Lecture 6

Cause and Independence

Joint Probability Distributions

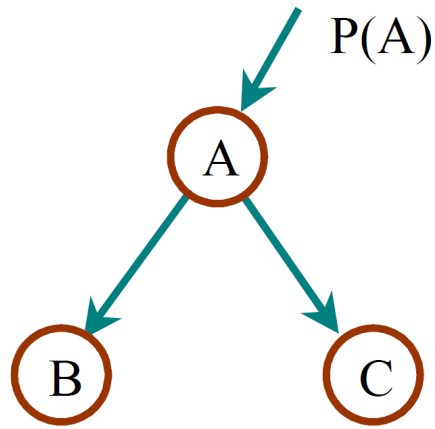
A data set has a joint probability distribution:

$$P(ai, bj, ck) = (\text{No } ([a_i, b_j, c_k]))/N$$

a_3	b_2	c_1
a_2	b_3	c_5
a_4	b_3	c_1
a_1	b_1	c_2
a_2	b_2	c_1
	etc	

And so does a network:

$$P(A\&B\&C) = P(A) \times P(B|A) \times P(C|A)$$



Network joint Probability Distribution

- Last lecture we introduced the Maximally Weighted Spanning tree algorithm which tries to find a network with a joint probability distribution as close as possible to the distribution of the data set.
- Calculating the joint probability of a data point from the network is very much faster than calculating it from the data set.
- Inference in networks is achieved by probability propagation. Theoretically it could also be calculated directly from the data distribution but for non trivial problems this is computationally infeasible.

Multi-Trees (Heckerman)

- A simple but rudementary method of using Bayesian networks for inference is to make use of the joint probability of the variables.
- A low joint probability implies that the data point under test does not fit the model well and *vice versa*.
- This property can be used to reduce the size of an classifier by one variable - along with other things which we will discuss in due course.

Multi-tree example

Given a data set with the root identified as D:

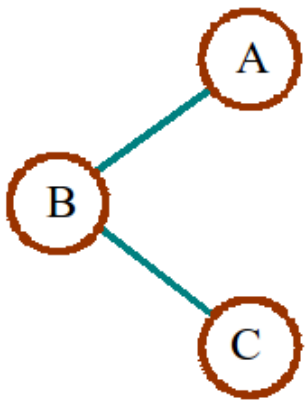
$$[a_1, b_1, c_1, d_1][a_2, b_1, c_1, d_1][a_2, b_1, c_2, d_1][a_2, b_2, c_1, d_1]$$
$$[a_1, b_2, c_2, d_2][a_2, b_1, c_1, d_2][a_2, b_2, c_2, d_2][a_1, b_1, c_1, d_2]$$
$$[a_1, b_1, c_1, d_3][a_2, b_2, c_2, d_3][a_2, b_2, c_1, d_3][a_2, b_2, c_2, d_3]$$

Split the data into different sets corresponding to the states of D:

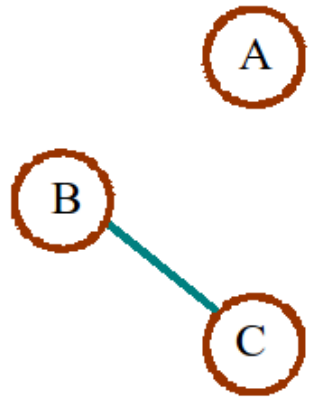
- $D = d_1$: $[a_1, b_1, c_1][a_2, b_1, c_1][a_2, b_1, c_2][a_2, b_2, c_1]$
- $D = d_2$: $[a_1, b_2, c_2][a_2, b_1, c_1][a_2, b_2, c_2][a_1, b_1, c_1]$
- $D = d_3$: $[a_1, b_1, c_1][a_2, b_2, c_2][a_2, b_2, c_1][a_2, b_2, c_2]$

Multi-tree example

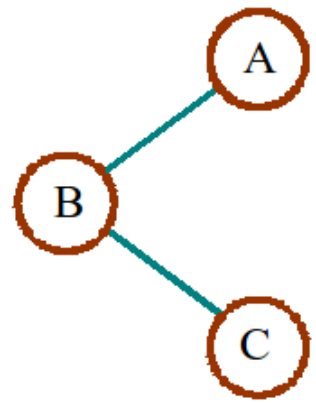
Now build a network for each data set. These "conditional" networks will have one less variable than the original network.



Tree found for D=d1



Tree found for D=d2



Tree found for D=d3

Causal directions may need to be determined for these networks.

