Uncertainty in AI

Unit 4

Uncertainty in AI:

- In AI, uncertainty refers to the inability to determine the exact outcome or state of a system due to various factors such as **incomplete information**, **inherent randomness**, Noisy or **Contradictory Data**, or **ambiguity** in the data or environment.
- AI systems frequently deal with uncertainty in real-world applications such as medical diagnosis, robotics, autonomous driving, and natural language processing.

Uncertainty in AI can arise from several sources, including:

- **Incomplete Information**: AI systems may not have access to all relevant information needed to make accurate predictions or decisions.
- **Ambiguity**: Data or observations may be ambiguous or subject to interpretation, leading to uncertainty about their meaning or implications.
- **Noise**: Data collected from sensors or real-world observations may contain random fluctuations or errors, introducing uncertainty into the analysis.
- **Modeling Assumptions**: AI models and algorithms often make simplifying assumptions about the underlying data or environment, leading to uncertainty when these assumptions do not hold true in practice.

Need of Considering Probability

- Take a self-driving car for example — you can set the goal to get from A to B in an efficient and safe manner that follows all laws.
- But what happens if the traffic gets worse than expected, maybe because of an accident ahead? Sudden bad weather?
- Random events like a ball bouncing in the street, or a piece of trash flying straight into the car’s camera?
- Self driving cars work on different sensors, which are not perfect always.

Causes of Uncertainty

- Information occurred from unreliable sources.
- Experimental Errors
- Equipment fault
- Temperature variation
- Climate change.

History of dealing with uncertainty

- Fuzzy logic was for a while a contender for the best approach to handle uncertain and imprecise information.
- Probability has turned out to be the best approach for reasoning under uncertainty, and almost all current AI applications are based, to at least some degree, on probabilities.

Why probability matters

- What are the chances of getting three of a king in poker? (about 1 in 47)
- What are the chances of winning in the lottery? (very small)

- Probability is used to quantify and compare risks in everyday life.
- What are the chances of crashing your car if you exceed the speed limit?

Probabilistic reasoning

- Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge.
- In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.

Need of probabilistic reasoning in AI

- When there are unpredictable outcomes.
- When specifications or possibilities of predicates becomes too large to handle.
- When an unknown error occurs during an experiment.

Probability

- Probability can be defined as a chance that an uncertain event will occur.
- It is the numerical measure of the likelihood that an event will occur.
- The value of probability always remains between 0 and 1 that represent ideal uncertainties.

- $0 \leq P(A) \leq 1$, where $P(A)$ is the probability of an event A.

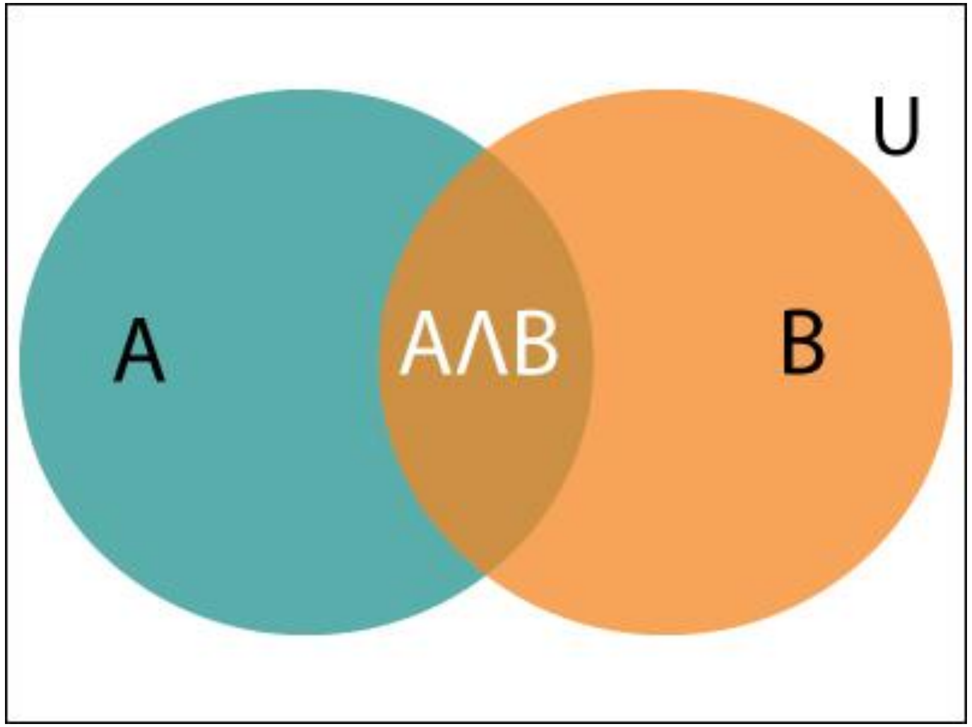
Conditional Probability

- Conditional probability is a probability of occurring an event when another event has already happened.

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

- Where $P(A \wedge B)$= Joint probability of A and B
- $P(B)$= Marginal probability of B.

Venn diagram - Conditional probability



Example

- In a class, there are 70% of the students who like English and 40% of the students who likes English and mathematics, and then what is the percent of students those who like English also like mathematics?

Solution

- Let, A is an event that a student likes Mathematics
- B is an event that a student likes English.
- $$P(A|B) = \frac{P(A \wedge B)}{P(B)} = \frac{0.4}{0.7} = 57\%$$
- Hence, 57% are the students who like English also like Mathematics.

What is Probability?

Definition:
Probability is "the chance that a given event will occur." (Webster's Dictionary)

Examples in Daily Life:

- Probability of rain tomorrow
- Probability of winning the lottery

Understanding Probability:

- Assigned a value between **0 and 1** (or **0% - 100%**)
- Example: *30% chance of rain* means that in similar conditions, rain occurs **3 out of 10 times**

Random Experiments & Probability:

Random Experiment: An event with uncertain outcomes (e.g., coin toss, lottery draw)

Set of Outcomes: Possible results of the experiment

Probability Assignment:

- Coin Toss: 50% Heads, 50% Tails
- Lottery: 1/N (N = total number of possible outcomes)

Statistical Regularity:

- Probability helps predict *averages* even when individual outcomes are uncertain.
- Example: Average summer rainfall:
    - Rhode Island = **9.76 inches**
    - Arizona = **4.40 inches**

Types of Probability Problems

1. Discrete Probability

- Used when the number of outcomes is **finite** or **countably infinite**.
- Example: Number of people on the phone in an office between 9:00 AM and 9:10 AM.
  - Possible outcomes: **{0, 1, 2, ..., N}**
  - If each outcome is equally likely, probability of each = **1 / (N + 1)**.

2. Continuous Probability

- Used when the number of outcomes is **uncountably infinite**.
- Example: Duration of a phone call within a 10-minute window.
  - Possible outcomes: Any value in the interval **[0, T]** where T = 10.
  - Since outcomes are infinite, assigning nonzero probabilities to each outcome is not feasible.
- **Discrete probability** deals with **countable** outcomes and assigned probabilities.
- **Continuous probability** deals with **uncountable** outcomes, requiring probability density functions.

Probabilistic Modeling

- **What is a Probability Model?**
  - A probability model is a **simplified representation** of a real-world random event.
  - It should be **detailed enough** to capture important patterns but **simple enough** to be useful.

Example: Telephone Callers

- Suppose we want to model how many people in an office are on the phone at a given time.
- We assign a probability **p** to each person being on the phone.
- If there are **4 people**, this is similar to flipping 4 coins, where:
  - **Heads** = Person is on the phone
  - **Tails** = Person is not on the phone
- The probability of exactly **3 people** being on the phone can be calculated using probability formulas.

Limitations of the Model

- **Different probabilities:** Some people may use the phone more than others.
- **Dependence:** If one person is on the phone, their neighbor might be more likely to be on too.
- **Time variation:** Phone usage changes throughout the day.

Keeping Models Simple

- A model should be **as simple as possible** while still being accurate.
- If a model is too complex, it might not be practical to use.

Continuous Probability Example

- Instead of counting calls, we could model **call duration** using a **bell-shaped curve (Gaussian distribution)**.
- Instead of predicting an exact call time (which has zero probability), we predict a **range** (e.g., call lasts between 5 and 6 minutes).
- The probability of a call lasting 7 minutes is the highest, while very short or very long calls are less likely.

Review of Set Theory

- **What is a Set?**
- A **set** is a **collection of distinct objects**.
- Example: The set of students in a probability class:
  - **Enumeration Method:** A = {Jane, Bill, Jessica, Fred}
  - **Description Method:** A = {students enrolled in the probability class}
- **Elements and Subsets**
  - **Elements**: Objects within a set (e.g., "Bill" is an element of A).
  - **Subsets**: A set B is a **subset** of A if all elements of B are also in A.
  - Example: B = {Bill, Fred} is a subset of A.
  - Notation: $B \subseteq A$

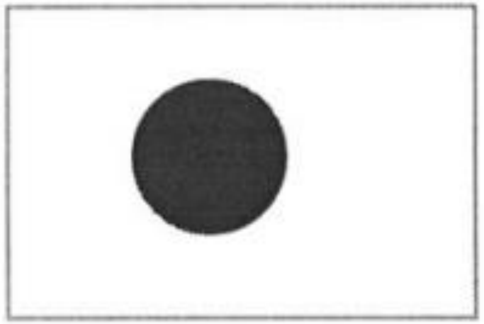
Special Types of Sets

- **Empty Set (Ø)**: A set with no elements (e.g., no students got an "A" grade).
- **Universal Set (S)**: The set containing all possible elements in a discussion (e.g., all students in class).

