기초인공지능

Assignment2 보고서

20171646 박태윤

(Minimax)

- 실행 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

다음과 같이 실행하여

텍스트이(가) 표시된 사진

자동 생성된 설명

이와 같은 승률을 얻었다.

- 알고리즘 설명

구현을 위해 cal\_minmax(gameState, agentIdx, depth)라는 함수를 따로 선언하였다. 현재 게임의 상태를 나타내는 gameState, 에이전트 번호(0번 : 팩맨, 1번~ : 고스트)를 나타내는 agentIdx, 미니맥스 트리의 깊이를 나타내는 depth가 인자로 들어간다. 처음에 Action에서 gameState.getLegalActions(0)으로 시작할 때 팩맨이 움직일 수 있는 경로에 대한 정보를 받아 canMove배열에 저장하고 이후 순서대로 이를 가지고 gameState.generateSuccessor(0, canMove[i])를 gameState로, agentIdx는 팩맨이 고스트에서 구한 min\_value를 통해 max\_value를 구해야 하므로 첫번째 고스트를 나타내는 1번, 와 depth는 0으로 하여 cal\_minmax함수를 호출한다. Action에서 최종적으로 움직일 수 있는 방향에 대한 모든 값들을 비교하여 가장 큰 값에 대한 움직임을 반환 받아야 하기 때문에 arr이라는 배열을 따로 선언하여 append해주었다.

cal\_minmax는 고스트에서 계산된 결과인 min\_value값을 통해 팩맨에서 max\_value값을 계산하는 등의 방식을 고려하여 재귀함수로 구현하였다. 크게 두 부분으로 나뉘는데, 가장 큰 value를 골라야 하는 팩맨(agentIdx == 0)과 가장 작은 value를 골라야 하는 고스트(agentIdx가 1보다 큼)로 나뉜다. cal\_minmax함수에서도 역시 현재 에이전트에 대한 움직임의 정보를 canMove로 받아 반복문을 돌면서 cal\_minmax를 호출하는데, 팩맨에서는 가장 큰 값을 골라야 하기 때문에 max\_value = max(max\_value, self.cal\_minmax(gameState.generateSuccessor(agentIdx, canMove[i]), 1, depth))로 가장 큰 값을 정제한 뒤 return max\_value해주었다. 앞선 함수의 호출에서도 볼 수 있듯이, 현재 팩맨을 보고 있으면 그 다음으로 agentIdx = 1인 첫번째 고스트를 호출한다. 다음으로 가장 작은 value를 골라야 하는 고스트에서는, 전체적인 동작 방식은 팩맨에서와 비슷하지만 마지막 팩맨에서는 재귀로 agentIdx = 0인 팩맨을 불러올 수 있도록 구현하였으며, 마지막 팩맨의 최소값까지 봤다는 것은 그 다음 minimax 알고리즘으로 값을 선택하려면 뎁스가 하나 더 증가하여야 하기 때문에 이 때에는 depth+1으로 cal\_minmax함수를 호출하였다. depth가 증가하면서 gamestate가 isLose 혹은 isWin이거나 depth == self.depth일 때 cal\_minmax가 종료되는 부분이 존재하는데, depth=4로 설정하여 명령어를 실행시켰다면 0,1,2,3 minimax 트리 깊이까지 탐색을 진행하고 4에서 해당 조건에 걸리게 된다. 마지막 고스트가 아닌 그 이외의 고스트들에서는 agentIdx만 하나씩 증가시키면서 재귀함수를 호출하였다. 가장 큰 값인 max\_value를 리턴하는 팩맨에서처럼 고스트에서는 가능한 동작 중 가장 작은 값을 나타내는 min\_value를 리턴하였다.

미니맥스 알고리즘을 지정된 depth만큼 돌면서 값이 모두 계산이 되어 다시 Action으로 돌아오면 최상단 팩맨 에이전트이므로 마지막으로 가장 큰 값을 골라 실제 동작을 return해주어야 한다. 이 때, 팩맨이 계속 한 자리에서 왔다 갔다 하는 문제를 발견하여 (왼쪽 = 100, 오른쪽 = 100 으로 arr값이 존재하면 계속 왼쪽으로만 감) 난수를 하나 생성하여 맥시멈 값을 고르는 정책을 난수가 짝수일 때는 arr[i] > maximum, 홀수일 때는 arr[i] >= maximum으로 해당 문제를 해결하였다.

