AI HW4

洪郡辰
R11944050

May 2, 2024

Q1

```
Question q1
===========
*** PASS: test_cases\q1\1-small-board.test
*** PASS: test_cases\q1\2-long-bottom.test
*** PASS: test_cases\q1\3-wide-inverted.test

### Question q1: 2/2 ###


Finished at 17:20:09

Provisional grades
==================
Question q1: 2/2
------------------
Total: 2/2

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

將定義好的參數轉換成需要用於 BayesNet 的參數。我們需要得到 variables,edges,constructBayesNet 用於建構 BayesNet。

在這個案例中的 variable 總共有五個參數分別為 [PAC,GHOST0,GHOST1,OBS0,OBS1]。

共有四種 edge 分別為 [(PAC,OBS0),(PAC,OBS1),(GHOST0,OBS0),(GHOST1,OBS1)]。

PAC,GHOST0,GHOST1 的 DOMAIN 為他們所能夠到達的位置，OBS0,OBS1 的 DOMAIN 代表 GHOST 與 PAC 加上 NOISE 的可能 MANHATTAN DISTANCE。

Q2

```
Question q2
===========
*** PASS: test_cases\q2\1-product-rule.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q2\2-product-rule-extended.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q2\3-disjoint-right.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q2\4-common-right.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q2\5-grade-join.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q2\6-product-rule-nonsingleton-var.test
***     Executed FactorEqualityTest

### Question q2: 3/3 ###


Finished at 17:21:15

Provisional grades
==================
Question q2: 3/3
------------------
Total: 3/3

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

形成一個 FACTOR 需要考慮三個部份分別為:

- inputUnconditionedVariables
- inputConditionedVariables
- inputVariableDomainsDict

因此需要透過傳入的 FACTORS 得到這三個部分。

首先是取出所有變數分別放入 UNCONDITION 及 CONDITION 內，如果在不同 FACTOR 內出現在兩種情況，只保留在 UNCONDITION 內。DOMAINDICT 部分則將所有 FACTOR 的 DOMAINDICT 都保存在 inputVariableDomainsDict。

得到這三個部分後就能形成一個全新的 FACTOR，這時我們要根據不同的 ASSIGNMENT 給其定

義 PROB, 所有 PROB 初始定義為 1。

我們透過在所有 FACTOR 的 OLDASSIGNMENT 檢查 NEWFACTOR 的所有 ASSIGNMENT，如果 OLDASSIGNMENT 跟 NEWFACTOR 中的 ASSIGNMENT 的 KEY 及 VALUE 都一樣就將其機率相乘設定為 NEWFACTOR 的 ASSIGNMENT 的機率。

檢查完所有情況後，就能得到正確的 NEW FACTOR。

Q3

```
Question q3
===========
*** PASS: test_cases\q3\1-simple-eliminate.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q3\2-simple-eliminate-extended.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q3\3-eliminate-conditioned.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q3\4-grade-eliminate.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q3\5-simple-eliminate-nonsingleton-var.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q3\6-simple-eliminate-int.test
***     Executed FactorEqualityTest

### Question q3: 2/2 ###


Finished at 17:21:46

Provisional grades
==================
Question q3: 2/2
------------------
Total: 2/2
```

第三題要實作在 factor 中刪減 variable 的動作，機率的部分定義成將刪減後有相同 assignment 的機率相加保存。variableDomainDict 則與原本的 factor 相同，不須因刪減變數而改變。

形成一個 FACTOR 需要考慮三個部份分別為:

- inputUnconditionedVariables
- inputConditionedVariables
- inputVariableDomainsDict

因此需要透過傳入的 FACTOR 得到這三個部分，並在 inputUnconditionedVariables 中刪除指定的變數。

得到這三個部分就能形成一個全新的 FACTOR，這時我們要根據不同的 ASSIGNMENT 給其定義 PROB，所有 PROB 初始定義為 0。

我們對 OLDFACTOR 中的 ASSIGNMENTS 做刪減變數的動作，刪減後的 ASSIGNMENT 就將他的機率值加到在 NEWFACTOR 中對應的 ASSIGNMENT 上。

檢查完所有情況後，就能得到正確的 FACTOR。

Q4

```
Question q4
===========
*** PASS: test_cases\q4\1-disconnected-eliminate.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q4\2-independent-eliminate.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q4\3-independent-eliminate-extended.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q4\4-common-effect-eliminate.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q4\5-grade-var-elim.test
***     Executed FactorEqualityTest
*** PASS: test_cases\q4\6-large-bayesNet-elim.test
***     Executed FactorEqualityTest

### Question q4: 2/2 ###


Finished at 17:22:14

Provisional grades
==================
Question q4: 2/2
------------------
Total: 2/2
```

