**Fundamentals of AI and KR- Module 3**

**Introduction to uncertainty and probabilistic reasoning**

**Basic probability notation**

**1. Logical Assertions**

A **logical assertion** specifies what is **definitely true or false**. In the context of possible worlds, it rules out certain worlds. Essentially, it tells us which worlds are impossible, based on strict logical rules.

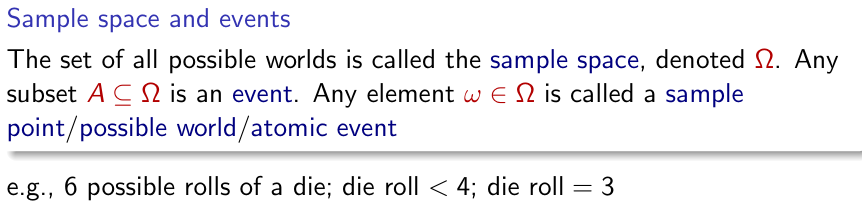
* For example, if we assert that "it is raining," this means that in any world we consider, it must be raining. Any world where it is **not raining** is ruled out.

**2. Probabilistic Assertions**

A **probabilistic assertion**, on the other hand, does not rule out worlds but instead tells us the **likelihood** of various worlds happening.

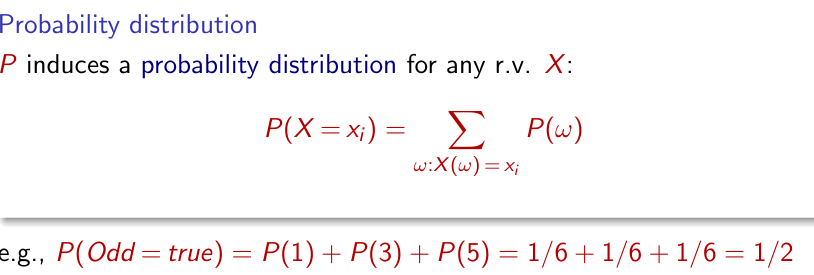
Example of probabilistic assertion:

* Assertion: "There is a 70% chance it will rain today."
* This probabilistic assertion means that, while there are possible worlds where it doesn't rain, the world where it does rain is more likely than one where it doesn't.



**Random Variables:**

A random variable is a function from sample points to some range.



**Prior Probability**

**Prior probability** refers to the **probability of a proposition** before **any new evidence** is considered. It represents our belief about the likelihood of an event or hypothesis based on **prior knowledge**

Example:

P(Weather=sunny) = 0.72  
This means that, based on prior knowledge you believe there is a 72% chance that the weather will be sunny.

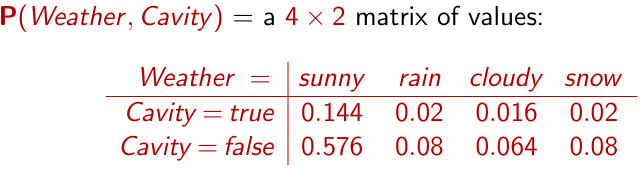
**Probability Distribution**:

A set of probabilities assigned to all possible outcomes of a random variable, with the condition that the sum of probabilities equals 1 (e.g., P(Weather)=⟨0.72,0.1,0.08,0.1⟩

**Joint Probability Distribution**

The **joint probability distribution** describes the likelihood of two or more random variables taking specific values simultaneously. It gives the probability for every possible combination of values of the random variables.

**Example**: Let's consider two random variables: **Weather** and **Cavity**. The joint probability distribution provides the probability of different combinations of values of these variables (such as the probability that the weather is sunny **and** the person has a cavity).

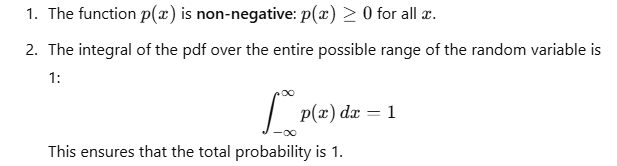


**Probability for Continuous Variables**

For continuous variables, the probability distribution is described differently, using a **probability density function** (pdf) instead of discrete probabilities.

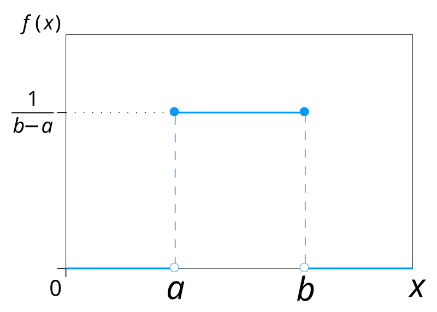
**Probability Density Function (pdf)**

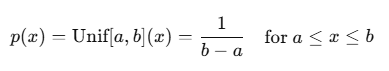
A **probability density function** (pdf) is a function that gives the likelihood of a random variable taking a particular value. The key properties of a pdf are:

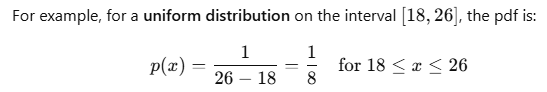


**Uniform Distribution**

The **uniform distribution** is a simple type of pdf where the probability is evenly distributed across a range. The **uniform distribution** has equal probability over a specific range.

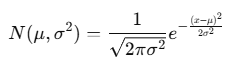




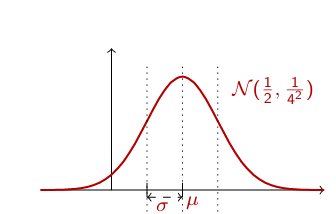


**Gaussian (Normal) Distribution**

The **Gaussian (Normal) distribution** is another common pdf, which is used for continuous variables with a bell-shaped curve.. The pdf for a normal distribution is given by:



here:

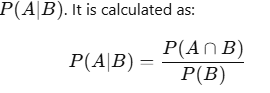
* μ is the **mean** (average) of the distribution.
* σ^2 is the **variance** (the square of the standard deviation σ).
* x is the variable being modeled.

