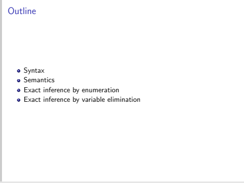


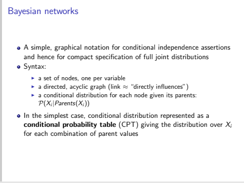
Outline

- Syntax
- Semantics
- Exact inference by enumeration
- Exact inference by variable elimination

We have already seen that logic is not omnipotent, and there are cases where we cannot make decisions that produce a sure result. Last time, we reviewed the basics of probability calculus and the concepts of independence and conditional independence.

This week, we will look at how probabilistic inference works. For this we will use the Bayesian network, which is very widespread and used very frequently by the industry.

We show how the probability of elementary events can be used to compute an inference, and how conditional probabilities can be used to obtain these probabilities. Unfortunately, if there are a lot of elementary events, this requires exponential work. We are introducing a method that significantly simplifies the calculation, although special matrix operations must be used. Here we can omit variables step by step, thus simplifying the task one by one until we reach the final outcome.



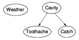
Bayesian networks

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions
- Syntax:
  - a set of nodes, one per variable
  - a directed, acyclic graph (link ≈ "directly influences")
  - a conditional distribution for each node given its parents: $\mathcal{P}(X_i|Parents(X_i))$
- In the simplest case, conditional distribution represented as a **conditional probability table** (CPT) giving the distribution over X_i for each combination of parent values

The Bayesian network is actually a graph, more precisely a directed graph. The vertices of the graph are the the probability variables in the problem. The edges of the graph show which probability variables are related and in what direction (what affects what). Each edge also has probability information, more precisely, the edges leading to each vertex together determine how the value of the probability variable is influenced by the value of the parents - probability variables on the other side of the edges. Thus, probability variables have in a discrete case conditional probability tables.

Example

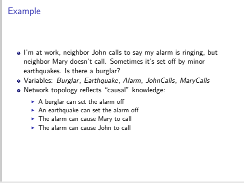
- Topology of network encodes conditional independence assertions:



- *Weather* is independent of the other variables
- *Toothache* and *Catch* are conditionally independent given *Cavity*

Let's see the example we examined earlier. Perhaps you still remember that the probability variable *Weather* is independent of the probability variables associated with the dentist. This is now reflected in the fact that there is no edge between these variables. You may also remember that the probability variables *Toothache* and *Catch* were conditionally independent in case of the probability variable *Cavity*, so the probability variables *Toothache* and *Catch* are not connected by an edge, in either direction. However, we have said that the *Cavity* may be the cause for the *Toothache* and the *Cavity* may be the cause of the *Catch*. In these cases, the appropriate edges are needed and the *Cavity* will be the parent of the probability variables *Toothache* and *Catch*.

Conditional independence can be read from the figure, and even if there is no direct relationship between *Toothache* and *Catch*, they are in a common component of a graph.



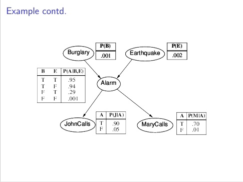
Example

- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?
- Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
- Network topology reflects "causal" knowledge:
  - A burglar can set the alarm off
  - An earthquake can set the alarm off
  - The alarm can cause Mary to call
  - The alarm can cause John to call

Consider the most typical example of the Bayesian networks! Imagine you work in Los Angeles and live somewhere in the suburbs. Your neighbours are John and Mary. They are good neighbours, and if there is something wrong with your house, they will call you. The apartment has an alarm. Of course, this will alert if the apartment is broken into. This is Los Angeles, so earthquakes are frequent here. Of course, the earthquake waves the chandelier, therefore the alarm may sound again. The neighbours, of course, do not constantly look at what is happening at your house, but they pay attention to the alarm signal and notify you.

AI #9

└Example contd.


Example contd.

