AIES — Unit III Notes
Uncertain Knowledge and Reasoning

Table of Contents

---

1. Uncertainty — Introduction

1. In real life, agents cannot always determine the complete state of the environment — partially observable or non-deterministic environments force agents to act under uncertainty rather than with guaranteed correct knowledge.
2. Uncertainty arises from four main sources — **uncertain data** (missing, noisy, inconsistent), **uncertain knowledge** (multiple causes leading to multiple effects), **uncertain knowledge representation** (restricted or incomplete models of reality), and **uncertain inference** (conclusions drawn from incomplete reasoning may not be fully accurate).
3. Even when a logical plan exists, agents never have access to the entire environment — so they must attach degrees of belief to their conclusions rather than asserting them as absolute truths.
4. Classical logic fails in uncertain domains because it forces every statement to be true or false — probability theory provides a principled mathematical framework for handling degrees of belief.
5. Probabilistic reasoning allows agents to combine prior knowledge with new evidence to update their beliefs, making decisions that are rational even when information is incomplete.

---

2. Basic Probability Notation

2a. Sample Space and Events

1. The **sample space (Ω)** is the set of all possible outcomes of an experiment — for tossing a coin twice, Ω = {HH, HT, TH, TT}; each outcome ω is called a sample point or element.
2. An **event** is any subset of the sample space — for example, the event of getting heads on the first toss is A = {HH, HT}.
3. The **complement** of event A (written A') is the event that A does not occur — $P(A') = 1 - P(A)$.
4. The **intersection** $A \cap B$ is the event containing all outcomes common to both A and B; the **union** $A \cup B$ contains all outcomes in A or B or both.
5. Events are **mutually exclusive** (disjoint) if they share no outcomes — $A \cap B = \{\}$ — and **collectively exhaustive** if their union covers the entire sample space, meaning at least one must always occur.

2b. Axioms of Probability

1. A probability P(A) is a real number assigned to every event A — it must satisfy three axioms: **non-negativity** ($P(A) \geqslant 0$ for all A), **normalisation** ($P(\Omega) = 1$), and **additivity** (if A and B are disjoint, $P(A \cup B) = P(A) + P(B)$).
2. A probability of 0 means the event is impossible; a probability of 1 means it is certain; all other probabilities represent degrees of likelihood between these extremes.
3. The additivity axiom is the most important — for disjoint events the probability of their union equals the sum of individual probabilities, ensuring that probabilities are consistent and do not double-count.
4. From the axioms it follows that $P(A') = 1 - P(A)$ and that $0 \leqslant P(A) \leqslant 1$ for every event A.
5. Probabilities can be expressed as proportions between 0 and 1 or as percentages between 0% and 100% — for example P = 0.45 means a 45% chance of occurrence.

---

3. Joint and Conditional Probability

1. **Joint probability** P(X, Y) is the probability that events X and Y both occur simultaneously — it measures the likelihood of the intersection of two events.
2. **Conditional probability** $P(A|B)$ is the probability of event A occurring given that event B has already occurred — formally $P(A|B) = P(A \cap B) / P(B)$, provided $P(B) > 0$.
3. Conditioning restricts the sample space to outcomes within the conditioning event B — so $P(A|B)$ renormalises probabilities over only those outcomes where B is true.
4. The **product rule** follows directly from conditional probability — $P(A \cap B) = P(A|B) \cdot P(B) = P(B|A) \cdot P(A)$ — this is the foundation of Bayes' theorem.

5. Joint probability tables (full joint distributions) list the probability for every possible combination of values of all random variables and can in principle answer any probabilistic query about the domain.

Notation	Meaning
P(X, Y)	Joint probability of X and Y together
P(A|B)	Conditional probability of A given B
P(A ∩ B)	Probability of both A and B occurring
P(A|B) = P(A∩B)/P(B)	Formula for conditional probability

---

4. Unconditional (Marginal) Probability

1. **Unconditional (marginal) probability** is the probability of a single event without considering any other event — it is also called simple probability.
2. It is computed by summing the joint probabilities over all values of the other variables — this process is called **marginalization** or **summing out**.
3. Marginalisation extracts the distribution over a subset of variables by adding up the probabilities for each possible value of all remaining variables.
4. Example — P(Cavity) is found by summing P(Cavity, Toothache=T) + P(Cavity, Toothache=F) across all values of Toothache, effectively removing Toothache from the calculation.
5. Marginal probability is "unconditional" because the denominator is the full population or sample space — no prior condition has been applied to restrict which outcomes are eligible.

