적대적 멀티 에이전트 환경에서 효율적인 강화 학습을 위한 정책 모델링

(Policy Modeling for Efficient Reinforcement Learning in Adversarial Multi-Agent Environments)

권 기 덕 † 김 인 철 ††
(Kiduk Kwon) (Incheol Kim)

요 약 멀티 에이전트 강화 학습에서 해결해야 할 중요한 문제는 자신의 작업 성능에 영향을 미칠 수 있는 다른 에이전트들이 존재하는 동적 환경에서 한 에이전트가 시행착오적 상호작용을 통해 어떻게 자신의 최적 행동 정책을 학습할 수 있느냐 하는 것이다. 멀티 에이전트 강화 학습을 위한 기존 연구들은 대부분 단일 에이전트 MDP 기반의 강화 학습기법들을 큰 변화 없이 그대로 적용하거나 비록 다른 에이전트에 관한 별도의 모델을 이용하더라도 다른 에이전트에 관해 요구되는 정보나 가정이 현실적이지 못하다는 한계점을 가지고 있다. 본 논문에서는 멀티 에이전트 강화 학습기술에 기초가 되는 기본 개념들을 정형화하고 이들을 기초로 기존 연구들의 특징과 한계점을 비교한다. 그리고 새로운 행동 정책 모델을 소개한 뒤, 이것을 이용한 강화 학습 방법을 설명한다. 본 논문에서 제안하는 멀티 에이전트 강화학습 방법은 상대 모델을 이용하는 기존의 멀티 에이전트 강화 학습 연구들에서 주로 시도되었던 상대 에이전트의 Q 평가 함수 모델 대신 상대 에이전트의 행동 정책 모델을 학습하며, 표현력은 풍부하나 학습에 시간과 노력이 많이 요구되는 유한 상태 오토마타나 마코프 체인과 같은 행동 정책 모델들에 비해 비교적 간단한 형태의 행동 정책 모델을 이용함으로써 학습의 효율성을 높였다. 또한, 본 논문에서는 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐게임을 소개하고, 이 게임을 테스베드삼아 비교 실험들을 수행하고 그 결과를 설명함으로써 본 논문에서 제안하는 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석해본다.

키워드 : 멀티 에이전트, 강화 학습, 정책 모델링, 상대 정책 모델

Abstract An important issue in multiagent reinforcement learning is how an agent should learn its optimal policy through trial-and-error interactions in a dynamic environment where there exist other agents able to influence its own performance. Most previous works for multiagent reinforcement learning tend to apply single-agent reinforcement learning techniques without any extensions or are based upon some unrealistic assumptions even though they build and use explicit models of other agents. In this paper, basic concepts that constitute the common foundation of multiagent reinforcement learning techniques are first formulated, and then, based on these concepts, previous works are compared in terms of characteristics and limitations. After that, a policy model of the opponent agent and a new multiagent reinforcement learning method using this model are introduced. Unlike previous works, the proposed multiagent reinforcement learning method utilize a policy model instead of the Q function model of the opponent agent. Moreover, this learning method can improve learning efficiency by using a simpler one than other richer but time-consuming policy models such as Finite State Machines(FSM) and Markov chains. In this paper, the Cat and Mouse game is introduced as an adversarial multiagent environment. And effectiveness of the proposed multiagent reinforcement learning method is analyzed through experiments using this game as testbed.

Key words : Multiagent System, Reinforcement Learning, Opponent Policy Model

· 이 논문은 2007 한국컴퓨터종합학술대회에서 '적대적 멀티 에이전트 환경에서 효율적인 강화 학습을 위한 정책 모델링'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 경기대학교 전자계산학과
kdkwon@kyonggi.ac.kr
†† 종신회원 : 경기대학교 전자계산학과 교수
kic@kyonggi.ac.kr

논문접수 : 2007년 9월 27일
심사완료 : 2008년 1월 30일

Copyright@2008 한국정보과학회 : 개인 목적이나 교육 목적인 경우, 이 저작물의 전체 또는 일부에 대한 복사본 혹은 디지털 사본의 제작을 허가합니다. 이 때, 사본은 상업적 수단으로 사용할 수 없으며 첫 페이지에 본 문구와 출처를 반드시 명시해야 합니다. 이 외의 목적으로 복제, 배포, 출판, 전송 등 모든 유형의 사용행위를 하는 경우에 대하여는 사전에 허가를 얻고 비용을 지불해야 합니다.
정보과학회논문지: 소프트웨어 및 응용 제35권 제3호(2008.3)

1. 서 론

멀티 에이전트 시스템에 강화 학습을 적용해보려는 멀티 에이전트 강화 학습(multiagent reinforcement learning)에 관한 연구가 최근 들어 관심을 모으고 있다. 멀티 에이전트 강화 학습에서 해결해야 할 중요한 문제는 자신의 작업 성능에 영향을 미칠 수 있는 다른 에이전트들이 존재하는 동적 환경에서 한 에이전트가 시행착오적 상호작용을 통해 어떻게 자신의 최적 행동 정책을 학습할 수 있느냐 하는 것이다. 멀티 에이전트 강화 학습이 적용될 수 있는 분야로는 로봇 축구[1], 컴퓨터 게임[2], 전자 상거래[3], 분산 웹 검색, 자가-관리 컴퓨터시스템, 반-테러 응용시스템 등 다양한 응용분야들이 존재한다. 전통적인 강화 학습방법들은 하나의 마코프 결정 문제(Markov Decision Problem, MDP)로 정의되는 단일 에이전트 환경에서 개발되어왔다. MDP에서 환경 상태 전이는 시간이 흘러도 변하지 않는 하나의 전이 확률 함수(transition probability function)로 정의된다. 그러나 에이전트들이 자율적으로 학습할 수 있는 멀티 에이전트 환경을 생각한다면, 각 에이전트의 행위는 학습이 진행됨에 따라 변화하게 된다. 따라서 학습이 가능한 다수의 에이전트들이 공존하는 멀티 에이전트 환경에서는 시간이 경과함에 따라 상태 전이 함수도 변할 수 있다. 즉, 그러한 환경은 하나의 MDP로 정의할 수 없다. 그럼에도 불구하고 그동안 많은 멀티 에이전트 강화 학습연구들에서는 MDP 기반의 강화 학습 기법들이 큰 변화 없이 그대로 적용되어 왔다[4].