P(B)
.001

P(E)
.002

B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

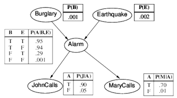
A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

Let us see the above, very general description more specifically, with precise chances. You may have noticed that from the description we can clearly follow where there exist cause-effect relationships.

As a priori information, we know that houses are broken into on average once in 1,000 days, which is a relatively good rate. So its odds are 0.1%. According to the statistics (again a priori), there is a double chance of having an earthquake in the same place, i.e. the odds are 0.2%. These probability variables are independent of everything, they can be treated as the beginning of the graph. Whether the alarm sounds is determined by the values of these two probability variables. If during a break in there is an earthquake, the alarm will sound with 95% probability. If during at break in there is no earthquake, the alarm will sound with a 94% probability. If there is only an earthquake and no burglary, the alarm will occur with a 29% probability. Eventually, there is also the possibility – that there is neither an earthquake nor a break in – but for some reason the alarm will malfunction and sound with a 0.1% probability.

Example contd.



P(B)
.001

P(E)
.002

B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

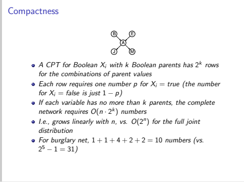
A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

If the alarm sounds, John is 90% likely to hear it and call. John occasionally confuses the sound of the alarm with something else and has a 5% chance of calling even if there is no alarm.

The other neighbour is Mary, who listens to music a lot – loudly – so she often does not hear the alarm. So there is only a 70% chance that she will notice the alarm and call. He hears slightly better than John, so she usually doesn't confuse the sound of the alarm with anything else, so he has a 1% chance of a false alarm.

The tables only list the chances of something happening. Of course, they may not happen, and their probabilities are obtained by subtracting the values in the table from 1.

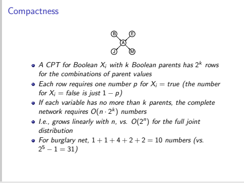


Compactness

$$\begin{tikzcd}
B \arrow[dr] & & E \arrow[dl] \\
& A \arrow[dl] \arrow[dr] & \\
J & & M
\end{tikzcd}$$

- *A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values*
- *Each row requires one number p for $X_i = true$ (the number for $X_i = false$ is just $1 - p$)*
- *If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers*
- *I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution*
- *For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)*

Lets see why are Bayesian networks so frequently used in practical applications!
If we consider that a variable has k parents and we only have logical variables in the given task, then the conditional probability table of the probability variable must include a probability for all possible values of the parents. This means 2^k cases if we have k parents. As pointed out earlier, here we give p, the probability that a given probability variable is satisfied in the given case. Correspondingly, the chance that it will not be satisfied is $1 - p$.

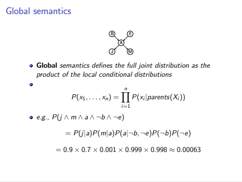


Compactness

$$\begin{tikzcd}
B \arrow[dr] & & E \arrow[dl] \\
& A \arrow[dl] \arrow[dr] & \\
J & & M
\end{tikzcd}$$

- *A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values*
- *Each row requires one number p for $X_i = true$ (the number for $X_i = false$ is just $1 - p$)*
- *If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers*
- *I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution*
- *For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)*

If for all probability variables it holds that they have at most k parents, then the space complexity will be $n \cdot 2^k$, which is linear with respect to n. If we want to calculate the probabilities of all elementary events, it means we have 2^n cases, which is exponential in n (time complexity). In our example, we were able to describe the complete problem with 10 probability values. If the combined probabilities had to be described, it would mean 32 numbers, 31 of which are independent (their sum is 1, and all but one can be freely entered). Of course, for multiple variables or multiple values, the difference will be significantly larger.



