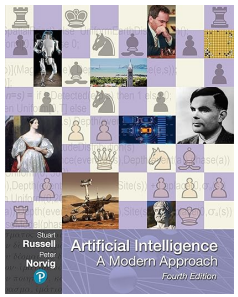
Bayesian Statistics

Instructor: Dr. GP Saggese - gsaggese@umd.edu

References:

- AIMA (Artificial Intelligence: a Modern Approach)
  - Chap 12, Quantifying uncertainty
  - Chap 13: Probabilistic reasoning
  - Chap 14: Probabilistic reasoning over time



- ***Logic-Based AI Under Uncertainty***
- Probabilistic Reasoning

Logic-Based AI Under Uncertainty: Problem

- **Logic-based AI systems**:
  - Based on **propositional logic**
  - Represent **actions** using **rules** like:
    - *"If preconditions P hold, then action A causes effect E"*
  - Example:
    - *"If I turn the car key, the engine starts"*
    - **But**: the battery might be dead, there's no fuel, the starter is broken, etc.
- Real-world agents face **uncertainty** from:
  - Partial observability
    - Agent can't see the full state of the world
  - Non-determinism
    - Actions don't always have predictable outcomes
  - Adversarial conditions
    - Other agents may interfere

Logic-Based AI Under Uncertainty: Naive Solution

- A possible solution to uncertainty is:
  1. Use a **belief state**, i.e., set of all possible current world states
  2. Use **causal and exhaustive augmentation** for rules
     - Must consider all possible preconditions for acting, even unlikely ones
  3. Construct **contingent plans** that handle every possible sensor report and belief
     - Plans become large and complex
     - No guaranteed plan may exist, yet action is required

Logic-Based AI Under Uncertainty: Example

- **Goal**: start a car by turning the car key
- **Obvious preconditions**
  - The car has fuel
  - The car battery is charged
  - The car key is the right one
  - The starter motor is working
  - ...
- **Less obvious preconditions**
  - The fuel lines are not clogged
  - The fuel pump is working
  - The electrical system is intact (no blown fuses)
  - The engine oil level is sufficient
  - ...
- Every precondition requires something different to do to achieve the goal!

Causal and Exhaustive Augmentation

- To use propositional logic under uncertainty, augment the left-side of $X \implies Y$ to make it:
  1. **Causal**: identify true causal-effect relationships
  2. **Exhaustive**: identify all possible conditions leading to the outcome
- **Logical qualification problem**
  - Enumerate all the preconditions necessary for an action to succeed
- **Difficulties**
  - Every action can have more hidden conditions in order to succeed
    - E.g., start the engine $\implies$ turn car key $\implies$ starter motor is working $\implies$ battery is charged $\implies$ ...
  - Preconditions can change depending on the situation
    - E.g., start the engine $\implies$ battery is charged $\implies$ it's not too cold $\implies$ ...

Causal and Exhaustive Augmentation

- **Limitations** of logical qualification
  1. **Laziness**
     - Too much work to create all possible rules
  2. **Theoretical ignorance**
     - Science doesn't always have a complete theory of the domain
     - E.g., medical science doesn't know all the "rules"
  3. **Practical ignorance**
     - Even if you knew all the rules, you might not have all the information needed
     - E.g., not all necessary medical tests can be run for a particular patient
- Expert systems failure and AI winter (mid 1980s, 1990s)
  - The real world is complex and open-ended
  - Logical rules can't capture all necessary and sufficient conditions

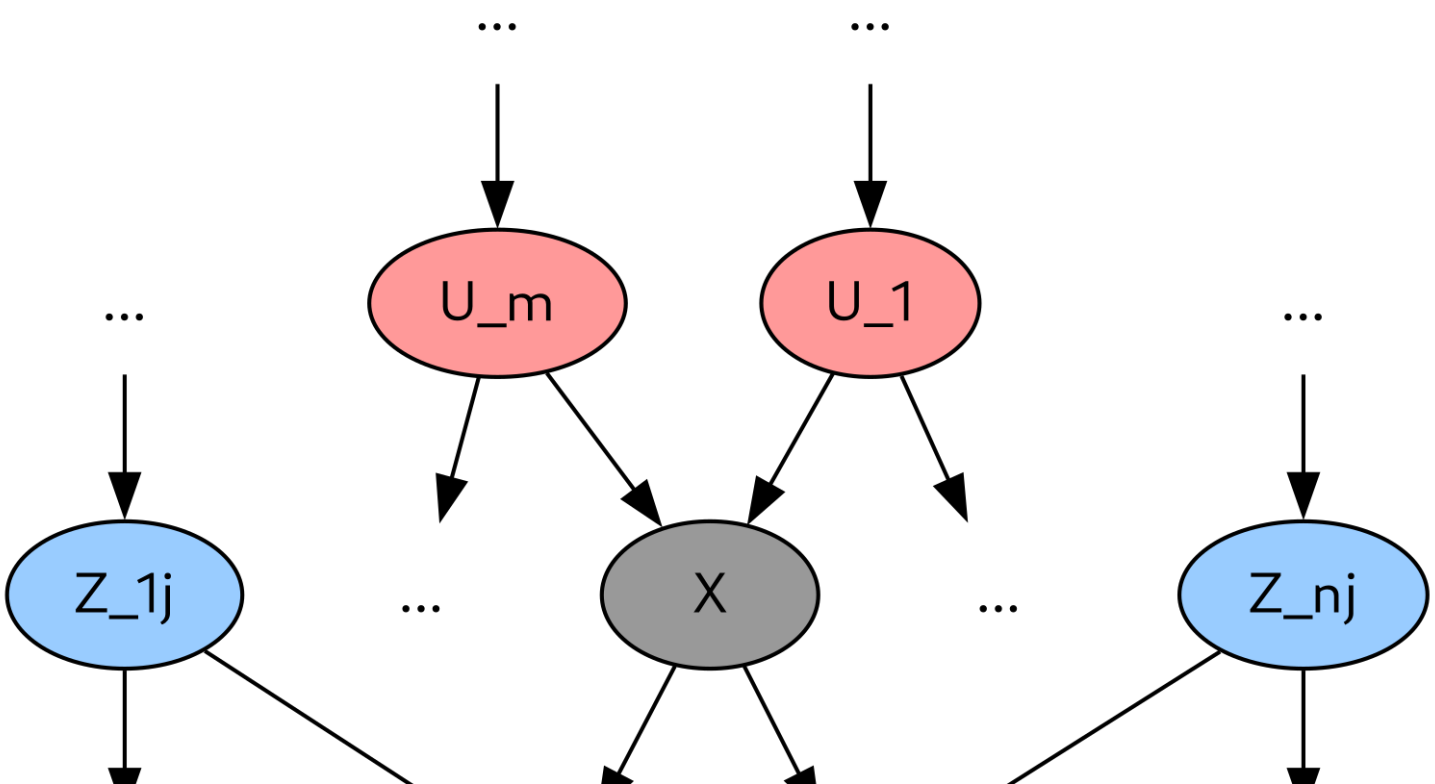
Failure of Logic-Based AI: Wet Grass Example

- Consider the propositions:
  - *Rain* = "it rains"
  - *WetGrass* = "the grass is wet"
- You would expect $Rain \implies WetGrass$

- "$Rain \implies WetGrass$" is not true in general
  - If it rains but there is a cover over the grass, the grass will not be wet
  - If it rains but there is high temperature, the wet grass might dry quickly
- "$WetGrass \implies Rain$" is not true in general
  - The grass could be wet because of a sprinkler system
  - The grass could be wet because of morning dew
- You need to consider also the propositions:
  - *Cover* = "there is a protective cover over the grass"
  - *Evaporate* = "the water evaporates quickly"
  - *Sprinkler* = "the sprinkler system is on"
  - *Dew* = "there is morning dew"

Failure of Logic-Based AI: Wet Grass Example

- Identify all exceptions and dependencies to make $X \implies Y$:
  1. **Causal**
     - "If it rains and there is no other source of water, the grass will be wet"
     - $Rain \implies WetGrass \vee Cover \vee Evaporate \ldots$
     - $Rain \wedge \neg(Cover \wedge Evaporate...) \implies WetGrass$
  2. **Exhaustive**
     - "The grass is wet, if it rains or sprinkler is on or there is morning dew"
     - $WetGrass \implies Rain \vee Sprinkler \vee Dew \ldots$



SCIENCE ACADEMY

Acting Under Uncertainty: Actual Solution

- Can't use propositional logic under uncertainty
- Acting under uncertainty requires combining:
  - **Probability**: to handle uncertainty and partial knowledge
  - **Utilities**: for evaluating desirability of each outcome
- **Key idea:**
  - Rational choice = plan that maximizes expected utility
    - Performance measure: combines goals like punctuality, comfort, legal compliance
    - Belief: agent's internal estimate of outcome likelihoods
  - Evaluate plans based on performance on average, given known information
  - But success is not guaranteed!

