گزارش پروژه دوم درس هوش مصنوعی: محمدرضا صادقیان 9731121

بخش اول) عامل عکس العمل

تغییرات کد تابع EvaluationFunction:

با استفاده از خطوط زیر اطلاعات مورد نیاز مانند غذا ها و موقعیت روح ها را استخراج می کنیم و غذا ها را در یک لیست قرار می دهیم دو لیست خالی برای قرار دادن فاصله غذا ها از پکمن و فاصله روح ها از پکمن میسازیم:

```
FoodsList=newFood.asList()
GhostPosition=successorGameState.getGhostPositions()
FoodDistance=[]
GhostDistance=[]
```

فاصله منهتنی همه غذا ها و روح ها از محاسبه می شود و در لیست مورد نظرشان افزوده می شود :

```
for Food in FoodsList:
    FoodDistance.append(manhattanDistance(Food,newPos))
for Ghost in GhostPosition:
    GhostDistance.append(manhattanDistance(Ghost,newPos))
```

```
88          for distance in GhostDistance:
89              if distance<2:
90                  return (-(float("inf")))
91
92          if len(FoodDistance)==0:
93              return float("inf")
94
95          if currentGameState.getPacmanPosition()==newPos:
96              return(-(float("inf")))
97
98          return 10000/sum(FoodDistance) +100000/len(FoodsList)
```

- در صورتی که فاصله منهتنی هر یک از روح ها از پکمن خیلی نزدیک شود تابع منفی بینهایت را برمیگرداند که به معنی باخت قطعی است، پس پکمن از خیلی نزدیک شدن به پکمن پرهیز می کند. (خطوط 88 تا 90)
- در صورتی که هیچ غذایی باقی نماند تابع بینهایت را باز می گرداند که به معنی برد قطعی است. (خط 92)
- در صورتی که جایگاه فعلی پکمن برابر جایگاه جدید آن شود مقدار منفی بینهایت برگردانده می شود که به معنی باخت پکمن است. پس این شرط باعث می شود پکمن هیچوقت در جایگاه خود ثابت نماند (خط 95)
- در صورتی که هیچکدام از حالات بالا اتفاق نیفتد تابع مقداری متناسب با عکس مجموع فاصله غذاها و تعداد غذا های باقی مانده بر می گرداند که در ضریبی بزرگ ضرب شده است برای مقایسه بهتر اعداد است که در نهایت منجر به این می شود پکمن مسیری را انتخاب کند که کمترین فاصله تا غذا ها را دارد. ترم دوم به این دلیل بزرگ تر است که کم شدن تعداد غذاها برای ما ارجحیت دارد. (خط 98)

خروجی:

```
Question q1
===========

Pacman emerges victorious! Score: 1195
Pacman emerges victorious! Score: 1205
Pacman emerges victorious! Score: 1168
Pacman emerges victorious! Score: 821
Pacman emerges victorious! Score: 1058
Pacman emerges victorious! Score: 629
Pacman emerges victorious! Score: 1042
Pacman emerges victorious! Score: 1212
Pacman emerges victorious! Score: 1156
Pacman emerges victorious! Score: 892
Average Score: 1037.8
```

بخش دوم) Minimax

در تست های انجام شده برای عامل minimax یک من معمولا خورده می شود اما با این وجود امتیاز مثبت دارد و می توان این نتیجه را گرفت که عامل توانسته تا جای ممکن کار خود را انجام دهد.

برای پیاده سازی minimax اکثرا از توابع موجود در پروژه استفاده می کنیم. مانند گرفتن حرکت های مجاز یا فرزند گره فعلی در درخت پیمایش شده با استفاده از متد getLegalActions.

برای رعایت شروط گفته شده در تعریف پروژه تنها دوبار از generateSuccessor استفاده شده که در دو چرخه مجزا برای بررسی وضعیت های min و max و بدست آوردن حرکات قابل انجام برای عامل استفاده شده اند.

```
176
177             if agentIndex != 0:
178                 min = infinity
179                 for legalAction in legalActionList:
180                     successorGameState = gameState.generateSuccessor(agentIndex, legalAction)
181                     newMin = advrminimax(successorGameState, childAgentIndex, depth)[0]
182                     if newMin == min:
183                         if bool(random.getrandbits(1)):
184                             pref_action = legalAction
185 
186                     elif newMin < min:
187                         min = newMin
188                         pref_action = legalAction
189                 return min, pref_action
190 
191             else:
192                 max = neg_infinity
193                 for legalAction in legalActionList:
194                     successorGameState = gameState.generateSuccessor(agentIndex, legalAction)
195                     newMax = advrminimax(successorGameState, childAgentIndex, depth)[0]
196                     if newMax == max:
197                         if bool(random.getrandbits(1)):
198                             pref_action = legalAction
199 
200                     elif newMax > max:
201                         max = newMax
202                         pref_action = legalAction
```