Global semantics

$$\begin{tikzcd}
B \arrow[dr] & & E \arrow[dl] \\
& A \arrow[dl] \arrow[dr] & \\
J & & M
\end{tikzcd}$$

- **Global** *semantics defines the full joint distribution as the product of the local conditional distributions*
- $$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i))$$
- *e.g.,* $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

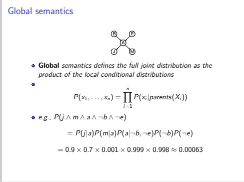
$$= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e)$$

$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 \approx 0.00063$$

Lets simplify the notation a bit, similar to the one used in the past (e.g. *cavity* instead of *Cavity*=*true*), i.e. we dont describe the probability variables, just their values.

Joint probability can be written using the chain rule, and since we can omit variables that are independent of the given variable, only the parents will count.

Consider an elementary event – when we have to give the values of all five probability variables – so that there was no burglary nor an earthquake, but the alarm sounded, and John and Mary both called. The phonecalls from John and Mary depend only on the sound of the alarm, so the product $P(j|r)$ and $P(m|r)$ will be included in the product. The alarm sounded but there was neither an earthquake nor a burglary, so the term $P(r|\neg b, \neg f)$ is included.



Global semantics

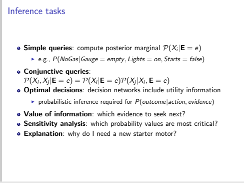
- **Global** *semantics defines the full joint distribution as the product of the local conditional distributions*
- $$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i))$$
- *e.g.,* $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$$= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e)$$

$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 \approx 0.00063$$

On the other hand, since there was no earthquake and no burglary, we have the terms $P(\neg b)$ and $P(\neg f)$. If we multiply these, we get the specific probability of this elementary event.

We see that the probability of this is very small, but not 0. Similarly, the probabilities of the other 31 elementary events can be calculated.

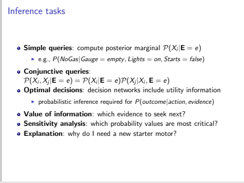


Inference tasks

- **Simple queries**: compute posterior marginal $\mathcal{P}(X_i|\mathbf{E}=e)$
  - e.g., $P(NoGas|Gauge=empty, Lights=on, Starts=false)$
- **Conjunctive queries**: $\mathcal{P}(X_i, X_j|\mathbf{E}=e) = \mathcal{P}(X_i|\mathbf{E}=e)\mathcal{P}(X_j|X_i, \mathbf{E}=e)$
- **Optimal decisions**: decision networks include utility information
  - probabilistic inference required for $P(outcome|action, evidence)$
- **Value of information**: which evidence to seek next?
- **Sensitivity analysis**: which probability values are most critical?
- **Explanation**: why do I need a new starter motor?

Let us also look at how the probability of inference works in reality! We are given the observed events (which are related to specific probability variables) and we should give the distribution of a given probability variable in this case, i.e. the probability of it taking each of its possible values.
If we are sitting in a car and we see that the pointer indicates that the tank is empty; the lights are on, so there is electricity, but the car still does not start, then what is the chance that there is no gasoline? (Or maybe the starter is broken?)

In a more complex case, we are interested in the joint distribution of not one, but several probability variables, but we can trace that back to this to simpler tasks, i.e. the distribution of one probability variable.

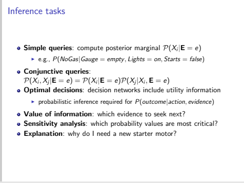


Inference tasks

- **Simple queries**: compute posterior marginal $\mathcal{P}(X_i|\mathbf{E} = e)$
  - e.g., $P(NoGas|Gauge = empty, Lights = on, Starts = false)$
- **Conjunctive queries**: $\mathcal{P}(X_i, X_j|\mathbf{E} = e) = \mathcal{P}(X_i|\mathbf{E} = e)\mathcal{P}(X_j|X_i, \mathbf{E} = e)$
- **Optimal decisions**: decision networks include utility information
  - probabilistic inference required for $P(outcome|action, evidence)$
- **Value of information**: which evidence to seek next?
- **Sensitivity analysis**: which probability values are most critical?
- **Explanation**: why do I need a new starter motor?