그동안 다른 에이전트의 존재를 명시적으로 고려하는 멀티 에이전트 강화 학습 연구들도 있었다. 대표적인 연구들로는 두 명의 제로-합 게임을 위한 Littman의 Minimax-Q 학습 알고리즘[5], 두 명의 일반-합 게임을 위한 Hu와 Wellman의 Nash-Q 학습 알고리즘[6], Minimax-Q를 일반-합 게임으로 확장한 Littman의 FFQ 학습 알고리즘[7] 등이 있다. 이들은 대부분 적용 가능한 멀티 에이전트 시스템의 유형이 제한적이거나 다른 에이전트에 관해 요구되는 정보나 가정(assumption)이 비현실적이라는 한계점을 가지고 있다. 한편, 멀티 에이전트 환경에서 존재하는 다른 에이전트에 대한 다양한 모델들이 제안되기도 하였는데, Claus와 Boutilier의 JAL(Joint Action Learners)연구[8]에서는 Nash-Q에서와 같이 연합 행동에 대한 다른 에이전트의 Q 함수, 즉 평가 함수 모델을 학습하고 이것을 자신의 최적 정책을 학습하는데 이용하였다. 하지만 Carmel과 Markovitch의 연구[9]에서는 멀티 에이전트 시스템을 구성하는 각 에이전트의 행동 정책 모델을 하나의 결정적 유한 상태 오토마타(Deterministic Finite Automata, DFA)로 표현하고 이것을 학습하였다. 즉, 이 연구에서는 하나의 DFA로 표현된 상대 에이전트의 행동 정책 모델을 학습하고 이것을 기초로 역시 하나의 DFA로 표현된 자신의 행동 정책을 결정하는 방법을 제시하였다. Riley와 Veloso의 연구[10]에서는 다른 에이전트들의 평가 함수 모델이나 행동 정책 모델을 직접 학습하지 않고 대신 환경 상태 전이들을 기록하여 두었다가 이들로부터 하나의 마코프 체인(Markov Chain)을 추출하고, 여기에 가능한 행동들을 붙여 하나의 마코프 결정 문제(MDP)를 만들었다. 그리고 이 마코프 결정 문제를 풀어 자신의 로봇 축구팀을 위한 코치 조언(coach advice)을 만들어 내었다. 이와 같은 연구들은 종래의 연구들과는 달리 좀 더 풍부한 상대 모델들을 이용한다는 특징을 보이나 이러한 상대 모델을 직접 강화 학습과 연계하여 이용하려는 노력을 보여주지는 않았다.

본 논문에서는 과거에 관찰된 상대 에이전트의 행동들을 기초로 상대 에이전트의 행동 정책 모델을 학습하고, 이 모델을 바탕으로 다시 자신의 최적 정책을 학습하는 강화 학습방법을 제시한다. 이 멀티 에이전트 강화 학습방법은 두 명의 에이전트로 구성된 적대적 멀티 에이전트 환경을 가정하고 있으며, 두 에이전트는 동시에 행동을 수행함으로써 자신의 행동을 결정하기 전에 미리 상대 에이전트의 행동을 알 수는 없으나 일단 동시에 행동을 수행하고 나면 상대 에이전트가 수행한 행동을 관찰할 수 있다. 하지만 두 에이전트 간에는 행동 결정에 영향을 미치는 어떠한 통신도 가능하지 않다고 가정한다. Q 학습 알고리즘을 확장한 이 멀티 에이전트 강화학습 방법은 상대 모델을 이용하는 기존의 멀티 에이전트 강화 학습 연구들에서 주로 시도되었던 상대 에이전트의 Q 평가 함수 모델 대신 상대 에이전트의 행동 정책 모델을 학습하며, 표현력은 풍부하나 학습에 시간과 노력이 많이 요구되는 유한 상태 오토마타(DFA)나 마코프 체인(Markov Chain)과 같은 행동 정책 모델들에 비해 비교적 간단한 형태의 행동 정책 모델을 이용함으로써 학습의 효율성을 높였다. 본 논문에서는 멀티 에이전트 강화 학습기술에 기초가 되는 기본 개념들을 정형화하고 이들을 기초로 기존 연구들의 특징과 한계점을 비교한다. 그리고 새로운 행동 정책 모델을 소개한 뒤, 이것을 이용한 강화 학습 절차를 설명한다. 또한, 본 논문에서는 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐(Cat and Mouse) 게임을 소개하고, 이 게임을 테스베드삼아 수행한 비교 실험 결과들을 설명함으로써 본 논문에서 제안하는 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석해본다.

2. 멀티 에이전트 강화 학습

단일 에이전트 환경에서 에이전트가 자신의 행동에 대한 보상 값(reward)을 기초로 스스로 최적의 행동 정책을 학습해나가는 과정은 하나의 마코프 결정 문제(MDP)[11]로 표현할 수 있다.

정의 1. (마코프 결정 문제) : 하나의 마코프 결정 문제(Markov Decision Problem, MDP)는 튜플 $<S, A, T, R>$로 정의한다.

- S는 환경 상태들의 유한집합.
- A는 에이전트가 선택할 수 있는 행동들의 집합.
- T: $S \times A \rightarrow \Pi(S)$는 에이전트의 행동에 따라 다음 상태를 결정하는 상태전이함수.
- R: $S \times A \rightarrow R$은 에이전트의 행동에 대한 보상 값을 결정하는 보상함수.

Π(S)는 상태 집합 S의 멱집합(power set)을 나타내며, 의미는 특정 상태 s∈S에서 특정 동작 a∈A의 실행에 의한 상태 전이 결과가 단 하나의 상태 s′로 결정되는 경우뿐 만 아니라 결과 상태가 {s′, s″} 중 하나로 결정되는 비 결정적(non-deterministic) 상태전이를 나타내기 위함이다. 따라서 이 때 상태전이함수는 상태 s를 상태들의 부분집합인 {s′, s″}으로 대응시키는 함수로 볼 수 있다. 이와 같은 이유로 Π(S)는 상태들의 부분집합들의 집합 즉, 멱집합으로 표현하였다. 비슷한 이유에서, 에이전트의 비결정적 행동선택 함수를 나타내기 위해 행동집합 A의 멱집합을 의미하는 Π(A)를 사용하였다. 즉, 특정 상태 s∈S에서 선택되어질 동작들이 {a, a′}과 같이 여러 개 존재할 때 고려하여 동작들의 부분집합들의 집합, 멱집합으로 표현하였다.

정의 2. (정책) : 하나의 정책(policy) π: S->Π(A)는 에이전트의 행동 선택 함수를 나타내며, 임의의 상태에 대해 에이전트가 선택할 수 있는 행동들을 확률분포로 표현한다. 각 상태마다 확률 값 1을 갖는 특정 행동이 하나씩 정해져 있는 정책을 결정적 정책(deterministic policy)이라고 하고, 그렇지 않은 정책을 비결정적 정책(non-deterministic policy)이라고 한다.

마코프 결정 문제에서 에이전트의 목표는 환경과의 상호작용을 통해 보상 값의 합이 최대가 되는 최적 정책(optimal policy)을 학습하는 것이다.

