

## Module 5: Planning and Learning

### What is planning

- Planning refers to the process of generating a sequence of actions that will achieve a specific goal or set of goals.
- Planning is a critical component of many AI applications, including robotics, autonomous vehicles, and game AI.

There are several types of planning that are commonly used in AI:

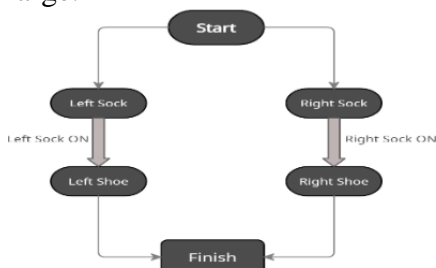
1. Forward planning: This involves starting from the current state and working towards the goal state by selecting appropriate actions.
2. Backward planning: This involves starting from the goal state and working backwards to determine the sequence of actions required to reach that goal state.
3. Hierarchical planning: This involves decomposing the overall planning problem into smaller sub-problems that can be solved independently.
4. Partial order planning: This involves generating a partial order of actions that need to be performed, without necessarily specifying the exact order in which they should be executed.

### Partial Order Planning :

Partial order planning is a type of planning algorithm that is particularly useful when the order of actions is not known beforehand, or when there are multiple valid orders that can achieve the same goal. The algorithm works by generating a partial order of actions, which is represented as a directed acyclic graph (DAG). Each node in the DAG represents an action, and the edges represent the causal relationships between actions.

To generate the partial order, the algorithm first identifies all possible actions that can be performed in the current state. It then generates a set of constraints that must be satisfied, such as preconditions and effects of the actions. These constraints are used to determine the causal relationships between the actions, and the DAG is constructed based on these relationships. Once the partial order has been generated, the algorithm can use various search techniques to find a valid sequence of actions that satisfies the constraints. This can involve searching through the DAG to identify all possible paths from the initial state to the goal state, and then selecting the path that satisfies the constraints.

Partial order planning is a powerful technique that can be used to generate plans for complex problems with uncertain or changing conditions. However, it can also be computationally expensive, particularly when there are many possible actions or when the problem space is very large.







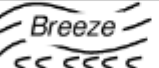
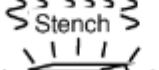



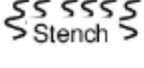





E.g. Partial order planning of Wearing Shoe

It combines two action sequence:

- i. First branch covers left sock and left shoe.
  - ii. In case, to wear a left shoe, wearing left sock is precondition, similarly.
  - iii. Second branch covers right sock and right shoe.
  - iv. Here, wearing a right sock is precondition for wearing the right shoe.
- Once these actions are taken we achieve our goal and reach the finish state.

### WUMPUS world environment giving its PEAS description

The Wumpus world is a classic artificial intelligence problem that involves finding a way out of a maze-like environment while avoiding dangers such as pits and the Wumpus monster. Here is a description of the Wumpus world environment in terms of its PEAS (Performance measure, Environment, Actuators, Sensors) attributes:

 Stench		 Breeze	 PIT
	 Breeze  Stench  Gold	 PIT	 Breeze
 Stench		 Breeze	
 START	 Breeze	 PIT	 Breeze

**Performance measure:** The goal of the agent is to safely navigate through the environment and exit the maze with the highest possible score. The agent receives a positive reward for finding the exit, a negative reward for falling into a pit or encountering the Wumpus, and a small negative reward for each step taken. The agent's performance is measured by the total accumulated reward.

**Environment:** The Wumpus world environment consists of a square grid of rooms, some of which contain pits or the Wumpus monster. The agent starts in a designated starting room and must navigate through the environment to reach the exit room while avoiding the dangers. The environment is partially observable, meaning that the agent cannot see the entire layout of the environment at once.

**Actuators:** The agent can take the following actions: move forward one square, turn left or right by 90 degrees, shoot an arrow in a straight line to try to kill the Wumpus (with a limited number of arrows), and grab the gold if it is present in the agent's current room.

**Sensors:** The agent has a set of sensors that allow it to perceive the environment, including: perceiving whether there is a breeze in the room (indicating a pit is nearby), perceiving whether there is a smell in the room (indicating the Wumpus is nearby), perceiving whether there is glitter in the room (indicating the gold is present), and perceiving the agent's current orientation.

