Machine Learning and Bayesian Inference

Dr Sean Holden

Computer Laboratory, Room FC06

Telephone extension 63725

Email: sbh11@cl.cam.ac.uk

www.cl.cam.ac.uk/~sbh11/

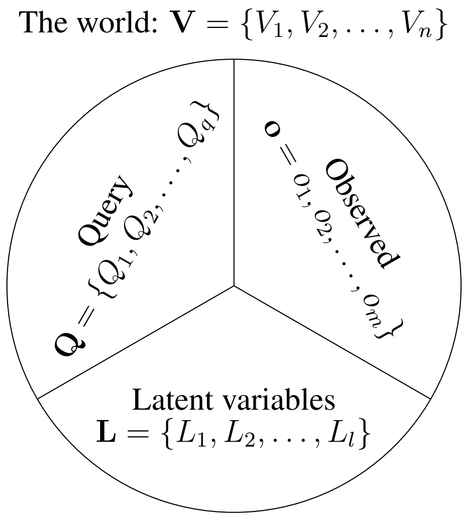
Part IV

Bayesian networks

Markov random fields

Uncertainty: Probability as Degree of Belief

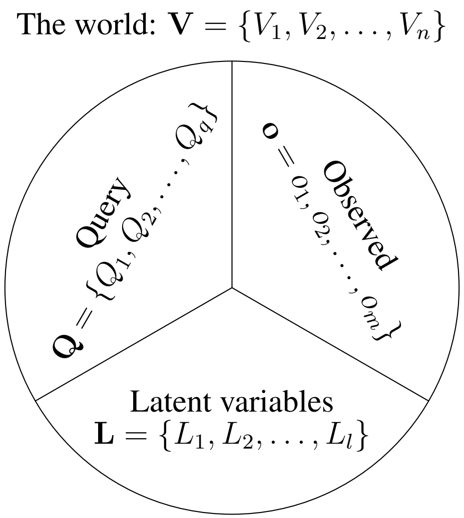
At the start of the course, I presented a *uniform approach* to *knowledge representation and reasoning* using *probability*.



The world is represented by RVs $\mathbf{V} = \{V_1, V_2, \dots, V_n\}$. These are partitioned:

1. Query variables $\mathbf{Q} = \{Q_1, Q_2, \dots, Q_q\}$. We want to *compute a distribution over these*.
2. Observed variables $\mathbf{O} = \{o_1, o_2, \dots, o_m\}$. We *know the values* of these.
3. Latent variables $\mathbf{L} = \{L_1, L_2, \dots, L_l\}$. *Everything else*.

The *latent variables* $\mathbf{L}$ are *all the RVs not in the sets* $\mathbf{Q}$ *or* $\mathbf{O}$.

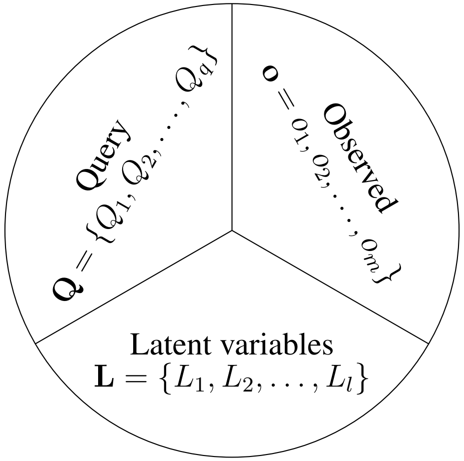


To compute a conditional distribution from a knowledge base $\Pr(\mathbf{V})$ we have to *sum over the latent variables*

$$
\begin{aligned}
\Pr(\mathbf{Q}|o_1, o_2, \ldots, o_m) &= \sum_{\mathbf{L}} \Pr(\mathbf{Q}, \mathbf{L}|o_1, o_2, \ldots, o_m) \\
&= \boxed{\frac{1}{Z} \sum_{\mathbf{L}} \underbrace{\Pr(\mathbf{Q}, \mathbf{L}, o_1, o_2, \ldots, o_m)}_{\text{Knowledge base}}}
\end{aligned}
$$

Bayes' theorem tells us how to update an inference when *new information* is available.

The world: $\mathbf{V} = \{V_1, V_2, \ldots, V_n\}$



For example, if we now receive a new observation $O' = o'$ then

$$\underbrace{\Pr(\mathbf{Q}|o', o_1, o_2, \ldots, o_m)}_{\text{After } O' \text{ observed}} = \frac{1}{Z}\Pr(o'|\mathbf{Q}, o_1, o_2, \ldots, o_m)\underbrace{\Pr(\mathbf{Q}|o_1, o_2, \ldots, o_m)}_{\text{Before } O' \text{ observed}}$$

Simple eh?

HAH!!! No chance...

Even if all your RVs are just Boolean:

- For n RVs knowing the knowledge base $\Pr(\mathbf{V})$ means storing 2^n numbers.
- So it looks as though storage is $O(2^n)$.
- You need to establish 2^n numbers to work with.
- Look at the summations. If there are n latent variables then it appears that time complexity is also $O(2^n)$.
- In reality we might well have $n > 1000$, and of course it's *even worse* if *variables are non-Boolean*.

And it *really is this hard*. The problem in general is *#P-complete*.

Even getting an *approximate solution* is provably intractable.

Having seen that in principle, if not in practice, the full joint distribution alone can be used to perform any inference of interest, we now examine a *practical technique*.

- We introduce the *Bayesian Network (BN)* as a compact representation of the full joint distribution.
- We examine the way in which a BN can be *constructed*.
- We examine the *semantics* of BNs.
- We look briefly at how *inference* can be performed.
- We briefly introduce the *Markov random field (MRF)* as an alternative means of representing a distribution.

Conditional probability—a brief aside...

A brief aside on the dangers of interpreting *implication* versus *conditional probability*:

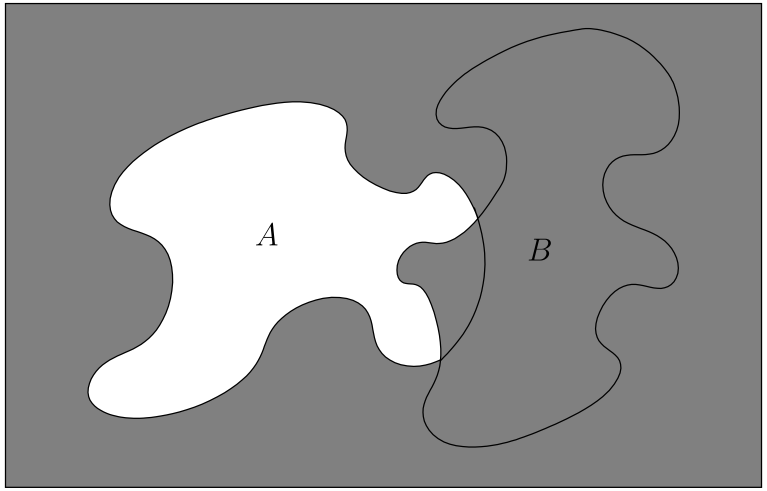
- $\Pr(X = x|Y = y) = 0.1$ does *not* mean that if $Y = y$ is then $\Pr(X = x) = 0.1$.
- $\Pr(X)$ is a *prior probability*. It applies when you *haven't seen* the value of Y.
- The notation $\Pr(X|Y = y)$ is for use when y is the *entire evidence*.
- $\Pr(X|Y = y \wedge Z = z)$ might be very different.