(AlphaBeta)

텍스트이(가) 표시된 사진

자동 생성된 설명

미니맥스로 스몰맵을 실행시켰을 때의 결과입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

알파베타로 스몰맵을 실행시켰을 때의 결과입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

미니맥스로 미들맵을 실행시켰을 때의 결과입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

알파베타로 미들맵을 실행시켰을 때의 결과입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

미니맥스로 minimax맵을 실행시켰을 때의 결과입니다.(time\_check)

텍스트이(가) 표시된 사진

자동 생성된 설명

알파베타로 minimax맵을 실행시켰을 때의 결과입니다.(time\_check)

- 알고리즘 설명

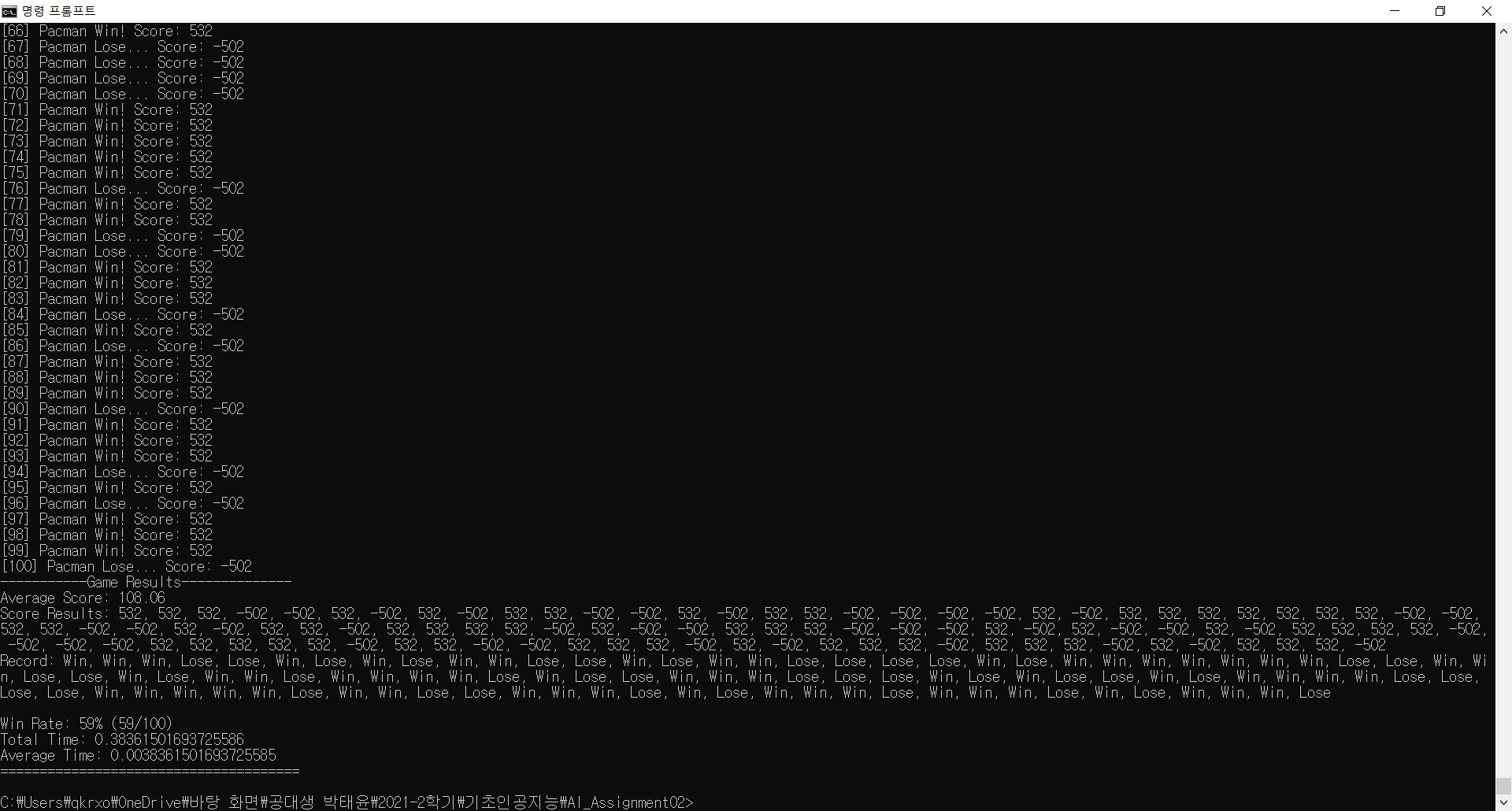
미니맥스 보다 더 효율적으로 동작하는 알파베타 에이전트를 구현한다. 기본 동작 방식은 미니맥스 에이전트와 같지만, cal\_minimax에 해당하는 함수인 cal\_alphabeta에서는 추가적인 인자로 alpha와 beta값을 받는다. 처음에 Action에서 이 함수를 호출할 때는 alpha에 매우 작은 값인 -999999, beta에는 매우 큰 값인 999999를 입력한다. cal\_minimax에서 agentIdx == 0인 팩맨에서 alpha값을 계속 현재 alpha값보다 max\_value값이 큰 경우 max\_value로 갱신을 해주는데, 만약 이 순간 beta보다 alpha가 커지는 경우 다른 움직임에 대해서 더 계산을 해봐야 다음 차례에서 min value를 고르는 고스트에 대해 의미가 없기 때문에 이 때는 max\_value를 더 계산하지 않고 바로 break해주었다(alpha-beta pruning). 고스트에서도 마찬가지로 작은 min\_value로 계속 갱신이 되는 beta가 alpha보다 커지는 순간은 더 계산을 해봐야 가장 큰 max\_value를 고르는 팩맨에서 의미가 없기 때문에 다른 움직임에 대해 더 계산을 하지 않고 바로 break를 해주었다(alpha-beta pruning).. 실행 결과에서 볼 수 있듯 같은 맵에 대해서 미니맥스에 비해 알파베타 알고리즘이 좀 더 적은 시간 결과를 출력하는 것을 확인할 수 있다.