The Paradox of Probability and Knowledge

- **Paradox**
  - *"There is no uncertainty in the actual world!"*
  - E.g., the grass is wet, but either it has rained or not
- **Knowledge is subjective**
  - Probabilities relate to a knowledge state, not to the real world
  - Updating knowledge changes probability statements

The Paradox of Probability and Knowledge

- E.g., updating belief about wet grass and rain
- Initially, you observe wet grass
  - From past data you know that $\Pr(Rain|WetGrass) = 0.8$
  - 80% chance it rained if grass is wet
  - This is the Bayesian prior
- You learn new information:
  - Sprinkler was on
  - Wet grass could be due to the sprinkler, not rain
  - Belief changes: $\Pr(Rain|WetGrass \wedge Sprinkler) = 0.4$
  - This is a Bayesian update
- You further observe:
  - Weather report says there was no rain
  - Certain it did not rain, despite wet grass
  - Evidence overrides prior: $\Pr(Rain|WetGrass \wedge WeatherReport) = 0$
- All the statements were true even if contradictory (given the state of knowledge!)
- You need non-monotonic logic

- Logic-Based AI Under Uncertainty
- ***Probabilistic Reasoning***
  - Conditional Independence
  - Bayesian Networks
  - Semantics of Bayesian Networks
  - Constructing a Bayesian Network
  - Exact Inference in Bayesian Networks
  - Approximate Inference in Bayesian Networks

- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - ***Conditional Independence***
  - Bayesian Networks
  - Semantics of Bayesian Networks
  - Constructing a Bayesian Network
  - Exact Inference in Bayesian Networks
  - Approximate Inference in Bayesian Networks

Full Joint Probability Distribution

- Consider a set of random variables $X_1, X_2, \ldots, X_n$
- The **full joint probability distribution**
  - Assigns a probability to every possible world:

$$\Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

    where a **possible world** is a particular assignment of values to all variables
  - Can answer any probabilistic query about the domain
- **Cons**
  - Size grows exponentially k^n with the number of variables n and number of values k
  - Impractical for real-world problems with many variables
  - Manually specifying each entry is tedious
- **Independence** (conditional and absolute) simplifies modeling
  - In the real world, many variables are not fully dependent on all others
  - Reduces the number of variables needed in the model
  - Makes compact and structured representations possible
  - E.g., factorized probabilistic models, Bayesian networks

Independence of Random Variables: Definition

- Two random variables X and Y are **independent** iff:

$$\Pr(X, Y) = \Pr(X) \cdot \Pr(Y)$$

  - Equivalently, knowing Y tells nothing about X

$$\Pr(X|Y) = \Pr(X)$$

- E.g.,
  - The events "coin flip" and "weather" are independent

$$\Pr(Coin = Heads|Weather = Rainy) = \Pr(Coin = Heads)$$

- **Independence** of random variables
  - Reduces the number of parameters needed to model a system, e.g.,

$$\Pr(X_1|X_2, X_3) = \Pr(X_1)$$

  - Allows factorization of joint distribution, e.g.,

$$\Pr(X_1, X_2, X_3) = \Pr(X_1) \cdot \Pr(X_2) \cdot \Pr(X_3)$$

Conditional Independence: Definition

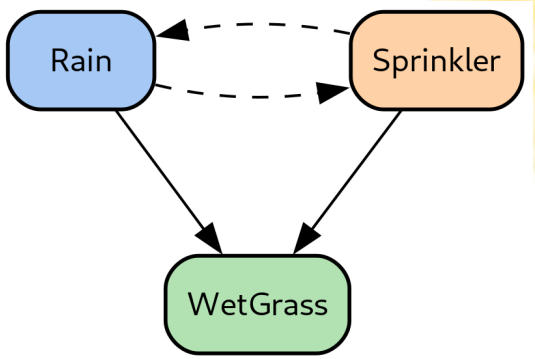
- Two random variables X and Y are **conditionally independent** given a random variable Z iff knowing Z makes X and Y independent:

$$X \perp Y|Z \iff \Pr(X, Y|Z) = \Pr(X|Z) \cdot \Pr(Y|Z)$$

- **Example**
  - $X =$ *"it is raining today"*
  - $Y =$ *"a person is carrying an umbrella"*
  - $Z =$ *"the weather forecast"*
  - **Without** Z: there is a relationship between X and Y (i.e., X and Y are not independent)
  - **Given** Z: rain X may not directly influence whether a person carries an umbrella Y
  - Thus, X and Y are conditionally independent given Z
- **Pros**
  - Conditional independence is more common than absolute independence
  - Simplify probabilistic models by factorizing the joint conditional distribution into product of individual conditional distributions

Conditional Independence: Example

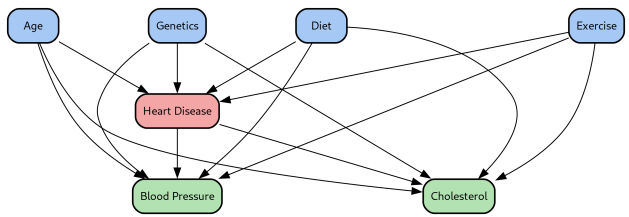
- Two events can become independent once we know a third event
- **Example**
  - *Fire* = "there is a fire"
  - *Toast* = "someone burned toast"
  - *Alarm* = "the fire alarm rings"
  - *Call* = "a friend calls to check on you"
- **Dependencies**
  - *Alarm* depends on *Fire* or *Toast*
  - *Call* depends on whether *Alarm* rings
- **Conditional independence**
  - Once we know the alarm rang, the specific cause doesn't affect whether the friend calls
  - $\Pr(Call \mid Alarm, Fire) = \Pr(Call \mid Alarm)$



- **Interpretation**
  - *Call* is conditionally independent of *Fire* and *Toast* given *Alarm*
  - Knowing the alarm rang "blocks" the path of influence from *Fire* and *Toast* to *Call*

Conditional Independence: Garden Example

- Garden world with *Rain*, *Sprinkler*, and *WetGrass*
  - Is $\Pr(Rain|Sprinkler) = \Pr(Rain)$?
    - **No**: if the sprinkler is on, it's less likely it rained
    - *Rain* and *Sprinkler* are not independent
  - Is $\Pr(Rain|Sprinkler, WetGrass) = \Pr(Rain|WetGrass)$?
    - **Yes**: knowing the grass is wet, whether the sprinkler was on tells you nothing more about the rain
    - *Rain* and *Sprinkler* are conditionally independent given *WetGrass*
- **Interpretation**:
  - Without *WetGrass*: *Rain* and *Sprinkler* affect each other because they both explain *WetGrass*
  - With *WetGrass*: once *WetGrass* is observed, the "explaining away" effect occurs



- **"Explaining away" occurs when**
  - Two variables (causes) independently influence a third variable (effect)
  - Observing the effect creates a dependence between the causes
  - Evidence for one explains the effect and reduces the need to believe in the other

- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - Conditional Independence
  - ***Bayesian Networks***
  - Semantics of Bayesian Networks
  - Constructing a Bayesian Network
  - Exact Inference in Bayesian Networks
  - Approximate Inference in Bayesian Networks

Bayesian Networks: Definition

- Aka:
  - "Bayes nets",
  - "Belief networks"
  - "Probabilistic networks"
  - "Graphical models" (somehow a broader class of statistical models)
  - "Causal networks" (arrows have special meaning)
- A **Bayesian network** is a Directed Acyclic Graph (DAG)
  1. **Nodes** X_i correspond to random variables (discrete or continuous)
  2. **Edges** $X \rightarrow Y$ connect nodes and represent direct dependencies among variables
     - We say that $X = Parent(Y)$, $Y = Child(X)$, ancestors, spouses, ...
  3. Each node X_i is associated with a **conditional probability** (CPD):

     $$\Pr(X_i | Parents(X_i))$$

     quantifying the effect of the parents on the node
     - If a node has no parents, it has a prior probability

