Uncertainty In Artificial Intelligence
Theory

Christian Rossi

Academic Year 2023-2024

Abstract

The topics of the course are:

- Uncertainty sources that affect models: typology, issues, and modeling approaches.
- Measure-based uncertainty modeling.
- Logic-based uncertainty modeling.
- Fuzzy models: fuzzy sets, fuzzy logic, fuzzy rules, motivations for fuzzy modeling, tools for fuzzy systems, design of fuzzy systems, applications.
- Bayesian networks: basics, design, learning, evaluation, applications.
- Hidden Markov Models: basics, design, learning, evaluation, applications.
- Applications: motivations, choices, models, case studies.

Contents

CHAPTER 1

Introduction

1.1 Definition

Definition (*Uncertainty*)**.** Uncertainty pertains to epistemic situations that involve imperfect or unknown information. It is relevant to predictions of future events, existing physical measurements, or the unknown.

Uncertainty can manifest in partially observable or stochastic environments, as well as stemming from ignorance, inaction, or a combination of both. This concept of uncertainty is prevalent across a wide array of disciplines, such as insurance, philosophy, physics, statistics, economics, finance, medicine, psychology, sociology, engineering, metrology, meteorology, ecology, and information science.

Uncertainty represents a state of incomplete knowledge, characterized by the impossibility of precisely defining the current state, a future result, or multiple potential outcomes. This underscores the connection between uncertainty and the necessity to describe an aspect of reality.

1.2 Modelling

Modeling forms the foundation of our existence. Our interaction with the world relies on models that interpret data from sensors, yielding knowledge and guiding actions. Moreover, modeling is the means by which we can represent entities in a computer and facilitate reasoning about them.

Definition (*Model*)**.** A model is a representation of an entity, defined for a particular purpose. A model includes only the relevant aspects of the modeled entity. It's important to note that a model inherently differs from the entity being modeled, giving rise to various forms of uncertainties within the modeling process.

Every Artificial Intelligence application relies on models, which can be either defined by someone or learned. These models are represented in various ways, but they commonly struggle with uncertainties, primarily concerning their inputs.

1.3 Uncertainty classification

Uncertainty types Uncertainty can be categorized into two primary types:

- *Epistemic uncertainty*: this arises from factors that, in theory, could be known but are not known in practice. This can result from inaccurate measurements, overlooked model effects, or intentional data concealment. Epistemic uncertainty, also termed systematic uncertainty, is, in principle, reducible by enhancing the model.
- *Aleatoric uncertainty*: this pertains to unknown unknowns that vary each time the same experiment is conducted. Aleatoric uncertainty, also referred to as statistical uncertainty, can only be described using statistical information. It may also be influenced by data acquisition and processing methods. In general, it is present when the model lacks comprehensive coverage.

Uncertainty sources The sources of uncertainty can be classified into several categories:

- *Parameter uncertainty*: arising from model parameters with exact values unknown to experimentalists and beyond experimental control, or whose values cannot be inferred through statistical methods.
- *Parametric variability*: stemming from the variability in input variables of the model.
- *Structural uncertainty*: also referred to as model inadequacy, model bias, or model discrepancy, this uncertainty arises from a lack of knowledge about the problem.
- *Algorithmic uncertainty*: alternatively known as numerical uncertainty or discrete uncertainty, this type results from numerical errors and approximations in the implementation of the computer model.
- *Experimental uncertainty*: commonly known as observation error, this uncertainty arises from variations in experimental measurements.
- *Interpolation uncertainty*: originating from a lack of variable data collected from computer model simulations and/or experimental measurements.

1.4 Uncertainty modelling

The choice of an uncertainty model is contingent upon the type of uncertainty, its origins, and the available information regarding uncertainty. It primarily pertains to the characterization and quantification of uncertainty. The conceivable models for representing uncertainty include statistical, logical, and cognitive models.

Artificial Intelligence and Machine Learning technologies are grounded in models that incorporate uncertainty models of various types. These models are crucial not only for constructing efficient models but also for defining learning models capable of addressing complex scenarios and evaluating the quality of learned or developed models.

Uncertainty quantification Now, regarding uncertainty quantification, there are two principal categories of problems:

- *Forward propagation of uncertainty*: in this approach, various sources of uncertainty are propagated through the model to predict the overall uncertainty in the system's response. This process entails:

  - Evaluating low-order moments of the outputs (mean and variance).
  - Assessing the reliability of the outputs.
  - Determining the complete probability distribution of the outputs.

  This methodology is related to the principles applied in Bayesian networks and graphical models.

- *Inverse assessment of model and parameter uncertainty*: in this scenario, model parameters are calibrated concurrently using test data. Given experimental measurements of a system and results from its mathematical model, inverse uncertainty quantification estimates the discrepancy between the experiment and the mathematical model (bias correction). It also determines the values of any unknown parameters in the model (parameter calibration).

Models classification As for the classification of models used in Artificial Intelligence, they can be broadly categorized into three main types:

- *Symbolic models*: elements of these models are expressed as terms related to entities to be modeled. The state of the world is represented by facts articulated in formal languages closely resembling natural languages.
- *Sub-symbolic models*: in these models, elements are expressed through code.
- *Black-box models*: these models are primarily seen as computational tools for mapping inputs to outputs, with their internal workings not explicitly considered.

In the context of symbolic models, a fact is true in a model if there is enough evidence to support it. The only facts considered truly accurate are those true by definition.

1.5 Ignorance Management

There are numerous potential sources of ignorance when reasoning in the real world:

- Insufficient data.
- Biased data: data collected by sensors affected by errors.
- Variable data: data collected by imprecise sensors.
- Reliability of data.
- Fuzziness.
- Reliability of the model: depends on the model design, implementation, and parametrization.
- Incompleteness of the model.

> **Example:**
> Let's consider the sentence "The elephant weighs 2 tons". This can be interpreted in various ways, each slightly different:
>
> - The elephant weighs exactly 2 tons.
> - The elephant weighs 2 tons ±10 kg, given the resolution of the weight scales of the instrument.
> - The elephant weighs approximately 2 tons, but we cannot say anything more precise.
> - We are not sure about any previous sentence because we do not have enough evidence.

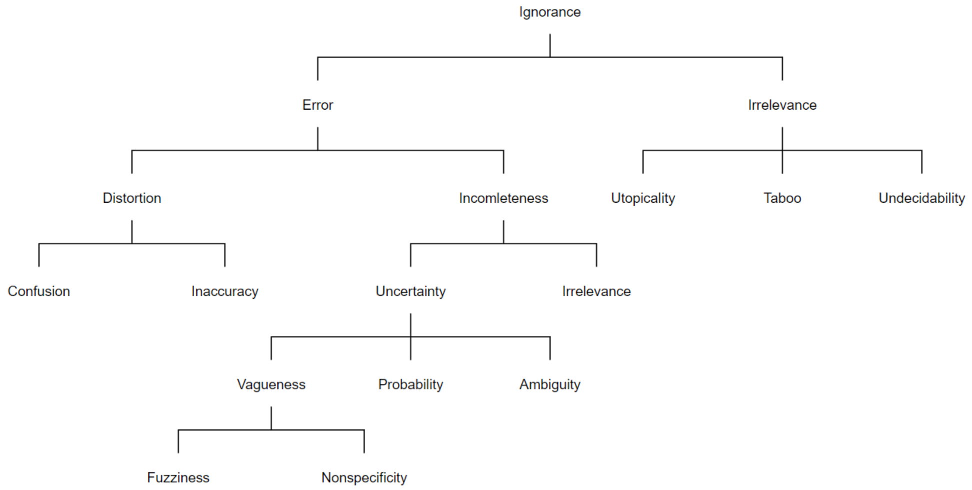


Figure 1.1: Smithson's taxonomy of ignorance and uncertainty

To model ignorance, it is often decided to associate measures of certain aspects. Let's distinguish between two aspects:

- The type of representation: numbers, labels, intervals, etc.
- The represented ignorance that we would like to model, such as probability, reliability, subjective evaluation, etc.

Probability Probability is represented with numbers between zero and one, and a well-established set of rules and properties are associated with its management. For example:

- The sum of probabilities should equal one.
- The probability a posteriori of a hypothesis h_i given some evidence e is given by the Bayes theorem:

$$\mathrm{P}(h_i \mid e) = \frac{\mathrm{P}(e \mid h_i)\mathrm{P}(h_i)}{\mathrm{P}(e)}$$

Probability has been used in applications like MYCIN, one of the first expert systems designed for diagnosing blood illnesses. MYCIN modeled certainty by considering two numerical factors:

- Measure of increased Belief:

$$\mathrm{MB} = \frac{\mathrm{P}\left(\frac{h}{e}\right) - \mathrm{P}(h)}{1 - \mathrm{P}(h)}$$

- Measure of decreased Disbelief:

$$\mathrm{MD} = \frac{\mathrm{P}(h) - \mathrm{P}\left(\frac{h}{e}\right)}{\mathrm{P}(h)}$$

The measure of a statement is given by the certainty factor:

$$\mathrm{CF} = \mathrm{MB} - \mathrm{MD} \in [-1; 1]$$

A key hypothesis for this solution is that the numbers given as MB and MD are not statistical probabilities but subjective probabilities, provided by different experts and combined using rules (which can introduce ambiguity).

In comparison to probabilities, linguistic terms are less ambiguous than numbers. Using a limited set of labels, it is possible to associate subjective evaluations with statements, making it relatively easy to achieve consensus on subjective judgments. A computational mechanism is then needed to define how to combine labels. This is achieved through the use of fuzzy systems, which represent the truth of a statement in linguistic terms and evaluate its fuzziness.

CHAPTER 2

Fuzzy sets

2.1 History

Fuzzy sets were introduced by Lotfi Zadeh in 1965 as a tool to model approximate concepts. In 1972, the first linguistic fuzzy controllers were implemented. By around 1980, fuzzy sets began to see frequent use worldwide. In the 1990s, there was a massive proliferation of fuzzy controllers in various end-user goods. Today, fuzzy systems form the core of many intelligent devices.

2.2 Fuzzy membership function

A crisp set is defined by a boolean membership function on some property of the considered elements. In contrast, a fuzzy set is a set whose membership function ranges between zero and one.

Definition (*Fuzzy membership function*)**.** A membership function defines a set by specifying the degree of membership of an element from the universe of discourse to the set.

A label is assigned to the set to provide a reference. Fuzzy sets can also be defined with a variable having discrete values.

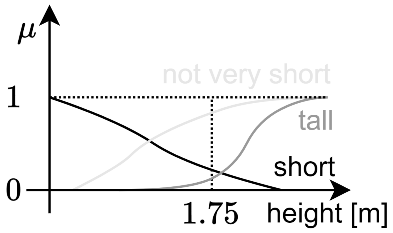


Figure 2.1: Example of a membership function

Membership function definition To define a membership function, we need to consider the following steps based on the purpose of the model and the available data:

1. Select a variable on which the membership function will be defined.

2. Define the range of the variable.
3. Identify the fuzzy sets needed for the application and define the labels.
4. Identify characteristic points for the membership function for each fuzzy set.
5. Define the shape of the membership function.
6. Verify the correctness of the membership function.

The shapes of the membership function can be chosen arbitrarily. The choice of shape affects the smoothness of the transition between two labels (e.g., a horizontal shape results in an immediate transition within intervals).

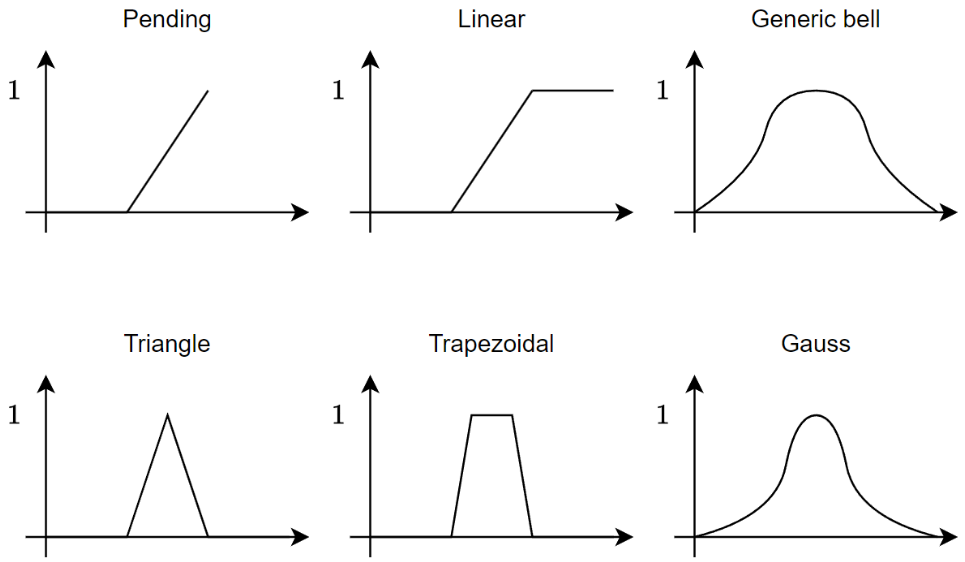


Figure 2.2: Possible shapes for a membership function

Definition (*Frame of cognition*)**.** A set of fuzzy sets that fully covers the universe of discourse is called a frame of cognition.

A frame of cognition has the following properties:

- *Coverage*: each element of the universe of discourse is assigned to at least one granule with membership greater than or equal to zero.
- *Uni-modality of fuzzy sets*: there is a unique set of values for each granule with maximum membership.

Definition (*Fuzzy partition*)**.** A frame of cognition for which the sum of the membership values of each value of the base variable is equal to one is called a fuzzy partition.

Definition (*Function's α-cut*)**.** The α-cut of a fuzzy set is the crisp set of values of x such that $\mu(x) \geq \alpha$:

$$\alpha_\mu(x) = \{x \mid \mu(x) \geq \alpha\}$$

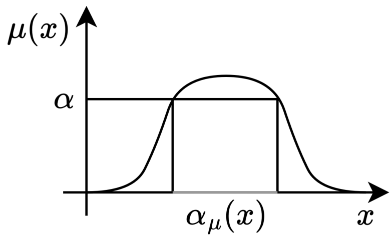


Figure 2.3: Alpha-cut of a membership function

Definition (*Function's support*)**.** The support of a fuzzy set is the crisp set of values x such that $\mu_f(x) > 0$.

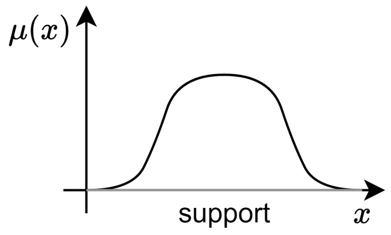


Figure 2.4: Support of a membership function

Definition (*Function's height*)**.** The height h_f of a fuzzy set f on the universe X is the highest membership degree of an element of X in the fuzzy set:

$$h_f(X) = \max_{x \in X} \mu_f(x)$$

A fuzzy set is considered normal if, and only if, $h_f(X) = 1$.

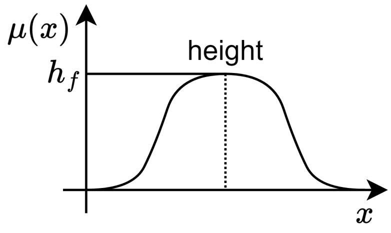


Figure 2.5: Height of a membership function

Definition (*Convex set*)**.** A fuzzy set is convex if and only if

$$\mu[\lambda x_1 + (1-\lambda)x_2] \geq \min[\mu(x_1), \mu(x_2)]$$

for any $(x_1, x_2) \in \mathbb{R}$ and any $\lambda \in [0,1]$.

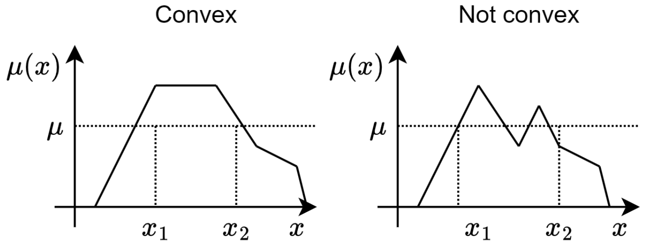


Figure 2.6: Graphical difference between a convex and a not convex set

The available operations on fuzzy sets encompass:

- *Complement*: $\mu_{\bar{f}}(x) = 1 - \mu_f(x)$.
- *Union*: $\mu_{f_1 \cup f_2}(x) = \max[\mu_{f_1}(x), \mu_{f_2}(x)]$.
- *Intersection*: $\mu_{f_1 \cap f_2}(x) = \min[\mu_{f_1}(x), \mu_{f_2}(x)]$.

CHAPTER 3

Fuzzy logic

3.1 Introduction

Logic is a tool that has been employed for thousands of years to formally represent knowledge. There are various types of logic, including:

- *Propositional logic*: assigning truth values to propositions.
- *First-order logic*: assigning truth values to predicates, involving variables and quantifiers.
- *Second-order logic*: dealing with predicates of predicates.

These types of logic operate in a binary fashion. It's worth noting that the specific meanings of terms within these logics are not inherently defined within the formalism itself, and such definitions are not necessary for the logic to function effectively.

3.2 Propositional logic

Propositional logics revolve around propositional operators that can be applied to one or more propositions to generate new propositions. The primary focus lies on the truth value of propositions and how these truth values are combined.

Definition (*Truth functional logic*)**.** A logic is considered truth functional if the truth value of a compound sentence depends solely on the truth values of the individual atomic sentences, without regard to their meaning or structure. For such a logic, the critical question regarding propositions is the range of truth values they may assume.

In classical, Boolean, or two-valued logic, each proposition is either true or false, and no other characteristic of the proposition is considered relevant. The fundamental operators in propositional logics include conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$).

3.3 First order predicate logic

First-order logic extends propositional logic by introducing the capability to define predicates involving variables. It also introduces the existential ($\exists$) and universal ($\forall$) quantifiers. In predicate logics, it is possible to deduce the truth value of a proposition through inferential mechanisms, such as Modus Ponens.

> **Example:**
> Given the sentences: "All man are mortal" and "Socrates is a man" we can infer that "Socrates is mortal".

Inference is employed to model a mental mechanism that allows us to store a reduced amount of information and establish a process for deriving additional information from existing information to deal with everyday situations.

Definition (*Knowledge*)**.** The combination of information and potential relationships constitutes what we refer to as knowledge.

3.4 Many-valued logics

Aristotle raised questions about the suitability of classical logic as a knowledge representation tool. For example, classical logic struggles to determine the truth value of a proposition in the context of the future. To address this, a third value (e.g., 0.5) can be introduced to represent undefined situations, leading to the concept of three-valued logic. This concept can be extended to infinite-value logics that consider a continuum of truth values between zero and one.

