# Part 1: Definition of Intelligent Agents

**Note**

* Whenever Hannes asks *why* questions, refer to definitions
* State *all* assumptions made and *why*.

Each of the following scenarios describes a situation that can be solved by one or more intelligent agents. For each, define the environment and the agent.

## One agent to rule them all, one agent to find them. This agent will patrol a particular farming property that contains many wireless sensor nodes, some of which are mobile (for example attached to large animals or vehicles). It will communicate with each sensor node to report its presence, receive a “report” from the sensor node covering the last day, and then return that report to a central repository. The report is based on what each sensor is designed for (such as the movements and eating habits of a cow) as well as information such as the battery life remaining of the sensor node.

### Agent

|  |
| --- |
| **Farmland-wellness-checking drone**, as a land-based vehicle might be an obstacle for other vehicles on the farm, in addition to animals and people. |
| **Performance Measures** |
| * Minimises patrol time * Minimises travel distance * Minimises energy consumption * Doesn’t endanger animals or humans on the property * Doesn’t impede on other vehicle |
| **Environment** |
| * Farm, containing:   + Living obstacles (animals and humans)   + Inanimate obstacles (other vehicles, trees, fences etc.) |
| **Actuators** |
| * Roll * Pitch * Yaw * Thrust * Brakes * Turn signals * Horn * Wi-Fi file transfer |
| **Sensors** |
| * GPS * Cameras * Battery life sensor * Speedometer * Odometer * Accelerometer |

### Environment

|  |
| --- |
| **Fully Observable Partially Observable or Unobservable?** |
| Partially-observable, as the agent cannot at any given time know the location of every *small* animal on the farm, as they apparently do not have sensor nodes attached to them. |
| **Single Agent or Multiagent?** |
| Single agent because the agent only interacts with “sensor nodes” and “a central repository” (i.e. some form of database), neither of which can be classified as an *agent*. |
| **Deterministic or Stochastic?** |
| Stochastic, as no agent can reliably predict the behaviour of animals and humans. |
| **Episodic or Sequential?** |
| Sequential because the agent making a short-term action such as visiting a randomly decided node could result in the agent having to navigate a longer path to visit each node. |
| **Static or Dynamic?** |
| Dynamic, as the animals and people on the farmland can move while the agent is deliberating the next sensor node to visit, for example. |
| **Discrete or Continuous?** |
| Both, because the discrete/continuous distinction applies to:   1. The *state* of the environment 2. The way *time* is handled 3. The *percepts* and *actions* of the agent   Surveying the wellness of a farm is a continuous-state and continuous-time problem as the speed and location of the agent and of animals, people and other vehicles sweep through a range of continuous state and do so smoothly with the passage of time. On the other hand, Input from digital cameras is discrete, but is typically treated as representing continuously varying aspects of the farm. |

## One agent to bring them all and in the darkness bind them. This agent is used to track bats who live in a cave system near to a town. It needs to fly around to do so, and its main purpose is to detect when excessive bats are settling in a location too close to a town (such as on a patio or under a tree). The location is generally detected through sound (bat groups can be noisy) but visual sensors are also used to check known risk areas. When this is detected, the agent will buzz the bats with ultrasonic sound and flashing lights to “encourage” them to move back to the cave.

### Agent

|  |
| --- |
| **Bat-tracking drone** |
| **Performance Measure** |
| * Minimises tracking time * Minimises travel distance * Minimises energy consumption * Doesn’t endanger bats in the cave |
| **Environment** |
| * Cave system, containing:   + Living obstacles (bats)   + Inanimate obstacles (stalactites, stalagmites, walls etc.) |
| **Actuators** |
| * Roll * Pitch * Yaw * Thrust * Ultrasonic sound * Flashing lights |
| **Sensors** |
| * GPS * Sonar * Cameras * Battery life sensor * Speedometer * Odometer * Accelerometer |

### Environment

|  |
| --- |
| **Fully Observable Partially Observable or Unobservable?** |
| Partially-observable, as the agent cannot at any given time know the location of every bat in the cave, as it needs to “detect when excessive bats are settling in a location too close to a town” and to do so it needs to first "track" (i.e. find) the bats. |
| **Single Agent or Multiagent?** |
| For the sake of the unit, single agent, as the actions of the bat do not maximise the drone’s performance measure. For example, a bat may want to settle with another group of bats, however, doing so will likely decrease the drone’s performance measure as it might mean the drone takes longer to displace the bats. |
| **Deterministic or Stochastic?** |
| Stochastic, as no agent can reliably predict the behaviour of the bats. |
| **Episodic or Sequential?** |
| Sequential because the agent making a short-term action such as visiting a randomly decided known risk area could result in the agent having to navigate an overall longer path to visit each area. |
| **Static or Dynamic?** |
| Dynamic, as the bats in the cave can move while the agent is deliberating which area it should visit next, for example. |
| **Discrete or Continuous?** |
| Both, because the discrete/continuous distinction applies to:   1. The *state* of the environment 2. The way *time* is handled 3. The *percepts* and *actions* of the agent   Tracking bats in a cave is a continuous-state and continuous-time problem as the speed and location of the agent and the bats sweep through a range of continuous states and do so smoothly with the passage of time. On the other hand, Input from digital cameras is discrete, but is typically treated as representing continuously varying aspects of the cave. |