第四題要實作的是刪減指定變數，需要透過交叉加入即刪減變數的方式來完成。

首先會透過 bayesNet.getAllCPTsWithEvidence(evidenceDict) 找到目前的 FACTORS，再透過 joinFactorsByVariable 加入需要被刪減的變數，形成一個 JOINEDFACTOR。如果這個 JOINEDFACTOR 有大餘一個的 UNCONDITIONEDVARIABLES，就透過 eliminate 將變數刪除，並把他加到 FACTORS 內。

最後再將 FACTORS 透過 joinFactors 將其轉變成一個 FACTOR 並將他 NORMALIZE 就完成了。

Q5

```
Question q5
===========
*** PASS: test_cases\q5\1-DiscreteDist.test
***      PASS
*** PASS: test_cases\q5\1-DiscreteDist-a1.test
***      PASS
*** PASS: test_cases\q5\1-ObsProb.test
***      PASS

### Question q5: 1/1 ###


Finished at 17:22:34

Provisional grades
==================
Question q5: 1/1
------------------
Total: 1/1
```

5a.

對 distribution 的 values 取 total，如果 total 為 0 則 do nothing。如果不為零就將數值轉為百分比 (val/total)。

對 distribution 的 values 取 total，如果 total 為 0 回傳 None。如果不為零則隨機轉一個 0-1 的數值，用此數值最為 threshold。透過這個 threshold 判斷 sample 的標準，回傳對應到的 key。

5.b

實作 getObservationProb，題目表示如果 noisyDistance = None 如果 ghostPosition == jailPosition 則回傳 1。如果不是則回傳 0。如果 noisyDistance != None 則如果 ghostPosition == jailPosition 就回傳

零。如果不是就計算 pacman 及 ghost 的 real distance，並且回傳 return busters.getObservationProbability(noisyDistance,re

Q6

```
Question q6
===========
*** q6) Exact inference stationary pacman observe test: 0 inference errors.
*** PASS: test_cases\q6\1-ExactUpdate.test
*** q6) Exact inference stationary pacman observe test: 0 inference errors.
*** PASS: test_cases\q6\2-ExactUpdate.test
*** q6) Exact inference stationary pacman observe test: 0 inference errors.
*** PASS: test_cases\q6\3-ExactUpdate.test
*** q6) Exact inference stationary pacman observe test: 0 inference errors.
*** PASS: test_cases\q6\4-ExactUpdate.test

### Question q6: 2/2 ###


Finished at 17:23:03

Provisional grades
==================
Question q6: 2/2
------------------
Total: 2/2

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

實作 observeUpdate，更新 self.beliefs。

對 self.allPosition 中的每個 pos 透過 self.getObservationProb 計算 ObsProb，將 self.beliefs[pos] 乘上 ObsProb 並記錄回去。

Q7

```
Question q7
===========
*** q7) Exact inference elapseTime test: 0 inference errors.
*** PASS: test_cases\q7\1-ExactPredict.test
*** q7) Exact inference elapseTime test: 0 inference errors.
*** PASS: test_cases\q7\2-ExactPredict.test
*** q7) Exact inference elapseTime test: 0 inference errors.
*** PASS: test_cases\q7\3-ExactPredict.test
*** q7) Exact inference elapseTime test: 0 inference errors.
*** PASS: test_cases\q7\4-ExactPredict.test

### Question q7: 2/2 ###


Finished at 17:23:33

Provisional grades
==================
Question q7: 2/2
------------------
Total: 2/2
```

實作 elapseTime，更新不同 time step 下的 beliefs。首先先定義一個 newBeliefs，用來記錄新的 beliefs。

對 self.allPositions 取得 each position，再對 each position 透過 self.getPositionDistribution 取得 action distribution for ghost 放在 newPostDist。

針對 NewPostDist 每個 pos 及對應到的 prob 去更新 newbeliefs，會透過 newBeliefs[newPosition] += self.beliefs[oldPos]*prob 紀錄新的 prob。

最後 self.beliefs 替換成 newBeliefs。

Q8

```
Question q8
===========
*** q8) Exact inference full test: 0 inference errors.
*** PASS: test_cases\q8\1-ExactFull.test
*** q8) Exact inference full test: 0 inference errors.
*** PASS: test_cases\q8\2-ExactFull.test
ExactInference
[Distancer]: Switching to maze distances
Average Score: 763.3
Scores:        778, 769, 759, 761, 776, 761, 758, 753, 763, 755
Win Rate:      10/10 (1.00)
Record:        Win, Win, Win, Win, Win, Win, Win, Win, Win, Win
*** Won 10 out of 10 games. Average score: 763.300000 ***
*** smallHunt) Games won on q8 with score above 700: 10/10
*** PASS: test_cases\q8\3-gameScoreTest.test

### Question q8: 1/1 ###


Finished at 17:24:05

Provisional grades
==================
Question q8: 1/1
------------------
Total: 1/1

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

透過定義好的 greedy strategy 去實作 chooseAction，Pacman assumes that each ghost is in its most likely position according to his beliefs, then moves toward the closest ghost.