> **Example: Logic L1, Łukasiewicz(1930)**
> In this type of infinite-value logic, the main rules include:
>
> - $\mathrm{T}(\neg a) = 1 - \mathrm{T}(a)$.
> - $\mathrm{T}(a \wedge b) = \min(\mathrm{T}(a), \mathrm{T}(b))$.
> - $\mathrm{T}(a \vee b) = \max(\mathrm{T}(a), \mathrm{T}(b))$.
> - $\mathrm{T}(a \implies b) = \min(1, 1 + \mathrm{T}(b) - \mathrm{T}(a))$.
> - $\mathrm{T}(a \Leftrightarrow b) = 1 - |\mathrm{T}(a) - \mathrm{T}(b)|$.

These innovations brought about a shift in society, where things are no longer categorically stated as true or false. Probability (Kolmogorov, 1929) and stochasticity (Markov, 1906) became the preferred ways to represent this new approach to science and life.

Differences with classical logic The differences between classical logic (L2) and many-valued logic (L1) include:

- L1 being isomorphic to fuzzy set theory, with standard operators, while classical logic L2 is isomorphic to set theory.
- Tautologies, which are true by definition and used to prove theorems in classical logic L2, may not be valid in L1. For example, the third excluded law ($\mathrm{T}(a \vee \neg a) = 1$) and the non-contradiction law ($\mathrm{T}(a \wedge \neg a) = 0$) are not valid in L1.

In classical logic, the sentence "I'm a liar" would be considered a paradox if we assign meaning to the term "liar," as no formula can have the same truth value as its negation. However, this may not be the case in many-valued logics. In Łukasiewicz logic, for instance, it's possible for a sentence to have a truth value of 0.5, and its negation to also have a truth value of 0.5, making the proposition consistent with the axioms and not a paradox.

3.5 Fuzzy logic

Fuzzy logic is an infinite-valued logic with truth values ranging from 0 to 1, where propositions are expressed in the form of "A is L," where:

- A is a linguistic variable.
- L is a label representing a fuzzy set.

Formally, a linguistic variable is defined by a 5-tuple (X, T(X), U, G, M), where:

- X is the name of the variable.
- $T(X)$ is the set of term for X, each corresponding to a fuzzy variable denoted by $T(X)$ and ranging on U.
- U is the universe of discourse defined on a base variable u.
- G is the syntactic rule used to generate the interpretation X of each value u.
- M is the semantic rule used to associate to X its meaning.

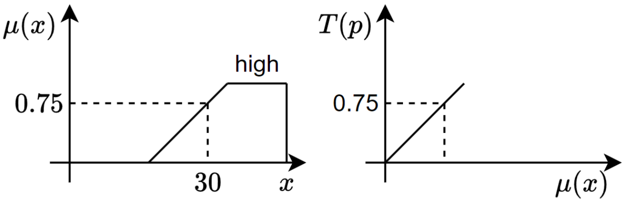
> **Example:**
> Let's define a linguistic variable for age as follows:
>
> - X is a linguistic variable labelled "age".
> - U $= [0, 100]$.
> - T(X) $= \{old, middle - aged, young, \dots\}$.
> - u $= [0, +\infty]$.
> - M represents the definition in terms of fuzzy sets for the values of X.
> - G is responsible for the fuzzy matching interpretation of u.

Now that we have defined the linguistic variable, it is possible to express a simple proposition as "p: X is F", where:

- X is a linguistic variable.
- F is the label of a fuzzy set defined on U, representing a fuzzy predicate.
- $\mu_{\mathrm{F(x)}}$ is the membership function defining F, and it is interpreted as the truth value for the proposition p (T(p) $= \mu_{\mathrm{F(x)}}$).

Hence, the truth value of the proposition P is a fuzzy set defined on the interval $[0, 1]$.

Example:
Consider the simple proposition "p: temperature is high", with X representing temperature and F representing high. We can determine the truth value of this proposition using the graph of the membership function as shown below:

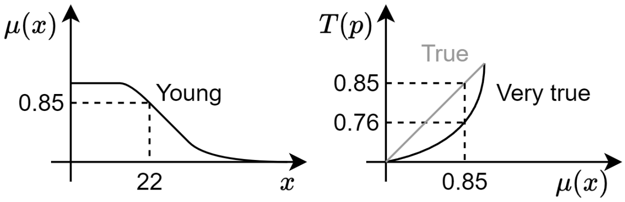


Consequently, the truth value of the given proposition is 0.75.

It is also possible to express qualified, non-conditional propositions using the syntax "p: (X is F) is S", where:

- S is a fuzzy truth qualifier.
- F is a fuzzy set.
- p is truth qualified.

Example:
Consider the conditional proposition "p: age of Tina is young is very true", where X represents age, F represents young, and S represents very true. To determine the truth value of this proposition, we can refer to the graph of the membership function as shown below:



In fuzzy logic, fuzzy modifiers are employed to adjust the truth values of propositions. These modifiers can be categorized into two main types:

- Strong ($m(a) \leq a \; \forall a \in [0 \ldots 1]$): these modifiers strengthen the predicate, leading to a reduction in the truth value of the proposition.
- Weak($m(a) \geq a \; \forall a \in [0 \ldots 1]$): these modifiers weaken the predicate, resulting in an increase in the truth value of the proposition.

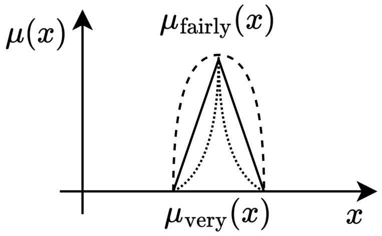
The key properties of fuzzy modifiers include:

- $m(0) = 0$ and $m(1) = 1$.
- m is a continuous function.
- If m is a strong modifier, then m^{-1} is a weak modifier, and vice versa.
- When combining another modifier g with m, and vice versa, the resulting composition is also a modifier. If both m and g are strong (or weak), their composition is also strong (or weak).

Example:
The sentence "x is young" can be represented as "(x is young) is true". This sentence can be modified using fuzzy modifiers in the following ways:

- x is very young is true.
- x is young is very true.
- x is very young is very true.

Graphically, we can visualize the modified membership function as follows:



Here, we have:

- $\mu_{\text{very}}(x) = \mu_a(x)^2$.
- $\mu_{\text{fairly}}(x) = \mu_a(x)^{\frac{1}{2}}$.

3.6 Inference rules

Definition (*Inference rule*)**.** An inference rule is a model, essentially defining a mapping from input to output. These rules are utilized to represent inferential relationships among various pieces of knowledge.

We will primarily focus on forward chaining rules, which typically have the structure "IF ⟨antecedent⟩ THEN ⟨consequent⟩", where:

- ⟨antecedent⟩ is a set of clauses related by logical operators.
- ⟨consequent⟩ is a set of clauses related by logical operators.

In these inference rules, the clauses can be either propositions (sequences of symbols) or patterns (sequences of symbols and variables).

Inference rules play a crucial role in the implementation of Knowledge-Based Systems, with Expert Systems standing out as highly successful applications in the field of Artificial Intelligence. Expert Systems are carefully designed to replicate or enhance human expertise in problem-solving. The process of Knowledge Acquisition, which is inherently intricate, leads to the development of rule-based systems that are executed on computer platforms.

Information generation A system can generate new information by following these steps:

1. Pattern matching: identify the rules with antecedents that match the known facts stored in the fact base. These rules can be considered for activation, provided the corresponding variables are assigned.

2. Rule selection: among the rules identified through pattern matching (candidate rules), select the ones to be activated.

3. Rule activation: assert the consequents of the selected rules in the fact base.

Example:
Let's consider a rule base consisting of the following four rules:

1. IF X croaks AND X eats flies, THEN X is a frog.
2. IF X chirps AND X sings, THEN X is a canary.
3. IF X is a frog, THEN X is green.
4. IF X is a canary, THEN X is yellow.

Now, let's observe the following facts in the fact base:

- Fritz croaks.
- Fritz eats flies.

From rule 1 and the facts (a and b), we can add the following fact to the fact base:

Fritz is a frog

With the updated fact base, we can use rule 3 to deduce the fact:

Fritz is green

3.7 Fuzzy rules

Definition (*Fuzzt rule*)**.** A fuzzy rule is a rule whose clauses have the form "V is L", where V is a linguistic variable, and L is a label representing a value for V associated with a fuzzy set. Each of these clauses is referred to as a linguistic clause.

Often, clauses in the antecedent are implicitly connected by the AND operator, which is not explicitly written. The antecedent is typically compared to facts represented as values of base variables corresponding to the linguistic variables. The consequent can be one of two types:

- *Linguistic rules*: the consequent is a conjunction of linguistic clauses. These rules can be viewed as a mapping between the interpretation of an input configuration and a symbolic description of the desired output. The general formula is:

$$\text{IF (A is } L_{A_i}\text{) AND (B is } L_{B_k}\text{) AND } \ldots \text{ THEN (U is } L_{U_m}\text{) AND } \ldots$$

- *Model rules*: associate a model with the linguistic interpretation of its applicability conditions. This can be seen as a mapping between the interpretation of an input configuration and a model that is applied to the input real values to obtain the output. The general formula is:

$$\text{IF (A is } L_{A_n}\text{) AND (B is } L_{B_k}\text{) AND } \ldots \text{ THEN U is } f\text{(A,B)}$$

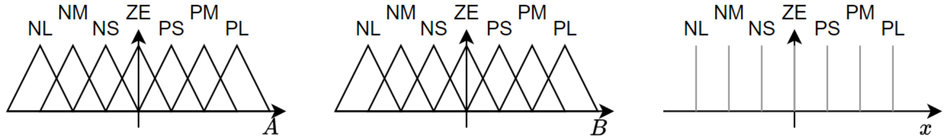
Fuzzy rules usage The steps for using fuzzy rules are as follows:

1. Input matching.
2. Combining matching degrees.
3. Combining with rule weight, if present.
4. Aggregating output from different rules.
5. Optionally, defuzzification of the output.

When defuzzifying the output, various operators can be considered in addition to the weighted mean. These operators include the centroid, bisector, average of maxima, the lowest maximum, the highest maximum, center of the highest area, and more. The choice of operator can impact the system's output and the level of optimization.

Example:
Let's consider the following scenario:

- Two input variables, A and B, each with fuzzy partitions distributed equally from Negative Large to Positive Large.
- One output variable U (equally distributed fuzzy set) from Negative Large to Positive Large. The fuzzy sets are all singletons.



To define the rules of the rule base, along with their weights, we have the following rules:

1. IF A is PL AND B is PS THEN X is PM (weight 1).
2. IF A is PM AND B is PS THEN X is PS (weight 0.5).
3. IF A is PL AND B is PM THEN X is PM (weight 1).

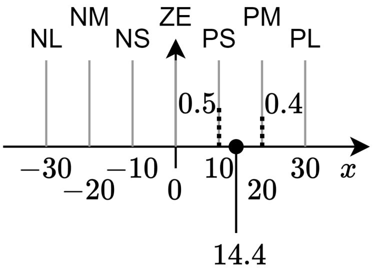
Now, let's set A to 22 and B to 140. The steps used to calculate the output value are as follows:

1. For the first step, we need to check the corresponding truth value for each label:
   - (A is PL) has a truth value of 0.2.
   - (B is PS) has a truth value of 0.6.
   - (A is PM) has a truth value of 0.8.
   - (B is PM) has a truth value of 0.4.
2. To consider the degree of truth of each predicate, we simply take the minimum between the two values (due to the AND operator). Thus, we get:
   - 0.2 for the first rule.

- 0.6 for the second.
- 0.4 for the third.

3. Now we have to consider the rule weight. To do this, we select the minimum between the previously calculated value and the weight value. So, the final values for the consequents are as follows:
   - 0.2 for the first rule.
   - 0.5 for the second.
   - 0.4 for the third.
4. To aggregate the output, we take the maximum value when we have a repeated expression. In this case, we obtain that:
   - (X is PM) has a truth value of 0.4.
   - (X is PS) has a truth value of 0.5.

   This result can be visualized graphically by cutting the initial graph:



5. Finally, to defuzzify the result and obtain a numerical value, we can use a simple weighted mean:

$$U = \frac{10 \cdot 0.5 + 20 \cdot 0.4}{0.5 + 0.4} = 14.44$$

Consider the same variables. This time, we are using different rules:

1. IF A is PL and B is PS THEN X is $A + 2B$.
2. IF A is PM and B is PS THEN X is $A + 3$.
3. IF A is PL and B is PM THEN X is $A + B$.

All the models used in these rules are linear. Pattern matching is the same as in the previous example, leading to the following degree of truth for the rules: the first one has a value of 0.2, the second has a value of 0.5, and the third has a value of 0.4. For the output aggregation, we again use the weighted mean and apply it to the initial values of $A = 22$ and $B = 140$:

$$U = \frac{0.2 \cdot (A + 2B) + 0.5 \cdot (A + 3) + 0.4 \cdot (A + B)}{0.2 + 0.5 + 0.4} = 125.18$$

3.8 Fuzzy system design

The structured approach to develop a fuzzy system for solving specific problems consist of:

1. Problem definition.
2. Parametrization of the model: concepts.
3. Mapping definition: rules.
4. Implementation.
5. Testing.

In the problem definition phase of designing a fuzzy system, it's essential to choose the input and output variables and clearly define the goal of the model. Input variables are typically numerical or ordinal variables, making it possible to define fuzzy sets on them. These variables can be categorized as follows:

- Perceived values: these variables come directly from sensors, collected data, or user inputs.
- Computed variables: these are derived from perceived variables through calculations or processing.

The choice of input variables is a design decision, and there aren't inherently best or worst input variables to select. Output variables, on the other hand, depend on the specific needs of the modeler and are the result of the fuzzy model's computations. The goals of the fuzzy model should be defined in advance and should guide the design process.

System parametrization During the system parametrization phase, several key decisions need to be made:

- *Selection of membership functions*: this involves choosing appropriate membership functions for all variables. Membership functions can be defined by a single expert based on objective evaluation or interviews, by multiple experts for increased reliability, or even by automatic systems working on data (e.g., using Neural Networks). The number of membership functions for each variable typically ranges from three to seven. It's important to ensure that every point within the range of input variables is covered by at least one fuzzy set that participates in at least one rule. The boundaries should be covered with the maximum value to avoid any gaps in coverage.
- *Selection of inferential mechanism*: the inferential engine depends on the operators selected for different parts of the rule-based system:
  - AND: the choice between using the minimum (where the worst degree of matching is the most relevant) or the product (where all degrees of matching are relevant).
  - OR: this involves combining the degree of truth with the rule weight. You can choose between using the maximum or the probabilistic sum.

    - Aggregation of degrees of the same consequent: you can choose between using the maximum (where the best degree is the most relevant) or the probabilistic sum (where all knowledge is considered). You must choose the same implementation used for the OR operator.

- *Selection of fuzzification and defuzzification*: this step involves deciding whether to apply fuzzification (converting crisp input values into fuzzy values) and defuzzification (converting fuzzy output values back into crisp values) in your system.

These decisions are crucial to the performance and behavior of your fuzzy system and should align with the goals and requirements defined in the problem definition phase. The design and definition of fuzzy rules can be carried out using various approaches, and the testing phase helps ensure the effectiveness and reliability of the system.

Rule definition Here are some methods for rule definition:

- *From experience*: rules can be formulated based on the domain knowledge and expertise of human operators or experts who understand the system. These rules are often based on their experience and intuition.
- *From another model*: in some cases, existing models, such as mathematical models or traditional control systems, can be used as a basis for defining fuzzy rules. These models can be adapted into fuzzy rule sets.
- *Machine Learning*: Machine learning techniques, including supervised learning methods, can be used to derive fuzzy rules from data. Algorithms like decision trees, genetic algorithms, or neural networks can automatically generate fuzzy rule sets from training data.
- *Self-tuning techniques*: some systems employ self-tuning mechanisms, like Neural Networks or reinforcement learning algorithms, to adapt and modify fuzzy rules based on ongoing system feedback. These methods can optimize rule sets over time.

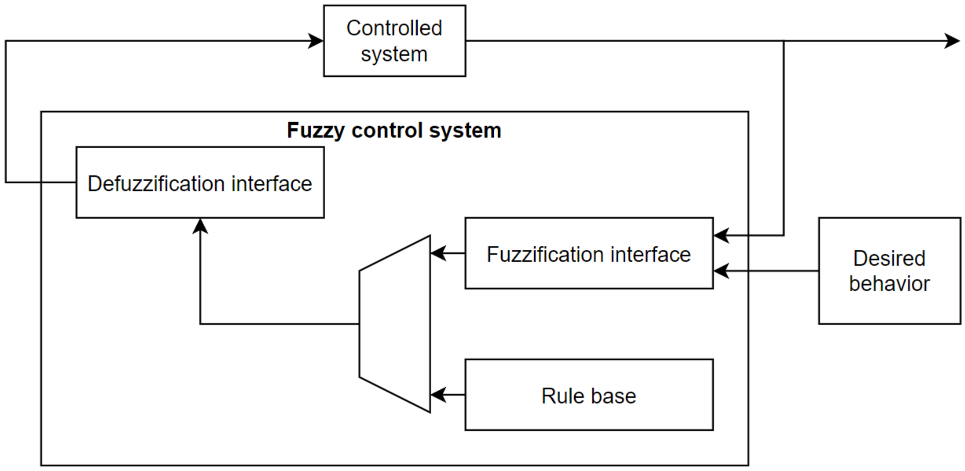
Testing Testing can be conducted using:

- *Dynamic simulation*: dynamic simulation involves running the fuzzy system in a simulation environment to assess its performance under various conditions and scenarios. This allows for a thorough evaluation of how the system responds to different inputs and situations.
- *Static simulation*: static simulation involves testing the fuzzy system with predefined inputs and observing its outputs. This can help evaluate the system's consistency and ensure that it adheres to expected behavior without the need for a real-time dynamic simulation.
- *Direct testing on the process*: in some cases, fuzzy systems can be tested directly on the real process, typically under safe or controlled conditions. This testing approach validates the system's performance in the actual operational environment, which is valuable for systems that control physical processes.

The choice of rule definition and testing methods depends on the specific application, available data, and the complexity of the system. Additionally, it's essential to validate and refine the fuzzy rule set to achieve the desired system behavior and performance.

3.9 Applications of fuzzy systems

Fuzzy control systems are designed to regulate the behavior of other systems, often employing a PID controller where the output relies on the disparity between the desired and observed behavior. The general structure of a fuzzy control system is illustrated in the figure below:



Key attributes of a fuzzy control system include:

- Robustness in the presence of noise.
- Control rules applicable over a broad range.
- Capability to model expert heuristics.
- Smooth action.
- Inherent non-linearity.

Additionally, fuzzy systems have the potential to facilitate flexible, human-like database queries. For instance, fuzzy sets can be employed to formulate queries such as ”Provide the names of individuals who have recently made substantial investments”, thereby imbuing ”recently” and ”a lot” with meaningful interpretations.