The s**tandard Gaussian distribution**, is a gaussian distribution where μ=0and σ=1

**Conditional probability**

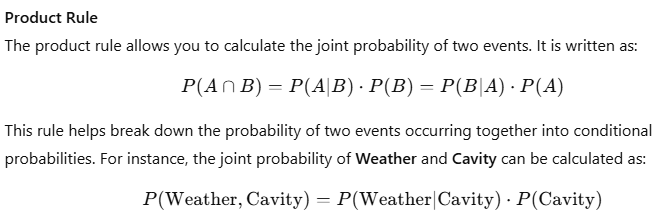
Conditional probability refers to the probability of an event A occurring given that some other event B has already occurred.

P (X | Evidence)

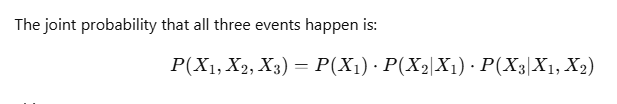


This concept allows us to update our beliefs about an event based on new evidence. For example, if a person has a toothache, the conditional probability of them having a cavity could be P(Cavity∣ Toothache)=0.8

This means that given the evidence of a toothache, there is an 80% chance the person has a cavity.

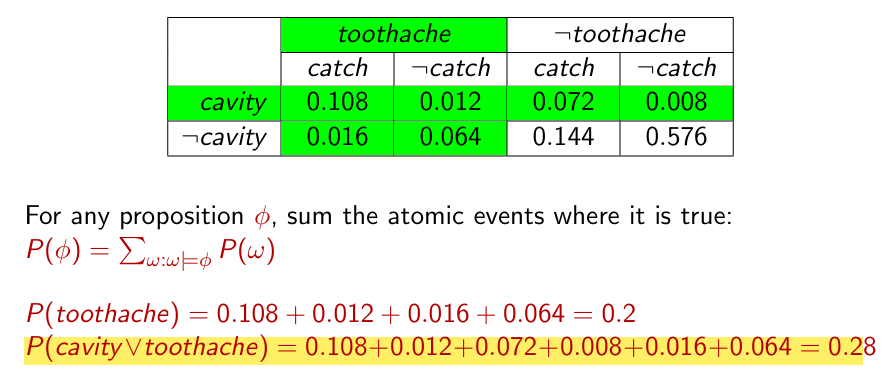


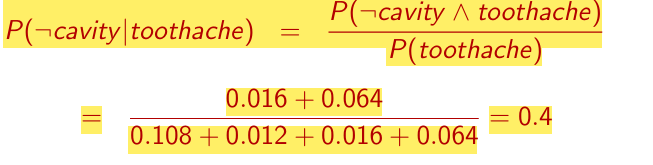
The **chain rule** extends the product rule to sequences of events, allowing for the computation of joint distributions over multiple random variables.

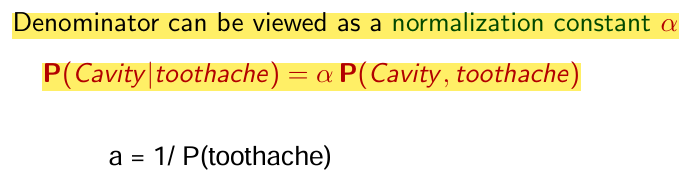


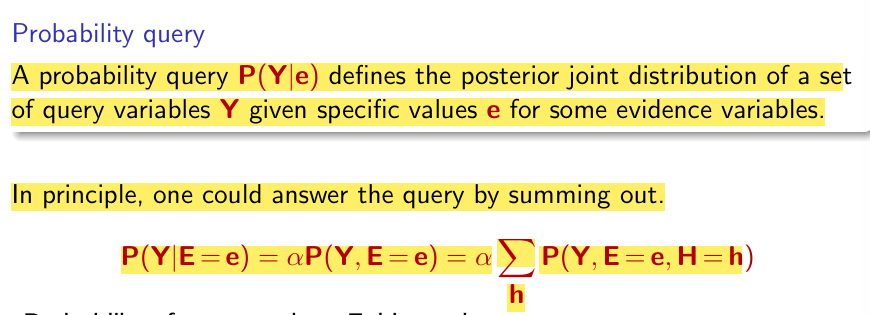
**Inference using Full Joint Distributions**

Inference using full joint distributions allows us to compute various probabilities based on a joint probability table.



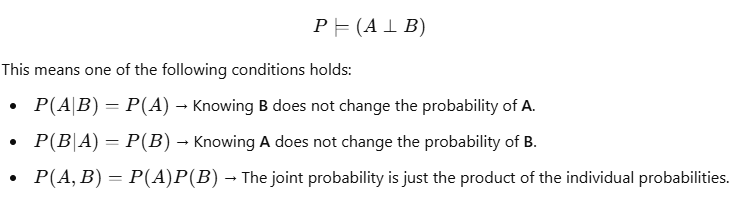






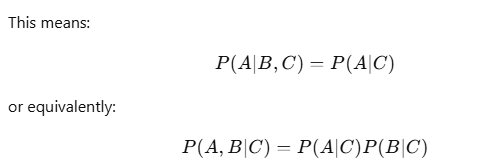
**INDEPENDENCE**

Two events **A** and **B** are **independent** if knowing one does not affect the probability of the other. This is written as:



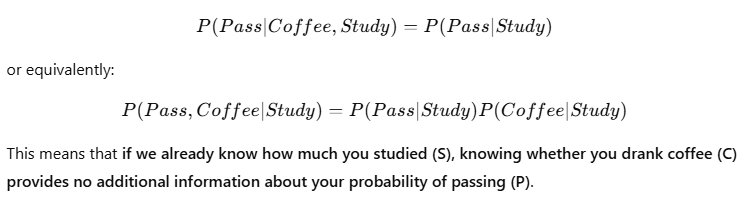
**Conditional Indipendence**

Conditional independence means that two events **A** and **B** are independent **only when given a third event C**.



Example:

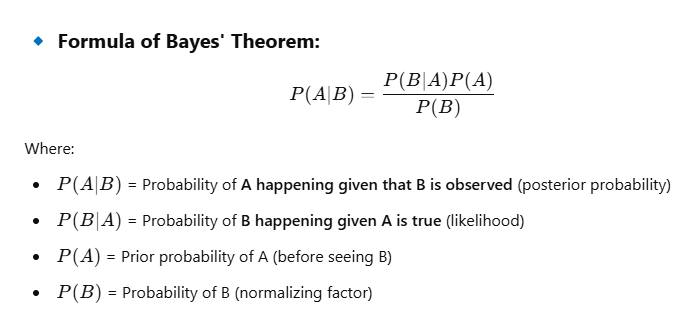
Imagine you are a **university student** taking a difficult exam. You study hard, and whether you **pass (P)** or **fail (F)** depends on how well you studied (**S**). Now, let's introduce another event: **Drinking coffee (C)** before the exam. Once we know how much you studied (S), coffee (C) does not directly affect whether you pass (P).