Bayesian Network: Intuition

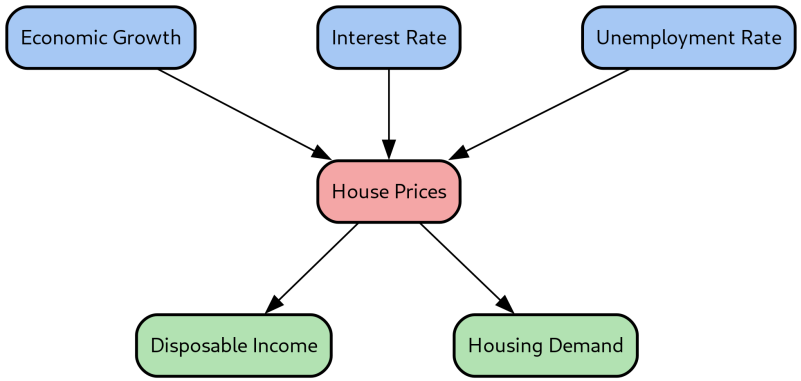
- Bayesian networks are the analogous for uncertain knowledge to propositional logic for definite knowledge
  - Propositional logic = rigid rules, i.e., `True` or `False`
  - Bayesian networks = flexible inference, i.e., degrees of belief
  - Replace $X \implies Y$ with $\Pr(Y|X)$
- E.g., wet grass example
  - R = "It is raining"
  - W = "The grass is wet"
  - Propositional logic
    - If $R \rightarrow W$ and R is true, then W must be true
  - Bayesian network
    - $\Pr(R = \mathit{True}) = 0.2$
    - $\Pr(W|R) = 0.9$
    - $\Pr(W|\neg R) = 0.1$
- E.g., medical diagnosis
  - Propositional logic
    - "Patient has disease D" $\implies$ "Patient has symptom S"
  - Bayesian network
    - "Probability of S given D is high, but not certain"

Bayesian Network and Full-joint distribution

- It can be shown that **topology** and **conditional probabilities** are sufficient to specify the full joint distribution
  - **Any full joint** distribution
  - **Very concisely** (often)
- Nodes are:
  - Directly influenced by their parents
  - Indirectly influenced by all their ancestors
- The topology of the network (nodes and edges) specifies conditional independence relationships
  - E.g., $X \rightarrow Y$ means *"X has a direct influence on Y"*, i.e., *"X relates to Y"* (not necessarily "causes")
- How to build a Bayesian Network:
  - Domain experts can decide what relationships exist among domain variables, determining the topology
  - Conditional probabilities can be specified or estimated

Bayesian Networks: Wet Grass Example

- Consider a world with 5 variables
  - *Weather*
    - Represents general environmental conditions (e.g., sunny, cloudy)
    - Can't be observed
  - *Rain*
    - Directly influenced by *Weather*
  - *Sprinkler*
    - Influenced by *Weather* (e.g., less likely when raining)
  - *WetGrass*
    - Represents whether the grass is wet
    - Affected by both *Rain* and *Sprinkler*
  - *StockMarketUp*
    - Indicates whether the stock market is up or down

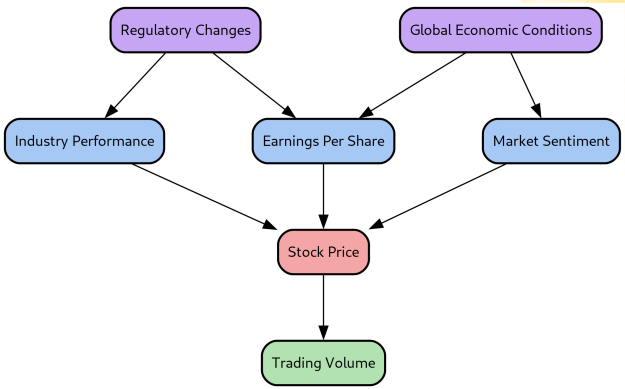


Bayesian Networks: Wet Grass Example

- *Rain* and *Sprinkler* are **dependent** (marginally)

$$Rain \not\perp Sprinkler \iff Rain \leftrightarrow Sprinkler$$

  - If there is no *Rain*, *Weather* = *sunny* and *Sprinkler* is more likely to happen (since sprinklers are usually turned on in sunny weather)
  - *Rain* and *Sprinkler* are related since they share a common parent *Weather*



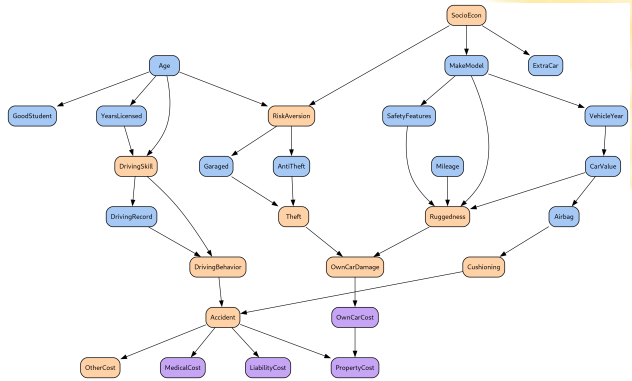
- *Rain* and *Sprinkler* are **conditionally independent** given *Weather*

$$Rain \perp Sprinkler | Weather$$

  - Once *Weather* is fixed, knowing whether it rained tells you nothing about the *Sprinkler*
  - The correlation is "explained away" by *Weather*

Bayesian Networks: Wet Grass Example

- *Rain* and *Sprinkler* are **conditionally independent** given *WetGrass*, but only if *Weather* is not observed



$$Rain \perp Sprinkler | WetGrass(notWeather)$$

  - If we condition on *WetGrass*, i.e., the grass is wet, knowing that the sprinkler was off increases the chances that it must have rained (and vice versa)

- *Rain* and *Sprinkler* are **conditionally dependent** given *WetGrass*, and *Weather*

$$Rain \not\perp Sprinkler | WetGrass, Weather$$

  - If you know that it's sunny, rain is unlikely and sprinkler likely
  - If you also know that *WetGrass*
    - If you learn *Sprinkler* off $\implies$ must have *Rain*
  - Fork path (common cause) and collider path (common effect)
- *StockMarketUp* is **(unconditionally) independent** of all other variables

Conditional Probability Table

- **Conditional Probability Table** (CPT) encodes the probability of one node X_i given its parents $Parents(X_i)$

$$\Pr(X_i|Parents(X_i))$$

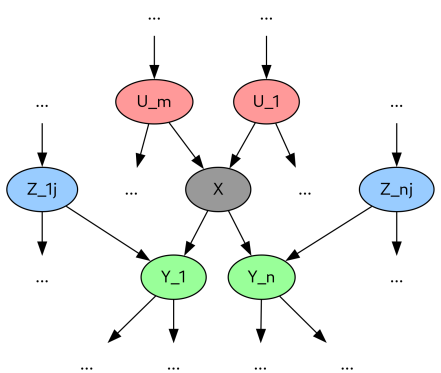
- Each row of the CPT contains the conditional probability of the node under a conditioning case (i.e., a possible combination of the values for the parent nodes)
- E.g., $\Pr(A|B,C)$

A	P(A|B,C)	P(A|B,-C)	P(A|-B,C)	P(A|-B,-C)
True	0.80	0.10	0.05	0.05
False	0.05	0.85	0.10	0.0

- **Note**:
  - Natural for discrete variables, but can be extended to continuous variables
  - A conditional probability table summarizes an infinite set of circumstances in the table

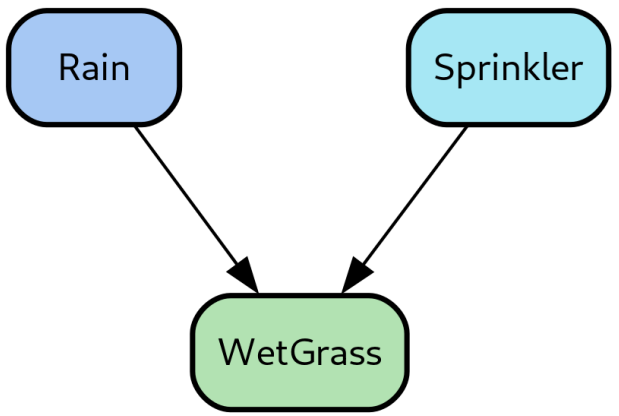
Bayesian Networks: Burglar Example

- Famous example from Judea Pearl
- An *Alarm* system installed at a home in LA
  - Fairly reliable at detecting *Burglary*
  - Also responds to minor *Earthquakes* (false positive)
- Two neighbors, *John* and *Mary* will *Call* you when they hear the *Alarm*
  - *John*:
    - Almost always *Call*s when he hears the alarm
    - Sometimes confuses telephone with the *Alarm* and *Call*s (false positives)
  - *Mary*:
    - Misses the alarm 30% of the cases (false negatives)
- The **structure of the graph** shows that:
  - *Burglary* and *Earthquake* affects the event *Alarm*
  - *JohnCalls* and *MaryCalls* depend only on the *Alarm*, and not on *Burglary* and *Earthquake*



