**Policy**

In reinforcement learning, a policy is a mapping from states to actions. It tells the agent what action to take when it is in a given state. The goal of RL is to learn an optimal policy that maximizes the agent's long-term rewards, such as achieving the highest score in a game.

**Action Space**

The action space in reinforcement learning refers to the complete set of all possible actions an agent can take in any given state. These actions define how the agent interacts with the environment. For example, moving left, right, or doing nothing.

The action space is crucial for designing AI agents because:

1. Defines Agent's Capabilities: It directly defines what the agent is capable of doing within the environment, which is fundamental to how it will achieve its goals. For instance, in a self-driving car, actions might include accelerating, braking, or steering.

**Markov Decision Process (MDP)**

A Markov Decision Process (MDP) is a mathematical framework used in reinforcement learning to model decision-making problems. It is defined by:

• States: The situations the agent can be in.

• Actions: The choices the agent can take.

• Transition probabilities: The likelihood of moving from one state to another after taking an action.

• Rewards: The feedback the agent receives for taking an action in a state.

• Policy: The strategy that defines which action to take in each state.

An MDP can be used to model a car's cruise control system.

States: The car's current speed compared to the target speed.

Actions: Accelerate, decelerate, or maintain speed.

Transitions: Accelerating increases speed, but may overshoot the target; decelerating reduces speed, but might slow down too much.

Rewards: +10 for staying close to the target speed, -5 for deviating too far, -1 for unnecessary fuel usage.

**Comparison of RL and SL**

Reinforcement Learning (RL):

• Training Data Structure: In RL, the agent does not receive labeled input–output pairs. Instead, it interacts with an environment, observing states, taking actions, and receiving delayed feedback in the form of rewards. The data comes from the agent's own trial-and-error experiences.

• Learning Objective: The goal is to learn a policy (mapping from states to actions) that maximizes cumulative future rewards over time. Learning is driven by balancing exploration (trying new actions) and exploitation (using known good actions).

Supervised Learning (SL):

• Training Data Structure: In SL, the agent is given a fixed dataset of input–output pairs (e.g., images with labels). Each input has a correct answer provided by a human or another system.

• Learning Objective: The goal is to learn a function that maps inputs to outputs as accurately as possible by minimizing a predefined error or loss function on the training data.

**Robotics RL**

Teaching a humanoid robot to walk or balance. The robot learns to coordinate its legs and maintain stability while moving forward or navigating obstacles.

Key Components of the RL Setup:

States: Robot's joint angles, velocities, and orientation, plus information about the environment (e.g., slope or obstacles).

Actions: Motor commands controlling the robot's joints, such as bending knees, rotating hips, or shifting weight.

Rewards: Positive reward for moving forward without falling, penalties for losing balance or inefficient movements.

Policy: The strategy that decides which motor actions to take in each state to maximize forward movement while staying balanced.

Environment: The physical robot or a simulation where the robot interacts with the ground, obstacles, and gravity, providing feedback for learning.

**Case Study: AI approach**

A reinforcement learning (RL) approach can be used to build the energy usage optimiser. The smart home system acts as the RL agent, and the environment consists of the home, appliances, and residents' behaviour.

States: Represent the current conditions, such as room temperatures, lighting levels, appliance status, and time of day.

Actions: Adjust heating, lighting, or appliance usage, either automatically or by making recommendations.

Rewards: Provide positive feedback when energy consumption is reduced while comfort is maintained, and negative feedback if comfort is compromised or energy use is high.

**Knowledge-Based Planning Solution**

Knowledge Representation

• Encode rules about energy usage, comfort preferences, and device constraints in a knowledge base.

• Example: "If room temperature < 20°C and resident is home, turn on heating" or "Turn off lights if no motion detected for 10 minutes."

Perception / State Monitoring

• Continuously gather sensor data: room temperatures, occupancy, appliance status, time of day, and outdoor conditions.

Goal Definition

• Minimise energy consumption while maintaining comfort levels for residents.

• Define measurable targets, e.g., temperature range, lighting levels, appliance usage thresholds.

Action Execution

• Send commands to appliances, heating, or lighting systems according to the planned actions.

Monitoring

• Observe the effects of actions on energy consumption and resident comfort.

• Update the knowledge base or rules as needed if residents' preferences or environmental conditions change.

Advantages & Disadvantages

1. Rule-Based System Developed by Experts

Advantage: Quick to implement – experts can define rules based on known routines and comfort preferences (e.g., "turn off lights at 11 PM"). This allows the system to operate immediately without needing extensive data.

