Fall 2024

BECE309L – AI & ML

Probability Reasoning & Uncertainty

Module – 4

Dr. Nitish Katal

Outline

- Quantifying uncertainty
- Knowledge representation in uncertainty,
- Decision making
  - Simple,
  - Complex.

Uncertainty

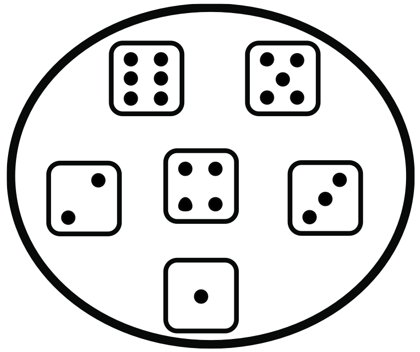
- General situation:
  - Observed variables (evidence): Agent knows certain things about the state of the world (e.g., sensor readings or symptoms)
  - Unobserved variables: Agent needs to reason about other aspects (e.g. where an object is or what disease is present)
  - Model: Agent knows something about how the known variables relate to the unknown variables
- Probabilistic reasoning gives us a framework for managing our beliefs and knowledge

Uncertainty

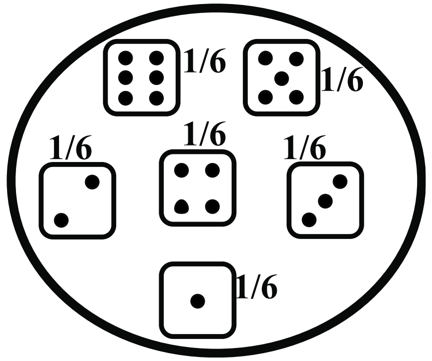
- The real world is rife with uncertainty!
  - E.g., if I leave for Airport from college 60 minutes before my flight, will I be there in time?
- Problems:
  - partial observability (road state, other drivers' plans, etc.)
  - noisy sensors (radio traffic reports, Google maps)
  - immense complexity of modelling and predicting traffic, security line, etc.
  - lack of knowledge of world dynamics (will tire burst? will I get in crash?)
- Probabilistic assertions summarize effects of ***ignorance*** and ***laziness***
- Combine probability theory + utility theory -> decision theory
  - ***Maximize expected utility*** : $a^* = \text{argmax}_a \sum_s P(s \mid a)\, U(s)$

Basic laws of probability (discrete)

- Begin with a set Ω of possible worlds
  - E.g., 6 possible rolls of a die, {1, 2, 3, 4, 5, 6}

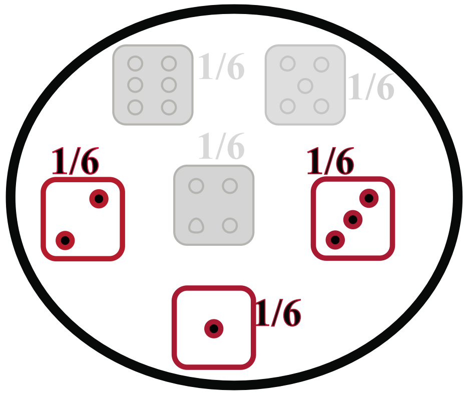


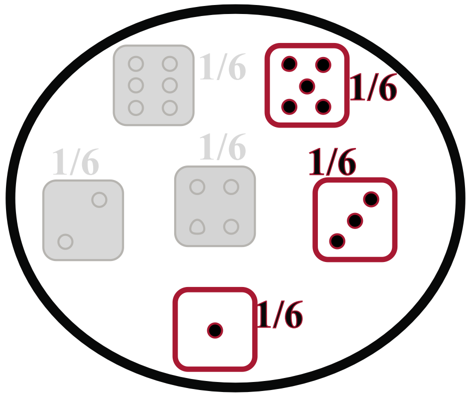
- A ***probability model*** assigns a number $P(\omega)$ to each world ω
  - E.g., $P(1) = P(2) = P(3) = P(5) = P(5) = P(6) = 1/6$.
- These numbers must satisfy
  - $0 \leq P(\omega)$
  - $\sum_{\omega \in \Omega} P(\omega) = 1$



Basic laws contd.

- An ***event*** is any subset of Ω
  - E.g., "roll < 4" is the set {1,2,3}
  - E.g., "roll is odd" is the set {1,3,5}



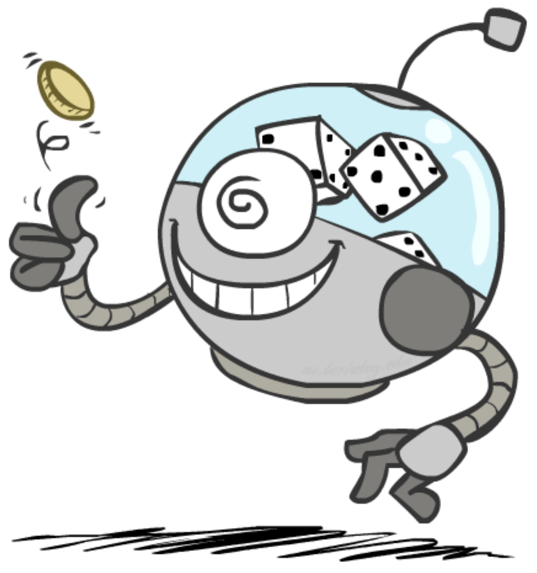


- The probability of an event is the ***sum*** of probabilities over its worlds
  - $P(A) = \sum_{\omega \in A} P(\omega)$
  - E.g., $P(roll < 4) = P(1) + P(2) + P(3) = 1/2$

Random Variables

- A random variable (usually denoted by a capital letter) is some aspect of the world about which we may be uncertain
- Formally a ***deterministic function*** of ω

- The ***range*** of a random variable is the set of possible values
  - *Odd* = Is the dice roll an odd number? $\rightarrow$ {true, false}
    - e.g. *Odd*(1)=true, *Odd*(6) = false
    - often write the event *Odd*=true as *odd*, *Odd*=false as $\neg$*odd*
  - *T* = Is it hot or cold? $\rightarrow$ {hot, cold}
  - *D* = How long will it take to get to the airport? $\rightarrow$ $[0, \infty)$



- The ***probability distribution*** of a random variable *X* gives the probability for each value *x* in its range (probability of the event *X*=*x*)
  - $P(X=x) = \sum_{\{\omega:\ X(\omega)=x\}} P(\omega)$
  - $P(x)$ for short (when unambiguous)
  - $P(X)$ refers to the entire distribution (think of it as a vector or table)

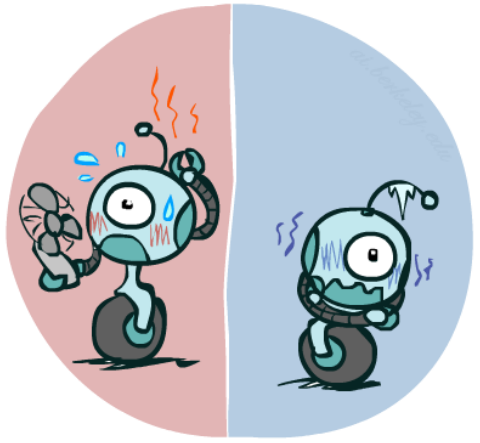
Probability Distributions

- Associate a probability with each value; sums to 1
  - Temperature:
  - Weather:
  - ***Joint distribution***

Marginal distributions

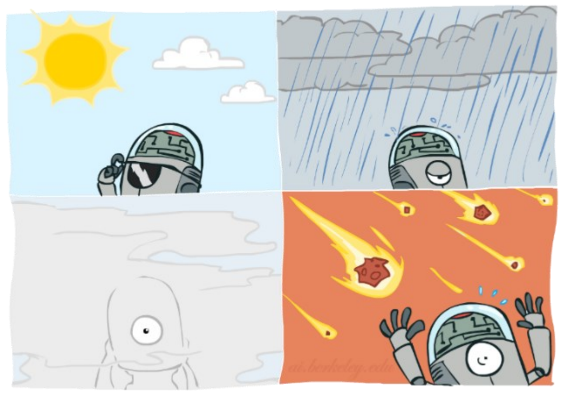
$P(T)$

T	P
hot	0.5
cold	0.5



$P(W)$

W	P
sun	0.6
rain	0.1
fog	0.3
meteor	0.0



$P(T,W)$

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

- *Can't* deduce joint from marginals
- *Can* deduce marginals from joint

Making possible worlds

- In many cases we
  - begin with random variables and their domains
  - construct possible worlds as assignments of values to all variables
- E.g., two dice rolls *$Roll_1$* and *$Roll_2$*
  - How many possible worlds?
  - What are their probabilities?
- Size of distribution for *n* variables with range size *d*?

$$d^n$$

- For all but the smallest distributions, cannot write out by hand!

Probabilistic Models

- A probabilistic model is a joint distribution over a set of random variables
- Probabilistic models:
  - (Random) variables with domains
  - Assignments are called *outcomes*
  - Joint distributions: say whether assignments (outcomes) are likely
  - Normalized: sum to 1.0
  - Ideally: only certain variables directly interact
- Constraint satisfaction problems:
  - Variables with domains
  - Constraints: state whether assignments are possible
  - Ideally: only certain variables directly interact

Distribution over T,W

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

Constraint over T,W

T	W	P
hot	sun	T
hot	rain	F
cold	sun	F
cold	rain	T

Probabilities of events

- Recall that the probability of an event is the sum of probabilities of its worlds:
  - $P(A) = \sum_{\omega \in A} P(\omega)$
- So, given a joint distribution over all variables, can compute any event probability
  - Probability that it's hot AND sunny?
  - Probability that it's hot?
  - Probability that it's hot OR not foggy?

- ***Joint distribution***

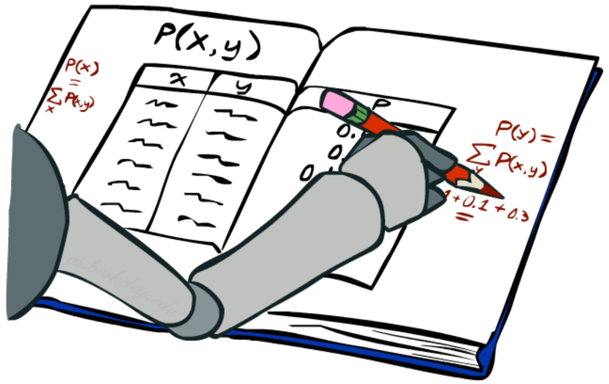
$$P(T,W)$$

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

Marginal Distributions

- Marginal distributions are sub-tables which eliminate variables
- ***Marginalization*** (***summing ou***t): Collapse a dimension by adding

$$P(X=x) = \sum_y P(X=x, Y=y)$$



		Temperature		
		hot	cold	
Weather	sun	0.45	0.15	0.60
	rain	0.02	0.08	0.10
	fog	0.03	0.27	0.30
	meteor	0.00	0.00	0.00
		0.50	0.50	

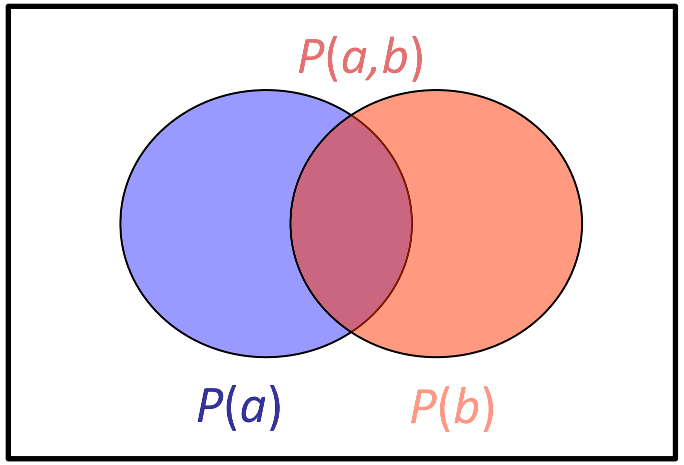
$P(W)$

$P(T)$

Conditional Probabilities

- A simple relation between joint and conditional probabilities
  - In fact, this is taken as the *definition* of a conditional probability

$$P(a \mid b) = \frac{P(a, b)}{P(b)}$$



$P(T,W)$

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

$$P(W{=}s \mid T{=}c) = \frac{P(W{=}s, T{=}c)}{P(T{=}c)} = 0.15/0.50 = 0.3$$

$= P(W{=}s, T{=}c) + P(W{=}r, T{=}c) + P(W{=}f, T{=}c) + P(W{=}m, T{=}c)$
$= 0.15 + 0.08 + 0.27 + 0.00 = 0.50$

Conditional Distributions

- Distributions for one set of variables given another set

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

P(*W* \| *T*=*h*)

hot
0.90
0.04
0.06
0.00

P(*W* \| *T*=*c*)

cold
0.30
0.16
0.54
0.00

P(*W* \| *T*)

hot	cold
0.90	0.30
0.04	0.16
0.06	0.54
0.00	0.00

Normalizing a distribution

- (Dictionary) To bring or restore to a normal condition

All entries sum to ONE

- Procedure:
  - Multiply each entry by α = 1/(sum over all entries)

$$P(W \mid T{=}c) = P(W,T{=}c)/P(T{=}c)$$
$$= \alpha\, P(W,T{=}c)$$

$P(W,T)$

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

$P(W,T{=}c)$

0.15
0.08
0.27
0.00

Normalize $\longrightarrow$

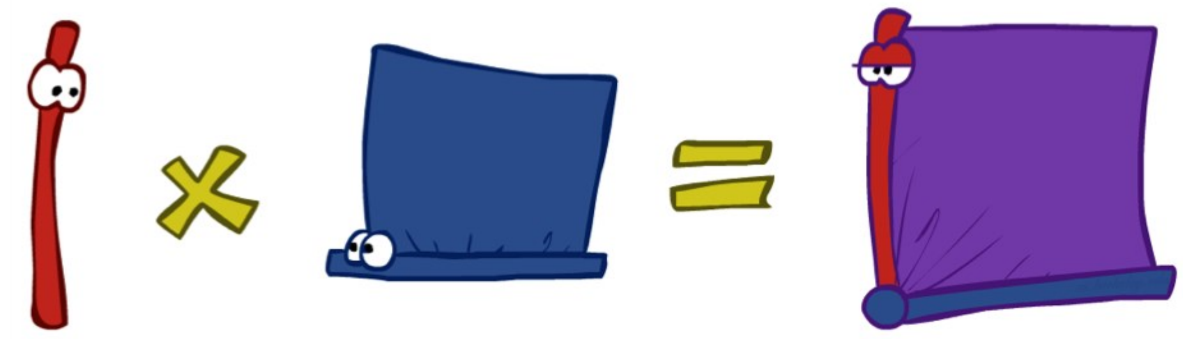
α = 1/0.50 = 2

0.30
0.16
0.54
0.00

The Product Rule

- Sometimes have conditional distributions but want the joint

$$P(a \mid b)\, P(b) = P(a, b) \quad \Longleftrightarrow \quad P(a \mid b) = \frac{P(a, b)}{P(b)}$$



The Product Rule: Example

$$P(W \mid T)\, P(T) = P(W, T)$$

$P(W \mid T)$

hot	cold
0.90	0.30
0.04	0.16
0.06	0.54
0.00	0.00

$P(T)$

T	P
hot	0.5
cold	0.5

$P(W, T)$

		Temperature	
		hot	cold
Weather	sun	0.45	0.15
	rain	0.02	0.08
	fog	0.03	0.27
	meteor	0.00	0.00

The Chain Rule

- A joint distribution can be written as a product of conditional distributions by repeated application of the product rule:
- $P(x_1, x_2, x_3) = P(x_3 \mid x_1, x_2)\, P(x_1, x_2) = P(x_3 \mid x_1, x_2)\, P(x_2 \mid x_1)\, P(x_1)$

- $P(x_1, x_2, \ldots, x_n) = \prod_i P(x_i \mid x_1, \ldots, x_{i-1})$

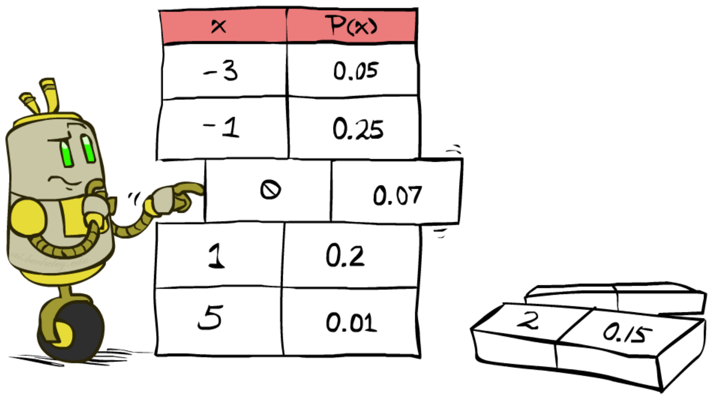
Inference by Enumeration

- Probability model $P(X_1, \ldots, X_n)$ is given
- Partition the variables $X_1, \ldots, X_n$ into sets as follows:
  - Evidence variables: $\boldsymbol{E} = \boldsymbol{e}$
  - Query variables: $\boldsymbol{Q}$
  - Hidden variables: $\boldsymbol{H}$

- We want:

$$P(\boldsymbol{Q} \mid \boldsymbol{e})$$

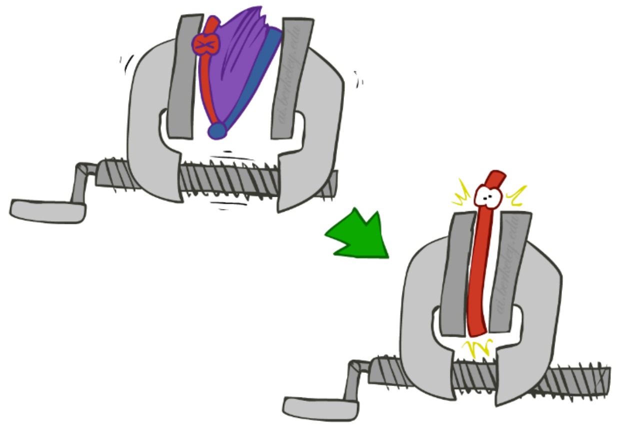
- Step 1: Select the entries consistent with the evidence



x	P(x)
-3	0.05
-1	0.25
0	0.07
1	0.2
5	0.01
2	0.15

- Step 2: Sum out $\boldsymbol{H}$ from model to get joint of query and evidence

$$P(\boldsymbol{Q}, \boldsymbol{e}) = \sum_{h} \underbrace{P(\boldsymbol{Q}, \boldsymbol{h}, \boldsymbol{e})}_{X_1, \ldots, X_n}$$



- Step 3: Normalize

$$P(\boldsymbol{Q} \mid \boldsymbol{e}) = \alpha \, P(\boldsymbol{Q}, \boldsymbol{e})$$

Inference by Enumeration

- P(S | sun)?
- 1. Enumerate options with sun
- 2. Sum out irrelevant variable(s)
- 3. Normalize
- P(S | sun) =

{summer: 0.45/(0.45+0.25), winter: 0.25/(0.45+0.25)}

Season	Temp	Weather	P
summer	hot	sun	0.35
summer	hot	rain	0.01
summer	hot	fog	0.01
summer	hot	meteor	0.00
summer	cold	sun	0.10
summer	cold	rain	0.05
summer	cold	fog	0.09
summer	cold	meteor	0.00
winter	hot	sun	0.10
winter	hot	rain	0.01
winter	hot	fog	0.02
winter	hot	meteor	0.00
winter	cold	sun	0.15
winter	cold	rain	0.20
winter	cold	fog	0.18
winter	cold	meteor	0.00

0.45

0.25

Inference by Enumeration

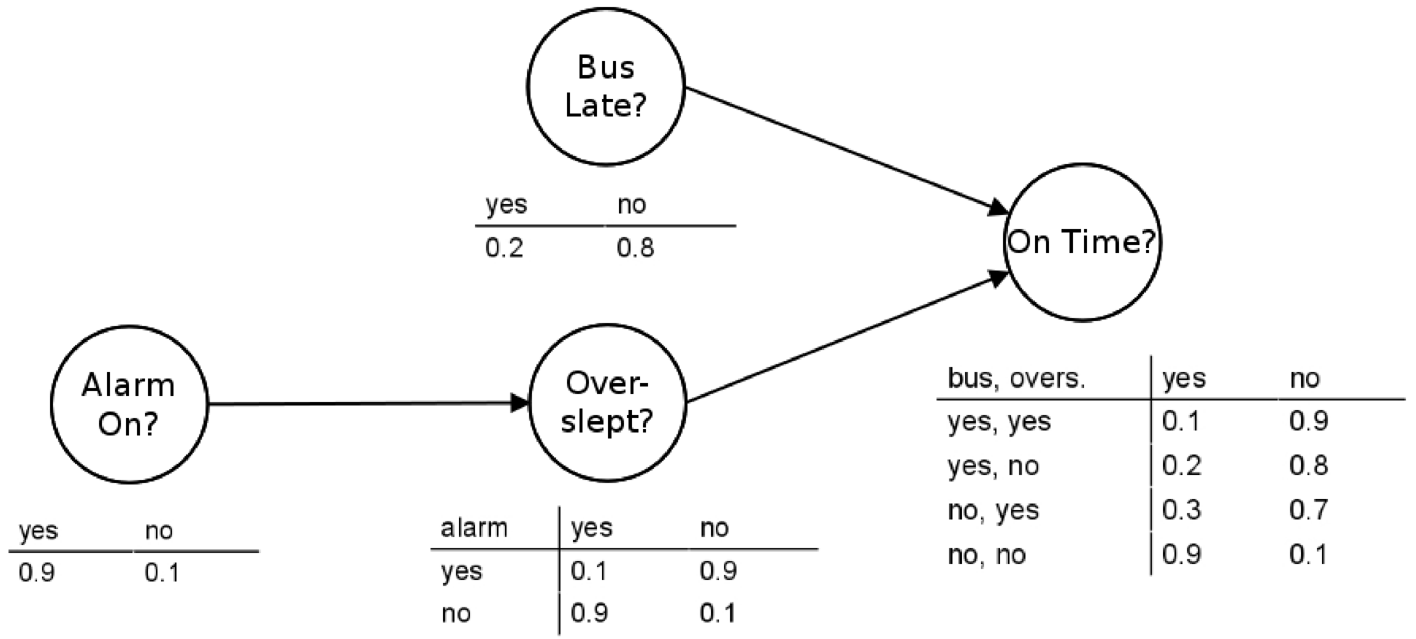
- Obvious problems:
  - Worst-case time complexity $O(d^n)$ (exponential in #hidden variables)
  - Space complexity $O(d^n)$ to store the joint distribution
  - $O(d^n)$ data points to estimate the entries in the joint distribution

Key Concepts in Probabilistic Reasoning

- Bayesian Networks
- Markov Models
- Hidden Markov Models
- Markov Decision Processes (MDPs)

Bayesian Networks

- A type of **Probabilistic Graphical Model**
  - aka. *Probabilistic Directed Acyclic Graph* {DAG}
  - **Consists of two parts**: DAG & Table of conditional probabilities.
  - Can be used to build models from data and/or expert opinion.
  - where each **node** corresponds to a **random variable** and
  - each **edge** represents the **conditional probability** for the corresponding random variables.



yes	no
0.2	0.8

yes	no
0.9	0.1

alarm	yes	no
yes	0.1	0.9
no	0.9	0.1

bus, overs.	yes	no
yes, yes	0.1	0.9
yes, no	0.2	0.8
no, yes	0.3	0.7
no, no	0.9	0.1

Applications: Diagnostics, reasoning, causal modeling, decision making under uncertainty, anomaly detection, automated insight and prediction.