Bayesian Networks: Burglar Example

- The probability of *Burglary* is 0.001
- The probability of *Earthquake* is 0.002
- *Alarm* system
  - Is fairly reliable at detecting *Burglary*
  - Responds to minor *Earthquakes*



Burglary	**Earthquake**	**P(Alarm| B,E)**
True	True	0.95
True	False	0.90
False	True	0.30
False	False	0.01

- Since events are independent:

$$\Pr(Alarm) = \Pr(Alarm|Burglary, Earthquake)\Pr(Burglary)\Pr(Earthquake)$$

Bayesian Networks: Burglar Example

- Two neighbors, *John* and *Mary* will *Call* you when they hear the *Alarm*
  - *John*:
    - Almost always *Call*s when he hears the alarm
    - Sometimes confuses telephone with the *Alarm* and *Call*s (false positives)
  - *Mary*:
    - Misses the alarm 30% of the cases (false negatives)
- *JohnCalls* and *MaryCalls* are represented by:

Alarm (A)	**P(JohnCalls| Alarm)**
True	0.90
False	0.05

Alarm (A)	**P(MaryCalls| Alarm)**
True	0.70
False	0.01

Conditional Probability Table

- A **node without parents** has an unconditional probability

P(Burglary)
.001

- The **sum of probabilities** must be 1
  - If there is a single input variable, it is possible to remove the redundancy

Alarm (A)	P(JohnCalls| .)	P(-JohnCalls | .)
True	0.90	0.10
False	0.05	0.95

Alarm (A)	P(JohnCalls| .)
True	0.90
False	0.05

- **A node with k parents** has 2^k possible rows in the table

Burglary	Earthquake	P(Alarm | .)
T	T	.95
T	F	.94
...	...	...

- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - Conditional Independence
  - Bayesian Networks
  - ***Semantics of Bayesian Networks***
  - Constructing a Bayesian Network
  - Exact Inference in Bayesian Networks
  - Approximate Inference in Bayesian Networks

Bayesian Networks: Semantics

- There are **two equivalent semantic interpretations** of a Bayesian Network

1. **Joint Distribution View**
   - The network encodes the *joint probability distribution* over all variables
   - Computed as the product of local conditional probabilities:

$$P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i \mid \text{Parents}(X_i))$$

   - Useful for constructing models and understanding overall behavior
2. **Conditional Independence View**
   - The structure encodes *conditional independency* between variables
   - Useful for inference and reasoning
   - A variable is conditionally independent of its non-descendants given its parents

Chain Rule for a Joint Distribution

- A **joint distribution** can always be expressed using the chain rule for any:
  - Subset of its RVs
  - Ordering of the RVs

1. You **express one variable** conditionally to the remaining ones

$$\Pr(x_1, \ldots, x_{n-1}, x_n) = \Pr(x_n | x_{n-1}, \ldots, x_1) \Pr(x_{n-1}, \ldots, x_1)$$

2. Apply the same formula **recursively**, until you get an unconditional probability

$$\begin{aligned}
&\Pr(x_1, x_2, \ldots, x_{n-2}, x_{n-1}, x_n) \\
&= \Pr(x_n | x_{n-1}, \ldots, x_1) \Pr(x_{n-1}, \ldots, x_1) \\
&= \Pr(x_n | x_{n-1}, \ldots, x_1) \Pr(x_{n-1} | x_{n-2}, \ldots, x_1) \Pr(x_{n-2}, \ldots, x_1) \\
&\ldots \\
&= \Pr(x_n | x_{n-1}, \ldots, x_1) \Pr(x_{n-1} | x_{n-2}, \ldots, x_1) \Pr(x_{n-2} | x_{n-3}, \ldots, x_1) \ldots \Pr(x_2 | x_1) \Pr(x_1) \\
&= \prod_{i=1}^{n} \Pr(x_i | x_{i-1}, \ldots, x_1)
\end{aligned}$$

Statement Probability from Bayesian Network

- The **full joint distribution** represents the probability of an assignment to each variable $X_i = x_i$:

$$\Pr(x_1, ..., x_n) \triangleq \Pr(X_1 = x_1 \wedge ... \wedge X_n = x_n)$$

- To **evaluate a Bayesian network**
  - Sort the nodes in topological order
    - There are several orderings consistent with the directed graph structure
  - Use the chain rule with the topological ordering:

$$\Pr(X_1, ..., X_n) = \prod_{i=1}^{n} \Pr(X_i | X_{i-1}, ..., X_1)$$

  - Since the probability of each node is conditionally independent of all its predecessors given its parents

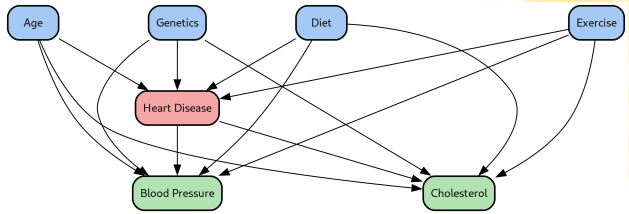
$$\Pr(X_i | X_{i-1}, ..., X_1) = \Pr(X_i | Parents(X_i))$$

  - Express the joint probability in terms of the Conditional Probability Tables (CPTs):

$$\Pr(X_1, ..., X_n) = \prod_{i=1}^{n} \Pr(X_i | Parents(X_i))$$

Statement Probability From Bayes Nets: Example

- Given Pearl LA example, you want to compute the probability that:
  - The alarm has sounded: *Alarm*
  - Neither a burglary nor an earthquake has occurred: $\neg Burglary \wedge \neg Earthquake$
  - Both John and Mary call: *JohnCalls*, *MaryCalls*



- **Solution**
  - Compute the probability as a product of conditional probabilities from the Bayesian Network

$$\begin{aligned}
&\Pr(JohnCalls, MaryCalls, Alarm, \neg Burglary, \neg Earthquake) \\
&= \Pr(JohnCalls|Alarm)\cdot \\
&\qquad \Pr(MaryCalls|Alarm)\cdot \\
&\qquad \Pr(Alarm|\neg Burglary \wedge \neg Earthquake)\cdot \\
&\qquad \Pr(\neg Burglary)\cdot \Pr(\neg Earthquake)
\end{aligned}$$

- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - Conditional Independence
  - Bayesian Networks
  - Semantics of Bayesian Networks
  - ***Constructing a Bayesian Network***
  - Exact Inference in Bayesian Networks
  - Approximate Inference in Bayesian Networks

Constructing a Bayesian Network

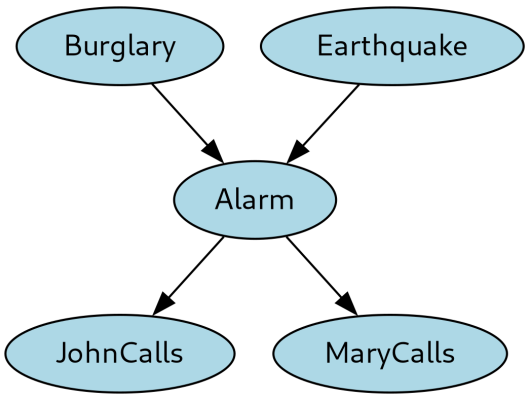
1. **Gather domain knowledge**
   - Identify key variables and their potential interactions
   - List all relevant random variables necessary to describe the system
2. **Order the nodes** according to cause-effects dependencies
   - So that the Bayesian network is minimal
3. For each node, **pick the minimum set of parents** $Parents(X_i)$
   - Add edges to represent the dependencies
   - Avoid redundant connections
4. **Estimate the conditional probability** $\Pr(X_i|Parents(X_i))$ for each node
   - Gather data or expert opinion
   - Use statistical techniques if necessary
5. **Validate the model**
   - Have domain experts review it
   - Ensure that the network is a Directed Acyclic Graph (DAG)
   - Test the network by predicting known outcomes and comparing with actual data