Multi-tree example

- For a given data point: (a_i, b_j, c_k) we calculate the joint probability using each tree found:
  - Evidence for d_1 is $P_{D=d_1}(a_i \& b_j \& c_k)$
  - Evidence for d_2 is $P_{D=d_2}(a_i \& b_j \& c_k)$
  - Evidence for d_3 is $P_{D=d_3}(a_i \& b_j \& c_k)$
- The evidence can be normalised to form a distribution over the states of D.

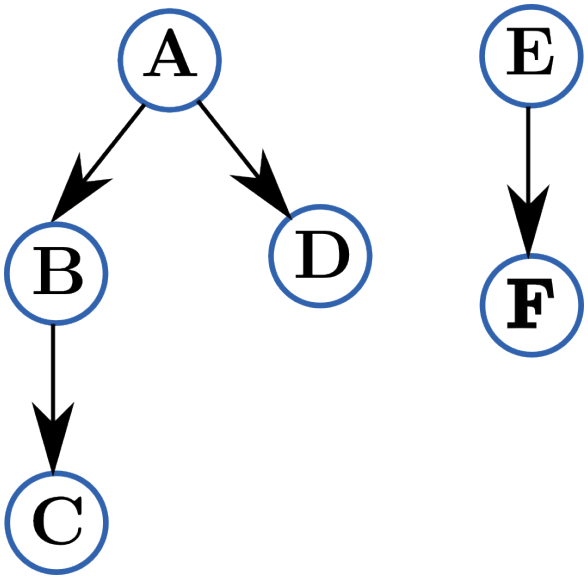
Note that there is a different spanning tree for each state of the class variable.

Independence

Independence is a clearly defined notion with well defined measures.

Given two variables X and Y, if changing X does not change Y and *vice versa* then X and Y are independent.

In a network if there is no path between two variables they are independent.



Nodes E and F are independent of nodes A, B, C and D.

Arcs and Independence

It is possible for two variables to be connected by a path in a network and still be independent. This is because a conditional probability matrix can express no dependency. Consider

$$P(B|A) = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$

For any λ evidence on B say $[b_1, b_2]$ we have that:

$$\lambda_B(A) = [b_1, b_2] \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} = [(b_1 + b_2)/2, (b_1 + b_2)/2]$$

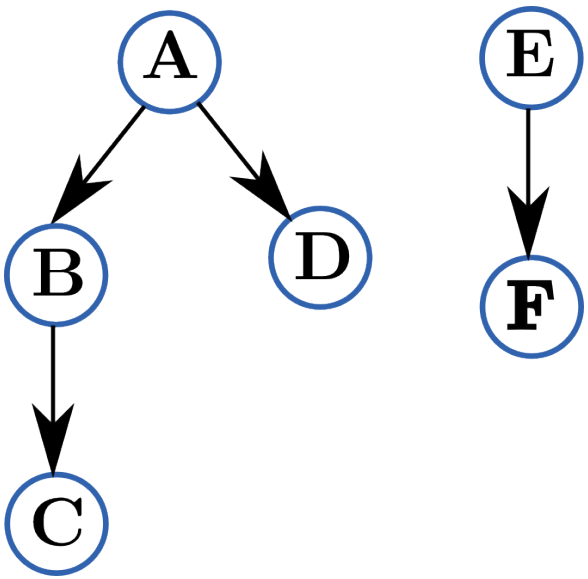
In other words the λ message contains no evidence and therefore A does not change when B changes.

Arcs and Independence

- So in theory, the absence of an arc in a network is more significant than the presence of an arc.
- However in practice we should avoid having arcs expressing very low dependency in networks.
- The spanning tree algorithm can help in this respect. Instead of continuing to include arcs which until we create a connected graph we can terminate the process when the dependency becomes too low to be significant.

Dependency Separation (d-separation)

Any two variables X and Y which are not directly connected by an arc are conditionally independent if there is a set of nodes Z such that knowledge of Z makes X and Y independent. The set Z is said to d-separate X and Y.



Both nodes A and B d-separate C and D.