Conditional probability is *not* analogous to *logical implication*.

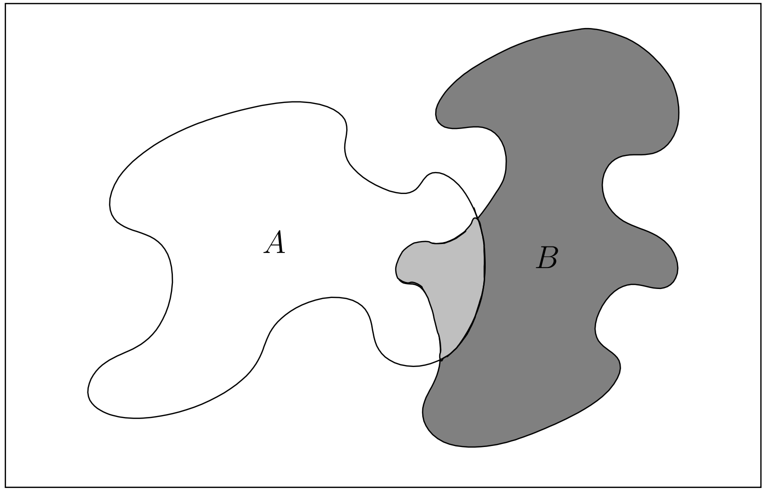
Implication and conditional probability

In general, it is difficult to relate *implication* to *conditional probability*.

$\Pr(A \to B) = \Pr(\neg A \vee B)$



$\Pr(A|B) = \frac{\Pr(A \wedge B)}{\Pr(B)}$



Imagine that `fish` are very rare, and most fish can `swim`.

With implication,

$$\Pr(\texttt{fish} \to \neg\texttt{swim}) = \Pr(\neg\texttt{fish} \vee \neg\texttt{swim}) = \text{LARGE!}$$

With conditional probability,

$$\Pr(\neg\texttt{swim}|\texttt{fish}) = \frac{\Pr(\neg\texttt{swim} \wedge \texttt{fish})}{\Pr(\texttt{fish})} = \text{SMALL!}$$

Bayesian networks: exploiting independence

One of the key reasons for the introduction of *Bayesian networks* is to let us *exploit independence*.

The initial pay-off is that this *makes it easier to represent* $\Pr(\mathbf{V})$.

A further pay-off is that it *introduces structure* that can lead to *more efficient inference*.

Here is a *very simple* example.

If I toss a coin and roll a die, the full joint distribution of outcomes requires $2 \times 6 = 12$ numbers to be specified.

	⚀	⚁	⚂	⚃	⚄	⚅
H	0.014	0.028	0.042	0.057	0.071	0.086
T	0.033	0.067	0.1	0.133	0.167	0.2

Here $\Pr(\texttt{Coin} = H) = 0.3$ and the die has probability $i/21$ for the ith outcome.

Exploiting independence

BUT: if we assume the outcomes are independent then

$$\Pr(\texttt{Coin}, \texttt{Dice}) = \Pr(\texttt{Coin})\Pr(\texttt{Dice})$$

Where $\Pr(\texttt{Coin})$ has two numbers and $\Pr(\texttt{Dice})$ has six.

So instead of 12 numbers we only need 8.

Exploiting independence

A slightly more complex example:

	$\texttt{CP}$		$\neg\texttt{CP}$	
	$\texttt{SB}$	$\neg\texttt{SB}$	$\texttt{SB}$	$\neg\texttt{SB}$
$\texttt{HD}$	0.024	0.006	0.016	0.004
$\neg\texttt{HD}$	0.0019	0.0076	0.1881	0.7524

- $\texttt{HD}$ = Heart disease
- $\texttt{CP}$ = Chest pain
- $\texttt{SB}$ = Shortness of breath

Similarly, say instead of just considering $\texttt{HD}$, $\texttt{SB}$ and $\texttt{CP}$ we also consider the outcome of the *Oxford versus Cambridge tiddlywinks competition* $\texttt{TC}$:

$$\texttt{TC} = \{\text{Oxford}, \text{Cambridge}, \text{Draw}\}.$$

Exploiting independence

Now

$$\mathbf{Pr}\left(\mathtt{HD}, \mathtt{SB}, \mathtt{CP}, \mathtt{TC}\right) = \mathbf{Pr}\left(\mathtt{TC} \middle| \mathtt{HD}, \mathtt{SB}, \mathtt{CP}\right) \mathbf{Pr}\left(\mathtt{HD}, \mathtt{SB}, \mathtt{CP}\right).$$

Assuming that the patient is not an *extraordinarily keen fan of tiddlywinks*, their cardiac health has nothing to do with the outcome, so

$$\mathbf{Pr}\left(\mathtt{TC} \middle| \mathtt{HD}, \mathtt{SB}, \mathtt{CP}\right) = \mathbf{Pr}\left(\mathtt{TC}\right)$$

and $2 \times 2 \times 2 \times 3 = 24$ numbers has been reduced to $3 + 8 = 11$.

Conditional independence

However although in this case we might not be able to exploit independence directly we *can* say that

$$\Pr(\texttt{CP}, \texttt{SB} | \texttt{HD}) = \Pr(\texttt{CP} | \texttt{HD}) \Pr(\texttt{SB} | \texttt{HD})$$

which simplifies matters.

Conditional independence: $A \perp B | C$

- A is *conditionally independent of* B *given* C, written $A \perp B | C$, if

$$\Pr(A, B | C) = \Pr(A | C) \Pr(B | C).$$

- If we know that C is the case then A and B are independent.
- Equivalently $\Pr(A | B, C) = \Pr(A | C)$. (Prove this!)

Although $\texttt{CP}$ and $\texttt{SB}$ are *not* independent, they do not directly influence one another *in a patient known to have heart disease*.

This is much nicer!

$$\Pr(\texttt{HD} | \texttt{CP}, \texttt{SB}) \propto \Pr(\texttt{CP} | \texttt{HD}) \Pr(\texttt{SB} | \texttt{HD}) \Pr(\texttt{HD})$$

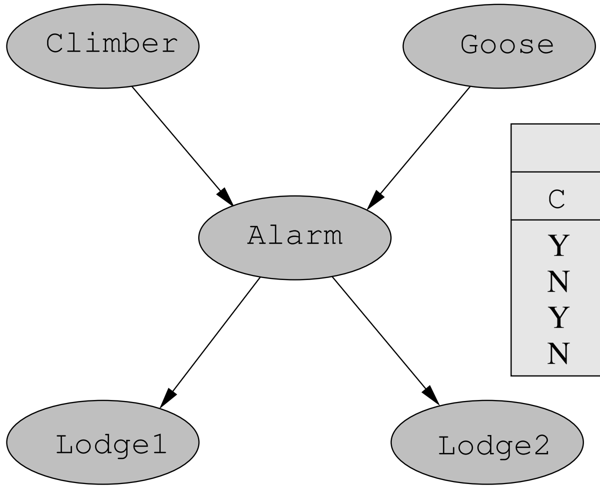
After a *regrettable incident* involving an *inflatable gorilla*, a famous College has decided to install an alarm for the detection of roof climbers.

- The alarm is *very* good at detecting climbers.
- Unfortunately, it is also sometimes triggered when one of the *extremely fat geese* that lives in the College lands on the roof.
- One porter's lodge is near the alarm, and inhabited by a chap with *excellent hearing* and a *pathological hatred* of roof climbers: he *always* reports an alarm. His hearing is so good that he sometimes thinks he hears an alarm, *even when there isn't one*.
- Another porter's lodge is a good distance away and inhabited by an *old chap* with *dodgy hearing* who likes to listen to his collection of *DEATH METAL* with the sound turned up.

