Ozgur Cem Birler
20.12.2017
50118

HW5: Reasoning over Time

Written Q1: Run the autograder on q2 and watch the probabilities. Why do they settle even though the ghost is moving? Can you tell the two ghosts apart and if so how? (Hint: Run these test cases q2/1-ExactElapse, q2/2-ExactElapse, if you need to observe them individually)

In 1st and 4th maps, distributions of probabilities are almost equal, which indicates ghosts in these maps move randomly, but in 2nd and 3rd maps southern places in map have higher probabilities which indicates that the ghosts in these maps tend to go south in general. In either case, probabilities settle because ghosts moving agent doesn't change by time, meaning if it tends to go south then it always have this tendency independent of time, or if it moves randomly then it will continue to move randomly.

Written Q2: Try the following lines of code and watch the probabilities settle: python autograder.py -t test cases/q1/2-ExactObserve python autograder.py -t test cases/q1/3-ExactObserve Why is it the case that in one of them we can find the ghost but not in the other one?

In 2-ExactObserve Pacman is not able to find the ghost because it is restricted in one box thus it is unable to change its point of observation. Since it has a limited observation 4 furthest corners of the 4 islands (disconnected areas from where Pacman is) are equally likely. However, in 3-ExactObserve Pacman can move, even though in limited area, the ability to change its point of observation gives him more information about the ghost positions, specifically the distance signal it receives is higher on average when Pacman is in the top left corner hence it infers that ghost must be at the bottom right island.

Written Q3: Run the autograder on q4 and watch the probabilities. Can you tell when the particles get re-initialized? Comment on the reason(s) on why pacman gets in that situation? Would increasing the number of particles be a solution?

Particles are re-initialized when Pacman can't find the ghost. We can this on 6th map when ghost is not in the place where Pacman thinks it is. Pacman gets in that situation because of based on the evidence particles get zero weight so Pacman can't reach a conclusion. Increasing the number of particles wouldn't be a solution because beliefs are updated by emission mode for stationary position and this doesn't include the movement of the ghost. If the number of particles are more than legal positions, then re-initialization number remains same. The time elapse should be included to avoid this problem

Written Q4: Compare how the probabilities evolve between the exact inference and the approximate inference cases (q1, q2 vs q4, q5). Also comment on if 5000 particles make sense for the problems you have seen.

In approximate inference probabilities evolve faster compared to exact inferences because we do less computation since we lose from the accuracy but gain from the speed. As explained in the previous question in q4 increasing the number of particles doesn't make any sense but in

q5 having 5000 particles make sense because the map has a finite size and more particles will enable pacman to get a more accurate reading.

Written Q5: In the DBN questions (q6, q7), you had to work on a particle that had multiple ghost positions. Their transition were dependent on each other but their emissions were not. How did you create a new particle with elapsed time and how did you calculate its weight? You can use mathematical equations to help you explain this.

In q7 we have multiple ghosts, so weights of new particles are calculated using the given getPositionDistributionForGhost method which takes the new gamestate, the index of the ghost that is to subject, agent it uses and it outputs the position distribution of the ghost. Since the function was already given, looping though the ghost with this function and then sampling from the result for each old particle gives the new particles.

Written Bonus: What is the difference between the sonar in this homework and the one we have seen in class?

Sonar in this project gives the noisyDistance between pacman and the ghost/s and the probabilities are derived from this reading. However, in the class, sonar detects the exact position of the ghost relative to pacman.