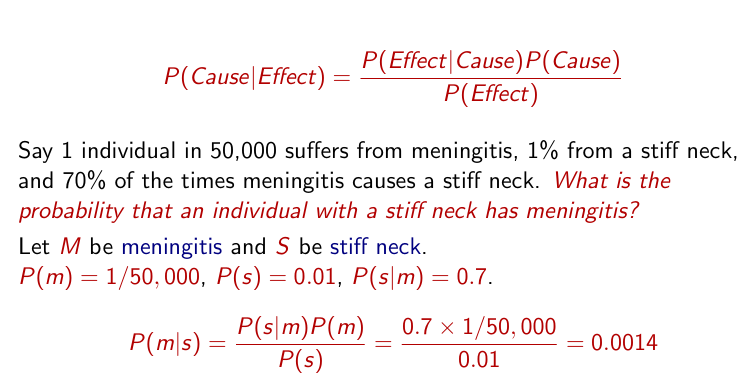
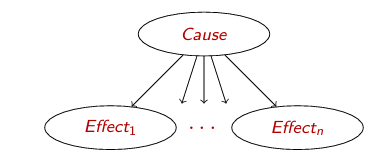


Conditional independence is useful in simplifying probability calculations (Reduces complexity in Bayesian networks). Conditional independence is our most basic and robust form of knowledge about uncertain environments.

Bayes' Rule

Bayes' Rule is a fundamental concept in probability that allows us to **update our beliefs** based on new evidence.

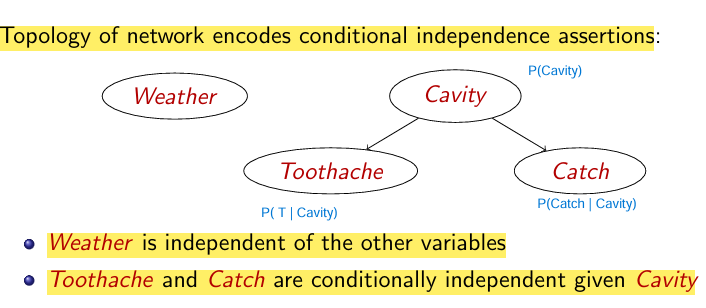




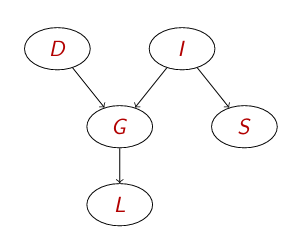
**Bayesian networks**

Graphical notation for conditional independence assertions and for compact specification of full joint distributions.

* Set of nodes, one per variable
* A directed, acyclic graph (link means directly infuences)
* A conditional distribution for each node given its parents: P (Xi | Parents (Xi))



Example independence in this



**P ⊨ (L ⊥ ...)**

* L only depends on G, so, L is **conditionally independent** of everything else given G.
* P ⊨ (L ⊥ D, I, S ∣ G)

**P ⊨ (S⊥ ...)**

* S is directly dependent on I.
* **But S is independent of D and G** because there is no direct path connecting them.
* P⊨ (S⊥ D, G, L∣ I)

**P ⊨ (G⊥ ...)**

* G depends on both D and I.
* However, it is independent of S and L when D and I are known.
* P ⊨ (G ⊥ S, L ∣ D, I)

**P ⊨ (I ⊥ ...)**

* I influences both G and S, but has no direct dependence on D.
* P ⊨ (I ⊥ D) (since there is no direct edge from D to I

**P⊨ (D⊥...?)**

* D only affects G, so it is independent of I and S.
* P⊨ (D⊥ I, S, L )

**ACTIVE TRAIL**

A trail between nodes is a sequence of edges connecting them. A trail is active if probabilistic influence can flow through it.

**When is a Trail Active?**

A **trail** is a sequence of connected edges between nodes in a **Bayesian Network**. When we **observe** a variable, it means we have **its value** based on data or measurements.

**🔹 Short Two-Edge Trails**

Consider these basic two-edge structures:

1️ **Causal Trail:**

X→Z→Y

* **Active unless Z is observed.**
* If Z is observed, the influence is blocked.

2️ **Evidential Trail:**

X←Z←Y

**Active unless Z is observed.**

* If Z is observed, the influence is blocked.

3️ **Common Effect (Collider) Trail:**

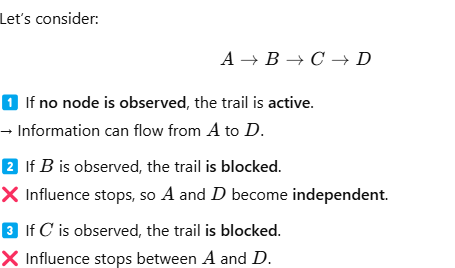
X→Z←Y

**Blocked by default.**

* **Active only if Z or any of its descendants is observed.**
* If Z is **not** observed, the influence does **not** flow.

Now, let’s look at a longer trail:

X1 ↔ X2 ↔ X3 ↔⋯↔ Xn

For **influence to flow from X1…Xn​**, it must pass through **every single node in between**.  
This is **only possible if**:

🔹 **Each two-edge trail (subpath) is active.**  
🔹 **No blocking nodes exist along the trail.**

**Definition of D-Separation**

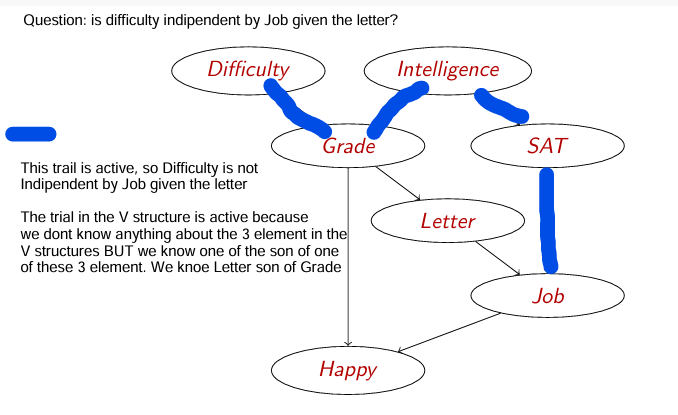
Two sets of nodes X and Y are **d-separated given Z** if **no active trail exists** between any node in X and any node in Y when considering the observed set Z.

