Homework 2

CS 411 - Artificial Intelligence I - Fall 2019

Matteo Corain 650088272

September 12, 2019

1 Question 1

1.1 Goal-based agent

A *goal-based agent* is a type of agent for which the decision to take in each situation, in terms of actions to be performed, cannot be described by a static set of rules, like in a *reflex agent* (either *simple* or *model-based*). Goal-based agents are *deliberative*, in the sense that they go through some sort of thinking process to figure out what action to take; in fact, they are able to predict the expected outcome of the different actions, which is then used to identify the one to be performed. The goal of an agent is represented by a *goal function* $g : \mathcal{S} \to \{0; 1\}$, defined by the designer, which associates to each possible state s of the environment a binary value according to the following logic:

$$g(s) = \begin{cases} 1 & \text{if state } s \text{ fulfills the goal} \\ 0 & \text{otherwise} \end{cases}$$

In order to achieve their objective, especially in case this is non-trivial and requires more than a single action to be fulfilled, goal-based agents cannot focus only on the present, but need to introduce a form of *planning* of the future. An essential precondition to planning is that the environment the agent works in is somehow sequential, i.e. the actions the agent has taken in the past somehow affect the reward it is going to receive for the subsequent ones.

During the planning phase, which happens offline (no action is performed in the meanwhile), the agent explores a planning space in order to identify a path (that is, a sequence of actions), possibly the shortest, that is expected to make the environment transition to a state in which the goal function returns a positive value. This operation is generally not trivial, given that the search space grows exponentially with the number of actions $|\mathcal{A}|$.

Once a path capable of satisfying the goal has been identified, the agent starts acting according to the identified sequence. If the environment is deterministic, then it is guaranteed that, after all actions in the path have been performed, its state will be such that the goal function returns 1; this means that the planning phase may be executed only once when the agent is first started. If the environment is stochastic, instead, the planning phase should be (partially) repeated after each action: in fact, in this case there is no guarantee that the environment has transitioned to the expected state and, if not, the agent must react accordingly, possibly recomputing the path inside the search space.

1.2 Utility-based agent

A goal-based agent acts in a way that, sooner or later, its goal will finally be achieved; in case there are many alternative ways for fulfilling it, this may in general lead to sub-optimal behaviors, since there is no concept of ranking of the different solutions (the goal function always returns 1 if the goal is fulfilled, in whatever way). A *utility-based agent* tackles with this problem by associating a score, called *utility*, to each alternative solution it has been able to compute that achieves the agent's goal.

In order to associate paths in the search space to their utility score, utility-based agents make use of a so-called *utility function* $u : \mathcal{S} \to \mathbb{R}$, which is the equivalent of the goal function for a goal-based agent, the difference being that the utility function associates a real number to each state. This function generally represents a sort of internal estimation (based on the limited knowledge of the agent) of the performance measure.

A utility-based agent behaves similarly to a goal-based agent, in the sense that it also has to perform a planning phase in order to compute possible paths in the search space. The stopping condition, however, is different: in this case, in fact, the utility function assigns to each path a real number. The agent will therefore select the sequence of actions that, within a certain future time window, yields the best expected utility value. Also in this case, if the environment is stochastic, the expected utility as computed by the agent should always be compared to the actual reward obtained by performing each action, possibly recomputing the path to be followed.

Utility-based agents are more flexible than goal-based agent, especially when they are required to fulfill conflicting goals. In fact, while a goal-based agent may simply decide to target a single goal (possibly the one achievable in the lower number of actions), an utility-based agent aims to find the solution able to yield the overall best expected reward.

2 Question 2

2.1 Fully/partially observable

An environment is classified as fully observable if the agent is equipped with sensors capable to perceive at any point in time all the aspects of the environment that are relevant for the selection of the action to be performed; otherwise, the environment is classified as partially observable.

In this case, in analogy with the example case of the vacuum cleaner, the variables of interest for a thorough description of the state of the environment are represented by:

- The *location* of the robot;
- The *presence of a star* in each location the robot can pass through.

Supposing to associate each location with a natural number $n \in [1; 12]$, the state of the environment may be represented in formal terms by a set $\mathcal{S}$ consisting of k-dimensional vectors, with $k = 13$, in which:

- The first component, $s_1 \in [1; 12]$, represents the square the robot is currently in;
- All the following components, $s_i \in \{\text{Star}; \text{No-Star}\}$, with $i = 2, ..., 13$, represent the presence or absence of a star in the $(i-1)$-th square.