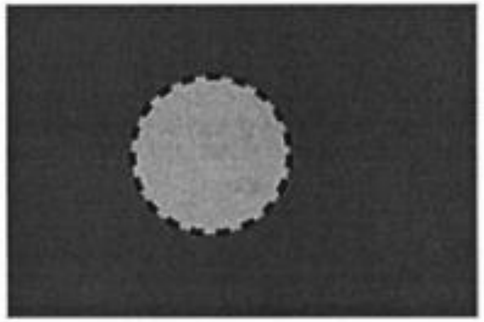
Set Operations

- **Union (A ∪ B)** – The set of elements in A **or** B or both.
  - Example: If A = {1,2,3} and B = {3,4,5}, then A ∪ B = {1,2,3,4,5}.
- **Intersection (A ∩ B)** – The set of elements in **both** A and B.
  - Example: A ∩ B = {3}.
- **Difference (A - B)** – The set of elements in A but **not** in B.
  - Example: A - B = {1,2}.
- **Complement (A^c)** – The set of elements **not in A** (within the universal set).

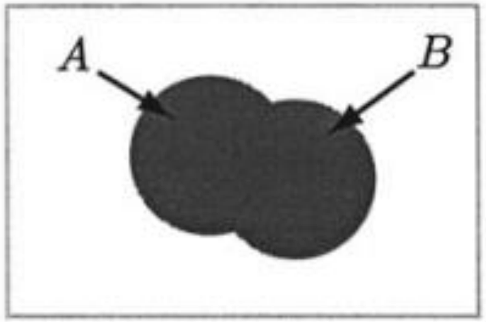
(a) Universal set $\mathcal{S}$



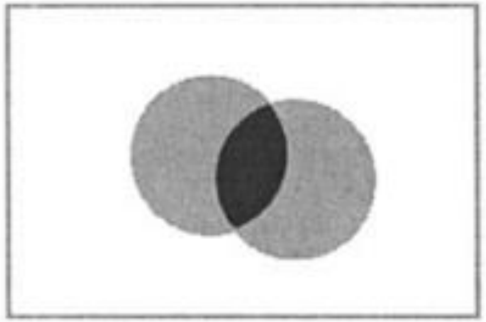
(b) Set A



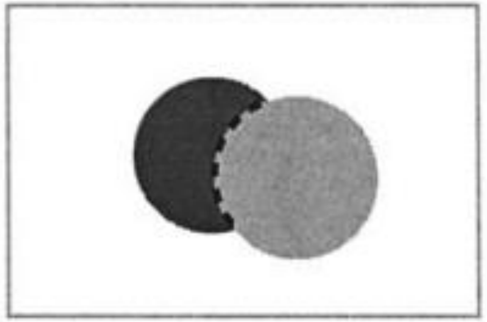
(c) Set A^c



(d) Set $A \cup B$



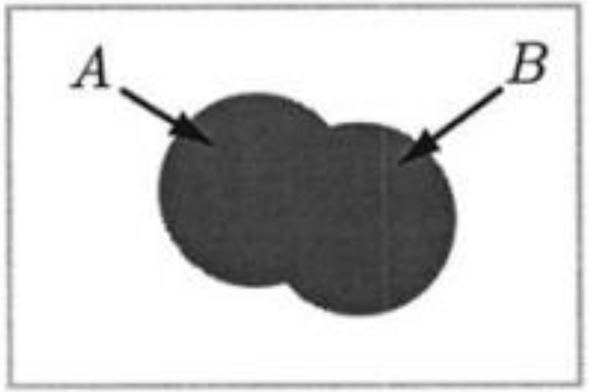
(e) Set $A \cap B$



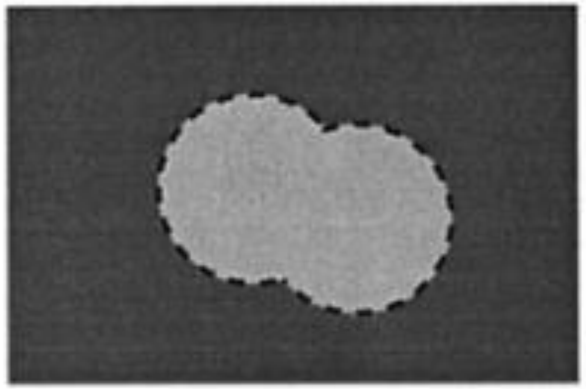
(f) Set $A - B$

De Morgan’s Laws

- $(A \cup B)^c = A^c \cap B^c$
- $(A \cap B)^c = A^c \cup B^c$



(a) Set $A \cup B$

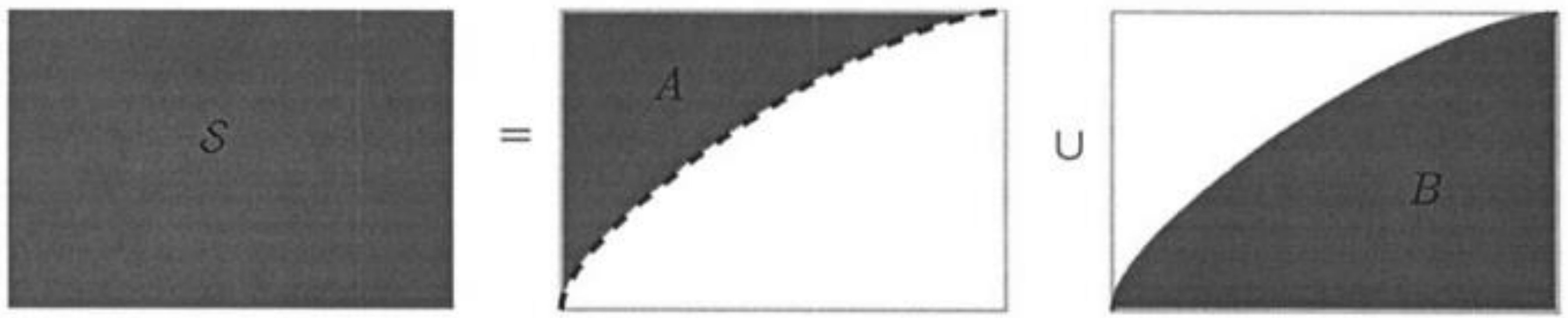


(b) Set $A^c \cap B^c$

Figure 3.4: Illustration of De Morgan’s law.

Disjoint and Partitioning Sets

- **Disjoint Sets**: Two sets with no common elements (**A ∩ B = Ø**).
- **Partitioning a Set**: Dividing a set into **non-overlapping** subsets that cover the entire universal set.



Finite, Countable, and Continuous Sets

- **Finite Sets**: Contain a limited number of elements (e.g., {1,2,3}).
- **Countably Infinite Sets**: Can be listed (e.g., {1,2,3,…}).
- **Continuous Sets**: Cannot be counted individually (e.g., $\{x \mid 0 \le x \le 1\}$, representing a line segment).

Assigning and Determining Probabilities

- **Sample Space (S):** The set of all possible outcomes of an experiment.
- **Event (E):** A subset of the sample space, representing outcomes of interest.
- **Simple Events:** Events containing only a single outcome (e.g., rolling a 2 on a die).
- **Certain Event:** The entire sample space (**S**) since an outcome always occurs.
- **Impossible Event:** The empty set (**Ø**) since no outcome can occur

Example: Tossing a Fair Die

- **Sample Space (S):** {1, 2, 3, 4, 5, 6}
- **Even Numbers (E_e):** {2, 4, 6}
- **Probability of rolling an even number:**
  - Since each outcome is equally likely, probability of any one number = **1/6**.
  - Probability of E_e = **P(2) + P(4) + P(6) = 1/6 + 1/6 + 1/6 = 3/6 = 1/2**.

Axioms of Probability

- **Non-Negativity:** Probability of any event is **≥ 0** ($P(E) \geq 0$).
- **Total Probability:** Probability of the entire sample space is **1** ($P(S) = 1$).
- **Addition Rule for Mutually Exclusive Events:** If two events cannot happen together (**disjoint events**), their combined probability is: $\mathbf{P(A \cup B) = P(A) + P(B)}$.

Example: Loaded Die

- If a die is unfair and **6 appears twice as often** as the other numbers: Probability of {1,2,3,4,5} = **1/7** each.
- Probability of {6} = **2/7**.
- Still, total probability = **1** (as per Axiom 2).

Properties of the Probability Function

- Probability functions follow several important properties that help in evaluating probabilities. These properties are derived from the **four fundamental axioms of probability**.

1. Probability of a Complement

$$P(E^C) = 1 - P(E)$$

- The probability of an event **not occurring** is **1 minus the probability of the event occurring**.
- Example: If the probability of rain today is **0.7**, the probability of no rain is **1 - 0.7 = 0.3**.

2. Probability of the Impossible Event

$$P(\emptyset) = 0$$

- The probability of an **impossible event** (i.e., an event that cannot happen) is always **zero**.
- Example: Rolling a **7** on a standard six-sided die is **impossible**, so $P(7) = 0$.

3. Probability is Always Between 0 and 1

$$0 \leq P(E) \leq 1$$

- Every probability must be between **0 (impossible)** and **1 (certain)**.

4. Probability of Union of Two Events

If **E and F are not mutually exclusive**, then:

$$P(E \cup F) = P(E) + P(F) - P(E \cap F)$$

- This ensures that the **overlap (intersection) is not counted twice.**
- Example:
  - Probability of rolling an **even number (2, 4, 6) = 3/6**
  - Probability of rolling a **number ≤ 3 (1, 2, 3) = 3/6**
  - Probability of rolling **2** (which is in both sets) = **1/6**
  - Final probability:

$$P(E \cup F) = 3/6 + 3/6 - 1/6 = 5/6$$

5. Monotonicity of Probability Function

If E is a subset of F (E ⊆ F), then:

$$P(E) \leq P(F)$$

- A larger event has a higher or equal probability compared to its subset.
- Example: The probability of rolling **an even number (2,4,6)** is ≤ the probability of rolling **any number (1,2,3,4,5,6)**.

6. Probability of a Union of Three Events

For three events:

$$P(E_1 \cup E_2 \cup E_3) = P(E_1) + P(E_2) + P(E_3) - P(E_1 \cap E_2) - P(E_1 \cap E_3) - P(E_2 \cap E_3) + P(E_1 \cap E_2 \cap E_3)$$

- This formula prevents overcounting of multiple intersections.

