효율적인 멀티 에이전트 강화학습을 위한 RBFN 기반 정책 모델
(RBFN-based Policy Model for Efficient Multiagent Reinforcement Learning)

권기덕[a], 김인철[b]

경기대학교 전자계산학과
경기도 수원시 영통 구 이의동 산 94-6번지, 200-706
Tel: +82-31-249-9670, Fax: +82-31-249-9673, E-mail: {kdkwon[a], kic[b]}@kyonggi.ac.kr

Abstract

멀티 에이전트 강화학습에서 중요한 이슈 중의 하나는 자신의 성능에 영향을 미칠 수 있는 다른 에이전트들이 존재하는 동적 환경에서 어떻게 최적의 행동 정책을 학습하느냐 하는 것이다. 멀티 에이전트 강화 학습을 위한 기존 연구들은 대부분 단일 에이전트 강화 학습기법들을 큰 변화 없이 그대로 적용하거나 비록 다른 에이전트에 관한 별도의 모델을 이용하더라도 현실적이지 못한 가정들을 요구한다. 본 논문에서는 상대 에이전트에 대한RBFN기반의 행동 정책 모델을 소개한 뒤, 이것을 이용한 강화 학습 방법을 설명한다. 본 논문에서 제안하는 멀티 에이전트 강화학습 방법은 기존의 멀티 에이전트 강화 학습 연구들과는 달리 상대 에이전트의 Q 평가 함수 모델이 아니라 RBFN 기반의 행동 정책 모델을 학습한다. 또한, 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. 본 논문에서는 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐 게임을 소개한 뒤, 이 게임을 테스트 베드 삼아 실험들을 전개함으로써 제안하는 RBFN 기반의 정책 모델의 효과를 분석해본다.

Keywords:

멀티 에이전트 강화 학습, 상대 정책 모델, RBFN, 고양이와 쥐 게임

1. 서론

단일 에이전트의 환경에서 에이전트가 자신의 행동에 대한 보상 값을 기초로 스스로 최적의 행동 정책을 학습해 나가는 과정은 하나의 마코프 결정 문제 (MDP)로 표현할 수 있다[1,6]. MDP (Markov Decision Problem)는 단일 에이전트가 상태 s 의 필요한 모든 정보를 함축하고 있다면 현재의 상태 s 를 기반으로 자신의 최적 행동 정책을 결정한다. 그러나 다수의 에이전트들이 행동하는 것을 고려하여 자신의 최적 행동 정책을 결정해야 하는 멀티 에이전트 환경은 MDP 로 표현하기 적합하지 않다[4,5]. 따라서 다수의 에이전트들이 공존하며 서로 상호 작용하는 멀티 에이전트 환경은 하나의 확률 게임으로 일반 화 할 수 있다[3,11].

멀티 에이전트 강화 학습은 확률 게임에 참여하는 다른 에이전트들을 고려하여 자신의 최적 행동 정책을 학습해 나가는 것이다. 그러나 일반적으로 멀티 에이전트 환경에서 자신의 최적 행동 정책을 수립해 나간다는 것은 상대 에이전트를 고려해야 하기 때문에 매우 어려운 문제이다[7,8]. 이 문제를 해결하기 위해 다른 에이전트에 대한 명시적인 모델을 학습한다.

그 동안 다른 에이전트의 존재를 모델링 하는 연구들도 있었다. 나이브 베이지안 (Naïve Bayesian)과 같은 가벼운 모델을 적용하여 상대 에이전트의 행동 정책이나 가치 함수를 모델링 하는 방법으로 다른 에이전트들에 대한 요구되는 정보나 가정이 비현실적이다라는 단점을 가지고 있다[4,5]. 다른 접근 방법으로 멀티 에이전트 환경에서 존재하는 다른 에이전트들에 대한 풍부한 모델을 이용하는 방법인데, 이 방법은 HMM (Hidden Markov Model) 등과 같이 깊고 복잡한 확률 모델을 만든다[7,8]. 이 연구 방법들은 상대 에이전트의 모델을 학습하는데 학습 시간이 오래 걸리는 단점을 가지고 있다. 또 다른 방법으로 본 논문에서는 상대

에이전트의 행동 정책 모델을 모델 일반화 방법인 RBFN 을 통해 학습하고, 이 모델을 바탕으로 다시 자신의 최적 정책을 학습하는 강화 학습방법을 제시한다.

본 논문에서 제안하는 멀티 에이전트 강화 학습방법은 제로-합 확률 게임으로 두 명의 에이전트로 구성된 적대적 멀티 에이전트 환경을 가정하며, 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 모델 수립에 RBFN 을 이용함으로써 학습의 효율성을 높였다.

본 논문의 구성은 멀티 에이전트 강화 학습 방법에 대한 기본 개념들을 살펴보고 상대 모델의 일반화 방법인 RBFN 에 대해 살펴본다. 그리고 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐 게임을 소개하고 이 게임을 테스트 베드 삼아 수행한 비교 실험 결과들을 설명함으로써 본 논문에서 제안하는 RBFN 기반 상대 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석해 본다.