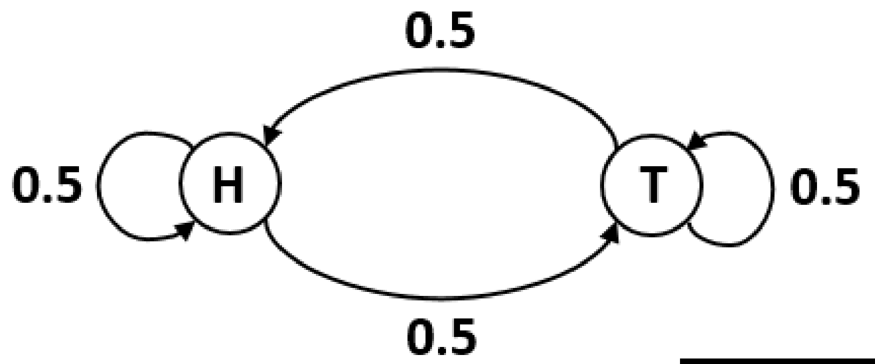
Markov Models

- Markov Models are mathematical frameworks used to model systems that transition from one state to another in a probabilistic manner.
- **Markov Chains:** A stochastic process with the Markov property, where the *probability of moving* to the *next state* **depends only** on the **current state** and *not on the sequence* of *events* that *preceded* it.

Components:

- ❑ ***States:***
  - ❑ A finite set of possible states the system can be in.
- ❑ ***Transition Probabilities:***
  - ❑ Probabilities of moving from one state to another.
  - ❑ These are usually represented in a transition matrix, where each *entry represents* the *probability* of *transitioning* from *one state to another*.

Markov Model for Coin Toss

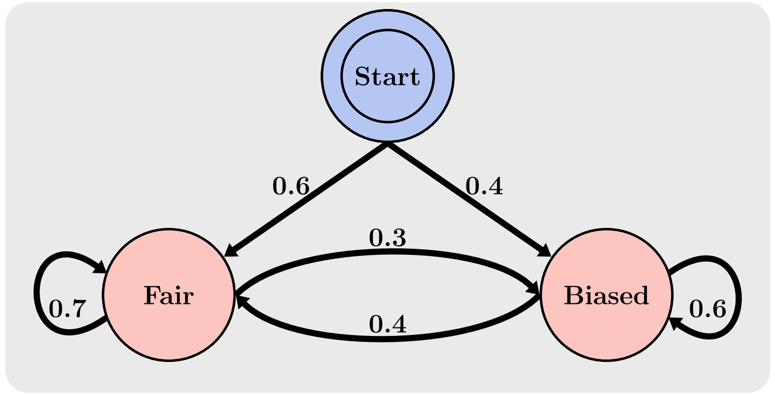


Heads	0.5	0.5
Tails	0.5	0.5

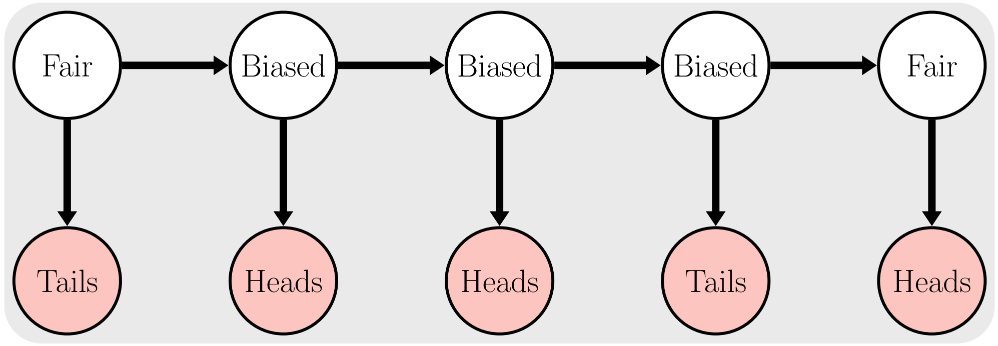
Hidden Markov Models (HMMs)

- Markov Chains where the system being modeled is assumed to be a Markov process with hidden states.
- The **system's actual state** is **not directly observable,**
  - But there is a set of observable outputs that provide information about the state.

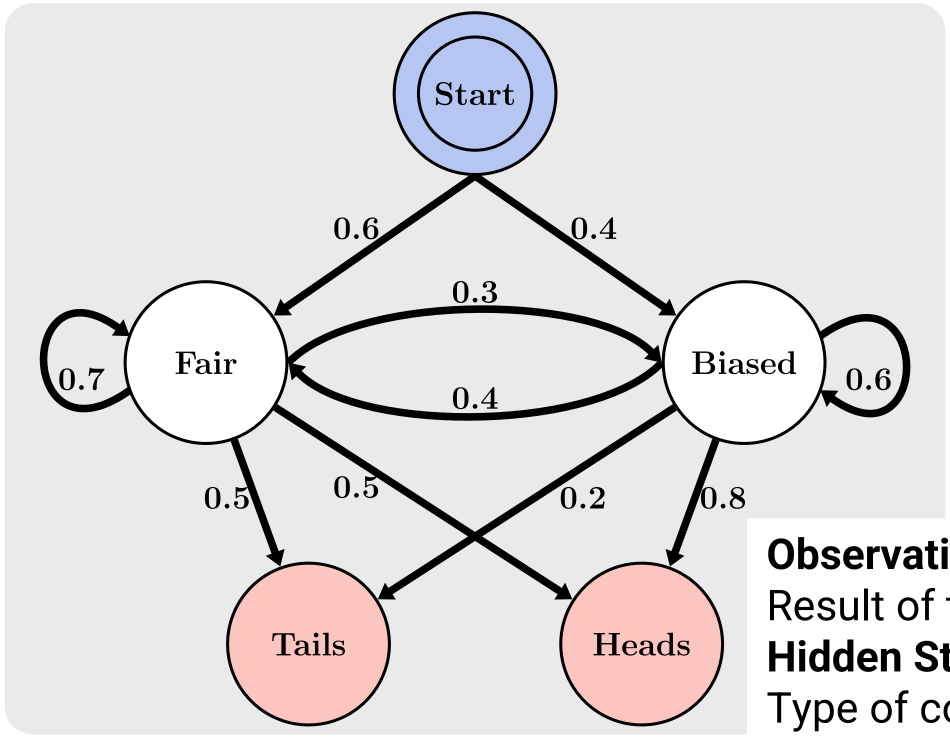
Model for Coin Toss



Hidden States



Hidden Markov Model for Coin Toss



Observation:
Result of the toss (Head/Tail)
Hidden State:
Type of coin (Fair/Biased)

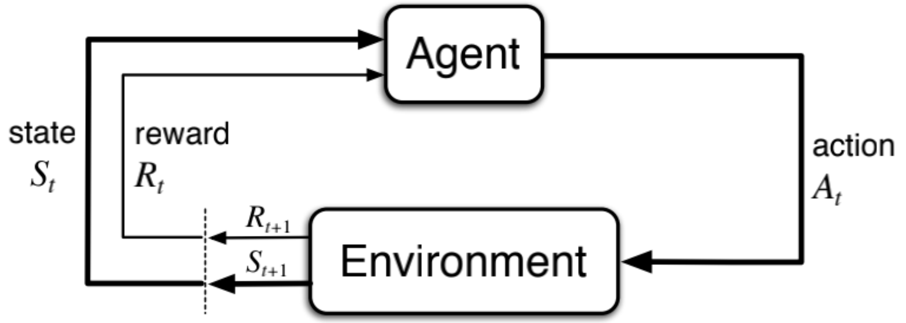
Markov Decision Processes (MDP)

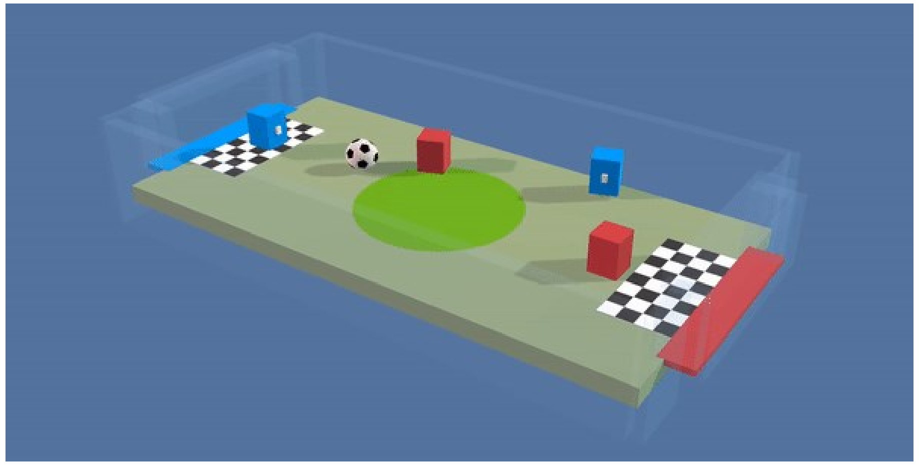
- An **extension** of **Markov Chains** that includes **actions** and **rewards**, used for **decision-making problems** under **uncertainty** where an **agent interacts** with an **environment**.

Components:

- **States**: The various *possible situations* or *configurations* of the environment.
- **Actions**: *Choices available* to the agent in *each state*.
- **Transition Probabilities**: Probabilities of *moving* from *one state* to *another given* a *particular action*.
- **Rewards**: *Immediate payoff* or *cost* received after *taking* an *action* in a particular state.
- **Policy**: A *strategy* that *defines* the *action* to *take* in *each state* to *maximize cumulative reward*.

- **Example**: *Reinforcement learning*, where an agent learns to make decisions to maximize its long-term rewards.





Techniques in Probabilistic Reasoning

- **Inference:**
  - Process of **computing** the **probability distribution** of *certain variables* given *known values of other variables*.
  - **Methods**:
    - ***Exact inference methods***: *Variable elimination* and *Junction tree algorithm*,
    - ***Approximate inference*** : *Markov Chain Monte Carlo (MCMC)* and *Belief propagation*.
- **Learning:**
  - Involves *updating* the *parameters* and *structure* of *probabilistic models* based on *observed data*.
  - **Techniques**: *Maximum likelihood estimation*, *Bayesian estimation*, and *Expectation-maximization (EM)*.
- **Decision Making:**
  - ***Utilizing probabilistic models*** to make ***decisions*** that ***maximize expected utility***.
  - Techniques involve ***computing expected rewards*** and ***selecting actions accordingly***.
  - **Methods:** Partially observable Markov decision process (POMDP).

Need of Probabilistic Reasoning in AI

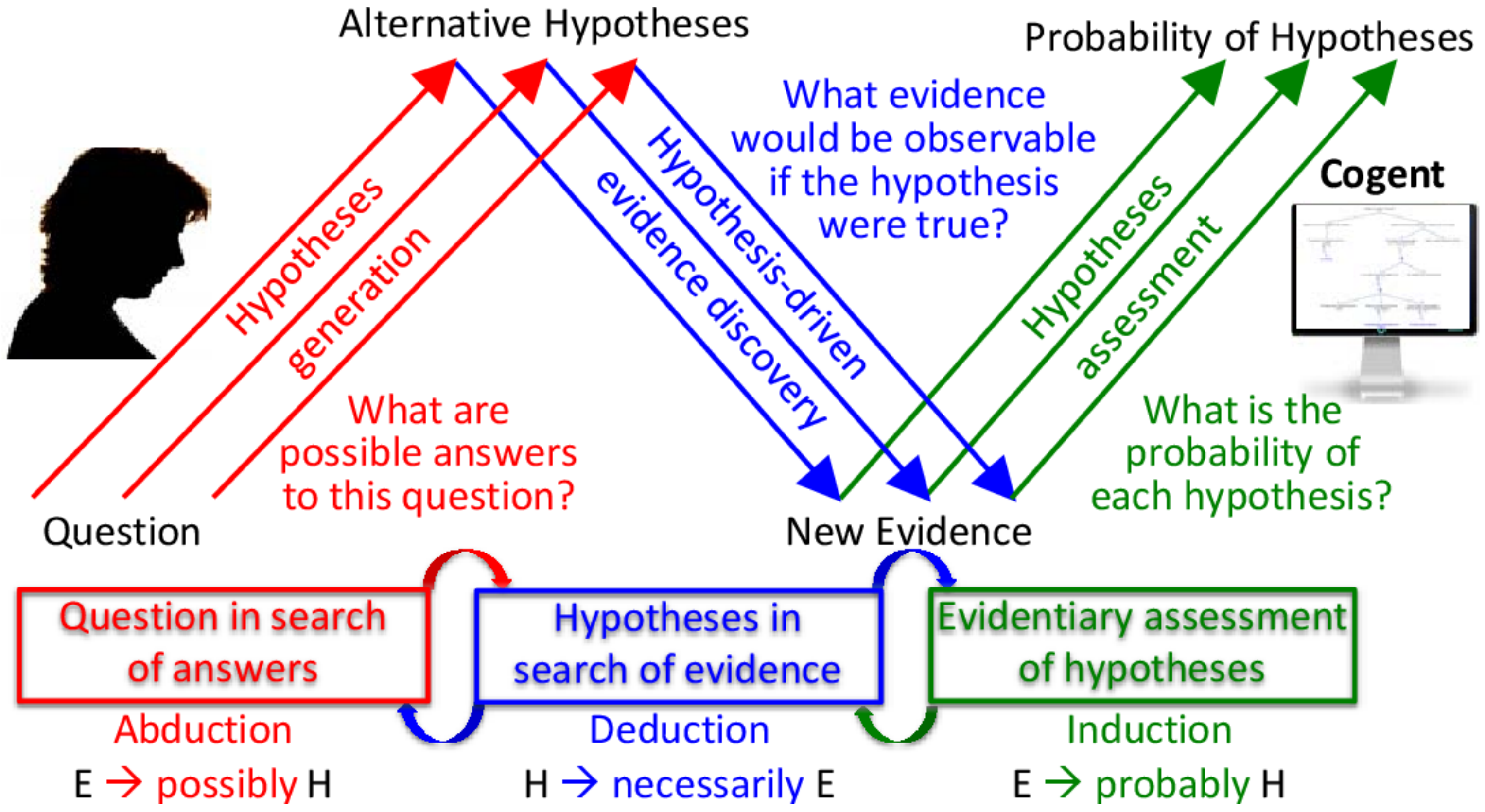
- **Quantifying Uncertainty:**
  - Probabilistic reasoning does not shrink from uncertainty.
  - It turns to the tools of probability theory to ***represent uncertainty*** by ***attaching degrees of likelihood***.

  - **For example**: ***Prediction of Rain***
    - Instead of a simple "**true**" or "**false**" to **whether it will rain tomorrow**,
    - Probabilistic reasoning might assign a 60% chance that it will.

Need of Probabilistic Reasoning in AI

- **Reasoning with Evidence:**
  - AI systems cannot enjoy the luxury of making decisions in isolation.
  - They have to consider the available evidence and act accordingly to help refine the probabilities.

- ***For example:***
  - The probability for a rainy day can be refined to increase to 80% if dark clouds come in the afternoon.


Alternative Hypotheses
Probability of Hypotheses
Hypotheses generation
Hypothesis-driven evidence discovery
What evidence would be observable if the hypothesis were true?
Hypotheses assessment
Cogent
What are possible answers to this question?
What is the probability of each hypothesis?
Question
New Evidence
Question in search of answers
Hypotheses in search of evidence
Evidentiary assessment of hypotheses
Abduction
Deduction
Induction
E → possibly H
H → necessarily E
E → probably H

Probabilistic Inference

- **Probabilistic inference**: compute a desired probability from a probability model
  - Typically for a ***query variable*** given ***evidence***
  - E.g., P(airport on time | no accidents) = 0.90
  - These represent the agent's *beliefs* given the evidence

- Probabilities change with new evidence:
  - P(airport on time | no accidents, 5 a.m.) = 0.95
  - P(airport on time | no accidents, 5 a.m., raining) = 0.80
  - Observing new evidence causes ***beliefs to be updated***

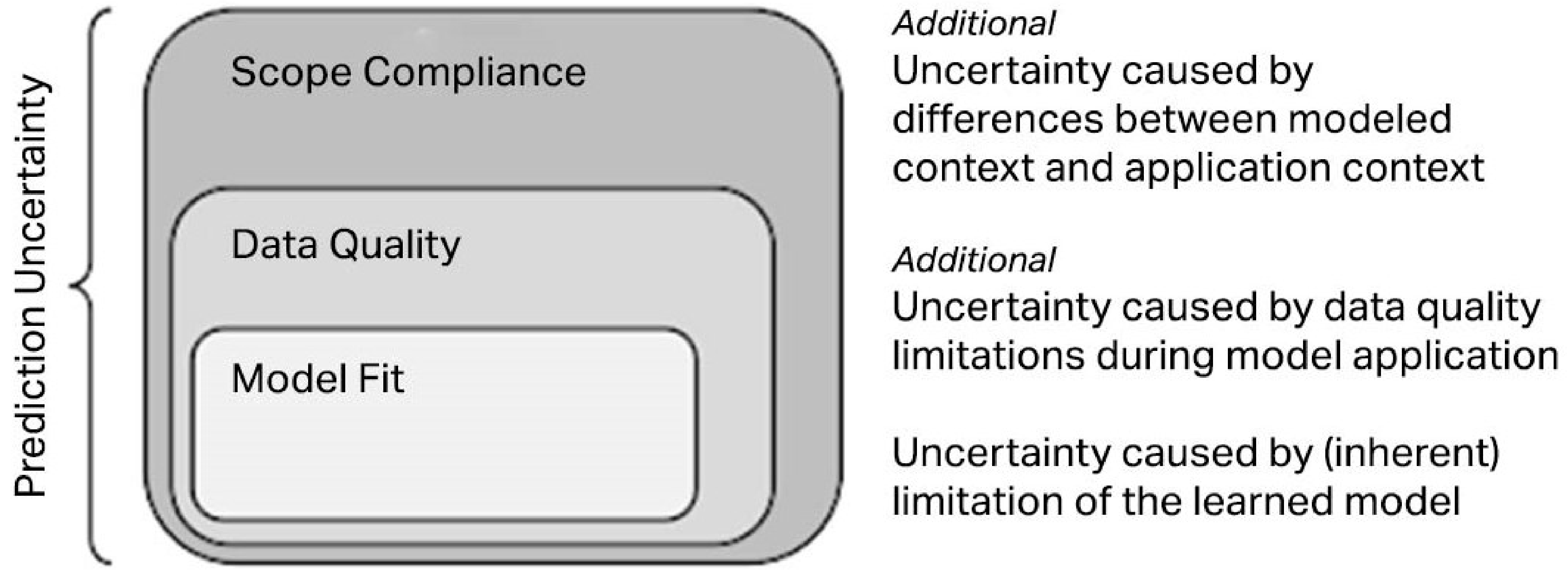
Need of Probabilistic Reasoning in AI

- **Based on Past Experience:**
  - AI systems can **learn** from **past experiences**.
  - Probabilistic reasoning factors in the **prior knowledge** of the **nature of decisions**.
  - ***For example***, an AI system that was *trained* in the *past on historical weather data* in your location might, therefore, consider seasonal trends when calculating the probability of rain.

- **Effective Decision-Making:**
  - Probabilistic reasoning will also enable AI systems to make effective and *well-informed decisions* based on *quantified uncertainty*, *evidence*, and *prior knowledge*.
  - **Returning to our maze analogy**, the AI would be able to ***actually weigh*** the ***probability*** of ***different paths***, given the *map* at *each point* in the *maze* and *whatever it's found its way through, making its reaching the goal much more likely*.