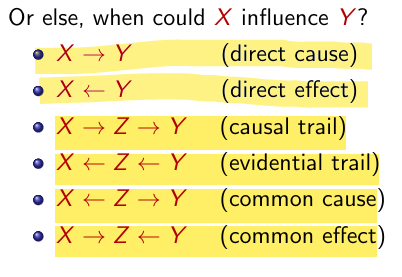
**🔹 How to Check D-Separation?**

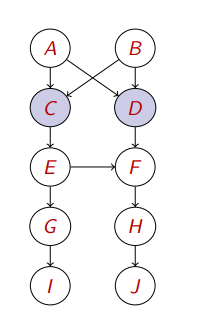
1. **Mark all nodes in Z** and their **descendants**.
2. **Try to traverse the graph** from X to Y, stopping when you reach a **blocked node**.
3. If **no path exists**, then X and Y are **conditionally independent given Z**.

**🔹 When is a Node Blocked?**

* A path between **C and D** is **blocked** if: 1️⃣ There is a **collider** (→ ● ←), and the collider is **not observed**.
* There is a **chain** (→ →) or **common cause** (← →), and an **intermediate node is observed**.

Example



**Example**

1) P |= (C⊥D)?

To check if C is independent of D, we analyze the paths between them.

* C and D are connected because they share parents A and B.
* If no variables are observed, information can flow between them.
* If we observe certain variables, some paths might get blocked.

Are C and D independent without observing any variables?

* No, a path exist between them so they are dependent!

C and D have common causes (A and B). If we don’t observe A and B, knowing D could give us information about C (because they are linked through A and B).

2) P |= (C⊥D|A)?

Are C and D independent given A?

* If we observe A, it block the path beetween C – A -- D
* B is still unobserved, so D can still "send" information to C through B.
* Therefore, a path exist so C and D are still dependent.

3) P|= (C⊥D|A, B)

* If both A and B are observed, the trails are blocked then C and D cannot influence each other anymore.
* Therefore a path in this case does not exist so they are independent given A and B

4) P |= (C⊥D|A, B, J)

* Since J it is not on the path and A and B are already blocking all paths, adding J does not change anything.
* C and D remain independent.

5) P|= (C⊥D|A, B, E, J)

* If we observe E, does that "unblock" the path between C and D?
* No! Because A and B already blocked the path.
* So, C and D remain independent

**Markov blanket**

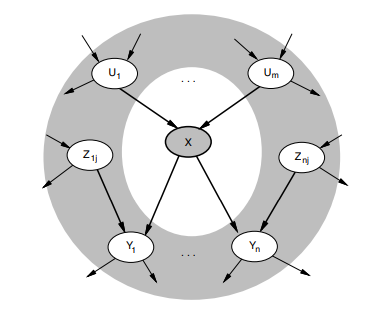
A Markov Blanket of a node X is the set of nodes that make X conditionally independent from all other nodes in the network.

**What does it include?**

For a node X, the Markov Blanket consists of:

1️ **Parents of X** (direct causes of X)  
2️ **Children of X** (direct effects of X)  
3️ **Other parents of X's children** (co-parents of X's children)

💡 **Key Insight**:  
Once you **know the values of the Markov Blanket**, you **don’t need any other nodes** in the network to predict X.



Thus, given X's Markov Blanket (**parents, children, children's parents**), X is **conditionally independent** of all other nodes.

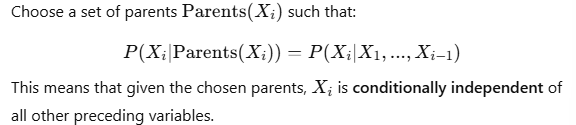
If we want to estimate X, we only need to look at its **Markov Blanket**, not the whole network.

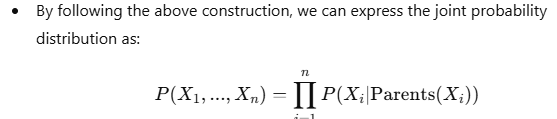
**Constructing Bayesian Networks**

The goal of constructing Bayesian Network (BN) is to ensure that local conditional independence relationships lead to the corret global joint distribution.

Step to Construct a Bayesian Network:

* Arrange the variables X1, X2 ... Xn in some order.

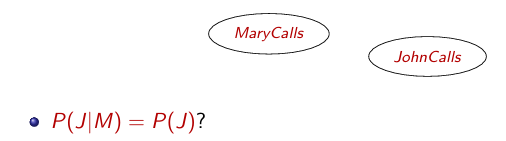
For each variable Xi​, determine its parents from the preceding variables X1…Xi−1​.

Once we know the values of Parents (Xi) the rest of the previous variables X1 … Xi−1 become for predicting Xi

The probability of the entire set of variables can be computed by multiplying the probabilities of each individual variable given its parents.

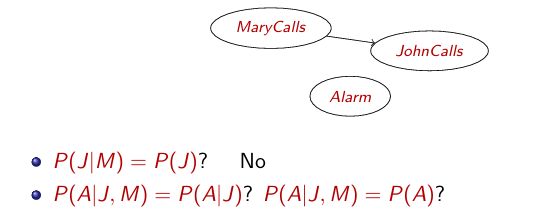
**Example**

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?



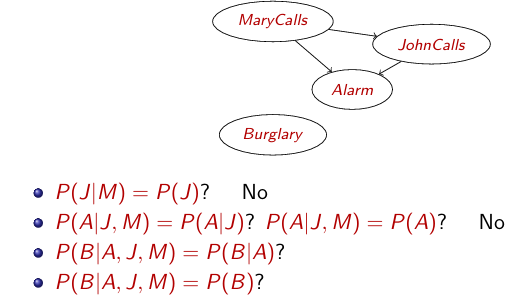
This asks: "Is John’s call independent of Mary’s call?"