2. 관련 연구

2.1 멀티 에이전트 강화 학습

멀티 에이전트 시스템은 동적이고 복잡한 환경에서 에이전트들은 자신의 행동을 결정하기 위해 다른 에이전트들의 행동을 고려하여야 한다. 단일 에이전트 시스템에서의 강화 학습은 보상 값을 최대화하는 정책만을 학습하면 된다. 그러나 멀티 에이전트에서의 강화 학습 문제는 개인의 보상 값뿐만 아니라 팀이나 또는 전체의 보상 값을 최대로 하는 정책을 학습하여야 한다.

멀티 에이전트 강화 학습의 경우 다른 에이전트들이 각자 자신의 행동을 수행한다는 것을 가정한다. 이 가정하에 효율적인 강화 학습을 하기 위해서는 상대편에 대한 정책이나 학습된 결과를 모델로 가지는데, 그 모델을 자기 자신의 학습에서 고려함으로써 학습에 대한 수렴 속도를 높인다.

다수의 에이전트들이 공존하며 서로 상호작용하는 멀티 에이전트 환경은 하나의 확률 게임으로 표현할 수 있다. 하나의 확률 게임(stochastic game, SG)은 튜플 $<N, S, \vec{A}, T, \vec{R}>$로 정의한다. N 은 에이전트들의 수, S 는 게임 상태들의 유한집합, $\vec{A} = A_1,...,A_n$, 이때 각 A_i 는 에이전트 i 가 선택할 수 있는 행동들의 집합, $T: S \times \vec{A} \rightarrow \Pi(S)$ 는 에이전트들의 행동에 따라 다음 상태를 결정하는 상태전이함수, $\vec{R} = R_1,...,R_n$ 이때 각 $R_i: S \times \vec{A} \rightarrow R$ 는 에이전트 i 의 보상 값을 결정하는 보상함수를 나타낸다.

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

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.

2.2 상대 모델의 일반화

본 논문에서는 제로-합 확률 게임에서 상대 에이전트를 대상으로 RBFN 을 통한 상대 행동 정책 모델을 수립한다. 상대에 대한 완전한 모델을 수립하기 위해서는 모든 상태에 대해 행동을 전부

알아야 하지만 실 세계의 복잡하고 동적인 환경에서는 불가능하다. 따라서 본 논문에서는 상대 모델을 수립하기 위해 모델 전체를 이용하는 것이 아니라 모델의 일부분만을 가지고도 상대의 행동 정책 모델을 수립하여 자신의 행동 정책 결정에 도움을 줄 수 있도록 하기 위한 모델 일반화 방법을 사용한다.

기존의 다른 에이전트 존재를 명시적으로 고려하는 모델 일반화 연구들도 있었는데, 나이브 베이지안 (Naïve Bayesian)과 같은 가벼운 모델을 적용하여 상대 에이전트의 행동 정책이나 가치 함수를 모델링하는 방법이다. 이 방법은 적용 가능한 멀티 에이전트 시스템의 유형이 제한적이고 다른 에이전트들에 대해 요구되는 정보나 가정이 비현실적이다라는 단점을 가지고 있다. 다른 접근 방법으로 멀티 에이전트 환경에서 존재하는 다른 에이전트들에 대한 풍부한 모델을 이용하는 방법인데, 이 방법은 HMM(Hidden MDP), DBN 등과 같이 깊고 복잡한 확률 모델을 적용한다. 이 방법의 대표적인 연구로는 JAL(joint action learner) 연구로 연합 행동에 대한 다른 에이전트의 평가 함수 모델을 학습한다. 그리고 상대 정책 모델을 결정적 유한 상태 오토마타로 표현하여 학습한다. 또한 다른 에이전트들의 평가 함수 모델이나 행동 정책 모델을 학습하는 것이 아니라 환경의 상태 전이를 기록해 두었다가 이들로부터 하나의 마코프 체인을 추출하고 여기에 가능한 행동들을 붙여 하나의 MDP 를 만드는 방법이다. 이 연구 방법들은 상대 에이전트의 모델들을 학습하는데 오랜 학습 시간을 필요로 한다는 단점을 가지고 있다.

다른 모델 일반화 방법에는 신경망과 RBF 방법 등이 있다[9]. 신경망은 유효한 예측을 하려면 많은 훈련 예제를 필요로 하고 훈련 예제들은 상태 공간에 골고루 분포되어 있어야 한다. 또한 신경망은 블랙박스로서 예측하는 과정을 설명 할 수 없다. 이런 신경망의 단점들을 보완하기 위한 방법으로 본 논문에서는 RBF 방법을 제안한다. RBF 방법은 상태 공간 전부에 대해 학습하는 것이 아니라 일부분 만을 가지고 예측이 가능하기 때문에 지역성의 특징을 가진다. 대표적인 RBF 는 가우시안(Gaussian) 함수이다. 1 차원 일 경우 식 1 과 같이 표현된다.

$$G(s) = \exp(-\frac{(s-c)^2}{\sigma^2}) \qquad [식\ 1]$$

여기서 c 는 중심, σ 은 그 함수가 지원하는 반경을 말한다.