همچنین اگر عمل قابل انجام از حدود مینی مکس کمتر یا بیشتر باشد از یک رندومایز برای انتخاب یک عمل موجود در محدوده استفاده می شود تا عامل در جای خود گیر نیفتد.

```
196                     if newMax == max:
197                         if bool(random.getrandbits(1)):
198                             pref_action = legalAction
```

خروجی:

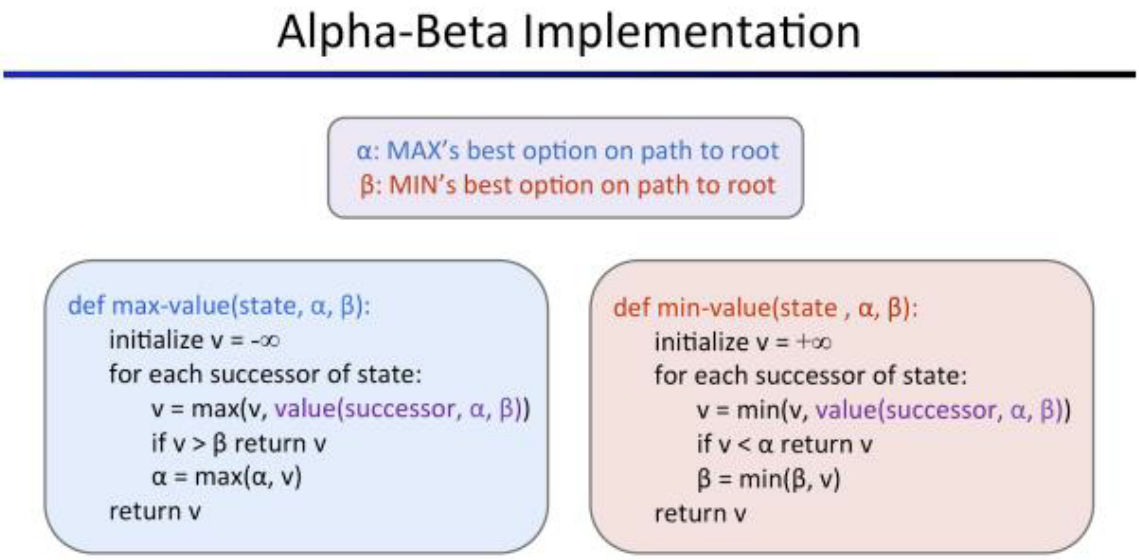
```
Pacman died! Score: 84
Average Score: 84.0
Scores:        84.0
Win Rate:      0/1 (0.00)
Record:        Loss
*** Finished running MinimaxAgent on smallClassic after 0 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases\q2\8-pacman-game.test

### Question q2: 5/5 ###


Finished at 17:19:43

Provisional grades
==================
Question q2: 5/5
------------------
Total: 5/5
```

بخش سوم) هرس آلفا-بتا

```
237                     elif new_max > max:
238                         max = new_max
239                         pref_action = legalAction
240                     if max > beta:
241                         return max, pref_action
242                     if max > alpha:
243                         alpha = max
```

به دو تابع Minimax تنها چهار خط کد بالا را اضافه می کنیم. کارکرد به این شکل است که alpha بزرگترین مقداری است که در شاخه های قبلی برای value-Max به دست آمده است. beta نیز کوچک ترین مقداری است که در شاخه های قبلی برای value-Min به دست آمده است. اگر در شاخه ای از value-Max بودیم که مقدار ارزیابی شده از beta بزرگ تر باشد، به دردمان نمی خورد زیر بیشترین مقدار آن شاخه از beta بزرگ تر خواهد بود، پس همیشه beta در value-Min پایینی پیروز می شود. همین حرف را می توان برای alpha زد.

از شبه کد زیر برای طراحی الگوریتم این بخش استفاده شده است.



