AI Questions & Answers

Marco Bernardi

January 3, 2024

1 Agents

A.1 Explain in detail the meaning of the acronym PEAS in the context of the definition of an intelligent agent. According to the previous answer, introduce the various types of agents discussed in class.

Answer:

PEAS = Performance, Environment, Actuators, Sensors

PEAS helps us to define the task environment in which the agent is situated.

- **Performance**: A measure of performance is set which evaluates the sequence of perceptions and a rational agent has to maximize the expected value of the performance measure up to the current time. Performance measures could be a combination of different criteria. (Ex. Automated taxi driver: safety, legality, comfort, profit, time, etc.)
- **Environment**: It describes the task environment in which the agent is situated. An environment could be fully **observable** (Chess game) because you can see the whole state of the environment, otherwise it will be or **partially observable** (Poker).

  It could be **deterministic** (chess game) because the next state of the environment is completely determined by the current state and the action executed by the agent, or **stochastic** (poker) because the next state of the environment can't be predicted in a certain way.

It could be **episodic** if the current action depend on the previous/next actions or **sequential** if the action are independent.

It could have only one agent or multiple agents.

It could be **static** if the environment doesn't change while the agent is deliberating or **dynamic** if the environment can change while the agent is deliberating.

Depending on the environment variables, the environment could be **discrete** or **continuous**.

(Ex. Automated taxi driver: roads and highways, other cars, pedestrians, traffic lights, etc.)

- **Actuators**: They are the mechanism or devices through which the agent acts upon the environment. Actuators translate decisions made by the agent into actions that can be performed on the environment. (Ex. Automated taxi driver: steering wheel, accelerator, brake, signal, etc.)
- **Sensors**: They are the mechanism or devices through which the agent perceives the environment. (Ex. Automated taxi driver: camera, GPS, speedometer, odometer, etc.)

An agent is a perceiving and acting entity, it is considered rational if it tries to achieve the best outcome given the available information.

Exist different types of agents:

- **Simple reflex agents**: they select actions on the basis of the current percept, ignoring the rest of the percept history. A simple reflex agent acts according to a rule whose condition matches the current state, as defined by the percept. The disadvantage of this type of agent is that if the environment is partially observable, then the agent may select a wrong action.
- **Model-based reflex agents**: they select actions on the basis of the current percept and some of the percept history. The disadvantage of this type of agent is that it has hard-coded rules, so it can't learn from experience and it can't be flexible.
- **Goal-based agents**: they select actions based on the goal they are trying to achieve, making the agent more flexible. The disadvantage of this type of agent is that it can have several goals that may conflict with each other.

- **Utility-based agents**: they select actions based on a utility function that measures the performance of the agent. The utility function helps the agent to choose the goal keeping in mind the likelihood of success and the importance of the goals. No remarkable disadvantages.
- **Learning agents**: they select actions based on the knowledge gained from the environment and past experiences, and there is a problem generator component that creates new situations to improve the agent's knowledge. No remarkable disadvantages.

2 Uninformed Search

US.1 Describe the principal uninformed search strategies, and compare them in terms of correctness, completeness (Can we find a solution?), time and space complexity.

Answer: Search strategies are used to find a solution to a problem defined by four components: **initial state**, **successor function**, **goal test** and **path cost**. The solution is a sequence of actions that leads from the initial state to a goal state. There are different uninformed search strategies (We're working with trees):

(a) **Breadth-first search**: it explore the tree level by level, so the fringe is implemented as a FIFO queue. The disadvantage of this strategy is that it requires a lot of memory, because it has to visit all the nodes. It is **complete** (if branching factor b is finite) and **optimal** if the path costs are all equal. It has a **time complexity** of $O(b^d)$ and a **space complexity** of $O(b^d)$.

(b) **Uniform cost search**: if the path costs are all equal, it is equivalent to breadth-first search. Otherwise it expands the least-cost unexpanded node (the node with the lowest path cost) and it uses a priority queue (fringe) orderd by path costs. It is **complete** if the path costs are bounded below by a small positive constant and b is finite. It is **optimal**. It has a **time complexity** of $O(b^{1+\lfloor C^*/\epsilon \rfloor})$ (it can be greater than BFS time complexity because once it found a solution, it has to check if there is a better solution) and a **space complexity** of $O(b^{1+\lfloor C^*/\epsilon \rfloor})$.