The percept sequence in the Wumpus world environment is generated as follows:

1. At the start of the game, the agent receives a percept indicating its starting location and orientation, as well as whether there is a breeze, smell, or glitter in the starting room.
2. The agent selects an action to take based on its current percept, as well as any internal state or reasoning it may have about the environment.
3. The environment updates based on the agent's action, potentially changing the agent's location or causing other effects (such as the arrow killing the Wumpus or the agent picking up the gold).
4. The environment generates a new percept based on the updated state of the environment, including the agent's new location and orientation, as well as any new information about nearby hazards or the location of the gold.
5. Steps 2-4 are repeated until the agent either finds the exit, falls into a pit, encounters the Wumpus, or runs out of arrows.

Overall, the Wumpus world environment presents a challenging problem for artificial intelligence agents, as it requires the agent to reason about a partially observable environment with hidden dangers and rewards, and to plan a path through the environment that maximizes its chances of reaching the exit with the highest possible score.

PEAS Description of the Wumpus World Environment:

**Performance Measure:**

1. +1000 for picking up the gold,
2. -1000 for falling into a pit,
3. -1 for each action taken,
4. -10 for using up the arrow.

**Environment:**

1. 4x4 Grid of rooms.
2. Initially agent is in room labelled [1, 1] facing to the right.
3. Locations of PITs, Wumpus and gold are chosen randomly.
4. Each square other than the start can be a pit with probability 0.2.

**Actuators:** Agent can perform five different actions.

1. Move forward/ backward.
2. Turn left 90 degrees.
3. Turn right 90 degrees.
4. Grab gold. 5
5. Shoot Wumpus

**Sensors:** Agent has five different sensors.

1. Breeze Sensor (Room adjacent to pit)
2. Stench Sensor (Room adjacent to wumpus)



3. Glitter Sensor (Room with gold)
4. Bump Sensor (Collide with wall)
5. Scream Sensor (Wumpus dead)

## Supervised learning and unsupervised Learning

Supervised learning and unsupervised learning are two main categories of machine learning. Here's a pointwise explanation of both:

### Supervised Learning:

1. Supervised learning is a type of machine learning in which the model learns to map input data to output data based on labeled training data.
2. The labeled training data consists of input-output pairs, where the input data is used to train the model to predict the corresponding output.
3. The model learns from the labeled data and tries to generalize the pattern it has learned to make predictions on new, unseen data.
4. Examples of supervised learning include classification tasks, such as predicting whether an email is spam or not, or regression tasks, such as predicting the price of a house based on its features.
5. Supervised learning algorithms include decision trees, support vector machines (SVMs), and neural networks.

### Unsupervised Learning:

1. Unsupervised learning is a type of machine learning in which the model learns patterns and relationships in the input data without any labeled output data.
2. The model tries to identify hidden structure or relationships within the data, such as grouping similar data points together or discovering the underlying factors that explain the variation in the data.
3. Examples of unsupervised learning include clustering tasks, such as grouping similar customers together based on their purchasing behavior, or dimensionality reduction tasks, such as identifying the most important features in a dataset.
4. Unsupervised learning algorithms include k-means clustering, principal component analysis (PCA), and autoencoders.

To summarize, supervised learning is used when we have labeled training data and want to predict a corresponding output, while unsupervised learning is used when we don't have labelled output data and want to identify patterns or relationships within the input data

## Reinforcement Learning

Reinforcement learning is a type of machine learning in which an agent learns to make decisions by interacting with an environment. Here's a pointwise explanation of reinforcement learning:

1. Reinforcement learning involves an agent, an environment, and a set of actions that the agent can take within the environment.
2. The agent's goal is to learn a policy, which is a mapping from states to actions, that maximizes some notion of cumulative reward over time.



3. The environment is characterized by a set of states and a set of rewards that the agent can receive for taking certain actions in certain states.
4. At each time step, the agent observes the current state of the environment, selects an action to take, and receives a reward based on the action taken and the resulting state transition.
5. The agent's objective is to maximize the cumulative reward it receives over a sequence of interactions with the environment.
6. Reinforcement learning algorithms use the observed state, action, and reward information to update the agent's policy, which in turn affects the actions the agent takes in the future.
7. Reinforcement learning can be used in a variety of applications, such as game playing, robotics, and recommendation systems.
8. Reinforcement learning algorithms include Q-learning, SARSA, and actor-critic methods.

To summarize, reinforcement learning involves an agent learning to make decisions in an environment by maximizing cumulative reward over time through a trial-and-error process of exploration and exploitation. The agent uses the observed state, action, and reward information to update its policy, and the goal is to learn a policy that maximizes cumulative reward over time.