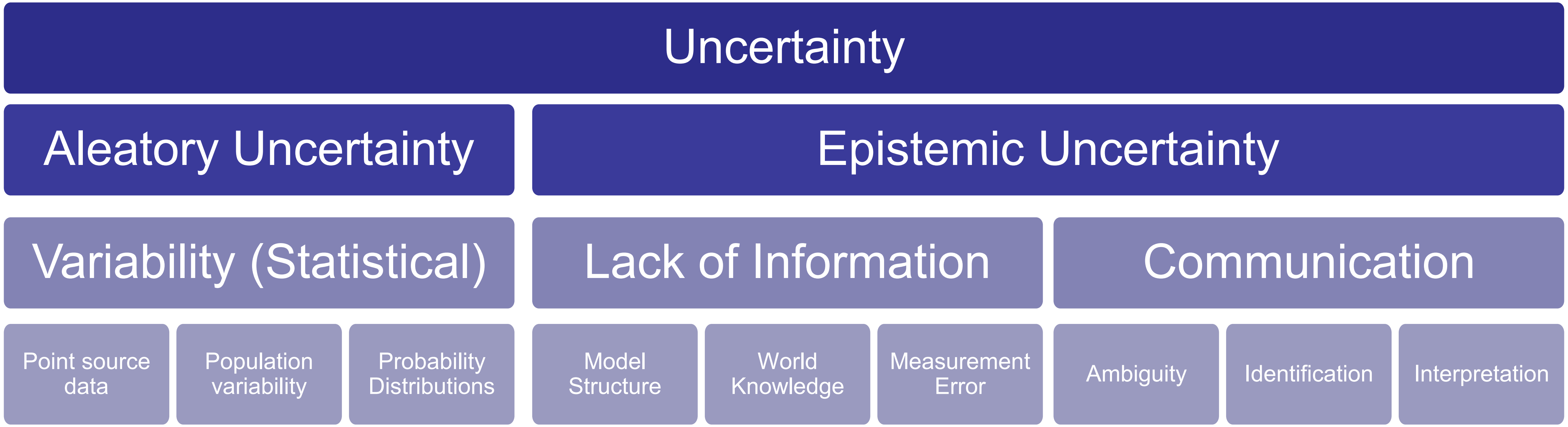
Importance of Quantifying Uncertainty

- **Uncertainty in data** and **decision-making** is **crucial** in **AI systems**,
  - As it affects **safety** and **accuracy** in applications like ***self-driving cars*** and ***medical diagnosis***, where models assign probabilities to outcomes.


Prediction Uncertainty
Scope Compliance
Data Quality
Model Fit
Additional
Uncertainty caused by differences between modeled context and application context
Additional
Uncertainty caused by data quality limitations during model application
Uncertainty caused by (inherent) limitation of the learned model

Onion layer model of uncertainty in AI/ML application outcomes.

Uncertainty in AI



Aleatory Uncertainty

- Rises from *inherent randomness* or *variability* in *a system* or *process*.
- Associated with *natural* or *stochastic processes*, where *outcomes* are *governed* by *chance rather* than *determinable* patterns.
- **Addressed** through *statistical methods* and *probabilistic models* to estimate and manage the variability in outcomes.

Epistemic Uncertainty

- Uncertainty due to *lack of understanding* or *incomplete information* about a *system*, which might be reduced with more research or better data.

Methods for
Uncertainty Quantification

Knowledge representation in uncertainty

- **Bayesian Methods:**
  - Techniques incorporate prior knowledge and update beliefs based on **new data**,
  - Allowing for the ***estimation*** of both *aleatoric* (inherent variability) and *epistemic* (model-related) ***uncertainties***.

- **Deep Evidential Regression:**
  - Quantifies uncertainty without additional computational burden after training, making it efficient for real-time applications.

Knowledge Representation in Uncertainty

- An **uncertain domain** in AI refers to a ***field*** or ***environment*** where the ***information available*** is *incomplete*, *ambiguous*, *noisy*, or *inherently unpredictable*.
  - Unlike deterministic domains where outcomes can be predicted with certainty given the inputs,
  - *Uncertain domains* require *AI systems* to *handle* and *reason* about *uncertainty* in a ***structured manner***.
- **Characteristics of Uncertain Domains**
  - *Incomplete Information*: System does not have access to all the data required to make a fully informed decision.
  - *Ambiguity*: Information might be unclear or open to multiple interpretations.
  - *Noise*: Data might be corrupted or imprecise due to measurement errors or external factors.
  - *Stochastic Processes*: The environment might involve random processes or events.
- **Importance of Uncertain Domains**
  - Make informed decisions based on probabilistic reasoning.
  - Adapt to new information and changing environments.
  - Provide robust and reliable performance in complex scenarios.

Methods for Knowledge Representation in Uncertainty

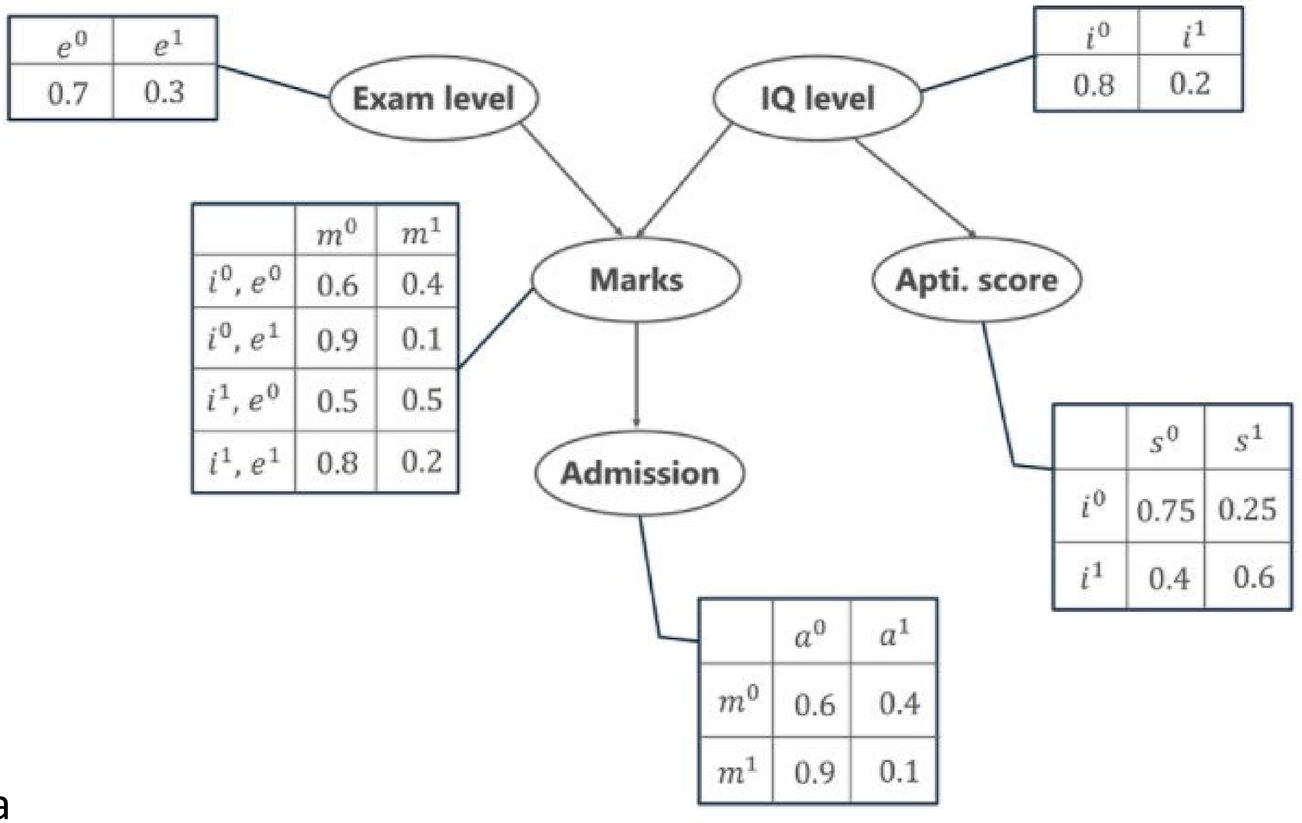
- **Probabilistic Reasoning**
  - Involves representing knowledge using probability theory to manage uncertainty.
  - Widely used in AI for tasks such as diagnosis, prediction, and decision-making under uncertainty.

- **Bayesian Networks**
  - BNs are graphical models that represent the probabilistic relationships among a set of variables.
  - **Node** in a BN represents a variable,
  - **Edges** represent conditional dependencies.
  - BNs allow for efficient computation of posterior probabilities given observed evidence.

Methods for Knowledge Representation in Uncertainty

Bayesian Networks : Example

- In this example, let us imagine that we are given the ***task of modeling a student's marks (m) for an exam he has just given.***
- From the given Bayesian Network Graph, we see that the marks depend upon two other variables. They are,
  - **Exam Level (e)–** This discrete variable denotes the **difficulty of the exam** and has **two values** (*0 for easy and 1 for difficult*)
  - **IQ Level (i) -** Represents the Intelligence Quotient level of the student and is also discrete (0 for low and 1 for high)
- Additionally, the IQ level of the student also leads us to another variable, which is the **Aptitude Score** of the student (s).
- Now, with marks the student has scored, he can secure admission to a particular university.
- The probability distribution for getting admitted (a) to a university is also given.



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

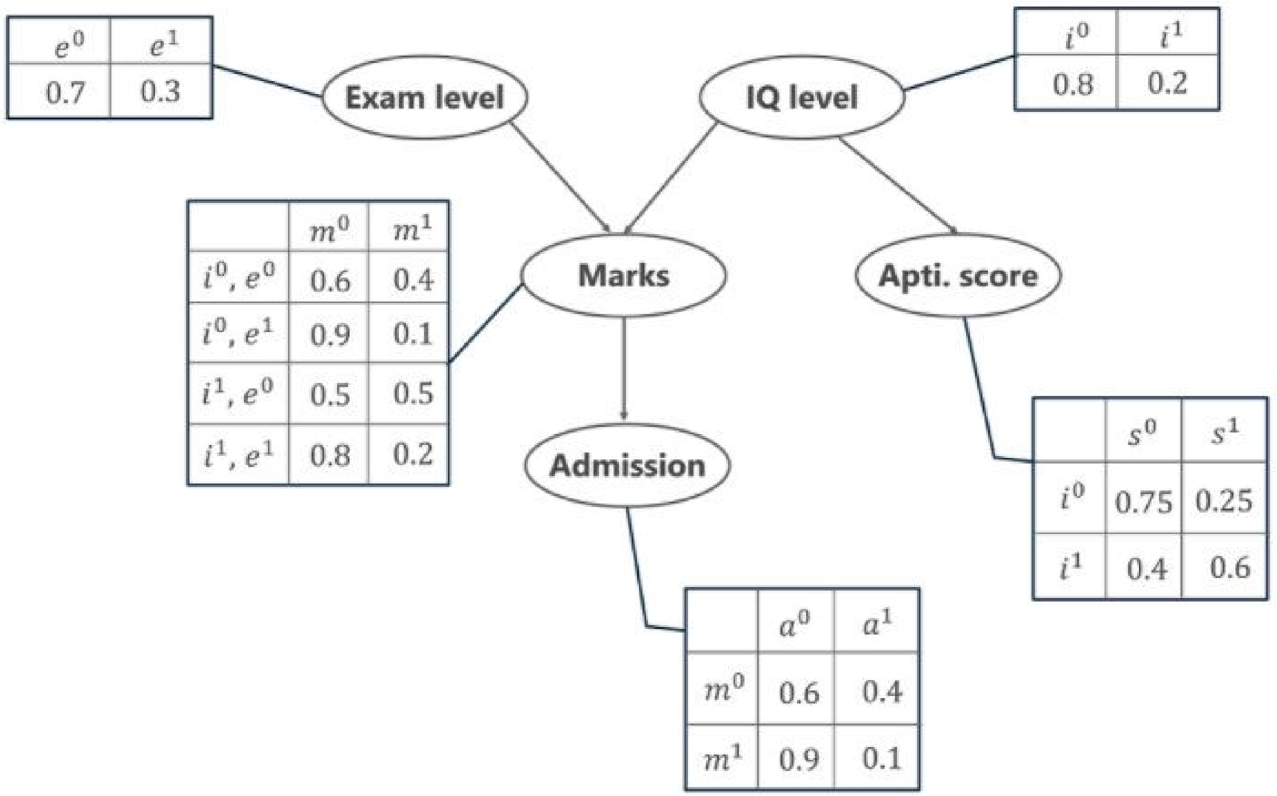
	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

Methods for Knowledge Representation in Uncertainty

- **Bayesian Networks : Example**
- In graphs, tables represent the probability distribution values of the given 5 variables. These tables are called the Conditional Probabilities Table or CPT.

- **Properties of the CPT given below –**
  - Sum of the CPT values in each row must be equal to 1.
  - If a variable is Boolean has k Boolean parents, then in the CPT it has 2^K probability values.

- Listing all the possible events that are occurring in the above-given table.
  - Exam Level (e); IQ Level (i); Aptitude Score (s); Marks (m); Admission (a)



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

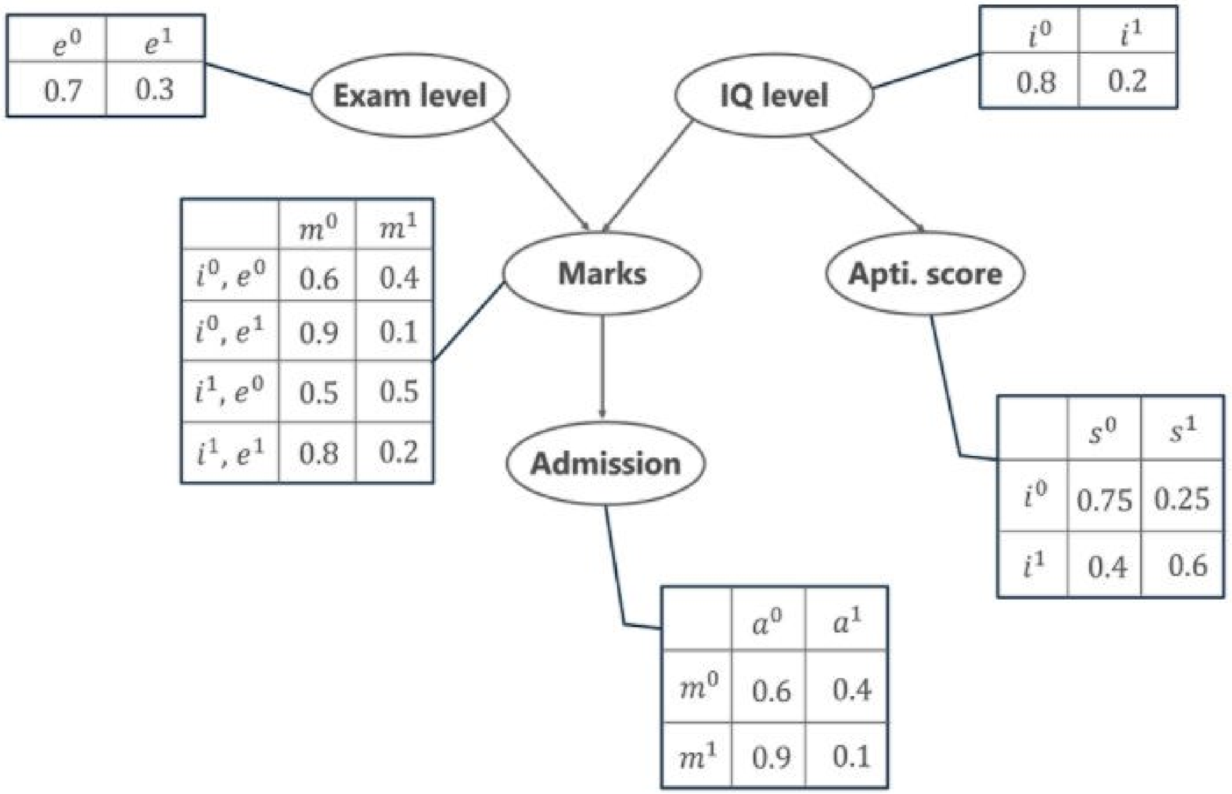
	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

Methods for Knowledge Representation in Uncertainty

- **Bayesian Networks : Example**
- Calculate the Joint Probability Distribution of the 5 variables the formula is given by,
- $P[a, m, i, e, s] = P(a \mid m) \,.\, P(m \mid i, e) \,.\, P(i) \,.\, P(e) \,.\, P(s \mid i)$
- From the above formula,
  - **P(a | m)** denotes the conditional probability of the student getting admission based on the marks he has scored in the examination.
  - **P(m | i, e)** represents the marks that the student will score given his IQ level and difficulty of the Exam Level.
  - **P(i)** and **P(e)** represent the probability of the IQ Level and the Exam Level.
  - **P(s | i)** is the conditional probability of the student's Aptitude Score, given his IQ Level.
  - With the following probabilities calculated, we can find the Joint Probability Distribution of the entire Bayesian Network.



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

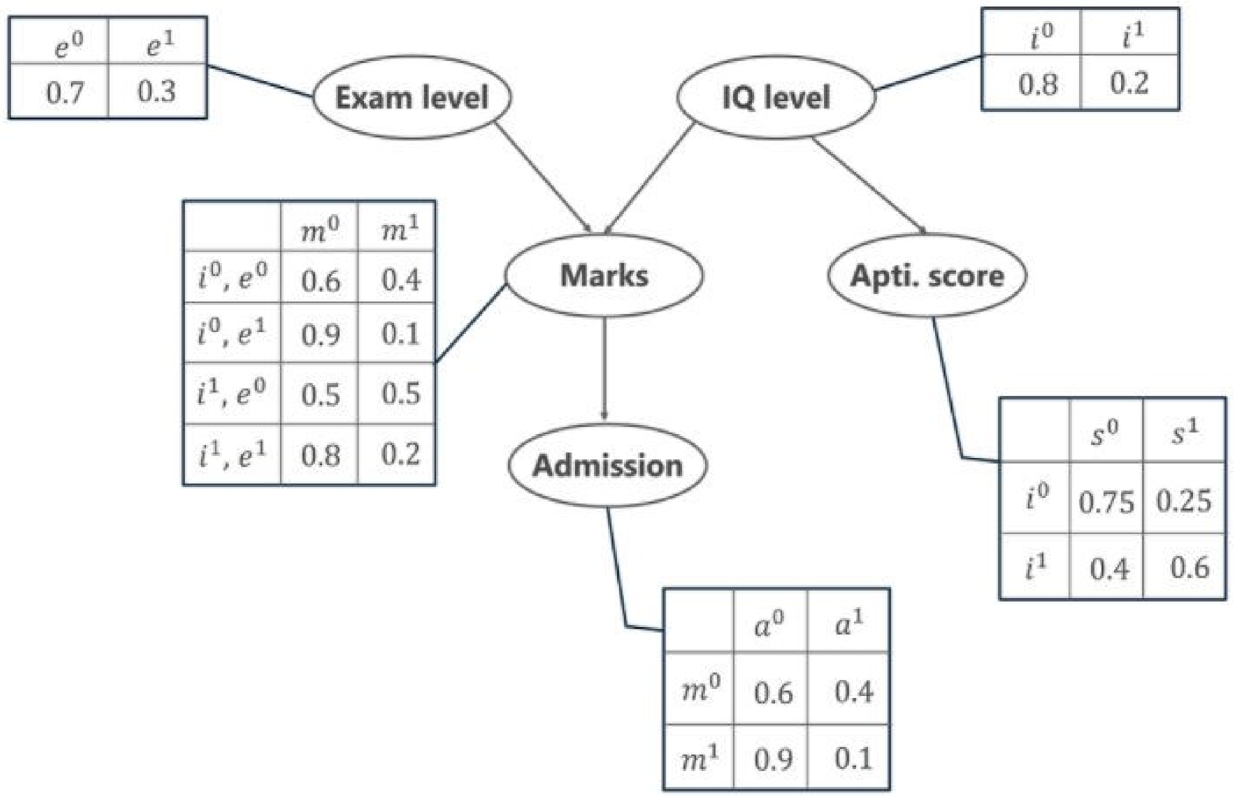
Methods for Knowledge Representation in Uncertainty

- **Bayesian Networks : Example**
- **Calculation of Joint Probability Distribution**
- **Case 1:** Calculate the probability that in *spite of the exam level being difficult, the student having a low IQ level and a low Aptitude Score, manages to pass the exam and secure admission to the university.*

- From the above word problem statement, the Joint Probability Distribution can be written as below,

- **P[a=1, m=1, i=0, e=1, s=0]**
- From the Conditional Probability tables,.