정의 3. (Q-함수) : Q-함수 Q^{π}(s, a)는 에이전트가 상태 s에서 행동 a를 수행하고, 그 이후에는 정책π에 따라 행동하였을 때 기대되는 보상 값의 합을 나타낸다. 따라서 함수 Q^{π}(s, a)는 다음과 같이 정의한다.

$$Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \times \sum_{a' \in A} \pi(s',a') Q^{\pi}(s',a') \quad (1)$$

$\pi(s',a')$는 행동정책(policy) π을 하나의 확률분포로 나타낸다고 가정한 것으로 정책 π에 따라 다음 상태 s'에서 행동 a'을 선택할 확률을 나타낸다.

Q-함수 Q^{*}(s, a)는 상태 s에서 출발하여 모든 상태에서 최적의 정책인 π^{*}에 따라 행동하였을 때, 기대되는 보상 값의 합을 나타내며, 다음과 같이 정의한다.

$$Q^{*}(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^{*}(s) \quad (2)$$

이때 $V^{*}(s) = \max_{a' \in A} Q^{*}(s',a')$

정의 4. (그리디 정책) : 각 상태 s에서 Q-함수 Q^{*}(s, a)의 값이 최대인 행동만을 선택하는 정책 π을 그리디 정책(greedy policy)라고 한다. 즉, 그리디 정책은 각 상태 s에 대해 Q-함수 Q^{*}(s, a)의 값이 최대인 행동에게만 확률 값 1을 배정하는 정책이다.

Q 학습은 각 상태와 행동의 쌍에 대한 Q 함수 값을 학습하는 강화학습의 하나이다. Q 학습은 환경에 대한 사전 모델이 필요하지 않은 강화학습이며, 시간 차 학습 방법(temporal difference learning)의 하나이다. Q 학습은 식 (3)에 따라 Q_t(s, a) 함수 값을 반복 갱신함으로써, 최적의 Q^{*}(s, a) 함수 값을 추정하는 과정으로 볼 수 있다.

$$Q_t(s,a) = (1-\alpha_t) Q_{t-1}(s,a) + \alpha_t [r_t + \gamma \max_{a' \in A} Q_{t-1}(s',a')] \quad (3)$$

이때 r_t는 보상값을,
γ는 감퇴율(*discount rate*)를 나타낸다.

학습 율은 일반적으로 0에서 1의 값 중 0에 가까운 값을 주면 탐험(exploration)을 많이 하게 되어 이전의 Q 값에 의한 행동을 많이 하게 되고, 1의 값에 가까운 값을 주면 탐색(exploitation)을 많이 하도록 하여 경험해 보지 않은 행동을 많이 수행하게 된다.

한편, 다수의 에이전트들이 공존하며 서로 상호작용하는 멀티 에이전트 환경은 하나의 확률 게임으로 표현할 수 있다.

정의 5. (확률 게임) : 하나의 확률 게임(stochastic game, SG)은 튜플 $<N, S, \vec{A}, T, \vec{R}>$로 정의한다.

- N은 에이전트들의 수
- S는 게임 상태들의 유한집합.
- $\vec{A} = A_1, \ldots, A_n$, 이때 각 A_i는 에이전트 i가 선택할 수 있는 행동들의 집합.
- T: $S \times \vec{A} \rightarrow \Pi(S)$는 에이전트들의 행동에 따라 다음 상태를 결정하는 상태전이함수.
- $\vec{R} = R_1, R_2, \ldots, R_n$ 이때 각 $R_i : S \times \vec{A} \rightarrow R$는 에이전트 i의 보상 값을 결정하는 보상함수.

확률 게임에 참여하는 각 에이전트는 동시에 각자의 행동을 선택하며, 이로 인해 각 에이전트가 받게 되는 보상 값과 다음 게임 상태는 현재의 게임 상태와 참여 에

이전트 모두에 의한 연합 행동(joint action)에 따라 결정된다. 이러한 확률 게임은 앞서 소개한 마코프 결정 문제(MDP)를 행동 결정권자인 에이전트가 다수 참여하는 멀티 에이전트 환경으로 일반화한 것이다. 따라서 확률 게임에 참여하는 각 에이전트의 목표는 자신의 보상 값과 게임 상태 전이에 영향을 주는 다른 에이전트들의 존재를 고려하면서 자신의 최적 정책을 학습하는 것이다.

정의 6. (제로-합 확률 게임) : 참여하는 에이전트들의 이득(gain)의 총합이 손실(loss)의 총합과 같은 확률 게임을 제로-합 확률 게임(zero-sum SG)이라고 한다.

제로-합 게임에서는 한 에이전트가 얻은 이득은 곧 다른 에이전트의 손실을 의미한다. 그러나 반드시 그렇지 않은 게임을 일반-합 확률 게임(general-sum SG)이라고 한다.

정의 7. (Nash 평형) : Nash 평형(Nash equilibrium)은 확률 게임에 참여하는 각 에이전트의 정책이 다른 모든 에이전트들의 정책에 대해 최적인 경우를 말한다. 그리고 이러한 정책들의 집합을 Nash 평형 정책들이라고 한다.

확률 게임에 참여하는 각 에이전트의 정책이 일단 Nash 평형을 이루고 나면, 어떤 에이전트도 자신의 정책을 일방적으로 변경하더라도 더 나은 이득을 얻을 수 없다.

멀티 에이전트 환경인 확률 게임에 참여하는 에이전트의 강화학습은 단일 에이전트 환경인 마코프 결정 문제(MDP)와는 달리, 게임에 참여하는 다른 에이전트들에 대한 고려를 자신의 정책 학습과정에 어떻게 반영하느냐가 중요한 문제가 된다. 따라서 멀티 에이전트 강화학습을 위한 기존의 연구들은 이 문제에 대한 나름의 해법을 제시한 것으로 볼 수 있다. 표 1은 멀티 에이전트 강화학습을 위한 대표적인 기존 알고리즘들을 두 가지 기준점을 중심으로 분류해 본 결과를 나타내고 있다. 첫 번째 기준점은 확률 게임의 유형으로서, 이 기준점에 따라 하나의 확률 게임, 즉 멀티 에이전트시스템을 제로-합(zero-sum) 게임 또는 일반-합(general-sum) 게임으로 분류한다. 일반적으로 제로-합 게임은 참여하는 에이전트들 간의 적대관계나 경쟁관계가 존재하는 게임을 나타내며, 일반-합 게임은 공동의 목표를 위해 참여 에이전트들의 일부나 전부가 협력하는 게임을 나타낸다. 따라서 확률 게임의 유형은 곧 적대적 멀티 에이전트시스템과 협력적 멀티 에이전트시스템을 구분하는 기준이 된다. 두 번째 기준점은 상대모델로서, 학습을 위해 다른 에이전트의 가치함수(value function)나 정책(policy)에 대한 명시적인 모델을 이용하느냐 그렇지 않느냐로 기존의 멀티 에이전트 강화학습 알고리즘들을 분류한다. 비록 명시적인 상대모델을 이용하지 않는 것으로 분류

표 1 멀티 에이전트 강화학습 알고리즘들의 분류

분류체계		게임 유형	
		제로-합 게임	일반-합 게임
상대 모델	암시적	Minimax-Q	FFQ
	명시적	Fictitious play	Nash-Q

된 멀티 에이전트 강화학습 알고리즘들이라 하더라도 사전에 다른 에이전트들의 정책이나 행동양식에 대한 특별한 가정(assumption)을 기초로 학습을 진행하는 경우가 대부분이다.