Fuzzy systems find applications in various domains within Artificial Intelligence, including Expert Systems, scheduling, and Decision Support Systems.

CHAPTER 4

Evidence theory

4.1 Fuzzy mathematics

Fuzzy numbers are a mathematical concept used to model our perception of approximate values. They are essentially fuzzy sets defined over the set of real numbers. Three important constraints are used to define fuzzy numbers:

1. *Normal fuzzy sets*: fuzzy numbers adhere to the principles of normal fuzzy sets, which allows them to capture the concept of approximate value.
2. *Convex fuzzy sets*: for a fuzzy number to be well-defined, all α-cut intervals should be closed. This is a key constraint for arithmetic operations.
3. *Bounded support*: the support of a fuzzy number (the range where it has significant membership) should be bounded, providing further constraints for their definition.

Fuzzy sets can be used to define various types of numerical representations, including fuzzy numbers, fuzzy intervals, defined intervals, and crisp numbers, as illustrated in the provided figure.

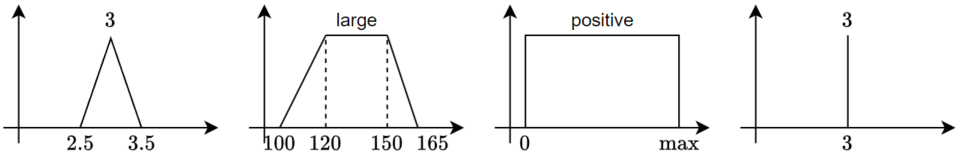


Figure 4.1: Possible representation of numbers

Properties The arithmetic of fuzzy numbers is based on two primary properties:

- *Uniqueness of α-cuts*: each fuzzy number can be fully represented by its α-cuts, which are unique for that number.
- *Closed intervals*: the α-cuts of fuzzy numbers are closed intervals of real numbers, which is crucial for arithmetic operations.

Operators The four main arithmetic operators for fuzzy numbers are defined by combining operations on the intervals (represented by α-cuts) that make up the fuzzy number:

- *Addition*: $[a, b] + [d, e] = [a + d, b + e]$.
- *Subtraction*: $[a, b] - [d, e] = [a - e, b - d]$.
- *Multiplication*: $[a, b] \times [d, e] = [\min(ad, ae, bd, be), \max(ad, ae, bd, be)]$
- *Division*: $[a, b] \div [d, e] = \left[\min\left(\frac{a}{d}, \frac{a}{e}, \frac{b}{d}, \frac{b}{e}\right), \max\left(\frac{a}{d}, \frac{a}{e}, \frac{b}{d}, \frac{b}{e}\right)\right]$ with the condition that $[d, e] \neq [0, 0]$ to avoid division by zero.

These operators allow for performing arithmetic operations on fuzzy numbers, taking into account the uncertainty or fuzziness associated with each value.

> **Example:**
> Given the fuzzy numbers $[1, 3]$ and $[4, 6]$, we have the following operations:
>
> - Addition of fuzzy numbers: the sum is calculated as the sum of the minimum and maximum values in each interval, resulting in
>
> $$[1, 3] + [4, 6] = [1 + 4, 3 + 6] = [5, 9]$$
>
> - Subtraction of fuzzy numbers: the difference is found by subtracting the minimum of the first interval from the maximum of the second and subtracting the maximum of the first interval from the minimum of the second, yielding
>
> $$[1, 3] - [4, 6] = [1 - 6, 3 - 4] = [-5, -1]$$
>
> - Multiplication of fuzzy numbers: the product is determined by taking the minimum and maximum values of the products of corresponding elements, resulting in
>
> $$[1, 3] \times [4, 6] = [\min(1 \cdot 4, 1 \cdot 6, 3 \cdot 4, 3 \cdot 6), \max(1 \cdot 4, 1 \cdot 6, 3 \cdot 4, 3 \cdot 6)] = [4, 18]$$
>
> - Division of fuzzy numbers: the division is obtained by finding the minimum and maximum values of the divisions of corresponding elements, leading to:
>
> $$[1, 3] \times [4, 6] = \left[\min\left(\frac{1}{4}, \frac{1}{6}, \frac{3}{4}, \frac{3}{6}\right), \max\left(\frac{1}{4}, \frac{1}{6}, \frac{3}{4}, \frac{3}{6}\right)\right] = \left[\frac{1}{6}, \frac{3}{4}\right]$$

From fuzzy arithmetic, it is also possible to define fuzzy functions, fuzzy integrals, and fuzzy derivatives. In general, fuzzy numbers are employed to represent approximations.

4.2 Fuzzy measure and probability assignment

Definition (*Borel field*)**.** A field is considered a Borel field if it possesses the property that when all the A_n sets belong to the field, the union and intersection of these sets also belong to the field.

Fuzzy measure A function g defined on a Borel field B within the universe of discourse X is referred to as a fuzzy measure if it satisfies the following properties:

1. $g(\varnothing) = 0$ and $g(X) = 1$.
2. If $A, B \in B$ and $A \subseteq B$, then $g(A) \leq g(B)$.
3. If $A_n \in B$ and $A_1 \subseteq A_2 \subseteq \ldots$ then $\lim_{n\to\infty} g(A_n) = g\left(\lim_{n\to\infty} A_n\right)$.

The concept of a fuzzy measure differs from a classical measure, as it relaxes the requirement of additivity.

Basic probabilistic assignment The basic probabilistic assignment is defined as follows:

- $m : \mathcal{P}(X) \to [0, 1]$.
- $m(\varnothing) = 0$.
- $\sum_{A\in\mathcal{P}(X)} m(A) = 1$.

Here, m provides, for any set A belonging to the power set of X($\mathcal{P}$(X)), an indication of how much the available and relevant evidence supports the notion that a given element belongs to set A. It's important to note that there is no requirement for $m(X)$ to be equal to 1, no necessity for $m(A) \leq m(B)$ when $A \subseteq B$, and no inherent relationship between $m(A)$ and $m(\neg A)$.

4.3 Evidence theory

We aim to establish a measure of evidence for or against a proposition, employing two fuzzy measures: Belief and Plausibility.

Definition (*Belief*)**.** Belief is an estimate of the minimum probability that can be assigned to an element, considering the collected evidence.

Belief is defined as follows:

- $Bel : \mathcal{P}(X) \to [0, 1]$.
- $Bel(\varnothing) = 0$ and $Bel(X) = 1$.
- $Bel(A_1 \cup A_2 \cup \cdots \cup A_n) \geq \sum_j Bel(A_j) - \sum_{j<k} Bel(A_j \cap A_k) + \cdots + (-1)^{n+1} Bel(A_1 \cap A_2 \cap \cdots \cap A_n)$
- $Bel(A) + Bel(\neg A) \leq 1$.

Definition (*Plausibility*)**.** Plausibility is an estimate of the maximum probability that can be assigned to an element, considering the collected evidence.

Plausibility is defined as follows:

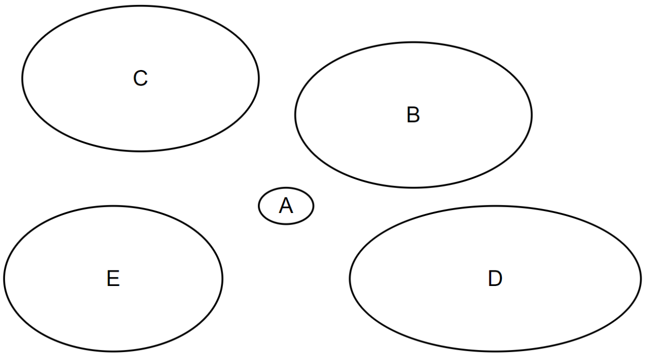
- $Pl : \mathcal{P}(X) \to [0, 1]$.
- $Pl(\varnothing) = 0$ and $Pl(X) = 1$.
- $Pl(A_1 \cap A_2 \cap \cdots \cap A_n) \geq \sum_j Pl(A_j) - \sum_{j<k} Pl(A_j \cup A_k) + \cdots + (-1)^{n+1} Pl(A_1 \cup A_2 \cup \cdots \cup A_n)$
- $Pl(A) + Pl(\neg A) \geq 1$.

Applications Evidence theory finds application when multiple sources of knowledge exist, and the basic probability assignment is distributed across different sets of statements or intervals. In such cases, we can utilize the characteristics of evidence theory to accumulate basic probability assignments and combine them to assess upper and lower bounds for the probability of a single statement. Evidence theory implies that:

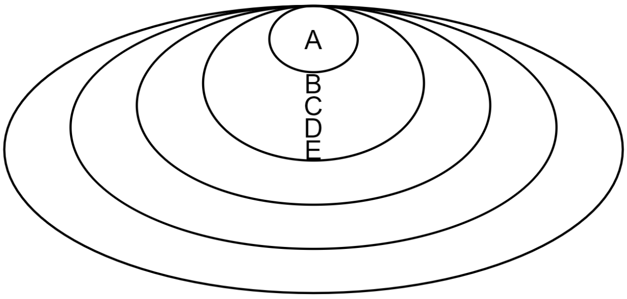
- There's no need to obtain a precise measurement from a knowledge source or experiment if it's not realistic or feasible.
- The principle of insufficient reason is not imposed. Statements can be made about the likelihood of multiple events together without making assumptions about the probabilities of individual events under ignorance.
- The axiom of additivity is not imposed. The measures can have the following properties:
  - Add up to exactly one, which corresponds to a traditional probabilistic representation.
  - Add up to less than one (sub-additive case), indicating incompatibility between multiple sources of information providing conflicting data.
  - Add up to more than one (super-additive), suggesting a cooperative effect between multiple sources of information, such as multiple sensors providing the same information.

Information sources Given five sources (A, B, C, D, E with A as the target) of information, four cases can arise:

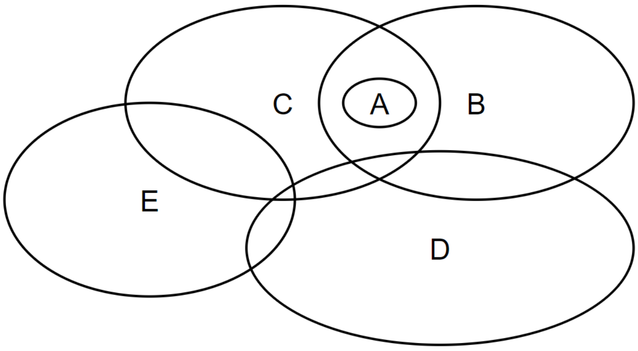
- Conflict: in this scenario, each source provides evidence for disjoint sets.



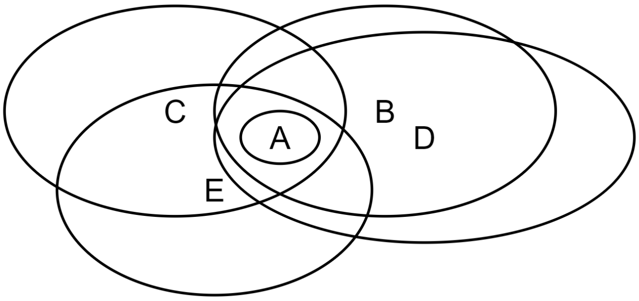
- Consonance: sources provide some evidence on nested sets, converging on the target.



- Arbitrary: in the arbitrary case, each source provides evidence for sets, but only some of them include the target hypothesis.



- Consistent: in this situation, all sources provide evidence for sets that include the same hypothesis.



Combining probability assignment To combine the basic probability assignments, the Dempster rule of combination can be used:

$$m_{1,2}(A) = \frac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1 - K}$$

Where K is the basic probability mass associated with conflict. The role of K in the denominator has the effect of completely ignoring conflict and attributing any probability mass associated with conflict to the null set. The value of this variable is determined by:

$$K = \sum_{B \cap C = O} m_1(B) m_2(C)$$

Example:
Let's consider the discovery of an old painting strongly resembling paintings by Raphael. This discovery raises various questions about the painting's status. We have three questions:

1. Is the discovered painting a genuine painting by Raphael?
2. Is the discovered painting a product of one of Raphael's many disciples?
3. Is the discovered painting a counterfeit?

Assume that two experts conducted careful examinations of the painting and provided us with basic assignments m_1 and m_2. Using the introduced formulas, we can compute the plausibility and belief of all the subsets of hypotheses.

4.4 Possibility and necessity

Definition (*Possibility*)**.** Possibility is another fuzzy measure that operates on sets.

A possibility measure is represented by the function $\Pi : \mathcal{P}(X) \to [0, 1]$, and it adheres to the following properties:

1. $\Pi(\varnothing) = 0$ and $\Pi(X) = 1$.
2. $A \subseteq B \implies \Pi(A) \leq \Pi(B)$.
3. $\Pi(A) = \sup_{x \in A} f(x)$ where $A \subset X$.

A possibility measure can be uniquely defined by a possibility relationship $f : \mathrm{X} \to [0, 1]$ such that:

$$\Pi\left(\bigcup_{i \in I} A_i\right) = \sup_{i \in I} \Pi\left(A_i\right)$$

This makes it possible to define f as $\Pi\left(\{X\}\right)$ for all $x \in X$.

Example:
Consider the set $x = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and $\Pi(x)$, representing the possibility that x is close to the value 8:

x	0	1	2	3	4	5	6	7	8	9	10
$\Pi\left(\{X\}\right)$	0	0	0	0	0	0.1	0.5	0.8	1	0.8	0.5

Now, let's compute $\Pi(A)$, which represents the possibility that A includes an integer close to 8. For a given set $A = \{2, 5, 9\}$, we can calculate its possibility as follows:

$$\Pi(A) = \sup\left[\Pi\left(\{2\}\right), \Pi\left(\{5\}\right), \Pi\left(\{9\}\right)\right] = \sup\left[0, 0.1, 0.8\right] = 0.8$$

Definition (*Necessity*)**.** Necessity is the dual concept to possibility and is defined as follows:

$$\Pi(A) = N(\neg A)$$

It also satisfies the condition:

$$\min\left[N(A), N(\neg A)\right] = 0$$

These two measures, possibility and necessity, are connected by the following relations:

- $\Pi(A) \geq N(A)$.
- $N(A) > 0 \implies \Pi(A) = 1$.
- $\Pi(A) < 1 \implies N(A) = 0$.

Definition (*Confirmation degree*)**.** The confirmation degree is a value that combines both possibility and necessity:

$$C(A) = N(A) + \Pi(A) - 1$$

Negative values of $C(A)$ correspond to a disconfirmation degree.

4.4.1 Summary

It is possible to demonstrate that if the focal set of elements for possibility (those with values of m different from 0) is composed of sets with a single element, then Bel and Pl have the same value. This value is equal to the sum of the probabilities of the elements of the set A to which they are applied, as given by m.

In the probabilistic model, evidence pertains to single elements, while in the possibility model, it can also be associated with sets. Both models have distribution functions, although they are normalized differently: in probabilities, they add up to one, while in possibilities, the maximum value is one.

In possibility theory, ignorance is expressed by assigning all the evidence to the universal set (i.e., anything is possible). In probability theory, on the other hand, it is expressed by distributing a uniform fraction of the evidence to each element (making each element equiprobable).

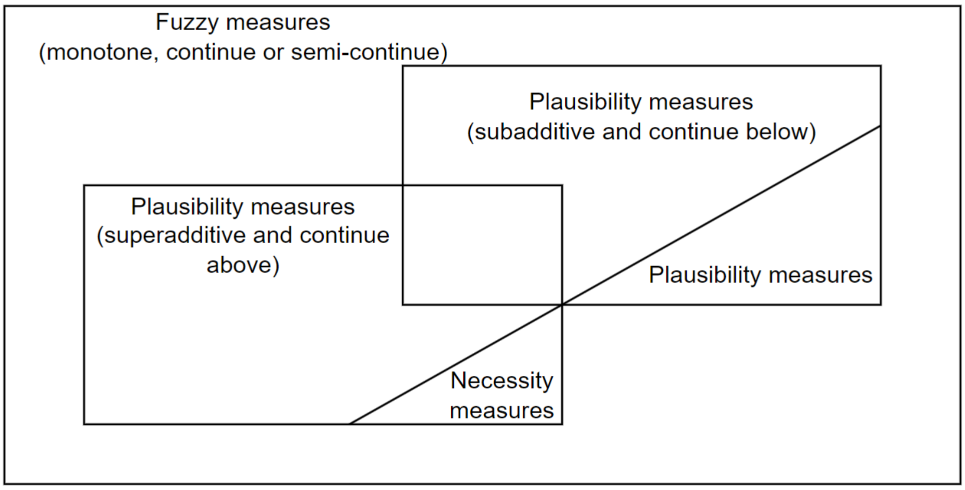

Fuzzy measures
(monotone, continue or semi-continue)
Plausibility measures
(subadditive and continue below)
Plausibility measures
(superadditive and continue above)
Plausibility measures
Necessity measures

Figure 4.2: Inclusion relationships among fuzzy

4.5 Fuzziness measure

The fuzziness measures provide the degree of fuzziness of a fuzzy set. A fuzziness measure is often quantified as the entropy of a fuzzy set.

Definition (*Fuzziness measure*)**.** Given a fuzzy set $A = \{x, \mu_A(x)\}$, the fuzziness measure (entropy) is defined as:

$$d(A) = K \sum_{i=1}^{n} S(\mu_A(x_i))$$

Where $S(x)$ is the Shannon's function:

$$S(x) = -x \ln(x) - (1 - x) \ln(1 - x)$$

Example:
Let's define set A as the set of integers close to ten. We have the following data:

x	7	8	9	10	11	12	13
$\mu_{\mathrm{A}(x)}$	0.1	0.5	0.8	1	0.8	0.5	0.1

The entropy of set A is calculated as:

$$d(A) = 0.325 + 0.693 + 0.673 + 0.501 + 0 + 0.501 + 0.693 + 0.325 = 3.711$$

Now, let's define set B as the set of integers quite close to ten. We have the following data:

x	6	7	8	9	10	11	12	13	14
$\mu_A(x)$	0.1	0.3	0.4	0.7	1	0.8	0.5	0.3	0.1

The entropy of set B is calculated as:

$$d(A) = 4.35$$

It's worth noting that set B exhibits more fuzziness compared to set A, as indicated by the fact that $d(B) > d(A)$.

CHAPTER 5

Probabilistic reasoning

5.1 Basic probability

Definition (*Boolean-valued random variable*)**.** A random variable, denoted as A, is considered a boolean-valued random variable when it represents an event, and there exists a level of uncertainty regarding whether this event will occur.

Definition (*Probability*)**.** The concept of probability associated with A is defined as the proportion of possible outcomes or worlds in which the event represented by A is true.