- **P[a=1, m=1, i=0, e=0, s=0] = P(a=1 | m=1) . P(m=1 | i=0, e=1) . P(i=0) . P(e=1) . P(s=0 | i=0)**
- **= 0.1 * 0.1 * 0.8 * 0.3 * 0.75**
- **= 0.0018**



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

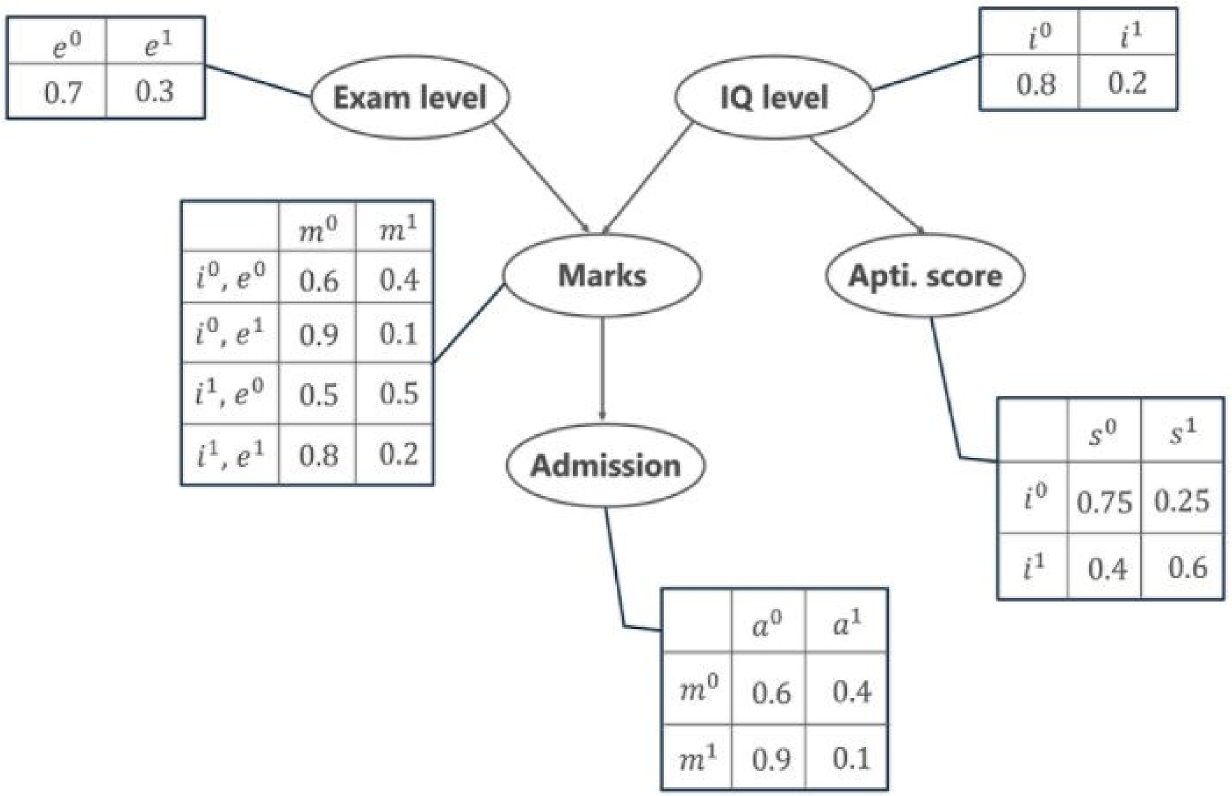
	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

Methods for Knowledge Representation in Uncertainty

- **Bayesian Networks : Example**
- **Calculation of Joint Probability Distribution**
- **Case 2:** Calculate the probability *that the student has a High IQ level and Aptitude Score, the exam being easy yet fails to pass and does not secure admission to the university.*
- The formula for the JPD is given by
- **P[a=0, m=0, i=1, e=0, s=1]**
- Thus,
- **P[a=0, m=0, i=1, e=0, s=1] = P(a=0 | m=0) . P(m=0 | i=1, e=0) . P(i=1) . P(e=0) . P(s=1 | i=1)**
- = 0.6 * 0.5 * 0.2 * 0.7 * 0.6
- = 0.0252



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

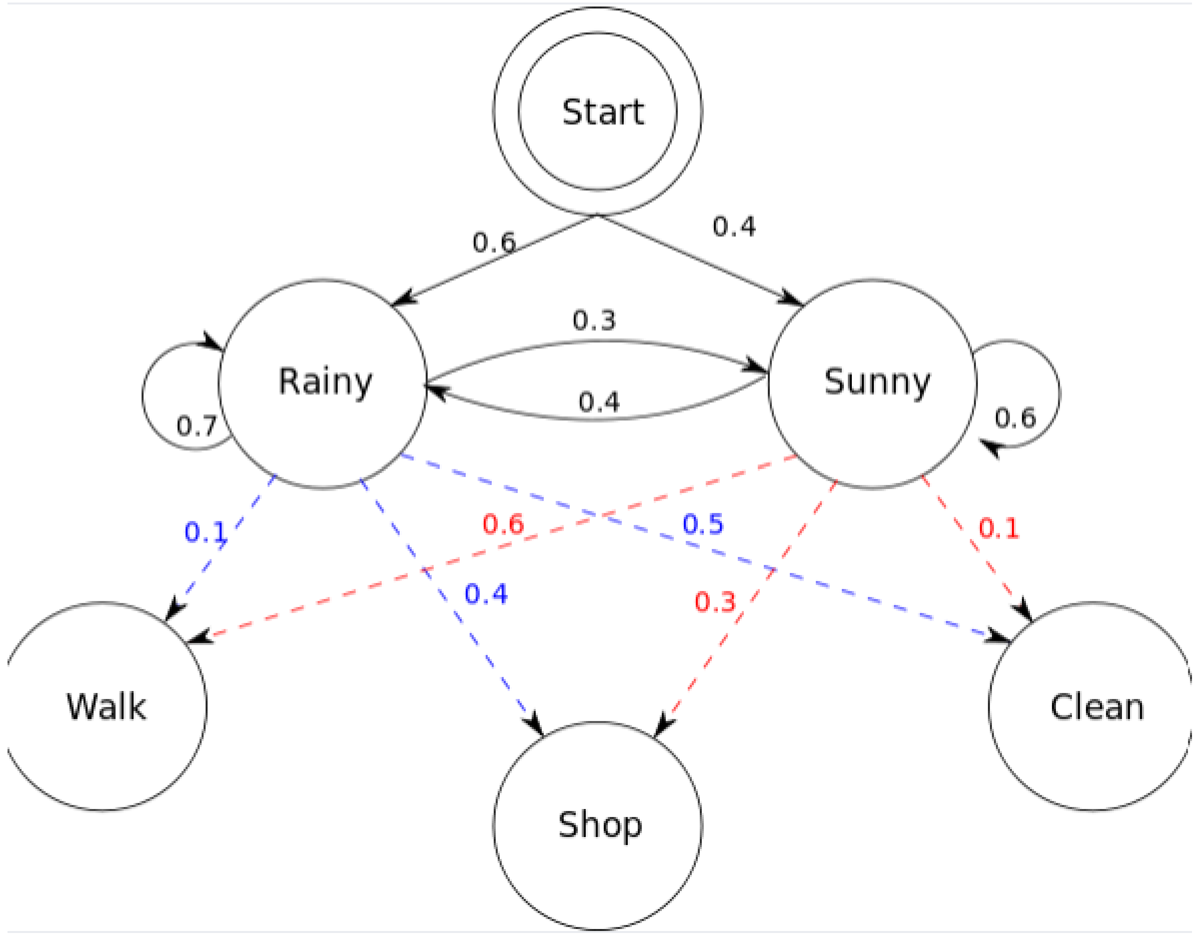
	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

Methods for Knowledge Representation in Uncertainty

- **Hidden Markov Models**
  - ***Model time series data*** where the system being modeled is assumed to be ***a Markov process*** with ***hidden states***.
  - HMMs are widely used in *speech recognition*, *bioinformatics*, and *other sequential data applications*.
  - HMM- future is independent of the past given the present

- **Two hidden states** such as **rainy** and **sunny**.
- These states are hidden because *what is observed* as the *process output* is *whether* the *person* is *shopping*, *walking*, or *cleaning*.
- The sequence of observations is shop, walk, and clean.

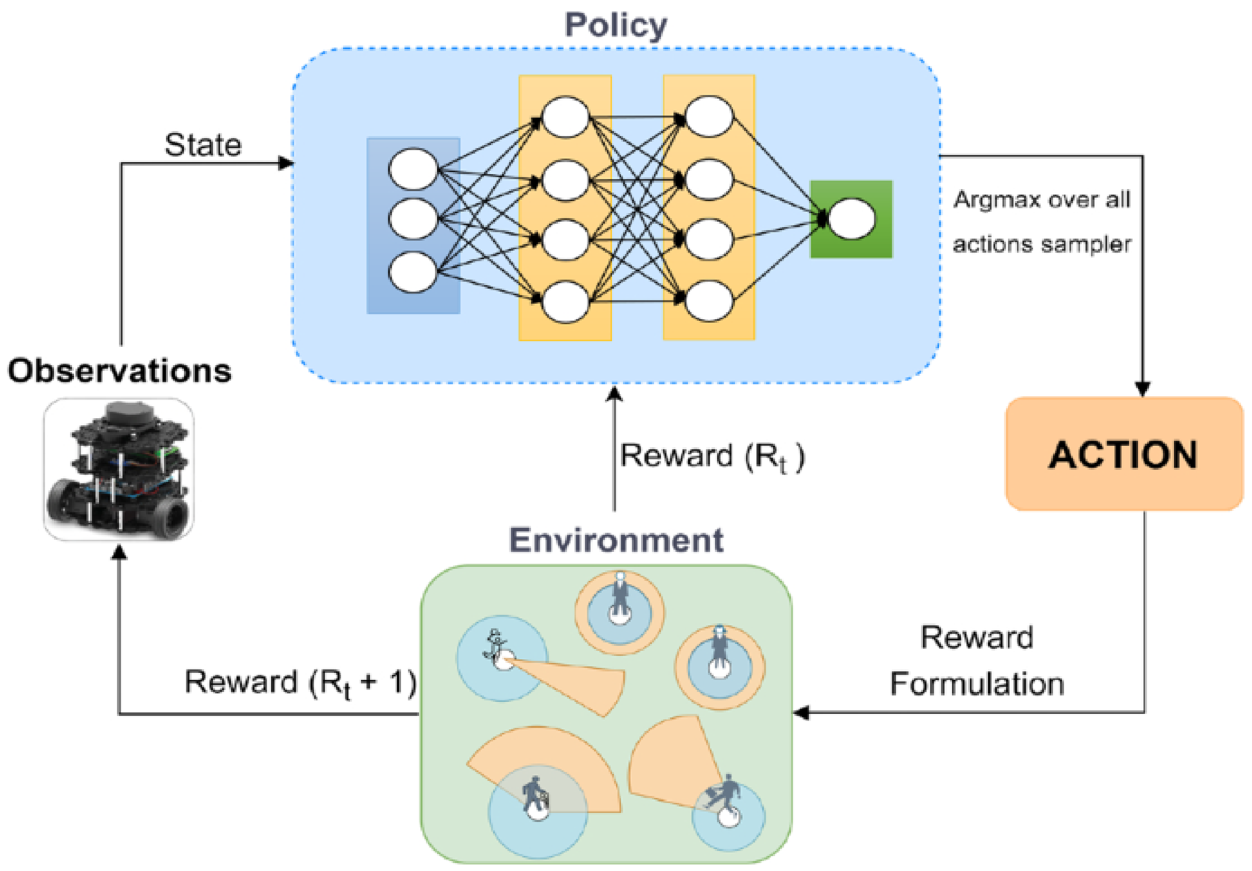
Scenario: weather, the hidden variable, can be hot, mild or cold and the observed variables are the type of actions



Methods for Knowledge Representation in Uncertainty

Markov Decision Processes

- A framework for **modeling decision-making** in **environments** with **stochastic dynamics**.
- MDPs consist of *states*, *actions*, *transition probabilities*, and *rewards*, enabling the computation of optimal policies for decision-making.



Scenario: An autonomous robot navigating a grid world can use an MDP to determine the optimal path to its destination while accounting for uncertain movements and rewards.

Methods for Knowledge Representation in Uncertainty

Markov Decision Processes

- **Scenario**: An autonomous robot navigating a grid world.
- **Objective**: Find the optimal path to its destination.

MDP Components for Robot Navigation:

States (S):

- ❑ Each grid cell is a state.
- ❑ The robot's position on the grid defines the current state (e.g., (x, y) coordinates).

Actions (A):

- ❑ At each state, the robot can take actions like move up, down, left, right.
- ❑ *Actions can result in uncertain movements (e.g., intended move up but slipping to the side).*

Transition Probabilities (P):

- ❑ The probability of moving to a new state given the current state and action.
- ❑ Uncertainty is modeled by assigning probabilities (*e.g., 80% chance of moving up, 20% chance of slipping*).

Rewards (R):

- ❑ The robot receives rewards or penalties for entering certain states.
- ❑ *Example*: Reaching the goal gives a high reward; obstacles or falling off the grid yield penalties.

Policy (π):

- ❑ A policy is a strategy that defines the action the robot should take in each state.
- ❑ The goal of the MDP is to *find an optimal policy (π*) that maximizes total expected rewards*.

Value Function (V):

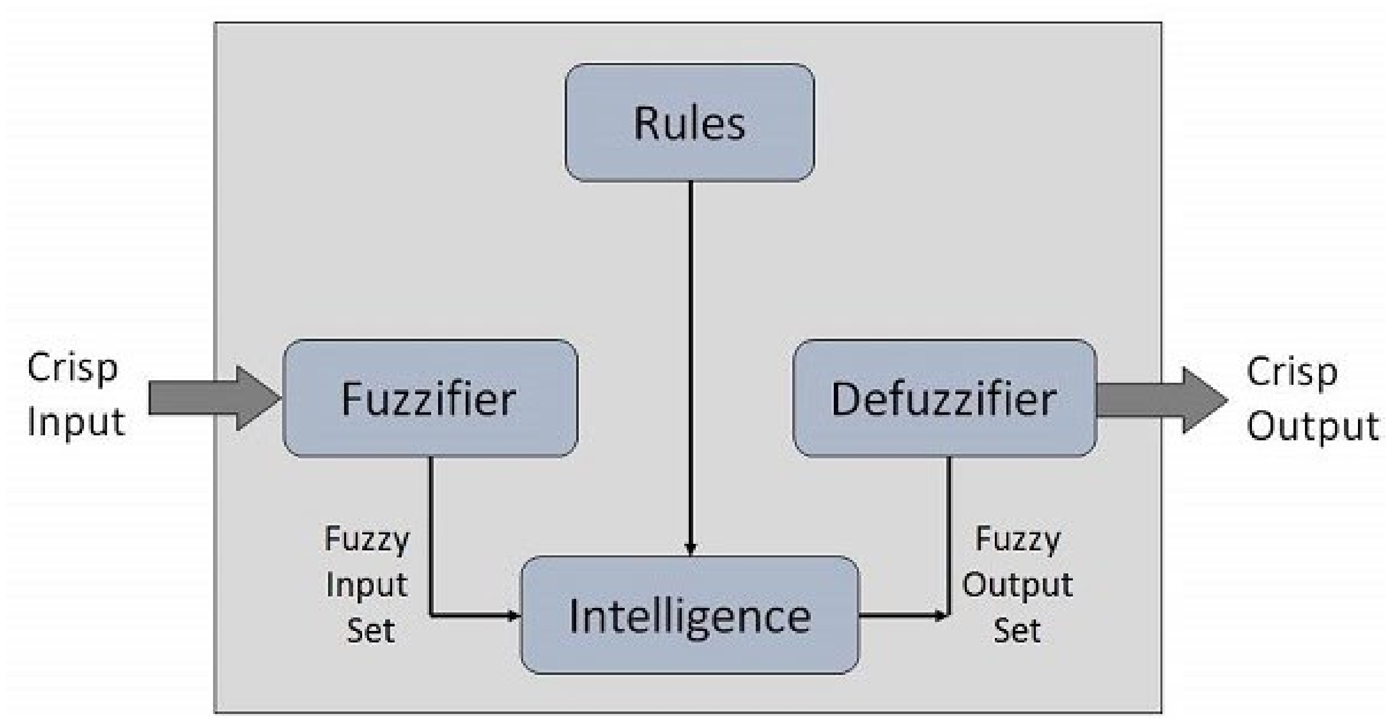
- ❑ The value of each state represents the maximum expected reward the robot can achieve from that state onwards.
- ❑ Helps the robot decide the best path to the goal.

Methods for Knowledge Representation in Uncertainty

- **Fuzzy Logic**
  - An approach to reasoning that deals with **approximate** rather than *fixed* and *exact values*.
  - Unlike traditional binary logic, *fuzzy logic* variables can have a *truth value* that *ranges between 0 and 1*, representing the **degree of truth**.

Fuzzy Sets and Membership Functions

- ❑ ***Fuzzy sets*** allow for the representation of concepts with vague boundaries.
- ❑ Each element in a fuzzy set has a ***membership value*** indicating its degree of belonging to the set.

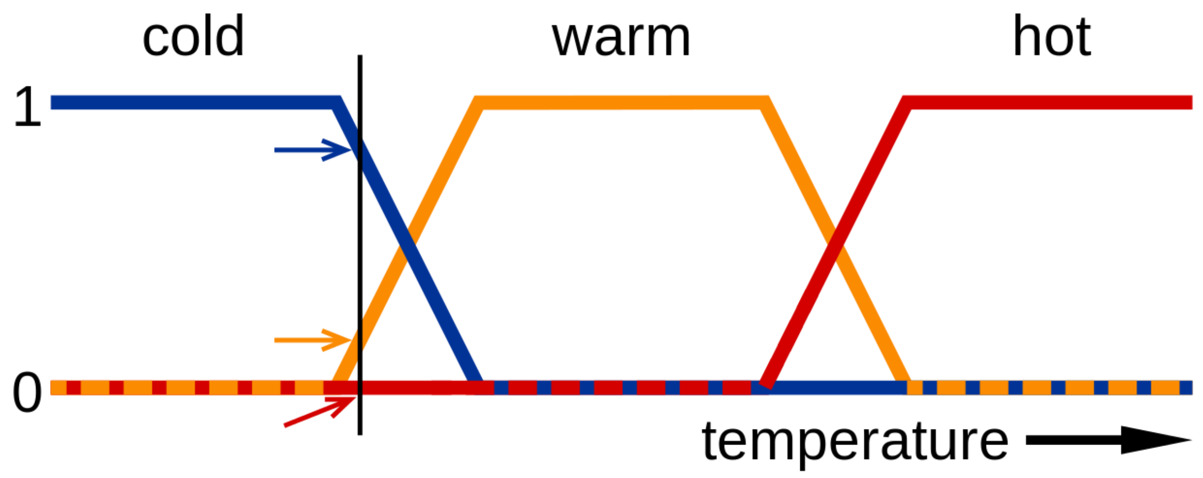


Methods for Knowledge Representation in Uncertainty

- **Fuzzy Logic**
  - **Fuzzy Rules and Inference**
    - Fuzzy rules define the *relationships between fuzzy variables using if-then statements*.
    - *Fuzzy inference systems apply* these *rules* to *input data* to *derive conclusions*.

Fuzzy Logic Example:

- In a temperature control system, the concept of "cold" can be represented as a fuzzy set with a membership function assigning values between 0 (not cold) and 1 (completely cold) to different temperatures.



Fuzzy Rules and Inference

A fuzzy rule for a temperature control system might be: "*If the temperature is high, then reduce the heater power*."

Methods for Knowledge Representation in Uncertainty

- **Dempster-Shafer Theory**
  - A mathematical framework for modeling uncertainty without the need for **precise** probabilities.
  - It allows for the *combination of evidence* from *different sources* to *calculate the degree of belief* (or plausibility) for *various hypotheses*.

- **Uncertainty in this model is given by:-**
  - Consider all possible outcomes.
  - Belief will lead to belief in some possibility by bringing out some evidence. (What is this supposed to mean?)
  - Plausibility will make evidence compatible with possible outcomes.

Methods for Knowledge Representation in Uncertainty

- **Dempster-Shafer Theory**
  - Example: *Let us consider a room where four people are present, A, B, C, and D. Suddenly the lights go out and when the lights come back, B has been stabbed in the back by a knife, leading to his death. No one came into the room and no one left the room. We know that B has not committed suicide. Now we have to find out who the murderer is.*

  - *To solve these there are the following possibilities:*
    - Either {A} or {C} or {D} has killed him.
    - Either {A, C} or {C, D} or {A, D} have killed him.
    - Or the three of them have killed him i.e.; {A, C, D}
    - None of them have killed him {o} (let's say).

Methods for Knowledge Representation in Uncertainty

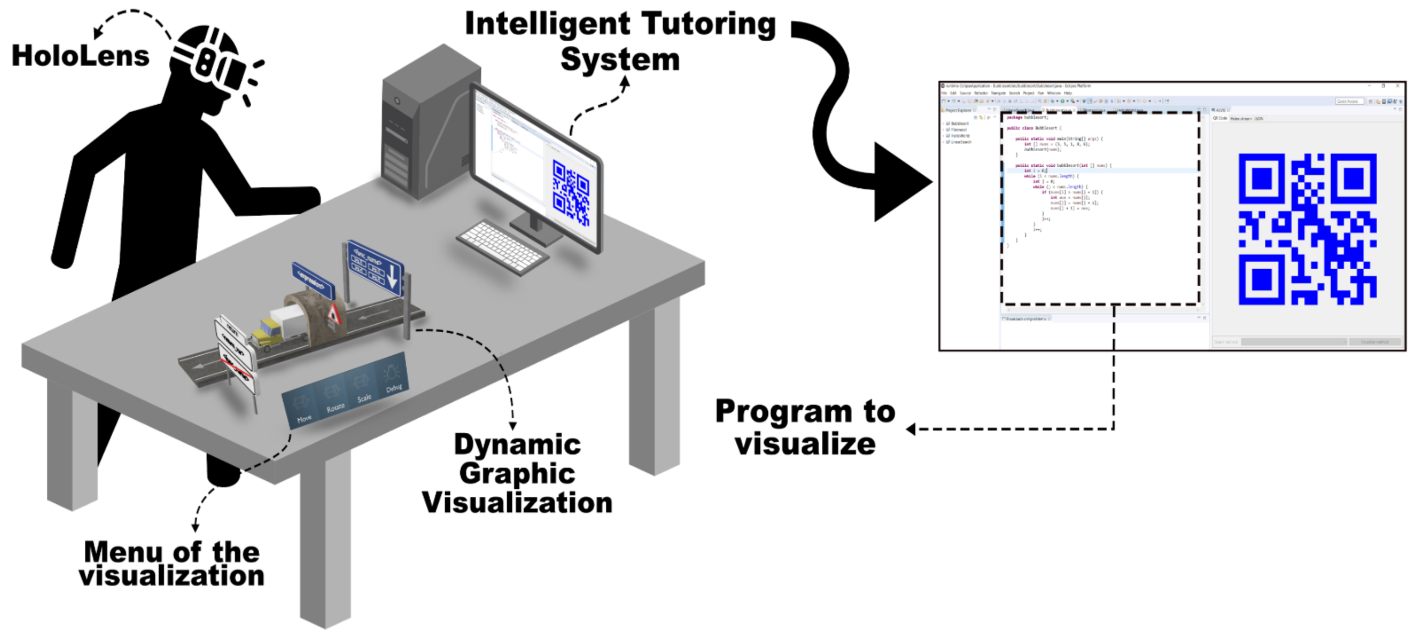
- **Dempster-Shafer Theory**
  - There will be possible evidence by which we can find the murderer by the measure of plausibility.
  - ***Using the above example we can say:***
    - Set of possible conclusion (P): {p_1, p_2….p_n}
    - where ***P*** is a *set* of *possible conclusions* and cannot be exhaustive, i.e. at least one (p) I must be true.
    - **(p)I** must be mutually exclusive.
    - Power Set will contain 2^n elements where n is the number of elements in the possible set.
  - ***For e.g.:-***
    - If P = { a, b, c}, then Power set is given as
    - $\{o, \{a\}, \{b\}, \{c\}, \{a, d\}, \{d\ , c\}, \{a, c\}, \{a,\ \ c\ , d\ \}\}$ = 2^3 elements.