خروجی:

```
*** Finished running AlphaBetaAgent on smallClassic after 0 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases\q3\8-pacman-game.test

### Question q3: 5/5 ###


Finished at 17:26:44

Provisional grades
==================
Question q3: 5/5
------------------
Total: 5/5
```

بخش چهارم) مینیماکس احتمالی

در این قسمت نیز مشابه minimax عمل می کنیم؛ با این تفاوت که:

1. زمانی که agent روح است، یک میانگین وزنی از تمام مقادیر value action ها می گیریم و آن را به عنوان مقدار value باز می گردانیم.(از آنجایی که در این مثال state هاي تمام خانه ها با یکدیگر یکسان است، وزن آنها یکسان بوده و برابر با یک تقسیم بر تعداد Action هاي مجاز است)
2. بخشی که pacman، Agent است دقیقا همانند minimax است.

نحوه ارزیابی سوال چهارم:
روش مینیماکس در موقعیتی که در دام قرار گرفته باشد خودش اقدام به باختن و پایان سریع تر بازی می کند ولی در صورت استفاده از مینیماکس احتمالی در ۵۰ درصد از موارد برنده می شود. این سناریو را با هر دو روش امتحان کنید و درستی این گزاره را نشان دهید.

- python pacman.py -p AlphaBetaAgent -l trappedClassic -a depth=3 -q -n 10

```
PS C:\Users\MrSadeghian\Desktop\multiagents> python pacman.py -p AlphaBetaAgent -l trappedClassic -a depth=3 -q -n 10
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Pacman died! Score: -501
Average Score: -501.0
Scores:        -501.0, -501.0, -501.0, -501.0, -501.0, -501.0, -501.0, -501.0, -501.0, -501.0
Win Rate:      0/10 (0.00)
Record:        Loss, Loss, Loss, Loss, Loss, Loss, Loss, Loss, Loss, Loss
```

Pacman با تصور اینکه هیچ شانسی برای برد ندارد، سریعتر به بازي پایان می دهد تا امتیازي کمتری از بدست دهد و هر 10 بار تست را با 501 - امتیاز می بازد.

- python pacman.py -p ExpectimaxAgent -l trappedClassic -a depth=3 -q -n 10

```
Pacman died! Score: -502
Pacman died! Score: -502
Pacman emerges victorious! Score: 531
Pacman emerges victorious! Score: 531
Pacman emerges victorious! Score: 531
Pacman emerges victorious! Score: 531
Pacman died! Score: -502
Pacman died! Score: -502
Pacman emerges victorious! Score: 531
Pacman died! Score: -502
Average Score: 14.5
Scores:        -502.0, -502.0, 531.0, 531.0, 531.0, 531.0, -502.0, -502.0, 531.0, -502.0
Win Rate:      5/10 (0.50)
Record:        Loss, Loss, Win, Win, Win, Win, Loss, Loss, Win, Loss
```

Pacman با امتحان کردن شانس خود، سریع خودکشی نمی کند و به همین طریق 6 بار بازی را میبرد و 4 بار می بازد و چون شانس خود را امتحان میکند و خودکشی نمی کند، در 4 بار باخت خود، با 502- به جای 501- امتیاز بازی را می بازد.

بخش پنجم) تابع ارزیابی

در این قسمت مجموع 4 معیار را به عنوان score نهایی باز می گردانیم.

1. Score که در state کنونی داریم.
2. فاصله (منهتن) تا نزدیکترین نقطه (غذا): مشابه بخش اول که داشتیم.
3. فاصله (منهتن) تا نزدیکترین روح و بررسی اینکه آیا در آن لحظه ترسیده است یا خیر:
در این قسمت این گونه عمل می کنیم که اگر روح ها در حالتی باشند که نترسیده باشند، یک مقدار (STEPS=10) را به ازاي هر روح از score کلی کم می کنیم.
اگر روح ها در حالتی باشند که ترسیده باشند، تعداد گام های در حال ترس را از آن عدد (STEPS=10) کم می کنیم و حاصل را با Score کلی جمع می کنیم. برای آنکه تاثیر مقدار عددهایی که کم و یا اضافه کردیم، بیشتر شود؛ قبل از آنکه آنها را در Score نهایی تاثیر دهیم به توان 2 می رسانیم.
4. فاصله (منهتن) تا نزدیکترین کپسول:
همانند غذا با آن رفتار می کنیم، فقط با این تفاوت که چون ارزش کپسول ها بیشتر از غذا برای ما است؛ یک وزن (50) نیز در مقدار معکوس فاصله آن ها ضرب می کنیم تا تاثیر آنها را بیشتر کنیم.

خروجی:

```
### Question q5: 6/6 ###


Finished at 17:49:50

Provisional grades
==================
Question q5: 6/6
------------------
Total: 6/6
```