Minicontest2 Agent explanation

Anonymous Authors[1]

Abstract

Minicontest2에서 Pacman/Ghost Agent의 Actions를 정의하는 Baseline1, 2, 3, 그리고 이 중 가장 작동을 잘하는 Baseline 3을 Best로 선정하였습니다. Baseline 1은 기존 Baseline에 Features를 추가하여 구성했습니다. Baseline 2는 Baseline 1에 추가 feature을 넣어 구성했습니다. Baseline 3은 기존에 분리되어 있었던 Offensive, Defensive Agents를 통합하여 유기적으로 동작하도록 구성했습니다.

1. Methods

각 Baseline의 Method를 1, 2, 3의 순서대로 설명하겠습니다.

1.1. Baseline 1

ReflexCaptureAgent: registerInitialState, chooseAction, getSuccessor, evaluate, getFeatures, getWeights의 Methods가 있습니다.

registerInitialState: Agent들의 초기상태를 정해줍니다. 시작 좌표를 넣어주고, Offensive, Defensive를 구분하기 위한 boundary도 설정해줍니다. 또한, boundary중 벽이 없는 곳의 좌표들도 boundaries에 넣어줍니다.

chooseAction: 현재 상태에서 가능한 actions들 중, evaluate해서 나온 values에서 가장 큰 값의 actions를 bestActions에 넣습니다.또한, 남은 food가 2개 이하라면, 가장 가까운 음식을 향해 가도록 설정합니다.

getSuccessor: 현재 상태에서 해당 Agent, action을 이용해서 successor를 만들고 return합니다.

evaluate: 해당하는 features와 weght를 곱한 값을 return합니다.

getFeatures: features를 정의하고 그 array를 return합니다.

getWeights: weights를 정의하고 return합니다.

OffensiveReflexAgent: ReflexCaptureAgent를 입력으로 받으며 getFeatures, getWeights Methods를 override합니다.

getFeatures: Agent의 진영에 가까운 것에 가중치를 주는 distanceToTeam, 음식과의 거리로 가중치를 주는 distanceToFood, Ghost와의 거리로 가중치를 주는 distanceToGhost를 array로 return합니다.

getWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

DefensiveReflexAgent: ReflexCaptureAgent를 입력으로 받으며 getFeatures, getWeights Methods를 override합니다.

getFeatures: defense를 구분하는 onDefense, 자신의 팀에 있는 상대 Agent로 가중치를 주는 numInvaders, invader와 거리로 가중치를 invaderDistance 등을 return합니다.

getWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

1.2. Baseline 2

ReflexCaptureAgent: registerInitialState, chooseAction, getSuccessor, evaluate, getFeatures, getWeights의 Methods가 있습니다.

registerInitialState: Agent들의 초기상태를 정해줍니다. 시작 좌표를 넣어주고, Offensive, Defensive를 구분하기 위한 boundary도 설정해줍니다. 또한, boundary중 벽이 없는 곳의 좌표들도 boundaries에 넣어줍니다.

chooseAction: 현재 상태에서 가능한 actions들 중, evaluate해서 나온 values에서 가장 큰 값의 actions를 bestActions에 넣습니다.또한, 남은 food가 2개 이하라면, 가장 가까운 음식을 향해 가도록 설정합니다.

getSuccessor: 현재 상태에서 해당 Agent, action을 이용해서 successor를 만들고 return합니다.

evaluate: 해당하는 features와 weght를 곱한 값을 return합니다.

getFeatures: features를 정의하고 그 array를 return합니다.

getWeights: weights를 정의하고 return합니다.

OffensiveReflexAgent: ReflexCaptureAgent를 입력으로 받으며 getFeatures, getWeights Methods를 override합니다.

getFeatures: Agent의 진영에 가까운 것에 가중치를 주

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the Workshop on Theory of Mind in Communicating Agents (ToM). Do not distribute.

는 distanceToTeam, 음식과의 거리로 가중치를 주는 distanceToFood, Ghost와의 거리로 가중치를 주는 distanceToGhost를 array로 return합니다. 또한, 만약 Power Capsule을 상대편이 먹었다면 피하게 해줍니다.

getWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

DefensiveReflexAgent: ReflexCaptureAgent를 입력으로 받으며 getFeatures, getWeights Methods를 override합니다.

getFeatures: defense를 구분하는 onDefense, 자신의 팀에 있는 상대 Agent로 가중치를 주는 numInvaders, invader와 거리로 가중치를 invaderDistance 등을 return합니다.

getWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

1.3. Baseline 3

DummyAgent: ReflexCaptureAgent를 입력으로 받으며 getFeatures, getWeights Methods를 override합니다. Provided Baseline에 가장 점수를 높게 획득하는 Agent입니다.

registerInitialState: Agent들의 초기상태를 정해줍니다. 시작 좌표를 넣어주고, Offensive, Defensive를 구분하기 위한 boundary도 설정해줍니다. 또한, boundary중 벽이 없는 곳의 좌표들도 boundaries에 넣어줍니다.

chooseAction: 각 action마다 evaluate하고 가장 높은 값을 가지는 actions중 랜덤 선택해서 return합니다.

getSuccessor: 현재 상태에서 해당 Agent, action을 이용해서 successor를 만들고 return합니다.

evaluate: 해당하는 features와 weght를 곱한 값을 return합니다. 이때, 0번 Agent는 좀 더 공격적으로 공략하기 위해 자신의 food가 17개 남았을 때, 수비적으로 전환합니다. 2번 Agent는 방어적으로 19개 남았을 때, 수비적으로 전환합니다.

getOffensiveFeatures: 최대한 많이 움직이게 하기 위해 멈춰있을 때 가중치 'stop'을 설정합니다. 만약 food를 1개 초과로 먹었다면 집으로 돌아오게 하기 위해 'distanceToHome'을 설정합니다. 먹지 않았다면, 'leftFood', 'distanceToFood'를 설정해서 적절하게 feature를 할당합니다. 만약 상대 Agent가 있다면 회피하도록 'distanceToGhost'도 설정합니다.

getDefensiveFeatures: 'numInvaders' 자신의 진영에 상대가 있다면 수비적으로 되도록 설정합니다. 'invaderDistance'로 거리가 멀면 크게 할당합니다.'closeBoundary'로 경계선 쪽에 있어야 대응을 빨리 할 수 있으므로 선호하도록 설정합니다. 'closeGhost'로 가까이 있을 때를 선호하도록 설정합니다.

getOffensiveWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

getDefensiveWeights: 해당하는 features의 가중치 값을 array에 넣어 return합니다.

2. Results

Average Winning Rate

your_base1	0.4
your_base2	0.8
your_base3	0.0
baseline	1
Num_win	15
Average_Winning_Rate	0.55

Average Scores

your_base1	0
your_base2	2.2
your_base3	0.0
baseline	6.3
Num_win	2.125

Conclusion & Free Discussion

각 Agent들에게 세세하게 조건을 걸어줄 수록 좀 더 수월하게 작동하는 것 같습니다. 그리고 하나의 역할만 부여하는 것보다는 여러 역할을 유기적으로 설정하고, 각 action마다 어떻게 가중치를 부여하냐에 따라서 결과가 달라진다고 결론을 내릴 수 있습니다. 보완사항으로는, 나올 수 없는 길로 들어가서 필연적으로 상대에게 잡히는 상황을 회피해야 한다는 것과, food수로 역할을 전환하는 것 뿐만 아니라 Agent 해당 진영에 가까운 Agent가 우선적으로 Defensive하게 해야 점수 획득에 유리할 것이라고 생각합니다.