma(b_{11})} \times \frac{\Gamma(m_{21})}{\Gamma(m_{21}+n_{21})} \times \frac{\Gamma(a_{21}+s_{21})\Gamma(b_{21}+t_{21})}{\Gamma(a_{21})\Gamma(b_{21})}.$$

# Examples

- Compute P($G_1$ |D) and P($G_2$ |D) for the following cases. Assume P($G_1$) = P($G_2$) = 0.5

| Case | $J$ | $F$ |
|---|---|---|
| 1 | $j_1$ | $f_1$ |
| 2 | $j_1$ | $f_1$ |
| 3 | $j_1$ | $f_1$ |
| 4 | $j_1$ | $f_1$ |
| 5 | $j_1$ | $f_2$ |
| 6 | $j_2$ | $f_1$ |
| 7 | $j_2$ | $f_2$ |
| 8 | $j_2$ | $f_2$ |

| Case | $J$ | $F$ |
|---|---|---|
| 1 | $j_1$ | $f_1$ |
| 2 | $j_1$ | $f_1$ |
| 3 | $j_1$ | $f_1$ |
| 4 | $j_1$ | $f_1$ |
| 5 | $j_2$ | $f_2$ |
| 6 | $j_2$ | $f_2$ |
| 7 | $j_2$ | $f_2$ |
| 8 | $j_2$ | $f_2$ |

| Case | $J$ | $F$ |
|---|---|---|
| 1 | $j_1$ | $f_1$ |
| 2 | $j_1$ | $f_1$ |
| 3 | $j_1$ | $f_2$ |
| 4 | $j_1$ | $f_2$ |
| 5 | $j_2$ | $f_1$ |
| 6 | $j_2$ | $f_1$ |
| 7 | $j_2$ | $f_2$ |
| 8 | $j_2$ | $f_2$ |

- (0.517, 0.483) (0.959, 0.041) (0.331,0.661)



# Learning DAG Patterns

- The DAG F → J is Markov equivalent to the DAG J → F.
- As long as we use a prior equivalent sample size, they will have the same scores.
- In general, we cannot distinguish Markov equivalent DAGs based on data.
- So, we are actually learning Markov equivalence classes (DAG patterns) when we learn a DAG model from data.

# Scoring Larger DAG Models

- Suppose we have a DAG G = (V, E) where V is a set of binomial random variables, we assume exchangeability, and we use a Dirichlet distribution to represent our prior belief for each conditional probability distribution of every variable in V.
- Suppose further we have data D consisting of a set of data items such that each data item is a vector of values of all the variables in V. Then

$$P(\mathrm{D}|\mathbb{G}) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N_{ij})}{\Gamma(N_{ij}+M_{ij})} \frac{\Gamma(a_{ij}+s_{ij})\Gamma(b_{ij}+t_{ij})}{\Gamma(a_{ij})\Gamma(b_{ij})}$$



# Scoring Larger DAG Models (Cont'd)

- n is the number of variables
- $q_i$ is the number of different instantiations of the parents of $X_i$.
- $a_{ij}$ is our ascertained prior belief concerning the number of *times* $X_i$ *took*
- *its first value when the parents of* $X_i$ *had their* $j_{th}$ *instantiation.*
- $b_{ij}$ is our ascertained prior belief concerning the number of *times* $X_i$ *took*
- *its first value when the parents of* $X_i$ *had their* $j_{th}$ *instantiation.*
- $s_{ij}$ is *the number of times in the data that* $X_i$ *took its first value when the*
- *parents of* $X_i$ *had their* $j_{th}$ *instantiation.*
- $t_{ij}$ is *the number of times in the data that* $X_i$ *took its first value when the*
- *parents of* $X_i$ *had their* $j_{th}$ *instantiation.*
- $N_{ij}$ *and* $M_{ij}$ *are as follows:*
  - $N_{ij} = a_{ij} + b_{ij}$
  - $M_{ij} = s_{ij} + t_{ij}$

# Number of Possible DAGs

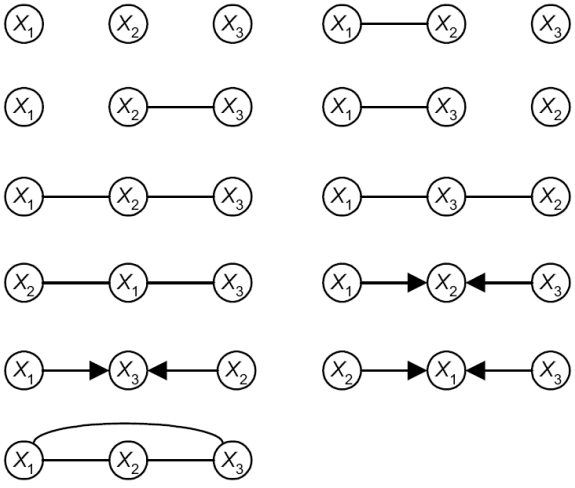
- When there are not many variables, we can exhaustively score all possible DAGs. We then select the DAG(s) with the highest score.
- However, when the number of variables is not small, it is computationally unfeasible to find the maximizing DAGs by exhaustively considering all DAG patterns.
- Number of DAGs containing n nodes is:

$$
\begin{aligned}
f(n) &= \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i) \qquad n > 2 \\
f(0) &= 1 \\
f(1) &= 1
\end{aligned}
$$

- f(2) = 3, f(3) = 25, f(5) = 29,000 and f(10) = $4.2 \times 10^{18}$



# DAG Pattern with 3 Variables



# Consistent Scoring Criterion

- A consistent scoring criterion for DAG models has the following two properties:
  - As the size of the data set approaches infinity, the probability approaches one that a DAG that includes P will score higher than a DAG that does not include P.
  - As the size of the data set approaches infinity, the probability approaches one that a smaller DAG that includes P will score higher than a larger DAG that includes P.



# K2 Algorithm

**Problem**: Find a DAG that approximates maximizing $score(\mathbb{G} : \mathsf{D})$.

**Inputs**: A set V of $n$ random variables; an upper bound $u$ on the number of parents a node may have; data D.

**Outputs**: $n$ sets of parent nodes $\mathsf{PA}_i$, where $1 \leq i \leq n$, in a DAG that approximates maximizing $score(\mathbb{G} : \mathsf{D})$.

```
void K2  (set_of_variables V, int u,
          data D, for 1 ≤ i ≤ n parent_set& PA_i)
{
  for (i = 1; i <= n; i++) {      // n is the number of variables.
    PA_i^G = ∅;
    P_old = score(X_i, PA_i^G : D);
    findmore = true;
```

# K2 Algorithm (Cont'd)

```
        while (findmore && |PA_i^G| < u) {
          Z = node in Pred(X_i) − PA_i that maximizes
                score(X_i, PA_i^G ∪ {Z} : D);
          P_new = score(X_i, PA_i^G ∪ {Z} : D) ;
          if (P_new > P_old) {
            P_old = P_new;
            PA_i^G = PA_i^G ∪ {Z};
          }
          else
            findmore = false;
        }
      }
    }
```