Bayesian networks

Pr(Climber)
Yes: 0.05
No: 0.95

Pr(Goose)
Yes: 0.2
No: 0.8



Pr(A|C, G)		
C	G	Pr(A|C, G)
Y	Y	0.98
N	N	0.08
Y	N	0.96
N	Y	0.2

Pr(L1|A)	
a	0.99
¬a	0.08

Pr(L2|A)	
a	0.6
¬a	0.001

Bayesian networks

Also called *probabilistic/belief/causal networks* or *knowledge maps*.

- Each node is a *random variable (RV)*.
- Each node N_i has a distribution

$$\mathbf{Pr}\left(N_i | \mathtt{parents}(N_i)\right)$$

- A Bayesian network is a *directed acyclic graph*.
- Roughly speaking, an arrow from N to M means N directly affects M.

Note that:

- In the present example all RVs are *discrete* (in fact Boolean) and so in all cases $\Pr(N_i|\text{parents}(N_i))$ can be represented as a *table of numbers*.
- `Climber` and `Goose` have only *prior* probabilities.
- All RVs here are Boolean, so a node with p parents requires 2^p numbers.

A BN with n nodes represents the full joint probability distribution for those nodes as

$$\mathbf{Pr}(N_1 = n_1, N_2 = n_2, \ldots, N_n = n_n) = \prod_{i=1}^{n} \mathbf{Pr}(N_i = n_i|\text{parents}(N_i)).$$

For example

$$\begin{aligned}\mathbf{Pr}(\neg\mathtt{C}, \neg\mathtt{G}, \mathtt{A}, \mathtt{L1}, \mathtt{L2}) = \mathbf{Pr}(\mathtt{L1}|\mathtt{A})\,\mathbf{Pr}(\mathtt{L2}|\mathtt{A})\,\mathbf{Pr}(\mathtt{A}|\neg\mathtt{C}, \neg\mathtt{G})\,\mathbf{Pr}(\neg\mathtt{C}))\,\Pr(\neg\mathtt{G} \\ = 0.99 \times 0.6 \times 0.08 \times 0.95 \times 0.8.\end{aligned}$$

Semantics

In general $\Pr(A, B) = \Pr(A|B)\Pr(B)$ so

$$\Pr(N_1, \ldots, N_n) = \Pr(N_n|N_{n-1}, \ldots, N_1) \boxed{\Pr(N_{n-1}, \ldots, N_1)}.$$

Repeating this gives

$$\begin{aligned}\Pr(N_1, \ldots, N_n) &= \Pr(N_n|N_{n-1}, \ldots, N_1) \boxed{\Pr(N_{n-1}|N_{n-2}, \ldots, N_1) \cdots \Pr(N_1)} \\ &= \prod_{i=1}^{n} \Pr(N_i|N_{i-1}, \ldots, N_1).\end{aligned}$$

Now compare equations. We see that BNs make the assumption

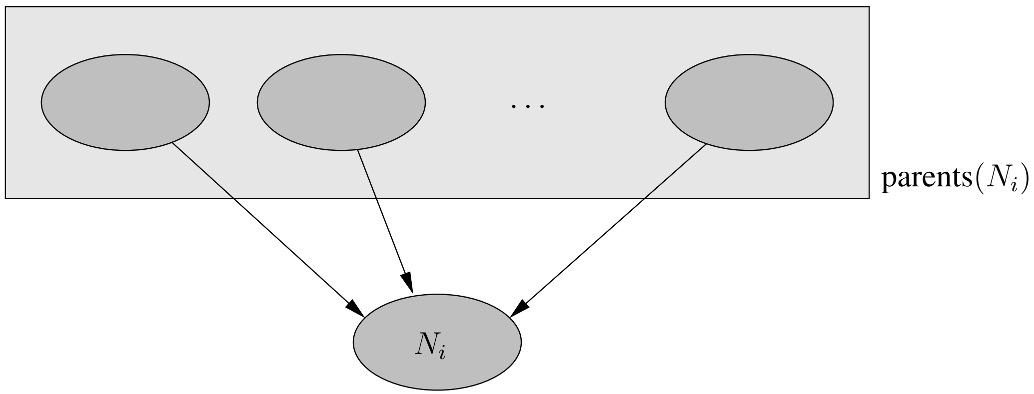
$$\Pr(N_i|N_{i-1}, \ldots, N_1) = \Pr(N_i|\text{parents}(N_i))$$

for each node, assuming that $\text{parents}(N_i) \subseteq \{N_{i-1}, \ldots, N_1\}$.

Each N_i is conditionally independent of its predecessors given its parents .

Semantics

- When constructing a BN we want to make sure the preceding property holds.
- This means we need to take care over *ordering*.
- In general *causes should directly precede effects*.

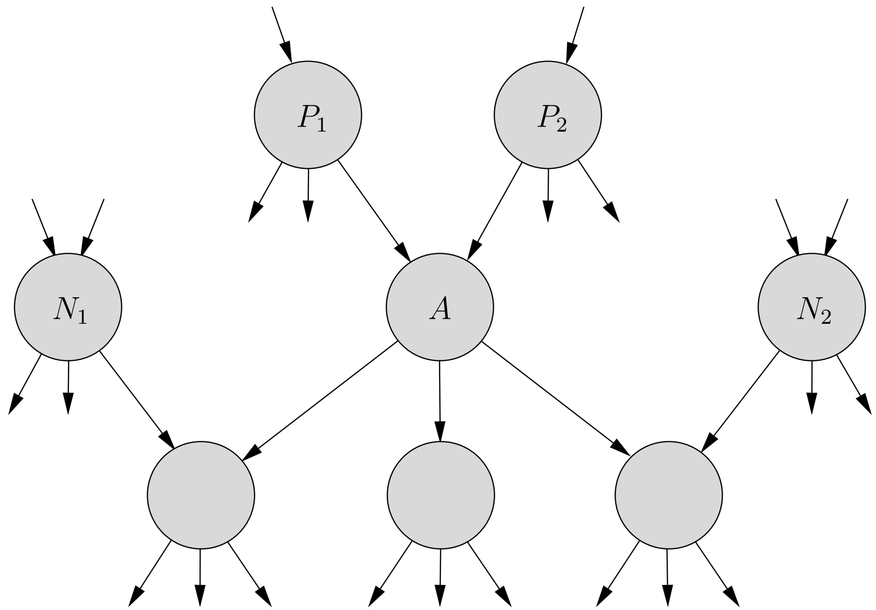


Here, parents(N_i) contains all preceding nodes having a *direct influence* on N_i.