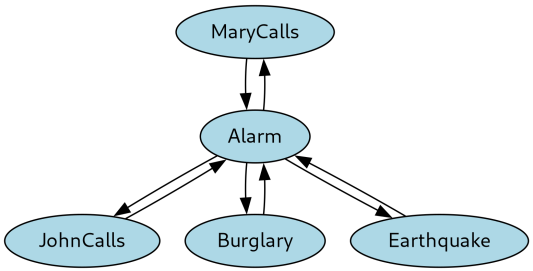
Bayesian Networks: Properties

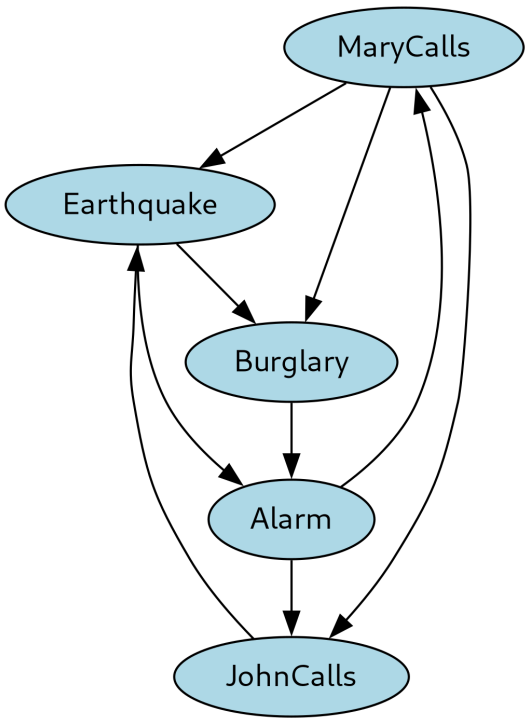
- Bayesian networks are a representation with several interesting properties
  - **Complete**
    - Encode all information in a joint probability
  - **Consistent** (non-redundant)
    - In a Bayesian network, there are no redundant probability values
    - One (e.g., a domain expert) can't create a Bayesian network violating probability axioms
  - **Compact** (locally structured, sparse)
    - Each subcomponent interacts directly with a limited number of other components
    - Typically yields linear (not exponential) growth in complexity
    - Sometimes we ignore real-world dependency to keep the graph simple
- In **fully connected systems**
  - Each variable is influenced by all others
  - The Bayesian network has the same complexity as the joint probability

Ordering of Nodes

- The complexity of the Bayesian network depends on the choice in ordering the nodes



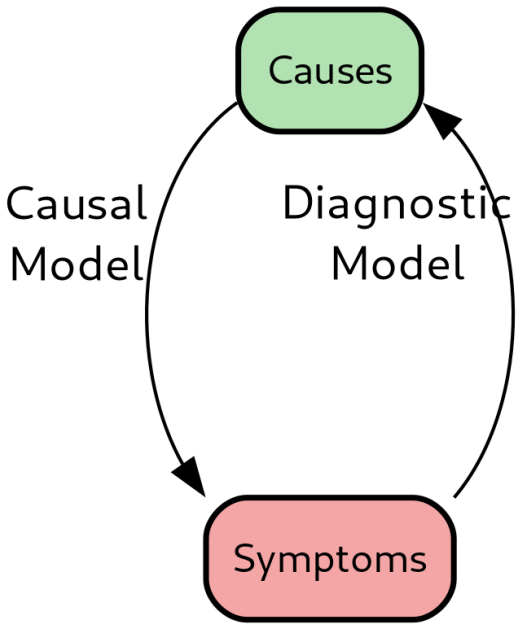




- The graph is "minimal" in terms of connectivity when all edges are causal

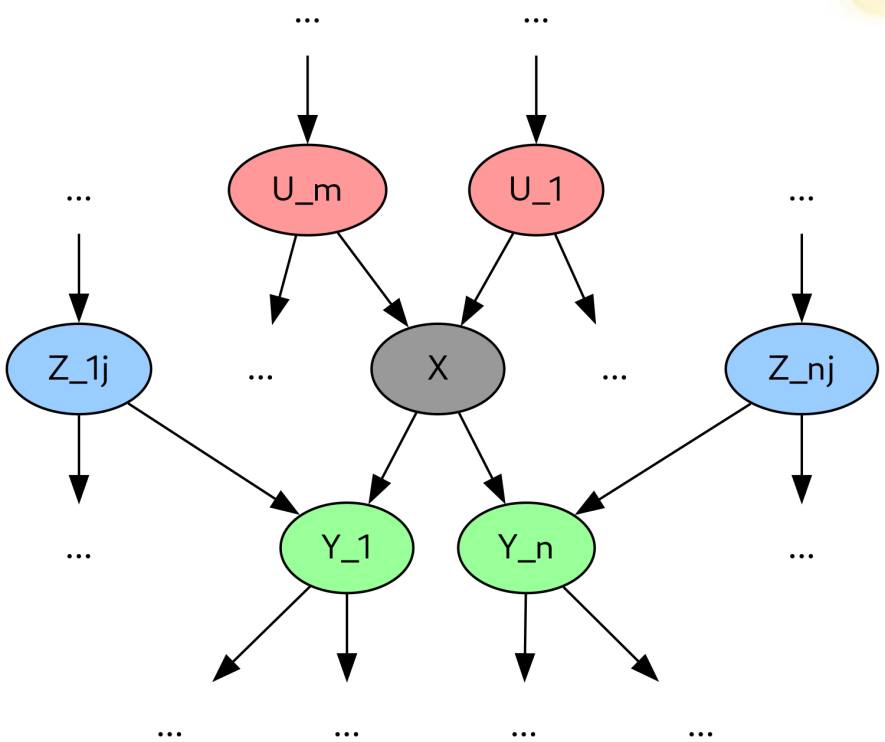
Causal vs Diagnostic Models

- A **causal model** goes from causes to symptoms
  - E.g., *Burglary* $\rightarrow$ *Alarm*
  - Simpler (i.e., fewer and more robust dependencies)
  - "Easier" to estimate
- A **diagnostic model** goes from symptoms to causes
  - E.g., *MaryCalls* $\rightarrow$ *Alarm* or *Alarm* $\rightarrow$ *Burglary*
  - Tenuous / unstable
  - Difficult to estimate
  - This is what we care about: use Bayes' rule to invert the probability



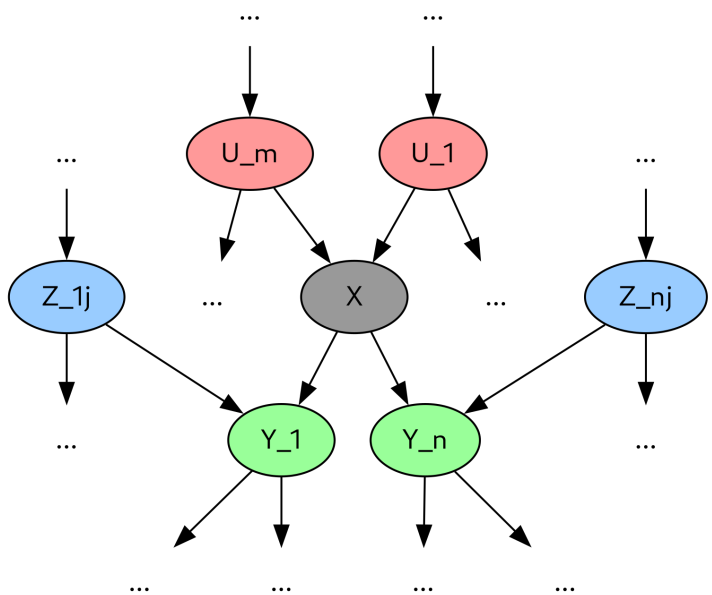
Markov Blanket of a Node

- The **Markov blanket** of a **node** X consists of:
  1. The **parents** of X
     - The nodes that influence X
  2. The **children** of X
     - The nodes that are directly influenced by X
  3. The **spouses** of X
     - The nodes that are parents of the children nodes
     - I.e., "co-parent"



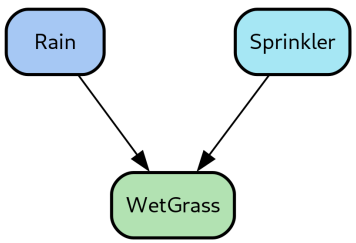
Conditional Independence on Markov Blanket

- In a Bayesian network, each variable is conditionally independent of:
  - **Its predecessors given its parents**
  - **All other nodes given its Markov blanket**, i.e., its parents, its children, and its spouses
- The **Markov blanket** of a node X_i:
  - Contains all the nodes necessary to predict the state of the node X_i, making the network irrelevant
  - Enables efficient and localized inference