---

5. Independence

1. Two events A and B are **independent** if knowing one gives no information about the other — formally P(A|B) = P(A), which means conditioning on B does not change A's probability.
2. The equivalent definition for independence is P(A ∩ B) = P(A) · P(B) — the joint probability equals the product of the individual probabilities.
3. Independence is a very strong assumption — it means the two variables have absolutely no influence on each other, which is often an idealisation in real-world domains.
4. **Conditional independence** between A and B given C means P(A|B, C) = P(A|C) — A and B may be correlated in general but become independent once C is known.

5. Conditional independence is crucial for Bayesian networks — it allows large joint distributions to be factored into smaller, manageable conditional probability tables (CPTs), drastically reducing storage and computation.

---

6. Inference Using Full Joint Distributions

1. **Probabilistic inference** is the computation of posterior probabilities for query variables given observed evidence — it answers questions like "given that the patient has a stiff neck, what is the probability of meningitis?"
2. The **full joint probability distribution** specifies a probability for every possible complete assignment of values to all random variables — it can answer any query but is exponentially large and impractical for many variables.
3. A **query** is answered by summing (marginalising) over all possible values of hidden (non-query, non-evidence) variables — this is the foundation of exact inference.
4. The **normalisation trick** simplifies calculations — instead of computing the denominator P(evidence) directly, compute the joint for all query values and normalise so they sum to 1.
5. Although the full joint distribution is the gold standard for inference, it requires 2^n entries for n binary variables — Bayesian networks provide a compact factored representation using conditional independence to make inference feasible.

---

7. Bayes' Theorem and Its Use

1. **Bayes' theorem** relates the posterior probability of a hypothesis H given evidence E to the likelihood of the evidence given the hypothesis, the prior probability of the hypothesis, and the marginal probability of the evidence.
2. The formula is: **$P(H|E) = P(E|H) \cdot P(H) / P(E)$** — derived by applying the product rule to both $P(H \cap E)$ and $P(E \cap H)$ and equating them.
3. The four components are — **posterior $P(H|E)$** (what we want to know), **likelihood $P(E|H)$** (how probable is the evidence if the hypothesis is true), **prior $P(H)$** (our belief before seeing evidence), and **marginal $P(E)$** (overall probability of the evidence).
4. Bayes' theorem allows belief to be **updated** when new evidence arrives — the prior is revised by the likelihood ratio to produce the posterior, which then becomes the new prior for the next update.
5. Real-world applications include medical diagnosis (disease given symptoms), spam filtering (spam given word patterns), weather forecasting, and robotics navigation — any domain where a belief must be updated based on noisy or partial evidence.

Classic Example — Meningitis:

- P(stiff neck | meningitis) = 0.8
- P(meningitis) = 1/30,000
- P(stiff neck) = 0.02
- **P(meningitis | stiff neck) = (0.8 × 1/30,000) / 0.02 = 0.00133**

Classic Exam-Style Template:

> $P(A|B) = P(B|A) \cdot P(A) / P(B)$

---

8. Bayesian Networks

1. A **Bayesian Network (BN)** is a directed acyclic graph (DAG) where each node represents a random variable and each directed edge represents a direct probabilistic influence from parent to child.
2. Each node Xi has an associated **Conditional Probability Table (CPT)** — P(Xi | Parents(Xi)) — that quantifies the probability of each value of Xi given every combination of parent values.
3. The network topology encodes **conditional independence** assertions — a node is independent of all its non-descendants given its parents, dramatically reducing the number of probabilities that need to be specified.
4. The **full joint distribution** over all variables can be recovered by multiplying the CPT entries: $P(X1, X2, ..., Xn) = \prod P(Xi \mid Parents(Xi))$ for i = 1 to n.
5. BNs are both compact and expressive — they can represent complex probabilistic relationships using far fewer parameters than the full joint distribution, and they support efficient inference algorithms.

Classic Example — Toothache/Cavity:

- Variables: Weather, Cavity, Toothache, Catch
- Topology: Weather is independent of all others; Toothache and Catch are conditionally independent given Cavity.
- Cavity is the parent of both Toothache and Catch; Weather has no parents and no connections.

8a. Semantics of Bayesian Networks

1. A BN has two equivalent interpretations — as a **compact representation of the full joint distribution**, and as an **encoding of conditional independence statements** among variables.
2. As a representation of the joint: every entry in the full joint can be calculated by multiplying the appropriate CPT entries, making the network a complete description of the domain.