RBFN 은 RBF 의 선형 조합으로 식 2 와 같이 표현한다.

$$f(s) = \sum_{i=1}^{m} \omega_i G(\|S - c_i\|) \qquad [식\ 2]$$

c_i 는 RBF 의 중심을 나타내는 벡터이고, $G(\|S-c_i\|)$ 는 RBF 의 조합으로 표현되며, RBF 의 개수가 근사화 할 대상의 점들 수보다 작게 만들어진 네트워크 구조를 가진다.

3. *RBFN 기반 멀티 에이전트 강화 학습*

3.1 제로 – 합 확률 게임

본 논문에서 가정하는 적대적 멀티 에이전트 환경은 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.

두 명의 제로-합 확률 게임(two-player zero-sum SG)은 튜플 <S, $\vec{A}$, T, $\vec{R}$ >로 정의한다. $S = \{s \mid s = (s_{self}, s_{opponent})\}$ 는 게임 상태들의 유한집합, $\vec{A} = \{(a_{self}, a_{opponent}) \mid a_{self} \in A_{self}, a_{opponent} \in A_{opponent}\}$ 는 연합 행동(joint action)들의 집합, 이때 각각 A_{self} 와 $A_{opponent}$ 는 에이전트 Self와 에이전트 Opponent가 선택할 수 있는 행동들의 집합, T: S X -> Π(S)는 에이전트 Self와 에이전트 Opponent의 연합 행동에 따라 다음 상태를 결정하는 상태전이함수, $\vec{R} = R_{self}, R_{opponent}$ 는 에이전트 Self와 에이전트 Opponent의 보상함수, 이때 $R_{self} : S \times \vec{A} \rightarrow R$ 는 에이전트 Self의 보상 값을, $R_{opponent} : S \times \vec{A} \rightarrow R$ 는 에이전트 Opponent의 보상 값을 결정하는 보상함수를 말한다.

3.2 RBFN 기반 상대 정책 모델

본 논문에서는 상대 모델을 이용하는 기존의 멀티 에이전트 강화 학습 연구들에서 주로 시도되었던 상대 에이전트의 Q 평가 함수 모델 대신 상대

에이전트의 행동 정책 모델 수립에 RBFN 을 이용한다. 이를 위해 대표적인 RBF 방법으로 식 1 과 같은 가우시안 함수를 이용한다. 여기서 RBFN 의 중심은 하나의 상태에서 지금까지 실행한 행동들의 집합으로 다음 시점에서 어떤 행동을 실행하면 유추해서 하나의 상태에서 이런 행동을 할 것이라고 예측한다. 그리고 σ 는 함수가 지원하는 반경으로 지역성의 특성을 나타내게 되는데, σ 이 크면 너무 많은 범위로 나눠지기 때문에 자신의 행동 정책을 선택하는데 있어서 많은 행동 패턴들을 고려하여야 하기 때문에 학습 시간이 오래 걸린다. 만약 σ 을 작게 하면 고려해야 할 범위가 너무 광범위해져 잘못된 선택을 하게 된다. RBFN 의 파라미터들은 RBF 들의 선형 조합으로 주어진 상태를 근사화 할 때 사용되는 가중치들을 말한다[11]. 따라서 이 파라 미터는 최선의 결과를 가져올 수 있도록 결정해야 한다. 이것은 에러 역 전파와 같은 반복적인 에러 수정을 통하여 구할 수 있지만 이것은 시간이 매우 많이 걸리고 그 결과가 안정적이지 않다. 따라서 선형 최소 제곱 알고리즘을 이용하여 가중치를 구한다.

원래의 상태 한 부분을 $f(s)$라는 함수로 볼 때 원하는 $f(s)$라는 함수와 가장 에러가 적은 함수 $f(s)$를 구하는 것이다. 따라서 수식으로 표현하면 $f(s)$를 RBFN 으로 구성하므로 식 3 과 같은 비용 함수의 값을 최소화하는 가중치들을 구한다.

$$\delta(f) = \sum_{i=0}^{N} d_i - \sum_{j=0}^{m} \omega_j G(\|S_i - c_j\|)^2 \qquad \text{[식 3]}$$

여기서 N 은 점들의 개수로 상태와 행동의 집합을 말한다. d_i는 $f(s)$상의 i번째 상태와 행동 집합에 대한 함수 값을 나타낸다. $\omega = [\omega_1, \omega_2, ..., \omega_m]$ 이고, $\|d - G\omega\|^2$의 형태가 된다. 따라서 이것을 최소화하는 ω 는 선형 최소 제곱 방법을 이용하여 식 4 를 이용하여 구할 수 있다.

$$\omega = (\Phi^T \Phi + \lambda \Phi_0)^{-1} \Phi^T d \qquad \text{[식 4]}$$

여기서 d 는 학습 데이터의 출력 값으로 현재 상태에서 실행할 행동에 대한 확률 값이다.

RBFN 을 통해 실행된 상대 에이전트의 행동을 관찰함으로써 얻는 상대 에이전트의 행동 추정 함수 $PM : S \times A_{opponent} \rightarrow (0,1)$ 을 상대방 정책 모델(opponent's policy model)이라고 한다.