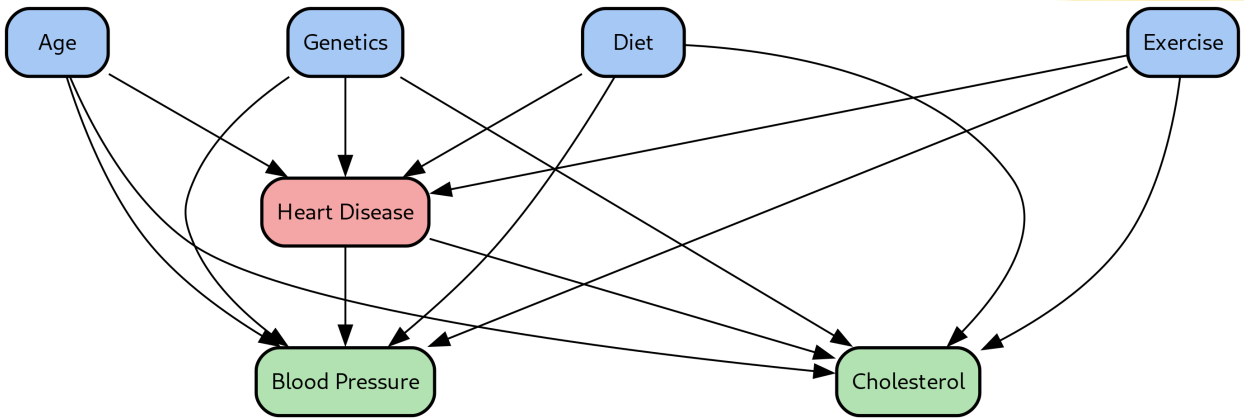
How Can a Node Be Influenced by Its Children?

- A **descendant can influence its ancestor** indirectly through *"explaining away"*
  - Evidence about the descendant can change what you believe about the ancestor through dependent paths
  - Information flows both ways in Bayesian networks
- E.g.,
  - Consider the Garden World
  - You know the grass is wet *WetGrass*
  - This evidence increases the probability of either causes *Rain* or *Sprinkler*
  - If you find out that the *Sprinkler* was on, this "explains away" the *WetGrass*, and the probability of *Rain* goes down
  - The evidence from a descendant *WetGrass* can update your belief about an ancestor *Rain*



Markov Blanket: Medical Example

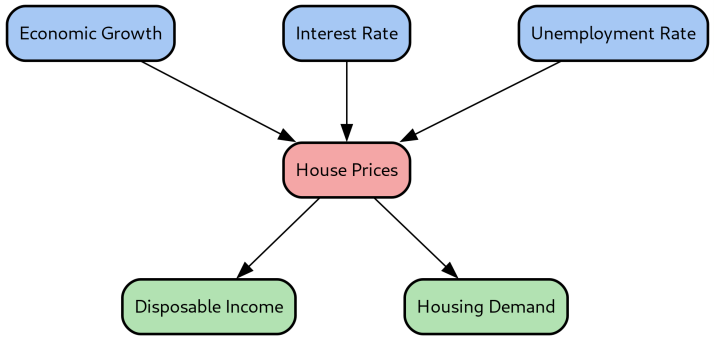
- Consider risk factors and outcomes for heart disease



- **Target node**
  - Heart disease
- **Parent nodes**
  - Risk factors of heart disease
  - Direct influence of H
- **Children nodes**
  - Outcomes of heart disease
  - Directly influenced by $H \rightarrow$
- **Spouse nodes**
  - A, G, D, E also influence BP and C
- Knowing the state of A, G, D, E, BP, C (Markov Blanket) allows to compute H, without any other information

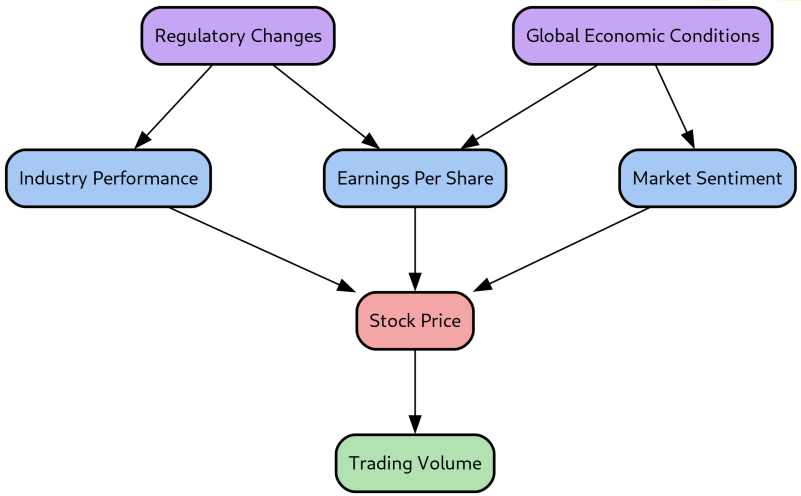
Markov Blanket: Economic Example

- Consider factors affecting house prices in a particular region
- **Target node**
  - House prices
- **Parent nodes**
  - Economic growth
  - Interest rate
  - Unemployment rate
- **Children nodes**
  - Disposable income
    - The house price affects how much money people have left after housing costs
  - Demand for houses
    - Higher prices can reduce demand



Markov Blanket: Finance Example

- Consider factors affecting an individual company's stock price
- **Target node**
  - *SP*: Stock Price
- **Parent nodes**
  - *IP*: Industry performance
  - *EPS*: Earnings per share
  - *MS*: Market sentiment



- **Children nodes**
  - *TV*: Trading volume
    - Changes in stock price influence how much stock is being traded
- **Grandparents nodes**
  - *RC*: Regulatory changes in the technology sector
    - Influences *IP* and *EPS*, but not directly *TV*
  - *GE*: Global economic conditions
    - Influences *MS* and *EPS*, but not directly *TV*
- You don't need to know *RC* and *GE* to estimate Stock Price

Specifying a Conditional Probability Table

- Conditional Probability Table (CPT) for a node requires $O(2^k)$ values in the worst case
  - Difficult to specify even with a small number of parents k
- Often, the relationship is not completely arbitrary, e.g.,
  - Deterministic nodes
  - Noisy logical relationships
  - Context-specific independence

Deterministic Nodes

- **Deterministic nodes** have values specified by their parents, without uncertainty, e.g.,
  - A logical relationship:
    - Useful when a condition is met if any of the sub-conditions are true
    - $IsNorthAmerican = IsCanadian \vee IsUS \vee IsMexican$
  - A numerical relationship:
    - $BestPrice = \min(Price_i)$
- **Note**
  - Deterministic nodes do not involve randomness or probability
  - Often used in models to simplify relationships and computations

Noisy Logical Relationships

- **Noisy logical relationships** (e.g., noisy-OR, noisy-MAX):
  - Are a probabilistic version of a logical relationship
  - Can be simpler to describe given k parents
- **Example**
- In propositional logic

$$Fever \iff Cold \vee Flu \vee Malaria$$

- In Bayesian networks
  - The assumptions are:
    1. All the possible causes are listed (you can use a leak node for "misc causes")
    2. There is uncertainty about the parents to cause the child node
    3. The probabilities of parents are independent
  - Under these assumptions:

$$\begin{aligned}&\Pr(Fever|parents(Fever))\\&\quad = 1 - \Pr(\neg Fever|Cold, \neg Flu, \neg Malaria)\cdot\\&\quad \Pr(\neg Fever|\neg Cold, Flu, \neg Malaria)\cdot\\&\quad \Pr(\neg Fever|\neg Cold, \neg Flu, Malaria)\end{aligned}$$