Causal Directions

- Bayesian networks have a causal direction associated with the arcs. The arrow points from cause to effect.
- The notion of cause in a Bayesian network comes from the idea of conditional probability:

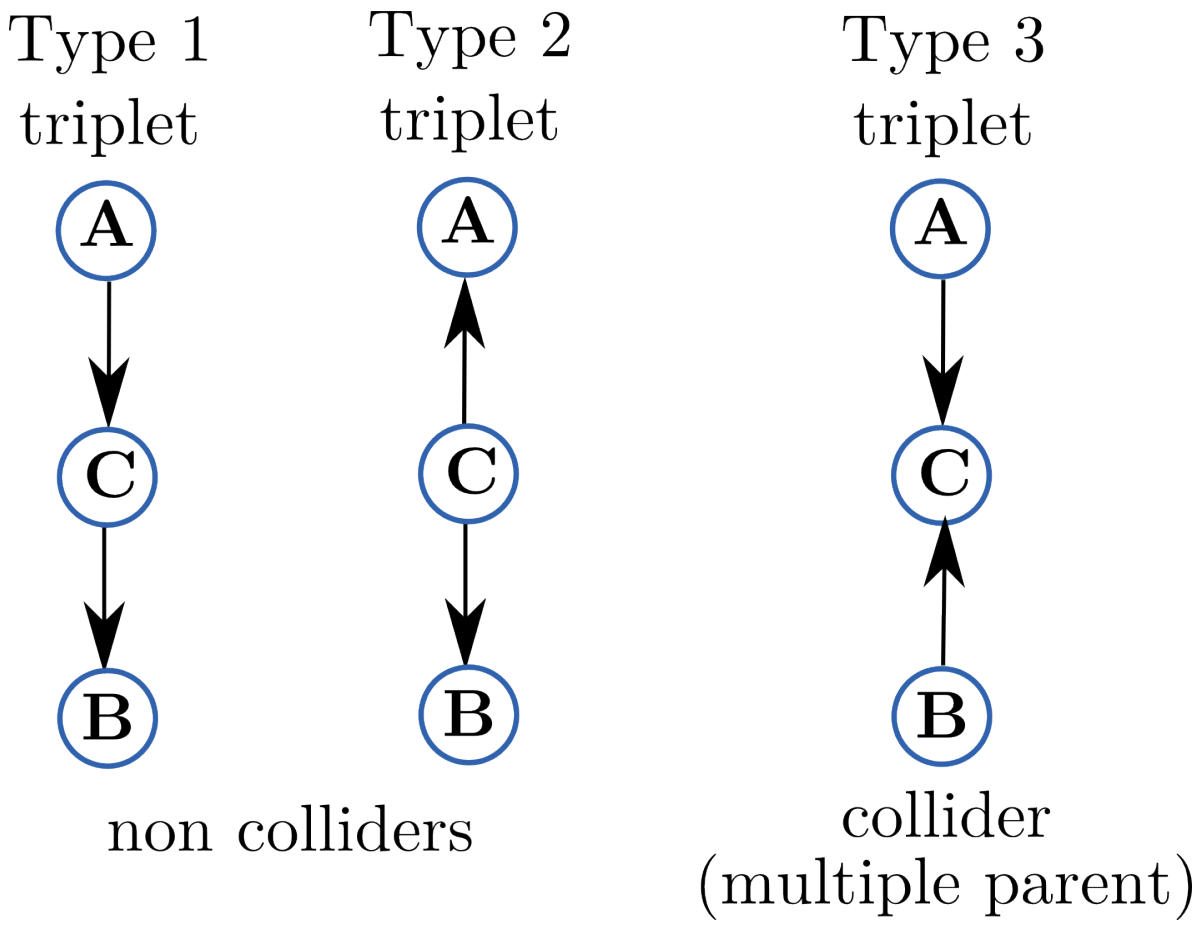
$$P(A\&B) = P(A)P(B|A)$$

- If $P(B|A) = P(B)$ then A and B are independent, if not we think of A causing (or influencing) B, since, given A, B's probability distribution has changed
- However there is nothing in the underlying mathematical theory that helps us to determine causal directions. Instead we usually turn to the semantics of the application to do this.

Causal Directions

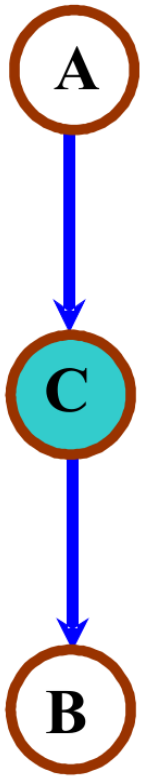
- So far we have taken our causal directions from knowledge about the root variables in our network.
- Knowledge of this kind can come from:
  - Our understanding of the semantics (expert advice)
  - Temporal sequences of events (Granger Causality)
- In certain circumstances, it is also possible to determine cause by examining variable independence.

