COSE361 Artifical Intelligence
Final project 2/2

Sangmin Hwang [1]

1. Introduction

이번 Final project는 팩맨 간 대결 구도인 게임에서의 인공지능을 만드는 것입니다. 격자 모양의 맵 안에서 진행되며, 상대 진영의 음식을 자신의 진영 쪽으로 가져옴과 동시에 상대편이 자신 진영의 음식을 가져가지 못하도록 해야 하는 게임이었습니다. 본 프로젝트에서는 총 3가지 종류의 agent를 디자인하였으며, 기존 baseline과 각 agent 간의 대결을 통해 성능을 비교하였습니다.

첫 번째 agent는 AggressiveAgent로, 수비 없이 공격에만 치중한 Agent입니다. 두 agent가 거리를 두고 서로 다른 음식을 탐색합니다. 두 번째 agent는 PassiveAgent로, 공격 없이 수비에만 치중한 Agent입니다. 각자 하나의 상대 agent를 맡아 전담하여 수비하는 agent입니다. 마지막 agent는 HybridAgent입니다. 공격과 수비 모두 균형 잡혀 수행하는 agent이며, 현재 score에 따라 공격적으로 행동할지, 수비적으로 행동할지 선택하여 행동하는 agent입니다. 본 프로젝트에서는 마지막 agent를 중점적으로 성능을 평가하였습니다.

2. Methods

모든 Agent는 Reflex agent 기반으로 설계되었습니다. agent의 수가 많고 game의 state가 시시각각 변화하기 때문에 planning agent로는 효과적인 전략을 수립하기 힘들다고 판단하였기 때문입니다.

Reflex agent는 현재 game state(s)와 취할 수 있는 action(a)을 기반으로 utility score를 계산합니다(Q(s,a)). Utility score는 (s, a)에서 Feature vector를 추출하고, 추출한 Feature에 Feature Weight를 Linear combination 하는 방식으로 구하였습니다. 이때 각 agent마다 feature와 weight를 다양화하여 여러 전략을 수행하는 agent를 구현하였습니다.

2.1. Baseline 1

Baseline 1은 동일한 Aggressive agent 2개로 구성된 baseline입니다. 두 agent가 협력하여 공격할 때의 효과적인 공격 방식을 찾고, Best agent의 수비 성능을 평가하기 위하여 설계하였습니다.

2.1.1. AGGRESIVE AGENT

기본적으로 가장 가까운 음식을 찾아 움직이며, 음식을 하나라도 찾으면 아군 진영으로 돌아옵니다. 움직이는 과정에서 주변에 ghost가 있으면 ghost를 피해 움직이도록 합니다. 추가로, 팩맨인 상태에서 주변에 아군 팩맨이 있으면 그쪽 방향으로 가지 않도록 합니다. 최대한 두 Pacman이 떨어져서 음식을 찾도록 하여 Ghost에게 한꺼번에 잡힐 위험을 줄이도록 하는 전략입니다.

2.1.2. FEATURES

- **foodCount**: 상대 진영에 남아 있는 음식의 개수입니다.
- **distanceToFood**: 가장 가까운 음식까지의 거리입니다.
- **distanceToGhost**: 가장 가까운 유령까지의 거리의 역수입니다. 음식을 들고 있을 경우에는 유령을 더 조심하도록 하기 위해 10이 곱해집니다.
- **distanceToHome**: 가장 가까운 아군 진영의 좌표까지의 거리입니다. 음식을 먹고 나면 이 Feature를 활용하여 가장 가까운 아군 진영 쪽으로 돌아옵니다.
- **distanceToTeammate**: 두 agent가 모두 팩맨인 경우, 둘 사이의 거리입니다. 팩맨인 상태에서 서로 거리를 두고 다른 음식을 찾게 만들기 위해 설정한 Feature입니다.
- **stop**: 취하고자 하는 action이 Stop인 경우 1로 설정됩니다. 멈춰 있는 것을 방지하기 위해 설정하였습니다.
- **score**: 현재 score입니다. 음식을 가지고 돌아옴으로써 score를 늘리게 유도하기 위해 설정하였습니다.

2.1.3. WEIGHTS

클수록 좋은 값에는 양수 weight를, 작을수록 좋은 값에는 음수 weight를 설정하였습니다. 단, distanceToGhost는 이미 역수로 설정이 되어 있으므로 양수가 아닌 음수 weight를 설정하였습니다.