상대방 정책 모델 $PM(s, a_{opponent})$ 은 상태 s 에서 상대 에이전트가 행동 $a_{opponent}$ 을 수행할 가능성을 0 과 1 사이의 값으로 추정한 것이다. 그리고 이러한 상대방은 RBFN 을 통해 나온 값에 따라 적응적으로 조정된다. 즉, 상태 s 에서 상대 에이전트가 행동 $a^*_{opponent}$ 을 수행하는 것을 관찰하면, 상태 s 에서 수행 가능한 모든 행동 $a_{opponent}$ 에 대한 상대방 정책 모델 $PM(s, a_{opponent})$은 식 5 와 같이 갱신된다.

$$PM(s, a_{opponent})^N = PM(s, a_{opponent})^O + \arg\max PM(s, a_{opponent}) \qquad \text{[식 5]}$$

3.3 강화 학습 알고리즘

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.

RBF 모델은 자신의 행동 정책을 수립하는데 있어서 다른 에이전트의 상대 행동을 예측하여 나의 행동을 수행함으로써 좀 더 효율적인 학습을 할 수 있도록 하기 위해 사용된다.

그림 1 은 RBF 를 이용한 상대 에이전트의 행동을 모델링 하여 자신의 행동 정책을 수립해 가는 과정을 도식화 한 것이다.

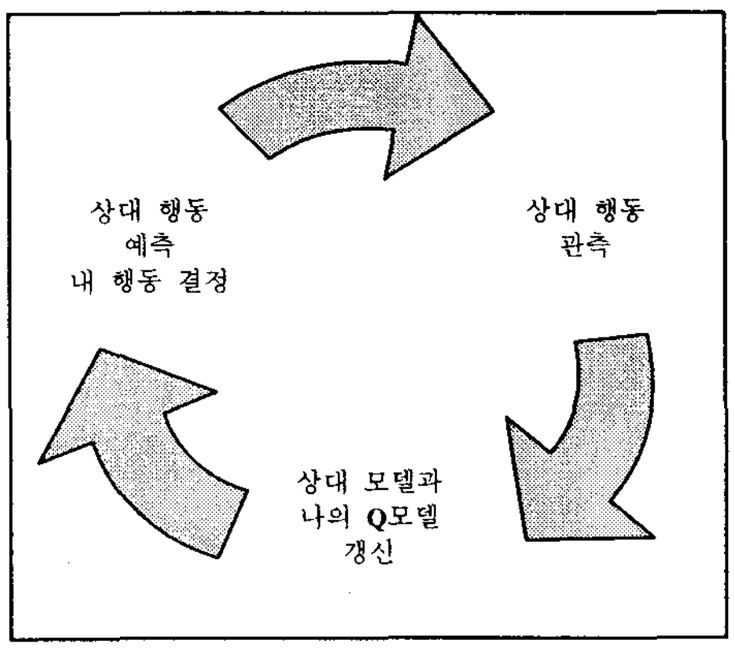

그림 1. 전체 실행도

3.3.1 상대 행동 예측

상대 행동을 바탕으로 나의 행동을 결정한다. 학습 초기 상대의 행동에 대한 예제(Sample)가 없기 때문에 상대 행동에 대해 나의 최적 행동 결정은 어렵다. 따라서 나의 행동은 에피소드 초기에는 랜덤 하게 선택된 상대의 행동을 바탕으로 Q-학습에 의해 선택한다. RBFN 에서 중심은 지금까지 실행한 행동들의 집합을 기준으로 다음에 어떤 행동을 실행하면 유추해서 이런 행동을 할 것이라고 예측한다.

3.3.2 상대 행동 관측

상대 에이전트의 행동은 관측할 수 있으며 이 행동을 바탕으로 하나의 에피소드가 끝날 때까지 나의 Q 학습에 의한 Q 테이블의 값을 갱신하며, 하나의 에피소드가 끝나면 행동에 대한 보상 값을 받게 되며, 관측된 상대의 행동 패턴을 저장한다.

강화 학습 에이전트는 보상 값이 큰 행동을 선택해야 할 뿐만 아니라 장기간 동안 받은 보상 값을 최대화 하는 행동을 학습해야 한다. 보상 값은 시간 t 이후에 받은 일련의 보상 값 $r_{t+1}r_{t+2}r_{t+3}...$의 함수 R_t 로 정의할 수 있다. 함수 R_t 의 가장 단순한 형태는 식 6과 같이 보상 값의 합이 된다. 식 6에서 마지막 보상 값 r_T의 T는 에이전트가 목표에 도달한 시점이거나 문제에서 정한 종료 시점을 의미한다. 에이전트와 환경의 상호 작용은 최종 상태를 갖는 에피소드들로 나눌 수 있고 그러한 문제를 에피소딕 작업(task)이라 한다.

$$R_t = r_{t+1} + r_{t+2} + ... + r_T \qquad [식\ 6]$$

하지만 에이전트와 환경의 상호 작용은 많은 경우에 에피소드들로 쉽게 나눌 수 없고 끝없이 계속된다. 이러한 지속적 문제인 경우에 위의 식 6 을 이용한다면 $T=\infty$가 되어서 R_t 의 값은 무한대가 될 수 있다. 이러한 문제를 해결하기 위하여, 감퇴율(discount rate)을 추가하여 함수 R_t 는 새롭게 정의할 수 있다.