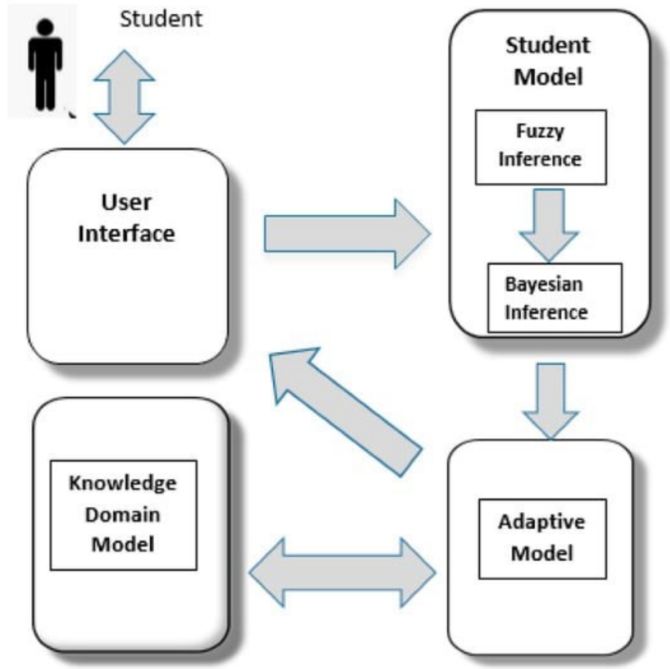
Methods for Knowledge Representation in Uncertainty

- **Dempster-Shafer Theory**
  - **Mass function m(K):**
    - It is an interpretation of **m ({K or B})**
    - i.e; it means there is *evidence for {K or B}* which *cannot be divided among more specific beliefs* for K and B.
  - **Belief in K:**
    - **Belief in element K of Power Set** is the sum of masses of the element which are subsets of K.
    - Lets say $\boldsymbol{K = \{a, d, c\}}$
    - $\boldsymbol{Bel(K) = m(a) + m(d) + m(c) + m(a,d) + m(a,c) + m(d,c) + m(a,d,c)}$
  - **Plausibility in K:**
    - It is the ***sum*** of ***masses*** of the ***set*** that ***intersects with K***.
    - i.e.; $\boldsymbol{Pl(K) = m(a) + m(d) + m(c) + m(a,d) + m(d,c) + m(a,c) + m(a,d,c)}$

Methods for Knowledge Representation in Uncertainty

- **Belief Networks**
  - Bayesian networks by allowing for the representation of uncertainty in the strength of the dependencies between variables.
  - Provide a way to handle imprecise and incomplete knowledge.
  - A **belief network** for an intelligent tutoring system might include nodes for student knowledge, engagement, and performance, with edges representing uncertain dependencies between these factors.

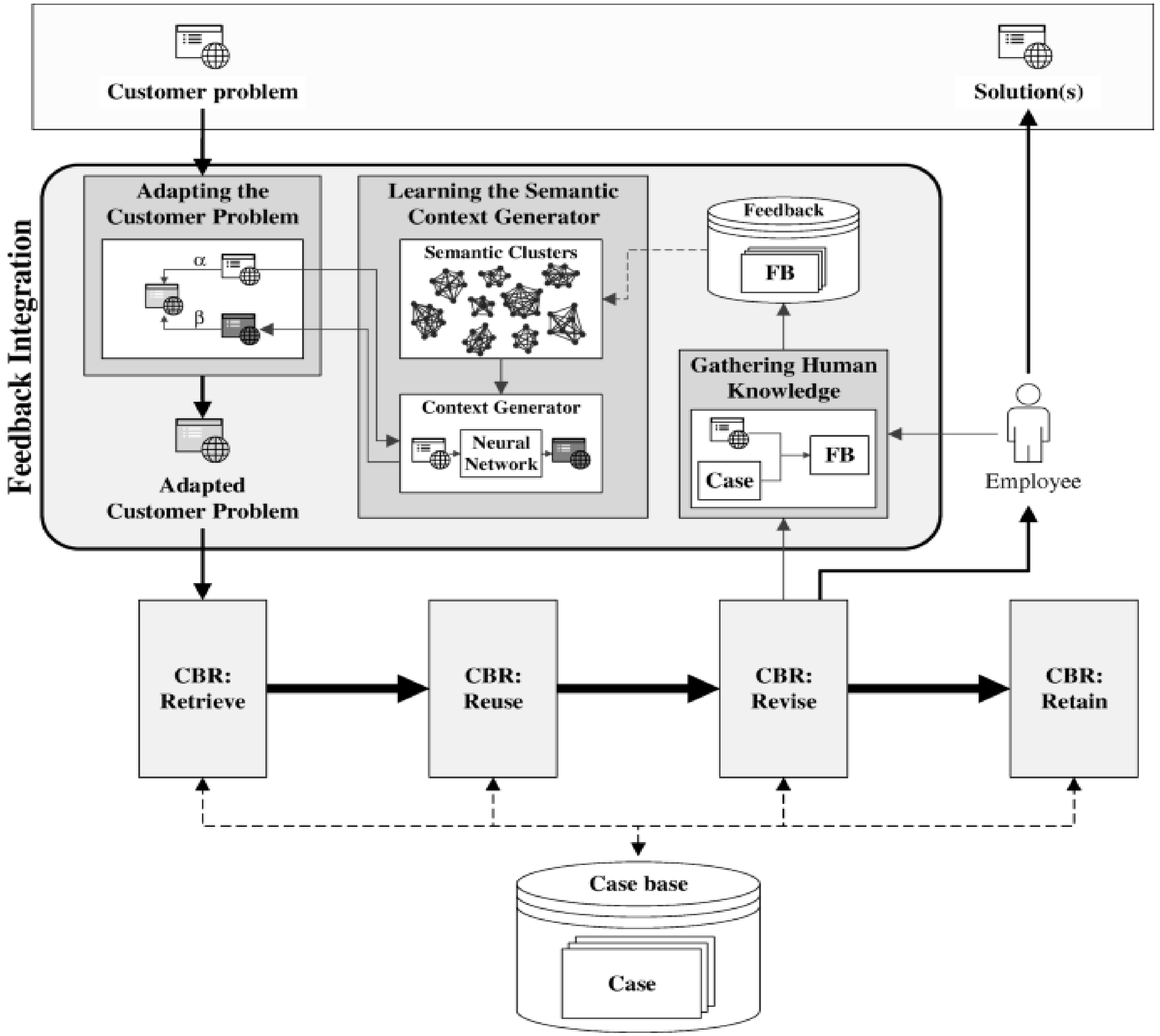

HoloLens
Intelligent Tutoring System
Program to visualize
Dynamic Graphic Visualization
Menu of the visualization



Methods for Knowledge Representation in Uncertainty

- **Case-Based Reasoning**
  - An approach where past cases (experiences) are used to solve new problems.
  - In uncertain domains, CBR can be combined with *probabilistic methods* to *estimate* the *likelihood* of *various outcomes based* on *similar past cases*.

A customer support system can use CBR to suggest solutions based on previous similar customer queries, adjusting recommendations based on the uncertainty of the current context.



Methods for Knowledge Representation in Uncertainty

- **Applications of Uncertain Knowledge Representation**
  - **Medical Diagnosis:**
    - Probabilistic models like Bayesian networks are used to diagnose diseases based on symptoms and medical history.
  - **Autonomous Vehicles:**
    - Fuzzy logic and MDPs help autonomous vehicles navigate and make decisions in dynamic environments.
  - **Natural Language Processing:**
    - HMMs and probabilistic context-free grammars are used for tasks like speech recognition and language modeling.
  - **Robotics:**
    - Robots use probabilistic reasoning to handle sensor noise and uncertain environments for navigation and manipulation tasks.
  - **Finance:**
    - Probabilistic models are employed for risk assessment, fraud detection, and market prediction.

Decision Making

Simple & Complex

Simple Decision Making

- COMBINING BELIEFS AND DESIRES UNDER UNCERTAINTY
  - **Beliefs:**
    - These are our *perceptions* or *judgments* about the *state of the world*.
    - They reflect our *understanding of facts*, *probabilities*, or *truths* about *how things are or will be*.
  - **Desires:**
    - These are **our preferences** or **goals**.
    - They represent **what we want to achieve** or prefer to happen.

Decision theory = Probability theory + Utility theory

Simple Decision Making

▪ COMBINING BELIEFS AND DESIRES UNDER UNCERTAINTY

UTILITY FUNCTION	Agent's preference is captured using the Utility Function $U(s)$, assigning a simple number to express desirability of the state.
EXPECTED UTILITY	Expected utility of an action is given by the EVIDENCE $EU(a\|e)$, is just the average utility values of the outcomes, weighted by the probability that the outcome occurs. $EU(a\|e) = \sum_{s'} P(Result\ (a) = s'\|a, e)U(s')$
MAXIMUM EXPECTED UTILITY	A rational agent must choose the action that maximizes the agent's overall expected utility: $Action = \arg\max_{a} EU(a\|e)$

Simple Decision Making

- UTILITY THEORY
  - Provides a **framework** for **quantifying preferences** and **making decisions** under **uncertainty**.
    - It involves ***assigning numerical values*** (utilities) to ***different outcomes***, which ***reflect*** a ***decision-maker's preferences***.

  - Goal is to **maximize expected utility**,
    - Which is calculated by ***weighing the utility*** of ***each outcome*** by its ***probability***.
    - It assigns a numerical value, referred to as utility, to different outcomes based on their ***satisfaction level***.

Simple Decision Making

- UTILITY THEORY
  - **Utility function** is a **quantitative measure** of the **system's subjective preferences**.
    - It is used to **guide decision-making** in AI systems.
    - An agent or system typically defines the **utility function** based on its **goals**, **objectives**, and **preferences**.
    - **Maps different outcomes** to their **corresponding utility values**, where higher utility values represent more desirable outcomes.
    - Utility function is ***subjective*** and ***can vary from one agent*** or ***system to another***, depending on the specific context or domain of the AI application.

Simple Decision Making

- **UTILITY FUNCTION**
  - Denoted as ***U***.
  - **Mathematical function** that takes as **input** the **different features** of **an outcome** and maps them to a real-valued utility value.

  - Utility function mathematically as $\overline{\boldsymbol{U(x)}}$,
    - where $\bar{x}$ represents the attributes or features of an outcome.

Simple Decision Making

- **Decision Making**
  - **Maximize the expected utility,**
    - which considers the *probability* of *different outcomes* occurring.
    - **Expected utility** is calculated by ***multiplying the utility*** of each ***outcome*** by its ***corresponding probability*** and ***summing*** up the ***results***.

  - Let ***x*** represent an **outcome** or **option**.
  - Let ***U*(*x*)** denote the **utility function**,
    - which maps *x* to its utility value.
  - Let ***p*(*x*)** denote the **probability** of **outcome *x* occurring**.
  - Let ***E*[*U*(*x*)]** denote the **expected utility** of **outcome *x***,
    - which is the sum of the utility values of all possible outcomes weighted by their respective probabilities.

Simple Decision Making

- **Decision Making**
  - $\boldsymbol{E[U(x)]}$ is calculated using the following expression:

$$\boldsymbol{E[U(x)] = \sum_i p(x_i) \times U(x_i)}$$

    - Where, $E[U(x)]$ represents the expected utility of a decision or action,
    - which is the sum of the product of the probability of each outcome $\boldsymbol{x_i}$ (denoted as $\boldsymbol{p(x_i)}$ and
    - its corresponding utility value (denoted as $U(\boldsymbol{x_i})$).
    - The summation is taken over all possible outcomes *i*.

Simple Decision aking

- **Maximum Expected Utility (MEU).**
- The idea of Maximum Expected Utility (MEU). MEU is a decision-making principle that suggests choosing the option that maximizes the expected utility.
  - **Utility Theory Axioms**
    - Utility theory in artificial intelligence is based on a set of axioms or principles that define the properties of a rational utility function.
    - These axioms serve as the foundation for understanding how we can use utility functions to model decision-making under uncertainty.
  - **Orderability**
    - A rational utility function should allow for comparing different outcomes based on their utility values. In other words, if U(x)>U(y), then outcome x is preferred to outcome y.
  - **Monotonicity**
    - If the probability of an outcome increases, its expected utility should also increase. This axiom ensures that an increase in the likelihood of an outcome increases its perceived value or utility.

Simple Decision Making

- Transitivity
  - If outcome x is preferred to outcome y, and outcome y is preferred to outcome z, then outcome x should be preferred to outcome z.
  - This axiom ensures that the preferences modeled by the utility function are consistent and do not lead to contradictions.

$$\begin{tikzcd}
x \arrow[r] \arrow[rr, bend left, leftrightarrow] & y \arrow[r] & z
\end{tikzcd}$$

- Continuity
  - Small changes in the probabilities of outcomes should result in small changes in the expected utility.
  - This axiom ensures that the utility function is smooth and well-behaved and that small changes in probabilities do not result in abrupt changes in decision-making.

Simple Decision Making

- Substitutability
  - If two outcomes, x and y, are equally preferred, then we should equally prefer any combination of x and y. This axiom allows for substituting equally preferred outcomes without affecting the decision-making process.

- Decomposability
  - The utility function should be able to represent preferences over multiple attributes or features of an outcome in a decomposable manner.

Simple Decision Making

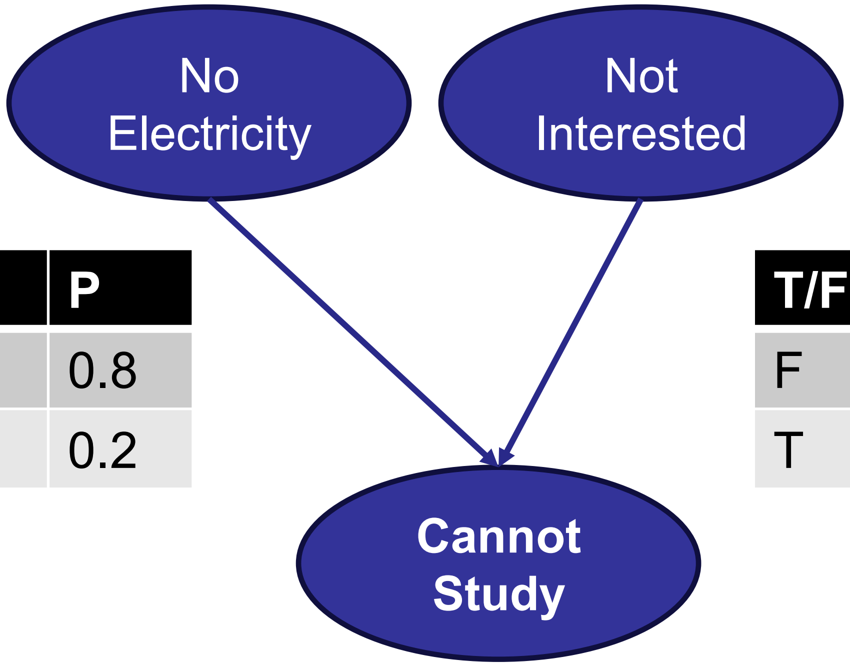
- Example:
  - Suppose a company is considering launching a new product. The probability of success is estimated to be 0.7, and the probability of failure is 0.3. The expected revenue from success is $100,000, and the expected loss from failure is $50,000. Using expected utility theory, the expected utility of launching the product can be calculated as:

  - Expected Utility = (0.7 x $100,000) + (0.3 x -$50,000) = $70,000 - $15,000 = $55,000

  - If the **expected utility is positive**, the company should launch the product.
  - However, if the expected utility is negative, the company should not launch the product.

Simple Decision Making

- Probabilistic Reasoning
  - Combines probability theory with logic to handle uncertainty.
  - Allows decision-makers to *model uncertain situations quantitatively*, enabling them to make informed decisions based on the likelihood of different outcomes.
  - This reasoning is particularly useful in artificial intelligence applications, where systems must operate in environments with incomplete or noisy data.



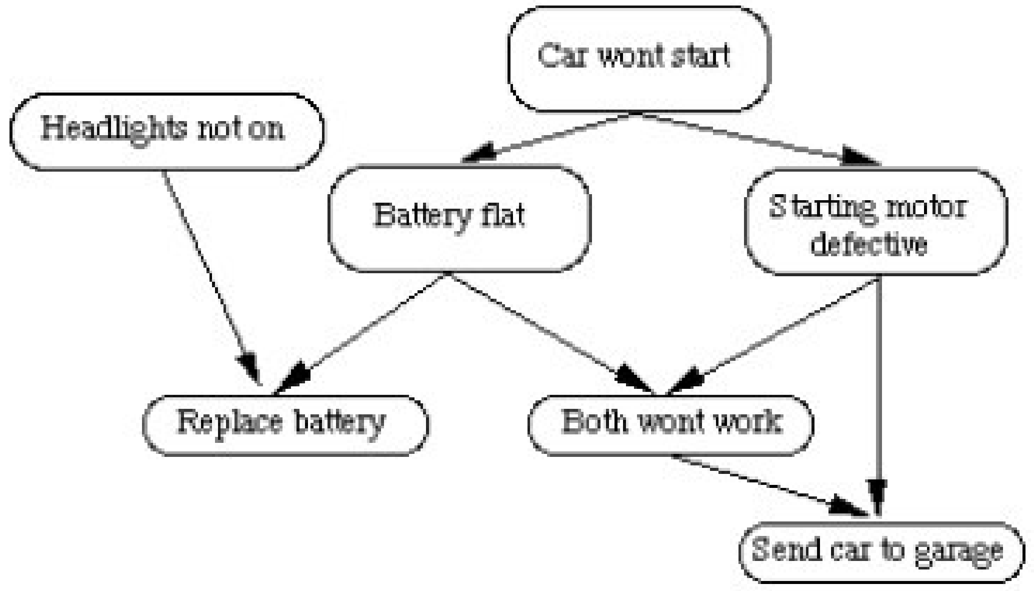
T/F	P
F	0.8
T	0.2

T/F	P
F	0.7
T	0.3

A Sample example on Probabilistic reasoning

Simple Decision Making

- Decision Tree
  - Decision trees are another tool used for *making decisions under uncertainty*.
  - They **visually map** out different decision paths and their possible outcomes, allowing for a structured analysis of choices.
  - By evaluating the expected utility of each path, decision-makers can choose the most advantageous option



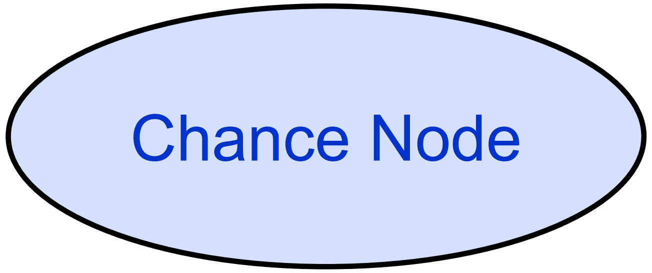
A Sample example on Decision Tree

Simple Decision Making

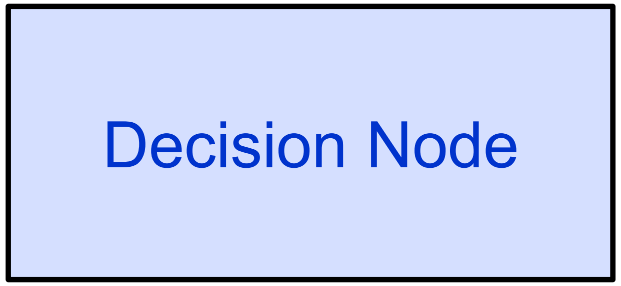
Decision Networks

- Decision networks, or influence diagrams, are crucial in artificial intelligence for ***structured decision-making under uncertainty***, ***integrating decision theory*** and ***probability to evaluate actions*** and ***potential outcomes***.
- Decision networks are graphical models that **extend Bayesian networks** by incorporating **decision and utility nodes**, enabling a comprehensive analysis of decision scenarios.

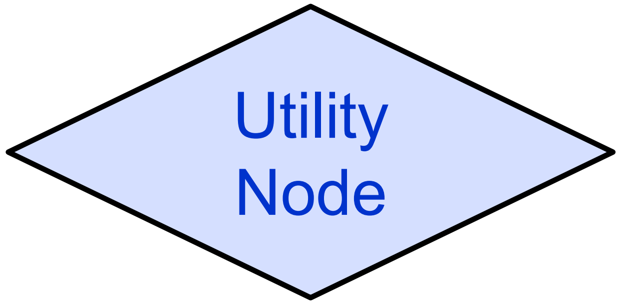
Components of Decision Networks



Represent ***random variables*** and their ***possible values***, capturing the uncertainty in the decision-making process.