Semantics

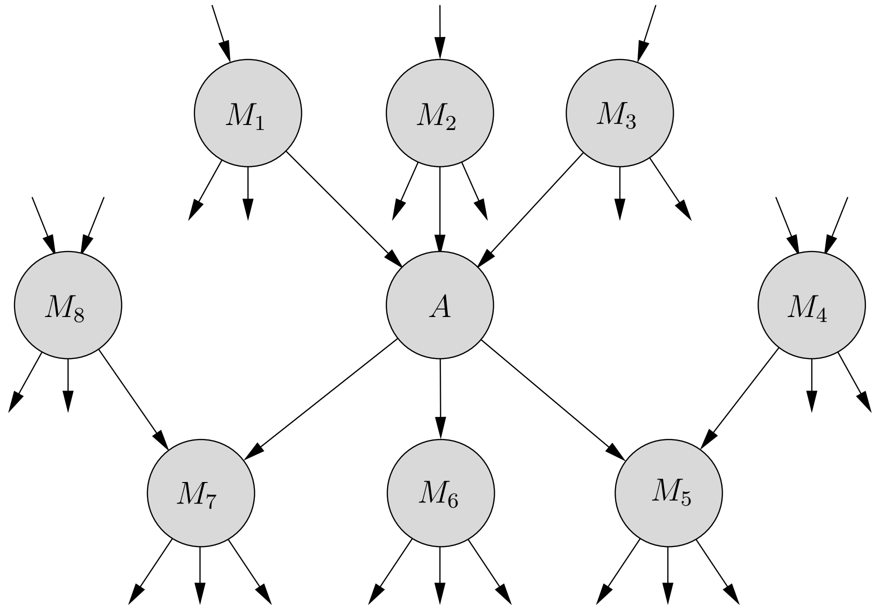
But its not quite that straightforward: what if we want to talk about nodes *other than predecessors and parents*?

For example, it is possible to show:



Any node A is conditionally independent of the N_i—its *non-descendants*—given the P_i—its parents.

It is also possible to show:



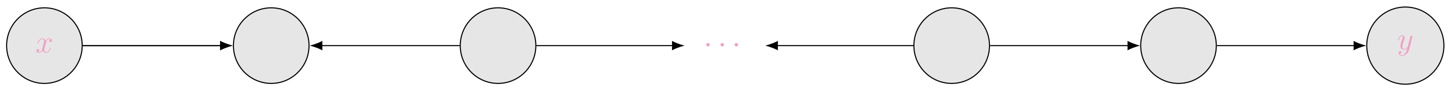
Any node A is conditionally independent of all other nodes given the *Markov blanket* M_i—that is, its *parents*, its *children* and its *childrens' parents*.

Semantics: what's REALLY going on here?

There is a *general method* for inferring exactly *what conditional independences are implied by a Bayesian network*.

Let X, Y and Z be disjoint subsets of the RVs.

Consider a *path* p consisting of directed (in any orientation) edges from some $x \in X$ to some $y \in Y$. For example

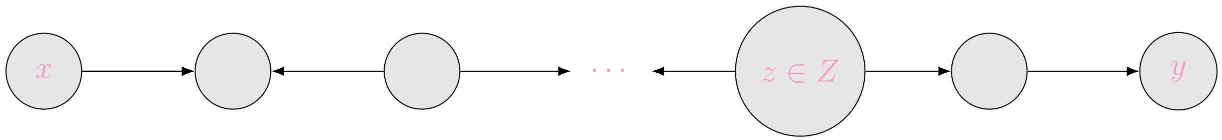


The path p is said to be *blocked* by Z if one of *three conditions* holds...

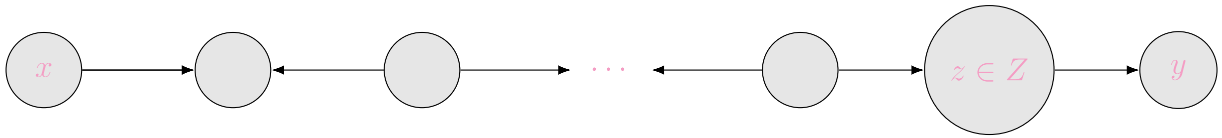
Semantics: what's REALLY going on here?

Path p is *blocked* with respect to Z if:

1. p contains a node $z \in Z$ that is *tail-to-tail*:

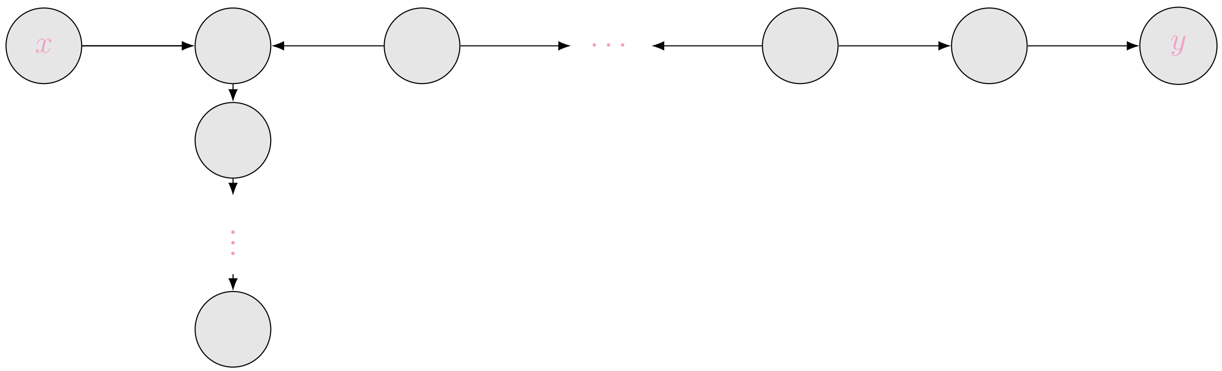


2. p contains a node $z \in Z$ that is *head-to-tail*:



   (Similarly if the node is *tail-to-head*.)

3. p contains a node N that is *head-to-head*, $N \notin Z$, and none of N's descendents is in Z:



Semantics: what's REALLY going on here?

Finally:

1. X and Y are *d-separated by* Z if *all paths* p from some $x \in X$ to some $y \in Y$ are blocked.
2. If X and Y are *d-separated by* Z then $X \perp Y | Z$.

More complex nodes

How do we represent

$$\Pr\left(N_i | \text{parents}(N_i)\right)$$

when nodes can denote *general discrete and/or continuous RVs*?

- BNs containing both kinds of RV are called *hybrid BNs*.
- Naive *discretisation* of continuous RVs tends to result in both a reduction in accuracy and large tables.
- $O(2^p)$ might still be large enough to be unwieldy.
- We can instead attempt to use *standard and well-understood* distributions, such as the *Gaussian*.
- This will typically require only a small number of parameters to be specified.

More complex nodes

Example: a continuous RV with one continuous and one discrete parent.

$$\Pr(\texttt{Speed of car}|\texttt{Throttle position}, \texttt{Tuned engine})$$

where $\texttt{SC}$ and $\texttt{TP}$ are continuous and $\texttt{TE}$ is Boolean.

- For a specific setting of $\texttt{ET} = \texttt{true}$ it might be the case that $\texttt{SC}$ increases with $\texttt{TP}$, but that some uncertainty is involved

$$\Pr(\texttt{SC}|\texttt{TP}, \texttt{et}) = N(g_{\texttt{et}}\texttt{TP} + c_{\texttt{et}}, \sigma^2_{\texttt{et}}).$$

- For an un-tuned engine we might have a similar relationship with a different behaviour

$$\Pr(\texttt{SC}|\texttt{TP}, \neg\texttt{et}) = N(g_{\neg\texttt{et}}\texttt{TP} + c_{\neg\texttt{et}}, \sigma^2_{\neg\texttt{et}}).$$

There is a set of parameters $\{g, c, \sigma\}$ for each possible value of the discrete RV.