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma r_{t+3} ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} (0 \le \gamma \le 1) \qquad [식\ 7]$$

감퇴율은 미래 보상 값의 현재 가치를 정한다. 미래 k 시간 단계에서 받은 보상 값은 γ^{k-1} 만큼의 가치를 갖는다. 만일 $\gamma<1$ 이라면, 무한의 합 R_t 는 유한의 값을 갖게 된다. 만일 γ가 0 이라면 일 단계 그리디 전략(one-step greedy policy)이 되고 최적의 행동은 다음 시간 단계 $t+1$ 에서 가장 큰 보상 값을 주는 행동이 된다. γ 의 값이 1 에 가까워질수록 미래의 보상 값을 좀 더 많이 고려하게 된다.

상대의 행동 패턴은 Q 함수 값과 함께 테이블에 저장된다. Q 함수 값이 크다는 것은 다음 상태에서 그 행동을 실행 할 확률이 높다는 것을 말한다.

3.3.3 모델 갱신

RBFN 기반 상대 모델과 나의 Q 모델을 갱신한다. RBFN 기반 상대 모델은 위에서 제시한 식 5 에 의해 모델을 갱신하다. 기존의 최적 확률을 가지는 모델과 현재 시점에서 최고의 확률을 가지는 모델을 업데이트한다.

Q-함수 $Q^{\pi}(s, a)$는 에이전트가 상태 s 에서 행동 a 를 수행하고, 그 이후에는 정책 π 에 따라 행동하였을 때 기대되는 보상 값의 합을 나타낸다. 따라서 함수 $Q^{\pi}(s, a)$는 다음과 같이 정의한다.

$$Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \times \sum_{a' \in A} \pi(s',a') Q^{\pi}(s',a') \qquad [식\ 8]$$

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

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s) \qquad [식\ 9]$$

이 때 $V^*(s) = \max_{a' \in A} Q^*(s',a')$ 이다.

각 상태 s 에서 Q-함수 $Q^*(s, a)$의 값이 최대인 행동만을 선택하는 정책 π 을 그리디 정책(greedy policy)라고 한다. 즉, 그리디 정책은 각 상태 s 에 대해 Q-함수 $Q^*(s, a)$의 값이 최대인 행동에게만 확률 값 1 을 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 학습방법(temporal difference learning)의 하나이다. Q 학습은 식 10 에 따라 $Q_t(s, a)$ 함수 값을 반복 갱신함으로써, 최적의 $Q^*(s, a)$ 함수 값을 추정하는 과정으로 볼 수 있다.

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

[식 10]

이 때 r_t는 보상 값을, γ는 감퇴 율(discount rate)를 나타낸다.

위의 과정을 풀어보면 다음과 같다.

[표 1] RBFN 기반 MARL 알고리즘

```
For (episode = 0; episode++; episode = n) {
  Initialize the agents state as s(0)
    If (!mouse position = cat position) {
      Cataction execution by Random
      Mouseaction execution by Q-Learning
        For each Cataction[i] {
          Set s(t) to RBFN's inputs x(t),
          and calculate the output z(t) from Gaussian with Center[][]
          Save z(t) of probability RM[state][Cataction[i]]
          Save Center[][] = maxRM[state][Cataction[i]]
          Calculate Mouse Q-Value with Center[][]
      }
      Observe Cataction, receive rewards
      update Mouse Q-table
    }
 update Center[][] += Center[][]
}
```

4. 실험 및 분석

4.1 고양이와 쥐 게임

고양이와 쥐 게임(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.

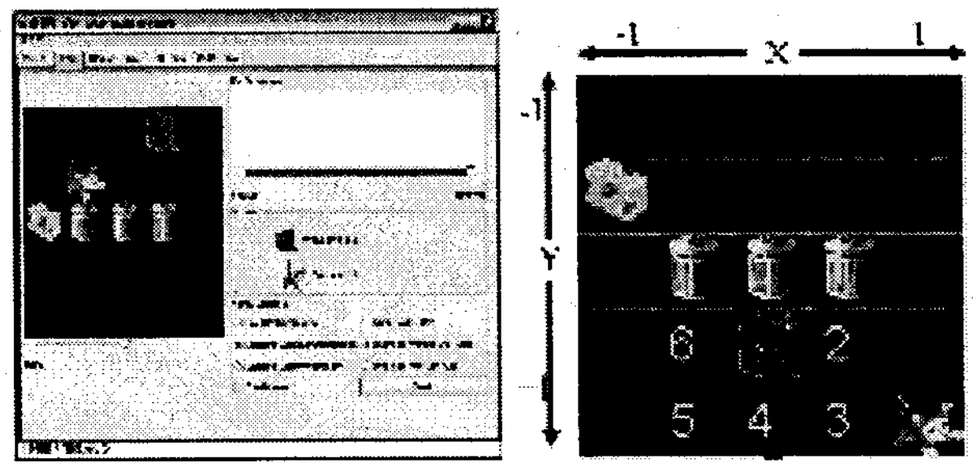

그림 2. 고양이와 쥐 게임 실행 화면(좌)과 행동(우)

쥐와 고양이는 그림 2 와 같이 벽과 장애물들이 존재하는 5 x 5 의 2 차원 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 크기는 $(5 \times 5)^3 = 15625$ 이고, 가능한 연합 동작들의 수는 $8 \times 8 = 64$ 이므로, 전체 상태공간의 크기는 $15625 \times 64 = 1000000$ 이다. 고양이와 쥐 게임에서 상태전이는 언제나 현재 상태와 에이전트들이 수행한 연합 행동에 따라 일정한 하나의 후속 상태가 정해진다. 따라서 고양이와 쥐 게임에서 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.

