Assignment 2: Probabilistic Reasoning

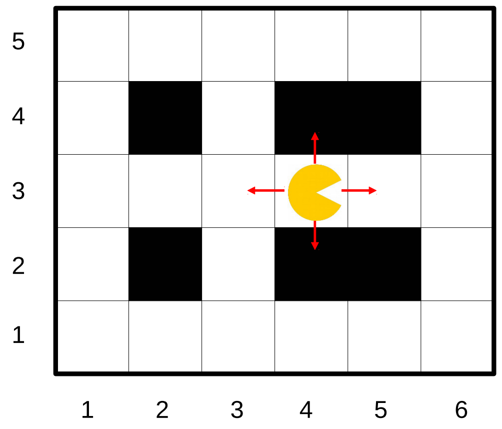
Submission: Tuesday April 19th
Groups of maximum 2 students

Prof. Fabio A. González
Intelligent Systems - 2016-I
Maestría en Ing. de Sistemas y Computación

---

1. (2.5) Bayes' net inference

   Consider the following Pacman maze:



The Pacman can sense his environment through four sensors that tell him if there is a wall, or not, north, south, east or west of his position. The Pacman knows the configuration of the maze but, in general, he does not know his position in it. He must determine it from his perceptions. However, the perceptions are not perfect, sometimes the reading from a sensor may be erroneous with an $0 \leq \epsilon < 1$ probability. To model this situation we will use the following random variables:

- $X \in \mathbb{R}^2$: the Pacman position.
- $E_N, E_S, E_E, E_W \in \{0, 1\}$: the Pacman perceptions.

(a) Build a Bayes' net that represent the relationships between the random variables. Based on it, write an expression for the joint probability distribution of all the variables.

(b) Assuming an uniform distribution for the Pacman position probability, write functions to calculate the following probabilities:

   i. $P(X = x | E_N = e_N, E_S = e_S)$
   ii. $P(E_E = e_E | E_N = e_N, E_S = E_S)$
   iii. $P(S)$, where $S \subseteq \{e_N, e_S, e_E, e_W\}$

(c) Now we will consider a scenario where the Pacman moves a finite number of steps n. In this case we have n different variables for the positions $X_1, \ldots, X_n$, as well as for each one of the perceptions, e.g. $E_{N_1}, \ldots, E_{N_n}$ for the north perception. For the initial Pacman position, assume an uniform distribution among the valid positions. Also assume that at each time step the Pacman choses, to move to, one of the valid neighbor positions with uniform probability. Draw the corresponding Bayes' net for $n = 4$.

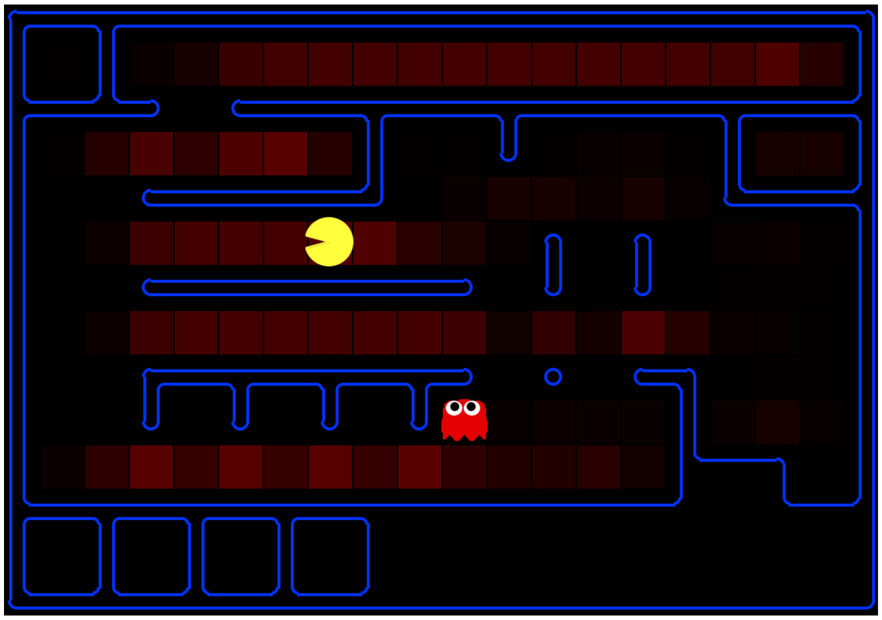
(d) Write functions to calculate the following probabilities:

  i. $P(X_4 = x_4|E_1 = e_1, E_3 = e_3)$
  ii. $P(X_2 = x_2|E_2 = e_2, E_3 = e_3, E_4 = e_4)$
  iii. $P(E_4 = e_4|E_1 = e_1, E_2 = e_2, E_3 = e_3)$
  iv. $P(E_{E_2} = e_{E_2}|E_{N_2} = e_{N_2}, E_{S_2} = E_{S_2})$

  where E_i and e_i correspond to $(E_{N_i}, E_{S_i}, E_{E_i}, E_{W_i})$ and $(e_{N_i}, e_{S_i}, e_{E_i}, e_{W_i})$ respectively.

(e) The solution has to be reported in an IPython notebook following the format and instructions in the notebook in `https://github.com/fagonzalezo/is-2016-1/blob/gh-pages/assign2.ipynb`.

2. (2.5) Pacman localization problem



This problem is based on the search problems posed in the Project 4 of [AI-edX]. However, instead of inferring the position of the ghosts, we need to infer the position of the Pacman based on perceptions. The situation is similar the one described in question 1; however, here we need to use probabilistic reasoning over time, i.e., hidden Markov models.

To start, download and uncompress the file in `http://fagonzalezo.github.io/is-2016-1/blindPacman.zip`.

(a) First, you have to make the Pacman able to sense his environment. His sensors work as discussed in question 1. You must implement the function `getObservationDistributionNoisyWall` in `busters.py` to calculate $P(\text{noisyWall} \mid \text{truePerception})$.

(b) You have to make the Pacman able to calculate his beliefs about his position based on his perceptions. You must implement the methods `initializeUniformly` and `observe` of the class `ExactInference` in `inference.py`.

(c) Now, you have to make the Pacman able to take the time into account to update is belief. You must implement the method `elapseTime` of the class `ExactInference` in `inference.py`.

(d) In this step, you will implement a strategy for Pacman that takes into account the beliefs he calculates. To do so, You must implement the method `chooseAction` of the class `BustersAgent` in `bustersAgents.py`. The process to choose the actions is the following:

   i. Given de belief distribution choose the position with the maximum probability. If there is a draw choose the position with the maximum position according to lexicographic order $(x_1, y_1) \leq (x_2, y_2) \iff x_1 \leq x_2 \vee (x_1 = x_2 \wedge y_1 \leq y_2)$. **Tip**: use the method `argMax` of the class `Counter` in `util.py`.

   ii. From this position, choose the action that brings you closer to the ghost, i.e., the position with the minimum distance, in terms of the number of steps in the maze, to the position of the ghost. Take into account that the ghosts does not move and always stays in the same position ($(1, 3)$).

(e) In this question, and the following, you will use approximate inference based on particle filtering. Implement the methods `initializeUniformly`, `getBeliefDistribution` and `observe` of the class `ParticleFiltering` in `inference.py`.

(f) Implement the method `elapseTime` of the class `ParticleFiltering` in `inference.py`.

To visualize your implementation in action you can use the following command for the agent using exact inference:

```
python autograder.py -t test_cases/q3/3-gameScoreTest
```

And the following command for the approximate inference agent:

```
python autograder.py -t test_cases/q5/3-gameScoreTest
```

Evaluate your solution using the following grader:

```
python autograder.py
```

The assignment must be submitted as a compressed file containing the files (and ONLY these files): `assign2.ipynb`, `inference.py`, `busters.py` and `bustersAgents.py` through the following Dropbox file request, before midnight of the deadline date. The file must be named as `is-assign2-unalusername1-unalusername2`.zip, where `unalusername` is the user name assigned by the university (include the usernames of all the members of the group).

References

[Russell10] Russell, S y Norvig, P. 2010 Artificial Intelligence: a Modern Approach, 3rd Ed, Prentice-Hall

[AI-edX] CS188x_1 Artificial Intelligence, UC Berkley, edX,Fall 2013, `https://www.edx.org/course/artificial-intelligence-uc-berkeleyx-cs188-1x`.