More complex nodes

Example: a discrete RV with a continuous parent

$$\mathbf{Pr}\left(\texttt{Go roofclimbing}|\texttt{Size of fine}\right).$$

We could for example use the *probit distribution*

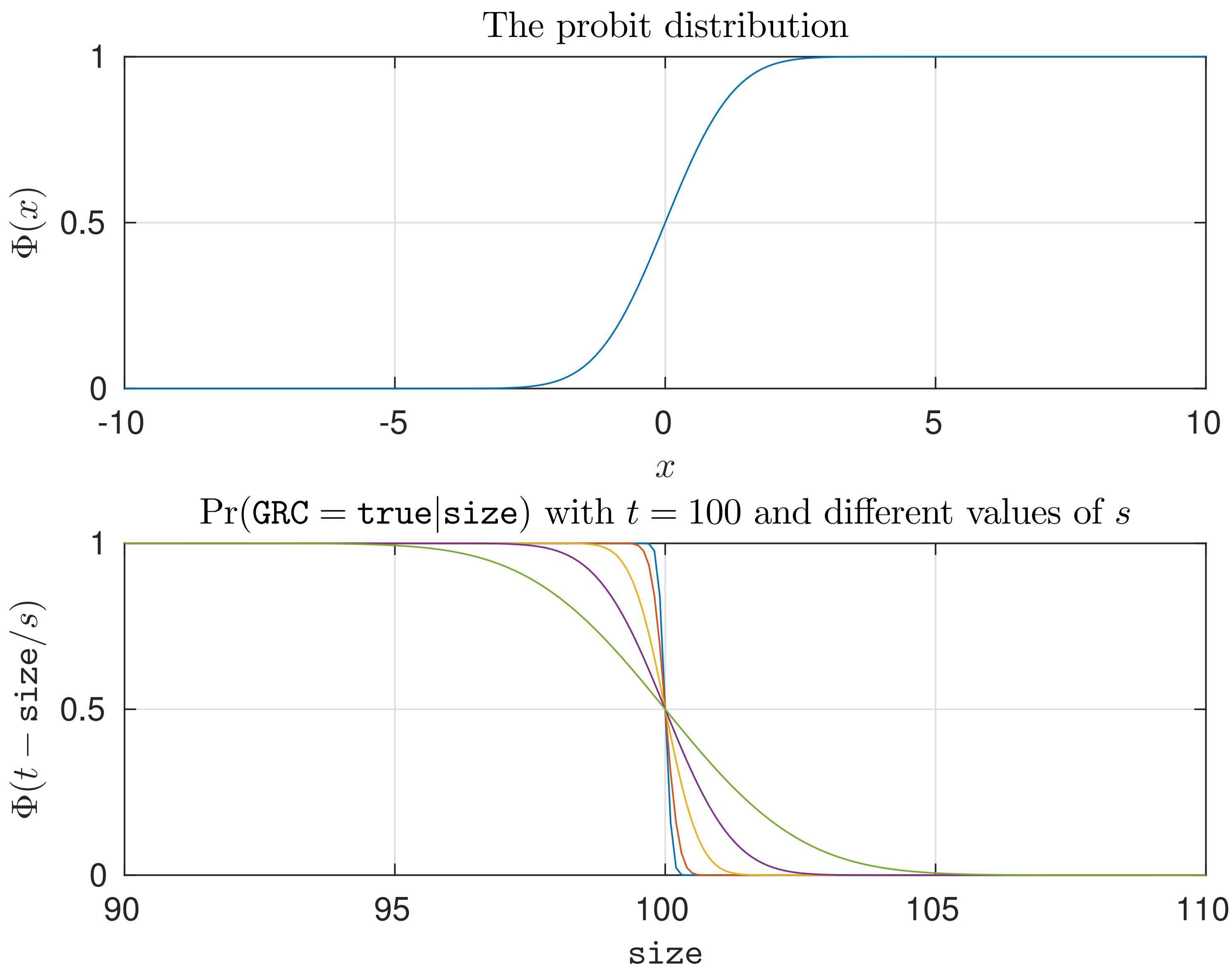
$$\mathbf{Pr}\left(\texttt{Go roofclimbing} = \texttt{true}|\texttt{size}\right) = \Phi\left(\frac{t - \texttt{size}}{s}\right)$$

where

$$\Phi(x) = \int_{-\infty}^{x} N(y)dy$$

and N is the Gaussian density with *zero mean and variance* 1.

More complex nodes



Basic inference

We saw earlier that the full joint distribution can be used to perform *all inference tasks*:

$$\Pr\left(\mathbf{Q} | o_1, o_2, \ldots, o_m\right) = \frac{1}{Z} \sum_{\mathbf{L}} \Pr\left(\mathbf{Q}, \mathbf{L}, o_1, o_2, \ldots, o_m\right)$$

where

- $\mathbf{Q}$ is the query.
- $o_1, o_2, \ldots, o_m$ are the observations.
- $\mathbf{L}$ are the latent variables.
- $1/Z$ normalises the distribution.
- The query, observations and latent variables are a partition of the set $\mathbf{V} = \{V_1, V_2, \ldots, V_n\}$ of all variables.

Basic inference

As the BN fully describes the full joint distribution

$$\Pr\left(\mathbf{Q}, \mathbf{L}, o_1, o_2, \ldots, o_m\right) = \prod_{i=1}^{n} \Pr(V_i | \text{parents}(V_i))$$

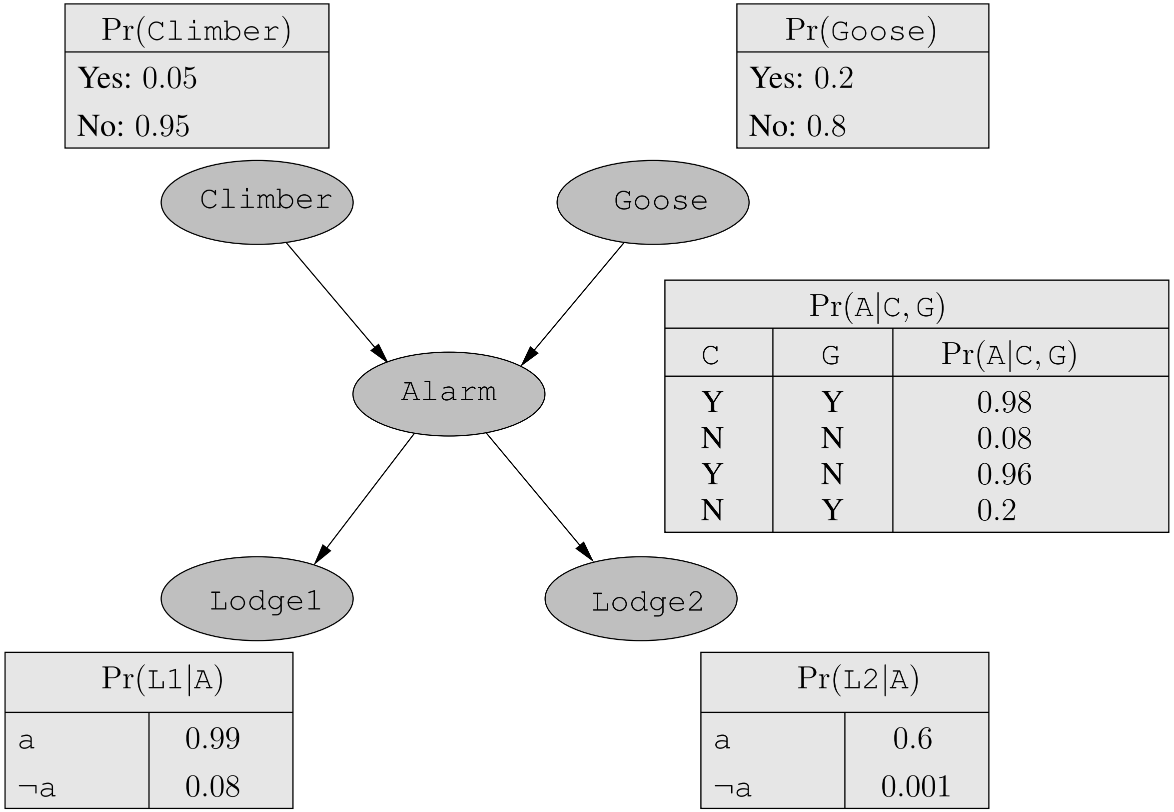
it can be used to perform inference in the *obvious* way