4.2 분석

본 논문에서는 앞서 제안한 정책 모델 기반의 멀티 에이전트 강화 학습의 효율과 에이전트의 성능을 분석하고 상대 정책 모델의 수렴을 분석하기 위해 고양이와 쥐 게임을 이용한 실험을 전개하였다. 고양이와 쥐 게임은 에피소드의 기간이 짧고 실행할 수 있는 행동이 제한적인 특징을 가지고 있다.

4.2.1 강화 학습 효율성

RBFN 기반의 상대방 정책 모델 PM 이 강화 학습의 효율성에 미치는 영향을 분석한다. 상대 에이전트가 학습에 의해 정책을 변경하는 경우에는 상대 에이전트를 환경에서 분리하여 모델을 구축하는 것이 학습에 효과적일 것이다. 그러나 이 모델 수립 과정에서 학습 초기에 발생할 수 있는

자신의 최적 정책 수립에 영향을 주는 모델에 대한 정보가 제대로 이루어지지 않을 수 있다. 이를 방지하기 위해 RBFN 기반의 상대 모델을 수립함으로써 효율적인 강화 학습이 이루어진다고 볼 수 있다. 이를 위해 실험에서는 상대 에이전트(고양이)가 Q 학습을 통해 정책을 변경하는 경우에 학습 에이전트(쥐)는 (i) 고정된 정책을 사용하는 경우와 (ii) Q 학습을 통해 정책을 변경하는 경우, 그리고 (iii) 상대방 정책 모델 PM 을 이용하는 Q 학습을 하는 경우, (iv) RBFN 기반 상대 정책 모델을 사용하는 경우에 대한 비교 실험하였다. 그리고 각 경우의 실험에서 학습의 효율성을 분석하기 위해 식 11 과 같은 Bellman 오차(Bellman residual)를 측정해 보았다.

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

여기서 $V_t(s) = \max_{a_{self} \in A_{self}} Q(s, a_{self})$ 이다. 그림 3 은 상대방 정책 모델 PM 이 학습 효율성에 미치는 영향을 분석하기 위한 비교 실험 결과를 나타내고 있다. 그래프의 가로축은 학습에 소요되는 시간을 반복주기(epoch)*1/1000 로 나타내고 있고, 세로축은 학습시간에 따른 Bellman 오차를 나타내고 있다. 실험에서 상대 에이전트인 고양이의 행동 정책 모델을 계산하기 위한 파라 미터 θ 와 τ 의 값은 0.2 로 설정하였다. 또 학습 율 α 는 0.9 로 설정하였다. 그림 3 을 통해 발견할 수 있는 중요한 사실들은 다음과 같다. 먼저, 전체적으로 그림 3 의 경우, 즉 쥐가 상대인 고양이에 대한 행동 정책 모델 PM 을 이용하여 Q 학습을 하는 경우보다 RBFN 을 통한 상대 정책 모델을 가지고 학습하는 경우가 어떤 명시적인 상대 모델도 이용하지 않은 Q 학습보다 Bellman 오차가 더 빨리 일정 수준 이하로 감소된다는 것이다.