Disadvantage: Inflexible – the system cannot easily adapt to changes in residents' habits or unexpected scenarios. If routines change, the predefined rules may no longer optimise energy usage effectively.

2. Reinforcement-Learning System That Adapts Over Time

Advantage: Adaptive – the RL agent can learn from residents' behaviour and feedback, gradually improving recommendations and actions to minimise energy use while maintaining comfort, even as routines change.

Disadvantage: Requires time and data – initially, the system may make suboptimal decisions as it explores actions to learn the optimal policy, potentially causing temporary discomfort or higher energy use.

**Bio-inspired computing**

Bio-inspired computing refers to the creation of algorithms and computational methods by simulating natural biological processes and systems. It is inspired by biological phenomena such as evolution, heredity, neural activity, and self-regulation, just like a living organism.

• Problem Understanding and Exploration: Traditional computing uses clear rules and exact steps designed by humans to solve problems.

But, bio-inspired computing, like genetic algorithms, does not rely on fixed instructions. Instead, it uses trial-and-error and probabilistic search to explore many possible solutions, sometimes finding creative results humans might not think of.

Fitness Function

The term 'fitness function' in genetic algorithms (GAs) refers to a mathematical function to measure how performance or "fitness" each solution is within a population. It gives a score that shows how well a candidate solution meets the goal or solves the problem at hand.

Importance in the Context of Evolving Artificial Creatures:

Selection Pressure: The fitness score directly determines an individual's "selection probability". Creatures with higher scores are more likely to be chosen as parents for the next generation. This is like natural selection, where fitter individuals have a better chance of survival and reproduction. Therefore, traits that improve performance are more likely to be passed down, gradually producing better and more capable artificial creatures.

**Selection, Crossover, Mutation**

1. Selection:

◦ Role: Selection is the process of choosing individuals from the population to act as parents for the next generation, favouring fitter solutions. It provides selection pressure so that better solutions are more likely to survive and reproduce, driving evolutionary improvement.

2. Crossover:

◦ Role: Crossover combines genetic information from two parent solutions to form new offspring. It mimics biological reproduction by exchanging segments of genetic code. By doing so, it introduces new combinations of traits, helping the algorithm search more widely and discover potentially better solutions than those of the parents.

3. Mutation:

◦ Role: Mutation makes small, random alterations to an individual's genome. Mutation is crucial for maintaining genetic diversity within the population and preventing the algorithm from getting stuck in local maxima. Random changes allow the search to escape local optima and investigate unexplored areas of the solution space.

**Real-world example**

A real-world example is using a genetic algorithm to help a humanoid robot (such as those developed by companies like Tesla or Boston Dynamics) learn how to walk or balance. Different walking patterns are represented as genomes, and the robot tests these movements in simulation. A fitness function evaluates performance, such as distance walked without falling or energy efficiency. Through selection, the best patterns are chosen, then improved using crossover and mutation. Over many generations, the humanoid robot evolves stable and efficient walking behaviours, without engineers having to program every step manually.

How a genetic algorithm would be applied to a humanoid robot learning to walk:

Start with random solutions → Generate different walking patterns (genomes) for the robot.

Test performance → Simulate each pattern on the humanoid robot and measure how well it walks (e.g., balance, distance, energy use).

Evaluate fitness → Assign a fitness score based on performance, with smoother, longer, or more efficient walking rated higher.

Select parents → Choose the best-performing patterns to act as parents for the next generation.

Crossover and mutation → Combine parent patterns and add small random changes to create new walking strategies.

Repeat the process → Over many generations, the robot evolves better and more stable walking behaviour.

**Diversity**

Importance of Diversity in a Genetic Population:

Exploration of Solutions: Diversity ensures that the genetic algorithm explores a wide range of possible solutions at the same time. A varied population allows the algorithm to search different regions of the solution space rather than focusing too narrowly, increasing the chance of finding better or unexpected solutions.

Avoiding Premature Convergence: A diverse population reduces the risk of the algorithm converging too quickly on a poor solution. If all individuals are too similar early on, the search may stagnate and fail to improve, missing out on the true global optimum.

What Might Happen if Diversity is Lost:

The algorithm may converge too early on a local optimum, where all solutions are very similar and no further improvements can be made. With little genetic variation left, crossover and mutation produce only small changes, leaving the population "stuck" and unable to escape to better regions of the search space.

**Role of Mutation**

In evolutionary algorithms, mutation adds random changes to individuals in the population. Its main role is to keep variety in the population and stop the algorithm from getting stuck at a weak solution (local optimum). By changing some genes randomly, mutation creates new possibilities that crossover alone cannot provide. This helps the population keep improving instead of staying the same.