We can assign utility values to each event, and then the Bayesian network can be thought of as a decision tree. This can help us make more complex decisions and see the expected values for each decision. These decisions can be parallel decisions or sequential decisions, where we can get extra information from each step. (Send the second-hand car you want to buy to a service center for inspection, or buy it without a survey? Or have it checked by a familiar car mechanic first? All actions have a price!)

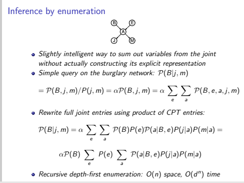
What should we check for when looking at a second-hand car? For example, there are model-specific defects that are quite expensive to fix. For a given type of car, its probably worth addressing this, instead of only considering how worn is the steering wheel or whether the odometer has been rewound.



Inference tasks

- **Simple queries**: compute posterior marginal $\mathcal{P}(X_i|\mathbf{E}=e)$
  - e.g., $P(NoGas|Gauge=empty, Lights=on, Starts=false)$
- **Conjunctive queries**: $\mathcal{P}(X_i, X_j|\mathbf{E}=e) = \mathcal{P}(X_i|\mathbf{E}=e)\mathcal{P}(X_j|X_i, \mathbf{E}=e)$
- **Optimal decisions**: decision networks include utility information
  - probabilistic inference required for $P(outcome|action, evidence)$
- **Value of information**: which evidence to seek next?
- **Sensitivity analysis**: which probability values are most critical?
- **Explanation**: why do I need a new starter motor?

With a similar approach, we can narrow down the weakest links, i.e., what failures are most likely to happen; what should we prepare for and potentially procure spare parts in advance, also taking into account how easy and fast it is to find each product. I hope that this example shows that it is worth learn this subject with those involved in control and operation, because you can get rid of a lot of inconveniences.



Inference by enumeration

$$\begin{tikzcd} B \arrow[dr] & & E \arrow[dl] \\ & A \arrow[dl] \arrow[dr] & \\ J & & M \end{tikzcd}$$

- *Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation*
- *Simple query on the burglary network:* $\mathcal{P}(B|j,m)$

$$= \mathcal{P}(B,j,m)/P(j,m) = \alpha\mathcal{P}(B,j,m) = \alpha \sum_e \sum_a \mathcal{P}(B,e,a,j,m)$$

- *Rewrite full joint entries using product of CPT entries:*

$$\mathcal{P}(B|j,m) = \alpha \sum_e \sum_a \mathcal{P}(B)P(e)\mathcal{P}(a|B,e)P(j|a)P(m|a) =$$

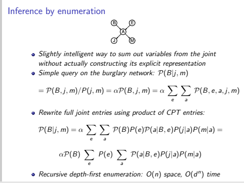
$$\alpha\mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e)P(j|a)P(m|a)$$

- *Recursive depth-first enumeration:* $O(n)$ *space,* $O(d^n)$ *time*

Lets go back to our burglar example. You are at work and all you know is that both Mary and John called you because of the alarm. What is the chance of a break in?

We need to calculate a conditional probability. More specifically, a conditional probability distribution because we want to calculate this conditional probability for both values of *Burglary*.

The conditional probability of the product rule is equal to the quotient of the combined probability and the probability of the condition. Denoting the reciprocal of the denominator by a constant alpha, we only need to deal with one joint distribution. We now get this by listing all possible cases and summarizing their probabilities. This is why all possible values of the hidden variables (*Earthquake* and *Alarm*) are included in the sum. Naturally, each joint probability can be decomposed into a product of conditional probabilities using the chain rule, and the sum-independent/constant terms can be moved out front of the sum to facilitate the calculation.