No, because both John and Mary’s calls depend on the alarm. If the alarm goes off, both are more likely to call. Thus we connect these 2 variables



This asks: "Does knowing both John and Mary called make Alarm independent?"

**No**, because both John and Mary calling are effects of **Alarm**. Even if we know **J** and **M**, we still need the **network structure** to confirm if **A happened**.



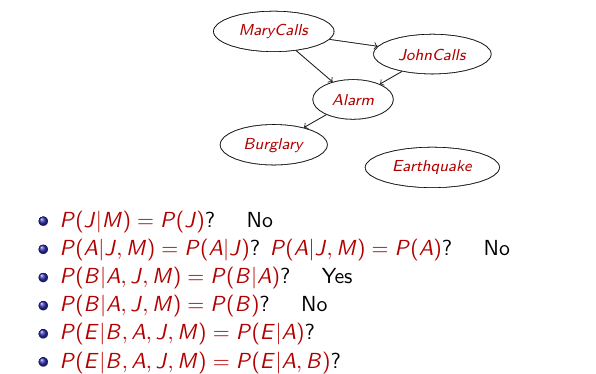
This asks: "Does knowing John and Mary called provide extra information about Burglary beyond Alarm?"

No they do not provide any extra informations, so they are independent given the alarm , so the first response is Yes

The other one asks: "Does knowing John and Mary called provide extra information about Burglary beyond Alarm?"

"Does knowing Alarm (A), JohnCalls (J), and MaryCalls (M) make Burglary (B) independent?"

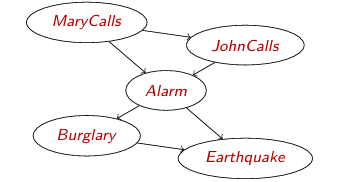
Burglary (B) or Earthquake (E) can trigger the Alarm. his means that A depends on both B and E.



This asks: "Does knowing Burglary, Alarm, JohnCalls, and MaryCalls make Earthquake independent?"

No, because John and Mary’s calls do not influence earthquakes.

But once we know both the alarm and burglary, we can better determine whether an earthquake happened.



If 2 nodes A and B are conditionally independent given C, meaning no direct edge between A and B.

**CASUAL NETWORKS**

Causal Networks are a subset of Bayesian Networks that explicitly model causal relationships rather than just statistical dependencies.

Key Idea:

* Any variable ordering can be used to construct a Bayesian Network, but for causal networks, the order must follow cause-and-effect relationships.

Consider the following example:



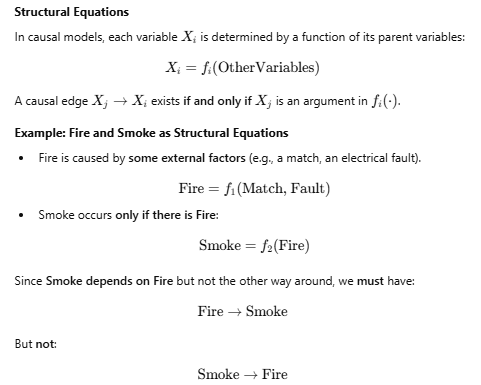
While **both networks can represent the same probabilistic dependence** (i.e., if we observe smoke, it increases the probability of fire), they are **not equivalent** in terms of causality.

**re these networks equivalent?**

**No, because causality is asymmetrical.**

* Fire causes Smoke (**Fire → Smoke**) makes sense.
* Smoke does not cause Fire (**Smoke → Fire**) does **not** make sense causally.

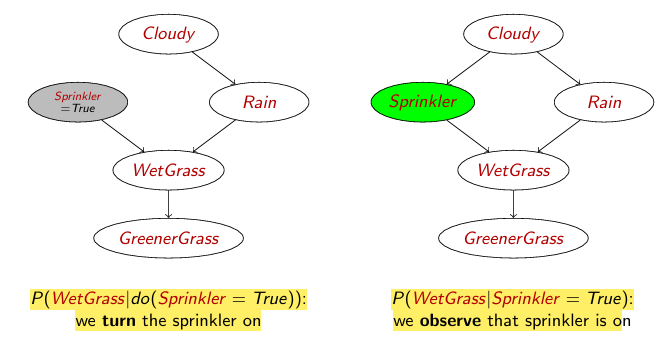
Even if both models can explain observed data, **only (a) respects the true causal relationship.**

To create these relations, we can respond to the question, which responds to which?

**Do-Operator in Causal Networks**

The **do-operator** is a crucial concept in causal inference. It helps differentiate between **observing** a variable and **intervening** on it.

Bayesian Networks describe how variables influence each other **under normal conditions**. However, when we **intervene** (e.g., force a variable to take a value), we change the causal relationships.



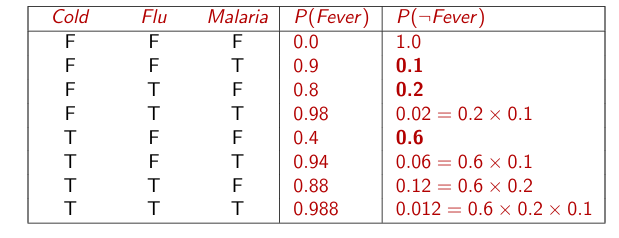
On the right we simply observe that the sprinkler is on and check if the grass is wet. Rain might also contribute to wet grass, so we must account for both the sprinkler and rain.

On the left with intervention we force the Sprinkler to be on, ignoring its natural causes (e.g., whether it's cloudy). This **removes the effect of "Cloudy" on "Sprinkler"** because we are directly setting it. Wet Grass now depends **only** on Sprinkler and Rain (not on Cloudy).

Conditional Probability Tables (CPTs) grow exponentially with the number of parent variables. The solution is to use deterministic nodes (fully determined by its parents) and canonical distributions (like Noisy-OR).

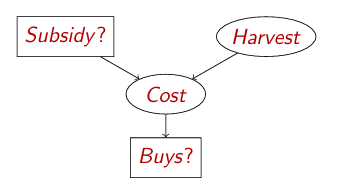
Noisy-OR Model

Used when multiple independent causes contribute to an effect (e.g., **Cold, Flu, Malaria → Fever**). In this case if all the possible causes fail, the effect does not occur. On the other hand, if at least one cause succeeds, the effect occurs with high probability.