Littman이 제안한 Minimax-Q 알고리즘[5]은 두 명의 에이전트로 구성된 제로-합 게임을 위한 강화학습 알고리즘이다. Minimax-Q 학습에서 학습 에이전트는 상대가 선택할 수 있는 최악의 행동을 감안하여 자신이 얻을 수 있는 기대 보상 값을 최대화할 수 있는 최적 정책을 학습한다. Minimax-Q 학습에서 가치함수 V는 [식 4]와 같이 Q 함수 값들의 최소최대(minimax)로 갱신된다.

$$V_1(s) = \max_{\pi_1(s,-)\in\Pi(A_1)} \min_{a_2\in A_2} \sum_{a_1\in A_1} \pi_1(s,a_1) Q_1(s,(a_1,a_2)) \quad (4)$$

여기에서 V와 S의 아래 첨자의 의미는 두 명의 제로-합 확률 게임을 가정하고 있기 때문에, 1은 에이전트 자신을 의미하고, 2는 상대 에이전트를 의미한다.

Minimax-Q 학습알고리즘은 자신이 받은 보상 값을 토대로 두 에이전트의 연합 행동(joint action)에 대한 Q-함수 $Q_1(s,(a_1,a_2))$을 학습하지만 특별히 상대 에이전트에 대한 별도의 명시적인 모델을 요구하지는 않는다. 대신 이 알고리즘은 상대 에이전트가 언제나 최악의 행동선택을 할 것이라는 가정을 토대로 하고 있으며, 두 명의 제로-합 게임에만 적용 가능하다는 제한점이 있다.

Hu와 Wellman이 제안한 Nash-Q 학습 알고리즘[6]은 제로-합 게임에 적용 가능한 Minimax-Q 학습 알고리즘을 협력 에이전트들이 존재하는 일반-합 게임에 적용 가능 하도록 확장한 알고리즘이다. Nash-Q 학습에서는 Nash 평형 정책을 얻기 위해 각 상태에 대한 Nash 평형을 찾아야 한다. 각 에이전트 i의 Q 함수 값은 식 (5)와 같이 갱신된다.

$$Q_i(s,a_1,...,a_n) = (1-\alpha) Q_i(s,a_1,...,a_n) + \alpha[r_i + \gamma Nash Q_i(s')] \quad (5)$$

이 때 $Nash Q_i(s') = \pi_1(s')...\pi_n(s') Q_i(s')$

s'은 게임에 참여하는 다수의 에이전트들이 상호작용에 의해 도달 가능한 하나의 Nash 평형상태 중의 하나를 나타내며, $Q_i(s')$는 이러한 Nash 평형상태 s'에서 각 에이전트들이 각기 자신의 정책에 따라 동작 $\pi(s')$

을 수행한다고 가정했을 때 이에 대한 i번째 에이전트가 판단하는 동작의 가치함수를 의미한다.

Nash-Q 학습 알고리즘은 Minimax-Q 학습 알고리즘과는 달리 일반-합 게임에 적용 가능하다는 장점은 있으나, 학습 에이전트 자신뿐만 아니라 다른 에이전트들의 Q 함수 값까지 유지하기 위해 다른 에이전트들의 보상 값과 학습 파라미터들(learning parameters)까지 알아야 한다는 제한점이 있다.

FFQ(Friend-or-Foe Q) 학습알고리즘[7] 역시 Minimax-Q 알고리즘을 기초로 일반-합 게임에도 적용 가능 하도록 확장한 알고리즘이다. FFQ 학습에서는 게임에 참여하고 있는 각 에이전트를 친구(friend)와 적(foe)로 구분하여, 각각 달리 Nash-Q 함수 값을 계산한다. 두 명의 게임에서 상대가 친구인 경우에는 식 (6)과 같이 계산하고, 상대가 적인 경우에는 Minimax-Q의 경우와 마찬가지로 식 (7)과 같이 계산한다.

$$Nash_1(s,Q_1,Q_2)=\max_{a_1\in A_1,a_2\in A_2} Q_1(s,a_1,a_2) \quad (6)$$

$$Nash_1(s,Q_1,Q_2)=\max_{\pi_1(s,-)\in \Pi(A_1)} \min_{a_2\in A_2}\sum_{a_1\in A_1}\pi_1(s,a_1)Q_1(s,a_1,a_2) \quad (7)$$

FFQ 학습 알고리즘은 게임에 참여하는 각 에이전트가 친구인지 적인지 미리 알아야 하며, Minimax-Q와 동일한 가정을 기초로 다른 에이전트들의 Q 함수 값을 예측한다는 제한점이 있다.

Fictitious Play 알고리즘[12]은 제로-합 게임을 위한 강화학습 알고리즘이다. 이 알고리즘은 Nash 평형과 Nash-Q 함수 값에 의존하는 다른 알고리즘과는 달리, 다른 에이전트들의 정적인 정책모델을 이용한다. Fictitious Play 알고리즘에서 다른 에이전트들의 정책에 대한 믿음은 과거 플레이의 실험적 분포에 따라 표현된다. Fictitious Play 알고리즘에서는 Nash 평형을 찾아야 할 필요는 없으나, 학습 에이전트가 다른 에이전트들을 모델링하여야 하고 학습의 수렴성이 휴우리스틱 규칙에 매우 의존적이라는 제한점이 있다.

3. 상대방 정책 모델 기반의 강화 학습

3.1 상대방 정책 모델

본 논문에서는 Q 학습을 확장하여 적대적 멀티 에이전트 환경에 적합한 멀티 에이전트 강화학습 방법을 제시한다. 특히 본 논문에서는 관찰되는 상대 에이전트의 행동을 바탕으로 상대 에이전트의 행동선택함수인 상대방 정책 모델을 점진적으로 학습하고, 이 모델을 기초로 자신의 최적 정책을 학습하는 멀티 에이전트 강화학습 방법을 제시한다. 본 논문에서 가정하는 적대적 멀티 에이전트 환경은 아래에 정의하는 바와 같은 두 명의 제로-합 확률 게임으로서, 적대관계의 두 에이전트가 동시에 각각 자신의 행동을 수행하는 동기화된 환경이다. 그리고 두 에이전트는 서로 관찰을 통해 상대방이 수행하는 행동을 알 수 있으나, 서로 간에는 어떤 통신도 없다고 가정한다.

정의 8. (두 명의 제로-합 확률 게임) : 두 명의 제로-합 확률 게임(two-player zero-sum SG)은 튜플 $<S, \vec{A}, T, \vec{R}>$로 정의한다.