$$\Pr\left(\mathbf{Q} | o_1, o_2, \ldots, o_m\right) \propto \sum_{\mathbf{L}} \prod_{i=1}^{n} \Pr(V_i | \text{parents}(V_i))$$

but this is *in practice problematic* for obvious reasons.

- More sophisticated algorithms aim to achieve this *more efficiently*.
- For complex BNs we resort to *approximation techniques*.

$\Pr(\mathbf{Q}, \mathbf{L}, o_1, \ldots, o_m)$ has a particular form expressing conditional independences:



Pr(Climber)
Yes: 0.05
No: 0.95

Pr(Goose)
Yes: 0.2
No: 0.8

C	G	Pr(A|C, G)
Y	Y	0.98
N	N	0.08
Y	N	0.96
N	Y	0.2

Pr(L1|A)	
a	0.99
¬a	0.08

Pr(L2|A)	
a	0.6
¬a	0.001

$$\Pr(C, G, A, L1, L2) = \Pr(C)\Pr(G)\Pr(A|C,G)\Pr(L1|A)\Pr(L2|A).$$

Consider the computation of the query $\Pr\left(C|l1, l2\right)$

We have

$$\Pr\left(C|l1, l2\right) \propto \sum_{A} \sum_{G} \Pr\left(C\right) \Pr\left(G\right) \Pr\left(A|C, G\right) \Pr\left(l1|A\right) \Pr\left(l2|A\right).$$

Here there are 5 multiplications for each set of values that appears for summation, and there are 4 such values.

In general this gives time complexity $O(n2^n)$ for n Boolean RVs.

The naive implementation of this approach yields the *Enumerate-Joint-Ask* algorithm, which unfortunately requires $O(2^n)$ time and space for n Boolean RVs.

The *enumeration-ask* algorithm improves matters to $O(2^n)$ time and $O(n)$ space by performing the computation *depth-first*.

However matters can be improved further by avoiding *duplication of computations*.

Performing exact inference

Looking more closely we see that

$$\begin{aligned}\Pr(C|l1,l2) &\propto \sum_A \sum_G \Pr(C)\Pr(G)\Pr(A|C,G)\Pr(l1|A)\Pr(l2|A)\\ &= \frac{1}{Z}\Pr(C)\sum_A \Pr(l1|A)\Pr(l2|A)\sum_G \Pr(G)\Pr(A|C,G)\\ &= \frac{1}{Z}\Pr(C)\sum_G \Pr(G)\sum_A \Pr(A|C,G)\Pr(l1|A)\Pr(l2|A).\end{aligned}$$

There is some freedom in terms of how we *factorize* the expression.

This is a result of introducing *assumptions about conditional independence*.

Taking the second possibility:

$$\underbrace{\Pr(C)}_{C}\sum_{G}\underbrace{\Pr(G)}_{G}\sum_{A}\underbrace{\Pr(A|C,G)}_{A}\underbrace{\Pr(l1|A)}_{L1}\underbrace{\Pr(l2|A)}_{L2}$$

where C, G, A, $L1$, $L2$ denote the relevant *factors*.

The basic idea is to evaluate this from right to left (or in terms of the tree, bottom up) *storing results* as we progress and *re-using them* when necessary.

$\Pr(l1|A)$ depends on the value of A. We store it as a table $\mathbf{F}_{L1}(A)$. Similarly for $\Pr(l2|A)$.

$$\mathbf{F}_{L1}(A) = \begin{pmatrix} 0.99 \\ 0.08 \end{pmatrix} \quad \mathbf{F}_{L2}(A) = \begin{pmatrix} 0.6 \\ 0.001 \end{pmatrix}$$

as $\Pr(l1|a) = 0.99$, $\Pr(l1|\neg a) = 0.08$ and so on.

Similarly for $\Pr(A|C,G)$, which is dependent on A, C and G

$$\mathbf{F}_A(A,C,G) = \begin{array}{|ccc|c|} \hline A & C & G & \mathbf{F}_A(A,C,G) \\ \hline \top & \top & \top & 0.98 \\ \top & \top & \bot & 0.96 \\ \top & \bot & \top & 0.2 \\ \top & \bot & \bot & 0.08 \\ \bot & \top & \top & 0.02 \\ \bot & \top & \bot & 0.04 \\ \bot & \bot & \top & 0.8 \\ \bot & \bot & \bot & 0.92 \\ \hline \end{array}$$

Can we write $\Pr(A|C,G)\Pr(l1|A)\Pr(l2|A)$ as

$$\mathbf{F}_A(A,C,G)\mathbf{F}_{L1}(A)\mathbf{F}_{L2}(A)$$

in a reasonable way?

Yes, provided *multiplication of factors* is defined correctly. Looking at

$$\Pr(C) \sum_{G} \Pr(G) \sum_{A} \Pr(A|C,G) \Pr(l1|A) \Pr(l2|A)$$

note that:

1. The values of the product

   $$\Pr(A|C,G) \Pr(l1|A) \Pr(l2|A)$$

   in the summation over A depend on the values of C and G external to it, and the values of A.

2. So

   $$\mathbf{F}_A(A,C,G)\mathbf{F}_{L1}(A)\mathbf{F}_{L2}(A)$$

   should be a table collecting values where correspondences between RVs are maintained.

This leads to a definition for *multiplication of factors* best given by example.

Performing exact inference: variable elimination

$$\mathbf{F}(A,B)\mathbf{F}(B,C) = \mathbf{F}(A,B,C)$$

where

A	B	$\mathbf{F}(A,B)$	B	C	$\mathbf{F}(B,C)$	A	B	C	$\mathbf{F}(A,B,C)$
$\top$	$\top$	0.3	$\top$	$\top$	0.1	$\top$	$\top$	$\top$	0.3×0.1
$\top$	$\perp$	0.9	$\top$	$\perp$	0.8	$\top$	$\top$	$\perp$	0.3×0.8
$\perp$	$\top$	0.4	$\perp$	$\top$	0.8	$\top$	$\perp$	$\top$	0.9×0.8
$\perp$	$\perp$	0.1	$\perp$	$\perp$	0.3	$\top$	$\perp$	$\perp$	0.9×0.3
						$\perp$	$\top$	$\top$	0.4×0.1
						$\perp$	$\top$	$\perp$	0.4×0.8
						$\perp$	$\perp$	$\top$	0.1×0.8
						$\perp$	$\perp$	$\perp$	0.1×0.3