Inference by enumeration

$$\begin{tikzcd} B \arrow[dr] & & E \arrow[dl] \\ & A \arrow[dl] \arrow[dr] & \\ J & & M \end{tikzcd}$$

- *Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation*
- *Simple query on the burglary network:* $\mathcal{P}(B|j,m)$

$$= \mathcal{P}(B,j,m)/P(j,m) = \alpha\mathcal{P}(B,j,m) = \alpha \sum_e \sum_a \mathcal{P}(B,e,a,j,m)$$

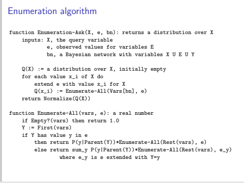
- *Rewrite full joint entries using product of CPT entries:*

$$\mathcal{P}(B|j,m) = \alpha \sum_e \sum_a \mathcal{P}(B)P(e)\mathcal{P}(a|B,e)P(j|a)P(m|a) =$$

$$\alpha\mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e)P(j|a)P(m|a)$$

- *Recursive depth-first enumeration:* $O(n)$ *space,* $O(d^n)$ *time*

However, all elementary events that correspond to evidence (John and Mary call) should be considered. If we go through the possibilities like depth search, we only need to store some conditional probabilities (linear storage complexity), but all combinations have to be taken into account (exponential time complexity).

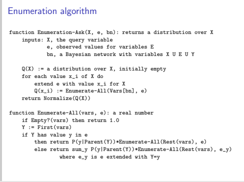

Enumeration algorithm

```
function Enumeration-Ask(X, e, bn): returns a distribution over X
    inputs: X, the query variable
            e, observed values for variables E
            bn, a Bayesian network with variables X U E U Y

    Q(X) := a distribution over X, initially empty
    for each value x_i of X do
        extend e with value x_i for X
        Q(x_i) := Enumerate-All(Vars[bn], e)
    return Normalize(Q(X))

function Enumerate-All(vars, e): a real number
    if Empty?(vars) then return 1.0
    Y := First(vars)
    if Y has value y in e
        then return P(y|Parent(Y))*Enumerate-All(Rest(vars), e)
        else return sum_y P(y|Parent(Y))*Enumerate-All(Rest(vars), e_y)
                where e_y is e extended with Y=y
```

Let's see the algorithm of this solution method. We need the probability variable whose distribution should be returned, as well as the evidence. In addition to this data, we also use the Bayesian network.

All we have to do is take all the values of the probability variable in question and add them to the evidence one by one. We then look at the probability values for each combination of the missing hidden variables and normalize them so that their sum yields one.

The routine Enumerate-All initially receives all the variables as well as the evidence. If the variables run out due to the recursive calls, 1 is returned so as not to affect the final result. If there are still unprocessed variables, consider the one closest to the beginning of the Bayesian graph (take a topological sort). In other words, its parents (if any) are already known.

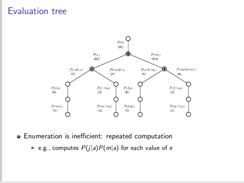


Enumeration algorithm

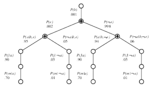
```
function Enumeration-Ask(X, e, bn): returns a distribution over X
    inputs: X, the query variable
            e, observed values for variables E
            bn, a Bayesian network with variables X U E U Y

    Q(X) := a distribution over X, initially empty
    for each value x_i of X do
        extend e with value x_i for X
        Q(x_i) := Enumerate-All(Vars[bn], e)
    return Normalize(Q(X))

function Enumerate-All(vars, e): a real number
    if Empty?(vars) then return 1.0
    Y := First(vars)
    if Y has value y in e
        then return P(y|Parent(Y))*Enumerate-All(Rest(vars), e)
        else return sum_y P(y|Parent(Y))*Enumerate-All(Rest(vars), e_y)
                where e_y is e extended with Y=y
```

We have two options.

1. This variable also belongs to the evidence, so its value is known. Then the condition belonging to the other variables must be multiplied by the conditional probability for it (recursive call).
2. This variable is not yet part of the evidence, so we need to work with all its possible values. Therefore, we multiply the function values of the other variables and the evidence extended by them by the conditional probabilities associated with each of their values, and then sum them up.