7. Boole’s Inequality (Union Bound)

$$P(E_1 \cup E_2 \cup ... \cup E_n) \leq P(E_1) + P(E_2) + ... + P(E_n)$$

- The probability of the **union of events** is at most the **sum of their individual probabilities.**
- Useful for estimating an upper bound.

Example: Probability in Parallel and Series Circuits

Parallel Circuit (At Least One Switch Works)

- P(Switch 1 works) = 1/2, P(Switch 2 works) = 1/2
- P(Both switches work) = 1/4
- Using the union formula:

$$P(\text{Circuit Works}) = P(S_1) + P(S_2) - P(S_1 \cap S_2) = \frac{1}{2} + \frac{1}{2} - \frac{1}{4} = \frac{3}{4}$$

Series Circuit (Both Switches Must Work)

- P(Both switches work) = 1/4
- Series circuits have **lower reliability** than parallel circuits.

HOME WORK

Create two probability questions based on the previously discussed concepts.

Conditional Probability

What is Conditional Probability?

- **Definition:** The probability of an event **A**, given that another event **B** has already occurred.
- This is denoted as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

  - **P(A ∩ B):** Probability of both A and B occurring.
  - **P(B):** Probability of B occurring.

Example: Conditional Probability

- Suppose we roll a fair die.
- **Event A:** Rolling a **4**.
- **Event B:** Rolling an **even number (2,4,6)**.
- The probability of rolling a **4**, given that we already know the outcome is even:

$$P(4|even) = \frac{P(4 \cap even)}{P(even)} = \frac{1/6}{3/6} = \frac{1}{3}$$

- Since we already know the number is **even**, we exclude odd numbers and recalculate probabilities.

Concepts in Conditional Probability

- **Joint Probability (P(A ∩ B))**
  - The probability that **both** events A and B happen together.
  - Example: Probability that a randomly chosen person is both **taller than 6 feet and weighs over 200 lbs.**
- **Marginal Probability (P(A))**
  - The probability of an event **without considering other conditions**.
  - Example: Probability that a randomly chosen student weighs **over 200 lbs.** without considering height.

Concepts in Conditional Probability

Law of Total Probability

- If events B_1, B_2, ..., B_n form a **partition** of the sample space (i.e., they cover all possibilities), then:

$$P(A) = P(A|B_1)P(B_1) + P(A|B_2)P(B_2) + ... + P(A|B_n)P(B_n)$$

- **Example:** In a school, students are either **male (B_1) or female (B_2).** If we want the probability that a randomly chosen student is in the **math club (A)**, we sum over all groups:

$$P(math) = P(math|male)P(male) + P(math|female)P(female)$$

Example

- Suppose a weather forecaster predicts that there is a 30% chance of rain tomorrow. Based on historical data, they know that when it rains, there's an 80% chance that it will be cloudy. On the other hand, when it doesn't rain, there's only a 20% chance of it being cloudy. What is the probability that it will be cloudy and raining tomorrow?

Solution: Probability of Rain and Cloudy Weather

Using the definition of **joint probability**:

$$P(\text{Cloudy and Rain}) = P(\text{Cloudy}|\text{Rain}) \times P(\text{Rain})$$

where:

- $P(\text{Rain}) = 0.30$ (30% chance of rain)
- $P(\text{Cloudy}|\text{Rain}) = 0.80$ (80% chance of being cloudy when it rains)

Now, calculating:

$$P(\text{Cloudy and Rain}) = 0.80 \times 0.30 = 0.24$$

Bayes' theorem

- Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.
- In probability theory, it relates the conditional probability and marginal probabilities of two random events.
- Bayes' theorem was named after the British mathematician **Thomas Bayes**. The **Bayesian inference** is an application of Bayes' theorem, which is fundamental to Bayesian statistics.

Bayes' Theorem

- **Used to update probabilities when we get new information.**
- Formula:

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

- **Example: Medical Test**
  - Suppose **5% of a population** has a disease.
  - A test is **90% accurate** for infected people but **10% false positive** for healthy people.
  - If a person tests positive, what is the probability they actually have the disease?
  - Using Bayes' Theorem:

$$P(Disease|Positive) = \frac{P(Positive|Disease)P(Disease)}{P(Positive)}$$

Solution for the Medical Test Problem

Using **Bayes' Theorem**, we calculate:

$$P(Disease|Positive) = \frac{P(Positive|Disease)P(Disease)}{P(Positive)}$$

where:

- $P(Disease) = 0.05$ (5% of the population has the disease)
- $P(Positive|Disease) = 0.90$ (90% accurate test)
- $P(Positive|NoDisease) = 0.10$ (10% false positives)
- $P(NoDisease) = 1 - 0.05 = 0.95$

Solution for the Medical Test Problem

Using the **Law of Total Probability:**

$$P(Positive) = P(Positive|Disease)P(Disease) + P(Positive|NoDisease)P(NoDisease)$$

$$= (0.90 \times 0.05) + (0.10 \times 0.95) = 0.045 + 0.095 = 0.14$$

Now, applying Bayes' Theorem:

$$P(Disease|Positive) = \frac{0.90 \times 0.05}{0.14} = \frac{0.045}{0.14} \approx 0.3214$$

Examples on Bayes' Theorem

Example 1: Spam Email Detection

- A company uses a spam filter to detect spam emails. The probability of an email being spam is **30%**. The filter correctly identifies spam **95% of the time** but also wrongly classifies a **legitimate email as spam 5% of the time**. If an email is flagged as spam, what is the probability that it is actually spam?

Example 2: Drug Testing

- A company conducts drug tests for its employees. **2%** of employees actually use a certain drug. The test correctly identifies a drug user **99% of the time** but has a **2% false positive rate**. If an employee tests positive, what is the probability they are actually using the drug?

Solution for Example 1: Spam Email Detection

Using **Bayes' Theorem**, we calculate:

$$P(Spam|Flagged) = \frac{P(Flagged|Spam)P(Spam)}{P(Flagged)}$$

where:

- $P(Spam) = 0.30$ (30% of emails are spam)
- $P(Flagged|Spam) = 0.95$ (95% accuracy in detecting spam)
- $P(Flagged|NotSpam) = 0.05$ (5% false positives)
- $P(NotSpam) = 1 - 0.30 = 0.70$

Solution for Example 1: Spam Email Detection

Using the **Law of Total Probability**:

$$P(Flagged) = P(Flagged|Spam)P(Spam) + P(Flagged|NotSpam)P(NotSpam)$$

$$= (0.95 \times 0.30) + (0.05 \times 0.70) = 0.285 + 0.035 = 0.32$$

Now, applying Bayes' Theorem:

$$P(Spam|Flagged) = \frac{0.95 \times 0.30}{0.32} = \frac{0.285}{0.32} \approx 0.8906$$

Solution for Example 2: Drug Testing

Using **Bayes' Theorem**, we calculate:

$$P(DrugUser|Positive) = \frac{P(Positive|DrugUser)P(DrugUser)}{P(Positive)}$$

where:

- $P(DrugUser) = 0.02$ (2% of employees use the drug)
- $P(Positive|DrugUser) = 0.99$ (99% accuracy in detecting drug users)
- $P(Positive|NoDrugUser) = 0.02$ (2% false positives)
- $P(NoDrugUser) = 1 - 0.02 = 0.98$

Solution for Example 2: Drug Testing

Using the **Law of Total Probability**:

$$P(Positive) = P(Positive|DrugUser)P(DrugUser) + P(Positive|NoDrugUser)P(NoDrugUser)$$

$$= (0.99 \times 0.02) + (0.02 \times 0.98) = 0.0198 + 0.0196 = 0.0394$$

Now, applying Bayes' Theorem:

$$P(DrugUser|Positive) = \frac{0.99 \times 0.02}{0.0394} = \frac{0.0198}{0.0394} \approx 0.5025$$

Example

- Suppose a weather forecaster predicts that there is a 30% chance of rain tomorrow. Based on historical data, they know that when it rains, there's an 80% chance that it will be cloudy. On the other hand, when it doesn't rain, there's only a 20% chance of it being cloudy. What is the probability that it will be raining tomorrow given that it's cloudy?

We are given:

- $P(R) = 0.30$ → Probability of rain
- $P(C|R) = 0.80$ → Probability of clouds given that it rains
- $P(C|R^c) = 0.20$ → Probability of clouds given that it does not rain
- $P(R^c) = 1 - P(R) = 0.70$ → Probability of no rain

We need to find $P(C)$, **the probability of clouds.**

Using the **Law of Total Probability**:

$$P(C) = P(C|R)P(R) + P(C|R^c)P(R^c)$$

$$P(C) = (0.80 \times 0.30) + (0.20 \times 0.70)$$

$$P(C) = 0.24 + 0.14 = 0.38$$

Now, we find $P(R|C)$, the probability that it will rain given that it's cloudy, using Bayes' Theorem:

$$P(R|C) = \frac{P(C|R)P(R)}{P(C)}$$

$$P(R|C) = \frac{(0.80 \times 0.30)}{0.38}$$

$$P(R|C) = \frac{0.24}{0.38} = 0.6316$$

Statistically Independent Events

Two events A and B are **statistically independent** if the occurrence of one event **does not** affect the probability of the other:

$$P(A|B) = P(A)$$

From this, we derive the condition for statistical independence:

$$P(A \cap B) = P(A)P(B)$$

This means that the probability of both events occurring together is simply the product of their individual probabilities.

Example 1: Rolling a Fair Die

- Event A: Rolling a 2 or 3 → $P(A) = 2/6 = 1/3$
- Event B: Rolling an even number (2, 4, 6) → $P(B) = 3/6 = 1/2$
- Joint Probability: $P(A \cap B) = P(2) = 1/6$
- Checking Independence:

$$P(A \cap B) = P(A)P(B) \Rightarrow \frac{1}{6} = \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$$

Since this holds, A and B are independent.

Difference Between Independence and Mutual Exclusivity

- **Mutually Exclusive Events:** If one event happens, the other **cannot**.
  - $P(A \cap B) = 0$
  - Example: Rolling an even number and rolling an odd number are mutually exclusive.
- **Independent Events:** The occurrence of one event **does not** change the probability of the other.
  - Example: Rolling a **3** and flipping **heads** on a coin toss are independent.