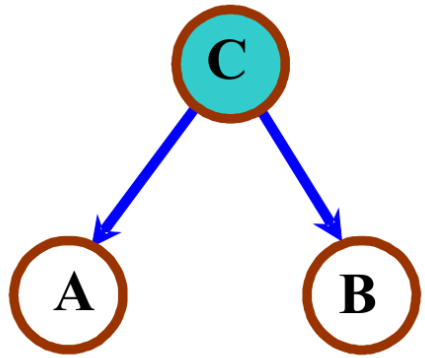
Possible configurations of connected triplets



Non-colliders

For the non colliders A and B are conditionally independent given C.

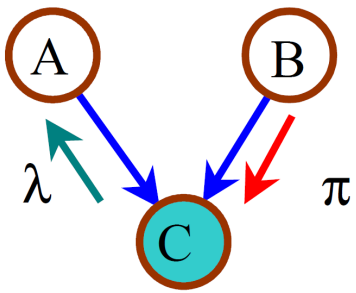




If C is instantiated no messages pass from A to B.

Colliders

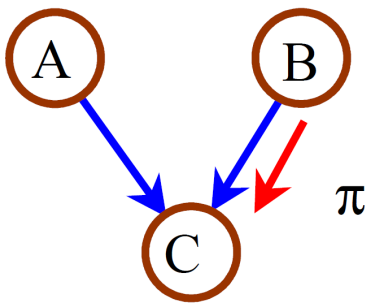
However for a collider (multiple parent), the nodes A and B are only independent if there is no information on C.



Operating Equation 1:

$$\lambda_C(a_i) = \sum_{j=1}^{m} \pi_C(b_j) \sum_{k=1}^{n} P(c_k|a_i\&b_j)\lambda(c_k)$$

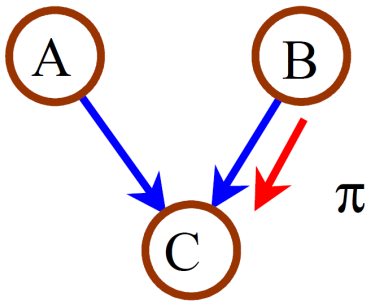
Converging paths are blocked when there is no λ evidence.



$$\lambda(C) = [1, 1, 1 \cdots 1]$$

Marginal Independence

Converging paths are blocked when there is no λ evidence.

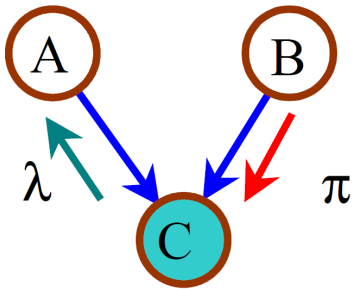


$$\lambda(C) = [1, 1, 1 \cdots 1]$$

Given a data set for the triplet $A - C - B$ we can measure the dependence between A and B using all the data. (ie with no information on C).

If this is low we may suspect that the configuration is a multiple parent.

Conditional Independence



C is instantiated:

$$\lambda(C) = [0, 0, 1 \cdots 0]$$

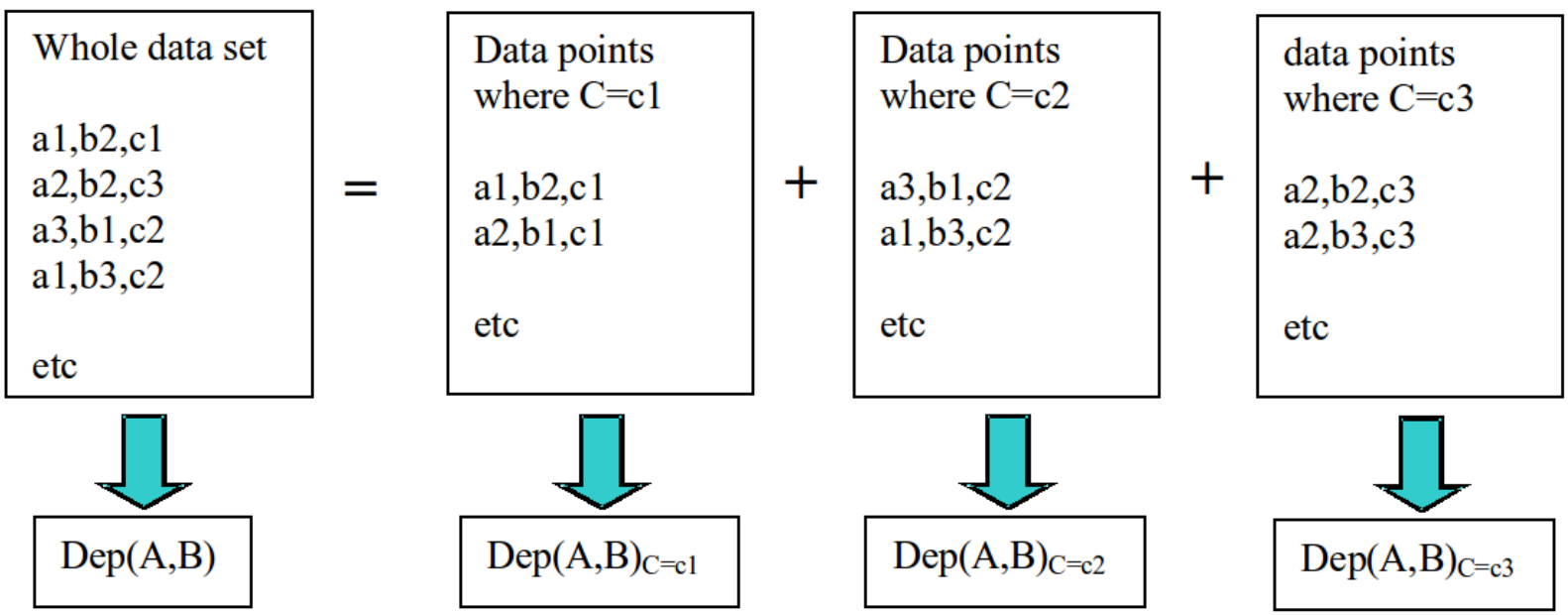
Alternatively we can partition our data according to the states of C, and then compute a set of dependency values (one for each state of C).