Feature	Weight
distanceToFood	-100
distanceToGhost	-70
distanceToHome	-10
distanceToTeammate	2
stop	-100
score	10000

Table 1. Baseline1 Feature Weights

Feature	Weight
numInvaders	-1000
onDefense	100
distanceToInvader	-10
distanceToMarkman	-10
stop	-100
reverse	-2

Table 2. Baseline2 Feature Weights

2.2. Baseline 2

Baseline 2은 동일한 Passive agent 2개로 구성된 baseline 입니다. 두 agent가 협력하여 수비할 때의 효과적인 수비 방식을 찾고, Best agent의 공격 성능을 평가하기 위하여 설계하였습니다.

2.2.1. PASSIVE AGENT

각 agent별로 하나씩 담당 상대 agent를 설정합니다(markman). 현재 아군 진영에 침입한 상대 팩맨이 없을 경우, markman과 가까운 거리를 유지하며 돌아다닙니다. 그러다가 침입한 상대 팩맨이 생기면 그 팩맨을 잡도록 유도합니다. 단, 만약 상대방이 Capsule을 먹어 자신이 scared인 상황이라면 상대방과 거리를 일정하게 유지하며 따라가도록 합니다. 기본적으로 Pacman은 주변에 ghost가 있으면 음식을 먹기 힘들다는 점을 이용, 각 ghost별로 하나씩 Pacman을 맡아 수비하도록 한 전략입니다.

2.2.2. FEATURES

- **numInvaders**: 아군 진영에 남아 있는 상대 팩맨의 개수입니다.
- **onDefense**: 자신이 ghost이면 1로 설정됩니다. 팩맨이 되지 않고 ghost로 수비하도록 유도하기 위한 Feature입니다.
- **distanceToInvader**: 가장 가까운 아군 진영의 상대 팩맨까지의 거리(d)입니다. 만약 자신이 scared인 상황이라면, $|d-5|$의 값으로 정의하여 일정 거리(5)를 유지하며 상대 팩맨을 따라가도록 합니다.
- **distanceToMarkman**: markman까지의 거리입니다. 아군 진영에 Invader가 없을 경우에만 이 Feature가 적용됩니다.
- **stop**: 취하고자 하는 action이 Stop인 경우 1로 설정됩니다. 멈춰 있는 것을 방지하기 위해 설정하였습니다.
- **reverse**: 취하고자 하는 action이 직전의 action과 반대되는 action, 즉 했던 것을 되돌아가는 action이면 1로 설정됩니다. 이전의 행동으로 되돌아가는 무의미한 행동을 방지하기 위해 설정하였습니다.

2.2.3. WEIGHTS

Baseline 1과 마찬가지로 클수록 좋은 값에는 양수 weight를, 작을수록 좋은 값에는 음수 weight를 설정하였습니다.

2.3. Baseline 3

Baseline 3은 동일한 Hybrid agent 2개로 구성된 baseline 입니다. Hybrid agent는 공격과 수비 모두 균형 있게 수행하는 agent로서, 현재 score에 따라 서로 다른 전략을 수립하여 플레이합니다.

2.3.1. HYBRID AGENT

먼저 2점차 이상으로 이기고 있지 않은 경우에는 Balanced 모드가 되어 공격 Agent와 수비 Agent로 나뉘어 플레이합니다. 공격 Agent는 Baseline 1에서 구현했던 Aggressive agent와 유사하게 행동하되, 공격을 혼자 수행하므로 아군 팩맨과의 거리를 고려하지 않고 행동하도록 설계하였습니다. 수비 Agent는 Baseline 2에서 구현한 Passive agent와 유사하게 행동하되, 수비를 혼자 수행하므로 markman을 지정하지 않고 맵의 중앙 주위에서 수비하도록 설계하였습니다.

2점 차 이상으로 이기고 있을 경우에는, Passive 모드로 전환하여 두 agent 모두 공격 없이 수비에만 전념합니다. 이때는 Baseline 2에서 구현하였던 Passive agent와 동일하게 행동합니다. 각자 markman을 설정하고 그 markman과 가까운 거리를 유지하면서, 아군 진영에 침입한 팩맨이 나타나면 쫓아가도록 하였습니다.

이렇게 설계한 이유는 게임의 구조 자체가 다득점이 어렵다고 판단했기 때문입니다. 상대방 agent가 baseline 정도 수준의 수비 능력만 되어도, 공격 agent를 아주 정밀하게 설계하지 않는 이상 2점 이상 점수를 득점하기 어려웠습니다. 따라서 agent를 설계할 때 2점차 이상의 득점만 내어도 전원 수비로 전환하여 게임을 굳히도록 하였습니다.