Independence in Three Events

For three events A, B, C to be independent:

$$P(A \cap B) = P(A)P(B)$$

$$P(A \cap C) = P(A)P(C)$$

$$P(B \cap C) = P(B)P(C)$$

$$P(A \cap B \cap C) = P(A)P(B)P(C)$$

- If only the first three conditions hold but **not** the fourth, the events are **pairwise independent** but **not fully independent.**

Example 2: Probability Chain Rule

Instead of assuming independence, we can use the **chain rule** to calculate joint probability:

$$P(A \cap B \cap C) = P(A|B, C)P(B|C)P(C)$$

This is useful when probabilities are **not independent**.

Derivation of Bayes' Theorem

Step 1: Definition of Conditional Probability

The conditional probability of an event A given that event B has occurred is defined as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Similarly, the conditional probability of event B given that event A has occurred is:

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

Since both equations contain $P(A \cap B)$, we can rewrite it as:

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

Derivation of Bayes' Theorem

Step 2: Solving for $P(A|B)$

Rearranging the above equation:

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

This is the **Bayes' Theorem**.

Derivation of Bayes' Theorem

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

- $P(A|B)$ → Probability of A given that B has occurred (Posterior probability).
- $P(B|A)$ → Probability of B given A (Likelihood).
- $P(A)$ → Prior probability of A (before considering B).
- $P(B)$ → Total probability of B, calculated as:

$$P(B) = P(B|A)P(A) + P(B|A^c)P(A^c)$$

(where A^c is the complement of A).

Example

Question: What is the probability that a patient diagnosed with a stiff neck also has meningitis?

Given Data:

- Meningitis leads to a stiff neck in **80% of cases** ($P(\text{Stiff Neck}|\text{Meningitis}) = 0.80$).
- The probability of a person having meningitis is **1 in 30,000** ($P(\text{Meningitis}) = \frac{1}{30{,}000}$).
- The probability of any patient experiencing a stiff neck is **2%** ($P(\text{Stiff Neck}) = 0.02$).

Using this information, apply **Bayes' Theorem** to determine the likelihood that a patient with a stiff neck actually has meningitis.

Solution: Probability of Meningitis Given a Stiff Neck

Using Bayes' Theorem:

$$P(\text{Meningitis}|\text{Stiff Neck}) = \frac{P(\text{Stiff Neck}|\text{Meningitis})P(\text{Meningitis})}{P(\text{Stiff Neck})}$$

where:

- $P(\text{Meningitis}) = \frac{1}{30{,}000}$ (Probability of having meningitis)
- $P(\text{Stiff Neck}|\text{Meningitis}) = 0.80$ (Probability of a stiff neck given meningitis)
- $P(\text{Stiff Neck}) = 0.02$ (Probability of having a stiff neck)

Solution: Probability of Meningitis Given a Stiff Neck

Now, calculating:

$$P(\text{Meningitis}|\text{Stiff Neck}) = \frac{0.80 \times \frac{1}{30{,}000}}{0.02}$$

$$= \frac{0.00002667}{0.02} = 0.00133$$

Example

A single card is drawn from a standard deck of 52 playing cards. The probability of drawing a king is $\frac{4}{52}$. Determine the **posterior probability** $P(King|Face)$, which represents the probability that the drawn card is a **king**, given that it is a face card.

- **Total number of cards in a deck = 52**
- **Number of kings in the deck = 4 →** $P(King) = \frac{4}{52}$
- **Number of face cards (Jack, Queen, King) in the deck = 12**
  - (4 Jacks + 4 Queens + 4 Kings)
  - So, $P(Face) = \frac{12}{52}$
- **Probability of drawing a face card given that it's a king** is 1:
  - Since all **kings** are **face cards**, $P(Face|King) = 1$

Bayes' theorem states:

$$P(King|Face) = \frac{P(Face|King)P(King)}{P(Face)}$$

Substituting the values:

$$P(King|Face) = \frac{(1) \times \frac{4}{52}}{\frac{12}{52}}$$

$$P(King|Face) = \frac{4}{12}$$

$$P(King|Face) = \frac{1}{3}$$

Bayes' Theorem applies when we have **two dependent events**, and we want to compute the **posterior probability** given observed evidence:

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

- Bayes' Theorem **updates** our belief in event **A** given that B has occurred.
- Example: **Disease Diagnosis**
    - $P(\text{Disease}|\text{Fever}) = \frac{P(\text{Fever}|\text{Disease})P(\text{Disease})}{P(\text{Fever})}$
    - This tells us how likely a person has a disease if they have a fever.

However, **real-world problems** often involve **multiple variables**. This is where **Bayesian Networks** come in.

Complementary Probabilities

Two probabilities are **complementary** when they refer to the **same condition or event**, and one is the **opposite** of the other.

Example of Complementary Events:

- $P(Cloud \mid Rain) + P(\neg Cloud \mid Rain) = 1$
    - "Cloud **given** rain" and "**No** cloud **given** rain" are complementary.
- $P(Rain) + P(\neg Rain) = 1$

What is complementary?

Examples of actual complementary pairs:

- $P(\text{Cloud} \mid \text{Rain})$ and $P(\neg\text{Cloud} \mid \text{Rain})$
- $P(\text{Cloud} \mid \neg\text{Rain})$ and $P(\neg\text{Cloud} \mid \neg\text{Rain})$

In both cases, the condition is the same, and the outcomes are opposites (cloud vs. no cloud).

Complementary probabilities must:

- Refer to the **same condition**, and
- Be **opposite outcomes** (e.g., cloudy vs. not cloudy under the same condition)

These two probabilities:

- $P(\text{Cloud} \mid \text{Rain})$: Probability it's cloudy **given that it rains**
- $P(\text{Cloud} \mid \neg\text{Rain})$: Probability it's cloudy **given that it does not rain**

They refer to **different conditions** (rain and no rain), so they are **not** complementary.

Bayesian Network (Multiple Relationships)

A **Bayesian Network** extends **Bayes' Theorem** to model **complex dependencies** between multiple variables.

- Instead of a **single conditional probability**, we create a **graph (DAG - Directed Acyclic Graph)**.
- Each node represents a **random variable**.
- Directed edges represent **causal dependencies**.
- Each node has a **Conditional Probability Table (CPT)**.

Example: Medical Diagnosis

Instead of just "Fever" and "Disease", we consider **more factors**:

- **Disease** → causes **Fever** and **Cough**
- **Smoking** → increases risk of **Lung Cancer**
- **Lung Cancer** → increases risk of **Cough**

Bayesian Belief Network (BBN) in AI

A **Bayesian Belief Network (BBN)**, also called a **Bayesian Network (BN)**, is a **probabilistic graphical model** used in **Artificial Intelligence (AI)** to represent and reason about **uncertainty**.

A **Bayesian Belief Network** is a **directed acyclic graph (DAG)** where:

- **Nodes** represent **random variables** (events, observations, or states).
- **Edges** represent **conditional dependencies** between variables.
- Each **node** has a **probability distribution** that quantifies how it depends on its parent nodes.

Why Use Bayesian Networks in AI?

- AI applications often deal with **uncertain data** (e.g., medical diagnosis, speech recognition, robotics).
- Bayesian Networks provide a **structured way** to represent relationships between uncertain events.
- They allow AI systems to **update probabilities** when new evidence is observed (using **Bayes' Theorem**).

Components of a Bayesian Network

- **Nodes (Random Variables)**
  - Each node represents a **variable** (e.g., "Rain", "Traffic", "Headache").
  - Variables can be **observable** (evidence) or **hidden** (latent).
- **Directed Edges (Dependencies)**
  - A **directed edge** from node A → B means A **influences** B.
  - Example: "Rain" → "Traffic" (If it rains, traffic is likely).
- **Conditional Probability Tables (CPTs)**
  - Each node has a **Conditional Probability Table (CPT)**.
  - CPT specifies the **probability of a node** given its parent nodes.

How Does a Bayesian Network Work?

A **Bayesian Network** allows AI to **infer probabilities** by applying **Bayes' Theorem**:

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

- AI updates beliefs when **new evidence is observed.**
- Example: If we observe "Traffic is High", we can **update** the probability of "Rain".

Applications of Bayesian Belief Networks in AI

- **Medical Diagnosis** → Predict diseases based on symptoms.
- **Speech Recognition** → Probabilistic modeling of words in sentences.
- **Robotics** → Navigation and sensor fusion.
- **Spam Filtering** → Classifying emails as spam or non-spam.
- **Autonomous Systems** → Decision-making under uncertainty.

Example Bayesian Network: Disease Diagnosis

Question: Diagnosing a Disease Using a Bayesian Belief Network (BBN)

- A doctor wants to determine the probability that a patient has **Tuberculosis (T)** given that they have tested **Positive (P)** in a medical test. The Bayesian Belief Network consists of three nodes:
- **Tuberculosis (T)** → Probability of having tuberculosis.
- **Test Result (P)** → Probability of testing positive depends on whether the patient has tuberculosis.
- **Cough (C)** → Probability of coughing depends on whether the patient has tuberculosis.

Given Data

- **Prior probability of Tuberculosis:** $P(T) = 0.01$ (1% of patients have tuberculosis).
- **Test Accuracy:**
  - If a patient has tuberculosis, the test correctly detects it **95%** of the time: $P(P|T) = 0.95$.
  - If a patient does not have tuberculosis, the test gives a **false positive 5%** of the time: $P(P|\neg T) = 0.05$.
- **Cough Probability:**
  - If a patient has tuberculosis, they cough **80%** of the time: $P(C|T) = 0.80$.
  - If a patient does not have tuberculosis, they cough **10%** of the time: $P(C|\neg T) = 0.10$.

1. The **Bayesian Network** can be represented as:

```
        Tuberculosis (T)
           /        \
    Test (P)      Cough (C)
```

- **T** influences **P (Test Result)** and **C (Cough)**.
- The probability of **testing positive** and **coughing** is **conditioned on tuberculosis**.