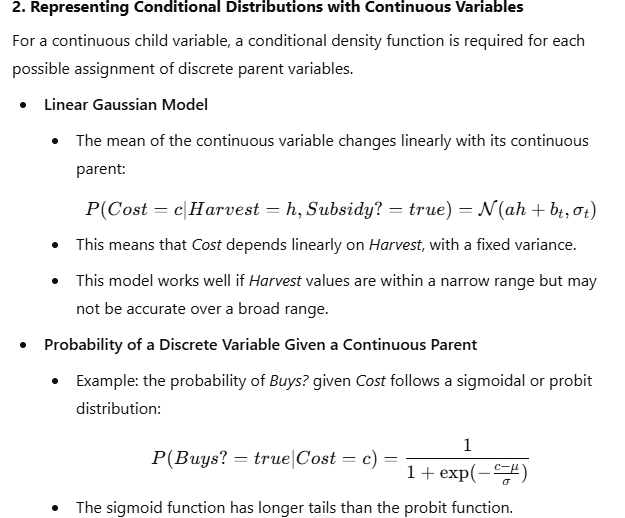


Hybrid Networks

Hybrid networks combine **discrete** variables (e.g., *Subsidy?*, *Buys?, they have a discrete range of values*) and **continuous** variables (e.g., *Harvest*, *Cost, continuous range of value*).



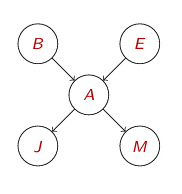
For a continuous child variable, a conditional density function is required for each possible assignment of discrete parent variables.

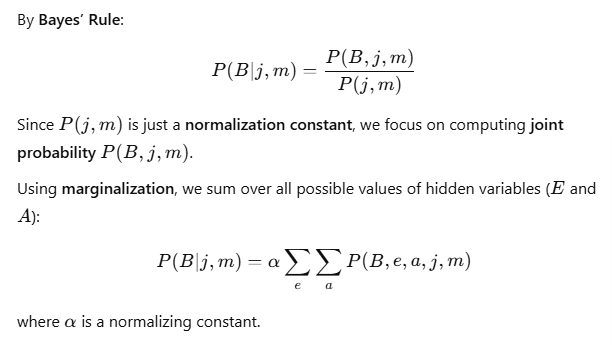


**EXACT INFERENCE**

Inference in Bayesian Networks involves computing the probability of a variable given evidence. Two key exact inference methods are **Inference by Enumeration** and **Variable Elimination**.

**Inference by Enumeration**

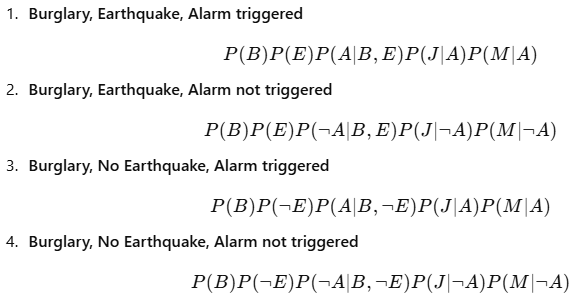
Instead of explicitly constructing the full joint probability distribution (which is exponential in size), we sum out unwanted variables by using conditional probability rules.

Let’s think that we want to compute P (B | j, m) given this network:

Using the chain rule we obtain that:



This means we **sum over all possible values of hidden variables E and A**.

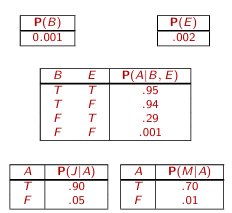


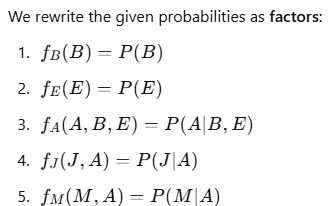
**Inference by Variable elimination**

Variable Elimination (VE) is a more efficient method for **exact inference** in Bayesian networks. Instead of computing the full joint probability table (as in Enumeration), it **eliminates unnecessary variables step by step**, reducing computation by storing intermediate results (factors).

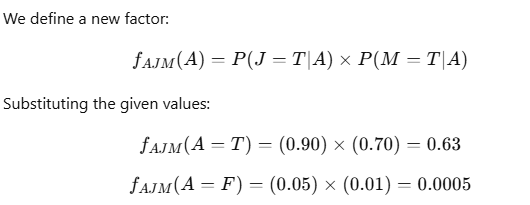
For example

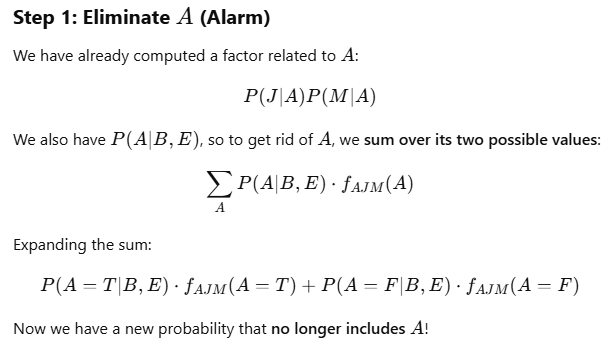
Let’s think that we want to compute P (B | j, m) given the network above

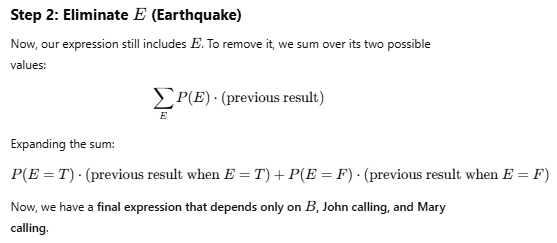
We have these probabilities:

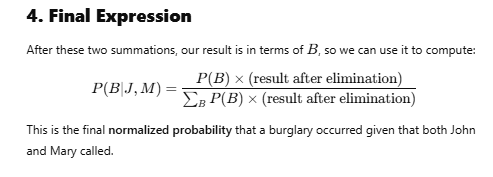


Since John and Mary call **only if the alarm goes off**, we first compute how likely the alarm goes off based on them calling.