首先對每個 ghost 的 positiondistribution 取 argmax() 保存再 ghostPos(list) 中，就能得到每個 ghost most likely position。

接下來是透過 pacmanposition 及 ghostposition 找到最近的 ghost 並記錄在 closet。

接下來是找到正確的方向前進，pacman 可以前進的方向紀錄在 legal 內，透過 Actions.getSuccessor 得到 pacman 選擇方向並前進後的 position，計算其與 closet 的距離並找出最近的位置，並將其 action 回傳。

Q9

```
Question q9
===========
*** q9) Particle filter initialization test: 0 inference errors.
*** PASS: test_cases\q9\1-ParticleInit.test
*** q9) numParticles initialization test: 0 inference errors.
*** PASS: test_cases\q9\2-ParticleInit.test

### Question q9: 1/1 ###


Finished at 17:24:42

Provisional grades
==================
Question q9: 1/1
----------------
Total: 1/1

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

實作 initializeUniformly 及 getBeliefDistribution，在這個案例中 particle is a ghost position。

首先是 initializeUniformly，這個部分是希望能夠 evenly 分配 particles 到每個 legalposition。我透過 self.numParticles 取得 particle 數量，並將其平均分配至各個 position，最後將 remain 的部分依 self.legalPositions 的順序分配給 legalPosition。

接下來是 getBeliefDistribution，這個部份是要將 particle 轉換成 DiscreteDistribution() 的形式。我將 particle 依照其 position 分配至 beliefdist[p]，在分配完後將 beliefdist normalize，並回傳 beliefdist。

Q10

```
Question q10
============
*** q10) Particle filter observe test: 0 inference errors.
*** PASS: test_cases\q10\1-ParticleUpdate.test
*** q10) Particle filter observe test: 0 inference errors.
*** PASS: test_cases\q10\2-ParticleUpdate.test
*** q10) Particle filter observe test: 0 inference errors.
*** PASS: test_cases\q10\3-ParticleUpdate.test
*** q10) Particle filter observe test: 0 inference errors.
*** PASS: test_cases\q10\4-ParticleUpdate.test
*** q10) successfully handled all weights = 0
*** PASS: test_cases\q10\5-ParticleUpdate.test
ParticleFilter
[Distancer]: Switching to maze distances
Average Score: 180.2
Scores:        188, 192, 198, 186, 167, 180, 184, 187, 164, 156
Win Rate:      10/10 (1.00)
Record:        Win, Win, Win, Win, Win, Win, Win, Win, Win, Win
*** Won 10 out of 10 games. Average score: 180.200000 ***
*** oneHunt) Games won on q10 with score above 100: 10/10
*** PASS: test_cases\q10\6-ParticleUpdate.test

### Question q10: 2/2 ###


Finished at 17:25:21

Provisional grades
==================
Question q10: 2/2
------------------
Total: 2/2

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

實作更新 particles by observation。

首先建立一個 dist(DiscreteDistribution())，設定初始 prob 為 1。

將 self.particles 中的每個 particle 計算出 ObservationProb 然後將其加至在 dist 中對應 particle 的欄位。如果 dist.total()==0 就執行 self.initializeUniformly(gameState)。如果不為零就更新 particles by dist，透過 dist.sample() 決定 particle 然後將其放入 self.particles。

```
Question q11
============
*** q11) Particle filter full test: 0 inference errors.
*** PASS: test_cases\q11\1-ParticlePredict.test
*** q11) Particle filter full test: 0 inference errors.
*** PASS: test_cases\q11\2-ParticlePredict.test
*** q11) Particle filter full test: 0 inference errors.
*** PASS: test_cases\q11\3-ParticlePredict.test
*** q11) Particle filter full test: 0 inference errors.
*** PASS: test_cases\q11\4-ParticlePredict.test
*** q11) Particle filter full test: 0 inference errors.
*** PASS: test_cases\q11\5-ParticlePredict.test
ParticleFilter
[Distancer]: Switching to maze distances
Average Score: 370.4
Scores:        380, 371, 366, 370, 365
Win Rate:      5/5 (1.00)
Record:        Win, Win, Win, Win, Win
*** Won 5 out of 5 games. Average score: 370.400000 ***
*** smallHunt) Games won on q11 with score above 300: 5/5
*** PASS: test_cases\q11\6-ParticlePredict.test

### Question q11: 2/2 ###


Finished at 17:27:23

Provisional grades
==================
Question q11: 2/2
------------------
Total: 2/2

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

基於 particle current state 及 pacman gamestate 決定 particle 下個 state。

將 self.particles 中的每個 particle 透過 self.getPositionDistribution(gameState, oldPos) 取得 positiondistribution，然後透過 distribution 進行 sample 在將新的 particle 放入 self.particles。