Represent the ***choices available to the decision-maker***.



Represent the ***utility*** or ***value*** of the ***outcomes***, helping to evaluate and compare different decision paths.

Simple Decision Making

- Decision Networks : Example
  - *Consider a simple medical diagnosis scenario where a doctor needs to decide whether to order a test for a patient based on the likelihood of a disease and the cost of the test. The decision network for this scenario might include:*

  - **Chance Nodes:** Disease presence (Yes/No), Test result (Positive/Negative)
  - **Decision Node:** Order test (Yes/No)
  - **Utility Node:** Overall patient health outcome and cost

  - The doctor can use the decision network to evaluate the expected utility of ordering the test versus not ordering it, taking into account the probabilities of disease presence and test results, and the utility values associated with different outcomes.

Simple Decision Making

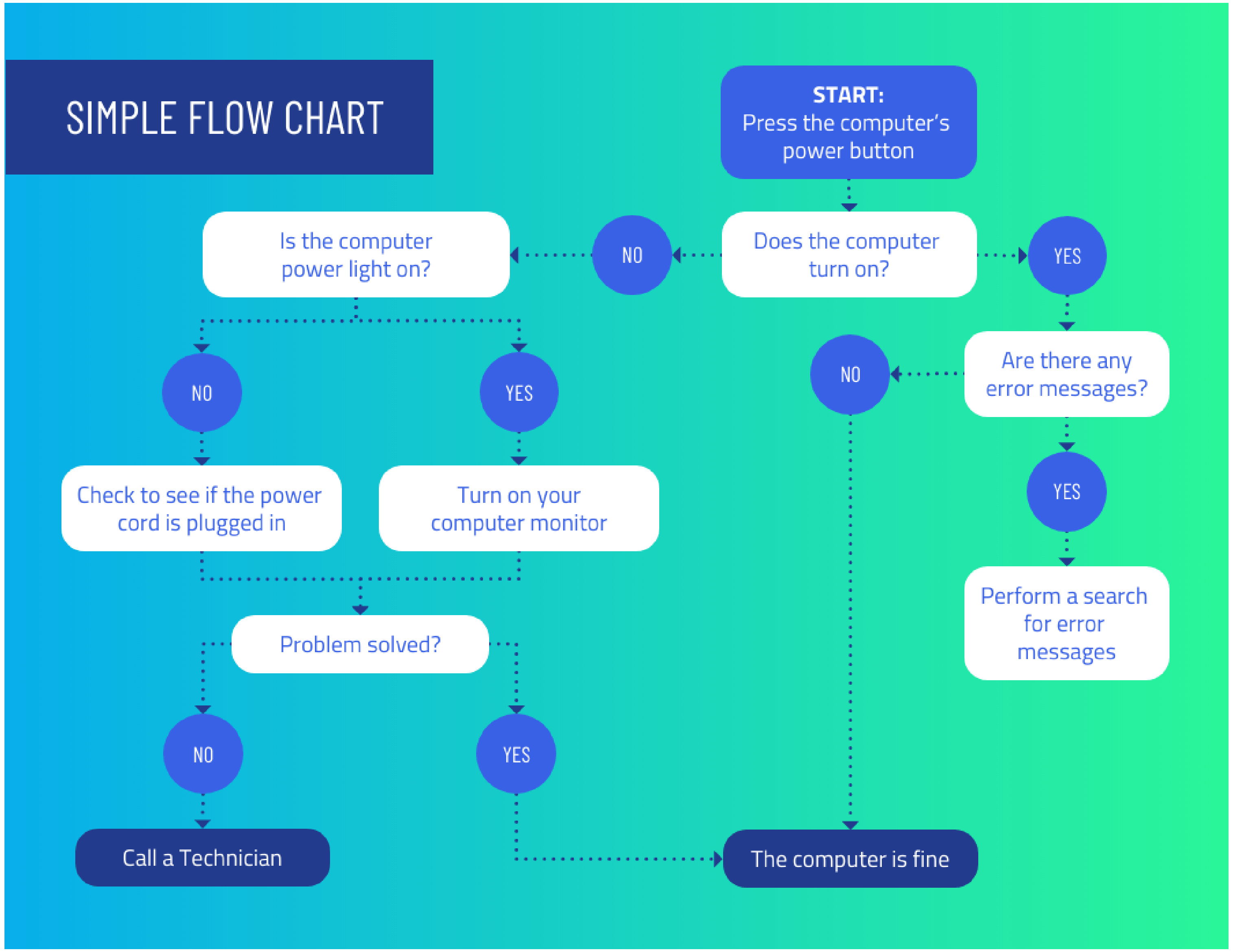
- Decision Networks : Structure
  - *To structure a decision network, follow these key steps:*
  - *Define Variables and Functions*: Identify random variables, decision variables, and utility functions crucial for the decision problem.
  - *Node Representation*: Represent random variables as chance nodes, decision variables as decision nodes, and utility functions as utility nodes.
  - *Connect Nodes*: Use directed arcs to represent dependencies between variables.

Simple Decision Making

- Decision Networks : Structure
  - Directed Arcs:
    - Arcs to decision nodes represent available information.
    - Arcs to chance nodes represent probabilistic dependencies.
    - Arcs to utility nodes represent utility dependencies.
  - Ensure DAG Structure:
    - Avoid cycles or feedback loops in the arcs to maintain a directed acyclic graph.
  - Define Domains:
    - Specify the domain for each random variable and decision variable. Utility nodes do not have domains.
  - Conditional Probability Distributions:
    - Provide conditional probability distributions for each random variable given their parents in the network.
  - Utility Function:
    - Define the utility function mapping the values of the variables it relies on to a real number representing the decision-maker's preferences.

Simple Decision Making

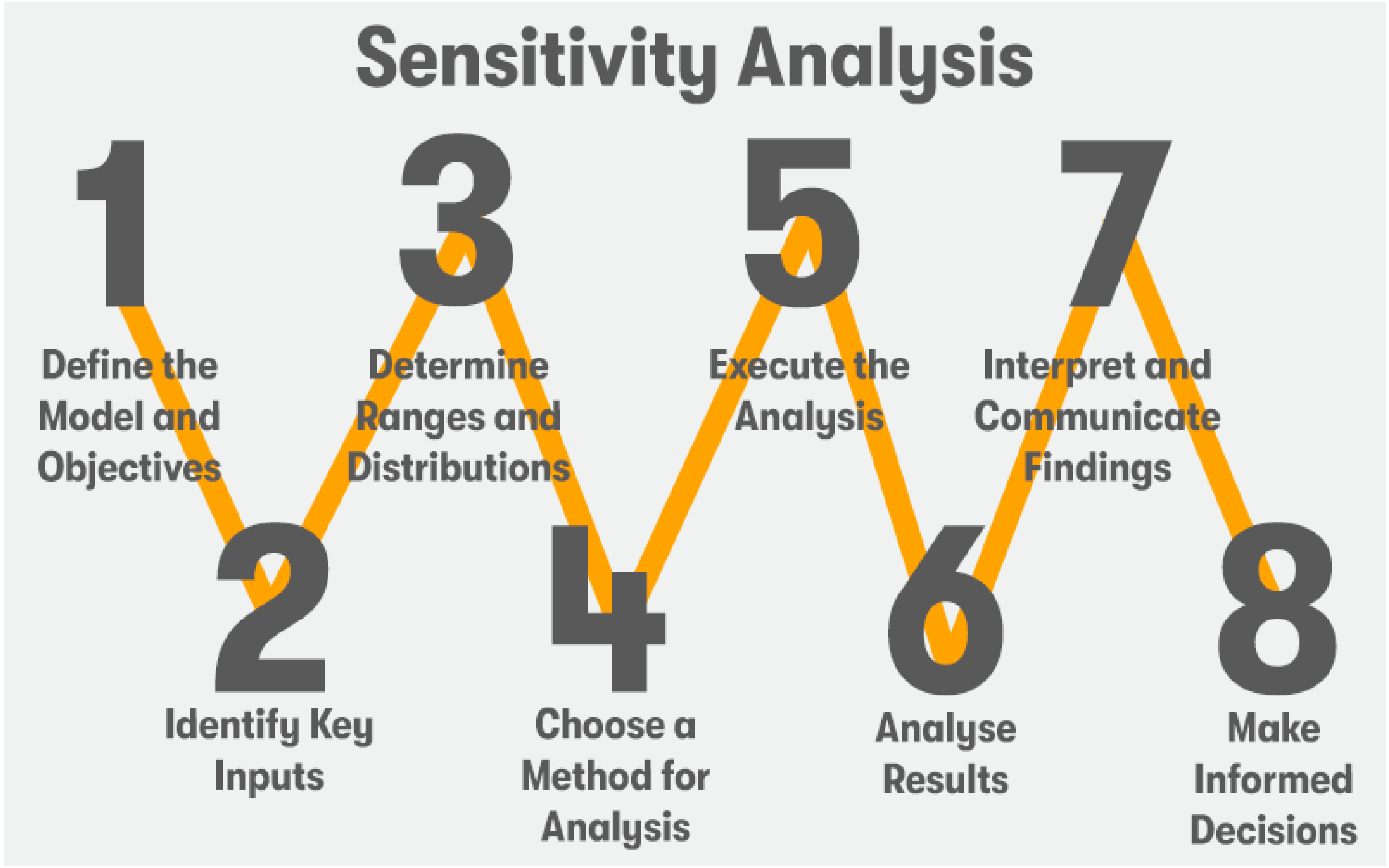
Example of Representing a Decision Problem


SIMPLE FLOW CHART
START: Press the computer's power button
Does the computer turn on?
NO
YES
Is the computer power light on?
Are there any error messages?
NO
YES
NO
YES
Check to see if the power cord is plugged in
Turn on your computer monitor
Perform a search for error messages
Problem solved?
NO
YES
Call a Technician
The computer is fine

Simple Decision Making

- Sensitivity Analysis
  - Involves analyzing *how changes in probability or utility affect the decision*.
  - It helps to identify the **most critical factors influencing** the **decision**.
  - **Sensitivity analysis**: A technique used to determine *how different values* of an *independent variable* affect a *particular dependent variable* under a *given set* of *assumptions*.
    - Useful in *assessing* the *impact of changes* in *input variables* on the *outcome* of a *model* or *system*.
    - By ***systematically varying*** the ***inputs*** within ***specific ranges***, *analysts can identify which inputs have the most influence on the outcome* and how the ***outcome changes*** in ***response*** to ***variations*** in ***input values***.

Simple Decision Making


Sensitivity Analysis
1
Define the Model and Objectives
2
Identify Key Inputs
3
Determine Ranges and Distributions
4
Choose a Method for Analysis
5
Execute the Analysis
6
Analyse Results
7
Interpret and Communicate Findings
8
Make Informed Decisions

Simple Decision Making

- Sensitivity Analysis : Example
  - Optimizing Incident Resolution Time:
    - An IT service provider is focused on **reducing the average time** to **resolve incidents** to **enhance customer satisfaction** and **operational efficiency**. The leadership team decides to conduct a **sensitivity analysis** to **understand** which **factors most significantly impact resolution time** and **where to focus improvement efforts**.
  - Define the Model and Objectives:
    - The objective is to **minimize** the **average incident resolution time**.
    - The model will **consider various inputs** such as the number of IT staff, their skill levels, the use of automation tools, and the complexity of incidents.

Simple Decision Making

- Sensitivity Analysis : Example
  - Identify Key Inputs:
    - Key inputs identified for the sensitivity analysis include:
      - Number of IT support staff
      - Skill level of IT support staff
      - Adoption level of automation tools (e.g., ticketing systems, AI-driven diagnostics)
      - Average complexity of incidents
  - Determine Ranges and Distributions:
    - For each key input, the team defines plausible ranges based on historical data and expert judgment.
    - For instance, the number of staff could vary from current levels up to a 20% increase, considering budget constraints.

Simple Decision Making

- Sensitivity Analysis
  - Choose a Method for Analysis:
    - The team opts for a combination of ***One-at-a-Time (OAT)*** and ***Monte Carlo simulation*** methods.
    - This approach allows them to *isolate the impact of individual variables* and assess the *overall effect of varying all inputs within their distributions*.
  - Execute the Analysis:
    - Using simulation software, the team systematically *varies each input* within its *defined range* while *holding others constant* and then conducts a *Monte Carlo simulation* to *understand* the *probabilistic impact* of all inputs changing simultaneously.

Simple Decision Making

- Sensitivity Analysis
  - Analyse Results:
    - The analysis reveals that *while increasing the number of IT support staff and enhancing their skill levels both significantly reduce resolution times*, the adoption of automation tools has the most substantial impact, especially for high-complexity incidents.
  - Interpret and Communicate Findings:
    - The findings suggest *prioritizing investments* in *automation tools* and *targeted training to upskill the IT staff*. These insights are *presented* to *stakeholders*, *highlighting* the *potential reduction* in *incident resolution time* and *improvements* in *service quality*.
  - Make Informed Decisions:
    - Guided by the sensitivity analysis, the IT service provider decides to allocate the budget towards implementing *advanced automation tools* and *developing* a *comprehensive staff training program* focusing on handling complex incidents efficiently.

Simple Decision making

- Bayesian Decision Theory
  - This framework updates probabilities based on new information, allowing for more accurate decision making.

$$\underset{\text{posterior}}{\boxed{P(A|B)}} = \underset{\text{prior}}{\boxed{P(A)}} \times \frac{\boxed{P(B|A)}\ \text{likelihood}}{\boxed{P(B)}\ \text{marginal}}$$

- *Prior Probability:*
  - This represents our initial beliefs or prior knowledge about a hypothesis, model, or parameter before observing any data. It encapsulates our subjective beliefs about how likely it is to be true.
- *Likelihood:*
  - The likelihood function quantifies how well the observed data aligns with each hypothesis, model, or parameter under consideration. It measures the probability of observing the data given a particular hypothesis.
- *Posterior Probability:*
  - The posterior probability is the updated belief about the hypothesis, model, or parameter after incorporating the observed data. It combines the prior probability and the likelihood using Bayes' theorem.
- *Evidence (or Marginal Likelihood):*
  - The evidence or marginal likelihood is the probability of observing the data under all possible hypotheses, models, or parameter values. It acts as a normalization constant in Bayes' theorem to ensure that the posterior probabilities sum up to 1.

Simple Decision Making

- Bayesian Decision Theory
- Bayesian inference follows a systematic process:
- Specify a Prior:
  - We start by specifying a prior probability distribution that reflects our initial beliefs or knowledge about the problem at hand.
  - This prior distribution can be subjective and may be based on existing data or domain expertise.
- Collect Data:
  - We then collect and observe data related to the problem.
  - This data can be in the form of measurements, observations, or any relevant information.
- Calculate the Likelihood:
  - The likelihood function is used to assess how well the observed data aligns with each possible hypothesis or model.
  - It quantifies the probability of observing the data given each hypothesis.

Simple Decision Making

- Bayesian Decision Theory
- Apply Bayes’ Theorem:
  - Bayes’ theorem is used to update the prior probabilities into posterior probabilities, taking into account the data and the likelihood.
  - Framework updates probabilities based on new information, allowing for more accurate decision making.
- Make Informed Decisions:
  - The posterior probabilities provide us with updated beliefs about the hypotheses, models, or parameters.
  - Posterior probabilities to make decisions, make predictions, or perform further analysis is used.

Simple Decision Making

- Nonmyopic Information Gathering
  - Refers to *planning strategies* that *consider* the *long-term impact of actions* on *reducing uncertainty*, as opposed to only optimizing for immediate information gain.
  - This is in **contrast to myopic approaches** that **greedily select actions based** on **short-term rewards**.

- Key aspects of Nonmyopic Information Gathering
  - *Considers future uncertainty reduction:*
    - Nonmyopic methods plan actions that minimize accumulated uncertainty over an extended horizon, not just the immediate step.
  - *Computationally challenging:*
    - Optimizing for long-term information gain is a complex stochastic optimization problem.
    - Existing nonmyopic approaches sacrifice computational efficiency to achieve optimality

Simple Decision Making

- Key aspects of Nonmyopic Information Gathering
  - Sampling-based planning:
    - Simultaneously explore the robot motion space and information space to enable scalable nonmyopic planning.
    - These methods are probabilistically complete and asymptotically optimal.
  - Biased sampling:
    - Introducing bias in the sampling process towards informative regions allows quickly computing sensor policies that achieve desired uncertainty reduction, even in large-scale estimation tasks.
  - Extensions:
    - The nonmyopic sampling-based planning framework can be extended to handle hidden states with no prior information and relaxed linearity assumptions.

Complex Decision Making

- Sequential Decision Problems
  - Sequential decision problems are crucial in *uncertain decision-making*, involving *interrelated*, *interdependent decisions* over time.
  - They **optimize objectives** like **cumulative rewards** and costs, considering probabilistic state transitions.

  - Key Concepts in Sequential Decision Problems
    - Sequential decision-making refers to a *structured approach* where *decisions* are *made in a sequence*, with each step informed by the outcomes of previous actions.
    - This process often employs tools such as *Markov Decision Processes (MDPs)* and *dynamic programming* to model the decision-making environment and optimize the decision-making policy based on expected utilities

Complex Decision Making

- Sequential Decision Problems
  - Components of Sequential Decision Problems
    - States: The various conditions or situations that can be encountered during the decision-making process.
    - Actions: The choices available to the decision-maker at each state.
    - Transition Probabilities: The likelihood of moving from one state to another given a specific action.
    - Rewards: The utility or value received after transitioning to a new state as a result of an action.
    - Policies: A strategy that specifies the best action to take in each state to maximize expected utility

Complex Decision Making

- Methods for Solving Sequential Decision Problems
  - *Value Iteration:* This algorithm computes the utility of each state iteratively, using the Bellman equation to derive optimal actions based on expected future rewards.
  - *Policy Iteration:* This method alternates between evaluating a policy's utility and improving it until an optimal policy is found.
  - *Dynamic Programming:* A technique used for solving complex problems by breaking them down into simpler subproblems, which can be solved independently and combined for an overall solution.

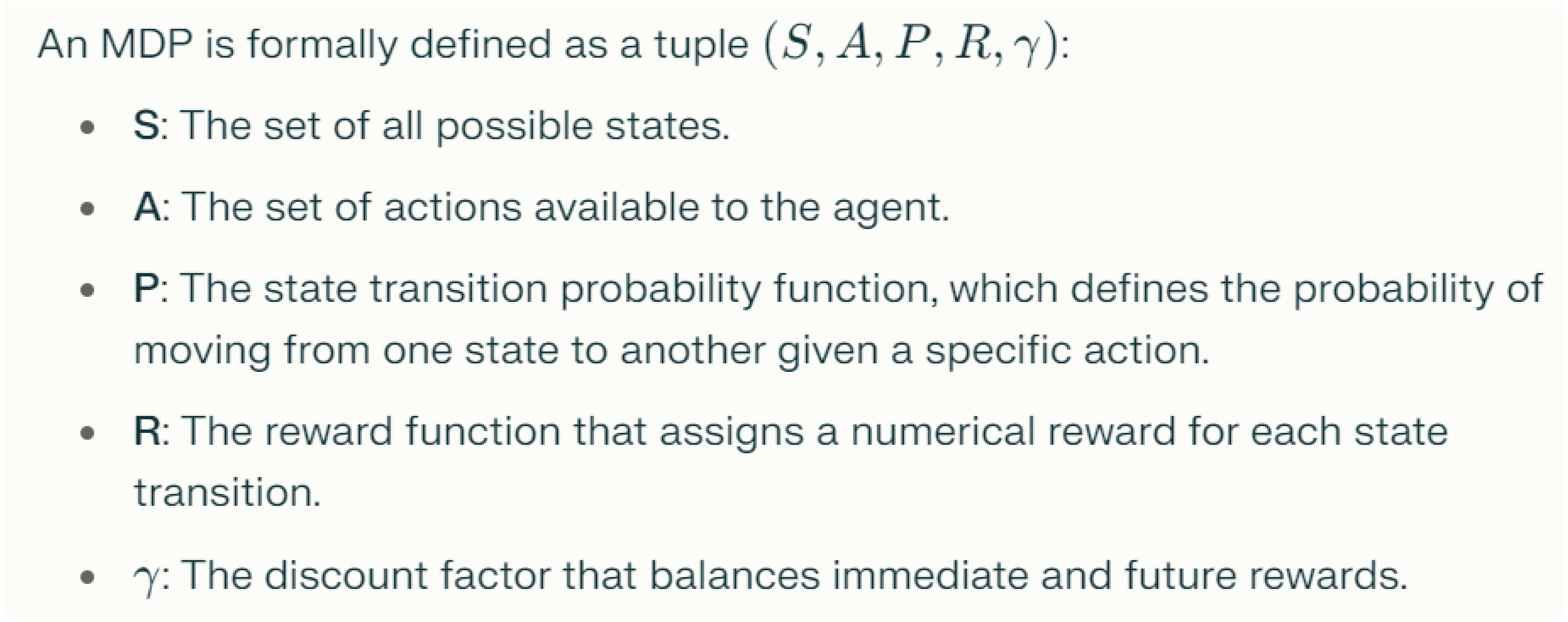
Complex Decision Making

- **Advantages**
  - *Structured Approach*: Provides clarity and a systematic method for tackling complex decisions.
  - *Risk Management*: Helps quantify risks associated with different choices, facilitating informed decision-making.
  - *Long-Term Planning*: Encourages consideration of future implications of current decisions, aiding strategic foresight

- **Disadvantages**
  - *Complexity*: The process can become intricate, especially with numerous states and actions involved.
  - *Time Consumption*: Sequential decision-making can be time-intensive due to its iterative nature.
  - *Probability Estimation Difficulties*: Accurately assigning probabilities to outcomes can be challenging and may rely on subjective judgment

Complex Decision Making

- Markov Decision Processes (MDPs)
  - Are a fundamental framework in AI for modeling decision-making in situations where outcomes are uncertain and depend on both random factors and the actions taken by an agent.
  - MDPs provide a structured approach to solving problems that involve sequential decision-making, where each choice affects future states and rewards.