Theorem 5.1. *The fundamental axioms of probability theory are:*

$$0 \leq \mathrm{P}(A) \leq 1$$

$$\mathrm{P}(A = true) = 1 \ \wedge \mathrm{P}(A = false) = 0$$

$$\mathrm{P}(A \vee B) = \mathrm{P}(A) + \mathrm{P}(B) - \mathrm{P}(A \wedge B)$$

Starting with these axioms, it becomes possible to derive various other formulas in probability theory, such as:

1. $\mathrm{P}(\overline{A}) = 1 - \mathrm{P}(A)$.
2. $\mathrm{P}(A) = \mathrm{P}(A \wedge B) + \mathrm{P}(\overline{A} \wedge B)$

Definition (*Multivalued random variable*)**.** A random variable A is classified as a multivalued random variable of arity k when it can assume one of the values from the set $\{v_1, v_2, v_3, \dots, v_k\}$.

Theorem 5.2. *The axioms for multivalued random variable are:*

$$\mathrm{P}(A = v_i \wedge A = v_j) \qquad i \neq j$$

$$\mathrm{P}(A = v_1 \vee A = v_2 \vee A = v_3 \vee \dots \vee A = v_k) = 1$$

These new axioms enable the derivation of other valuable formulas applicable to multivalued variables:

1. $\mathrm{P}(A = v_1 \vee A = v_2 \vee \dots \vee A = v_i) = \sum_{j=1}^{i} \mathrm{P}(A = v_j)$.

2. $\sum_{j=1}^{k} \mathrm{P}(A = v_j) = 1$.

3. $\mathrm{P}(B \land (A = v_1 \lor A = v_2 \lor \cdots \lor A = v_i)) = \sum_{j=1}^{i} B \land A = v_j$.

4. $\mathrm{P}(B) = \sum_{j=1}^{k} \mathrm{P}(B \land A = v_j)$.

Definition (*Conditional probability*)**.** The conditional probability of event A given event B represents the proportion of possible scenarios where event B is true, and event A is true as well.

Inference can primarily be accomplished using the following rules:

- *Chain rule*: $\mathrm{P}(A \land B) = \mathrm{P}(A|B)P(B)$
- *Bayes theorem*: $\mathrm{P}(A|B) = \dfrac{\mathrm{P}(B|A)\mathrm{P}(A)}{\mathrm{P}(B)}$.
- *Sum rule (marginalization)*: $\mathrm{P}(A) = \sum_b (A \land B = b)$.

Definition (*Idipendent random variables*)**.** Let's assume that A and B are boolean random variables. A and B are considered independent, denoted as $A \perp B$, if and only if:

$$\mathrm{P}(A|B) = \mathrm{P}(A)$$

Definition (*Joint distribution of random variables*)**.** When we have two random variables, A and B, the joint distribution of A and B is represented by $\mathrm{P}(A, B)$ and encompasses the combined distribution of both variables.

We can represent a joint distribution of m binary variables through the following steps:

1. Create a truth table listing all possible combinations of values (a total of 2^m entries).
2. Calculate the probability for each combination.
3. Verify that the sum of all probabilities equals one.

5.2 Probabilistic reasoning

Graphical models are employed to depict the factorization of joint distributions. These models primarily facilitate backward reasoning, relying on Bayes' theorem. When it comes to computing probabilities, graph-based algorithms play a pivotal role. These algorithms operate on three main types of graphs: directed, undirected, and factor graphs.

> **Example:**
> A perplexing murder has transpired, with two potential culprits in the spotlight: the butler and the cook. The arsenal of potential murder weapons comprises a butcher's knife, a pistol, and a fireplace poker.
>
> Taking into account the butler's lengthy, faithful service to the family and the cook's recent hiring, along with rumors of a questionable past, we can draw the following conclusions:
>
> $$\mathrm{P}(\text{Culprit} \rightarrow \textit{butler}) = 20\% \qquad \mathrm{P}(\text{Culprit} \rightarrow \textit{cook}) = 80\%$$
>
> The culprit is represented as a binary random variable, with probabilities that sum to 100%.

The butler, being ex-military, maintains a firearm securely stored in a locker drawer. On the other hand, the cook has easy access to a plethora of knives. Additionally, it's worth noting that the butler is aging and experiencing a decline in physical strength. As for the available weapons:

$$\text{Weapon} = \{pistol, knife, poker\}$$

Based on the provided evidence, we can assert that:

$$\text{P}(\text{Weapon}|\text{Culprit} \to \textit{butler}) = \begin{bmatrix} 80\% & 10\% & 10\% \end{bmatrix}$$

$$\text{P}(\text{Weapon}|\text{Culprit} \to \textit{cook}) = \begin{bmatrix} 5\% & 65\% & 30\% \end{bmatrix}$$

By applying the chain rule, we can ultimately calculate the joint distribution, which is as follows:

	pistol	**knife**	**poker**
cook	4%	52%	24%
butler	16%	2%	2%

Employing the sum rule, we can determine the marginal distribution of culprits, with an 80% probability attributed to the cook and a 20% probability assigned to the butler. Additionally, we can compute the marginal distribution of weapons: 20% for the pistol, 54% for the knife, and 26% for the poker. Should we come across the weapon at some point, we can leverage Bayes' theorem to ascertain the identity of the perpetrator.

5.3 Density estimation

For calculating the probability of a logical expression, we can employ the following summation formula:

$$\text{P}(E) = \sum_{row \backsim E} \text{P}(row)$$

To compute inference, we can use the formula:

$$\text{P}(E_1|E_2) = \frac{\text{P}(E_1 \wedge E_2)}{E_2} = \frac{\sum_{row \backsim E_1 \wedge E_2} \text{P}(row)}{\sum_{row \backsim E_2} \text{P}(row)}$$

Example:
Given the following truth table:

A	B	C	$\text{P}(A, B, C)$
0	0	0	0.30
0	0	1	0.05
0	1	0	0.10
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.25
1	1	1	0.10

$\mathrm{P}(A)$ can be found by summing all the probability where $A = 1$, that is:

$$\mathrm{P}(A) = 0.05 + 0.10 + 0.25 + 0.10 = 0.5$$

$\mathrm{P}(A \wedge B)$ can be found by summing all the probability where $A = 1$ and $B = 1$, that is:

$$P(A \wedge B) = 0.25 + 0.10 = 0.35$$

$\mathrm{P}(\overline{A} \vee B)$ can be found by summing all the probability where $A = 0$ or $B = 1$, that is:

$$\mathrm{P}(\overline{A} \vee B) = 0.30 + 0.05 + 0.10 + 0.05 + 0.25 + 0.10 = 0.85$$

$P(A|B)$ can be found by dividing the probability of $A = 1$ and $B = 1$ by the probability where $B = 1$, that is:

$$\mathrm{P}(A|B) = \frac{(0.25 + 0.10)}{(0.10 + 0.05 + 0.25 + 0.10)} = 0.7$$

$\mathrm{P}(C|A \wedge B)$ can be found by dividing the probability of $A = 1$ and $B = 1$ and $C = 1$ by the probability where $A = 1$ and $B = 1$, that is:

$$\mathrm{P}(C|A \wedge B) = \frac{(0.10)}{(0.25 + 0.10)} = 0.285$$

$\mathrm{P}(\overline{A}|C)$ can be found by dividing the probability of $A = 0$ and $C = 1$ by the probability where $C = 1$, that is:

$$\mathrm{P}(\overline{A}|C) = \frac{(0.05 + 0.05)}{(0.05 + 0.05 + 0.10 + 0.10)} = 0.333$$

Definition (*Density estimator*)**.** A density estimator is designed to acquire knowledge in the form of a mapping from a set of attributes to a probability distribution within the attribute space, defined as:

$$M : \{0, 1\}^I \to [0, 1]$$

Likelihood density estimators We can employ likelihood to assess density estimation. When presented with a record x, a density estimator M provides an estimate of how probable it is, denoted as $\widehat{\mathrm{P}}(x|M)$. When dealing with a dataset containing N records, a density estimator can inform us about the likelihood of the data, assuming that all records were generated independently from it:

$$\widehat{\mathrm{P}}(\text{dataset}) = \widehat{\mathrm{P}}(x_1, x_2, \dots, x_N) = \prod_{n=1}^{N} \widehat{\mathrm{P}}(x|M)$$

Due to the potential issue of likelihood values becoming extremely small, it's common practice to work with log-likelihood instead:

$$\log \widehat{\mathrm{P}}(\text{dataset}) = \log \prod_{n=1}^{N} \widehat{\mathrm{P}}(x_n|M) = \sum_{n=1}^{N} \log \widehat{\mathrm{P}}(x_n|M)$$

Density estimators offer a range of valuable capabilities, including the ability to order records by their probability, which can help identify outliers or anomalies. They are also instrumental in inference tasks and can serve as a foundation for Bayes classifiers. However, it's important to note that the primary challenge with joint density estimators is their susceptibility to severe overfitting.

Naïve density estimators The naïve Bayes estimator model operates under the assumption that each attribute is independent of the others. Using the notation where $x[i]$ represents the i^{th} field of record x the naïve density estimator posits that:

$$x[i] \perp \{x[1], x[2], \dots, x[i-1], x[i], x[i+1], \dots, x[I]\}$$

It's crucial to acknowledge that in the naïve Bayes estimator model, all attributes are treated as equally important, and the model assumes that the knowledge of one attribute conveys no information about the value of another. While this final assumption is rarely accurate in real-world scenarios, it often yields effective results when applied in practice.

> **Example:**
> In the context of the naïve Bayes estimator model, given four variables A, B, C, and D, they are all considered independent due to the model's assumptions. Consequently, this implies that:
> $$\mathrm{P}(A, \overline{B}, C, \overline{D}) = \mathrm{P}(A)\mathrm{P}(\overline{B})P(C)\mathrm{P}(\overline{D})$$

In order to train a naïve Bayes estimator, we must make the assumption that $x[1], \dots, x[n]$ are independently distributed. Under this hypothesis, it becomes feasible to construct any row of the resulting joint distribution as needed:

$$\widehat{\mathrm{P}}(x[1] = u_1, x[2] = u_2, \dots, x[I] = u_I) = \prod_{k=1}^{I} \widehat{\mathrm{P}}(x[k] = u_k)$$

	Joint estimator	**Naïve estimator**
Modelling	Anything	Boring distributions
Attributes	Few Boolean	Many multivalued
Overfitting	Yes	Quite robust

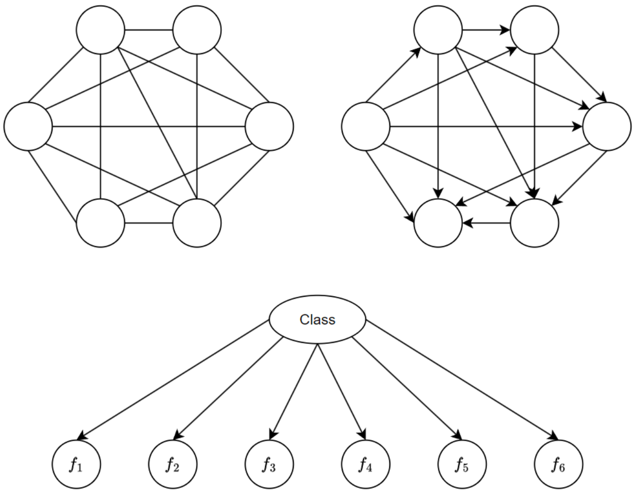


Figure 5.1: Comparison between a joint and a naïve estimator

CHAPTER 6

Bayesian networks

6.1 Introduction

In the real world, random variables often exhibit correlations. To address this reality, it is possible to represent a probability distribution using:

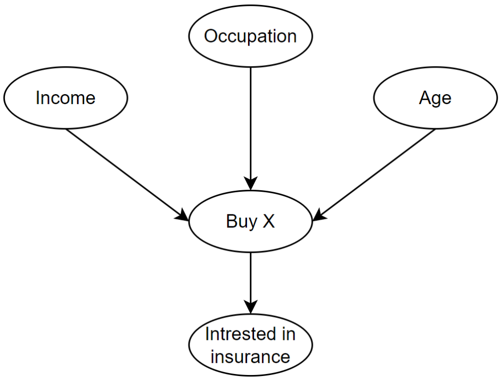
- Conditional independence assumptions that are valid for a subset of these variables.
- A set of conditional probabilities along with their priors.

Models that are constructed based on these principles are commonly referred to as graphical models.

6.2 Bayesian network

A Bayesian network describes the joint probability distribution of variables using a directed graph. Nodes represent random variables, and edges indicate direct influence.

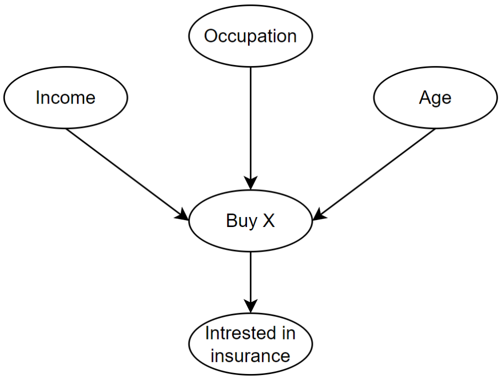
Example:
In the provided Bayesian network:



Random variables ”age,” ”income,” and ”occupation” are independent, whereas ”buy X” and ”Interested in insurance” are conditional probability distributions.

For a set of variables $x_1, x_2, \ldots, x_n$ we can determine any combination probability using a Bayesian network with $2^N - 1$ parameters. To represent these probabilities within the network, we require only the priors and conditional parameters. This is achieved by multiplying the number of nodes by 2^k, where k represents the number of incoming edges, giving us the total number of parameters needed.

Example:
In the provided Bayesian network:



For the full joint distribution, we require 2^N-1 parameters, which in this case is $2^5-1 = 31$ parameters. To represent the Bayesian network itself, we need to calculate the parameters required:

- For the "income", "occupation", and "age" nodes with no incoming edges: $3 \cdot 2^0 = 3$ parameters.
- For the "buy x" node with three incoming edges: $1 \cdot 2^3 = 8$ parameters.
- For the "Interested" node with one incoming edge: $1 \cdot 2^1 = 2$ parameters.

So, the total parameters needed to represent the Bayesian network is 3+8+2 = 13 parameters.

Definition (*Conditionally indipendent*)**.** We define X_1 to be conditionally independent of X_2 given X_3 if the probability of X_1 is not influenced by the value of X_2 when we have knowledge about X_3. This can be expressed as:

$$\mathrm{P}(X_1|X_2, X_3) = \mathrm{P}(X_1|X_3)$$

Likewise, for sets of variables, we can state that X_1, X_2, X_3 are independent of Y_1, Y_2, Y_3 given Z_1, Z_2, Z_3:

$$\mathrm{P}(X_1, X_2, X_3|Y_1, Y_2, Y_3, Z_1, Z_2, Z_3) = \mathrm{P}(X_1, X_2, X_3|Z_1, Z_2, Z_3)$$

Example:
Martin and Norman are tossing the same coin, and we have two variables: A represents "Norman's outcome," and B represents "Martin's outcome". If the coin might be biased, A and B are not independent. Observing that B is heads leads us to revise our belief in A being heads. Therefore, we have:

$$\mathrm{P}(A|B) \neq \mathrm{P}(A)$$

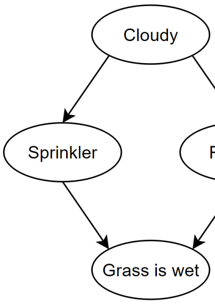
Both variables A and B are dependent on another variable, C, representing "the coin is biased towards heads with probability θ". Once we know the value of C, any evidence about B cannot alter our belief about A. This can be expressed as:

$$\mathrm{P}(A|B,C) = \mathrm{P}(A|C)$$

Definition (*Prior probability*)**.** A prior probability is a probability associated with a variable that has no incoming edges in a Bayesian network.

It's important to note that in a Bayesian network, a node is independent of its ancestors, given its parent node.

Example:
The event "grass is wet" ($W = true$) can be caused by two possible factors: either the "sprinkler" is on ($S = true$), or it is "raining" ($R = true$). The corresponding Bayesian network is as follows.



Where the probabilities for cloudy are:

Cloudy	**P(C)**
0	0.5
1	0.5

Where the probabilities for sprinkler are:

Sprinkler	**Cloudy**	**P(S|C)**
0	0	0.1
0	1	0.5
1	0	0.9
1	1	0.5

Where the probabilities for raining are:

Raining	**Cloudy**	**P(R|C)**
0	0	0.8
0	1	0.5
1	0	0.2
1	1	0.5

Where the probabilities for the wet grass are:

Wet	**Sprinkler**	**Raining**	**P(W|S,R)**
0	0	0	1
0	0	1	0.1
0	1	0	0.1
0	1	1	1
1	0	0	0
1	0	1	0.9
1	1	0	0.9
1	1	1	0.99

With all these values it is possible to compute all the probabilities with the formula:

$$\begin{aligned}\mathrm{P}(C,S,R,W) &= \mathrm{P}(W|S,R,C)\mathrm{P}(S,R,C) = \\ &= \mathrm{P}(W|S,R)\mathrm{P}(S,R,C) = \\ &= \mathrm{P}(W|S,R)\mathrm{P}(S|R,C)P(R,C) = \\ &= \mathrm{P}(W|S,R)\mathrm{P}(S|C)P(R,C) = \\ &= \mathrm{P}(W|S,R)\mathrm{P}(S|C)P(R|C)P(C)\end{aligned}$$

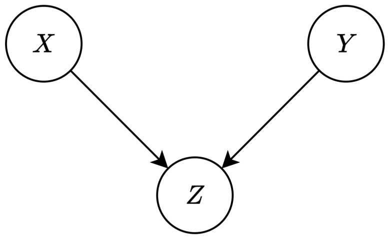
With this formula we can compute all the joint probabilities.

C	**S**	**W**	**R**	**P(C, S, W, R)**
0	0	0	0	0.04
0	0	0	1	0
0	0	1	0	0.001
0	0	1	1	0.009
0	1	0	0	0.036
0	1	0	1	0.324
0	1	1	0	0.0009
0	1	1	1	0.0891
1	0	0	0	0.125
1	0	0	1	0
1	0	1	0	0.0125
1	0	1	1	0.1125
1	1	0	0	0.125
1	1	0	1	0.1125
1	1	1	0	0.00125
1	1	1	1	0.12375

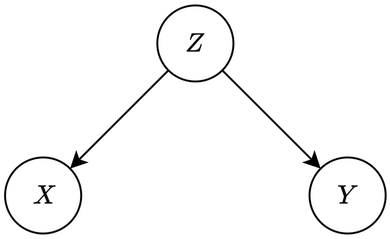
In statistics, the phenomenon of explaining away is often referred to as Berkson's paradox or selection bias. It pertains to situations where two variables become dependent due to the observation of a third variable.

Classification In a Bayesian network, the relationships between variables can exhibit various types of dependencies and independencies:

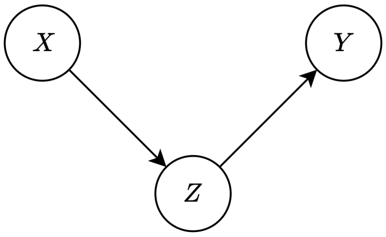
- $\mathrm{P}(X,Y,Z) = \mathrm{P}(X)\mathrm{P}(Y)\mathrm{P}(Z|X,Y)$ and $\mathrm{P}(X,Y|Z) = \mathrm{P}(X)\mathrm{P}(Y)$ if the nodes are connected in the following way.