If any of these is high we may again suspect that the configuration is a multiple parent.

Practical Computation

Given a triplet, partition the data according to the states of the middle node C, and calculate the dependency of the other variables A, B for each set.

if $Dep(A,B)$ small, and some $Dep(A,B)_{C=c_j}$ is large then the configuration is likely to be a multiple parent.



Computation from the joint probability matrix

Rather than compute the dependencies from the data we can marginalise the joint probability of the triplet:

$$P(A\&B) = \sum_C P(A\&B\&C)$$

For any joint state $[a_i, b_j]$ we sum all the matrix entries $[a_i, b_j, c_k]$.

We use $P(A\&B)$ to calculate the $Dep(A, B)$ which is the dependence of A and B with no information about C and is called the marginal independence.

Computation from the joint probability matrix

Similarly we can use the joint probability matrix to calculate the conditional probabilities:

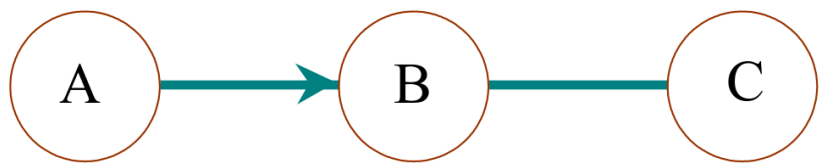
$$P(A\&B|C) = P(A\&B\&C)/P(C)$$

If we think of $P(A\&B\&C)$ as having a column for each state of C and a row for each joint state of A and B, then we normalise each column into a probability distribution.

We then compute a separate value of $Dep(A, B)$ for each column. Adding these together gives us a measure of conditional dependence. If this is close to zero the configuration is not a collider.

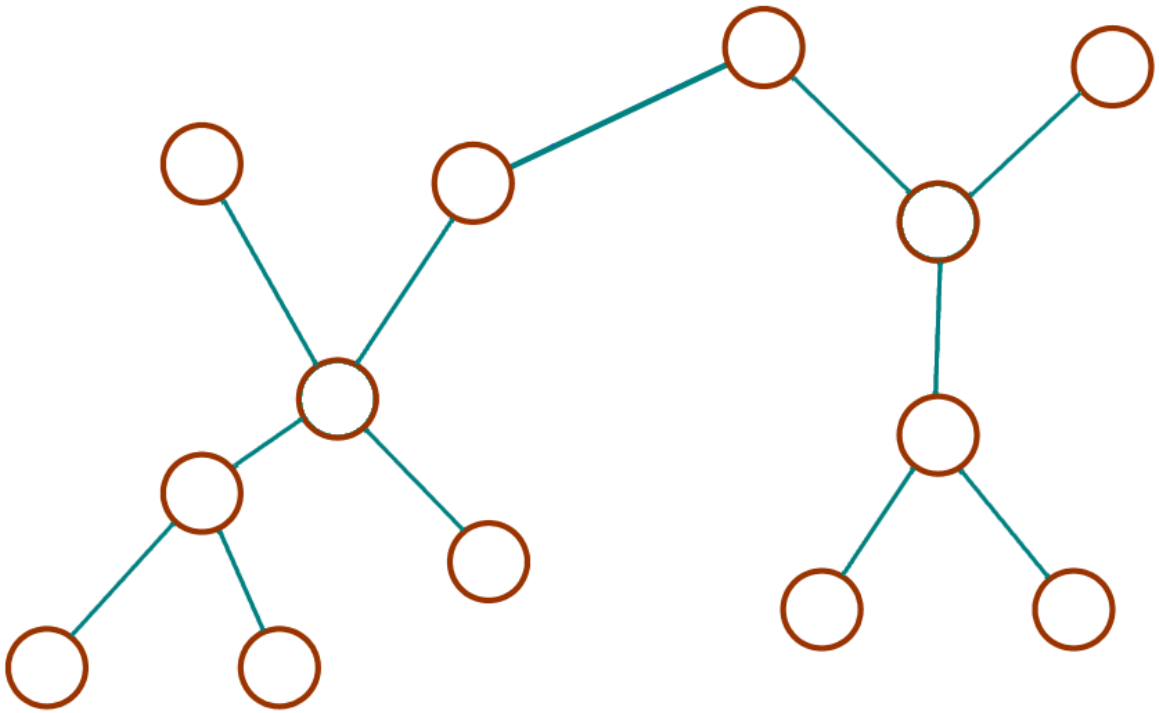
Algorithm for determining causal directions