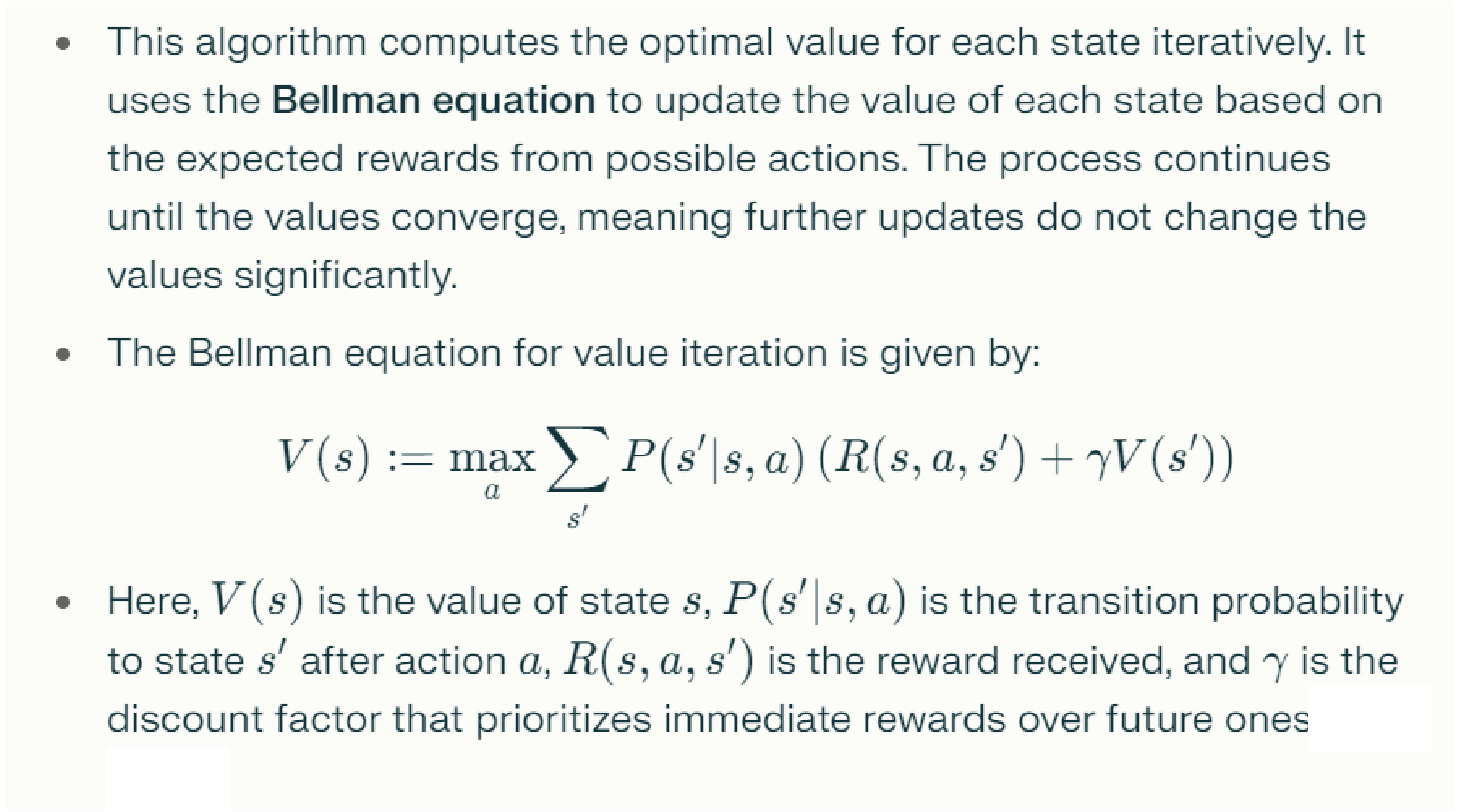


An MDP is formally defined as a tuple (S, A, P, R, γ):

- **S**: The set of all possible states.
- **A**: The set of actions available to the agent.
- **P**: The state transition probability function, which defines the probability of moving from one state to another given a specific action.
- **R**: The reward function that assigns a numerical reward for each state transition.
- γ: The discount factor that balances immediate and future rewards.

Complex Decision Making

- Algos. for MDPs: VALUE ITERATION



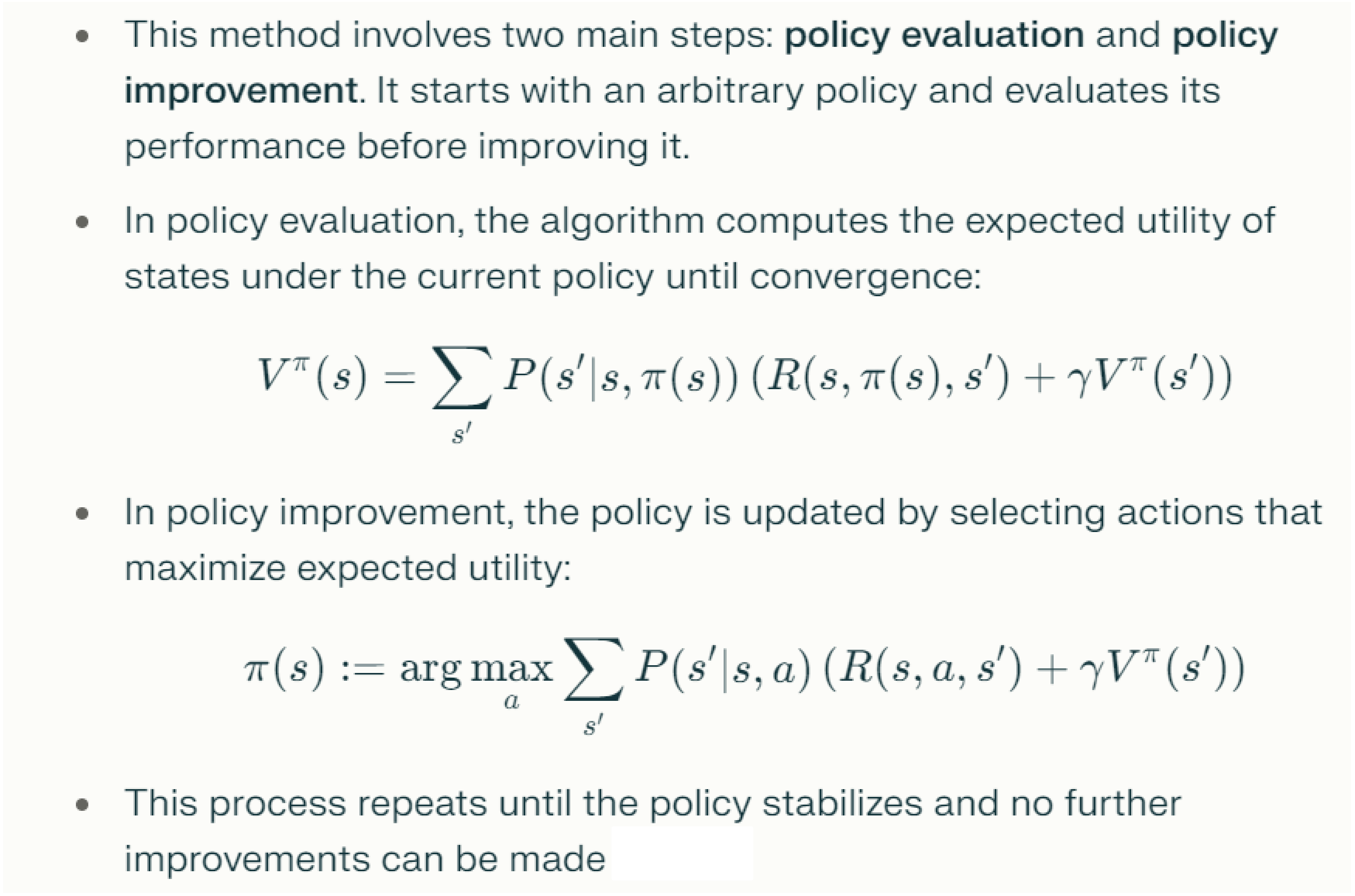
- This algorithm computes the optimal value for each state iteratively. It uses the **Bellman equation** to update the value of each state based on the expected rewards from possible actions. The process continues until the values converge, meaning further updates do not change the values significantly.
- The Bellman equation for value iteration is given by:

$$V(s) := \max_a \sum_{s'} P(s'|s,a)\left(R(s,a,s') + \gamma V(s')\right)$$

- Here, $V(s)$ is the value of state s, $P(s'|s,a)$ is the transition probability to state s' after action a, $R(s,a,s')$ is the reward received, and γ is the discount factor that prioritizes immediate rewards over future ones

Complex Decision Making

- Algos. for MDPs:
- POLICY ITERATION



- This method involves two main steps: **policy evaluation** and **policy improvement**. It starts with an arbitrary policy and evaluates its performance before improving it.
- In policy evaluation, the algorithm computes the expected utility of states under the current policy until convergence:

$$V^{\pi}(s) = \sum_{s'} P(s'|s, \pi(s)) \left(R(s, \pi(s), s') + \gamma V^{\pi}(s')\right)$$

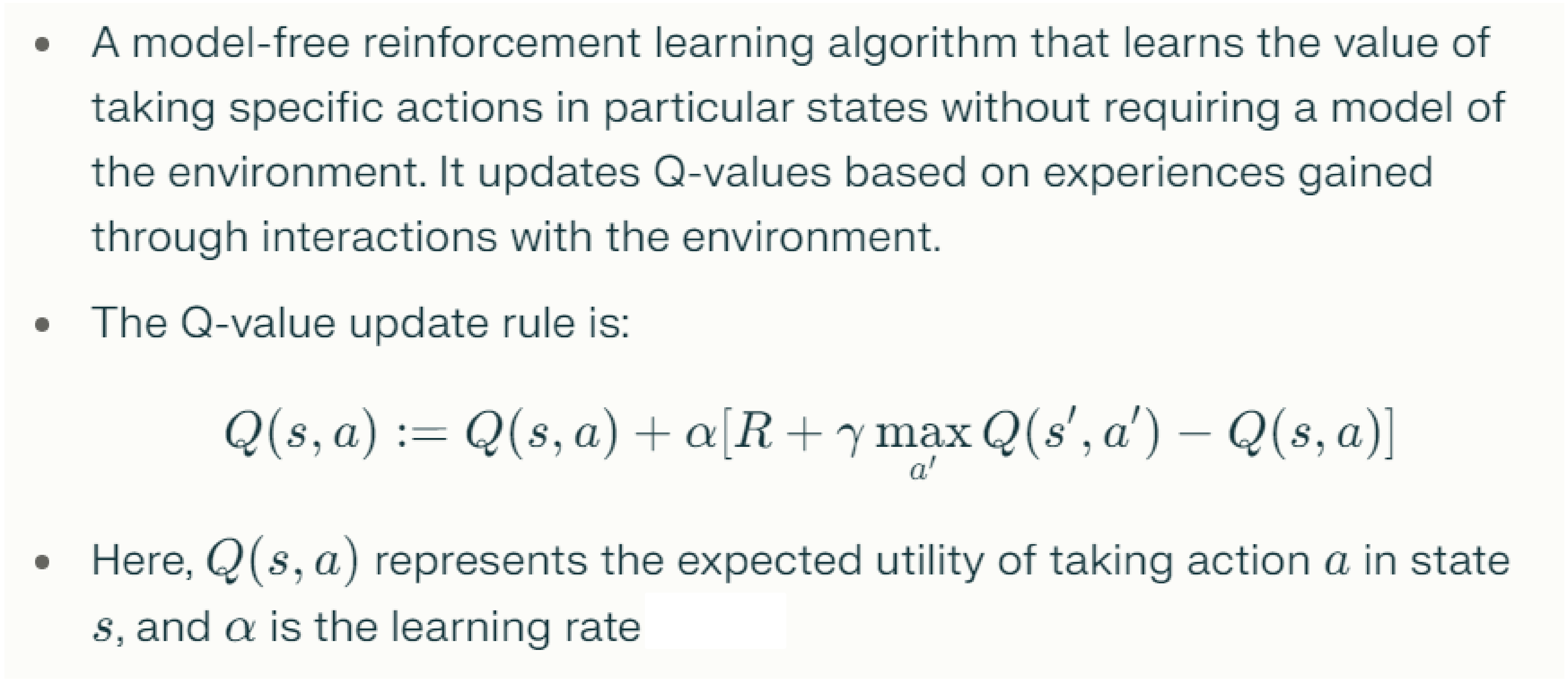
- In policy improvement, the policy is updated by selecting actions that maximize expected utility:

$$\pi(s) := \arg\max_{a} \sum_{s'} P(s'|s, a) \left(R(s, a, s') + \gamma V^{\pi}(s')\right)$$

- This process repeats until the policy stabilizes and no further improvements can be made

Complex Decision Making

- Algos. for MDPs: Q Learning



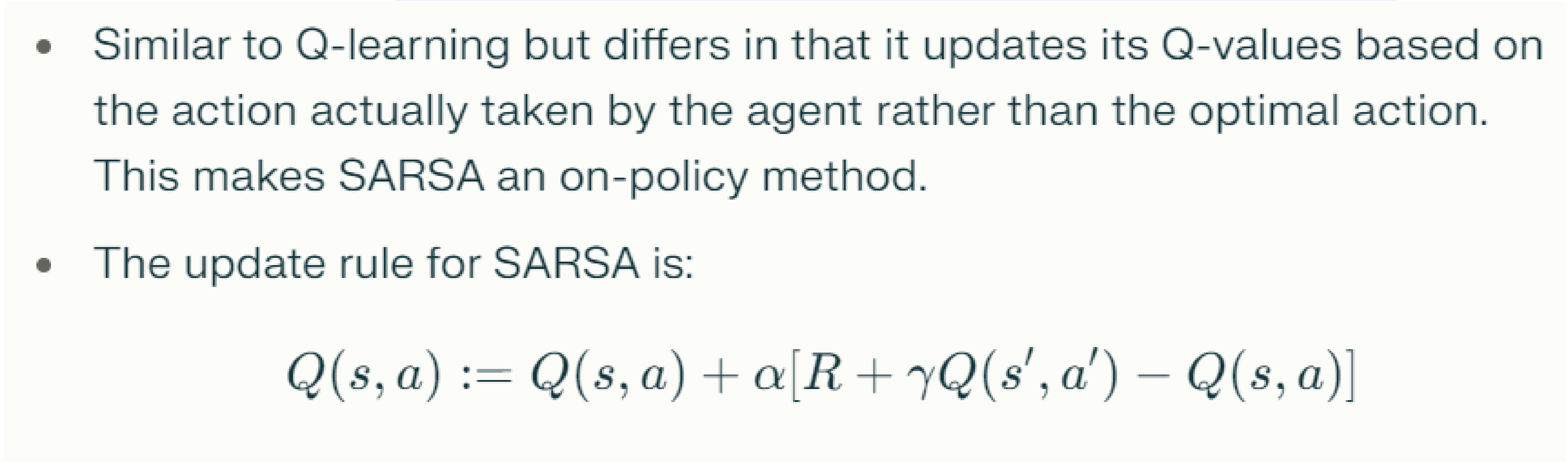
- A model-free reinforcement learning algorithm that learns the value of taking specific actions in particular states without requiring a model of the environment. It updates Q-values based on experiences gained through interactions with the environment.
- The Q-value update rule is:

$$Q(s,a) := Q(s,a) + \alpha[R + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

- Here, $Q(s,a)$ represents the expected utility of taking action a in state s, and α is the learning rate

Complex Decision Making

- Algos. for MDPs: SARSA (State-Action-Reward-State-Action



- Similar to Q-learning but differs in that it updates its Q-values based on the action actually taken by the agent rather than the optimal action. This makes SARSA an on-policy method.
- The update rule for SARSA is:

$$Q(s,a) := Q(s,a) + \alpha[R + \gamma Q(s',a') - Q(s,a)]$$

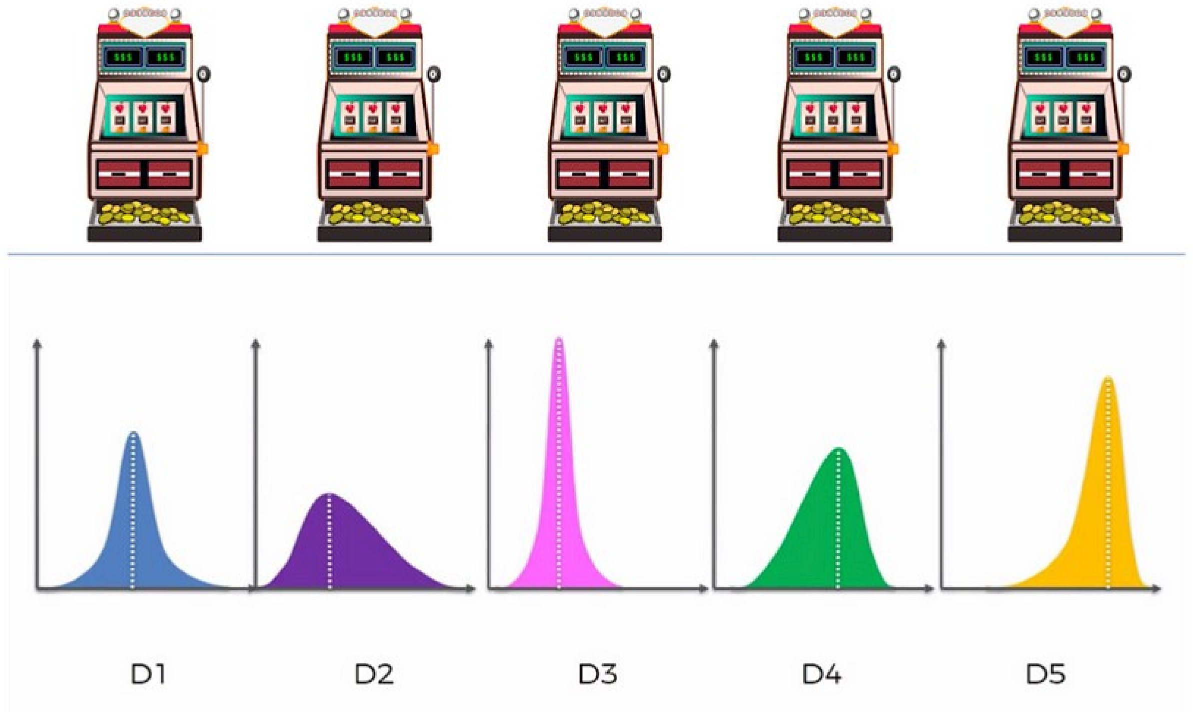
A Markov decision process (MDP) is a discrete-time stochastic control process.

Key Components

- At each time step, the process is in a state and the decision maker selects an action.
- The process then transitions to a new state based on the chosen action, influenced by a state transition function, and provides a corresponding reward.

Complex Decision Making

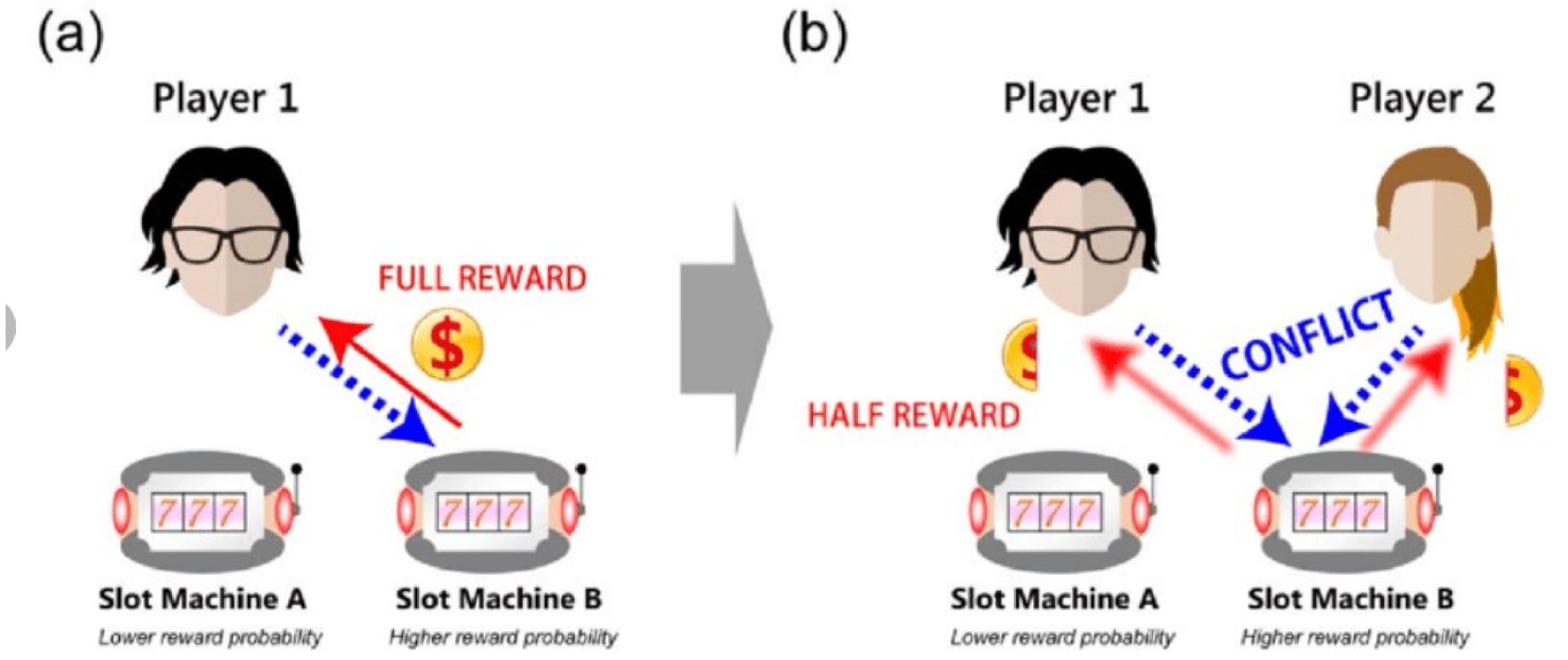
- Bandit Problems
  - A decision-making problem where a player chooses between *multiple options* with *uncertain outcomes* to *maximize cumulative rewards*.
  - **Exploration-Exploitation Tradeoff:** Involves balancing between exploiting the known best option and exploring other options to gather more information
  - **Reward Maximization:** Objective is to maximize total rewards by selecting actions to achieve the best possible outcomes



- **Scenario:** Pull machine k → sample from ***unknown*** reward distribution D_k → observe reward.
- **Problem:** Given a finite number of pulls T, how can I optimize my winnings?
- How much should I **explore**? How much should I **exploit**?