- $\mathrm{P}(X, Y, Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z)\mathrm{P}(Z)$ and $\mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z)$ if the nodes are connected in the following way.



- $\mathrm{P}(X, Y, Z) = \mathrm{P}(X)\mathrm{P}(Z|X)\mathrm{P}(Y|Z)$ and $\mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z)$ if the nodes are connected in the following way.



Definition (*Conditional indipendence*)**.** Two sets of nodes, denoted as A and B, exhibit what is termed as conditional independence or are often referred to as being d-separated, given a set of nodes C if and only if all paths from A to B are effectively blocked by the presence of C.

When we classify C based on its role within a Bayesian network:

- C is categorized as a root when it remains hidden or unobserved. In this scenario, the children of C are dependent on each other due to the influence of a common unobserved cause. However, if C is observed, the children become conditionally independent, meaning that the common unobserved cause is no longer exerting an impact.

- C is termed a leaf when it is hidden, and its parent nodes are marginally independent of one another. Nevertheless, if C (or any descendant of C) is observed, the parent nodes become dependent on each other, introducing conditional dependence.

- C is described as a bridge when the nodes both upstream and downstream of C become dependent solely when C remains hidden. Conditioning on C, by observing or introducing knowledge about it, effectively disrupts the graph's structure at that particular point, resulting in dependence between the nodes on either side of C.

These distinctions in the role of C are essential in understanding how the presence or absence of observations can affect the conditional independence relationships within a Bayesian network.

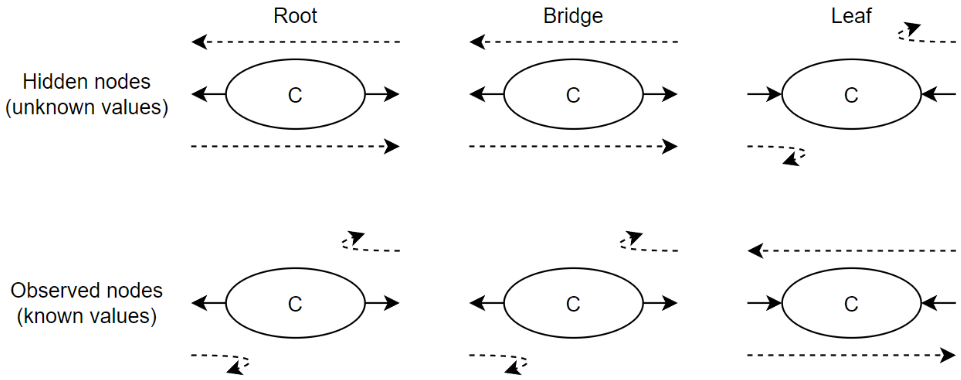


Figure 6.1: Graphical representation of root, bridge, and leaf

Bayesian networks reasoning Two distinct modes of reasoning are applicable when working with Bayesian networks:

- *Bottom-up reasoning*: this approach involves inferring the cause when provided with evidence. In other words, it seeks to determine the underlying factors or causes that may have led to observed outcomes.

- *Top-down reasoning*: in this mode, we calculate the probability of one event given another event. This represents a predictive utilization of Bayesian networks, as they function as "generative" models. It's about estimating the likelihood of certain events based on other known events.

One of the most intriguing attributes of Bayesian networks is their ability to facilitate causal reasoning grounded in a robust mathematical foundation. These networks allow us to explore and understand causal relationships within complex systems using a formal and structured approach.

Moreover, Bayesian networks can encompass nodes with both continuous (real) and discrete values. This versatility provides us with a diverse and powerful toolkit for constructing probabilistic models that can represent a wide range of real-world scenarios and phenomena.

6.3 Inference

Inference in a Bayesian network exhibits exponential complexity, typically on the order of $O(|X_i|^N)$, where $|X_i|$ represents the cardinality of the variables involved. To mitigate this computational complexity, several techniques can be applied:

- *Variable elimination*: this method involves systematically eliminating variables to simplify the calculation of probabilities and reduce computational complexity.

- *Belief propagation*: by passing messages between nodes in the network, this technique efficiently computes marginal probabilities and updates beliefs, helping to alleviate the computational burden.

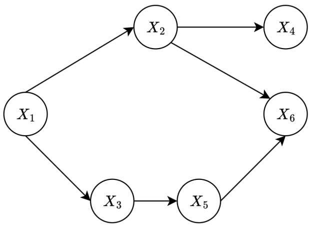
- *Junction trees*: these data structures are used to represent Bayesian networks, and they facilitate efficient computations for probabilistic inference. The process of building junction trees can significantly reduce complexity.
- *Loopy belief propagation*: in situations where the network contains loops or cycles, loopy belief propagation can be employed to approximate inferences by iteratively updating beliefs.
- *Sampling based methods*: these methods rely on random sampling to estimate probabilities and make inferences. They can be effective for handling complex Bayesian networks when exact methods become impractical.

It's worth noting that the first three methods, namely variable elimination, belief propagation, and junction trees, are exact methods that provide precise solutions, while the last two, sampling-based methods and loopy belief propagation, are approximate techniques used when obtaining an exact solution is challenging due to computational constraints.

6.4 Variable elimination

The factored representation of joint probability offers an efficient approach to marginalization by consolidating summations as deeply as feasible. As we carry out these innermost summations, we generate fresh terms in the process.

Example:
In the Bayesian network depicted below:



We can compute the probability distribution for X_5 as follows:

$$\begin{aligned}
\mathrm{P}(X_5) =& \sum_{X_1}\sum_{X_2}\sum_{X_3}\sum_{X_4}\sum_{X_6} \mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\mathrm{P}(X_3|X_1)\mathrm{P}(X_4|X_2)\mathrm{P}(X_5|X_3)\mathrm{P}(X_6|X_5,X_2)\\
=& \sum_{X_1}\sum_{X_2}\sum_{X_3}\sum_{X_6} \mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\mathrm{P}(X_3|X_1)\mathrm{P}(X_5|X_3)\mathrm{P}(X_6|X_5,X_2)\sum_{X_4}\mathrm{P}(X_4|X_2)\\
=& \sum_{X_1}\sum_{X_2}\sum_{X_3} \mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\mathrm{P}(X_3|X_1)\mathrm{P}(X_5|X_3)\mu_1(X_2)\sum_{X_6}\mathrm{P}(X_6|X_5,X_2)\\
=& \sum_{X_2}\sum_{X_3} \mathrm{P}(X_5|X_3)\mu_1(X_2)\mu_2(X_5,X_2)\sum_{X_1}\mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\mathrm{P}(X_3|X_1)\\
=& \sum_{X_3} \mathrm{P}(X_5|X_3)\sum_{X_2}\mu_1(X_2)\mu_2(X_5,X_2)\mu_3(X_2,X_3)\\
=& \sum_{X_3} \mathrm{P}(X_5|X_3)\mu_4(X_3,X_5)\\
=& \mu_5(X_5)
\end{aligned}$$

Dynamic programming The variable elimination procedure relies on dynamic programming. To utilize this approach, we break down the main problem into multiple smaller problems by leveraging the factorization of the joint distribution. This factorization not only guides us in establishing the most efficient order for variable elimination but also aids in determining the functions for the intermediate variables denoted as μ. To automate this procedure, we employ factor graph models, a type of graphical model wherein the box notation signifies terms dependent on specific variables.

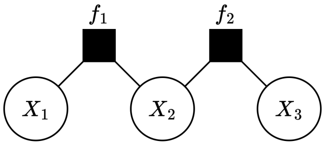


Figure 6.2: A simple example of a factor graph

The key characteristics of this graph include:

- It constitutes a bipartite graph.
- Each circular node represents a random variable, denoted as X_i.
- Each boxed node represents a factor, labeled as f_k, which can be expressed as a function $f_k(X_{C_k})$.
- The joint probability distribution is expressed as:

$$\mathrm{P}(X_1, X_2, \ldots, X_N) = \prod_{k=1}^{K} f_k(X_{C_k})$$

In this representation, the factor graph serves as a powerful tool for modeling and automating probabilistic calculations, offering a clear visual depiction of variable dependencies and factor relationships.

Factor graph To convert a Bayesian network into a factor graph, the process involves applying a step called moralization. In the typical scenario, this operation entails establishing links between the parents of nodes while preserving as many independence properties as possible.

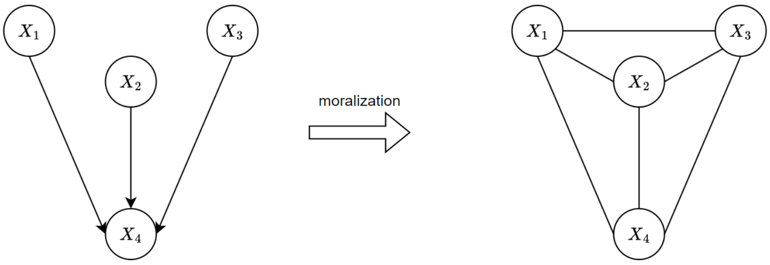
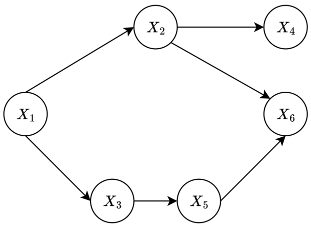


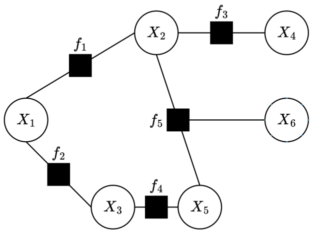
Figure 6.3: An example of moralization

To convert the Bayesian network into a factor graph, it's necessary to combine unconnected parents into a single factor.

Example:
In the provided Bayesian network:



The corresponding factor graph is represented as follows:

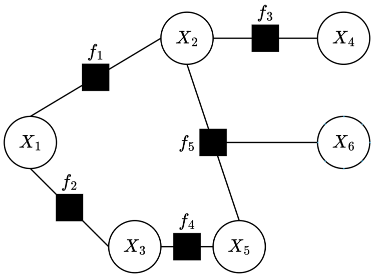


Definition (*Perfect map*)**.** A graph is a perfect map if and only if every independence property of a distribution is reflected in the graph and vice versa.

Please be aware of the following:

- Not all probability distributions can be faithfully depicted using directed or undirected graphs.
- Not all directed graphs can be transformed into undirected graphs while maintaining their original properties.
- Not all undirected graphs can be converted into directed graphs while preserving their original characteristics.

Example:
The chain can be portrayed as a factor graph in the following manner:



- $f_1(X_1) = P(X_1)$
- $f_2(X_1, X_2) = \mathrm{P}(X_1|X_2)$
- $f_3(X_2, X_3) = \mathrm{P}(X_3|X_2)$
- $f_4(X_3, X_4) = \mathrm{P}(X_4|X_3)$

- $f_5(X_4, X_5) = \mathrm{P}(X_5|X_4)$
- $f_5(X_5, X_6) = \mathrm{P}(X_6|X_5)$

It's important to note that factor graphs are not unique; there are various ways to represent any given Bayesian network.

Algorithm The variable elimination algorithm takes two inputs: a list denoted as F containing factors and a tuple labeled as C_0 representing the output variables to retain. The result of the algorithm is a single factor m that encompasses the variables in X_{C_0}.

Algorithm 1 Variable elimination algorithm

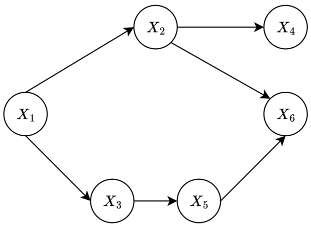
```
define all variables present in $F$ as $V \leftarrow \mathrm{vars}(F)$
define all variables to be eliminated as $E \leftarrow V - C_0$
**for** all $i \in E$ **do**
    call eliminate_single_variable$(F, i)$
**end for**
**for** all remaining factors **do**
    $m \leftarrow \prod_{f \in F} f$
**end for**
```

The function eliminate_single_variable(F, i) in the prior algorithm accepts as inputs the list of factors denoted as F and the variable identified by i. It returns the updated list of factors, which remains labeled as F.

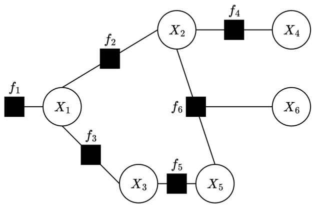
Algorithm 2 eliminate_single_variable(F, i)

```
find relevant subset $f \subset F$ of factors over $i$ as $f \leftarrow \{C|i \in C\}$
define the remaining clique as $C_t \leftarrow \mathrm{vars}(f) - \{i\}$
compute the temporary factor as $\mu(X_{C_t}) = \sum_{X_i} \prod_{f \in F} f$
remove old factors $f$ and append new temporary factor $t$ to $F$
**return** $F$
```

Example:
Consider the Bayesian network illustrated below:



We can compute the factor graph with moralization, resulting in the following factor graph:



Now, let's compute the marginal probability $\mathrm{P}(X_1, X_6) = \mu(X_1, X_6)$. To do this, we'll utilize the variable elimination algorithm with the following inputs:

- $F = \{f_1, f_2, f_3, f_4, f_5, f_6\}$
- $C_0 = \{X_1, X_6\}$

The algorithm progresses as follows:

1. Define all the variables present in F:

$$V = \{X_1, X_2, X_3, X_4, X_5, X_6\}$$

2. Calculate the set of variables to be eliminated:

$$E = V - C_0 = \{X_1, X_2, X_3, X_4, X_5, X_6\} - \{X_1, X_6\} = \{X_2, X_3, X_4, X_5\}$$

Now, we need to eliminate the single variables contained in set E using the eliminate_single_variable function. Here's the step-by-step process for each variable:

- Eliminate X_4:
  - Check the connected functions: $f = \{f_4\}$.
  - The clique containing X_4 (excluding itself) is: $C_t = \{X_2\}$.
  - Calculate the temporary factor: $\mu_1(X_2) = \sum_{X_4} P(X_4|X_2)$.
  - Update F to: $F = \{f_1, f_2, f_3, f_5, f_6, \mu_1\}$.
  - Remove f from E, resulting in $E = \{X_2, X_3, X_5\}$
- Eliminate X_3:
  - Check the connected functions: $f = \{f_3, f_5\}$.
  - The clique containing X_3 (excluding itself) is: $C_t = \{X_1, X_5\}$.
  - Calculate the temporary factor: $\mu_2(X_1, X_5) = \sum_{X_3} P(X_3|X_1)P(X_5|X_3)$.
  - Update F to: $F = \{f_1, f_2, f_6, \mu_1, \mu_2\}$.
  - Remove f from E, resulting in $E = \{X_2, X_5\}$
- Eliminate X_5:
  - Check the connected functions: $f = \{f_6, \mu_2\}$.
  - The clique containing X_5 (excluding itself) is: $C_t = \{X_1, X_2, X_6\}$.
  - Calculate the temporary factor: $\mu_3(X_1, X_2, X_6) = \sum_{X_5} \mu_2(X_1, X_5)P(X_6|X_2, X_5)$.
  - Update F to: $F = \{f_1, f_2, \mu_1, \mu_3\}$.

- Remove f from E, resulting in $E = \{X_2\}$

- Eliminate X_2:
  - Check the connected functions: $f = \{f_2, \mu_1, \mu_3\}$.
  - The clique containing X_2 (excluding itself) is: $C_t = \{X_1, X_6\}$.
  - Calculate the temporary factor: $\mu_4(X_1, X_6) = \sum_{X_2} \mu_1(X_2)P(X_2|X_1)\mu_3(X_1, X_2, X_6)$.
  - Update F to: $F = \{f_1, \mu_4\}$.
  - Remove f from E, resulting in $E = \varnothing$

After these iterations, we have the following results:

- $F = \{f_1, \mu_4\}$
- $C_0 = \{X_1, X_6\}$

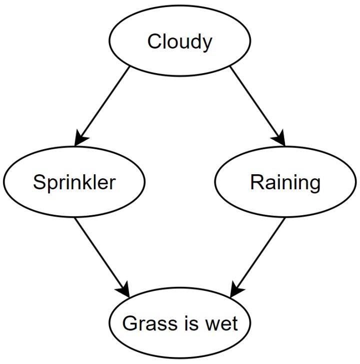
The final steps involve multiplying all the elements in f:

$$\mathrm{P}(X_1, X_6) = \mu_4 \cdot f_1 = P(X_1)\mu_4(X_1, X_6)$$

This yields the marginal probability $\mathrm{P}(X_1, X_6)$ using the variable elimination algorithm.

It's worth mentioning that the order in which variables are selected as input for the function can be determined using a heuristic function. One common approach is to order the nodes based on the ascending number of connections they have in the factor graph.

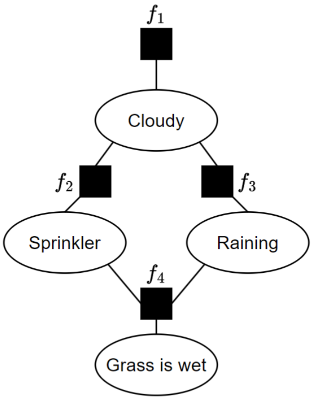
Example:
Consider the sprinkler example with the probabilities given before:



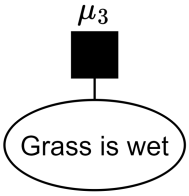
We want to compute $\mathrm{P}(W)$. To do this, we start by transforming the Bayesian network into a factor graph, where we define the factors as follows:

- $f_1(C) = \mathrm{P}(C)$.
- $f_2(S, C) = \mathrm{P}(S|C)$.
- $f_3(R, C) = \mathrm{P}(R|C)$.
- $f_4(W, S, R) = \mathrm{P}(W|S, R)$.

The resulting factor graph looks like this:



After applying the variable elimination algorithm, we obtain the following graph:



The corresponding truth table for the variable $\mu_3(W)$ is as follows:

Wet	$\boldsymbol{\mu_3}$**(W)**
0	0.22915
1	0.77085

Variable elimination possesses several advantages, including its simplicity of implementation and the faithful representation of manual calculations. With an optimal ordering, its complexity is $O(N2^K)$. However, it does have its drawbacks. Finding the optimal ordering is an $\mathcal{NP}$-hard problem, it can compute only one marginal at a time, and computing all marginals necessitates N executions, which can be inefficient for more extensive networks.

6.5 Belief propagation

Belief propagation, often referred to as message passing, bears similarities to the variable elimination algorithm. This approach is amenable to parallel execution at each node, where it computes messages exchanged between variables and factors. It can be summarized in two fundamental steps: message computation and probability estimation based on the derived messages. In cases of circular dependencies, this method is effective when applied to a graph structured as a tree or polytree (a tree with no designated root).