2. Apply Bayes' Theorem Using BBN

We need to compute:

$$P(T|P,C) = \frac{P(P,C|T)P(T)}{P(P,C)}$$

Step 2.1: Compute Joint Probability $P(P,C|T)$

$$P(P,C|T) = P(P|T) \times P(C|T)$$

$$= (0.95) \times (0.80) = 0.76$$

Step 2.2: Compute Joint Probability $P(P,C|\neg T)$

$$P(P,C|\neg T) = P(P|\neg T) \times P(C|\neg T)$$

$$= (0.05) \times (0.10) = 0.005$$

Step 2.3: Compute Total Probability $P(P,C)$

Using **the Law of Total Probability:**

$$P(P,C) = P(P,C|T)P(T) + P(P,C|\neg T)P(\neg T)$$

$$= (0.76 \times 0.01) + (0.005 \times 0.99)$$

$$= 0.0076 + 0.00495 = 0.01255$$

Step 2.4: Compute Posterior Probability $P(T|P,C)$

$$P(T|P,C) = \frac{P(P,C|T)P(T)}{P(P,C)}$$

$$= \frac{(0.76 \times 0.01)}{0.01255}$$

$$= \frac{0.0076}{0.01255} = 0.6056$$

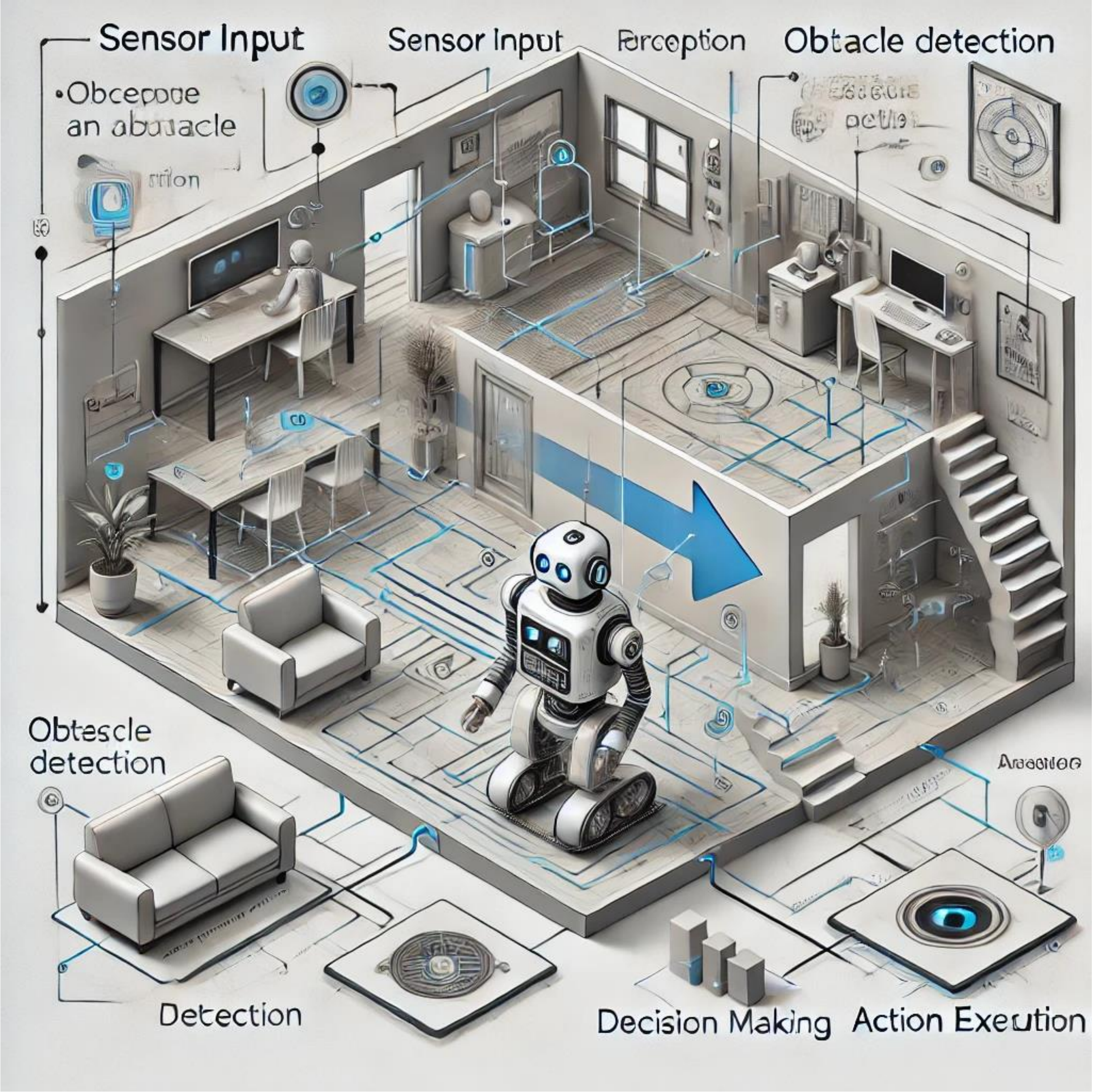
The probability that the patient **actually has tuberculosis** given that they **tested positive** and have a **cough** is **60.56%**.

Bayesian Belief Networks answered the question:

“What is likely true?”

In **planning**, the system goes a step further:

“Given what I believe to be true, what should I do next to achieve my goal?”


Sensor Input
Sensor Input
Obtacle detection
•Obcepoe
an abuacle
Obtescle
detection
Detection
Decision Making
Action Exeution

What is Perception in AI?

- Perception is how an AI system observes and interprets its environment.
- It converts sensor data into meaningful information.
- Helps the agent understand the current state of the world.
- **Example:** A self-driving car uses cameras and LIDAR to detect pedestrians, road signs, and traffic lights.

A self-driving car uses cameras and LIDAR to detect pedestrians, road signs, and traffic lights

What is Planning in AI?

- Planning is the process of deciding on actions to achieve a goal.
- It involves reasoning about future steps based on the current state.
- Helps agents select the best path among many possible options.
- **Example:** A vacuum cleaning robot plans the shortest route to clean the floor while avoiding furniture.

A vacuum cleaning robot plans the shortest route to clean the floor while avoiding furniture.

Relationship Between Perception and Planning

- Perception tells the agent “What is happening?”
- Planning tells the agent “What should I do next?”
- They work in a loop: perceive → plan → act → perceive again

BBN v/s Planning

- Bayesian Belief Networks handle uncertainty in knowledge.
- Planning builds on this knowledge to choose actions.
- We move from “What is likely true?” to “What should I do now?”

Example: If BBN says there’s a 70% chance of rain, planning system decides to take a covered path.

Introduction to Perception and Planning

- Perception and planning are fundamental components of artificial intelligence (AI) systems, particularly in the context of robotics and autonomous agents.

Key Components of a Planning Problem

- **Initial State** – The starting condition of the system.
- **Goal State** – The condition we want the system to reach.
- **States** – All possible situations the system can be in.
- **Actions (Operators)** – Transitions that move the agent from one state to another.
- **Plan** – A sequence of actions leading from the initial state to the goal.
- **Example:** Initial: Block A is on the table, Goal: Block A is on Block B.

Key components of planning in AI include:

- **State Representation**: Planning algorithms typically operate on a formal representation of the environment's state, which includes information about the current situation, available actions, and possible outcomes of actions.
- **Action Selection**: Given the current state of the environment, the planning algorithm selects a sequence of actions that will lead to the desired goal. This involves evaluating different action sequences and selecting the one that maximizes a predefined objective or utility function.
- **Search Algorithms**: Many planning problems can be formulated as search problems, where the goal is to find a sequence of actions that leads from the initial state to a goal state. Search algorithms such as depth-first search, breadth-first search, A* search, and Monte Carlo tree search are commonly used to explore the space of possible action sequences efficiently.
- **Decision Making under Uncertainty**: In real-world environments, there is often uncertainty about the outcomes of actions or the state of the environment. Planning algorithms need to account for uncertainty and make decisions that are robust to uncertainties in the environment.

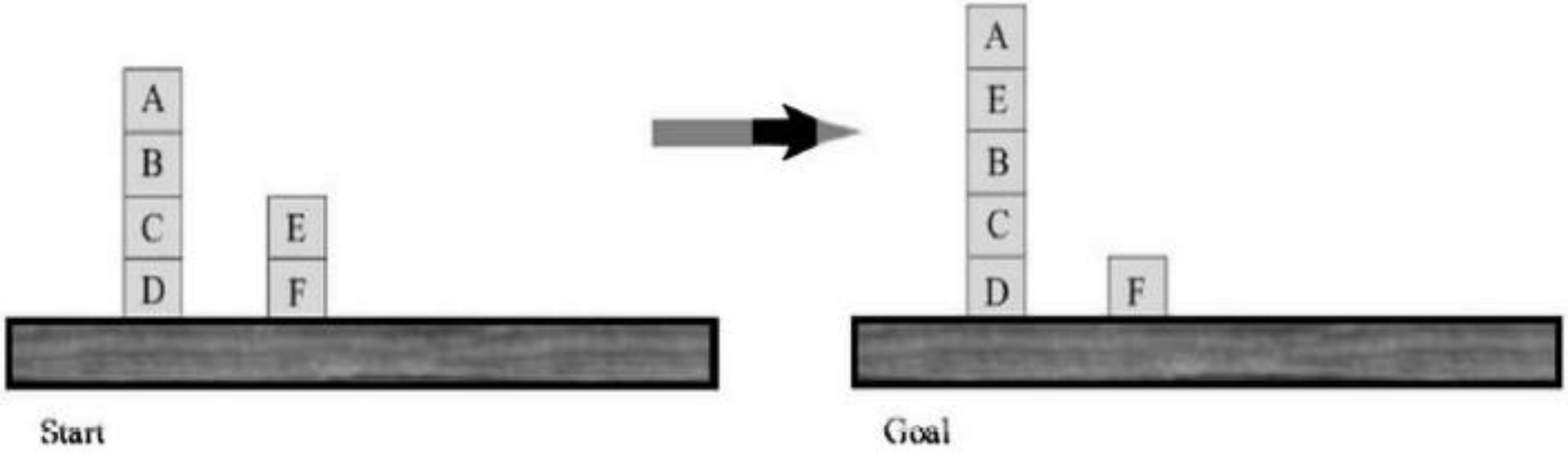
State Space Representation

- The state space is a graph where:
  - Nodes = States
  - Edges = Actions
- A planning algorithm searches this space to find a path to the goal.
- Search can be forward (from start) or backward (from goal).
- Algorithms like DFS, BFS, or A* are used to find valid action sequences.