Complex Decision Making

- Multi-Armed Bandit Problems in Complex Decision Making
  - Multi-armed bandit (MAB) problems are a framework for *sequential decision making* under *uncertainty* that is particularly relevant in complex decision scenarios.
  - They **model the tradeoff between exploration** (gathering more information) and **exploitation** (making decisions based on current knowledge) in an iterative manner.



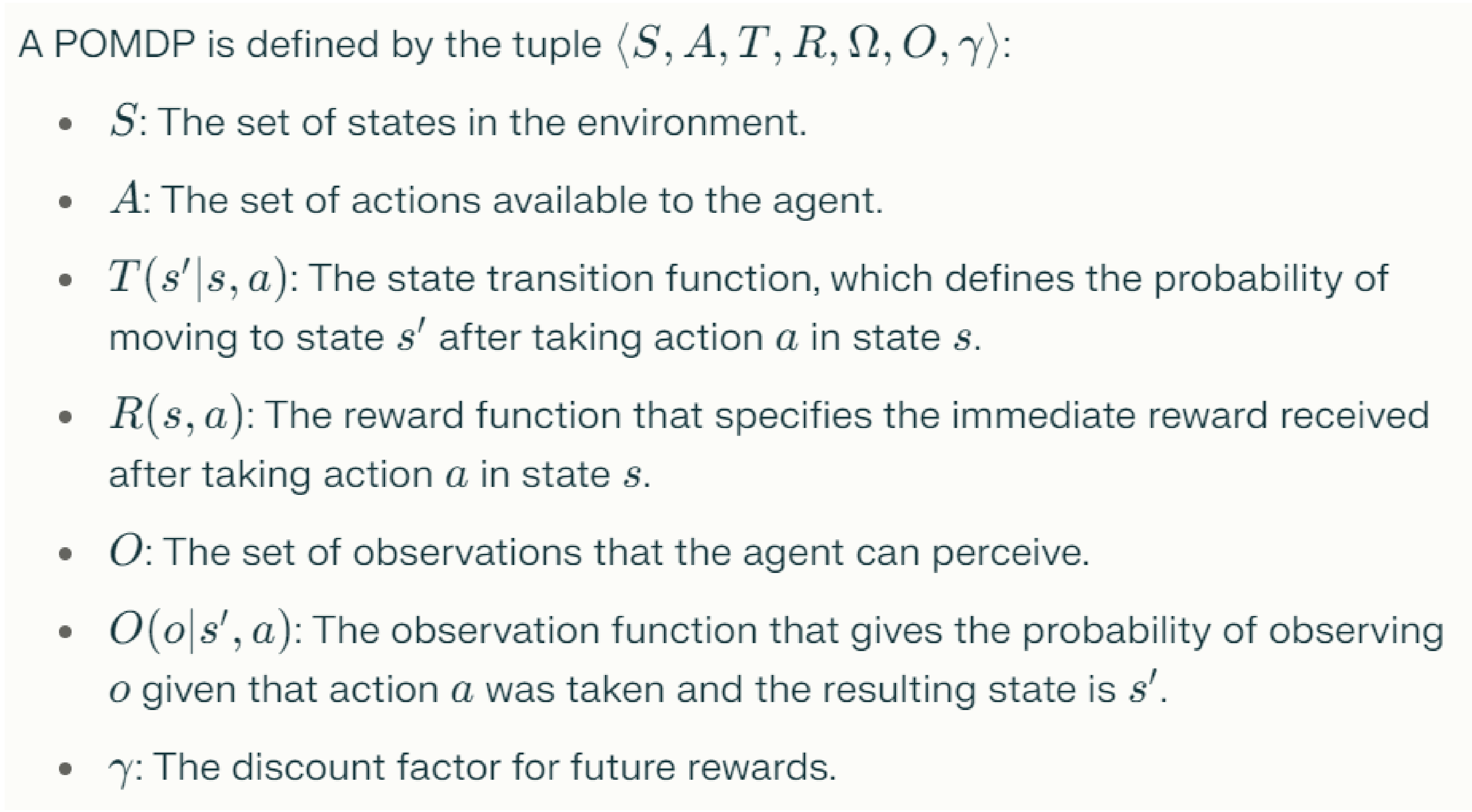
CMAB problem. (a) Two-armed bandit problem where a single player tries to maximize

Complex Decision Making

- Key Aspects of MAB Problems
  - A decision-making problem where a player chooses between multiple options with uncertain outcomes to maximize cumulative rewards.
  - Exploration-Exploitation Tradeoff: Involves balancing between exploiting the known best option and exploring other options to gather more information
  - Reward Maximization: Objective is to maximize total rewards by selecting actions to achieve the best possible outcomes
- Algorithms for MAB
  - Epsilon-greedy: Selects the arm with the highest estimated reward most of the time, but with probability ε explores a random arm
  - Upper Confidence Bound (UCB): Selects the arm that maximizes the sum of the estimated reward and a confidence interval, encouraging exploration of uncertain arms
  - Thompson Sampling: Maintains a posterior distribution over the arm rewards and samples from it to select the arm, providing a balance of exploration and exploitation
  - Linear UCB: Extends UCB to settings where the rewards are linear functions of context features, allowing the algorithm to exploit structure in the problem

Complex Decision Making

- Partially Observable Markov Decision Processes (POMDPs)
  - An extension of Markov Decision Processes (MDPs) that address decision-making in environments where the agent does not have complete knowledge of the underlying state.
  - This framework is particularly relevant in *complex decision-making scenarios characterized by uncertainty and incomplete information*.

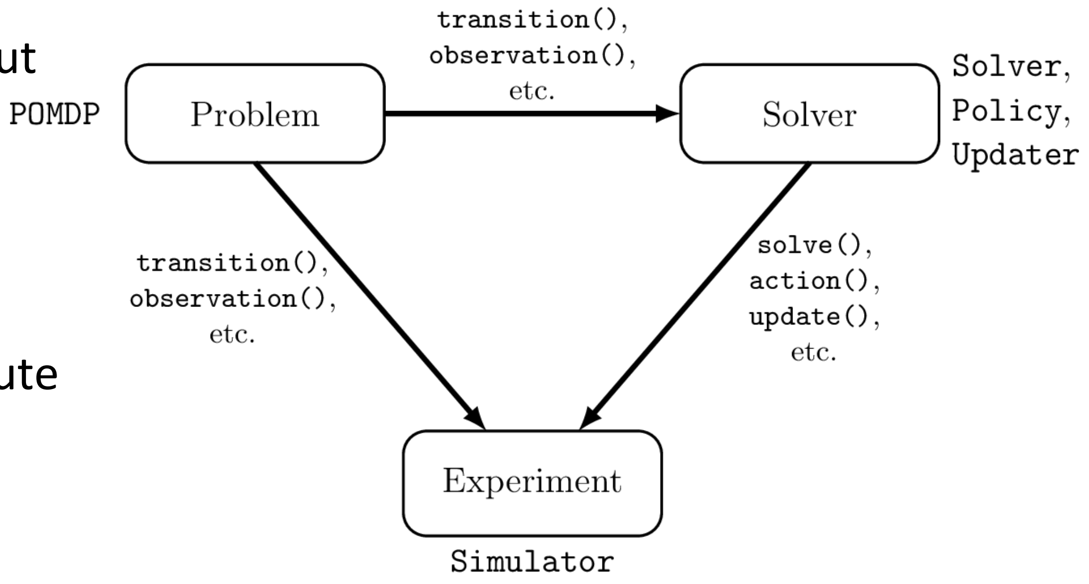


A POMDP is defined by the tuple $\langle S, A, T, R, \Omega, O, \gamma \rangle$:

- S: The set of states in the environment.
- A: The set of actions available to the agent.
- $T(s'|s,a)$: The state transition function, which defines the probability of moving to state s' after taking action a in state s.
- $R(s,a)$: The reward function that specifies the immediate reward received after taking action a in state s.
- O: The set of observations that the agent can perceive.
- $O(o|s',a)$: The observation function that gives the probability of observing o given that action a was taken and the resulting state is s'.
- γ: The discount factor for future rewards.

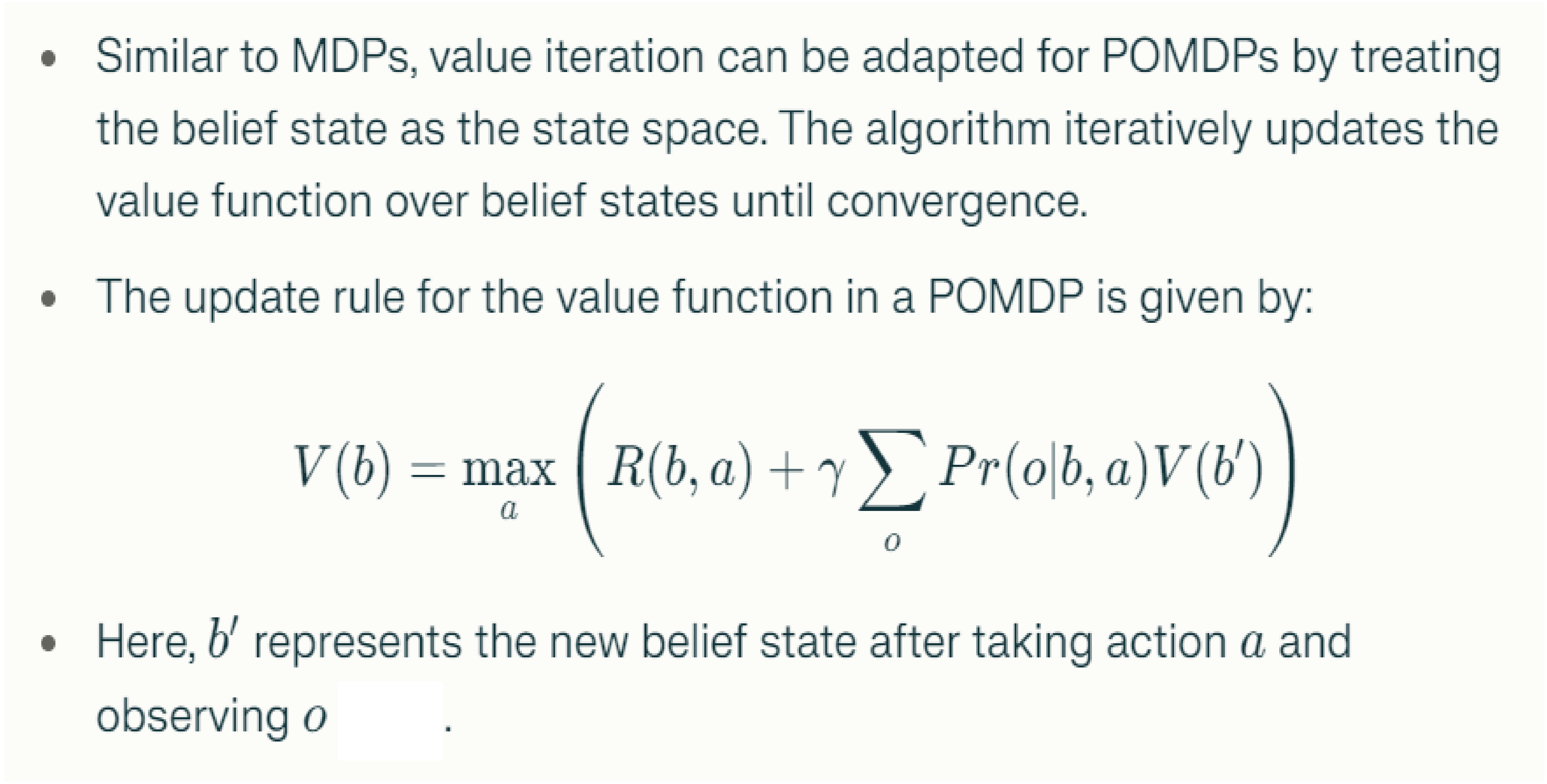
Complex Decision Making

- Solving POMDPs
  - Exact Algorithms: These algorithms compute optimal policies but can be computationally intensive. They often rely on dynamic programming techniques similar to those used for MDPs.
  - Approximate Solutions: Due to computational complexity, approximate methods are frequently employed: Point-Based Value Iteration: Focuses on a finite set of belief points to compute value functions efficiently.
  - Heuristic Search Value Iteration (HSVI): Uses heuristics to guide searches through the belief space, focusing on relevant beliefs and actions.
  - Monte Carlo Methods: Approximate solutions by sampling from belief states and using statistical techniques to estimate value functions.
  - Policy Search Methods: These involve directly searching for policies that maximize expected rewards based on current beliefs rather than computing value functions explicitly.


POMDP
Problem
transition(), observation(), etc.
Solver
Solver, Policy, Updater
transition(), observation(), etc.
solve(), action(), update(), etc.
Experiment
Simulator

Complex Decision Making

- Algorithms for Solving POMDPs | Value Iteration



- Similar to MDPs, value iteration can be adapted for POMDPs by treating the belief state as the state space. The algorithm iteratively updates the value function over belief states until convergence.
- The update rule for the value function in a POMDP is given by:

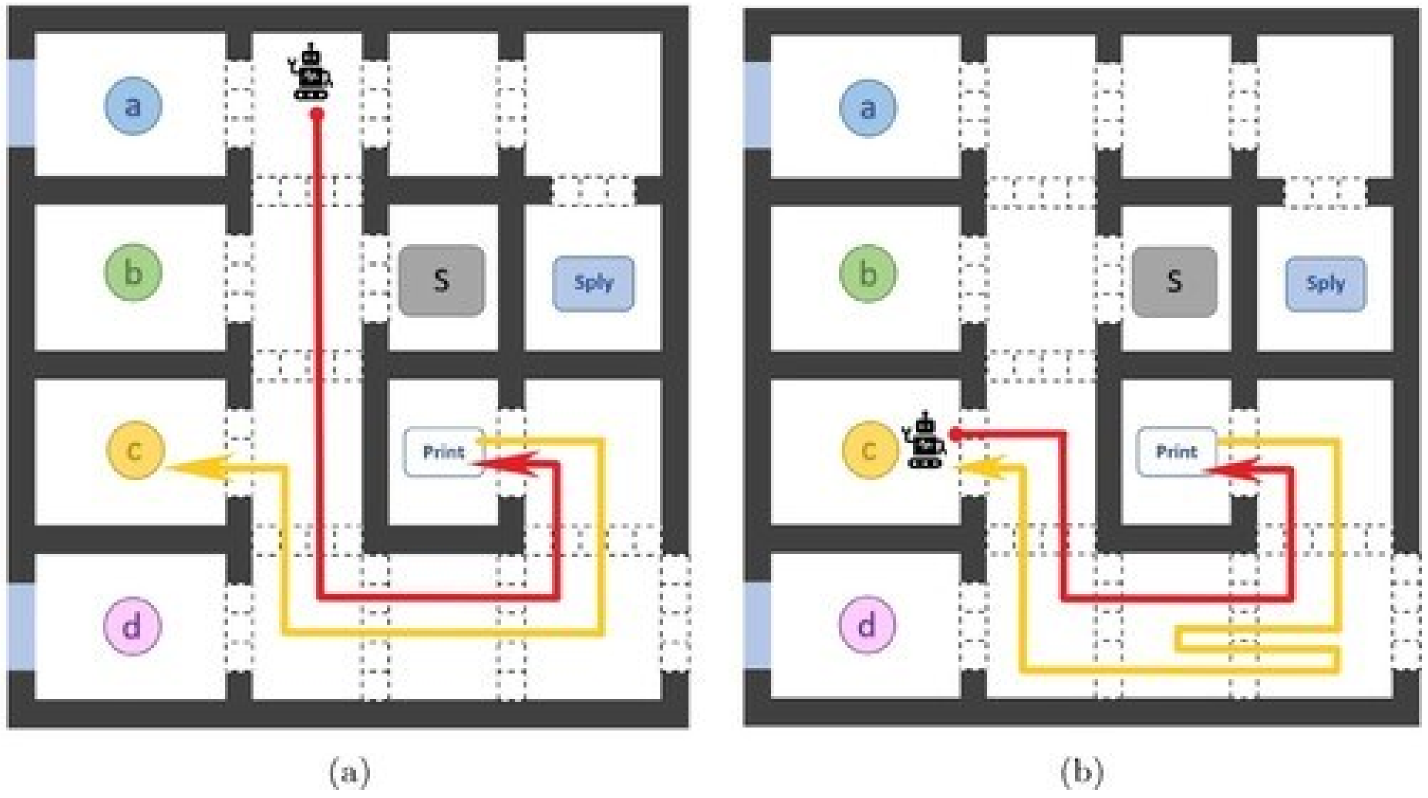
$$V(b) = \max_a \left(R(b,a) + \gamma \sum_o Pr(o|b,a) V(b') \right)$$

- Here, b' represents the new belief state after taking action a and observing o .

Complex Decision Making

Algorithms for Solving POMDPs | Point-Based Value Iteration (PBVI)

- This algorithm focuses on a finite set of belief points rather than the entire belief space, making it more computationally feasible for larger problems.
- It computes value functions only at these sampled points and uses interpolation to estimate values for other beliefs.
- PBVI is particularly useful in practice due to its efficiency in handling large state spaces while providing good approximations of the optimal policy

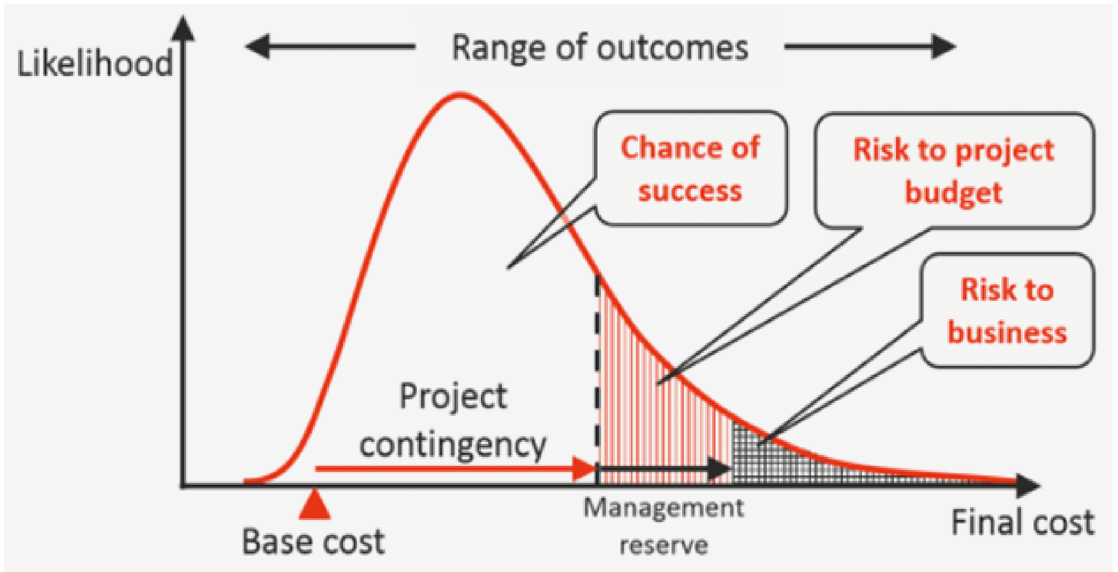


Complex Decision Making

- Algorithms for Solving POMDPs | Monte Carlo Methods
  - Monte Carlo approaches sample from the belief space to estimate value functions or policies. These methods can be particularly effective in high-dimensional spaces where traditional methods struggle.
  - Variants such as Monte Carlo Tree Search (MCTS) can be applied to POMDPs to construct policies based on simulated trajectories through belief states
- Algorithms for Solving POMDPs | Real-Time Dynamic Programming (RTDP)

❑ RTDP algorithms update policies in real-time as new observations are received. They maintain a heuristic function that guides the search through the belief space, allowing for efficient decision-making even in large POMDPs.

❑ The RTDP approach combines aspects of both optimal and heuristic methods, producing good solutions quickly



Complex Decision Making

- Algorithms for Solving POMDPs | SARSOP (Successive Approximations of the Reachable Space under Optimal Policies)
  - This algorithm focuses on reachable belief states and approximates solutions efficiently by iteratively refining a set of belief points that are likely to be visited during execution.
  - SARSOP is particularly effective in large-scale problems where traditional methods may be computationally prohibitive
- Algorithms for Solving POMDPs | Heuristic Search Algorithms
  - Heuristic search techniques leverage domain-specific knowledge to guide exploration in the belief space. These methods can include adaptations of A* search or greedy algorithms that prioritize promising actions based on past experiences