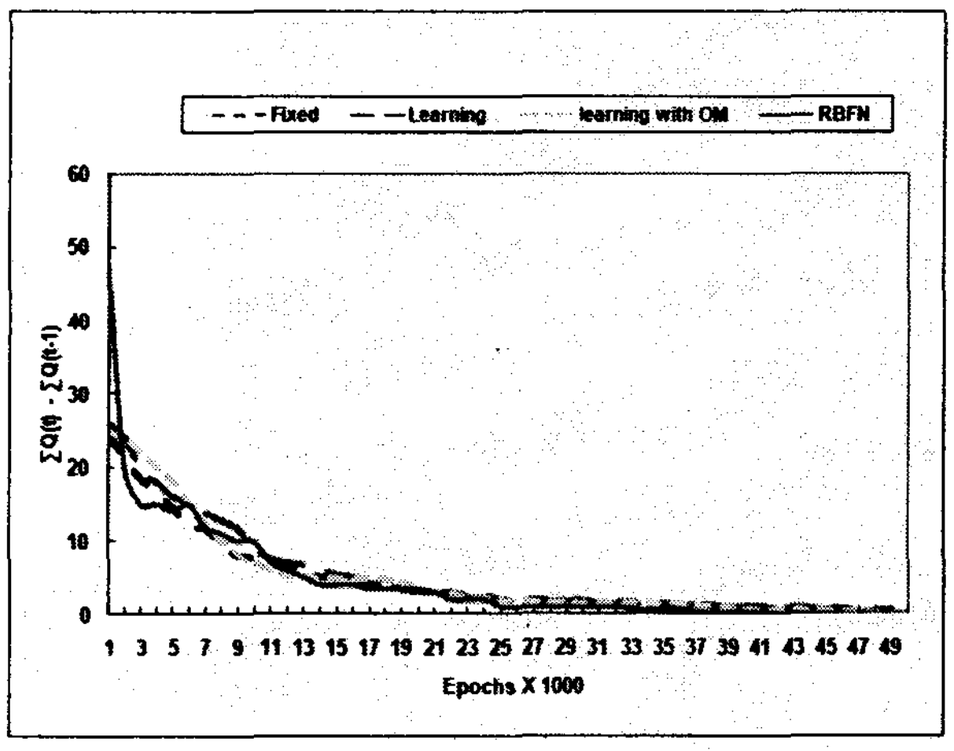

그림 3. Bellman 오차 비교

그림의 그래프에서 가로축으로 내린 수직선들은 Bellman 오차가 일정 수준 이하로 감소하면서 수렴하기 시작하는 시점들을 표시한 것이다. Bellman 오차의 감소는 곧 Q 함수의 수렴을 의미한다. 그림 3 을 통해 발견할 수 있는 또 다른 사실은, 행동의 관찰 후 수립된 상대 정책 모델과 RBFN 기반 상대 정책 모델의 현저한 차이는 느낄 수 없지만 처음에 RBFN 이 랜덤한 기준을 가지고 상대의 행동을 예측하기 때문에 벨만의 오차가 현저히 크지만 그 이후 급속한 속도로 오차의 범위가 줄어드는 것을 알 수 있다.

4.2.2 에이전트 성능

상대방 정책 모델 PM 이 학습의 결과로 나타나는 에이전트의 성능에 미치는 영향을 분석해 본다. 이 목적을 위해 서로 다른 정책(고정 정책, 단순 Q 학습, PM 기반의 Q 학습, RBFN 기반의 Q 학습)을 사용하는 학습 에이전트(쥐)가 상대 모델이 없는 Q 학습을 수행하는 상대 에이전트와 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.