**Eliminating a variable** means summing over all its possible values to remove it from the equation.

We **eliminate A** first, because we don’t need to know exactly whether the alarm went off—we just care about burglary.

We **eliminate E** because we don’t care about whether an earthquake happened—we just want to find out if a burglary did.

**Irrelevant Variables**

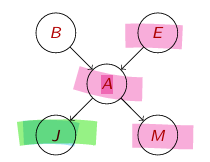
The idea behind **irrelevant variables** is that **some variables do not affect the answer to our query**. If a variable is **not an ancestor** of the variables in our query, then it **does not contribute to the probability calculation** and can be ignored.

This makes our computation much more efficient because we **only focus on relevant variables**.

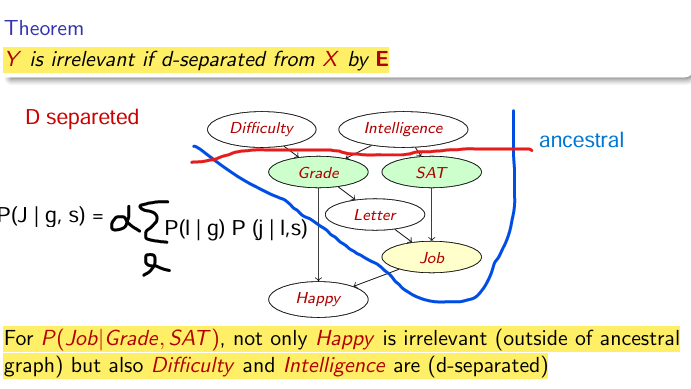
**General Rule**: A variable **Y** is irrelevant **unless it is an ancestor** of the query variable X or the given evidence E.

Example:

Our **query is** P(J ∣ B=true), meaning we are only interested in whether John calls given that a burglary happened.



In this case X is JohnCalls, E is Bulgary = True. The ancestors of E and X are Alarm and Earthquake, so MaryCalls is irrelevant in this case.



**Difficulty** and **Intelligence** are **d-separated** from **Job** when we know **Grade** and **SAT**

**Approximate Inference**

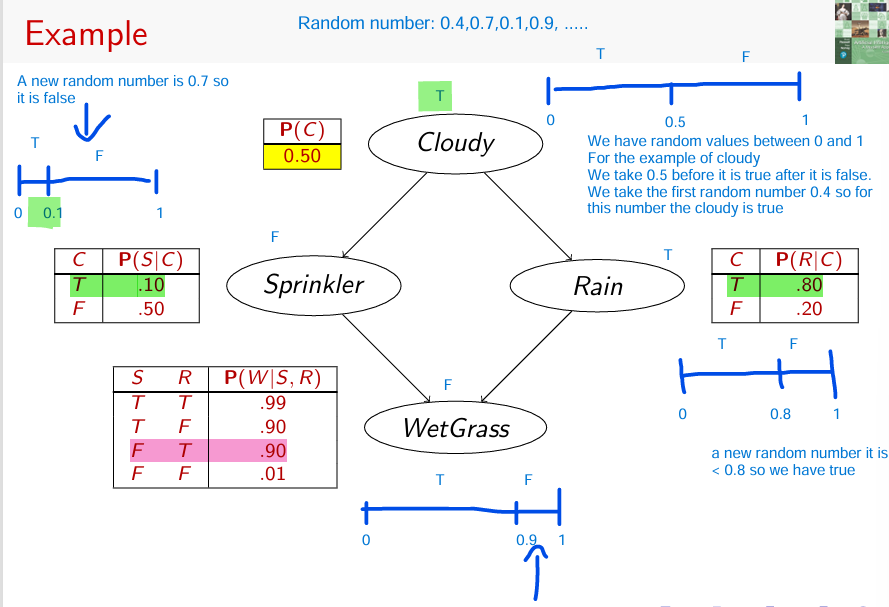
Stochastic simulation methods are used to perform approximate inference in probabilistic graphical models, such as Bayesian networks. The idea is to estimate posterior probabilities by drawing samples from a **sampling distribution** and using these samples to approximate the true probability distribution.

**Sampling from an Empty Network**

This method is called **Prior Sampling**, and it is used when we have no observed evidence (i.e., the network is empty in terms of given information).

**Sample each variable in topological order**:

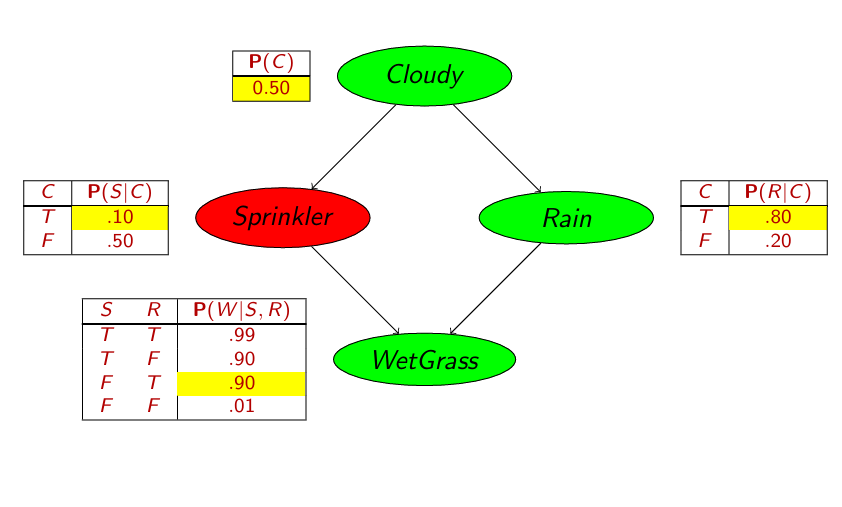
* The **topological order** means that we sample **parent nodes first**, then their **children**, and so on.
* A variable is sampled using the **conditional probability table (CPT)** based on its parents' sampled values.
* Repeat this process N times to generate multiple samples.
* Use the collected samples to estimate probabilities.