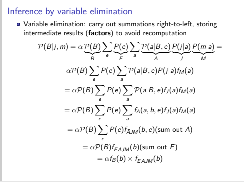


Evaluation tree



- Enumeration is inefficient: repeated computation
  - e.g., computes $P(j|a)P(m|a)$ for each value of e

The calculation for a specific problem can also be thought of as a tree. Here, the conditional probabilities in the problem belong to the edges. For vertices marked with empty circles, these values must be multiplied, whilst for vertices marked with a cross, we must sum the products, i.e. the partial results.

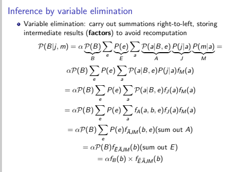


Inference by variable elimination

- Variable elimination: carry out summations right-to-left, storing intermediate results (**factors**) to avoid recomputation

$$\mathcal{P}(B|j,m) = \alpha \underbrace{\mathcal{P}(B)}_{B} \sum_e \underbrace{P(e)}_{E} \sum_a \underbrace{\mathcal{P}(a|B,e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M} =$$
$$\alpha \mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e) P(j|a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e) f_J(a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) \sum_a f_A(a,b,e) f_J(a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) f_{\bar{A}JM}(b,e) (\text{sum out } A)$$
$$= \alpha \mathcal{P}(B) f_{\bar{E}\bar{A}JM}(b) (\text{sum out } E)$$
$$= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)$$

If we have many variables and they have multiple values, then the method described earlier is accurate but very slow. So let us look at another approach that achieves the same result with far fewer calculations, albeit is a little more complicated. The starting point is the same conditional distribution function. Its value is given by the product and sum of conditional probabilities as seen above. However, instead of starting to count them separately, count them all at once! At the end of the formula is the term $P(m|r)$, which is not constant, because r can indicate that the alarm sounds, but also that it does not. So this term can be considered a function of r, which is why the $f_M(r)$ notation is used, but it can also be considered as a two-element vector instead of a function. Similarly, $P(j|r)$ can be considered as a function $f_J(r)$ or a two-element vector (with two conditional probabilities in it). For $P(r|B,f)$, $\mathcal{P}$ is calligraphic because we have to work with both values of B.

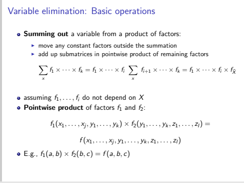


Inference by variable elimination

- Variable elimination: carry out summations right-to-left, storing intermediate results (**factors**) to avoid recomputation

$$\mathcal{P}(B|j,m) = \alpha \underbrace{\mathcal{P}(B)}_{B} \sum_e \underbrace{P(e)}_{E} \sum_a \underbrace{\mathcal{P}(a|B,e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M} =$$
$$\alpha \mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e) P(j|a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) \sum_a \mathcal{P}(a|B,e) f_J(a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) \sum_a f_A(a,b,e) f_J(a) f_M(a)$$
$$= \alpha \mathcal{P}(B) \sum_e P(e) f_{\bar{A}JM}(b,e) (\text{sum out } A)$$
$$= \alpha \mathcal{P}(B) f_{\bar{E}\bar{A}JM}(b) (\text{sum out } E)$$
$$= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)$$

But r and f also mean two possibilities, since they both belong to a sum. That is, $P(r|B,f)$ contains 8 values as a $2 \times 2 \times 2$ three-dimensional matrix. Next is the sum according to r, so we have to take the parts of the corresponding matrices where r is satisfied and perform the multiplication between the 2×2 matrix and two constants; then take the parts where r is not satisfied and perform a similar multiplication. The two 2×2 matrices obtained by these multiplications need to be added together. This is called point multiplication. The same needs to be done for the two values of f, and in the end there is only one ambiguous vector left, which after normalization gives the previously calculated values.