- $S=\{s|s=(s_{self}, s_{opponent})\}$는 게임 상태들의 유한집합.
- $\vec{A}=\{(a_{self}, a_{opponent})|a_{self}\in A_{self}, a_{opponent}\in A_{opponent}\}$는 연합 행동(joint action)들의 집합, 이때 각각 A_{self}와 $A_{opponent}$는 에이전트 *Self*와 에이전트 *Opponent*가 선택할 수 있는 행동들의 집합.
- $T: S\ X\ \vec{A} \rightarrow \Pi(S)$는 에이전트 *Self*와 에이전트 *Opponent*의 연합 행동에 따라 다음 상태를 결정하는 상태전이함수.
- $\vec{R}=R_{self}, R_{opponent}$는 에이전트 *Self*와 에이전트 *Opponent*의 보상함수, 이때 $R_{self}: S\times\vec{A}\rightarrow R$는 에이전트 *Self*의 보상 값을, $R_{opponent}: S\times\vec{A}\rightarrow R$는 에이전트 *Opponent*의 보상 값을 결정하는 보상함수.

정의 9. (상대방 정책 모델): 과거에 수행된 상대 에이전트의 행동을 관찰함으로써 얻는 상대 에이전트의 행동 추정 함수 $PM: S X A_{opponent}\rightarrow(0,1)$을 상대방 정책 모델(opponent's policy model)이라고 한다.

상대방 정책 모델 $PM(s, a_{opponent})$은 상태 s에서 상대 에이전트가 행동$a_{opponent}$을 수행할 가능성을 0과 1사이의 값으로 추정한 것이다. 그리고 이러한 상대방 정책 모델 $PM(s, a_{opponent})$은 상대 에이전트의 실제 행동을 관찰함에 따라 적응적으로 조정된다. 즉, 상태 s에서 상대 에이전트가 행동 $a^*_{opponent}$을 수행하는 것을 관찰하면, 상태 s에서 수행 가능한 모든 행동 $a_{opponent}$에 대한 상대방 정책 모델 $PM(s, a_{opponent})$은 식 (8)과 같이 갱신된다.

$$PM(s,a_{opponent})=(1-\theta)PM(s,a_{opponent})+\begin{cases}\theta\,(a_{opponent}=a^*_{opponent})\\ 0\,(otherwise)\end{cases} \quad (8)$$

이때, $\theta(0\le\theta<1)$는 실제 상대 에이전트가 수행한 행동인 $a^*_{opponent}$의 효과를 조절하는 파라미터(parameter)이다. 식 (8)에 따르면, 임의의 한 상태 s에서 자주 반복 관찰되는 상대 에이전트의 행동에 대해서는 PM 함수 값이 증가하고, 그렇지 못한 행동에 대해서는 PM 함수 값이 감소한다. 그리고 PM 함수 값의 증감폭은 파라미터 Θ에 의해 결정된다.

3.2 강화 학습 과정

여기서는 앞서 소개한 상대 에이전트의 정책 모델을 기초로 자신의 최적 정책을 학습하는 멀티 에이전트 Q 학습 방법을 설명한다. 상대방 정책 모델을 이용한 멀티 에이전트 Q 학습은 다음과 같은 절차대로 진행된다.

단계 1: 에이전트는 현재 상태 s와 상대방 정책 모델 $PM(s, a_{opponent})$을 기초로 상대 에이전트가 수행할 행동 $a_{opponent}$들을 예측한다. 그리고 식 (9)와 같이 계산되는 자신의 확률 정책에 따라 자신이 취할 행동 a_{self}을 결정한다.

$$\pi[a_i|s] = \frac{e^{\overline{Q}(s,a_i)/\tau}}{\sum_{a_j \in A_{self}} e^{\overline{Q}(s,a_j)/\tau}} \quad (9)$$

이때, τ는 행동 선택의 임의성(random)을 제어하는 변수이고, τ가 0에 가까울 경우에는 알려진 것 중 가장 Q값이 큰 행동이 선택되게 되고 τ가 클 경우에는 임의의 선택에 가깝게 행동이 선택된다.

$$\overline{Q}(s, a_{self}) \equiv \sum_{a_{opponent} \in A_{opponent}} PM(s, a_{opponent}) Q(s, a_{self}, a_{opponent})$$

는 상대방 정책 모델을 토대로 계산된 자신의 행동 a_{self}에 대한 Q-함수 기대 값이다.

단계 2: 에이전트는 단계 1에서 선택된 자신의 행동 a_{self}을 실행한다. 이와 동시에 상대 에이전트도 행동 $a^*_{opponent}$을 선택하고 실행한다. 그 결과, 환경은 새로운 상태 s'로 변경되고 에이전트는 환경으로부터 보상 값 r을 받는다. 그러면 에이전트는 식 (8)에 따라 상대방 정책 모델을 갱신하고, 또 식 (10)과 같이 Q 함수 값도 갱신한다.

$$Q(s, a_{self}, a^*_{opponent}) \leftarrow (1-\alpha) Q(s, a_{self}, a^*_{opponent}) + \alpha(r + \gamma \max_{a'_{self} \in A_{self}} Q(s', a'_{self}, a'_{opponent})) \quad (10)$$

여기서

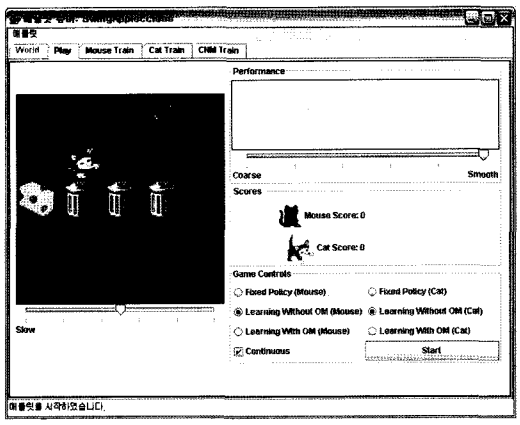
$a'_{opponent} = \arg max_{a'_{opponent} \in A_{opponent}} PM(s', a'_{opponent})$이다.

단계 3: 만약 새로운 상태 s'가 종결 조건을 만족하면, 게임종료와 더불어 학습을 위한 한 번의 에피소드가 끝나게 된다. 그렇지 않으면 $s' \rightarrow s$로 상태를 변경한 다음, 단계 1부터 다시 반복한다.