Contribution to the Search Process:

Mutation helps the search by adding new genetic material, which expands the range of solutions the algorithm can explore. It allows the population to escape from local optima by making changes that open paths to better solutions. Mutation also maintains variety in the population, ensuring the algorithm does not stagnate. By continuously introducing fresh traits, it supports both exploration of new areas and steady progress toward stronger solutions.

**High Level how AI approach builds a system for warehouse**

Genetic algorithms can be used to optimize warehouse layout. Each product placement scheme can be considered as a potential solution. By defining a fitness function, the impact of different layout schemes on picking time can be evaluated; for example, placing

products that are frequently ordered together in close proximity can significantly reduce picking time. The algorithm uses operations such as selection, crossover, and mutation to continuously generate new layout schemes, gradually optimizing the layout through multiple iterations. By continuously updating order data, the system can adapt to changes and maintain efficient warehouse operations.

**Steps**

Initialization

◦ Generate an initial population of random warehouse layouts.

◦ Each layout assigns products to specific storage locations.

Fitness Evaluation

For each layout, calculate a fitness score:

◦ Measure average picking time (lower = better).

◦ Layouts with frequently co-ordered items stored closer get higher scores.

Selection

◦ Choose the best layouts as parents based on their fitness scores.

Crossover

◦ Combine parts of two parent layouts to create new layouts (as offspring).

Mutation

◦ Randomly swap or move a small number of products in a layout to introduce variation.

Replacement

◦ Form a new population by keeping the best layouts and replacing weaker ones with new offspring.

Iteration

◦ Repeat steps 2–6 for many generations until improvement slows down or a stopping condition is

**Advantages & Disadvantages**

1. Manually reconfiguring based on expert knowledge

Advantage: Warehouse managers can use their knowledge of product demand to quickly rearrange certain items, for example, moving best-selling or seasonal products closer to the picking area without waiting for a system to run.

Disadvantage: Human decisions might miss hidden patterns in thousands of co-ordered products, so the new layout could still result in longer picking times compared to an optimised one.

2. Re-running the genetic algorithm periodically with updated data

Advantage: The GA can process the full order history and automatically discover new item groupings (e.g., products often bought together) that humans might not notice, leading to more efficient layouts.

Disadvantage: The layouts suggested by the GA may change a lot between runs, which could be disruptive because staff would need to constantly adjust the warehouse setup, leading to extra labour and downtime.

MCQ

According to the article "King, R. D., Rowland, J., Oliver, S. G., Young, M., Aubrey, W., Byrne, E., & Clare, A. (2009). The automation of science. Science, 324(5923), 85-69.". What was the main activity of the Adam system?

You should see a PDF version of this article embedded with the question.

**A. AI Researcher (True)**

B. Generate Top 40 pop song (False)

C. Answer Computer Science exam essay questions (False)

D. Write a New York Times bestseller (False)

In the research paper "Grace, Katja, et al. "When will AI exceed human performance? Evidence from AI experts." Journal of Artificial Intelligence Research 62 (2018): 729-754., which of the following are milestones in the progress of AI?

You should see a PDF version of this article embedded with the question.

Select ALL that apply:

A. AI researcher (False)

**B. Generate Top 40 pop song (True)**

**C. Answer Computer Science exam essay questions (True)**

**D. Write a New York Times bestseller (True)**

**Which of the following is the teamed space for an auto-encoder?**

**A. Latent space (True)**

B. Variational encoding space (False)

C. Combinatorial space (False)

D. High dimensional space (False)

Which term refers to the general method used to generate text from GPT-2:

**A. Auto-regression (True)**

B. Statistical feature recognition with context embeddings (False)

C. Feed-forward (False)

D. Recurrent neural network architecture (False)

Genetic algorithms are: (Select ALL that apply)

A. A method for iteratively optimising solutions to a genetically-encoded problem **(True)**

B. An example of bio-inspired computing **(True)**

C. A method for estimating the computational characteristics of an animal's genome (False)

D. An old type of algorithm which is no longer studied in computer science (False)

E. An appropriate algorithm for automatically designing neural network architectures **(True)**

**Which of the following are true about the fitness function in the creatures case study seen in the AI course?**

**Select ALL that apply:**

A. The fitness function provides various processes which can be used to generate new genomes (False)

B. The fitness function assigns a score to each member of a population **(True)**

C. The fitness function uses a simulation to estimate the performance of the members of a population **(True)**

D. The fitness function selects which members of a population should be allowed to breed (False)