It's important to note that messages in belief propagation correspond to the μ factors in the variable elimination algorithm, with leaf μ factors always initialized to 1. To initiate the algorithm, we require message values from a prior state, making it practical to start from the leaf nodes.

As previously mentioned, belief propagation can be expressed as a parallel procedure that adheres to the following steps:

1. Initialization of all messages:

$$\mu_{f_s \to X_i} = 1$$

$$\mu_{X_i \to f_s} = 1$$

2. Message updates:

$$\mu^{new}_{f_s \to X_i}(X_i) = \sum_{X_s \setminus X_i} f_s(X_i, X_s) \prod_{X_j \in new(X_i), j \neq i} \mu^{old}_{X_i \to f_s}(X_j)$$

$$\mu^{new}_{X_i \to f_s}(X_i) = \prod_{f_l \in new(X_i), f_l \neq f_s} \mu^{old}_{f_l \to X_i}(X_i)$$

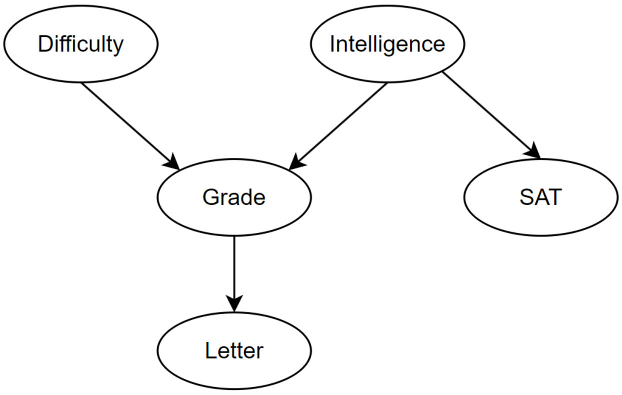
3. Belief updates:

$$b_{f_s}(X_s) = f_s(X_s) \prod_{j \in f_s} \mu_{X_j \to f_s}(X_j)$$

$$b_{X_i}(X_i) = \prod_{f_s \in new(i)} \mu_{f_s \to X_i}(X_i)$$

This iterative procedure is executed until convergence is achieved.

Example:
Consider the Bayesian network shown below along with the associated probabilities:



With these probabilities:

d^0	d^1
0.6	0.4

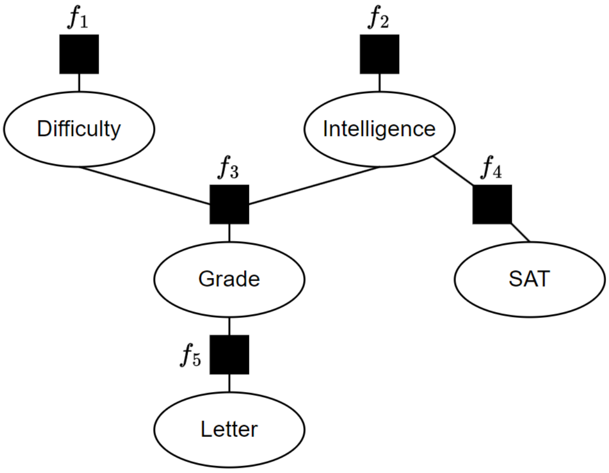
i^0	i^1
0.7	0.3

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^1, d^0	0.9	0.08	0.02
i^1, d^1	0.5	0.3	0.2

From this network, we create the corresponding factor graph:



The corresponding functions are:

D	$f_1(D)$
0	0.6
1	0.4

I	$f_2(I)$
0	0.7
1	0.3

G	I	D	$f_3(G, I, D)$
1	0	0	0.3
1	0	1	0.05
1	1	0	0.9
1	1	1	0.5
2	0	0	0.4
2	0	1	0.25
2	1	0	0.08
2	1	1	0.3
3	0	0	0.3
3	0	1	0.7
3	1	0	0.02
3	1	1	0.2

S	I	$f_4(S,I)$
0	0	0.95
0	1	0.2
1	0	0.05
1	1	0.8

L	G	$f_5(L,G)$
0	1	0.1
0	2	0.4
0	3	0.99
1	1	0.9
1	2	0.6
1	3	0.01

In the initial step, all messages are set to one:

$\mu_{D\to 1} = [1,1]$	$\mu_{1\to D} = [1,1]$
$\mu_{D\to 3} = [1,1]$	$\mu_{2\to I} = [1,1]$
$\mu_{G\to 3} = [1,1]$	$\mu_{3\to D} = [1,1]$
$\mu_{G\to 5} = [1,1]$	$\mu_{3\to I} = [1,1]$
$\mu_{I\to 3} = [1,1]$	$\mu_{3\to G} = [1,1,1]$
$\mu_{I\to 2} = [1,1]$	$\mu_{4\to I} = [1,1]$
$\mu_{I\to 4} = [1,1]$	$\mu_{4\to S} = [1,1]$
$\mu_{L\to 5} = [1,1]$	$\mu_{5\to G} = [1,1,1]$
$\mu_{S\to 4} = [1,1]$	$\mu_{5\to L} = [1,1]$

In the second step, we update the messages from factors to variables, resulting in:

- $\mu_{1\to D}(D)$:
  - $0 \to 0.6$
  - $1 \to 0.4$
- $\mu_{2\to I}(I)$:
  - $0 \to 0.7$
  - $1 \to 0.3$
- $\mu_{3\to D}(D)$:
  - $0 \to 0.3\cdot 1\cdot 1 + 0.9\cdot 1\cdot 1 + 0.4\cdot 1\cdot 1 + 0.08\cdot 1\cdot 1 + 0.3\cdot 1\cdot 1 + 0.02\cdot 1\cdot 1 = 2$
  - $1 \to 0.05\cdot 1\cdot 1 + 0.5\cdot 1\cdot 1 + 0.25\cdot 1\cdot 1 + 0.3\cdot 1\cdot 1 + 0.7\cdot 1\cdot 1 + 0.2\cdot 1\cdot 1 = 2$
- $\mu_{3\to I}(I)$:
  - $0 \to 0.3\cdot 1\cdot 1 + 0.05\cdot 1\cdot 1 + 0.4\cdot 1\cdot 1 + 0.25\cdot 1\cdot 1 + 0.3\cdot 1\cdot 1 + 0.7\cdot 1\cdot 1 = 2$
  - $1 \to 0.9\cdot 1\cdot 1 + 0.5\cdot 1\cdot 1 + 0.08\cdot 1\cdot 1 + 0.3\cdot 1\cdot 1 + 0.02\cdot 1\cdot 1 + 0.2\cdot 1\cdot 1 = 2$
- $\mu_{4\to S}(S)$:

  - $0 \rightarrow 0.95 \cdot 1 + 0.2 \cdot 1 = 1.15$
  - $1 \rightarrow 0.05 \cdot 1 + 0.8 \cdot 1 = 0.85$

- $\mu_{4\rightarrow I}(I)$:
  - $0 \rightarrow 0.95 \cdot 1 + 0.05 \cdot 1 = 1$
  - $1 \rightarrow 0.2 \cdot 1 + 0.8 \cdot 1 = 1$
- $\mu_{3\rightarrow G}(G)$:
  - $1 \rightarrow 0.3 \cdot 1 \cdot 1 + 0.05 \cdot 1 \cdot 1 + 0.9 \cdot 1 \cdot 1 + 0.5 \cdot 1 \cdot 1 = 1.75$
  - $2 \rightarrow 0.4 \cdot 1 \cdot 1 + 0.25 \cdot 1 \cdot 1 + 0.08 \cdot 1 \cdot 1 + 0.3 \cdot 1 \cdot 1 = 1.03$
  - $3 \rightarrow 0.3 \cdot 1 \cdot 1 + 0.7 \cdot 1 \cdot 1 + 0.02 \cdot 1 \cdot 1 + 0.2 \cdot 1 \cdot 1 = 1.22$
- $\mu_{5\rightarrow G}(G)$:
  - $1 \rightarrow 0.1 \cdot 1 + 0.9 \cdot 1 = 1$
  - $2 \rightarrow 0.4 \cdot 1 + 0.6 \cdot 1 = 1$
  - $3 \rightarrow 0.99 \cdot 1 + 0.01 \cdot 1 = 1$
- $\mu_{5\rightarrow L}(L)$:
  - $0 \rightarrow 0.1 \cdot 1 + 0.4 \cdot 1 + 0.99 \cdot 1 = 1.49$
  - $1 \rightarrow 0.9 \cdot 1 + 0.6 \cdot 1 + 0.01 \cdot 1 = 1.51$

$\mu_{D\rightarrow 1} = [1, 1]$	$\mu_{1\rightarrow D} = [0.6, 0.4]$
$\mu_{D\rightarrow 3} = [1, 1]$	$\mu_{2\rightarrow I} = [0.7, 0.3]$
$\mu_{G\rightarrow 3} = [1, 1]$	$\mu_{3\rightarrow D} = [2, 2]$
$\mu_{G\rightarrow 5} = [1, 1]$	$\mu_{3\rightarrow I} = [2, 2]$
$\mu_{I\rightarrow 3} = [1, 1]$	$\mu_{3\rightarrow G} = [1.75, 1.03, 1.22]$
$\mu_{I\rightarrow 2} = [1, 1]$	$\mu_{4\rightarrow I} = [1, 1]$
$\mu_{I\rightarrow 4} = [1, 1]$	$\mu_{4\rightarrow S} = [1.15, 0.85]$
$\mu_{L\rightarrow 5} = [1, 1]$	$\mu_{5\rightarrow G} = [1, 1, 1]$
$\mu_{S\rightarrow 4} = [1, 1]$	$\mu_{5\rightarrow L} = [1.49, 1.51]$

The messages from variables to factors are updated as well:

- $\mu_{D\rightarrow 1}(D)$:
  - $0 \rightarrow 2$
  - $1 \rightarrow 2$
- $\mu_{I\rightarrow 2}(I)$:
  - $0 \rightarrow 2 \cdot 1 = 2$
  - $1 \rightarrow 2 \cdot 1 = 2$
- $\mu_{D\rightarrow 3}(D)$:
  - $0 \rightarrow 0.6$

  - $1 \to 0.4$

- $\mu_{I \to 3}(I)$:
  - $0 \to 0.7 \cdot 1 = 0.7$
  - $1 \to 0.3 \cdot 1 = 0.3$
- $\mu_{S \to 4}(S)$:
  - $0 \to 1$
  - $1 \to 1$
- $\mu_{I \to 4}(I)$:
  - $0 \to 0.7 \cdot 2 = 1.4$
  - $1 \to 0.3 \cdot 2 = 0.6$
- $\mu_{G \to 3}(G)$:
  - $1 \to 1$
  - $2 \to 1$
  - $3 \to 1$
- $\mu_{G \to 5}(G)$:
  - $1 \to 1.75$
  - $2 \to 1.03$
  - $3 \to 1.22$
- $\mu_{L \to 5}(L)$:
  - $0 \to 1$
  - $1 \to 1$

$\mu_{D \to 1} = [2, 2]$	$\mu_{1 \to D} = [0.6, 0.4]$
$\mu_{D \to 3} = [0.6, 0.4]$	$\mu_{2 \to I} = [0.7, 0.3]$
$\mu_{G \to 3} = [1, 1, 1]$	$\mu_{3 \to D} = [2, 2]$
$\mu_{G \to 5} = [1.75, 1.03, 1.22]$	$\mu_{3 \to I} = [2, 2]$
$\mu_{I \to 3} = [0.7, 0.3]$	$\mu_{3 \to G} = [1.75, 1.03, 1.22]$
$\mu_{I \to 2} = [2, 2]$	$\mu_{4 \to I} = [1, 1]$
$\mu_{I \to 4} = [1.4, 0.6]$	$\mu_{4 \to S} = [1.15, 0.85]$
$\mu_{L \to 5} = [1, 1]$	$\mu_{5 \to G} = [1, 1, 1]$
$\mu_{S \to 4} = [1, 1]$	$\mu_{5 \to L} = [1.49, 1.51]$

We continue this process until convergence, assuming the following table exhibits this property (we assume that the values reported below are convergent):

$\mu_{D\to 1} = [2, 2]$	$\mu_{1\to D} = [0.6, 0.4]$
$\mu_{D\to 3} = [0.6, 0.4]$	$\mu_{2\to I} = [0.7, 0.3]$
$\mu_{G\to 3} = [1, 1, 1]$	$\mu_{3\to D} = [1, 1]$
$\mu_{G\to 5} = [1.75, 1.03, 1.22]$	$\mu_{3\to I} = [1, 1]$
$\mu_{I\to 3} = [1.4, 0.6]$	$\mu_{3\to G} = [0.362, 0.288, 0.349]$
$\mu_{I\to 2} = [2, 2]$	$\mu_{4\to I} = [1, 1]$
$\mu_{I\to 4} = [0.7, 0.3]$	$\mu_{4\to S} = [1.45, 0.55]$
$\mu_{L\to 5} = [1, 1]$	$\mu_{5\to G} = [1, 1, 1]$
$\mu_{S\to 4} = [1, 1]$	$\mu_{5\to L} = [1.7948, 2.2052]$

Now, we can compute the belief for each variable:

- $b(D) = [0.6 \cdot 1 = 0.6, 0.4 \cdot 1 = 0.4]$.
- $b(I) = [0.7 \cdot 1 \cdot 1 = 0.7, 0.3 \cdot 1 \cdot 1 = 0.3]$.
- $b(S) = \left[\frac{1.45}{1.45 + 0.55} = 0.72, \frac{0.55}{1.45 + 0.55} = 0.28\right]$.
- $b(L) = \left[\frac{1.79}{1.79 + 2.20} = 0.45, \frac{2.20}{2.20 + 1.79} = 0.55\right]$.
- $b(G) = [0.362 \cdot 1 = 0.362, 0.288 \cdot 1 = 0.288, 0.349 \cdot 1 = 0.349]$.

Marginal probabilities Let's examine the computation of marginal probabilities in a factorized probability distribution $\mathrm{P}(X) = \mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\mathrm{P}(X_3|X_2)\mathrm{P}(X_4|X_3)$ where we focus on X_2. Given specific observations for $X_1 = x_{e_1}$ and $X_3 = x_{e_3}$, we can calculate $\mathrm{P}(X)$ as follows:

$$\mathrm{P}(X_2|X_1 = x_{e_1}, X_3 = x_{e_3}) = \mathrm{P}(X_1 = x_{e_1})\mathrm{P}(X_2|X_1 = x_{e_1})\mathrm{P}(X_3 = x_{e_3}|X_2)\sum_{X_4}\mathrm{P}(X_4|X_3)$$

On the other hand, if we want to compute the marginal probability of X_2 with no observations, it can be expressed as:

$$\mathrm{P}(X_2) = \sum X_1 \mathrm{P}(X_1)\mathrm{P}(X_2|X_1)\sum_{X_3}\mathrm{P}(X_3|X_2)\sum_{X_4}\mathrm{P}(X_4|X_3)$$

These computations can be efficiently performed on a tree structure using belief propagation. If certain nodes X_e are observed, their values are used directly when computing messages, instead of summing over all possible values. After normalization, this provides the conditional probability given the evidence. The belief propagation variant used for this problem involves the following steps:

1. Initialize all messages to one.
2. Update the belief values.
3. Update the messages.

This process is iterated until convergence. n the case of marginal consistency, the marginal beliefs in two cliques, denoted as C and D, that share a variable X_i, should be equal:

$$b(X_i) = \sum_{X_C \setminus X_i} b(X_C) = \sum_{X_D \setminus X_i} b(X_D)$$

Additionally, marginal consistency implies that:

$$b(X_i) = \mu_{C \to i}(X_i)\mu_{i \to C}(X_i)$$

Marginal consistency serves as a fixed point of the belief propagation updates, meaning that Belief Updates do not alter the messages. Marginal consistency has the following implications:

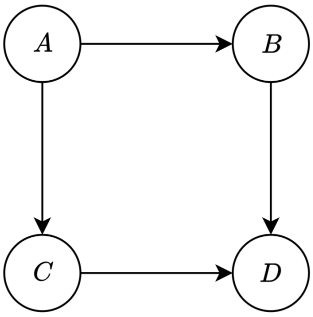
- On trees, the parallel message updates will converge to the true messages.
- On polytrees, convergence to the true messages is also ensured.
- On non-tree structures, it reaches a state of marginal consistency, which may not guarantee true convergence.

While belief update equations can be applied to loopy graphs, it's important to note that in the presence of loops, branches of nodes do not represent independent information because belief propagation involves the multiplication (fusion) of messages from dependent sources. n practice, there can be echo effects and convergence may not always occur as expected. It typically converges to a perturbed result, potentially leading to overconfident estimates.

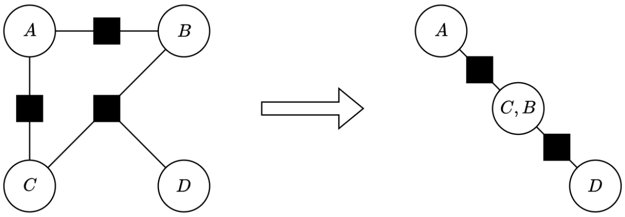
6.6 Junction trees

Several models incorporate loops, and it is feasible to transform them into a tree structure by delineating variable groups (separators) that define message interactions.

Example:
Consider the Bayesian network depicted below:



Upon applying the junction tree algorithm, the resulting structure is illustrated as follows:



The process of constructing a junction tree involves the following steps:

1. Moralize the graph in case it is directed.
2. Choose a node ordering and identify cliques generated through variable elimination, thereby creating a graph triangulation.

3. Construct a junction graph based on the eliminated cliques.

4. Determine a suitable spanning tree.

6.7 Sampling based methods

Consider a Bayesian network encompassing random variables $X_1, \dots, X_N$, some of which are observed ($X_{\text{obs}} = y_{\text{obs}}$). The objective is to compute marginal posteriors $\text{P}(X_i | X_{\text{obs}} = y_{\text{obs}})$ conditioned on the provided observations. To achieve this, a set of K joint samples $S = \{(x_1, x_2, \dots, x_N)\}_{k=1}^{K}$ is generated, where each sample $(x_1, x_2, \dots, x_N)$ represents a list of instantiations for all0 $X_1, X_2, \dots, X_N$. With these samples in hand, the computation of $\text{P}(X_i = x | X_{\text{obs}} = y_{\text{obs}})$ is approximated as follows:

$$\text{P}(X_i = x | X_{\text{obs}} = y_{\text{obs}}) \approx \frac{\text{count}_S\left(x_i^k = x \wedge x_{\text{obs}}^k = y_{\text{obs}}\right)}{\text{count}_S\left(x_{\text{obs}}^k = y_{\text{obs}}\right)}$$

Various sampling methods can be employed, including:

- *Rejection sampling*: discards false samples, potentially slowing down when a significant number of samples are rejected.
- *Likelihood sampling*: force the usability of all samples.
- *Gibbs sampling*: directly generates samples based on the provided probability distribution.