4. 실험 및 분석

4.1 고양이와 쥐게임

고양이와 쥐게임(Cat and Mouse game)은 멀티 에이전트 연구에 많이 이용되어 온 추적 게임(pursuit game)의 한 변형이다. 이 게임은 전형적인 제로-합 확률 게임으로서, 적대관계인 두 명의 에이전트가 서로 생사를 걸고 경쟁을 벌이는 멀티 에이전트 환경이다. 이 게임에서 고양이와 쥐는 서로 상반된 목표를 가지고 있다. 고양이는 쥐를 잡아 점수를 획득하는 것이 목표이며, 반면에 쥐는 고양이에게 잡히지 않으면서 많은 치즈를 먹어 점수를 높이는 것이 목표이다. 임의의 위치에서 시작하여 도망가는 쥐와 쫓아가는 고양이가 같은 셀에 위치하게 되면 고양이가 쥐를 잡은 것으로 간주하고 게임이 종료된다. 쥐와 고양이는 그림 1과 같이 벽과 장애물들이 존재하는 5 × 5의 2차원 격자월드(grid world)에서 동시에 동, 서, 남, 북, 북동, 남동, 남서, 북서 등 여덟 방향에 위치한 인접 셀(cell) 중 하나로 이동(a_0~a_7)할 수 있다. 단, 벽이나 장애물이 놓여있는 셀로의 이동은 허용되지 않다. 게임의 상태(s)는 고양이의 위치좌표, 쥐의 위치좌표, 치즈의 위치좌표의 조합으로 표현한다. 따라서 상태집합의 크기$|S|$는 $(5 \times 5)^3$ = 15625이고, 가능한 연합 동작들의 수$|\vec{A}|$는 8 × 8 = 64이므로, 전체 상태공간의 크기는 15625 × 64 = 1000000이다. 고양이와 쥐게임에서 상태전이는 언제나 현재 상태와 에이전트들이 수행한 연합 행동에 따라 일정한 하나의 후속 상태가 정해진다. 따라서 고양이와 쥐게임에서 상태전이는 하나의 결정적 함수(deterministic function)로 표현될 수 있다. 그리고 쥐는 치즈를 먹었을 때, 고양이는 쥐를 잡았을 때 각각 일정한 양의 보상 값을 받는 것으로 가정한다.



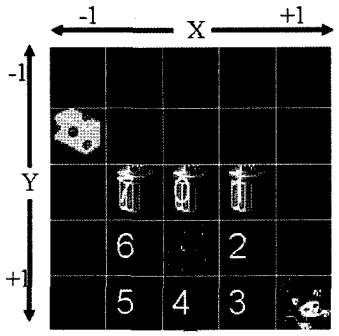


그림 1 고양이와 쥐게임: 실행 화면(상)과 행동(하)

4.2 실험 목적 및 방법

본 논문에서는 앞서 제안한 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석하기 위해 고양이와 쥐게임을 이용한 실험을 전개하였다.

실험의 첫 번째 목적은 상대방 정책 모델 PM이 강화 학습의 효율성에 미치는 영향을 분석해 보는 것이다. 상대 모델(opponent model)의 효과는 상대 에이전트가 어떤 정책을 쓰느냐에 매우 의존적이다. 상대 에이전트가 언제나 고정된 하나의 정책(fixed policy)에 따라 행동을 한다고 가정하면, 다른 특별한 이유가 없는 한 학습 에이전트에게 주어지는 보상 값과 상태전이는 동적으로 변화하지 않고 정적이다. 이런 경우, 상대 에이전트를 환경의 한 부분으로 간주하고 환경에서 분리하여 별도의 모델을 만들지 않아도 학습이 효과적으로 이루어 질 수 있을 것으로 기대된다. 하지만 반대로 상대 에이전트 역시 학습 기능을 가지고 시간에 따라 정책을 변경하는 경우에는 상대 에이전트를 환경에서 분리하여 별도의 모델을 구축하는 것이 학습에 효과적일 것으로 기대된다. 따라서 실험에서는 상대 에이전트(고양이)가 (i) 고정된 정책을 사용하는 경우와 (ii) Q 학습을 통해 정책을 변경하는 경우, 그리고 (iii) 상대 에이전트 역시 상대방 정책 모델 PM을 이용하는 Q 학습을 하는 경우로 구분하였고, 또한 학습 에이전트(쥐)도 (a) 상대 모델이 없는 Q 학습과 (b) 상대방 정책 모델 PM을 이용한 Q 학습을 전개하는 경우로 나누어, 각각의 경우에 대한 비교 실험하였다.

실험에 사용한 고양이의 고정 정책(fixed policy)은 현재 쥐가 놓여 있는 위치를 향해 한 칸 다가가는 행동을 선택하는 것이다. 그리고 각 경우의 실험에서 학습의 효율성을 분석하기 위해 식 (11)과 같은 Bellman 오차(Bellman residual)를 측정해 보았다.

$$BE_t = \max_{s \in S} |V_t(s) - V_{t-1}(s)| \qquad (11)$$

여기서 $V_t(s) = \max_{a_{self} \in A_{self}} \overline{Q_t}(s, a_{self})$이다.

실험의 두 번째 목적은 상대방 정책 모델 PM이 학습의 결과로 나타나는 에이전트의 성능에 미치는 영향을 분석해 보는 것이다. 이 목적을 위해 위에서 설명한 바와 같이 서로 다른 정책(고정 정책, 단순 Q 학습, PM 기반의 Q 학습)을 사용하는 상대 에이전트들과 서로 다른 학습(상대 모델이 없는 Q 학습, PM 기반의 Q 학습)을 수행하는 학습 에이전트들 간의 게임을 차례대로 전개하면서 각 경우의 실험으로부터 게임 지속 시간(game duration time)을 측정하여 비교하였다. 게임 지속 시간은 새로운 게임이 시작되어 쥐가 고양이에 의해 잡힐 때까지 유지한 시간을 말한다. 게임 지속 시간이 길어지면 쥐의 성능이 향상되고 있는 것으로 판단할 수 있다.

실험의 세 번째 목적은 상대 에이전트의 정책에 따른 상대방 정책 모델 PM의 수렴속도를 분석하는 것이다. 이를 위해 서로 다른 정책들(고정 정책, 상대방 모델이 없는 Q 학습, PM 기반의 Q 학습)을 사용하는 상대 에이전트들과 PM 기반의 학습 에이전트 간의 게임들을 수행하면서 식 (12)와 같이 모델 오차를 계산해 보았다.

$$PE_t = \max_{s \in S} |P_t(s) - P_{t-1}(s)| \qquad (12)$$

여기서 $P_t(s) = \max_{a_{opponent} \in A_{opponent}} PM(s, a_{opponent})$이다.

4.3 실험 결과

그림 2와 그림 3은 상대방 정책 모델 PM이 학습 효율성에 미치는 영향을 분석하기 위한 비교 실험 결과를 나타내고 있다. 그림 2와 그림 3은 각각 쥐가 (a) 상대 모델이 없는 단순 Q 학습을 수행하는 경우와 (b) 상대방 정책 모델 PM을 이용한 Q 학습을 수행하는 경우의 Bellman 오차를 보여주고 있다.

그림 2와 그림 3에는 상대 에이전트인 고양이가 (i) 고정 정책을 쓰는 경우, (ii) 상대 모델이 없는 단순 Q 학습을 하는 경우, (iii) 고양이 역시 상대방 정책 모델 PM기반의 Q 학습을 하는 경우 등 총 세 가지 경우의 실험결과들을 함께 그래프로 나타내고 있다.