- compute the maximally weighted spanning tree
- for each connected triplet in the spanning tree
  - compute the joint probability of the triplet
  - compute the marginal dependence and conditional dependence
  - if the marginal dependence is low and the conditional dependence is high put in causal directions corresponding to a collider (multiple parent).
- propagate the causal arrows as far as possible. eg: given:



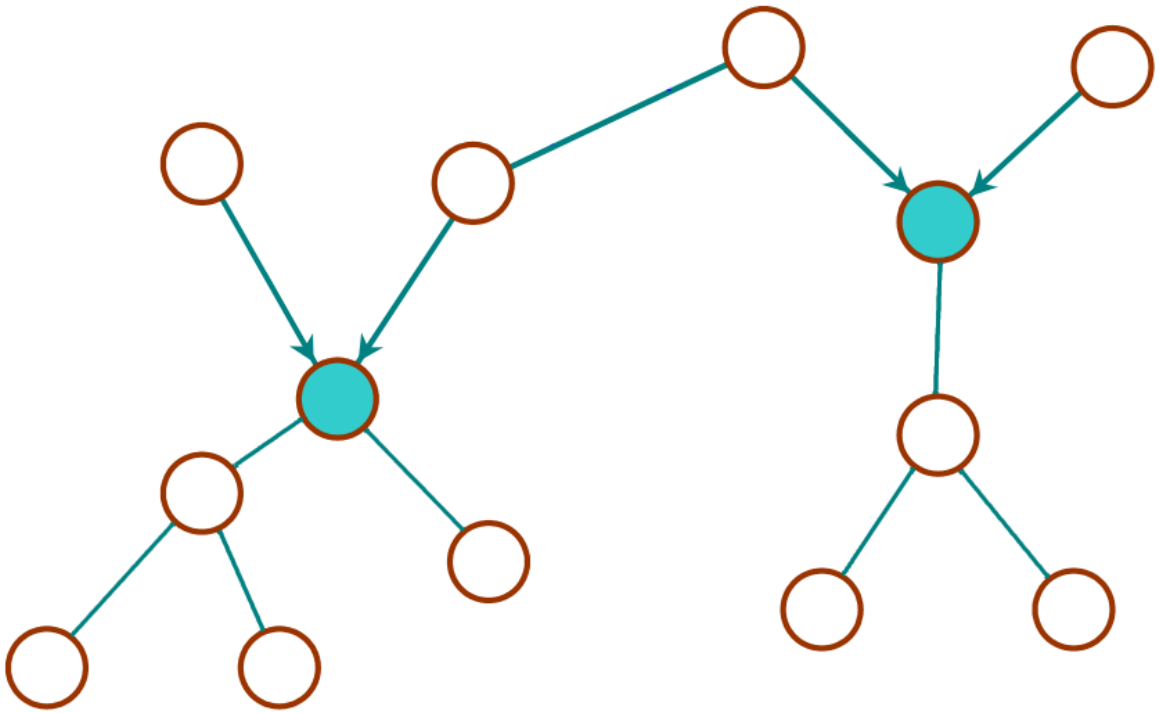
If A and C are independent given B, B is the parent of C and *vice versa*.