(c) **Depth-first search**: it explores the tree by expanding the deepest node in the current frontier of the search tree. The fringe is implemented as a LIFO queue. It is **not complete** because it can get stuck in an infinite path. It is **not optimal**. It has a **time complexity** of $O(b^m)$ and a **space complexity** of $O(bm)$.

It requires less memory than breadth-first search, but it can get stuck in an infinite path.

(d) **Iterative deepening search**: it is a combination of depth-first search and breadth-first search. It performs depth-first search up to a certain depth, then it performs breadth-first search up to that depth, then it performs depth-first search up to the next depth, etc. It is **complete** if b is finite and **optimal** if the path costs are all equal. It has a **time complexity** of $O(b^d)$ and a **space complexity** of $O(bd)$.

It advantage is that it requires less memory than breadth-first search

(e) **Depth-limited search**: it is a depth-first search with a depth limit that doesn't change. It isn't **complete** and not **optimal**. It has a **time complexity** of $O(b^l)$ and a **space complexity** of $O(bl)$.

(f) **Bidirectional search**: it performs two simultaneous searches, one forward from the initial state and one backward from the goal. It is **complete** if b is finite and both searches use breadth-first search. It is **optimal** if the path costs are all equal and both searches use breadth-first search. It has a **time complexity** of $O(b^{d/2})$ and a **space complexity** of $O(b^{d/2})$.

May not find the optimal solution, but if it's applicable it is faster and lighter than breadth-first search.

3 Informed Search

IS.1 Define the concept of informed search, and describe the greedy search and the A* algorithm. Discuss the properties (optimality, completeness, time and space complexity) and conditions of applicability of the two algorithms.

Answer:

Informed search algorithms use problem-specific knowledge beyond the definition of the problem itself. Greedy search and A* algorithm are some special case of best-first search. Best-first search is a search algorithm that explores a graph by expanding the most promising node chosen according to a specified rule. The rule is specific to the problem, defined by a evaluation function *f(n)* that estimates the cost of the cheapest path from the node n to a goal node. Most of the best-first search algorithms include a heuristic function *h(n)* in *f(n)* that estimates the cost of the cheapest path from the node n to a goal node.

- **Greedy search**: it expands the node that appears to be closest to goal, according to the heuristic function *h(n)*. In this case *f(n)* = *h(n)*. It is **not complete** because it can get stuck in a loop. In a finite space with repeated-state checking it is complete. It is **not optimal**. It has a **time complexity** of $O(b^m)$ and a **space complexity** of $O(b^m)$.
- **A* algorithm**: The idea at the base of this algorithm is to avoid expanding paths that are already expensive. It expands the node that has the lowest value of *f(n)* = *g(n)* + *h(n)*. Where:
  - *g(n)* is the cost of the path from the initial state to the node n.
  - *h(n)* is the heuristic function that estimates the cost of the cheapest path from the node n to a goal node.
  - *f(n)* is the evaluation function that estimates the cost of the cheapest solution through n.

  A* uses an admisible heuristic function, that is a heuristic function that never overestimates the cost to reach the goal. It is **optimal** for tree search because it will never select a suboptimal goal G_2 over an optimal goal G_1, since $f(G_2) > f(n)$ where n is an unexpanded node that leads to G_1. For graph search the previous statement is not true, and we need another proof that use consistent heuristic. A heuristic is consistent if $h(n) \leq c(n, a, n') + h(n')$ where n is the current node, a is the action that leads to n' and n' is the successor of n. $f(n)$ is non-decreasing along any path. It is **complete**, unless there are infinitely many nodes with *f(n)* $\leq$ *f(G)*. It has a **time complexity** exponential and a **space complexity** of $O(nodescount)$.

No other optimal algorithm is guaranteed to expand fewer nodes than A*.

A* could be exponential in space ⇒ Solutions:

- **Iterative deepening A* (IDA*)**: it acts like A* but it use a cutoff value instead of a depth limit. During an iteration if $f(n) > cutoff$ then the node is not expanded. When the queue is empty, the cutoff is increased to the lowest value of $f(n)$.
- **Recurisve Best-First Search**: imitates a deep search, using only linear space. It keeps track of the best alternative path available from any ancestor of the current node. When a node is expanded, the algorithm updates the value for the best alternative path. If the value of the best alternative path is smaller than the value of the current node, recursion goes back to the alternative path. On the return from the recursion, the algorithm updates the f-value of the best child node.