## An agent that “reads” texts as they are added (to World Online Library) and uses heuristic methods to assign keywords to them. The idea is that this can be used to quickly find books on particular subjects. The agent needs to be able to communicate with customers using natural language (via online text chat) although there is no requirement to pretend to be anything other than a software agent.

### Agent

|  |
| --- |
| Virtual librarian |
| **Performance Measure** |
| * Minimises run-time * Minimises space usage * Assigns relevant keywords to texts * User satisfaction |
| **Environment** |
| * Texts * Users * World online library API |
| **Actuators** |
| * readText() method and accompanying readLine() method, which are used to read texts into memory * assignKeyword() method, which assigns a keyword to a text * storeText() method, which is used to add the text to some kind of data structure * writeText() method, which is used to write the stored texts to file for long-term storage * readUserKeyStrokes(), which is used to read user queries |
| **Sensors** |
| * Contents of text files * Keyboard entries from users |

### Environment

|  |
| --- |
| **Fully Observable Partially Observable or Unobservable?** |
| Partially-observable, as the agent cannot at any given time know the entire contents of a text. After all, it needs to *read* them. |
| **Single Agent or Multiagent?** |
| Multiagent as a customer using the app and leaving a 5-star review of their experience using the agent in turn maximises the virtual librarian’s performance measure as the customer found its services very useful. |
| **Deterministic or Stochastic?** |
| Stochastic, as the virtual librarian doesn’t know what queries a user may ask. |
| **Episodic or Sequential?** |
| Sequential because the agent providing a user with some particular response may influence how the user interacts with the agent in the future. |
| **Static or Dynamic?** |
| Dynamic, as the user will begin typing a query when the agent is on standby (perhaps just after opening the app or web page) |
| **Discrete or Continuous?** |
| Both, because the discrete/continuous distinction applies to:   1. The *state* of the environment 2. The way *time* is handled 3. The *percepts* and *actions* of the agent   For example, the agent communicating with users is a continuous-state and continuous-time problem as the user's thoughts as to what queries to ask the virtual librarian sweep through a range of continuous states and do so smoothly with the passage of time. On the other hand, Input from the keyboard is discrete, but is typically treated as representing continuously varying queries of the user. |

## An agent that can be activated to receive audio input that is expected to contain natural language in English. The agent should summarize what it hears and send that summary to a predefined email address via Wi-Fi. A full recording and text log is kept on the agent for manual download.

### Agent

|  |
| --- |
| Voicenote capturing agent |
| **Performance Measure** |
| * Minimises speech-to-text parsing time * Accurately converts user speech to text * Minimises space usage in storing audio logs * Minimises space usage in storing text logs * Minimises sending time when sending the summarised voice note to the email address * User satisfaction |
| **Environment** |
| * User |
| **Actuators** |
| * parseSpeech() method to convert user speech to text * formulateResponse() method, which formulates a response to what the user asked * giveResponse() method which vocalises the response it created * saveSpeech() method, which is used to save what the user asked to a log file * saveText() method which is used to save the text-converted version of the users speech * sendToEmail() method which is used to send the summarised voice note to an email. |
| **Sensors** |
| * Speech detection sensor * Voice recorder * Some kind of “record-speech” button in an accompanying app. |

### Environment

|  |
| --- |
| **Fully Observable Partially Observable or Unobservable?** |
| Fully Observable as, in essence, all the agent is really doing is capturing what the user is saying, start to finish, observing the full content of their speech. |
| **Single Agent or Multiagent?** |
| Multiagent as a customer using the app and leaving a 5-star review of their experience using the agent in turn maximises the voice note performance measure as the customer found its services very useful. |
| **Deterministic or Stochastic?** |
| Stochastic, as the app doesn’t know what notes a user may want to store. |
| **Episodic or Sequential?** |
| Sequential as storing particular speech and log files will result in less space to use for storage future log files. Simply put, the agent’s future capacity to store data will be impacted by how much data it stored previously (due to less space available). |
| **Static or Dynamic?** |
| Dynamic, as the user may use another app while the agent is processing a voice note |
| **Discrete or Continuous?** |
| Both, because the discrete/continuous distinction applies to:   1. The *state* of the environment 2. The way *time* is handled 3. The *percepts* and *actions* of the agent   For example, the agent receiving and summarising user voice notes is a continuous-state and continuous-time problem as the user's thoughts as to what voice notes they want to record sweep through a range of continuous states and do so smoothly with the passage of time. On the other hand, input from the voice recorder is discrete (limited by bitrate for sample), but is typically treated as representing continuously changing voice of the user. |