If the robot is equipped with a set of complex sensors, capable of perceiving all the thirteen components that describe the environment's state at once, then the environment may be considered *fully observable*. If we consider instead a much simpler robot, capable for example of sensing its location and the presence of a star only within a more limited number of squares from its position (e.g. only in adjacent squares), then the environment has to be considered *partially observable*.

2.2 Deterministic/stochastic

An environment is classified as deterministic if it is possible to precisely state what the next state of the environment will be, given the current state and the action performed by the agent; if this is not possible, then the environment is classified as stochastic.

In this case, conceptually speaking, once that the state of the environment and the action the robot is going to perform are known, it is possible to infer without any doubt what the next state of the environment will be: if the robot moves we know how the location value is going to change, and if it picks up a star we know that the state of the current square is going to be set to No-Star. In other words, from a theoretical point of view, this environment does not present any source of uncertainty: it is very clear what the next state will be when the agent will perform its actions; thus, this environment may be considered *deterministic*.

However, if we were to implement this agent into a real-world object, then many different sources of uncertainty would be introduced, possibly preventing it from modifying the state of the environment in the expected way: for example, it may find unexpected obstacles on its way or it may break itself. Since it would be impossible to keep track of all unforeseen events that may affect the way the robot behaves in a physical environment, this has to be considered *stochastic*.

2.3 Dynamic/static

An environment is classified as dynamic if it can change its state without the direct intervention of the agent; otherwise, the environment is classified as static.

In this case, there are many ways in which the environment may change without the intervention of the agent; for example, stars may be dynamically added or removed. If we take into consideration those factors in the description of the environment, then it has to be considered *dynamic*; if instead we suppose that there is no element that can possibly modify the state of the environment without the intervention of the agent, then this may be considered *static*.

2.4 Episodic/sequential

An environment is classified as episodic when the actions that the agent performed in the past do not affect its decision for the following one; if instead the consequences of past actions influence in some way the decision process of the agent, then the environment is classified as sequential.

The described case represents a typically *sequential* scenario: the ability of the robot to move in the environment, in fact, impacts on its future decisions because it has an effect on the state of the environment, and in particular on the location variable. In order to pick up the stars, in fact, the robot is required to move in a way that allows it to reach the location in which the desired object is; after each move, the state of the environment changes and the robot has to take into account this modification to select the next action to take.

2.5 Known/unknown

An environment is classified as known when the agent knows the rules according to which this environment works, which can be used to predict the outcome of the different actions it can take; if the agent (at least initially) lacks this knowledge, then the environment is classified as unknown.

In this case, it is possible to suppose that the described environment works according to the following set of rules:

- The robot can only move inside the presented 3x4 matrix of locations;
- If the robot finds itself in a spot where a star is present, then it can pick it up.

If the robot knows this boundaries to the actions it can choose from the beginning, for example because it was programmed in a way that enforces the fulfillment of those rules, then the described environment has to be considered *known*. Instead, if the robot is initially unaware of the presence of these rules (for example, it does not know the geometry of the environment) and has to figure them out during its operation through a learning process, then the described environment has to be considered *unknown*.

2.6 Continuous/discrete

An environment is classified as continuous if the state of the environment, the time, the percepts and the actions of the agent are described by continuous variables; otherwise, the environment is classified as discrete.

In this case, we have the following:

- The state of the environment is represented by a discrete variable, that is a 13-dimensional vector as previously described;
- The time may be handled both in a discrete way (e.g. the robot perceives and acts only at defined instants of time) or in a continuous way (e.g. the robot continuously perceives the environment and acts accordingly);
- The percepts of the agent are described again by a discrete variable, represented by a vector whose components depend on the sensory capabilities of the robot;
- The actions of the agent may be described both in a discrete way (if we suppose that all actions are atomically performed and distances are measured in multiples of the dimension of a square) or in a continuous way (the robot may perform an action only partially or move for a distance that is not multiple of the dimension of a square).

If we consider both time and actions to be handled in a discrete way, then the described environment has to be classified as *discrete* (no continuous variable is necessary for its characterization); otherwise, it has to be classified as *continuous*.