3. As an encoding of independence: the topology directly specifies which variables are conditionally independent of which others — this guides both construction of the network and design of inference procedures.
4. The intuitive meaning of a directed arrow from X to Y is that X has a **direct causal influence** on Y — though arrows represent probabilistic influence, not strict causation.
5. Both interpretations are mathematically equivalent — choosing one or the other is a matter of convenience for a given task (construction vs. inference).

8b. Constructing Bayesian Networks

1. Choose a causal **ordering** of variables X1, X2, ..., Xn — causes should come before effects in this ordering to produce a natural and minimal network structure.
2. For each variable Xi, select parents from the preceding variables X1, ..., Xi-1 such that P(Xi | Parents(Xi)) = P(Xi | X1, ..., Xi-1) — this is the **Markov condition**.
3. The **chain rule** guarantees correctness — P(X1, ..., Xn) = ∏ P(Xi | X1, ..., Xi-1) and by the Markov condition this equals ∏ P(Xi | Parents(Xi)).
4. Specify the CPT for each node by eliciting probabilities from domain experts or learning them from data — a node with k binary parents needs a CPT with 2^k rows.
5. Choosing a good variable ordering matters — a natural causal ordering typically yields a sparse network with few edges and small CPTs; a bad ordering can create unnecessarily dense networks.

---

9. Exact Inference in Bayesian Networks

1. **Exact inference** computes the exact posterior probability P(Query | Evidence) by systematic manipulation of the joint distribution encoded in the network.
2. The most common exact method is **Variable Elimination (VE)** — it sums out hidden variables one at a time, performing pointwise products of factors to compute the marginal distribution of the query variable.
3. Factors are intermediate tables produced by multiplying and marginalising CPTs — VE avoids recomputing the same factor multiple times by caching intermediate results.
4. The complexity of variable elimination depends on the **treewidth** of the network — for polytree (singly-connected) networks inference is linear; for multiply-connected networks it can be exponential.
5. Exact inference is guaranteed to return the correct answer, but it becomes **intractable** for large multiply-connected networks — approximate methods are needed in those cases.

---

10. Approximate Inference in Bayesian Networks

1. **Approximate inference** (also called Monte Carlo inference) estimates probabilities by generating random samples from the network rather than computing exact answers — it trades accuracy for tractability.
2. The two main families of algorithms are **Direct Sampling** (generate samples from the prior distribution) and **Markov Chain Sampling** (generate samples by making random local changes to a current state).
3. **Rejection Sampling** — generate samples from the prior, then reject any sample that does not match the observed evidence; estimate P(Query|Evidence) by counting Query values in the remaining samples.
4. **Markov Chain Monte Carlo (MCMC) / Gibbs Sampling** — start from a random state consistent with evidence, then repeatedly sample one variable at a time conditioned on the **Markov Blanket** (parents, children, and children's other parents); tally results over many iterations.
5. As the number of samples grows, the frequency of outcomes converges to the true probability — approximate methods are the practical choice when exact inference is #P-hard due to network complexity.

Method	Description	When to Use
Direct Sampling	Sample from prior distribution	When evidence is not rare
Rejection Sampling	Sample then reject non-evidence samples	Simple conditional queries
MCMC / Gibbs	Perturb current sample using Markov Blanket	Large multiply-connected networks

---

11. Fuzzy Logic

1. **Fuzzy logic** is a computing framework based on **degrees of truth** rather than the strict binary values (0 or 1) of classical Boolean logic — it handles imprecise, vague, or uncertain information naturally.
2. Invented by **Lotfi Zadeh**, fuzzy logic imitates human reasoning, which operates with intermediate possibilities such as "certainly yes," "possibly yes," "cannot say," "possibly no," and "certainly no" rather than just yes/no.
3. Unlike Boolean logic where temperature is simply HIGH or LOW, fuzzy logic allows gradations — "very cold," "cold," "warm," "very warm," "hot" — making it better suited to real-world control problems.

4. A **fuzzy set** is characterised by a **membership function** that maps each element to a value between 0 and 1, representing the degree to which that element belongs to the set (not just whether it does or does not).
5. Key advantages of fuzzy logic include handling imprecise inputs, resembling human reasoning, having a simple rule-based structure that is easy to understand and modify, and requiring little data to operate.