Context-specific Independence

- A variable exhibits **context-specific independence** if it is conditionally independent of its parents given certain values of others
- **Example**
  - *Damage* occurs depending on the *Ruggedness* of your car and whether an *Accident* occurred in that period:

$$\Pr(Damage \mid Ruggedness, Accident) = \begin{cases} d_1 & \text{if } Accident = True \\ d_2(Ruggedness) & \text{if } Accident = False \end{cases}$$

where $d1$ and $d2$ are distributions

Bayesian Networks with Continuous Variables

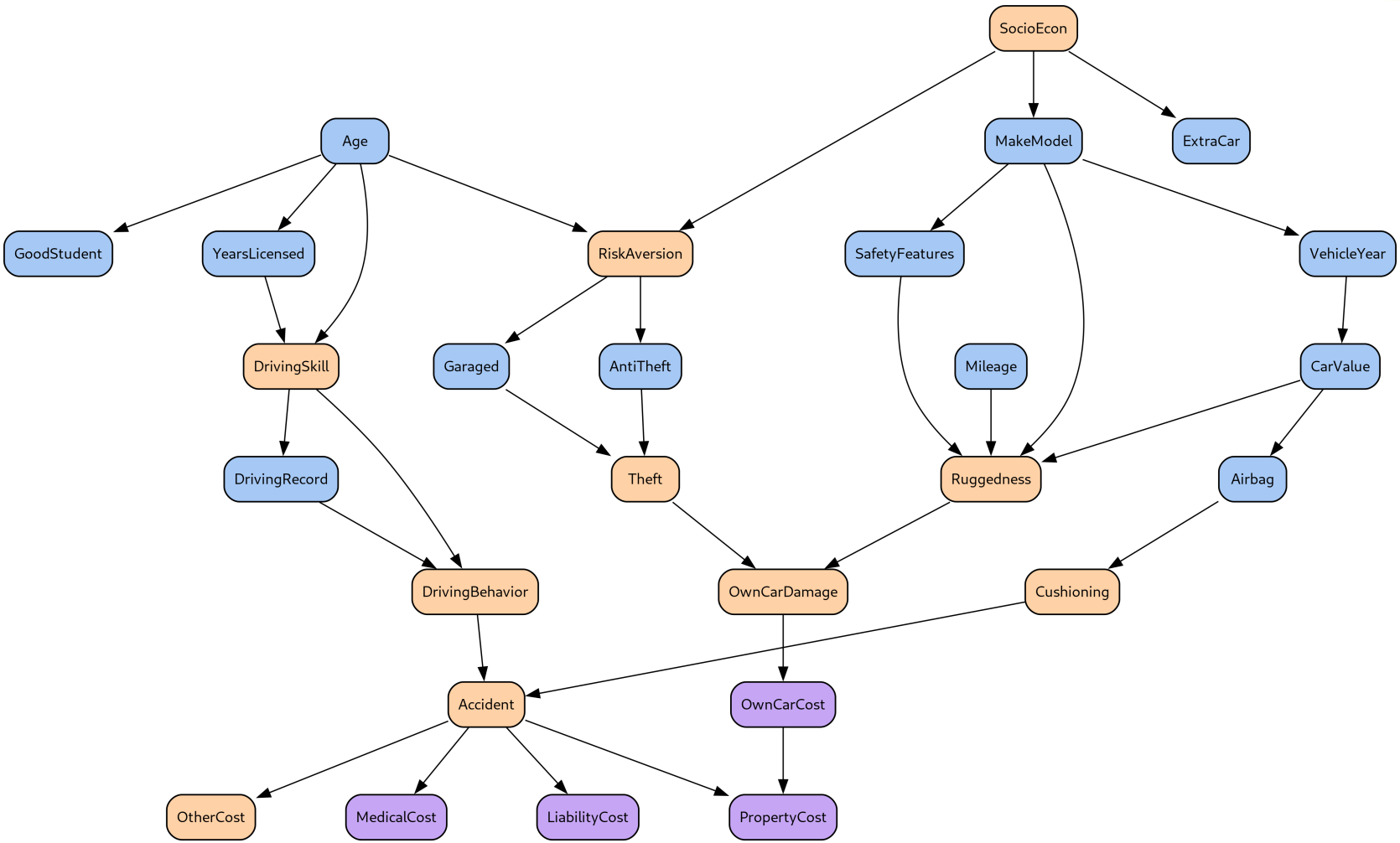
- Many real world problems involve continuous quantities
  - E.g., height, mass, temperature, money
- **Problem**: Conditional Probability Table (CPT) is not suitable for continuous RVs
- **Solution**:
  1. Discretization (i.e., use intervals)
     - Cons: loss of accuracy and large CPTs
  2. Continuous variables
     - Families of probability density functions (e.g., Gaussian distribution)
     - Non-parametric PDFs
- **Hybrid Bayesian networks** mix discrete and continuous variables, e.g.,
  - A customer buys a number of apples (discrete) depending on its cost (continuous)
  - Decide annual premium to charge to insure a vehicle based on applicant information (e.g., make model)

Bayesian Network: Car Insurance Company (1/2)

- A **car insurance company**:
  - Receive an application from an individual to insure a specific vehicle
  - Analyze information about the individual and its car
  - Decide on appropriate annual premium to charge
  - Pay out a claim, based on the type of claim
- Build a Bayesian network that captures the causal structure of the domain
  - **Input information**:
    - About the applicant: *Age*, *YearsWithLicense*, *DrivingRecord*, *GoodStudent*
    - About the vehicle: *MakeModel*, *VehicleYear*, *Airbag*, *SafetyFeatures*
    - About the driving situation: *Mileage*, *HasGarage*
  - Some input informations are **important but not available**:
    - *RiskAversion*
    - *DrivingBehavior*
  - **Type of claims**:
    - *MedicalCost*: injuries sustained by the applicant
    - *LiabilityCost*: lawsuits filed by other parties against applicant
    - *PropertyCost*: vehicle damage to either party and theft of the vehicle

Bayesian Network: Car Insurance Company (2/2)

- **Blue nodes**: information provided by the applicants
- **Brown nodes**: hidden variables (not observable)
- **Violet nodes**: target variables



- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - Conditional Independence
  - Bayesian Networks
  - Semantics of Bayesian Networks
  - Constructing a Bayesian Network
  - ***Exact Inference in Bayesian Networks***
  - Approximate Inference in Bayesian Networks

Exact Inference in Bayesian Networks

- **Goal of exact inference**
  - Compute the posterior $P(X|\underline{\boldsymbol{E}} = \underline{\boldsymbol{e}})$ for query variable X given evidence $\underline{\boldsymbol{e}}$
- **Variables involved**
  - Query variable X
  - Evidence variables $\underline{\boldsymbol{E}} = \{E_1, \ldots, E_m\}$
  - Hidden variables $\underline{\boldsymbol{Y}} = \{Y_1, \ldots, Y_\ell\}$
- **Inference by Enumeration**
  - Use full joint distribution and sum over all hidden variables:

$$P(X|e) = \alpha \sum_{Y} P(X, e, Y)$$

- **Variable Elimination**
  - Improves efficiency by caching intermediate results
  - Eliminates variables systematically to avoid redundant sums
  - Removing irrelevant variables
    - Variables not ancestors of query or evidence can be ignored
- **Problems**
  - Exact inference is efficient $O(n)$ for trees, but intractable $O(2^n)$ in general
  - It doesn't work for continuous variables
  - Basis for approximate methods when exact is impractical

Exact Inference in Bayesian Networks: Example

- You get a call from both John and Mary, what is the probability of the burglary?

$$P(Burglary|JohnCalls = True, MaryCalls = True)$$

- A conditional probability can be computed summing terms from the full joint distribution

$$\Pr(X|\underline{\boldsymbol{e}}) = \alpha \Pr(X, \underline{\boldsymbol{e}}) = \alpha \sum_{y} \Pr(X, \underline{\boldsymbol{e}}, \underline{\boldsymbol{y}})$$

- Terms of the joint distribution can be written as products of conditional probabilities from the Bayesian network

$$\Pr(b|j, m) = \alpha \Pr(B, j, m) = \alpha \sum_{e} \sum_{a} \Pr(B, j, m, e, a)$$

- Then the joint probability is written in terms of CPTs of the Bayesian network

$$\Pr(b|j, m) = \alpha \sum_{e} \sum_{a} \Pr(b) \Pr(e) \Pr(a|b, e) \Pr(j|a) \Pr(m|a)$$

- Logic-Based AI Under Uncertainty
- Probabilistic Reasoning
  - Conditional Independence
  - Bayesian Networks
  - Semantics of Bayesian Networks
  - Constructing a Bayesian Network
  - Exact Inference in Bayesian Networks
  - ***Approximate Inference in Bayesian Networks***

Monte Carlo Algorithms

- **Monte Carlo algorithms** are randomized sampling algorithms used to estimate quantities that are difficult to calculate exactly
  - E.g., samples from the posterior probability of a Bayes network
- **Pros**
  - The accuracy of the approximation depends on the number of samples generated
  - You can get arbitrarily close to the true probability distribution with enough samples
  - Is used in many branches of science
- **Cons**
  - Difficult to understand how the variables interact
  - Computationally intensive