This process gives us

$$\mathbf{F}_A(A,C,G)\mathbf{F}_{L1}(A)\mathbf{F}_{L2}(A) = \begin{array}{|ccc|c|}\hline A & C & G & \\ \hline \top & \top & \top & 0.98 \times 0.99 \times 0.6 \\ \top & \top & \bot & 0.96 \times 0.99 \times 0.6 \\ \top & \bot & \top & 0.2 \times 0.99 \times 0.6 \\ \top & \bot & \bot & 0.08 \times 0.99 \times 0.6 \\ \bot & \top & \top & 0.02 \times 0.08 \times 0.001 \\ \bot & \top & \bot & 0.04 \times 0.08 \times 0.001 \\ \bot & \bot & \top & 0.8 \times 0.08 \times 0.001 \\ \bot & \bot & \bot & 0.92 \times 0.08 \times 0.001 \\ \hline \end{array}$$

How about

$$\mathbf{F}_{\overline{A},L1,L2}(C,G) = \sum_{A} \mathbf{F}_A(A,C,G)\mathbf{F}_{L1}(A)\mathbf{F}_{L2}(A)$$

To denote the fact that A has been summed out we place a bar over it in the notation.

$$\begin{aligned}\sum_{A} \mathbf{F}_A(A,C,G)\mathbf{F}_{L1}(A)\mathbf{F}_{L2}(A) =& \mathbf{F}_A(a,C,G)\mathbf{F}_{L1}(a)\mathbf{F}_{L2}(a) \\ & + \mathbf{F}_A(\neg a,C,G)\mathbf{F}_{L1}(\neg a)\mathbf{F}_{L2}(\neg a)\end{aligned}$$

where

$$\mathbf{F}_A(a,C,G) = \begin{array}{|cc|c|}\hline C & G & \\ \hline \top & \top & 0.98 \\ \top & \bot & 0.96 \\ \bot & \top & 0.2 \\ \bot & \bot & 0.08 \\ \hline \end{array} \quad \mathbf{F}_{L1}(a) = 0.99 \quad \mathbf{F}_{L2}(a) = 0.6$$

and similarly for $\mathbf{F}_A(\neg a,C,G)$, $\mathbf{F}_{L1}(\neg a)$ and $\mathbf{F}_{L2}(\neg a)$.

Performing exact inference: variable elimination

$$\mathbf{F}_A(a, C, G)\mathbf{F}_{L1}(a)\mathbf{F}_{L2}(a) = \begin{array}{|cc|c|}\hline C & G & \\ \hline \top & \top & 0.98 \times 0.99 \times 0.6 \\ \top & \bot & 0.96 \times 0.99 \times 0.6 \\ \bot & \top & 0.2 \times 0.99 \times 0.6 \\ \bot & \bot & 0.08 \times 0.99 \times 0.6 \\ \hline \end{array}$$

$$\mathbf{F}_A(\neg a, C, G)\mathbf{F}_{L1}(\neg a)\mathbf{F}_{L2}(\neg a) = \begin{array}{|cc|c|}\hline C & G & \\ \hline \top & \top & 0.02 \times 0.08 \times 0.001 \\ \top & \bot & 0.04 \times 0.08 \times 0.001 \\ \bot & \top & 0.8 \times 0.08 \times 0.001 \\ \bot & \bot & 0.92 \times 0.08 \times 0.001 \\ \hline \end{array}$$

$$\mathbf{F}_{\overline{A},L1,L2}(C, G) = \begin{array}{|cc|c|}\hline C & G & \\ \hline \top & \top & (0.98 \times 0.99 \times 0.6) + (0.02 \times 0.08 \times 0.001) \\ \top & \bot & (0.96 \times 0.99 \times 0.6) + (0.04 \times 0.08 \times 0.001) \\ \bot & \top & (0.2 \times 0.99 \times 0.6) + (0.8 \times 0.08 \times 0.001) \\ \bot & \bot & (0.08 \times 0.99 \times 0.6) + (0.92 \times 0.08 \times 0.001) \\ \hline \end{array}$$

Now, say for example we have $\neg c$, g. Then doing the calculation explicitly would give

$$\sum_A \text{Pr}\,(A|\neg c, g)\text{Pr}\,(l1|A))\,\text{Pr}\,(l2|A)$$

$$= \text{Pr}\,(a|\neg c, g)\,\text{Pr}\,(l1|a)\,\text{Pr}\,(l2|a) + \text{Pr}\,(\neg a|\neg c, g)\,\text{Pr}\,(l1|\neg a)\,\text{Pr}\,(l2|\neg a)$$
$$= (0.2 \times 0.99 \times 0.6) + (0.8 \times 0.08 \times 0.001)$$

which matches!

Continuing in this manner form

$$\mathbf{F}_{G,\overline{A},L1,L2}(C,G) = \mathbf{F}_G(G)\mathbf{F}_{\overline{A},L1,L2}(C,G)$$

sum out G to obtain $\mathbf{F}_{\overline{G},\overline{A},L1,L2}(C) = \sum_G \mathbf{F}_G(G)\mathbf{F}_{\overline{A},L1,L2}(C,G)$, form

$$\mathbf{F}_{C,\overline{G},\overline{A},L1,L2} = \mathbf{F}_C(C)\mathbf{F}_{\overline{G},\overline{A},L1,L2}(C)$$

and normalise.

What's the computational complexity now?

- For Bayesian networks with *suitable structure* we can perform inference in *linear time and space*.
- However in the worst case it is still *#P-hard*.

Consequently, we may need to resort to *approximate inference*.

Markov chain Monte Carlo (MCMC) methods also provide a method for performing *approximate inference* in *Bayesian networks*.

Say a system can be in a state $\mathbf{S}$ and moves from state to state in discrete time steps according to a probabilistic transition

$$\Pr\left(\mathbf{S} \rightarrow \mathbf{S}'\right).$$

Let $\pi_t(\mathbf{S})$ be the probability distribution for the state after t steps, so

$$\pi_{t+1}(\mathbf{S}') = \sum_{\mathbf{s}} \Pr\left(\mathbf{s} \rightarrow \mathbf{S}'\right) \pi_t(\mathbf{s}).$$

If at some point we obtain $\pi_{t+1}(\mathbf{s}) = \pi_t(\mathbf{s})$ for all $\mathbf{s}$ then we have reached a *stationary distribution* π. In this case

$$\forall \mathbf{s}' \pi(\mathbf{s}') = \sum_{\mathbf{s}} \Pr\left(\mathbf{s} \rightarrow \mathbf{s}'\right) \pi(\mathbf{s}).$$

There is exactly one stationary distribution for a given $\Pr\left(\mathbf{S} \rightarrow \mathbf{S}'\right)$ provided the latter obeys some simple conditions.

The condition of *detailed balance*

$$\forall \mathrm{s}, \mathrm{s}' \pi(\mathrm{s})\Pr(\mathrm{s} \rightarrow \mathrm{s}') = \pi(\mathrm{s}')\Pr(\mathrm{s}' \rightarrow \mathrm{s})$$

is sufficient to provide a π that is a stationary distribution. To see this simply sum:

$$\begin{aligned}\sum_{\mathrm{s}} \pi(\mathrm{s})\Pr(\mathrm{s} \rightarrow \mathrm{s}') &= \sum_{\mathrm{s}} \pi(\mathrm{s}')\Pr(\mathrm{s}' \rightarrow \mathrm{s}) \\ &= \pi(\mathrm{s}')\underbrace{\sum_{\mathrm{s}} \Pr(\mathrm{s}' \rightarrow \mathrm{s})}_{=1} \\ &= \pi(\mathrm{s}')\end{aligned}$$

If all this is looking a little familiar, it's because we now have another excellent application for the material in *Mathematical Methods for Computer Science*.