  It is **optimal** if the heuristic is consistent. Space complexity is $O(bd)$ and time complexity is exponential in the worst case. RBFS has a problem, it use too litle memory.
- **Memory-bounded A* (MA*)**:
- **Simplified memory-bounded A* (SMA*)**: it expand the best node until the memory is full. It removes the worst node from the memory to make space for a new node and backup the f-value of the removed node on the parent node. The parent node will eventually be expanded if the other paths are worse.

  It is **complete** only if solution can be kept in memory. It is **optimal** if any optimal solution is reachable otherwise it returns the best solution found.

IS.2 Introduce the A* algorithm in detail and exhaustively. Formally prove its optimality prop-erties. Finally, discuss the memory usage issues of A* and how this can be solved/reduced.

4 First Call 23-01-2023

A.1 Introduce hill-climbing search, appropriately placing it among the various categories of problem solving approaches we discussed in class. Present the variants, the properties, and in the case of a search with restart, formally demonstrate the result relative to the number of expected searches before finding an optimal solution.

Answer:

A.2 In the context of adversarial search, explain in detail how games with elements of chance can be deal with. In the case of resources limit, explain what is the property of the evaluation function that allows the search to preserve optimal decision (i.e. obtained with no resources limit). Finally, explain why the same approach is not returning an optimal strategy in the case of partially observable games.

Answer:

A.3 In the context of inference in propositional logic, explain what is the role of horns form, discussing its strengths and weaknesses.

Answer:

A.4 In the context of inference in the first-order logic, introduce the resolution rule. Discuss in what form the clauses must be represented in order to apply effectively the resolution and how this form can be obtained.

Answer:

A.5 In the context of treatment of uncertainty, discuss what is the main computational challenge when using the joint probability distribution of the main involved variables.

Answer:

A.6 Give the formal definition for a constrain satisfaction system, and discuss the approaches presented in class on how to find solutions.

Answer:

5 First and Second part 2021/2022

B.1 In the context of informed search, present the approaches discussed in class to manage the case in which the path to reach a goal state is irrelevant. For each approach discuss conditions of applicability, strengths and weaknesses.

Answer:

B.2 Give the abstract definition of knowledge-based agent, discussing its various components. In addition, in the case of propositional logic, introduce how inference can be made by enumeration, discussing its algorithmic and computational complexity aspects.

Answer:

B.3 (Similar to A3 non va l'autolabel dc) In the context of inference in first-order logic, introduce the resolution rule. Discuss in what form the clauses need to be represented in order to efficiently enforce the resolution and how that form can be achieved. Finally, present the various strategies that have been proposed regarding the choice of clauses to be used during the inference.

Answer:

B.4 What does a Bayesian Network consist of? Why is it useful? What is its computational complexity in space and time (also discuss special cases) for the exact inference?

Answer:

B.5 Introduce the main paradigms of machine learning, describing in particular the funda-mental ingredients of the supervised paradigm, and how the complexity of an hypothesis space can be measured in a useful way in the case of a binary classification task.

Answer:

6 Example 2020/2021

C.1 In the context of adversarial search, discuss how huge search spaces, such as those gen-erated by the game of chess or Go, can be computationally managed. Also explain howthe algorithm $\alpha - \beta$ pruning can help in such cases.

Answer:

C.2 In the context of first order logic, describe in a complete and exhaustive way the Unifica-tion algorithm, further explaining why this is particularly useful for logical inference. Explain in which case the logic inference couldn't help a smart agent.

Answer:

7 Call 13-02-2019 (Translated from Italian exam)

D.1 Explain in detail the supervised learning paradigm, describe the role of the training set, the validation set and the test set (how to use data in our hands). Give the definition of ideal error and empirical error, highlighting the role of them during the learning process.

Answer:

D.2 In the field of NLP, explain the difference between the n-gram (in particular unigram and bigram) and bag-of-words models. Introduce the concept of TF-IDF, highlighting its main features, its advantages compared to the bag-of-words model. Describe a possible application of TF-IDF in the field of information retrieval.

Answer:

D.3 Si descriva il modello della pinhole camera ideale, il processo di proiezione prospettica e le equazioni che regolano la formazione dell'immagine a partire da punti nel 3D. Si forniscano le motivazioni (e le principali differenze) che definiscono un modello basato su lenti ottiche.

Describe the model of ideal *pinhole camera*, the process of perspective projection and the equations that regulate the creation of the image starting from the points on 3D. Give the motivation (and the principal differences) that define a model based on optical lenses.

Answer: