Artificial Intelligence : Lab Exercise 1

chandru@iitpkd.ac.in

1 Motivation

The notions of *agents* and *environments* are key in AI and be specified as follows:

- An agent interacts with the environment (informaly surroundings) by performing *actions*. The agent can also *perceive* (informally sense) the environment. In general, an agent is expected to choose its actions based on what it perceives/senses.
- The environment is described by its *state* (informatlly status/configuration), and it responds to the action of an agent by changing its state.

Please do look at https://github.com/aimacode/aima-python/ for example implementation of agents and environments.

We will look at the some simple examples of agent-environment interaction.

2 Bunny needs help

Bunny is drifting in the middle of the ocean, and it does not know the location of the shore. Assume that the world is 1-dimensional (shore could be left or right of the bunny), and come up a python implementation of a) environment and b) wise-bunny agent which reaches shore. In order to complete a) and b) you have to **identify states, precepts, actions and the wise-bunny's strategy**.

What happens if the world is 2-dimensional? How will you modify?

3 Vaccum Robot

Imagine a 2-dimensional world with each location described as $l = (x, y)$, where x, y are integers. There is dirt in a location $l_d = (x_d, y_d)$ and the vaccum robot has to start from location $l_s = (x_s, y_s)$. Come up a python implementation of a) environment b) wise-vr agent which picks the dirt.

The following example is to illustrate the importance of states internal to agents.

4 Count Words

Write a python code that uses only `f.read(1)` command that reads one character at a time from a text file to count the number of words in that file. Note that words in a sentence could be separated by multiple spaces.

Artificial Intelligence : Lab Exercise 2

chandru@iitpkd.ac.in

1 Motivation

We want to see for ourselves how *Google* maps works when it is suggesting routes based on congestion. First step is to simulate the traffic and congestion (which is the exercise of this lab session). Once we have created the traffic, the hope is that, we can run some heuristic search algorithm to find the best route (will change based on congestion) from source to destination (for future classes).

2 Simple Traffic Simulator

Creating environments is key, and in this lab session we will create an environment for the *Traffic Control* problem defined as below:

- The connection of the road network is specified as a matrix M (filename is *road*). There are 10 sites $(0, 1, \ldots, 9)$. Entry $M(i, j)$ denotes the length of the road between site i and j (the roads are uni-directional). If the entry is 0, it means no connection.
- Vehicles depart from a source and then take a path (filename *vehicle*).
- The time of departure (in minutes) is also given in filename *time*, n^{th} vehicle departs at time instance n.
- Load the three files mentioned above using command, for example,
  pickle.load(open("road", "r")).
  Each of them is a *numpy* matrix. So, you have to *import numpy* and *import pickle*.
- The roads are uni-directional (i.e., there could be a road from A to B, but not from B to A). The speed in unit distance per hour of the vehicle in a road is given by the formula

  $$exp(0.5x)/(1 + exp(0.5x)) + 15/(1 + exp(0.5x)), \qquad (1)$$

  where x is the number of vehicles ahead in a given road. If there are n vehicles in the road, then for the first vehicle $x = 0$, second vehicle $x = 1$, and for the last vehicle $x = n - 1$. The time taken to travel the road is $\frac{dist}{speed}$.

3 Some tips

The traffic system is a *event-driven* environment. Here, we have to keep track of what happens when, i.e., the state evolves based on activity. In particular, the state will contain: a) the current road the vehicle is in, b) the time it will leave the current road (equal to the time it enters plus the time taken to travel the road). Once the vehicle leaves the road network, we no longer care about it (it can be removed). Please evolve the data structure along this idea.

4 General Instructions

- You are allowed to work in groups of 2 or 3. Once you finish the exercise, please send a .zip file that contains your code (as well as an info.txt with your names/roll and a brief description of the code) to Rekha Raj C T (111804102@smail.iitpkd.ac.in). The evaluation will be $20, 20, 60$ (two internals and finals), but your submissions of lab exercise will be taken as additional credit (just in case to save you if you did badly in the finals or the internals).

Artificial Intelligence : Lab Exercise 1

chandru@iitpkd.ac.in

1 Motivation

Knowing the *right* decision is key. In this session, we will look at how *curse-of-dimensionality* makes decision making diffcult. To understand the difficulty, we will make *random* decisions, and study its effect as dimensions increase. Let us create the following three environments (in below S stands for state-space and A for action-space)

- *Line* environment whose state space is given by $S_{1-d} = \{0, 1, \ldots, L-1, L\}$, the agent starts at $s_0 = \lceil \frac{L}{2} \rceil$. There are 2 actions namely move left by one step and move right by one step.
- 2-*d Grid* environment has state space given by $S_{2-d} = S_{1-d} \times S_{1-d}$, the agent starts at $s_0 = (\lceil \frac{L}{2} \rceil, \lceil \frac{L}{2} \rceil)$. There are 4 actions namely move up, down, right, left by one step.
- Generalize the above to a 3-*d Grid* environment. The agent again starts at the center, and the agent has 8 actions (why?).
- Implement a random agent which performs a random action at any given state. Use *numpy.rand.randint* to generate random actions.
- The agent never leaves the grid, for illegal moves the agent stays in the same position.
- In all the evnironment the reward is 0 for all the states except for the goal state $s = (s(1), s(2), s(3))$, such that $\sum_{i=1}^{d} s(i) = dL$, where d is the dimension of the problem (this is the goal state). The reward at the goal state sis 1.
- The aim of the random agent is to reach the goal state.
- Measure the time it takes reach the goal in each of the cases. Try for cases $L = 2, 3, 4$. Is there any realtion to $|A|^{|S|}$?