그래프의 가로축은 학습에 소요되는 시간을 반복주기(epoch)*1/1000로 나타내고 있고, 세로축은 학습시간에 따른 Bellman 오차를 나타내고 있다. 실험에서 상대 에이전트인 고양이의 행동 정책 모델을 계산하기 위한 파

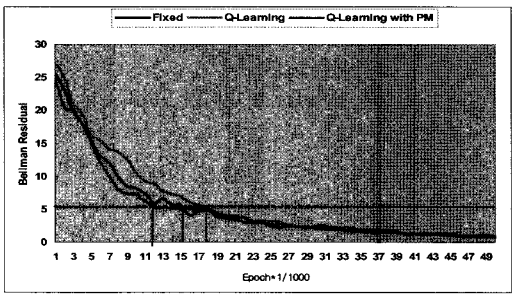


그림 2 Bellman오차 비교: 단순 Q 학습

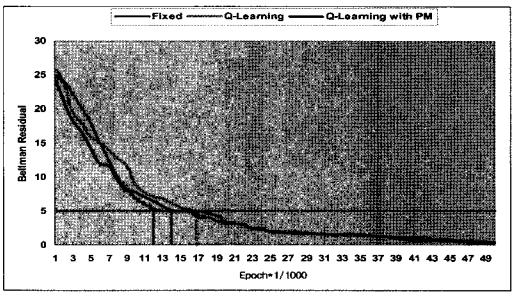


그림 3 Bellman오차 비교: PM을 이용한 Q 학습

리미터 θ와 τ의 값은 0.2로 설정하였다. 또 학습 율α는 0.9로 설정하였다.

그림 2와 그림 3을 통해 발견할 수 있는 중요한 사실들은 다음과 같다. 먼저, 전체적으로 그림 3의 경우, 즉 쥐가 상대인 고양이에 대한 행동 정책 모델 PM을 이용하여 Q 학습을 하는 경우가 그림 2의 경우, 즉 어떤 명시적인 상대 모델도 이용하지 않은 Q 학습보다 Bellman 오차가 더 빨리 일정 수준 이하로 감소된다는 것이다. 각 그림의 그래프에서 가로축으로 내린 수직선들은 Bellman 오차가 일정 수준이하로 감소하면서 수렴하기 시작하는 시점들을 표시한 것이다. Bellman 오차의 감소는 곧 Q 함수의 수렴을 의미한다. 예를 들어 그림 2의 경우 Bellman 오차가 5인 시점을 기준으로 보면 Q- Learning with PM의 경우는 12000 epoch, 단순 Q- Learning의 경우는 18000 epoch, 고정된 정책의 경우는 15000 epoch임을 나타내고 있다. Bellman 오차가 5를 기준으로 그 이하가 된다는 것은 상태에 대한 현재와 이전의 가치(Value)가 줄었다는 것을 의미하며 이는 상태에 대한 최적의 가치에 수렴해 간다고 말할 수 있다. 따라서 상대방 정책 모델 PM을 이용한 Q 학습이 상대 모델을 이용하지 않는 Q 학습 보다 빨리 Q 함수가 수렴한다는 것을 알 수 있다.

그림 2와 그림 3을 통해 발견할 수 있는 또 다른 사실은, 상대 에이전트인 고양이가 고정 정책을 쓰는 경우보다 학습을 통해 정책을 변경하는 경우에, 상대 모델 PM 기반의 Q 학습이 상대 모델이 없는 단순 Q 학습에 비해 Q 함수의 수렴이 빠르다는 점이다.

그림 4와 그림 5는 상대방 정책 모델 PM이 에이전트의 성능에 미치는 영향을 분석하기 위한 비교 실험 결과를 나타내고 있다. 그림 4와 그림 5는 각각 쥐가 (a) 상대 모델이 없는 단순 Q 학습을 수행하는 경우와 (b) 상대방 정책 모델 PM을 이용한 Q 학습을 수행하는 경우의 게임 지속 시간을 보여주고 있다. 두 그림을 통해 확인할 수 있는 첫 번째 사실은 두 그림 모두 학습이 진행됨에 따라 게임 지속 시간이 증가한다는 것이다. 이것은 학습이 진행됨에 따라 쥐의 성능도 향상되었다는 것을 입증한다. 하지만 일정 수준이상이 되면 성능 향상이 계속되지 않고 수렴한다는 것도 확인할 수 있다. 이것은 게임이 계속되는 동안 상대 에이전트인 고양이와 일종의 Nash 평형상태에 도달함으로써 성능이 일정한 수준을 유지하는 것으로 추측된다. 그림 4와 그림 5의 실험결과를 통해 발견할 수 있는 두 번째 사실은 쥐가 학습을 하는 경우 (상대 모델이 있든지 없든지 상관없이)에는 상대 에이전트인 고양이가 고정 정책을 쓸 때보다 고양이도 학습을 할 때가 더 빨리 성능 수렴상태에 도달한다는 것이다. 특히 상대 에이전트인 고양이가

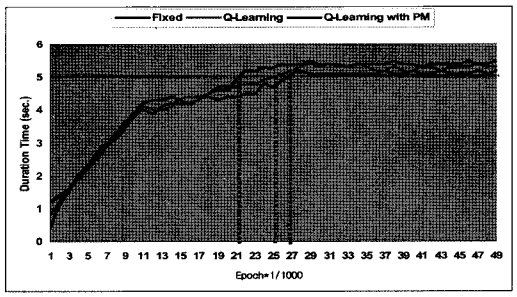


그림 4 게임 지속시간 비교: 단순 Q 학습

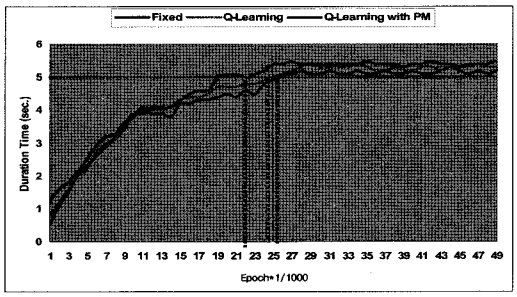


그림 5 게임 지속시간 비교: PM을 이용한 Q 학습

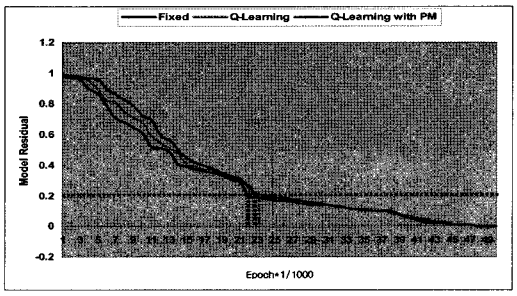


그림 6 모델 오차 비교: PM을 이용한 Q 학습

상대방 정책 모델 PM을 이용한 Q 학습을 할 때 가장 빨리 성능 수렴상태에 도달하였다는 것을 확인할 수 있다. 하지만 이 실험결과를 통해서는 쥐가 상대방 정책 모델인 PM을 이용하는 경우와 그렇지 않은 경우 간의 뚜렷한 성능 차이를 확인할 수는 없었다.