1. P(C) = 0.50, The first random number drawn is 0.4 (between 0 and 0.5), so **Cloudy = True**.
2. From the CPT: P(S∣C=T) = 0.10, A new random number 0.7 is drawn, which is greater than 0.1, so **Sprinkler = False**.
3. From the CPT: P(R ∣C=T) = 0.80, A new random number **0.1** is drawn, which is smaller than 0.8, so **Sprinkler = True**.
4. From the CPT: P(W ∣S = F, C = T) = 0.90, The random number is 0.9 that is equal to 0.9 so **WetGrass = True**

The process is repeated multiple times to generate a distribution of samples.

The more samples we generate, the more accurately we can approximate the true probability.

Final Result:

**Rejection Sampling**

When evidence is given (e.g., we know that a certain variable has a fixed value), **rejection sampling** helps us obtain only the valid samples.

**How Rejection Sampling Works**

1. **Use prior sampling** to generate complete assignments.
2. **Reject samples that do not match the evidence** (i.e., discard any sample where the evidence variables have different values from those observed).
3. **Estimate the posterior probability** P(X∣e) using the remaining valid samples.

Example:

P(Rain | Sprinkler = True) using 100 samples.

Only 27 samples from the 100 samples above have Sprinkler = True, so we can throw away the others. Of these 8 have Rain = True and 19 have Rain = False.

P(Rain|Sprinkler =true) = Normalize (⟨8,19⟩) = ⟨0.296, 0.704⟩

We normalize the value in respect of the number of sample:

8/27 = 0.296

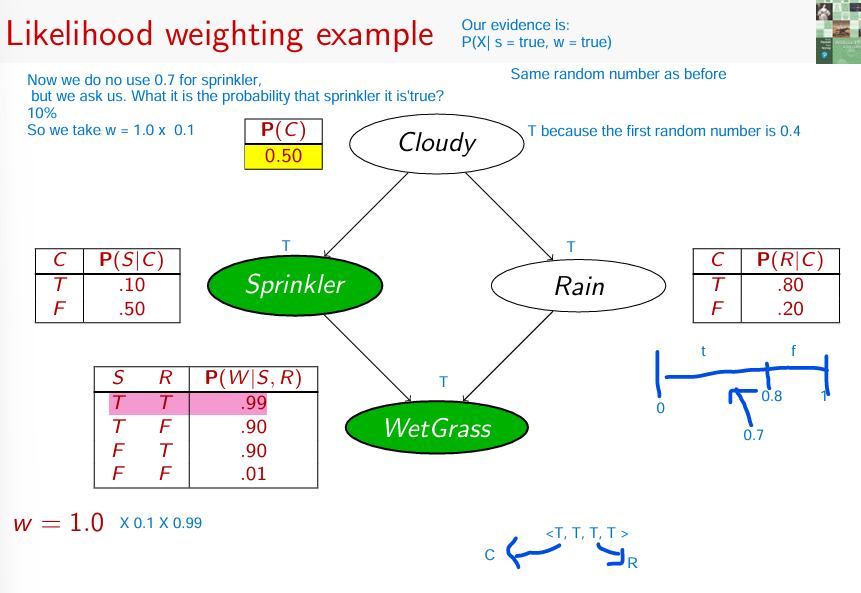
19/27 = 0.704

**Likelihood Weighting**

Likelihood weighting is another technique to approximate posterior probabilities, but it handles evidence more efficiently than rejection sampling.

Unlike rejection sampling where you would reject samples that don't match the evidence, likelihood weighting works by fixing the evidence values and only sampling from the non-evidence variables

1. **Fissare le Evidenze**:  
   Non è necessario campionare le variabili di evidenza, poiché conosci già i loro valori. Ad esempio, se sai che il **Sprinkler** è acceso, non devi campionare di nuovo il valore del **Sprinkler**.
2. **Campionare le Variabili Non Evidenti**:  
   Per le variabili rimanenti (quelle che non fanno parte delle evidenze), le campioni in base alle loro probabilità condizionate. Ad esempio, se stai campionando **Cloudy** e sai che il **Sprinkler** è acceso, campioni **Cloudy** in base alla sua distribuzione condizionata, dato che il **Sprinkler** è acceso.
3. **Pesare Ogni Campione**:  
   Una volta campionati i valori per le variabili (incluso l'evidenza), pesi quel campione in base a quanto è probabile che le evidenze osservate si verifichino, dato i valori campionati delle variabili non evidenti. È qui che entra in gioco la parte di **likelihood**.
4. **Raccogliere i Risultati**:  
   Dopo aver campionato molte volte, accumuli i conteggi pesati per i valori di **X** e li normalizzi per ottenere la probabilità finale.

In breve, il **likelihood weighting** permette di stimare la probabilità di un evento senza scartare campioni, pesandoli invece in base alla loro probabilità di essere compatibili con le evidenze osservate.

**Markov Chain Monte Carlo (MCMC) Example**

MCMC is a method used for approximate inference in probabilistic models, especially when exact solutions are computationally expensive. In the given example, we are estimating the probability P(Rain∣Sprinkler=true, WetGrass=true) using a Markov Chain approach.

**Steps in MCMC Inference:**

1. **Define the States**: In the network (which could represent a Bayesian Network), we have variables like **Cloudy**, **Sprinkler**, **Rain**, and **WetGrass**. These are all random variables, and the "state" of the network is a specific assignment of values to all these variables.
2. **Sample Variables Based on Markov Blanket**:
   * The **Markov Blanket** of a variable includes all the nodes that directly influence it and that it directly influences (its parents, children, and children's parents).
   * For instance, to sample the **Rain** variable, you use the **Markov blanket** (Cloudy, Sprinkler, WetGrass).
3. **Iterate the Sampling Process**:
   * You sample each variable in turn, ensuring that the evidence (the known values) is kept fixed during the process. For example, if you know **Sprinkler=true** and **WetGrass=true**, those values are fixed, and the other variables are sampled.
4. **Update the State**:
   * For each iteration, update the network's state based on the sampled values of each variable.
5. **Count and Normalize**:
   * Over a number of iterations (e.g., 100), count how many times **Rain=true** and **Rain=false** occur in the samples.
   * Finally, normalize these counts to get an approximate probability distribution.

For example:

* + After 100 samples: 31 samples had **Rain=true**, and 69 had **Rain=false**.
  + Therefore, the approximate probability is:

P(Rain∣Sprinkler=true, WetGrass=true)=⟨0.31,0.69)