11a. Fuzzy Logic Systems Architecture

A Fuzzy Logic System (FLS) has four main components:

1. **Fuzzification Module** — converts crisp (precise numeric) input values into fuzzy membership values using membership functions; the input is split into linguistic categories (e.g., Large Positive, Medium Positive, Small, Medium Negative, Large Negative).
2. **Knowledge Base** — stores the IF-THEN rules provided by domain experts; each rule maps combinations of linguistic input values to linguistic output actions (e.g., IF temperature is Cold AND target is Warm THEN action is Heat).
3. **Inference Engine** — simulates human reasoning by evaluating all applicable IF-THEN rules using fuzzy operations (AND = Min, OR = Max) and combining their conclusions to produce a fuzzy output set.
4. **Defuzzification Module** — converts the fuzzy output set produced by the inference engine back into a single crisp numeric value suitable for controlling an actuator or producing a precise output.
5. The FLS pipeline is: **Crisp Input → Fuzzification → Knowledge Base + Inference Engine → Defuzzification → Crisp Output** — it can produce acceptable definite outputs even from incomplete, noisy, or ambiguous inputs.

```
Crisp Input → [Fuzzification] → [Inference Engine ↔ Knowledge Base] →
[Defuzzification] → Crisp Output
```

11b. Membership Functions

1. A **membership function** μA(x) maps each element x in the universe of discourse X to a value in [0, 1] — this value is the **degree of membership** of x in fuzzy set A.
2. The x-axis of a membership function graph represents the universe of discourse (e.g., speed values); the y-axis represents degrees of membership between 0 and 1.
3. A degree of 0 means the element does not belong to the fuzzy set at all; a degree of 1 means full membership; fractional values represent partial membership.
4. Common membership function shapes include **triangular**, **trapezoidal**, and **bell-shaped** curves — the choice of shape depends on the domain and the granularity needed.
5. Example — for a fuzzy set "Fast" on speed: μ(x) = 0 if speed ⩽ 40, (speed − 40)/10 if 40 < speed < 50, and 1 if speed ⩾ 50; so speed=42 has membership 0.2 and speed=45 has

membership 0.5.

11c. Development of a Fuzzy Logic System (Air Conditioner Example)

Step 1 — Define Linguistic Variables and Terms

- Input: Room Temperature — {Very_Cold, Cold, Warm, Hot, Very_Hot}
- Input: Target Temperature — {Very_Cold, Cold, Warm, Hot, Very_Hot}
- Output: Action — {Cool, No_Change, Heat}

Step 2 — Construct Membership Functions

- Each linguistic term is represented by a graphical membership function over the numerical temperature range — for example "Warm" might peak at 25°C with triangular shape.

Step 3 — Construct Knowledge Base (IF-THEN Rules)

Condition	Action
IF temperature = (Cold OR Very_Cold) AND target = Warm	Heat
IF temperature = (Hot OR Very_Hot) AND target = Warm	Cool
IF temperature = Warm AND target = Warm	No_Change

Step 4 — Fuzzification and Inference (Obtain Fuzzy Output)

- Convert actual temperature reading into fuzzy membership values.
- Evaluate all rules using Min (AND) and Max (OR) operations.
- Combine all rule outputs into a single fuzzy output set.

Step 5 — Defuzzification

- Apply a defuzzification method (e.g., centroid/centre-of-gravity) to convert the fuzzy output into a precise control signal (e.g., set AC fan speed to a specific RPM).

Algorithm Summary:

1. Define linguistic variables and terms
2. Construct membership functions
3. Build IF-THEN rule knowledge base
4. Fuzzify crisp inputs using membership functions
5. Evaluate rules in inference engine
6. Combine rule results
7. Defuzzify to get crisp output

11d. Advantages and Disadvantages of Fuzzy Logic Systems

Advantages	Disadvantages
Mathematical concepts are simple	No systematic approach to system design
Rules can be added or deleted easily	Complex FLS are hard to understand
Handles imprecise, noisy input	Not suitable when high precision is needed
Resembles human reasoning	Verification and validation is difficult
Applicable to complex real-world problems	Results may not always be optimal

Application Areas: Automotive (automatic gearboxes, four-wheel steering), Consumer electronics (cameras, TVs), Domestic appliances (washing machines, microwave ovens, refrigerators), Environment control (air conditioners, heaters, humidifiers).

Quick Comparison — Key Concepts

Concept	Key Idea	Exam Focus
Joint Probability	$P(X,Y)$ — both events together	Formula and table reading
Conditional Probability	$P(A\|B) = P(A \cap B)/P(B)$	Derivation and problems
Bayes' Theorem	$P(H\|E) = P(E\|H) \cdot P(H)/P(E)$	Numeric problems (meningitis, insurance)
Bayesian Network	DAG + CPTs encoding joint distribution	Structure, CPT, inference
Approximate Inference	Monte Carlo sampling methods	Rejection sampling, MCMC
Fuzzy Logic	Degree of truth in [0,1], not just 0 or 1	AC example, membership functions, architecture

---

End of AIES Unit III Notes