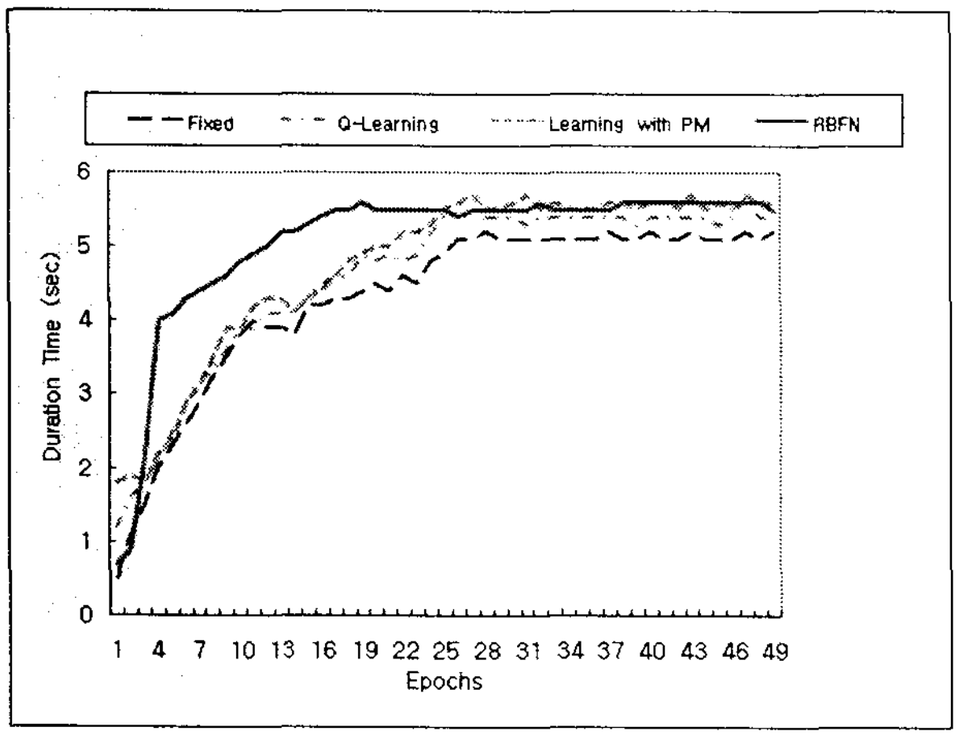

그림 4. 게임지속시간 비교

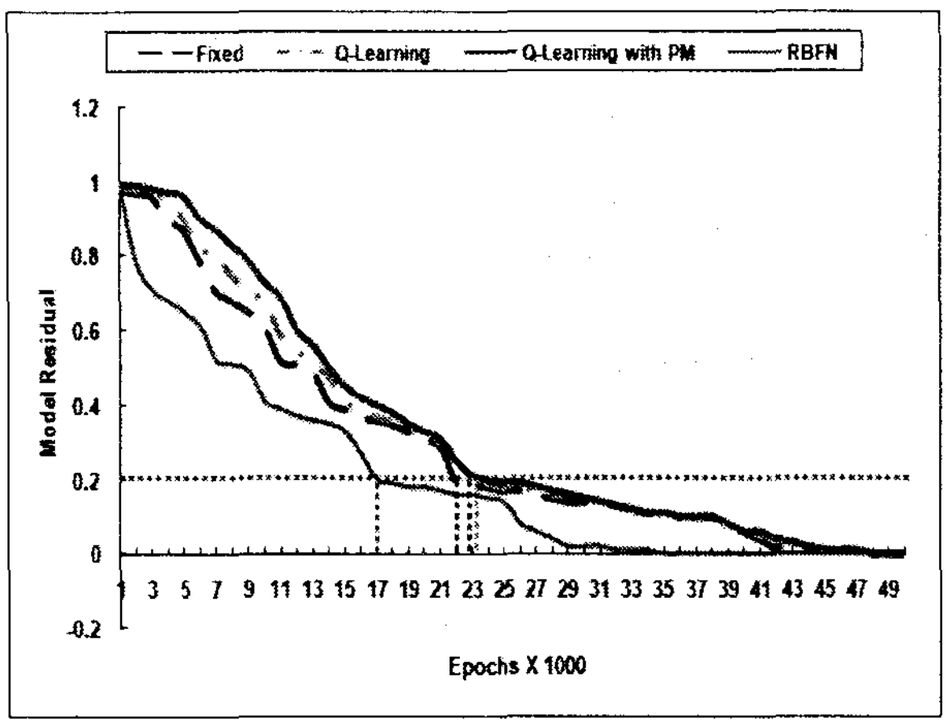

그림 5. 모델 오차 비교

그림 4 는 쥐가 서로 다른 정책을 수행하는 경우의 게임 지속 시간을 보여주고 있다. 이 그림을 통해 확인할 수 있는 사실은 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.

4.2.3 상대 정책 모델 수렴

상대 에이전트의 정책에 따른 상대방 정책 모델 PM 의 수렴속도를 분석해 본다. 이를 위해 서로 다른 정책들(고정 정책, 상대방 모델이 없는 Q 학습, PM 기반의 Q 학습, RBFN 기반의 Q 학습)을 사용하는 상대 에이전트들과 RBFN 을 이용한 PM 기반의 학습 에이전트 간의 게임들을 수행하면서 식 12 와 같이 모델 오차를 계산해 보았다.

$$PE_t = \max_{s \in S} \left| P_t(s) - P_{t-1}(s) \right| \qquad \text{[식 12]}$$

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

그림 5 에서는 상대 에이전트인 고양이의 서로 다른 정책 별로 모델 오차(model residual)의 변화를 나타내고 있다. 그림 5 의 실험결과로부터 알 수 있는 한 가지 사실은 고양이가 네 가지 정책 중 어떤 것을 쓰든지 상관없이 게임이 거듭됨에 따라 모델 오차가 0 에 가까워졌다. 이것은 곧 실험에 적용된 고양이의 서로 다른 네 가지 정책 모두에 대해, 쥐가 습득하는 RBFN 을 통한 상대방 정책 모델이 수렴할 수 있다는 것을 의미한다. 그림 5 를 통한 또 다른 결과는 RBFN 을 통한 상대 정책 모델이 상대적으로 빨리 수렴하는 것을 볼 수 있다. 이는 실험 환경이 짧은 에피소드를 가지기 때문이다. 그리고 단점으로는 이전에 했던 행동만을 되풀이 할 수 있다. 이를 해결하기 위한 방법을 모색해야 한다.

5. *결론*

본 논문에서는 적대적 멀티 에이전트 환경에서 상대 에이전트의 행동들에 의한 RBFN 을 통해 상대 에이전트의 행동 정책 모델인 PM 을 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 비교적 간단한 형태의 RBFN 을 통해 학습함으로써 학습의 효율성을 높인 것이 특징이다. 본 논문에서는 대표적인 적대적 멀티 에이전트 환경인 고양이와 쥐(Cat and Mouse) 게임을 테스트 베드로 삼아 다양한 비교 실험들을 전개하여 본 논문에서 제안한 RBFN 에 기반한

상대방 정책 모델 기반의 멀티 에이전트 강화 학습의 효과를 분석해보았다. 이 실험을 통해 RBFN 을 통한 상대방 정책 모델을 이용하는 것이 강화 학습의 효율성과 에이전트의 성능 향상에 도움이 되며, 상대 에이전트가 고정 정책을 쓰는 경우는 물론 Q 학습을 하는 경우에도 RBFN 을 통한 상대방 정책 모델 PM 의 수렴 성을 확보할 수 있다는 것을 확인하였다.

참고 문헌

[1] Sutton, R.S., Barto A.G. (1998), "Reinforcement Learning: An Introduction," MIT Press

[2] Seiichi Ozawa and Naoto Shiraga (2003), "Reinforcement Learning Using RBF Networks with Memory Mechanism," in Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 1149-1156

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

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

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

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

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

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

[9] Acharyya S. (2000), "Learning radial basis function based soccer strategies using ideal opponent model." Kanpure, India: Indian Institute of Technology

[10] Mark J. L. Orr (1996), "Introduction to Radial Basis Function Networks," Centre for Cognitive Science, University of Edinburgh

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