Sampling from Arbitrary Distributions

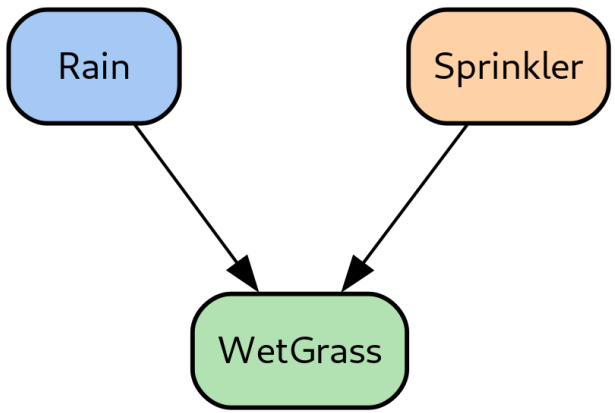
- **Goal**: Sample from a discrete or continuous probability distribution
- **Solution**
  - Start with uniform random numbers $r \in [0, 1]$
  - Construct CDF (cumulative distribution function) $F(x)$
    - $F(x) = \Pr(X \leq x)$
  - For discrete distributions:
    - Create table of outcomes and cumulative probabilities
    - Find smallest outcome where $F(x) > r$
  - For continuous distributions:
    - Use inverse transform: $x = F^{-1}(r)$, e.g.,

$$F(x) = 1 - e^{-\lambda x} \rightarrow x = F^{-1}(r) = -\frac{1}{\lambda} \ln(1 - r)$$

    - If F^{-1} has no closed form, use numerical methods

Sampling Bayesian Network Without Evidence

- **Goal**: Generate events from a Bayesian network without evidence (prior sampling)
- **Solution**
  - Sample variables in topological order (to ensure parents have values)
  - Source nodes have known unconditional probability distribution
    - E.g., $\Pr(Rain) = 0.5$
  - Conditional variable's probability distribution depends on parent's values
    - E.g., $\Pr(WetGrass|Rain = T) = 0.1$
  - Implement Bayesian network semantics, representing joint probability:



$$f_{PS}(x_1, ..., x_n) = \prod_{i=1}^{n} \Pr(x_i|parents(X_i))$$

where PS means "Prior Sampling"

Consistency of Sampling

- **Consistency of estimation**: distribution from prior sampling converges to true probability as $N \to \infty$
- If N_{PS} is the number of times event $x_1, ..., x_n$ occurs:

$$\lim_{N \to \infty} \frac{N_{PS}(x_1, ..., x_n)}{N} = \Pr(x_1, ..., x_n)$$

- Estimate probability using:

$$\Pr(x_1, ..., x_n) \approx \frac{N_{PS}(x_1, ..., x_n)}{N}$$

  - Converges with rate $\frac{1}{\sqrt{N}}$

Rejection Sampling

- **Rejection sampling** is a method for sampling from a hard-to-sample distribution
- **Goal**: Compute $\Pr(X = x|E = e)$ when evidence e is rare
  1. Generate samples from the prior distribution
     - Estimate $\Pr(x, e)$
  2. Reject samples not matching evidence, i.e., $X \wedge E \neq e$
     - Remaining samples $X \wedge E = e$ estimate $\Pr(X, E = e)$
  3. Count occurrences of $X = x$ in remaining samples $X \wedge E = e$
     - Estimate $\Pr(X = x|E = e)$
- **Example**:
  - You want to estimate $\Pr(Rain|Sprinkler = T)$
  - Sample 100 times
    - You get 73 samples with $\neg Sprinkler$ and they are rejected
    - You are left with 27 samples with $Sprinkler$
    - Out of them only 8 have $Rain$ and 19 have $\neg Rain$
  - Thus:

$$\Pr(Rain|Sprinkler) = \frac{8}{27}$$

Rejection Sampling: Pros and Cons

- **Pros**
  - Consistent estimate (converges to true value as samples increase)
- **Cons**
  - Many samples are rejected, depending on rarity of $\Pr(E = e)$
  - Fraction of samples matching evidence e decreases exponentially with more evidence variables
    - Not suitable for complex systems
  - Difficult with continuous variables, as $\Pr(E = e)$ is theoretically 0 (limited by floating-point precision)

Importance Sampling

- Importance sampling:
  - Draw samples from an easier distribution $Q(X)$
  - Weight each sample by its importance weight $w = \frac{P(X)}{Q(X)}$
  - Estimate desired probability by averaging weighted samples:

$$E[f(X)] \approx \frac{1}{N} \sum_{i=1}^{N} w_i f(X_i)$$

- For Bayesian networks use the network's structure to define $Q(X)$ that respects observed evidence
- **Example**:
  - Estimating $P(A|E = e)$ where E is rare
  - Standard sampling might never see $E = e$
  - Importance sampling focuses samples near $E = e$ with appropriate reweighting
  - Analogy: Like rebalancing a biased survey by giving underrepresented groups higher weights
- **Pros**
  - Reduces variance and improves efficiency of inference over rejection

Markov Chain Monte Carlo

- **Purpose**: Approximate inference method for Bayesian networks when exact inference is computationally hard
  - Markov Chain Monte Carlo (MCMC) are different than rejection and importance sampling
  - Instead of generating each sample as independent, make a random change to the preceding sample
- Construct a Markov Chain from the Bayesian network and the sampling technique (e.g., Gibbs sampling, Metropolis-Hastings sampling)
  - Markov Chain:
    - Is used as process to generate a sequence of states
    - Has initial state and transition matrix $\Pr(\underline{\boldsymbol{x}} \rightarrow \underline{\boldsymbol{x}}')$
    - After t steps, probability of being in state $\underline{\boldsymbol{x}}$ is $\pi_t(\underline{\boldsymbol{x}})$
    - Stationary distribution is achieved when $\pi_t(\underline{\boldsymbol{x}}) = \pi_{t+1}(\underline{\boldsymbol{x}})$
- Under certain conditions (ergodicity, aperiodicity) the stationary distribution is the posterior distribution for the non-evidence variables conditioned on the evidence

Markov Chain Monte Carlo: Mixing

- **Mixing** describes how quickly a Markov chain forgets its starting point and explores the whole state space efficiently
  - A well-mixed chain:
    - Moves between different high-probability regions often
    - Has low correlation between successive samples
  - Poor mixing:
    - Chain gets stuck in one mode for a long time
    - Leads to biased estimates and high variance
- **Example**
  - If sampling from a bimodal distribution:
    - "Poor mixing" means the chain stays in one peak
    - "Good mixing" jumps between both peaks to reflect the true posterior

Gibbs Sampling in Bayesian Networks

- Special case of Markov Chain Monte Carlo (MCMC) method that samples one variable at a time
- **Algorithm**:
  - Start with an initial complete assignment to all non-evidence variables
  - Keep evidence variables fixed at observed values
  - For each non-evidence variable X_i:
    - Sample X_i from $P(X_i|\text{MB}(X_i))$ where $\text{MB}(X_i)$ is the Markov blanket, i.e., parents, children, spouse of a node
- **Example**:
  - Weather network: $P(\textit{Cloudy}|\textit{Sprinkler}, \textit{Rain}, \textit{WetGrass})$
  - Fix $\textit{WetGrass} = \textit{true}$, $\textit{Sprinkler} = \textit{true}$
  - Sample *Cloudy* and *Rain* iteratively
- **Pros**
  - Simple to implement for any Bayesian network
  - Handles large, complex graphs with local updates
- **Cons**
  - Can mix slowly if variables are highly correlated

Metropolis–Hastings Sampling

- More general Markov Chain Monte Carlo method than Gibbs sampling
- **Algorithm**
  - Start at a current state $\boldsymbol{x}$
  - Propose a new state $\boldsymbol{x}'$ from a proposal distribution $q(x'|x)$, e.g.,
    - With 95% probability Gibbs sampling
    - Otherwise use importance sampling
  - Compute the acceptance probability:
    - $A(x, x') = \min(1, \frac{\pi(x')q(x|x')}{\pi(x)q(x'|x)})$
  - Move to x' with probability $A(x, x')$, otherwise stay at x
- **Intuition**
  - Propose local moves
  - Accept if they lead to higher probability, or sometimes accept lower-probability states to explore
  - Balances exploration and exploitation to avoid getting stuck in local modes
- **Pros**
  - Very flexible: works with any proposal distribution
  - Can handle high-dimensional spaces