STRIPS – A Classic Planning Language

- STRIPS is a structured way to define actions. It tells the planner what an action needs (preconditions) and what it results in (effects).
- STRIPS = Stanford Research Institute Problem Solver
  - Describes planning problems using:
  - Preconditions (what must be true to apply an action)
- Effects (what changes after the action)
- Used in early AI planning systems.

- A blocks world problem



Example: The blocks world

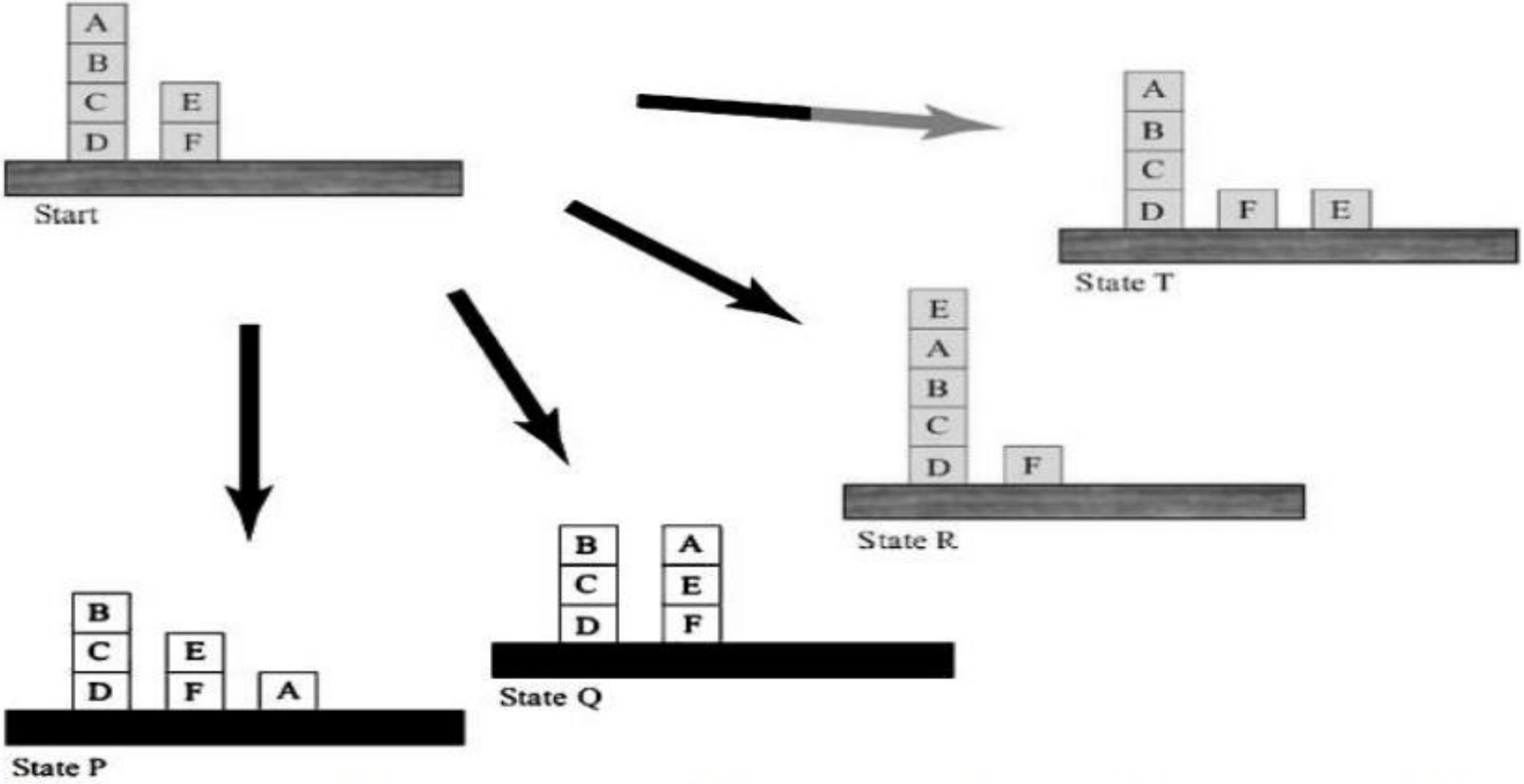
- **Blocks World Overview**:
  - Involves stacking cube-shaped blocks on a table.
  - A robot arm can pick up and move blocks, one at a time, to either the table or on top of other blocks.
  - Blocks cannot be moved if another block is stacked on top of them.
- **Key Predicate**:
  - **On(b, x)**: Block b is on x (where x is either another block or the table).

- **Action for Moving Blocks**:
  - **Move(b, x, y)**: Moves block b from x to y.
  - Preconditions: Block b and y must both be clear (nothing on them).
  - Effects: After the move, b is on y, x becomes clear, and y is no longer clear.
- **STRIPS Representation**:
  - **Move Action**:
    - Preconditions: `On(b, x) ∧ Clear(b) ∧ Clear(y)`.
    - Effects: `On(b, y) ∧ Clear(x) ∧ ¬On(b, x) ∧ ¬Clear(y)`.

- **Special Case for Table**:
    - The **Move action** does not correctly handle the case when x or y is the table.
    - Solution:
        1. Introduce a new action **MoveToTable(b, x)** for moving a block to the table.
        2. Interpret **Clear(Table)** as always true.

- **Additional Considerations**:
  - The planner might still use the **Move(b, x, Table)** action unnecessarily, leading to a larger search space, but this doesn't cause incorrect answers.
  - **Spurious actions** like **Move(B, C, C)** (moving a block onto itself) should ideally be handled by adding **inequality preconditions** to prevent no-op or contradictory effects.

The **blocks world** is a classic domain used to illustrate key concepts in planning, such as action preconditions, effects, and handling edge cases (like moving blocks to/from the table).



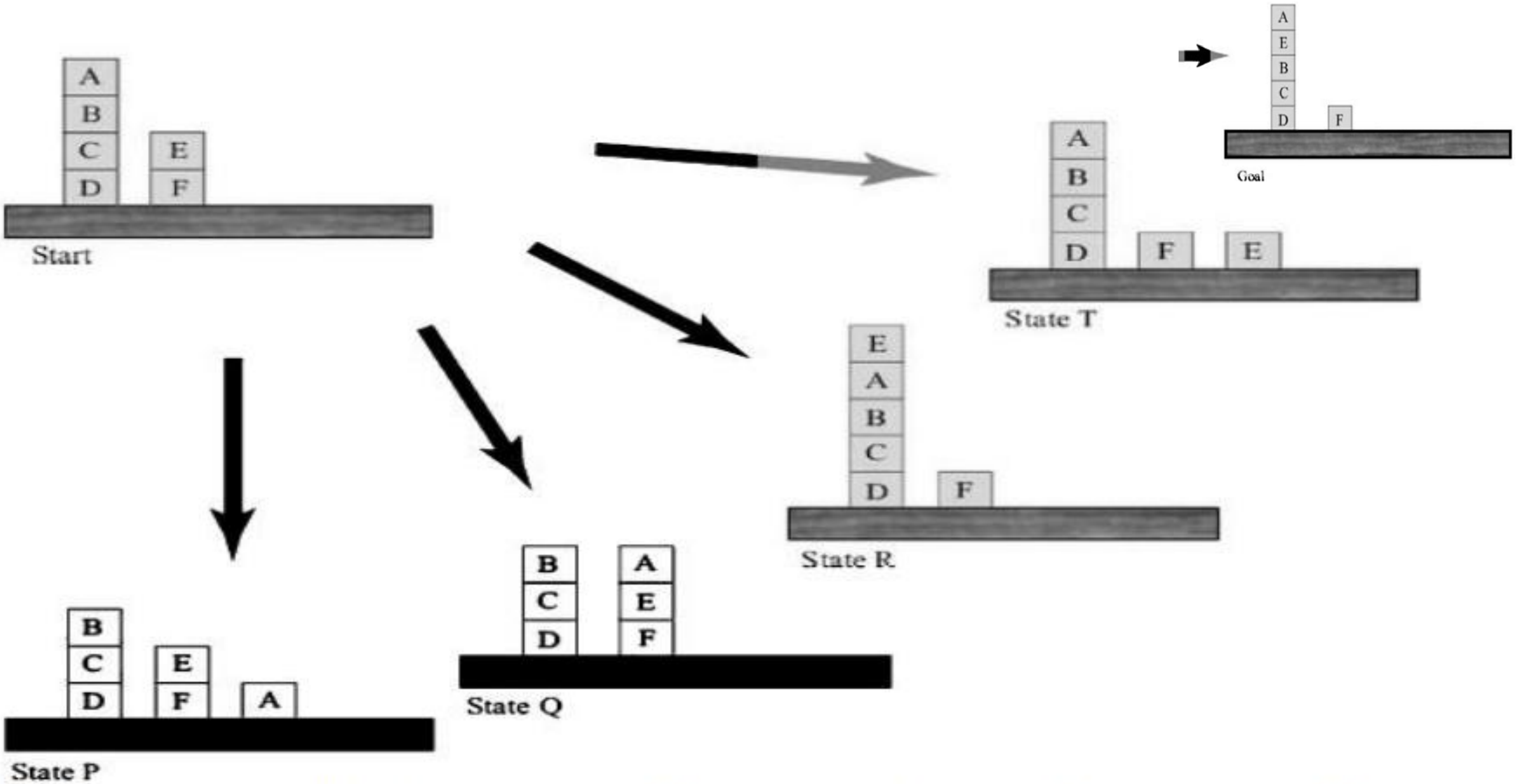
The first move possible. A move consist of moving one topmost block to another place

- $h_1(n)$ simply checks whether each block is on the correct block, with respect to final configuration. Add one for every block that is on the block it is supposed to be on, and subtract one for every one that is on a wrong one.