그림 6은 상대 에이전트의 정책에 따른 상대방 정책 모델 PM의 수렴속도를 분석하기 위한 실험 결과를 나타내고 있다. 그림 6에서는 상대 에이전트인 고양이의 서로 다른 정책별로 모델 오차(model residual)의 변화를 나타내고 있다. 그림 6의 실험결과로부터 알 수 있는 한 가지 사실은 고양이가 세 가지 정책 중 어떤 것을 쓰든지 상관없이 게임이 거듭됨에 따라 모델 오차가 0에 가까워졌다. 이것은 곧 실험에 적용된 고양이의 서로 다른 세 가지 정책 모두에 대해, 쥐가 습득하는 상대방

정책 모델 PM이 수렴할 수 있다는 것을 의미한다. 또한 비록 미미한 차이지만 상대 에이전트인 고양이가 고정 정책을 쓰는 경우가 학습에 의한 가변 정책을 쓰는 경우보다 좀 더 빨리 모델 오차를 줄일 수 있었다. 본 논문에서 말하는 상대모델은 특정 상태 s에서 상대 에이전트가 행동 a를 수행할 가능성을 0과 1사이의 값으로 표현한 것을 말한다. 그리고 이와 같은 상대모델 역시 에이전트의 경험과 관찰을 통해 갱신을 거듭하며 충분한 갱신이 이루어지고 나면 이 모델도 수렴을 한다고 가정하고 있다. 모델의 수렴 성을 알아보기 위한 척도로 갱신 이전의 모델 값 $PM_{t-1}(s, a_{opponent})$과 갱신 이후의 모델 값 $PM_t(s, a_{opponent})$의 차이를 이용할 수 있다. 따라서 모델 오차란 바로 갱신 이전과 이후의 모델 값 최대 차인 $\max_{s\in S}\sum a_{opponent}\in A\ |PM_{t-1}(s, a_{opponent}) - PM_t(s, a_{opponent})|$를 말한다. 모델 오차가 크다는 것은 아직 상대모델이 충분히 수렴되지 않았다는 것을 나타내며, 이러한 경우 현재 모델을 토대로 상대 에이전트의 행동을 예측하는 것은 신뢰도가 떨어진다는 것을 의미한다.

5. 결 론

본 논문에서는 적대적 멀티 에이전트 환경에서 과거에 관찰된 상대 에이전트의 행동들을 기초로 상대 에이전트의 행동 정책 모델인 PM을 학습하고, 이 모델을 바탕으로 다시 자신의 최적 정책을 학습하는 멀티 에이전트 강화 학습방법을 제시하였다. Q 학습 알고리즘을 확장한 이 멀티 에이전트 강화학습 방법은 상대 모델을 이용하는 기존의 멀티 에이전트 강화 학습 연구들에서 주로 시도되었던 상대 에이전트의 Q 평가 함수 모델 대신 상대 에이전트의 행동 정책 모델을 학습하며, 표현력은 풍부하나 학습에 시간과 노력이 많이 요구되는 유한 상태 오토마타나 마코프 체인과 같은 행동 정책 모델들에 비해 비교적 간단한 형태의 행동 정책 모델을 이용함으로써 학습의 효율성을 높인 것이 특징이다. 본 논문에서는 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐(Cat and Mouse) 게임을 테스베드로 삼아 다양한 비교 실험들을 전개하여 본 논문에서 제안한 상대방 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석해보았다. 이 실험을 통해 상대방 정책 모델을 이용하는 것이 강화 학습의 효율성과 에이전트의 성능 향상에 도움이 되며, 상대 에이전트가 고정 정책을 쓰는 경우는 물론 Q 학습을 하는 경우에도 상대방 정책 모델 PM의 수렴 성을 확보할 수 있다는 것을 확인하였다. 본 연구의 결과를 두 명 이상 다수의 에이전트가 참여하는 제로-합 게임이나 일반-합 게임으로 확장하는 것은 의미있는 향후 연구가 될 것으로 판단한다.

참 고 문 헌

[1] Yang E. and Gu D., "Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey," University of Essex Technical Report CSM-404, 2004.

[2] Tesauro G., "Multi Agent Learning: Mini Tutorial," IBM T.J.Watson Research Center, 2000.

[3] Rahimi K.A., Tabarraei H., Sadeghi B., "Reinforcement Learning Based Supplier-Agents for Electricity Markets," Proceedings of the IEEE International Symposium on Control and Automation, pp. 1405-1410, 2005.

[4] Shoham Y., Powers R., and Grenager T., "Multi-Agent Reinforcement Learning: A Critical Survey," Technical Report, Stanford University, 2003.

[5] Littman M.L., "Markov Games as Framework for Multi-Agent Reinforcement Learning," Proceedings of the 11th International Conference on Machine Learning, pp. 157-163, 1994.

[6] Hu J. and Wellman M.P., "Nash Q-learning for General-Sum Stochastic Games," Journal of Machine Learning Research, Vol.4, pp. 1039-1069, 2003.

[7] Littman M.L., "Friend-or-Foe Q-learning in General-Sum Games," Proceedings of the 18th International Conference on Machine Learning, Morgan Kaufman, pp. 322-328, 2001.

[8] Claus C. and Boutilier C., "The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems," Proceedings of AAAI-98, pp. 746-752, 1998.

[9] Carmel D. and Markovitch S., "Learning Models of Intelligent Agents," Proceedings of AAAI-96, pp. 62-67, 1996.

[10] Riley P. and Veloso M., "Advice Generation from Observed Execution: Abstract Markov Decision Process Learning," Proceedings of AAAI-2004, 2004.

[11] Sutton, R.S., Barto, A.G. Reinforcement Learning: An Introduction, MIT Press, 1998.

[12] Chalkiadakis G. and Boutilier C., "Multiagent Reinforcement Learning: Theoretical Framework and An Algorithm," Proceedings of the 2nd AAMAS-03, pp. 709-716, 2003.

권 기 덕

1998년 경기대학교 전자계산학과 학사 2002년 경기대학교 대학원 전자계산학과 석사. 2003년~현재 경기대학교 대학원 전자계산학과 박사과정. 관심분야는 멀티 에이전트, 강화 학습

김 인 철

1985년 서울대학교 수학과 학사. 1987년 서울대학교 대학원 계산통계학과 석사 1995년 서울대학교 대학원 계산통계학과 박사. 1996년~현재 경기대학교 정보과학부 교수. 관심분야는 멀티 에이전트, 강화 학습, 계획기