Example of finding causal directions



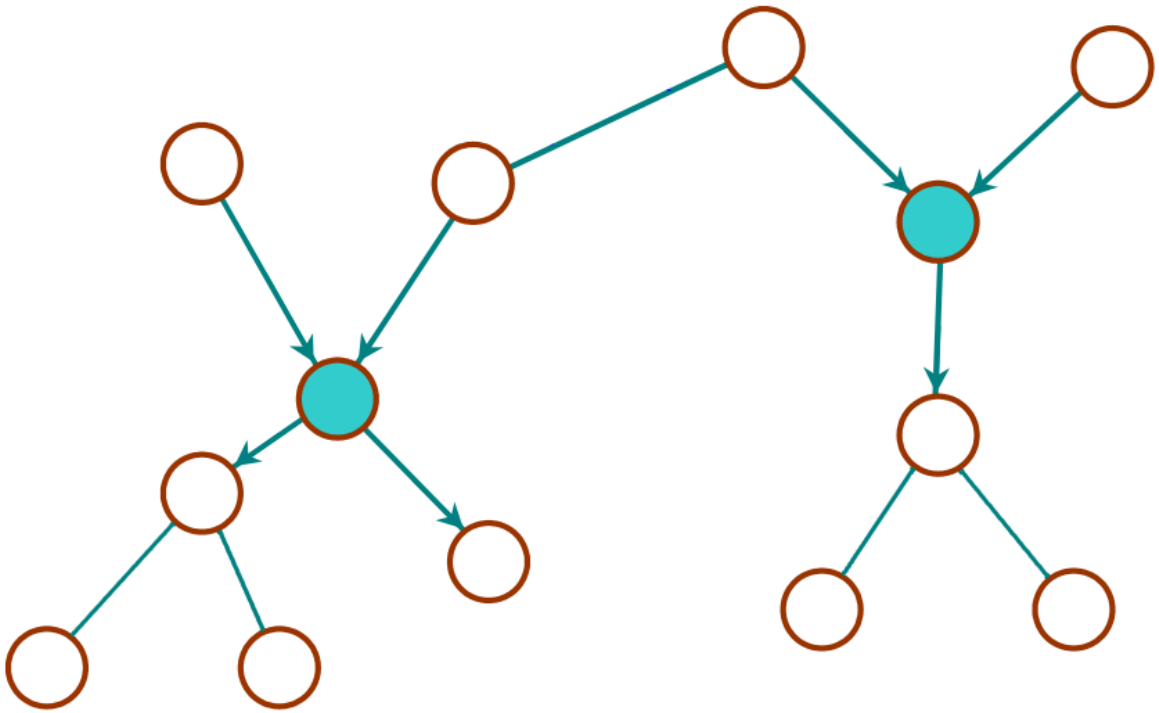
Start by finding the spanning tree

Example of finding causal directions



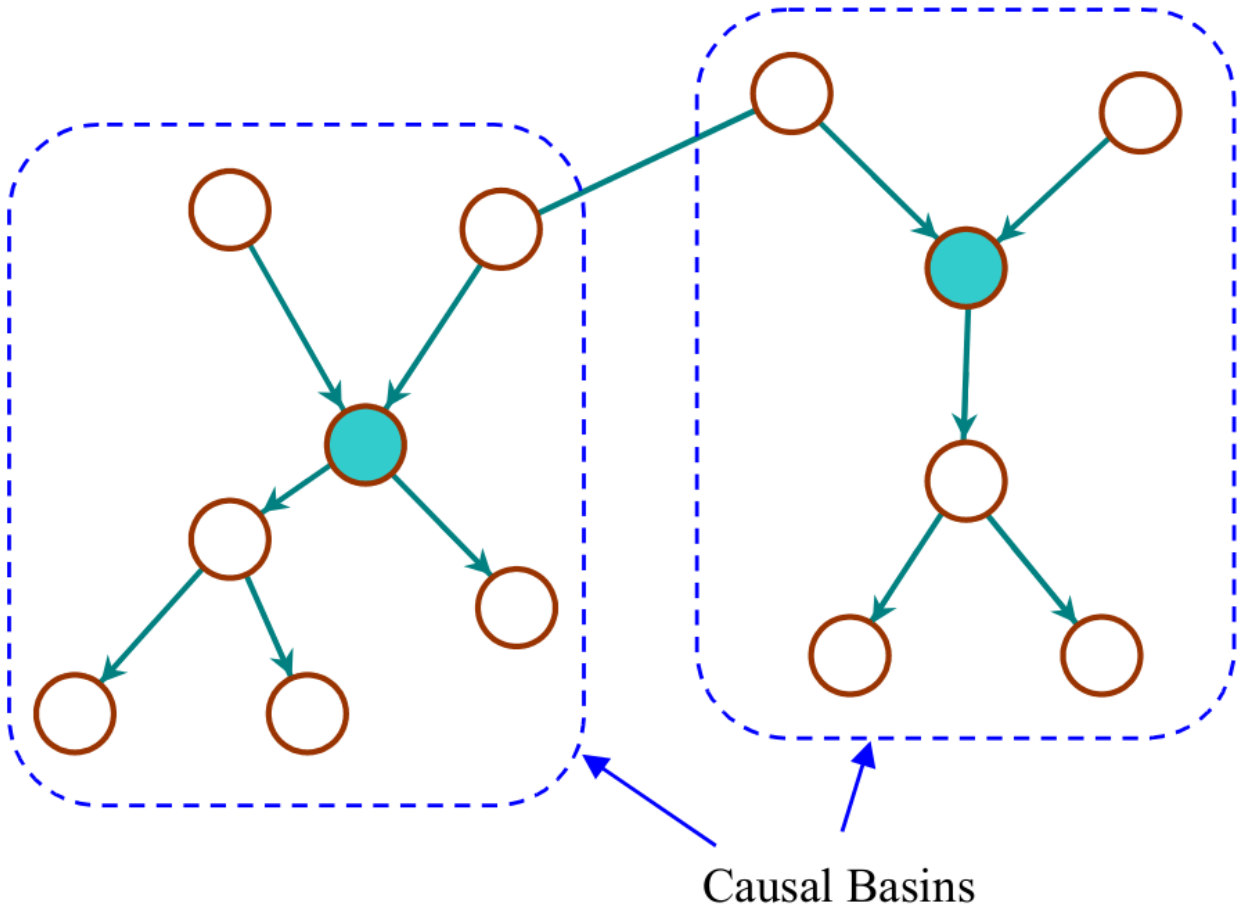
Find all multiple parents using marginal independence

Example of finding causal directions



Propagate arrows where possible

Example of finding causal directions



Continue to the leaf nodes

Problem

Given the following data, what arc directions would you give to the triple $A - B - C$.

a_1	b_1	c_1
a_1	b_1	c_1
a_2	b_1	c_2
a_2	b_1	c_2
a_1	b_2	c_2
a_1	b_2	c_2
a_2	b_2	c_1
a_2	b_2	c_1

Problem

Given the following data, what arc directions would you give to the triple $A - B - C$.

a_1	b_1	c_1
a_1	b_1	c_1
a_2	b_1	c_2
a_2	b_1	c_2
a_1	b_2	c_2
a_1	b_2	c_2
a_2	b_2	c_1
a_2	b_2	c_1

- If we ignore B then A and C are completely independent.
- However given $B = b_1$ or $B = b_2$ there is complete dependence.
- Thus the causal picture is $A \rightarrow B \leftarrow C$

Problems in identifying causal directions

- Our dependency measures are unlikely to give us a decisive result (independent or dependent). We need heuristic thresholds in practice.
- We may find few (or no) cases of multiple parents.
- If there is an unknown (or unaccounted) source of dependency between two variables then the method will give incorrect results.
- We need to compute joint probabilities of triples of variables, hence there may computational problems.

Structure and Parameter Learning

- Bayesian networks combine both structure and parameters.
- We can express our knowledge (if any) about the data by choosing a network structure.
- Alternatively we can learn our network structure from data.
- We then optimise the performance by adjusting the parameters (link matrices).

Other machine learning formalisms

- Neural Networks
  - Neural networks are a class of inference systems which offer just parameter learning.
  - Generally it is very difficult to embed knowledge into a neural net, or infer a structure once the learning phase is complete.
- Rule based systems (Logic)
  - Traditional rule based inference systems have just structure (sometimes with a rudimentary parameter mechanism).
  - They do offer structure modification through methods such as rule induction.
  - However, they are difficult to optimise using large data sets.

Inferring Cause

- Neural networks are most applicable when we have no causal knowledge, but correspondingly we cannot extract causal information from them.
- Rule based systems are most applicable when we have good causal knowledge as they can represent it well.
- Bayesian networks can incorporate known causal relations, but also have the potential to learn cause from data.

Learning Cause

- The notion of learning cause from data caused considerable interest in data mining applications - for example genetics.
- Microarray data can measure simultaneous activities of genes normal and cancerous cells. We could potentially learn causal networks from these using the methods described above.
- However to date the large number of variables, small number of data sets and high experimental error has meant the technique has not met with huge success.
- However it remains an intriguing idea as data volumes and accuracy expands.