6.7.1 Rejection sampling

To generate a single sample $x_{1:N}^k$ using rejection sampling, the following steps are employed:

1. Sort all random variables in topological order.
2. Initialize with $i = 1$.
3. Sample a value $x_i^k \sim \text{P}\left(X_i | x_{\text{parents}(X_i)}^k\right)$ conditional on $x_{i:i-1}^k$.
4. If i is in the set of observed variables (obs), compare the sampled value x_i^k with the observation y_i. Reject and repeat from the previous steps if the sample does not match the observation.
5. Repeat with $i = i + 1$.

After generating k, the computation of $\text{P}\left(X_i = x | X_{\text{obs}} = y_{\text{obs}}\right)$ is approximated as:

$$\text{P}\left(X_i = x | X_{\text{obs}} = y_{\text{obs}}\right) \approx \frac{\text{count}_S\left(x_i^k = x\right)}{K}$$

> **Example:**
> Consider the sprinkler example. The sampling of x^k involves generating random numbers with $U(0,1)$ for each variable. Specifically:
>
> - If $s \leq \text{P}(C)$, then $x_C^k = 1$; $x_C^k = 0$ otherwise.

- If $s \leq \mathrm{P}(S|x_C^k)$, then $x_S^k = 1$; $x_S^k = 0$ otherwise.
- If $s \leq \mathrm{P}(R|x_C^k)$, then $x_R^k = 1$; $x_R^k = 0$ otherwise.
- If $s \leq \mathrm{P}(W|x_S^k, x_R^k)$, then $x_W^k = 1$; $x_W^k = 0$ otherwise.

Generated samples that are entirely false are discarded. The process is repeated for the four variables to obtain various samples. One possible set of samples could be:

C	**S**	**R**	**W**
0	1	0	1
0	0	0	0
0	1	1	1
0	1	1	1
0	1	1	1
1	0	1	1

Eliminating samples that are entirely false, the final set might look like the following:

C	**S**	**R**	**W**
0	1	0	1
0	1	1	1
0	1	1	1
0	1	1	1
1	0	1	1

From these samples, probabilities such as $\mathrm{P}(C|W)$ can be computed:

$$\mathrm{P}(C|W) = \frac{\mathrm{P}(C \wedge W)}{\mathrm{P}(W)} = \frac{\#(C \wedge W)}{\#(W)} = \frac{1}{5} = 0.2$$

However, this method becomes inefficient when dealing with rare events due to the rejection of a significant number of samples.

6.7.2 Likelihood sampling

To generate a single weighted sample $(w_k, x_{1:N}^k)$ using importance sampling with likelihood weighting, the following steps are taken:

1. Sort all random variables in topological order.
2. Initialize with $i = 1$ and $w^k = 1$.
3. If i is not in the set of observed variables (obs), sample a value $x_i^k \sim \mathrm{P}\left(X_i|x_{\text{parents}(X_i)}^k\right)$ conditional on $x_{1:i-1}^k$.
4. If i is in obs, set the value $x_i^k = y_i$ and update $w^k = w^k \cdot \mathrm{P}\left(X_i = y_i|x_{1:i-1}^k\right)$.
5. Repeat with $i = i + 1$.

After generating k samples, the computation of $\mathrm{P}\left(X_i = x|X_{\text{obs}} = y_{\text{obs}}\right)$ is approximated as:

$$\mathrm{P}\left(X_i = x|X_{\text{obs}} = y_{\text{obs}}\right) \approx \frac{\sum_{k=1}^{K} w^k \cdot I_{x_i^k=x}}{\sum_{k=1}^{K} w^k}$$

Here, $I_{x_i^k=x}$ is 1 if $x_i^k = x$ or 0 otherwise.

Example:
Consider the sprinkler example. In the sampling of x^k, we initialize w^k to 1 and generate three random numbers from $U(0,1)$. The sampling proceeds as follows:

- If $s \leq \mathrm{P}(C)$, then $x_C^k = 1$; $x_C^k = 0$ otherwise.
- If $s \leq \mathrm{P}(S|x_C^k)$, then $x_S^k = 1$; $x_S^k = 0$ otherwise.
- If $s \leq \mathrm{P}(R|x_C^k)$, then $x_R^k = 1$; $x_R^k = 0$ otherwise.

The fourth part of the sample is determined by forcing $w_W^k = 1$ using the formula:

$$w^k = w^k \cdot \mathrm{P}\left(x_W^k = 1|x_S^k, x_R^k\right)$$

This probability is computed based on the other free variables, and the value can be obtained by checking the row with the values found for S and R. The process is repeated for all four variables to obtain various samples. One possible set of samples, along with their corresponding weights, could be:

C	S	R	W	w
0	1	0	1	0.9
0	0	0	1	0
0	1	1	1	0.99
0	1	1	1	0.99
0	1	1	1	0.99
1	0	1	1	0.9

After obtaining multiple samples, probabilities such as $\mathrm{P}(C|W)$ can be computed:

$$\mathrm{P}(C|W) = \frac{\sum_{k=1}^{K} w^k \cdot I_{x_i^k=x}}{\sum_{k=1}^{K} w^k} = \frac{0.9}{0.9 + 0 + 0.99 + 0.99 + 0.99 + 0.9} = 0.1886$$

The key distinction from rejection sampling is that all samples are considered, making this method suitable for handling rare events.

6.7.3 Gibbs sampling

In Gibbs sampling, the samples are dependent as each subsequent sample is influenced by the preceding one. To implement Gibbs sampling, the following steps are taken:

1. All observed variables (obs) are fixed to their observed values $x_i^k = y_i$ for any k.

2. To generate the $(k-1)$th sample, iterate over latent variables $i \notin \text{obs}$, updating:

$$\begin{aligned} x_i^{k+1} &\sim \mathrm{P}\left(X_i | x^k_{1:N\setminus i}\right) \\ &\sim \mathrm{P}\left(X_i | x_1^k, x_2^k, \dots, x_{i-1}^k, x_{i+1}^k, \dots, x_N^k\right) \\ &\sim \mathrm{P}\left(X_i | x^k_{\text{parents}(i)}\right) \prod_{j:i\in\text{parents}(j)} \mathrm{P}\left(X_j = x_j^k | X_i, x^k_{\text{parents}(j)\setminus i}\right) \end{aligned}$$

Each x_I^{k+1} is resampled conditionally on the other current sample values. After an initial set of samples is obtained, subsequent samples can be directly utilized.

Example:
Consider the sprinkler example. Applying Bayes' theorem, we can obtain the proportional relationships as follows:

- $\mathrm{P}(C|S^k, R^k, W^k) \propto \mathrm{P}(C)\mathrm{P}(S^k|C)\mathrm{P}(R^k|C)$.
- $\mathrm{P}(S|C^k, R^k, W^k) \propto \mathrm{P}(S|C^k)\mathrm{P}(W^k|S, R^k)$.
- $\mathrm{P}(R|C^k, S^k, W^k) \propto \mathrm{P}(R|C^k)\mathrm{P}(W^k|S^k, R)$.
- $\mathrm{P}(W|C^k, S^k, R^k) \propto \mathrm{P}(W|S^k, R^k)$.

The goal is to sample $\mathrm{P}(C|W=1)$ by generating samples x^k based on $U(0,1)$. For the initial sample, likelihood sampling is employed with the following steps:

- Force $x_W^k = 1$.
- If $s \leq \mathrm{P}(C|x_S^{k-1}, x_R^{k-1}, x_W^{k-1})$, then $x_C^k = 1$; $x_C^k = 0$ otherwise.
- If $s \leq \mathrm{P}(S|x_C^{k-1}, x_R^{k-1}, x_W^{k-1})$, then $x_S^k = 1$; $x_S^k = 0$ otherwise.
- If $s \leq \mathrm{P}(R|x_C^{k-1}, x_S^{k-1}, x_W^{k-1})$, then $x_R^k = 1$; $x_R^k = 0$ otherwise.

One possible result for the initial sample can be:

C	**S**	**R**	**W**
0	1	0	1

Following the initial sample, Gibbs' sampling iterations are applied. If we decide to fix the value of W to one, the other variables can be computed using the proportions found before. For instance:

- For cloudy (C):
  - $\mathrm{P}(C=1|S^{k-1}, R^{k-1}, W^{k-1}) \propto 0.125$
  - $\mathrm{P}(C=0|S^{k-1}, R^{k-1}, W^{k-1}) \propto 0.36$

  We can compute the value of $C=1$ given the other variables, which is: $\mathrm{P}(C = 1|S^{k-1}, R^{k-1}, W^{k-1}) = \dfrac{0.125}{0.125+0.36} = 0.257$ We generate the number and put 0 or 1 in the table.

- For sprinkler (S):

  - $P(S=1|C^{k-1},R^{k-1},W^{k-1}) \propto 0.81$
  - $P(S=0|C^{k-1},R^{k-1},W^{k-1}) \propto 0$

  We can compute the value of $C=1$ given the other variables, which is: $P(C=1|S^{k-1},R^{k-1},W^{k-1}) = \frac{0.81}{0.81+0} = 1$ We generate the number and put 0 or 1 in the table.

- For raining (R):
  - $P(R=1|C^{k-1},S^{k-1},W^{k-1}) \propto 0.198$
  - $P(R=0|C^{k-1},S^{k-1},W^{k-1}) \propto 0.72$

  We can compute the value of $C=1$ given the other variables, which is: $P(C=1|S^{k-1},R^{k-1},W^{k-1}) = \frac{0.198}{0.198+0.72} = 0.215$ We generate the number and put 0 or 1 in the table.

This procedure is iteratively applied, and the final table (which may vary depending on the samples) could be:

C	**S**	**R**	**W**
0	1	0	1
0	1	0	1
0	1	1	1
1	1	0	1
0	0	0	1
1	0	0	1

As an example computation, the probability of $P(C|W)$ is:

$$P(C|W) = \frac{P(C \wedge W)}{P(W)} = \frac{\#(C \wedge W)}{\#(W)} = \frac{2}{6} = 0.333$$

CHAPTER 7

Dynamic Bayesian networks

7.1 Introduction

To capture the dynamics of an ever-changing world, a series of random variables describing the state of the world at different time instants can be employed. This sequence of states, often representing time, is modeled as a chain. A crucial property in this context is the Markov property, which is defined as follows:

Property 7.1. The Markov property asserts that the transition from $X_{t-1} = x_i$ to $X_t = x_j$ depends only on the current state X_{t-1}:

$$\mathrm{P}\left(X_t|X_{t-1}, X_{t-2}, \dots, X_0\right) = \mathrm{P}(X_t|X_{t-1})$$

In a stationary process, transition probabilities remain consistent across different time points, ensuring that the system's behavior does not change over time.

7.2 Markov chain

Definition (*Discrete stochastic process*)**.** A discrete stochastic process is a stochastic process that characterizes the stochastic evolution of a system at distinct and separate time steps.

Definition (*Continuous stochastic process*)**.** A continuous stochastic process is a stochastic process in which the system's state can be observed at any continuous point in time.

Definition (*Markov chain*)**.** A discrete stochastic process is a first-order Markov chain when, for all t and for all N states, the following condition holds:

$$\mathrm{P}\left(X_t|X_{t-1}, X_{t-2}, \dots, X_0\right) = \mathrm{P}(X_t|X_{t-1})$$

Definition (*Stationary Markov chain*)**.** A stationary Markov chain is a Markov chain where the probability of an event is independent of time, expressed as:

$$\mathrm{P}\left(X_{t+1=j}|X_t = j\right) = p_{ij}$$

A Markov chain can be represented by a transition matrix, where p_{ij} denotes the probability of transitioning from state i to state j. Alternatively, this transition matrix can be illustrated using a directed graph.

For a Markov chain in state i at time m, the probability of being in state j after n steps is given by:

$$\mathrm{P}_{ij}(n) = ij^{th} \text{ element of the modified tranisiton matrix } P^n$$

The probability of occupying state j at time n, without knowing the exact state of the Markov chain at time 0, is expressed as:

$$\sum_i q_i \cdot \mathrm{P}_{ij}(n) = 1 \cdot (\text{ column } j \text{ of } P^n)$$

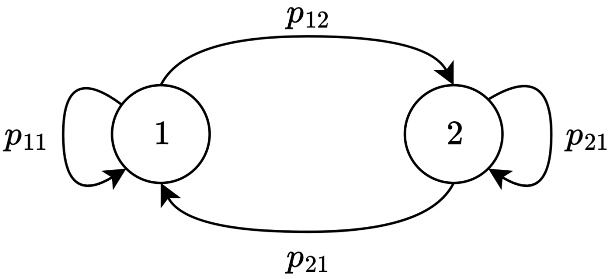
Here, q_i represents the initial state probabilities at time 0.

Example:

Consider a market with just two brands of Cola. A person buying Cola1 will buy Cola1 again with a probability of 0.9. A person buying Cola2 will buy Cola2 again with a probability of 0.8. The associated transition matrix is:

$$P = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$

The corresponding directed graph is shown below:



Suppose someone bought Cola2; we want to compute the probability of buying Cola1 after two time steps. To achieve this, we multiply the transition matrix by itself:

$$P^2 = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} = \begin{bmatrix} 0.83 & 0.17 \\ 0.34 & 0.66 \end{bmatrix}$$

The desired probability is given by the element in position p_{12} in P^2, which is 0.34.

Now, if someone bought Cola1, we want to compute the probability of buying Cola1 after three time steps:

$$P^3 = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} = \begin{bmatrix} 0.781 & 0.219 \\ 0.438 & 0.562 \end{bmatrix}$$

The desired probability is given by the element in position p_{11} in P^3, which is 0.781.

If, at some time, 60% of clients bought Cola1 and 40% bought Cola2, we want to determine the percentage of people buying Cola1 after three time steps:

$$\sum_i q_i \cdot P^3_{ij} = q \cdot \left(\text{column 1 of } P^3\right)$$

$$= \begin{bmatrix} 0.6 & 0.4 \end{bmatrix} \begin{bmatrix} 0.781 \\ 0.438 \end{bmatrix} = 0.6438$$

Definition (*k-th Markov chain*)**.** In a k-th order Markov chain, each state transition depends on the previous k states.

Definition (*Reachable state*)**.** The state j is said to be reachable from i if there exists a path from i to j.

Definition (*Communicating states*)**.** The states i and j are said to be communicating if i is reachable from j and vice versa.

Definition (*Closed set of states*)**.** A set of states S is closed if no state outside S is reachable from a state in S.

Definition (*Absorbing state*)**.** A state i is an absorbing state if $p_{ij} = 1$.

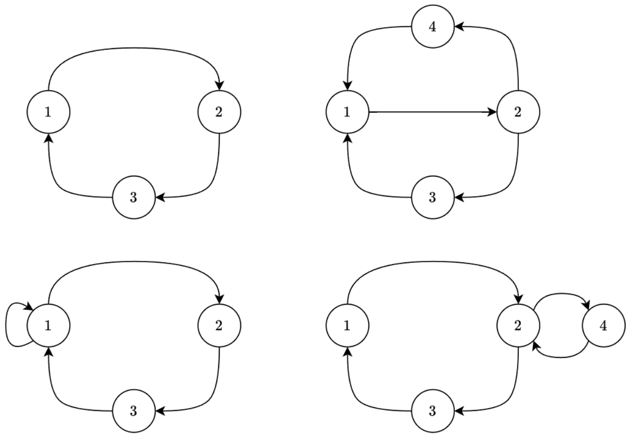
Definition (*Transient state*)**.** A state i is transient if there exists j reachable from i, but i is not reachable from j.

Definition (*Recurrent state*)**.** A state that is not transient is defined as recurrent.

Definition (*Periodic state*)**.** A state i is periodic with period $k > 1$ if k is the largest number that divides the length of all paths from i to i; a state that is not periodic is said to be aperiodic.

Definition (*Ergodic*)**.** If all states in a Markov chain are recurrent, aperiodic, and communicate with each other, it is said to be ergodic.

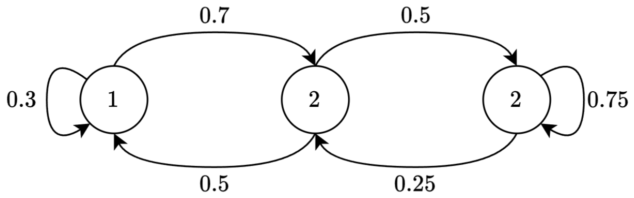
Example:
Consider the following graphs:



Two are periodic, and the other two (bottom ones) are aperiodic.
Consider the Markov chain described by the following transition matrix:

$$P = \begin{bmatrix} 0.3 & 0.7 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 0.25 & 0.75 \end{bmatrix}$$

The corresponding graph is:



We can see that this Markov chain is ergodic because it satisfies all the properties given in the definition.

Steady state distribution Being P the transition matrix of an ergodic Markov chain with N states, we have that:

$$\lim_{n\to\infty} P_{ij}(n) = \pi_j$$

With $\pi = [\pi_1, \pi_2, \dots, \pi_N]$ being the steady-state distribution.

The behavior of a Markov chain before getting to the steady state is defined as transitory; we can compute the expected number of transitions to reach state j being in state i for an ergodic Markov chain as:

$$m_{ij} = p_{ij}(1) + \sum_{k\neq j} p_{ik}(1 + m_{kj}) = 1 + \sum_{k\neq j} p_{ik} \cdot m_{kj}$$

Example:
It is possible to compute how many bottles on average a Cola1 buyer will have before switching to Cola2:

$$m_{12} = 1 + \sum_{k\neq j} p_{1k} \cdot m_{k2} = 1 + p_{11} \cdot m_{12} = 1 + 0.9 \cdot m_{12} = \frac{1}{1-0.9} = 10$$

And also how many bottles on average a Cola2 buyer will have before switching to Cola1:

$$m_{21} = 1 + \sum_{k\neq j} p_{2k} \cdot m_{k1} = 1 + p_{22} \cdot m_{21} = 1 + 0.8 \cdot m_{21} = \frac{1}{1-0.8} = 5$$

Absorbing states We have an absorbing Markov chain if there exist one or more absorbing states and all the others are transient. Its transition matrix is:

$$P = \begin{bmatrix} Q & R \\ 0 & 1 \end{bmatrix}$$

Here, Q is the transition matrix for transient states, and R is the transition matrix from transient to absorbing states.