That course used the alternative term *local balance*.

Approximate inference for Bayesian networks

Recalling once again the basic equation for performing probabilistic inference

$$\Pr(\mathbf{Q}|o_1, o_2, \ldots, o_m) \propto \sum_{\mathbf{L}} \Pr(\mathbf{Q}, \mathbf{L}, o_1, o_2, \ldots, o_m)$$

where

- $\mathbf{Q}$ is the query.
- $o_1, o_2, \ldots, o_m$ are the observations.
- $\mathbf{L}$ are the latent variables.
- $1/Z$ normalises the distribution.
- The query, observations and latent variables are a partition of the set $\mathbf{V} = \{V_1, V_2, \ldots, V_n\}$ of all variables.

We are going to consider obtaining samples from the distribution $\Pr(\mathbf{Q}, \mathbf{L}|o_1, o_2, \ldots, o_m)$.

Approximate inference for Bayesian networks

The observations are fixed. Let the *state* of our system be a specific set of values for *a query variable and the latent variables*

$$\mathbf{S} = (S_1, S_2, \ldots, S_{l+1}) = (Q, L_1, L_2, \ldots, L_l)$$

and define $\overline{\mathbf{S}}_i$ to be the state vector *with S_i removed*

$$\overline{\mathbf{S}}_i = (S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_{n+1}).$$

To move from $\mathbf{s}$ to $\mathbf{s}'$ we replace one of its elements, say s_i, with a new value s_i' sampled according to

$$s_i' \sim \Pr\left(S_i | \overline{\mathbf{s}}_i, o_1, \ldots, o_m\right)$$

This has detailed balance, and has $\Pr\left(Q, \mathbf{L} | o_1, \ldots, o_m\right)$ as its stationary distribution.

It is known as *Gibbs sampling*.

To see that $\Pr\left(Q, \mathbf{L} | \mathbf{o}\right)$ is the stationary distribution we just demonstrate *detailed balance*:

$$
\begin{aligned}
\pi(\mathbf{s})\Pr\left(\mathbf{s} \rightarrow \mathbf{s}'\right) &= \Pr\left(\mathbf{s}|\mathbf{o}\right)\Pr\left(s_i'|\bar{\mathbf{s}}_i, \mathbf{o}\right) \\
&= \Pr\left(s_i, \bar{\mathbf{s}}_i|\mathbf{o}\right)\Pr\left(s_i'|\bar{\mathbf{s}}_i, \mathbf{o}\right) \\
&= \Pr\left(s_i|\bar{\mathbf{s}}_i, \mathbf{o}\right)\Pr\left(\bar{\mathbf{s}}_i|\mathbf{o}\right)\Pr\left(s_i'|\bar{\mathbf{s}}_i, \mathbf{o}\right) \\
&= \Pr\left(s_i|\bar{\mathbf{s}}_i, \mathbf{o}\right)\Pr\left(s_i', \bar{\mathbf{s}}_i|\mathbf{o}\right) \\
&= \Pr\left(\mathbf{s}' \rightarrow \mathbf{s}\right)\pi(\mathbf{s}').
\end{aligned}
$$

As a further simplification we can exploit *conditional independence*.

For example, sampling from $\Pr\left(S_i|\bar{\mathbf{s}}_i, \mathbf{o}\right)$ may be equivalent to sampling S_i conditional on some smaller set.

So:

- We successively sample the query variable and the unobserved variables, conditional on the remaining variables.
- This gives us a sequence $\mathbf{s}_1, \mathbf{s}_2, \ldots$ sampled according to $\Pr(Q, \mathbf{L}|\mathbf{o})$.

Finally, note that as

$$\Pr(Q|\mathbf{o}) = \sum_{\mathbf{l}} \Pr(Q, \mathbf{l}|\mathbf{o})$$

we can just ignore the values obtained for the unobserved variables. This gives us $q_1, q_2, \ldots$ with

$$q_i \sim \Pr(Q|\mathbf{o}).$$

To see that the final step works, consider what happens when we estimate the expected value of some function of Q.

$$
\begin{aligned}
\mathbb{E}[f(Q)|\mathbf{o}] &= \sum_{q} f(q)\mathrm{Pr}\left(q|\mathbf{o}\right) \\
&= \sum_{q} f(q) \sum_{\mathbf{l}} \mathrm{Pr}\left(q, \mathbf{l}|\mathbf{o}\right) \\
&= \sum_{q} \sum_{\mathbf{l}} f(q)\mathrm{Pr}\left(q, \mathbf{l}|\mathbf{o}\right)
\end{aligned}
$$

so sampling using $\mathrm{Pr}\left(q, \mathbf{l}|\mathbf{o}\right)$ and ignoring the values for $\mathbf{l}$ obtained works exactly as required.

Markov random fields

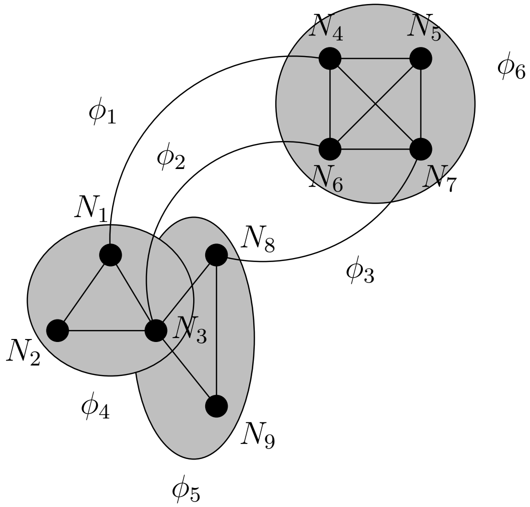
Markov random fields (MRFs) (sometimes called *undirected graphical models* or *Markov networks*) provide an *alternative approach* to representing a *probability distribution* while expressing *conditional independence assumptions*.

We now have:

1. An *undirected graph* $G = (N, E)$.
2. G has *a node* N_i *for each RV*.
3. For each *maximal clique* c in G there is a *clique potential* $\phi_c(N_c) > 0$ where N_c is the set of nodes in c.
4. The probability distribution expressed by G is

$$\Pr(N) \propto \prod_c \phi_c(N_c).$$

Example: 3 maximal cliques of size 2, 2 of size 3 and 1 of size 4.



$$\Pr(N_1, \ldots, N_9) \propto \phi_1(N_1, N_4) \times \phi_2(N_3, N_6) \times \phi_3(N_7, N_8) \times \phi_4(N_1, N_2, N_3)$$
$$\times\, \phi_5(N_3, N_8, N_9) \times \phi_6(N_4, N_5, N_6, N_7).$$

Markov random fields—conditional independence

The *test for conditional independence* is now simple: if X, Y and Z are disjoint subsets of the RVs then:

1. *Remove* the nodes in Z *and any attached edges* from the graph.
2. If there are *no paths from any variable in* X to *any variable in* Y then

$$X \perp Y | Z.$$

Final things to note:

1. MRFs have their *own algorithms for inference*.
2. They are an *alternative* to *BNs* for representing a probability distribution.
3. There are *trade-offs* that might make a BN or MRF *more or less favourable*.
4. For example: *potentials offer flexibility* because *they don't have to represent conditional distributions*...
5. ...BUT you have to *normalize* the distribution you're representing.