## An agent that is attached semi permanently to cars of certain drivers. The agent tracks its location on a map that it can update via mobile network, including the speed that it travels and any sudden motions (abrupt stops, etc.). The agent reports certain information if it receives a query via the mobile network and that query contains an appropriate code. The information can be used to determine the current location of the agent (if the car is stolen, for instance) or details on recent issues such as speeding and stops sudden enough (or other detected motions) that are likely caused by collisions.

### Agent

|  |
| --- |
| **GPS-tracking agent** |
| **Performance Measures** |
| * Accurately tracks its location * Accurately tracks its travel speed * Accurately identifies sudden motions * Updates location in a timely manner * Updates travel speed in a timely manner * Reports information in a timely manner * Minimises mobile data usage * Minimises energy consumption * User satisfaction |
| **Environment** |
| * Host (vehicle) |
| **Actuators** |
| * Location updater * Speed updater * Report producer * Data transmitter |
| **Sensors** |
| * GPS * Battery life sensor * Speedometer * Odometer * Accelerometer |

### Environment

|  |
| --- |
| **Fully Observable Partially Observable or Unobservable?** |
| Fully observable because the agent can observe all of the *relevant* aspects of the vehicle–namely, it’s location, travel speed and acceleration. |
| **Single Agent or Multiagent?** |
| Let’s use the queries as an example. The environment is most likely going to be multi-agent  as their needs to be some level *reasoning* as to what queries are sent so that the agent isn’t just reporting random information. The only entities which are capable of reasoning are agents. Secondly, because there seems to be some level of *cooperation* in this network–in the sense that one agent receives a query, prepares a report then returns a report–we can consider the entity which sent the query as an agent and not as part of the environment (because it’s not adversarial). |
| **Deterministic or Stochastic?** |
| Stochastic, as the agent’s actions don't really have any bearing as to how the human driving the vehicle behaves. |
| **Episodic or Sequential?** |
| Sequential because the agent making a short-term action such as visiting a randomly decided node could result in the agent having to navigate a longer path to visit each node. |
| **Static or Dynamic?** |
| Episodic as at every moment the agent merely tracks a certain aspect of the vehicle and updates it via a mobile network. Even if the agent reported the vehicle as speeding just a moment ago, it is still going to continue its tracking procedures. |
| **Discrete or Continuous?** |
| Both, because the discrete/continuous distinction applies to:   1. The *state* of the environment 2. The way *time* is handled 3. The *percepts* and *actions* of the agent   Tracking the state of a vehicle is a continuous-state and continuous-time problem as the speed, location and acceleration of the vehicle (and, in turn, agent) sweeps through a range of continuous states and does so smoothly with the passage of time. On the other hand, input from its accelerometer is discrete, but is typically treated as representing the continuously varying acceleration of the vehicle. |

# 

# Part 2: Search

Each of the following problems is something you may consider solving using some form of graph search:

## A chess board that is nxn squares on which you need to place n Queen pieces so that no two queens are attacking each other. This is an extension of the 8-Queens puzzle. Note that n is decided at the start of each “game” so your agent will need to be able to deal with boards of different size.Consider the map shown in the figure below (also taken from COMP3006). Each state is to be set to one of three ratings - “desirable”, “overcrowded” or “flooded”. No two adjacent states can have the same rating. (You can replace the ratings with numbers if you wish.)

## Consider the map shown in the figure below (also taken from COMP3006). Each state is to be set to one of three ratings - “desirable”, “overcrowded” or “flooded”. No two adjacent states can have the same rating.

## Some form of backtracking search

## A rubik’s cube.

## An instance of Wumpus world that is fully observable (i.e., the agent can perceive the full map and the location of all pits as well as the gold and the Wumpus). The agent starts at (1,1) and needs to get to the gold, Grab, and then return to (1,1).

## Tic-tac-toe or “naughts and crosses”.

For each of the above problems, consider the application of each of the following search methods. Which of the search methods are more appropriate and which are possibly not appropriate (or impossible)? Why?

* DFS
* BFS
* A\*
* RBFS
* Hill Climbing

*Note: The reason should relate to the problem, not your liking (or disliking) for a particular search strategy.*

For each problem, choose one appropriate method and apply that, either by coding it or on paper. As with the tutorials, you don’t need to do all of it, just enough to get an idea of what’s happening. In particular, you should be able to answer questions about how the search tree is generated and how the method performs on the tree.