2 Simple Traffic Simulator

It turns out that this can achieved by *Priority Queues* in a simple and neat manner. Please try that as well.

Deadline for submitting all the previous experiments is this weekend. As mentioned before, you can work in groups of 2-3. Please send the code to Rekha Raj C T (111804102@smail.iitpkd.ac.in)

Artificial Intelligence : Lab Exercise 1

chandru@iitpkd.ac.in

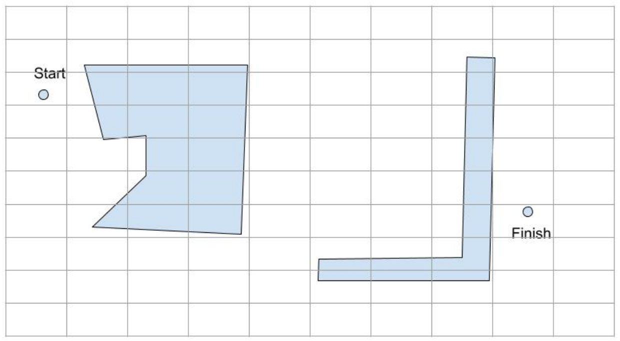
1 $(n^2 - 1)$-puzzle

Implement the $(n^2 - 1)$-puzzle environment. Also, implement parity function using the following definition:

- A position p_i is said to occur after p_j, if p_i occurs to the right of p_j in the same row or if p_i occurs at any row below p_j. This defines the ordering.
- For a state s, let $d(s)$ denote the number of rows + number of columns that the empty square is away from the bottom right corner.
- Let $I_{\text{True}} = 1$ and $I_{\text{False}} = 0$ (this is known as *indicator* function).
- For a state s, parity is given by $mod(d(s) + \sum_{p_i, p_j > p_i} I_{p_j(s) < p_i(s)}, 2)$, where $p_i(s)$ is the number at the i^{th} position.
- $mod(n, 2)$ is equal to the remainder on dividing n by 2.
- Imagine the empty square to be n^2.

2 Robot Navigation

Create the following navigation environment with blockades. Take $G = 100$ (grid-size), the blocked places can be 0's and the other ones can be represented by 1. The figure is only illustrative, and in a grid blocking would mean blocking the entire cell.



Artificial Intelligence Lab 5,6,7

September 14, 2018

1 Lab 5

1. Solve $n^2 - 1$ puzzle with BFS

2 Lab 6

1. Perform $A*$ on a grid with some of the cells blocked.
    - with 4 directions
    - with 8 directions

3 Lab 7

1. Consider the road network in the Figure 1. Find the shortest path from source to goal using A^* algorithm with heuristic as Euclidean distance.
2. Suppose there are two modes of transportation, Cycle and Bus. The speed of cycle in a road is 25 km/hr. The speed of bus is 50 km/hr. The speed of the bus reduces to 37.5km/hr on 50% congestion, and reduces to 10km/hr on full congestion. Speed of the cycle doesn't vary. Bus cannot travel on roads with distance less than or equal to 3km/hr. At each node the optimal mode of transportation is chosen. Assume that the budget is Rs. x and travel cost of bus is Rs. y per hour. Find the shortest path from source to destination and the time taken on
    (a) full congestion
    (b) 50% congestion
    (c) No congestion

    The node numbers along with their x and y coordinates are given in the file nodes.xlsx. The distance between the nodes are given in the file distance.xlsx. The nodes that are not connected are denoted by letter N.

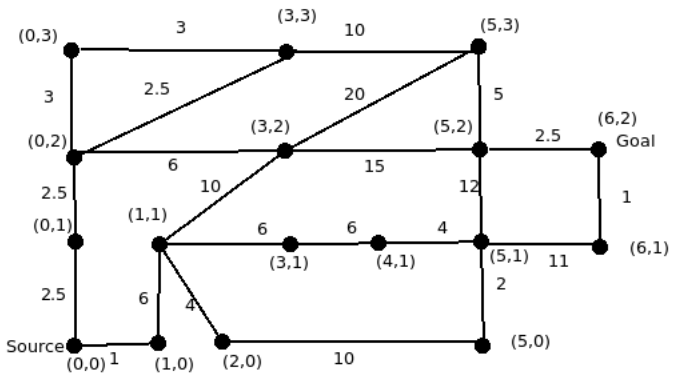


Figure 1: Road network