The average time spent in a transient state j, starting from a transient state j, is equal to the ij-th element of:

$$(I - Q)^{-1}$$

The average time spent to reach an absorbing state ij, starting from a transient state i, is equal to the ij-th element of:

$$(I - Q)^{-1} \cdot R$$

Example:
Consider a company with three hierarchical levels: junior, senior, and partner. At any time, a person could leave the company as a senior or not. We can define a transition matrix with five columns: junior, senior, partner, leave not partner, and leave partner. The matrix is the following:

$$P = \begin{bmatrix} 0.80 & 0.15 & 0 & 0.05 & 0 \\ 0 & 0.70 & 0.20 & 0.10 & 0 \\ 0 & 0 & 0.95 & 0 & 0.05 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

It is possible to separate the matrix into the four parts seen before. The average time spent in a transient state is:

$$\text{transient} = (I - Q)^{-1} = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.80 & 0.15 & 0 \\ 0 & 0.70 & 0.20 \\ 0 & 0 & 0.95 \end{bmatrix} \right)^{-1} = \begin{bmatrix} 5 & 2.5 & 10 \\ 0 & 3.3 & 13.3 \\ 0 & 0 & 20 \end{bmatrix}$$

We can compute how long a junior will remain in the company:

- As a junior: $m_{11} = 5$.
- As a senior: $m_{12} = 2.5$.
- As a partner: $m_{13} = 10$.

With a total of 17.5 years.

The average time to reach an absorbing state is:

$$\text{absorbing} = (I - Q)^{-1}R = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.80 & 0.15 & 0 \\ 0 & 0.70 & 0.20 \\ 0 & 0 & 0.95 \end{bmatrix} \right)^{-1} \begin{bmatrix} 0.05 & 0 \\ 0.10 & 0 \\ 0 & 0.05 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \\ 0 & 1 \end{bmatrix}$$

The probability for a junior to leave the company as a partner is given by the m_{12} element of the previous matrix, which has a value of 0.5.

7.3 Hidden Markov model

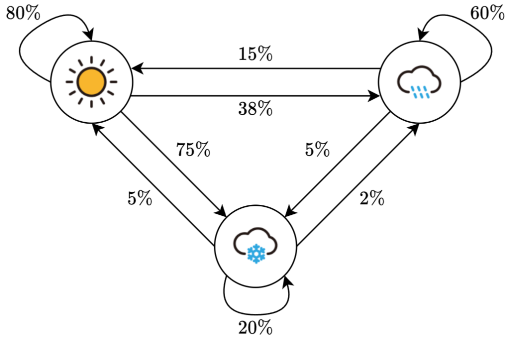
If we may not observe directly the states, we get another Bayesian network named as a Hidden Markov model. A Hidden Markov model is described by a quintuple $\langle S, E, P, A, B \rangle$, where:

- $S = \{S_1, \dots, S_N\}$: represents the values for the hidden states.
- $E = \{e_1, \dots, e_T\}$: represents the values for the observations.
- P: the probability distribution of the initial state.
- A: the transition probability matrix.
- B: the emission probability matrix.

Example:
Consider a Markov chain describing weather states with the set of states

$$\{S_{\text{sunny}}, S_{\text{rainy}}, S_{\text{snowy}}\}$$

The corresponding graph is as follows:



The corresponding transition matrix is given by:

$$P = \begin{bmatrix} 0.80 & 0.15 & 0.05 \\ 0.38 & 0.60 & 0.02 \\ 0.75 & 0.05 & 0.20 \end{bmatrix}$$

The initial state distribution is given by $q = \begin{pmatrix} 0.7 & 0.25 & 0.05 \end{pmatrix}$, representing a typical day in the considered region.

Suppose we have the series: sunny, rainy, rainy, rainy, snowy, snowy. The probability of this series is calculated as follows:

$$\begin{aligned} P(S) &= \mathrm{P}(S_{\text{sunny}})\mathrm{P}(S_{\text{rainy}}|S_{\text{sunny}})\mathrm{P}(S_{\text{rainy}}|S_{\text{rainy}})\mathrm{P}(S_{\text{rainy}}|S_{\text{rainy}})\mathrm{P}(S_{\text{snowy}}|S_{\text{rainy}})\mathrm{P}(S_{\text{snowy}}|S_{\text{snowy}}) \\ &= 0.7 \cdot 0.15 \cdot 0.6 \cdot 0.6 \cdot 0.02 \cdot 0.2 = 0.0001512 \end{aligned}$$

Now, we introduce three observations for each state $\{O_{\text{shorts}}, O_{\text{coat}}, O_{\text{umbrella}}\}$. In this case, in addition to the transition matrix A, we have an observation matrix B:

$$B = \begin{bmatrix} 0.60 & 0.30 & 0.10 \\ 0.05 & 0.30 & 0.65 \\ 0.00 & 0.50 & 0.50 \end{bmatrix}$$

Here, states are represented by rows, and observations are represented by columns. The objective is to compute the probability of a series of observations and, if possible, identify the underlying sequence of states.

Forward probability The forward probability represents the joint probability of actual states and observations and is computed as follows:

$$\mathrm{P}(X_t = s_i, e_{1:t}) = \sum_j A_{ij} B_{je_t} \mathrm{P}(X_{t-1} = s_j, e_{1:t-1})$$

This forward probability serves multiple purposes:

- It can be utilized to calculate the probability of the observations:

$$P(e_{1:t})$$

- It can be employed for making predictions about the next state:

$$P(X_{t+1} = s_i | e_{1:t})$$

Viterbi algorithm From the given observations, it is feasible to compute the most likely hidden state sequence:

$$\arg\max\left[\mathrm{P}(X_{1:t}|e_{1:t})\right] = \arg\max\left[\frac{\mathrm{P}(X_{1:t}, e_{1:t})}{\mathrm{P}(e_{1:t})}\right] = \arg\max\left[\mathrm{P}(X_{1:t}, e_{1:t})\right]$$

Applying Bayesian network factorization, we obtain:

$$\mathrm{P}(X_{1:t}, e_{1:t}) = \mathrm{P}(X_0) \prod_{i=1:t} \mathrm{P}(X_i|X_{i-1})\mathrm{P}(e_i|X_i)$$

The solution we seek minimizes:

$$-\log \mathrm{P}(X_{1:t}, e_{1:t}) = -\log \mathrm{P}(X_0) + \sum_{i=1:t} \left(-\log \mathrm{P}(X_i|X_{i-1}) - \log \mathrm{P}(e_i|X_i)\right)$$

To represent this problem graphically, construct a graph with $1 + tN$ nodes: one initial node and N nodes at each time i, where the j-th node represents $X_i = s_j$.

CHAPTER 8

Learning Bayesian networks

8.1 Learning classification

A Bayesian network is characterized by its graphical structure and the parameters associated with each conditional probability density. These components can be specified either through expert knowledge or learned from available data. It is important to note the following:

- The process of learning the network structure is more challenging compared to learning its parameters.
- Learning in scenarios where certain nodes are concealed or when data is incomplete is considerably more difficult than situations where all observations are available.

Learning approaches can be categorized based on both structure and observability, yielding the following classifications:

	Observability	
Structure	*Full*	*Partial*
Known	Maximum likelihood estimation	Expectation Maximization
Unknown	Search through model space	EM with search through model space

8.1.1 Known structure with full observability

In this scenario, our objective is to determine the values of parameters in the conditional probability distributions that maximize the likelihood of the observed data, denoted as $d \in D$:

$$L = \sum_{i}^{N} \sum_{d}^{D} \log \mathrm{P}(X_i | \text{parents}(X_i), d)$$

To simplify computation, we utilize the log-likelihood. The log-likelihood decomposes based on the graph structure, enabling the maximization of each node's contribution independently. Sparse data challenges are addressed by incorporating Dirichlet priors. For Gaussian nodes, we compute the sample mean and variance, employing linear regression to estimate the weight matrix.

Example:
Let's consider the sprinkler example with Wet being true. For the Wet node, given a set of training data, we count the occurrences of Wet when it is Raining and the Sprinkler is on (for all combinations):

$$\mathrm{P}(W = 1|R = 1, S = 1) = \frac{\#(W, S, R)}{\#(S, R)}$$

However, since this specific combination might never occur, we adjust the denominator to prevent it from being null:

$$\mathrm{P}(W = 1|R = 1, S = 1) = \frac{\#(W, S, R)}{\#(\overline{W}, S, R) + \#(W, S, R)}$$

To account for the possibility of some combinations never occurring, we introduce Dirichlet priors. This rule ensures that each combination occurs at least once by adding one to the total number of observed evidence instances.

8.1.2 Known structure with partial observability

When certain nodes are hidden, the Expectation Maximization algorithm can be employed to derive a locally optimal Maximum likelihood estimate, as follows:

- *E-Step*: compute expected values for unobserved variables using an inference algorithm. Treat these expected values as if they were observed.

- *M-Step*: consider the model as fully observable and apply the algorithm designed for full observability.

Given the expected counts, iteratively maximize parameters and recalculate expected counts. The Expectation Maximization algorithm converges to a local maximum likelihood.

Example:
Let's consider the sprinkler example with Wet being true. For the Wet node, observed counts are replaced with the expected occurrences of each event:

$$\mathrm{P}(W = 1|R = 1, S = 1) = \frac{E\left[\#(W, S, R)\right]}{E\left[\#(S, R)\right]} = \frac{\sum_d^D \mathrm{P}(W, S, R|d)}{\sum_d^D \mathrm{P}(S, R|d)}$$

However, it's worth noting that this algorithm becomes slow for large networks due to the necessity of applying inference for each data point at each iteration.

8.1.3 Unknown structure with full observability

Structure learning aims to discover a Directed Acyclic Graph that provides the most accurate explanation for the given data. Here are key points about structure learning:

- In general, it poses an $\mathcal{NP}$-hard problem, with the number of DAGs on N variables growing super-exponentially in N.

- If the node ordering is known, the parent set for each node can be learned independently.

The approach involves initializing the model structure and then conducting a local search. This search evaluates the scores of neighboring structures, moving to the best one until a local optimum is reached. Several algorithms can be employed for this task, including the Tabu search algorithm, genetic algorithms (for global optimization), and multiple restarts (for finding both global optimum and learning a model ensemble).

Start with an initial guess of the model structure, then perform local search, evaluating the score of neighboring structures, and move to the best one until reaches a local optimum. The possible algorithms to to this are: Tabu search algorithm, genetic algorithms (to find a global optimum), and use multiple restarts (to find global optimum and to learn a model ensemble).

Maximum likelihood model The Maximum likelihood model, denoted as G_{MLE}, corresponds to a complete graph. It possesses the maximum number of parameters, offering the best data fit. However, being a joint distribution, it tends to overfit.

Maximum a posteriori model To address overfitting, the maximum a posteriori model is considered. It involves the probability of the graph given the data:

$$\mathrm{P}(G|D) = \frac{\mathrm{P}(G|D)\mathrm{P}(G)}{\mathrm{P}(D)}$$

For computational simplicity, the formula is often expressed in logarithmic terms:

$$\mathrm{P}(G|D) = \log \mathrm{P}(G|D) + \log \mathrm{P}(G) - \log \mathrm{P}(D)$$

While $\mathrm{P}(G)$ could be used to penalize complex models, it's not always necessary, as the term:

$$\mathrm{P}(D|G) = \int_\theta \mathrm{P}(D|G, \theta)$$

already has a similar effect.

8.1.4 Unknown structure with partial observability

This situation is typically challenging due to the unknown structure and the presence of hidden variables and/or missing data. To address this, learning employs posterior approximation using the Bayesian information criterion:

$$\log \mathrm{P}(G|D) \approx \log \mathrm{P}(D|G, \widehat{\Theta}_G) - \frac{N \log R}{2}$$

Here, R represents the number of samples, $\widehat{\Theta}_G$ denotes the model parameters, and N is the number of variables.

In cases of full observability, the model's dimension is determined by the number of free parameters. However, in models with hidden variables, this dimension might be lower.

The Bayesian information criterion breaks down into a sum of local terms. Despite this decomposition, local search remains computationally expensive, involving Expectation Maximization at each step to compute $\widehat{\Theta}_G$, with the local search executed in the M-step.

CHAPTER 9

Casual inference

9.1 Introduction

Definition (*Causal inference*)**.** Causal inference is the process of identifying the independent and actual effect of a specific phenomenon within a larger system.

It is crucial to emphasize that correlation does not imply causation.

Example:
Let's consider a new pandemic where we need to choose the most effective treatment among two possibilities. treatment $T = \{A, B\}$, condition $C = \{Mild, Severe\}$, and outcome $Y = \{Alive, Dead\}$ are the variables of interest. The initial mortality rate table is presented below:

Treatment	**Mortality**	**Proportion**
A	16%	240/1500
B	19%	105/550

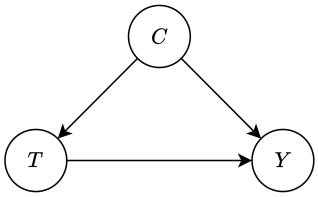
Initially, treatment A seems preferable as it exhibits a lower mortality percentage. However, introducing patient condition data alters the assessment:

Treatment	**Total**		**Mild**		**Severe**	
A	16%	240/1500	15%	210/1400	30%	30/100
B	19%	105/550	10%	5/50	20%	100/500

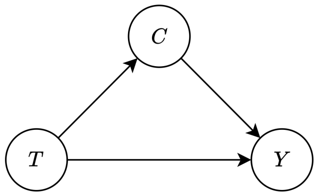
Surprisingly, with this updated table, treatment B emerges as the superior choice, leading to a paradox known as Simpson's paradox. This paradox results from the unequal distribution of patients across treatments.

To address this paradox, considering different treatments for mild and severe patients may be a solution. Two scenarios are explored:

1. Assigning treatment A to mild patients and treatment B to severe patients. The corresponding causality graph is as follows:



2. Prioritizing mild patients with treatment A due to its quick availability, while administering the slower treatment B to severe patients. The corresponding causality graph is illustrated below:



Notably, the choice of treatment is influenced by the underlying causality graph.

To assess the correlation between two events, we can choose to either implement or abstain from the action causing the event. In the former case, we refer to it as factual evidence, while in the latter, it constitutes counterfactual evidence. The causal effect is quantified as the difference between factual and counterfactual evidences:

$$Y_i(1) - Y_i(0) = 1$$

Definition (*Individual treatment effect*)**.** The individual treatment effect is defined as the disparity between taking an action and doing nothing within a population:

$$Y_i(1) - Y_i(0)$$

Definition (*Average treatment effect*)**.** The average treatment effect is defined as the expected value of the individual treatment effect:

$$E\left[Y_i(1) - Y_i(0)\right] = E\left[Y_i(1)\right] - E\left[Y_i(0)\right] \neq E\left[Y|T=1\right] - E\left[Y|T=0\right]$$

The inequality in the previous definition arises from the distinction that the first term considers only causal associations, while the second term includes both causal and confounding associations.

Randomized control trials A potential remedy to this challenge is to address confounding associations through randomized control trials In RCTs, subjects are randomly assigned to treatment groups, ensuring that T does not have causal parents, and the groups are comparable. However, the applicability of this technique may be limited due to ethical concerns, infeasibility, or outright impossibility.

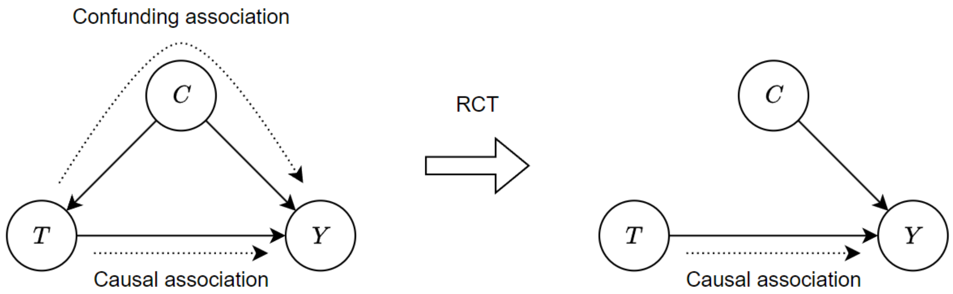


Figure 9.1: Differences between base case and RCT case

Confounding adjustments An alternative strategy is to control for the appropriate variables W. If W forms a sufficient adjustment set, we can express:

$$E\left[Y(t)|W=w\right]=E\left[Y|do(T=t),W=w\right]=E\left[Y|t,w\right]$$

By marginalizing W, we arrive at the backdoor adjustment formula:

$$E\left[Y(t)\right]=E\left[Y|do(T=t)\right]=E_W E\left[Y|t,W\right]$$

It is important to note that the set of variables leading to d-separation acts as a sufficient adjustment set.

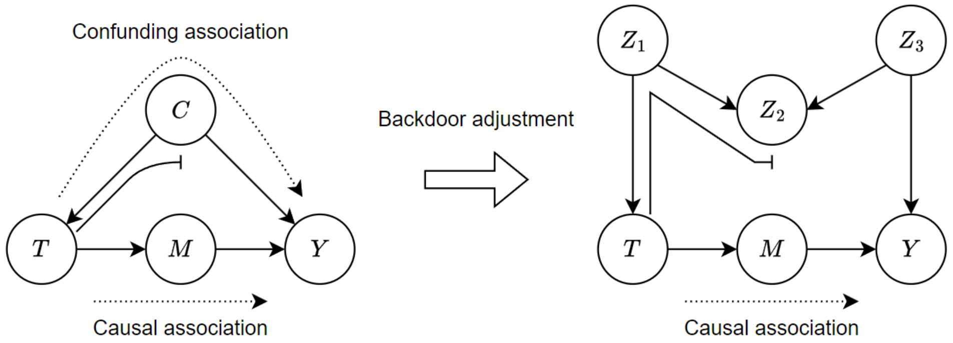
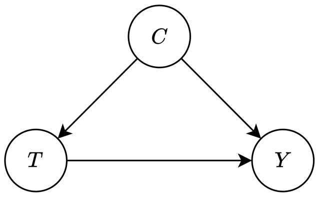


Figure 9.2: Differences between base case and RTC case

Example:
Consider Simpson's paradox illustrated by the mortality table:

Treatment	Total		Mild		Severe	
A	16%	240/1500	15%	210/1400	30%	30/100
B	19%	105/550	10%	5/50	20%	100/500

Let's consider the first scenario, leading to the following Bayesian network:



Choosing treatment A based on the total percentage is inaccurate due to the uneven distribution of mild and severe cases. To make the best decision, we break the confounding correlation:

$$= E\left[Y|do(T=t)\right] = E_C E\left[Y|t,C\right] = \sum_C E\left[T|t,c\right]P(c)$$

Normalizing the number of people checked, we obtain:

$$P_{\text{causal}}(A) = \frac{1450}{2050}(0.15) + \frac{600}{2050}(0.30) = 0.194$$

$$P_{\text{causal}}(B) = \frac{1450}{2050}(0.10) + \frac{600}{2050}(0.20) = 0.129$$

In contrast, the naïve probabilities were:

$$P_{\text{naïve}}(A) = \frac{1400}{1500}(0.15) + \frac{100}{1500}(0.30) = 0.16$$

$$P_{\text{naïve}}(B) = \frac{50}{550}(0.10) + \frac{500}{550}(0.20) = 0.19$$