Variable elimination: Basic operations

- **Summing out** a variable from a product of factors:
  - move any constant factors outside the summation
  - add up submatrices in pointwise product of remaining factors

$$\sum_x f_1 \times \cdots \times f_k = f_1 \times \cdots \times f_i \sum_x f_{i+1} \times \cdots \times f_k = f_1 \times \cdots \times f_i \times f_{\bar{X}}$$

- assuming $f_1, \ldots, f_i$ do not depend on X
- **Pointwise product** of factors f_1 and f_2:

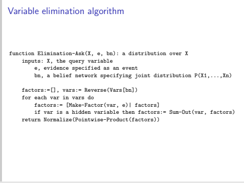
$$f_1(x_1, \ldots, x_j, y_1, \ldots, y_k) \times f_2(y_1, \ldots, y_k, z_1, \ldots, z_l) =$$

$$f(x_1, \ldots, x_j, y_1, \ldots, y_k, z_1, \ldots, z_l)$$

- E.g., $f_1(a, b) \times f_2(b, c) = f(a, b, c)$

As we said before summing out r, we need to multiply the results for different values of r, and sum them.

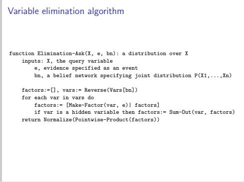
Since we deal with very simple tasks in the exams, it is sufficient to calculate the probabilities associated with the elementary event. Moreover I am not interested in the specific values, but in the process of the calculation.



Variable elimination algorithm

```
function Elimination-Ask(X, e, bn): a distribution over X
    inputs: X, the query variable
        e, evidence specified as an event
        bn, a belief network specifying joint distribution P(X1,...,Xn)

    factors:=[], vars:= Reverse(Vars[bn])
    for each var in vars do
        factors:= [Make-Factor(var, e)| factors]
        if var is a hidden variable then factors:= Sum-Out(var, factors)
    return Normalize(Pointwise-Product(factors))
```

Let us see the algorithm! Whilst at first we worked from the beginning of the graph, here we explain our probability variables backwards. We first create the matrices for the given variables, and if it is a hidden variable, we perform the point-by-point multiplication accordingly. Finally, if we have processed all the variables, we normalize the values of the obtained vector.

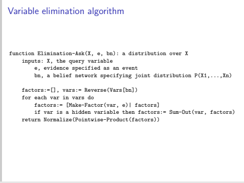


Variable elimination algorithm

```
function Elimination-Ask(X, e, bn): a distribution over X
    inputs: X, the query variable
        e, evidence specified as an event
        bn, a belief network specifying joint distribution P(X1,...,Xn)

    factors:=[], vars:= Reverse(Vars[bn])
    for each var in vars do
        factors:= [Make-Factor(var, e)| factors]
        if var is a hidden variable then factors:= Sum-Out(var, factors)
    return Normalize(Pointwise-Product(factors))
```

Outlook. It is nice to have an accurate definition of each probability, but very often it is not necessary. If there are many probability variables, each variable has not only two but several values if the number of parents is high, which significantly complicates the calculations. Therefore, in such cases approximation by random sampling is widely used. For example, we start in the topological order of the variables and generate the value of the next probability variable according to the values of the parents and the matching conditional probability.
If we have generated a larger sample using this method, we can see how many cases satisfy our conditions, thus obtaining a relative frequency that converges towards the theoretical probability.

We can choose to only retain those cases from the random sample that correspond to the given evidence. However, if the evidence is at the back of the topological order, we have to discard a lot of cases.


Variable elimination algorithm

```
function Elimination-Ask(X, e, bn): a distribution over X
    inputs: X, the query variable
        e, evidence specified as an event
        bn, a belief network specifying joint distribution P(X1,...,Xn)

    factors:=[], vars:= Reverse(Vars[bn])
    for each var in vars do
        factors:= [Make-Factor(var, e)| factors]
        if var is a hidden variable then factors:= Sum-Out(var, factors)
    return Normalize(Pointwise-Product(factors))
```

Outlook cont. However, we can also start from the evidence and generate the values of the other variables based on them (probability weighting). In this case, just like in the previous, the individual samples are created from scratch.

The MCMC algorithm wanders randomly in the Bayesian network, changing the value of a probability variable in each case, taking into account the additional variables associated with that variable (Markov cover). Thus, for each sample, only the value of one variable needs to be recalculated. The decision is made similarly to the above from the relative frequency.