(Expectimax)

텍스트이(가) 표시된 사진

자동 생성된 설명

다음과 같이 expectimax에 대한 명령어를 수행하였다.



59%의 승률을 보이는 것을 확인할 수 있었다.

- 알고리즘 설명

기존에 동작 중 value가 가장 작은 또는 가장 큰 값만을 select하여 action을 취하는 미니맥스나 알파베타와 달리 expectimax는 모든 동작의 value를 고려하여 action을 취한다. 재귀적인 방식을 취하는 것은 미니맥스, 알파베타와 비슷하지만 기존에 max\_value 또는 min\_value로 받았던 함수의 리턴 값을 이 알고리즘에서는 cal\_expect의 리턴값을 max\_tmp\_arr또는 min\_tmp\_arr에 모두 저장을 해주었다. 모든 동작에 대해 값을 받게 된다면 max\_tmp\_arr은 내림차순, min\_tmp\_arr은 오름차순으로 정렬을 시켰다. 각 배열에서는 0번부터 중요한 값을 나타내는데, 모든 값을 고려해야 하므로 예를 들어, max\_tmp\_arr의 길이가 2인 경우 즉 현재 state에서 agent가 갈 수 있는 방향이 2방향인 경우 max\_avg += 0.6 \* max\_tmp\_arr[0] + 0.4\*max\_tmp\_arr[1]로 가장 큰 값인 max\_tmp\_arr[0]에 좀 더 높은 가중치를 두어 계산을 한 뒤 이 max\_avg를 리턴하는 방식으로 구현하였다. 배열의 길이가 1인 경우는 굳이 가중치에 따라 계산을 할 필요 없이 0번 인덱스에만 값이 존재하는 경우이므로 max\_tmp\_arr[0] 또는 고스트에서는 min\_tmp\_arr[0]을 리턴하였다.

(추가)

minimax알고리즘 수행 시 각 depth에 따라 다르게 찍히는 initial value를 캡쳐했습니다.

코드상으로는 Action함수에서 리턴하기 전에 print(maximum)을 하여 이 값을 확인하였습니다.

- depth = 1

텍스트이(가) 표시된 사진

자동 생성된 설명

- depth = 2

텍스트이(가) 표시된 사진

자동 생성된 설명

- depth = 3

텍스트이(가) 표시된 사진

자동 생성된 설명

- depth = 4

텍스트이(가) 표시된 사진

자동 생성된 설명