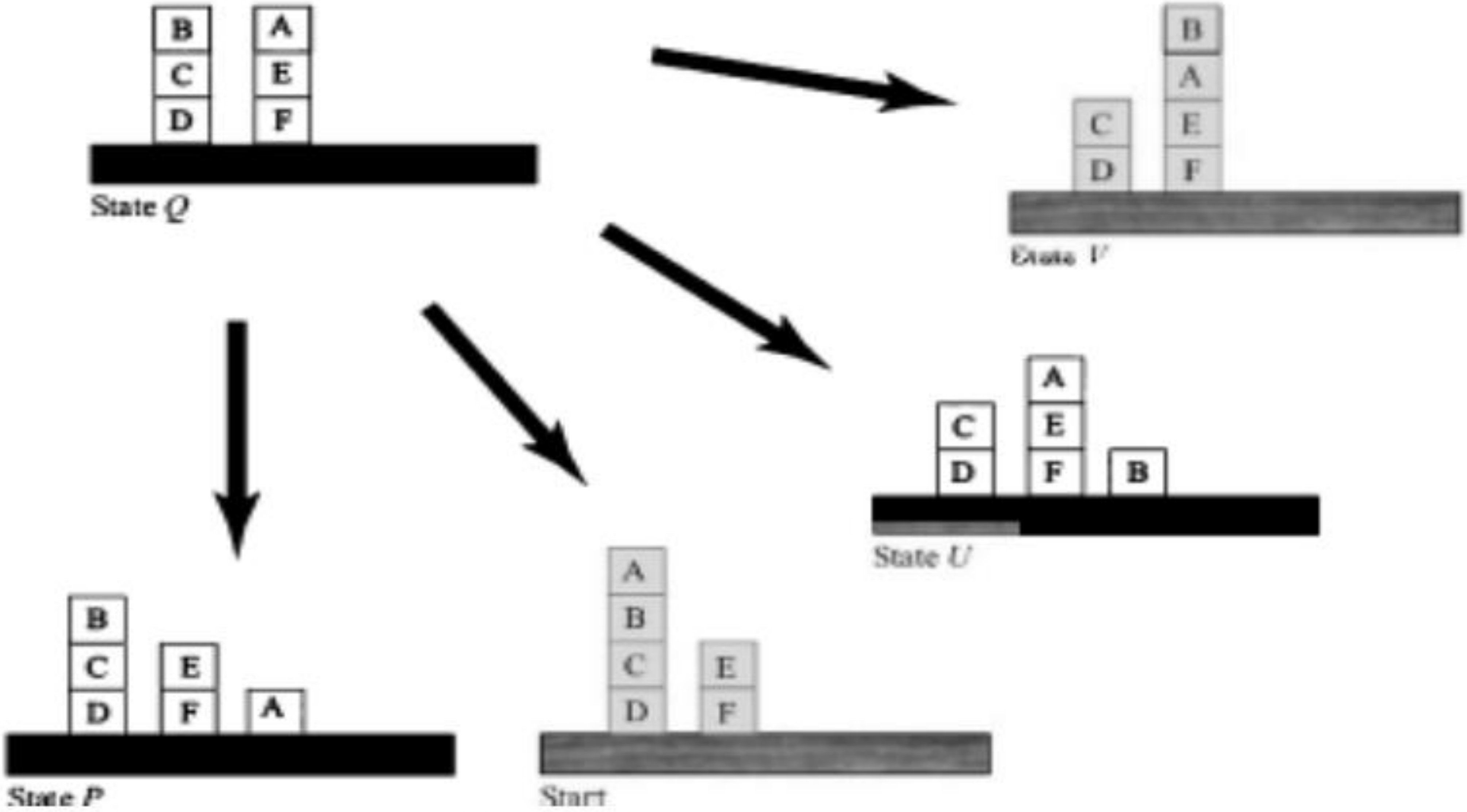
$$h_1(S) = (-1) + 1 + 1 + 1 + (-1) + 1 = 2$$
$$h_1(P) = (-1) + 1 + 1 + 1 + (-1) + 1 = 2$$
$$h_1(Q) = 1 + 1 + 1 + 1 + (-1) + 1 = 4$$
$$h_1(R) = (-1) + 1 + 1 + 1 + (-1) + 1 = 2$$
$$h_1(T) = (-1) + 1 + 1 + 1 + (-1) + 1 = 2$$

where,

$$h_1(n) = val_A + val_B + val_C + val_D + val_E + val_F$$



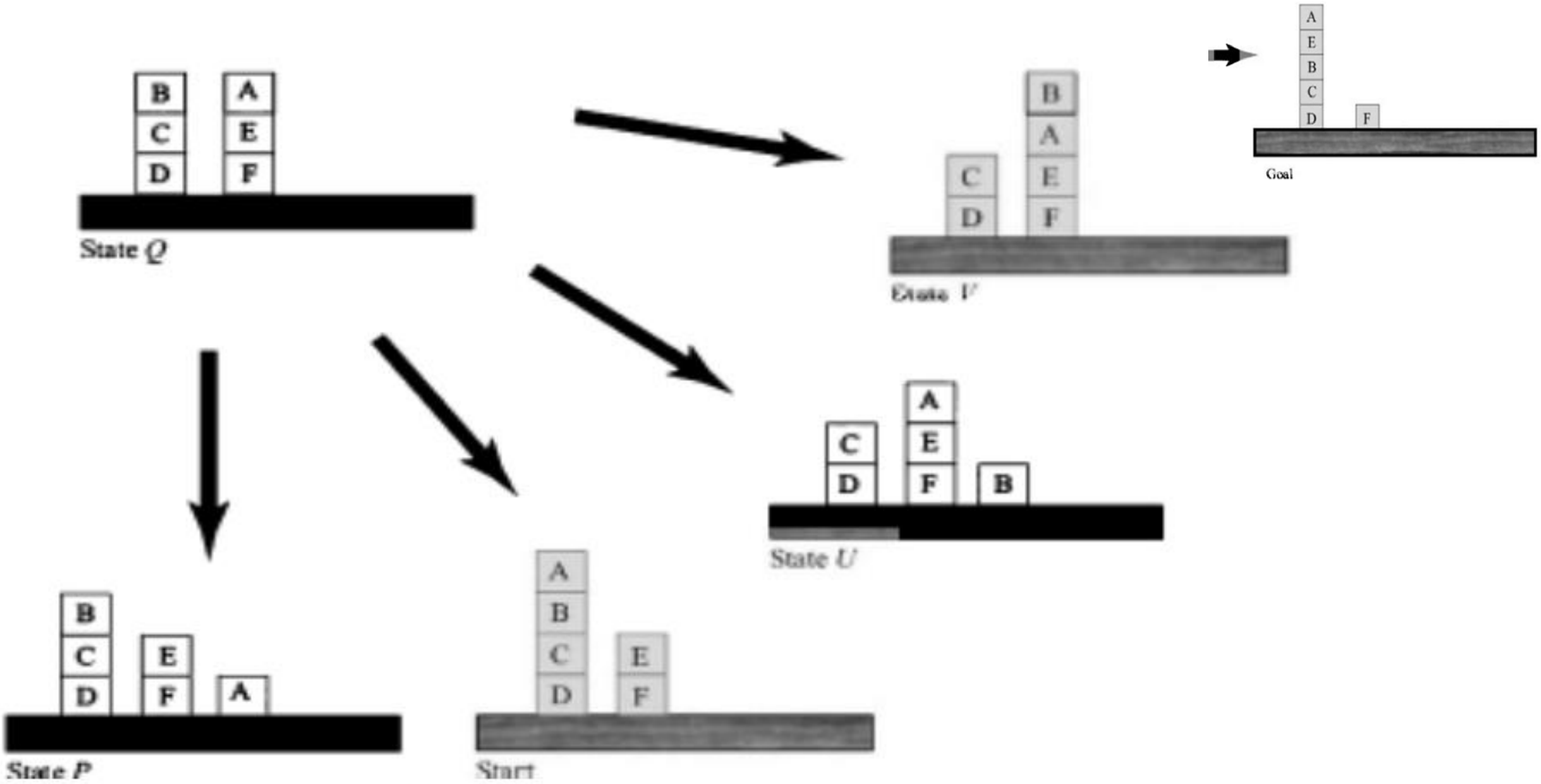
The first move possible. A move consist of moving one topmost block to another place