2.3.2. FEATURES FOR BALANCED MODE

Balanced mode에서는 첫 번째 agent는 공격만, 두 번째 agent는 수비만 수행합니다.

공격 agent가 사용하는 Feature는 Baseline 1의 Aggressive agent의 Feature에서 distanceToTeammate를 제외한 것입

Feature	Weight
distanceToFood	-100
distanceToGhost	-70
distanceToHome	-10
stop	-100
score	10000
numInvaders	-1000
onDefense	100
distanceToInvader	-10
distanceToCenter	-2
stop	-100
reverse	-2

Table 3. Balanced mode Feature Weights

니다. Weight 세팅은 동일합니다.

수비 agent가 사용하는 Feature는 Baseline 2의 Passive agent의 Feature에서 distanceToMarkman 대신 distanceToCenter를 추가한 것입니다. Weight 세팅은 동일합니다.

- **distanceToCenter**: 맵 중앙 좌표로부터의 거리입니다. 혼자서 2명을 방어해야 하므로 중앙 주변에서 수비하도록 설정하였습니다.

Passive mode에서는 두 agent 모두 공격 없이 수비에만 전념합니다. 사용한 Feature와 weight 모두 Baseline 2의 Passive agent와 동일합니다.

3. Results

Baseline 3을 기준으로 Baseline 1, Baseline 2, Baseline 3, 그리고 기존 baseline과 10번씩 대결한 결과입니다.

	your_best(red)
	<Average Winning Rate>
your_base1	0
your_base2	0.4
your_base3	0
baseline	1
Num_Win	2
Avg_Winning_Rate	0.35
	<Average Scores>
your_base1	0
your_base2	0.4
your_base3	0
baesline	2
Avg_Score	0.6

Figure 1. Simulation Results

전반적으로 우수한 결과를 내는 모습을 보였습니다. 기존 baseline의 경우에는 100%의 승률을 거두었으며, 2점차 이상일 때는 Passive agent로 전환하는 특성 때문에 모든 score가 2점으로 나오는 모습을 보였습니다. 2명 모두 방어만 하는 Baseline 2와의 경기에서도 좋은 승률을 거두었습니다. 공격 agent가 수비 agent를 잘 피해서 공격에 성공하는 것으로 보입니다.

다만, Baseline 1과 자기 자신인 Baseline 3와의 경기에서는 전부 무승부로 끝나는 모습을 보였습니다. 이는 Baseline 1과 Baseline 3의 설계 상의 특징 때문인데, 두 모델 모두 공격 agent가 초기에 가장 가까운 음식을 찾아 중앙 지점으로 가도록 설계가 되어 있습니다. 이 상황에서 공격 agent는 상대방과의 거리가 가까워 공격을 하지 않고, 수비 agent는 공격 agent가 근처에 있으므로 계속해서 그 주변을 맴돕니다. 공격도 수비도 할 수 없는 일종의 교착 상태에 빠지게 되어 모든 결과가 무승부로 나오게 되었습니다.

이러한 결과를 개선하기 위해서는 Hybrid agent의 공격 능력을 개선해야 할 필요가 있어 보입니다. 가장 간단한 개선점으로는 항상 가까운 Food를 찾는 것이 아니라, 확률에 따라 때로는 랜덤하게 다른 Food를 찾도록 하는 방법이 있을 수 있겠습니다.

4. Conclusion and Free discussion

본 프로젝트에서는 Aggressive agent, Passive agent, 그리고 둘의 특징을 종합한 Hybrid agent 총 3종류의 agent를 설계하고, Hybrid agent 기준으로 성능을 평가하였습니다. Hybrid agent는 Passive agent 및 기존 baseline과의 경기에서 좋은 성적을 거두었으며, Aggressive agent와의 경기와 Hybrid agent와의 경기에서는 모든 경기가 무승부로 끝나는 모습을 보였습니다.

이번 프로젝트를 진행하며 들었던 의문점은, adverserial한 게임이기 때문에 minimax 알고리즘을 사용하면 좋은 성적을 거둘 수 있지 않을까 하는 것이었습니다. 그러나 실제로 구현해보고 테스트해본 결과, 맵에서 고려해야 할 변수가 많고 취할 수 있는 action이 다양하기 때문에, minimax를 수행할 경우 좋은 결과를 얻기 위해서는 너무 큰 계산량이 소요된다는 것을 알게 되었습니다. 계산량을 고려한 agent 설계가 필수적이라는 것을 느끼게 된 프로젝트였습니다.