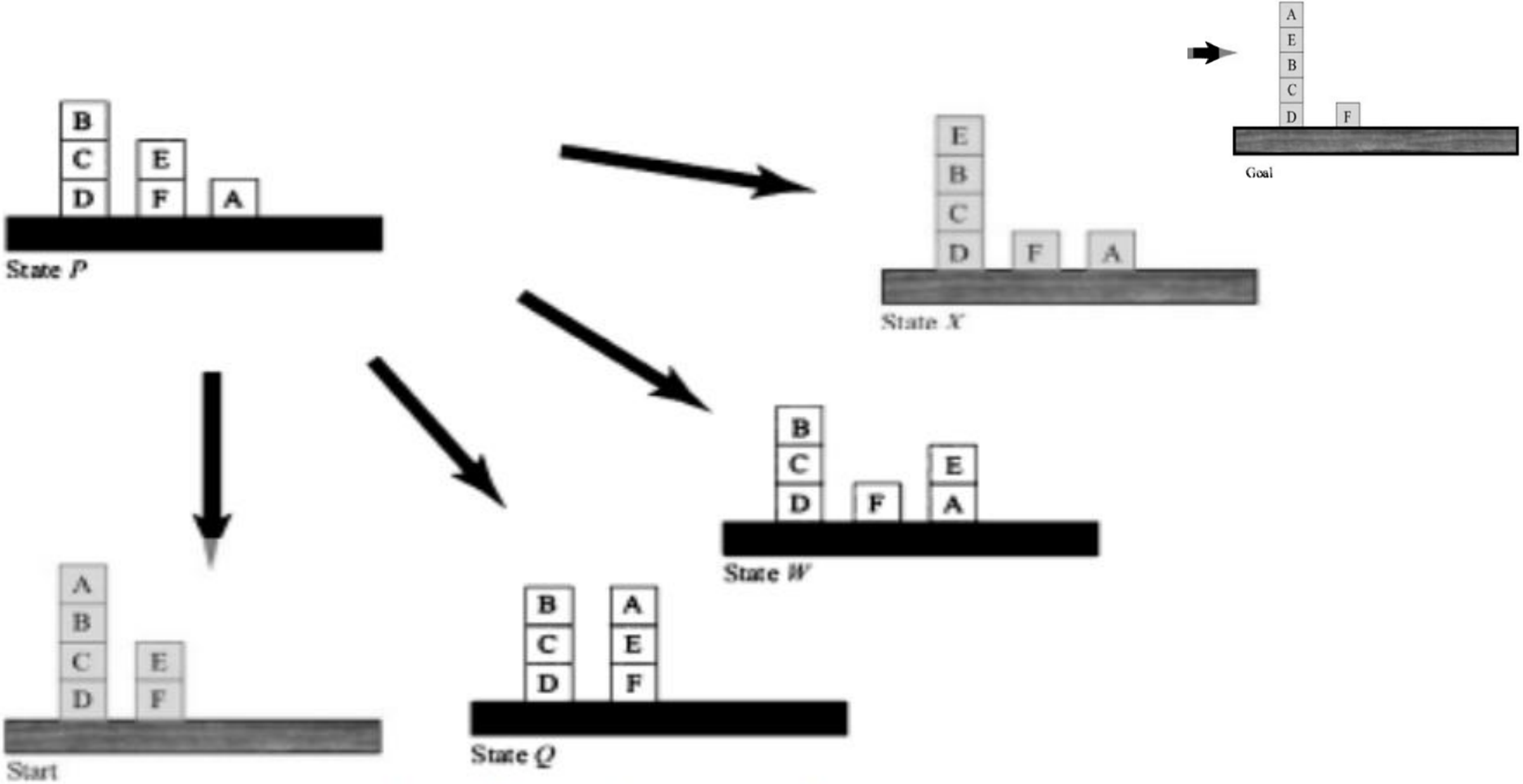
The choices from state Q

- $h_2(n)$ looks at entire pile that the block is resting on. If the configuration of the pile is correct, with respect to the goal, it adds one for every block in the pile, or else it subtracts one for every block in that pile.

$h_2(S) = (-3) + 2 + 1 + 0 + (-1) + 0 = -1$
$h_2(G) = 4 + 2 + 1 + 0 + 3 + 0 = 10$
$h_2(P) = 0 + 2 + 1 + 0 + (-1) + 0 = 2$
$h_2(Q) = (-2) + 2 + 1 + 0 + (-1) + 0 = 0$ $h_2(R) = (-3) + 2 + 1 + 0 +$
$(-4) + 0 = -4$
$h_2(T) = (-3) + 2 + 1 + 0 + 0 + 0 = 0$



The choices from state Q



The choices from state P

What Are Planning Algorithms?

- Planning algorithms search through state space to find a path to the goal.
- Different strategies guide how the search is performed.
- The main types are:
  - Forward Planning
  - Backward Planning
  - Heuristic Planning

Forward Planning (Progression Search)

- Starts from the **initial state**.
- Applies actions step by step to move **toward the goal**.
- At each step, it chooses applicable actions and generates new states.
- The agent begins from the known starting point and expands possible actions until the goal state is reached.
- **Example:** In a maze, the agent starts at the entrance and explores paths until it reaches the exit.
- **Pros:**
  - Easy to implement.
  - Works well when initial state is well-defined.
- **Cons:**
  - May explore irrelevant paths before finding the goal.

Backward Planning (Regression Search)

- Starts from the **goal state**.
- Works backward by identifying actions that could lead to the goal.
- Traces possible paths back to the **initial state**.
- Instead of asking “What can I do next?”, it asks “What would I have needed to do before reaching this goal?”
- **Example:** In a maze, the agent starts at the entrance and explores paths until it reaches the exit.
- **Pros:**
  - Easy to implement.
  - Works well when initial state is well-defined.
- **Cons:**
  - May explore irrelevant paths before finding the goal.

Heuristic Planning

- Uses a **heuristic function** to estimate the cost to reach the goal.
- Guides the search by focusing on the **most promising paths** first.
- Reduces unnecessary exploration.
- Heuristic planning introduces intelligence to the search by estimating which actions are likely to bring the agent closer to the goal.
- **Example:** In A* search, a function combines:
  - The cost so far (g)
  - Estimated cost to goal (h)
  - $f(n)=g(n)+h(n)$
- **Pros:**
  - More efficient than blind search.
  - Ideal for complex problems with large state spaces.
- **Cons:**
  - Requires a good heuristic function to be effective.

Comparison

Feature	Forward Planning	Backward Planning	Heuristic Planning
Search Direction	From start to goal	From goal to start	Guided by cost estimates
Focus	What can I do next?	What did I need to do?	What's the best next step?
Efficiency	Medium	Depends on goal	Often high
Uses Heuristics?	Optional	Optional	Yes

When to Use Each Strategy

- Use **forward planning** when the initial state is well-known.
- Use **backward planning** when the goal is specific and well-defined.
- Use **heuristic planning** when the problem space is large and needs efficient exploration.

PLANNING WITH STATE-SPACE SEARCH

Forward state-space search

Planning Algorithms - State-Space Search

- State-Space Search Overview:
  - Planning algorithms can search **forward** from the initial state or **backward** from the goal.
  - **Heuristics** can be automatically derived from action and goal representations to make the search more efficient.

Forward State-Space Search (Progression Planning):

- **Progression Planning**: Moves in the forward direction, starting from the initial state and exploring sequences of actions until a goal state is reached.
- **Formulation**:
  - The **initial state** is defined by a set of positive ground literals (literals not included are assumed false).
  - **Applicable actions**: Those whose preconditions are met in the current state.
  - **Successor state**: Formed by applying an action's effects, adding positive effect literals and removing negative ones.
  - **Goal test**: Determines if a state satisfies the goal conditions.
  - **Step cost**: Typically set to 1 for each action, though different costs could be used.

Example of Forward Search Inefficiency:

- **Air Cargo Problem**:
  - Goal: Move cargo from one airport to another.
  - Simple solution: Load cargo into a plane, fly the plane, and unload.
  - In practice: Large branching factor (~1000 actions per state) makes it difficult to find the solution without an accurate heuristic.

Example: Air cargo transport

- **Air Cargo Transport Problem**:
  - Involves three actions: **Load**, **Unload**, and **Fly**.
  - Two predicates define the state:
    - **In(c, p)**: Cargo **c** is inside plane **p**.
    - **At(x, a)**: Object **x** (either plane or cargo) is at airport **a**.
- **State Representation**:
  - When cargo is **In** a plane, it is not **At** any location.
  - **At** means "available for use at a specific location."

- **Example Plan (STRIPS Representation):**
    1. **Load(C1, P1, SFO)**: Load cargo C1 onto plane P1 at San Francisco (SFO).
    2. **Fly(P1, SFO, JFK)**: Fly plane P1 from SFO to JFK.
    3. **Unload(C1, P1, JFK)**: Unload cargo C1 at JFK.
    4. **Load(C2, P2, JFK)**: Load cargo C2 onto plane P2 at JFK.
    5. **Fly(P2, JFK, SFO)**: Fly plane P2 from JFK to SFO.
    6. **Unload(C2, P2, SFO)**: Unload cargo C2 at SFO.

Backward state-space search

Backward State-Space Search (Regression Planning)

- Overview:
  - Backward state-space search is also known as **regression planning**.
  - It works by starting from the goal state and finding actions that lead to this state, working backwards to determine the necessary preconditions.

- Process:
    - **Identify Relevant Actions**: For each conjunct in the goal, find actions that can achieve it. For example, to achieve `At(C1, B)`, identify actions like `Unload(C1, p, B)`.
    - **Regress the Goal**: Determine the states that, when an action is applied, result in the goal. This involves:
        - **Adding Preconditions**: For an action `Unload(C1, p, B)`, the predecessor state must include `In(C1, p)` and `At(p, B)`, in addition to all other conjuncts of the goal.
        - **Removing Achieved Goals**: The predecessor state should not include literals already achieved by the action.

- **Consistency**:
  - **Consistent Actions**: Ensure that actions not only help in achieving the goal but also do not undo any achieved goals. For instance, an action like `Load(C2, p)` is not consistent if it negates the goal `At(C2, B)`.
- **Constructing Predecessors**:
  - For a goal description `G` and a relevant, consistent action `A`:
    - **Delete Positive Effects**: Remove any positive effects of `A` that are already in `G`.
    - **Add Preconditions**: Include the preconditions of `A` in the predecessor description, ensuring they are not already present.

- **Search Algorithm**:
    - Use standard search algorithms to explore the backward search space.
    - **Termination**: Occurs when a predecessor state matches the initial state of the planning problem, possibly with variable substitutions. For example, the predecessor state might be satisfied with a substitution `{p/P12}`, resulting in the solution `[Unload(C1, P12, B)]`.

Example: The spare tire problem

- Spare Tire Problem Overview:
  - The goal is to have a **good spare tire mounted** on the car's axle.
  - The initial state: a **flat tire** on the axle and a **good spare tire** in the trunk.

- **Actions**:
  1. **RemoveSpare(Spare, Trunk)**: Remove the spare tire from the trunk.
  2. **RemoveFlat(Flat, Axle)**: Remove the flat tire from the axle.
  3. **PutOn(Spare, Axle)**: Mount the spare tire onto the axle.
  4. **LeaveOvernight**: If the car is left unattended overnight, the tires disappear.