제26회 한국정보처리학회 추계학술발표대회 논문집 제13권 제2호 (2006. 11)

# 로보컵 공 뺏기 테스트베드를 이용한 적대적 학습 에이전트들에 대한 실험적 분석

권기덕*, 김인철**
*, **경기대학교 전자계산학과
e-mail:{kdkwon*, kic**}@kyonggi.ac.kr

# Empirical Analysis of Adversarial Learning Agents Using the RoboCup Keepaway Test-bed

Ki-Duk Kwon*, In-Cheol Kim**
*,** Dept. of Computer Science, Kyonggi University

요 약

강화 학습은 시행착오를 통해 동적 환경과 상호작용하면서 학습을 수행하는 학습 방법으로 본 논문에서 테스트 환경으로 사용하는 Keepaway와 같은 동적인 환경에서 주로 사용하는 학습 방법이다. 본 논문에서는 학습을 통한 에이전트가 다른 정책을 사용하는 에이전트보다 성능이 더 높다는 것을 보이고자 한다. 학습 초기에는 다양한 전략을 평가하기 위해 최적이 아닌 행동을 선택하지만 시간이 지남에 따라 최적의 행동 선택에 수렴해 가는 것을 보이기 위한 실험을 수행한다. 이 실험을 통해 고정된 행동 양식을 가지는 정책보다 강화 학습을 이용한 에이전트들의 성능이 더 효과적인 것을 알 수 있었다.

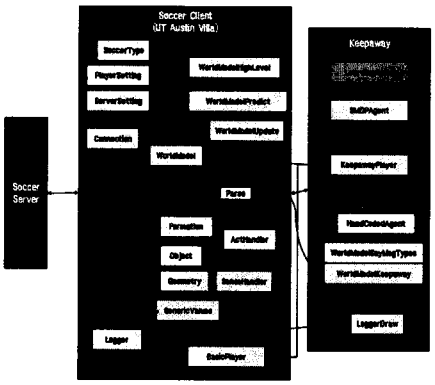
## 1. 서론

학습이란 단순하게 데이터를 인식하는 것이 아니라 새로운 정보를 통해서 얻어진 지식을 수정할 수 있어야 하고 보다 정확한 지식으로 개선해 나가는 능력을 갖추게 하는 것이다. 강화 학습은 동적 프로그래밍과 교사 학습을 혼합한 형태의 학습 방법으로 학습을 하는 에이전트는 목표에 도달할 수 있는 전략을 찾기 위해서 미지의 환경과 시행착오를 통해 상호 작용을 하면서 얻은 경험을 통해 점진적으로 학습한다[2]. 에이전트의 학습 목표인 최적의 전략은 상태와 행동의 쌍으로 된 형태이다[3]. 강화 학습은 가치함수를 계산하고 가치함수를 이용하여 최적의 전략을 구한다[5]. 가치함수는 "에이전트가 현재 상태에 있는 것이 목적 상태에 도달하는데 어느 정도 도움이 되는가?" 또는 "에이전트가 현재 상태에서 특정 행동을 수행하는 것이 어느 정도 가치가 있는가?"를 평가한다. 상태 또는 상태-행동 쌍의 값은 일련의 보상 값의 합을 이용하여 구할 수 있다. 전략 $\pi$의 행동보다 더 나은 행동이 있는지 알아보기 위해서는 다른 행동들을 시도해 보아야 한다. 그러므로 강화 학습은 동적 환경에서 성능이 좋은 학습이 이루어지기 위해 학습 초기에는 성능이 다소 떨어지더라도 최적의 행동이 아닌 다른 행동들도 수행하여야한다. 고정된 정책을 사용하는 경우 더 나은 행동이 있음에도 불고하고 정해진 행동에 의해서 행동이 결정되므로 행동에 대한 평가가 이루어지지 않는다. Keepaway는 실시간 환경을 제공하는 로보컵 축구 시뮬레이션 리그를 이용한 기계 학습 테스트용으로 만들어진 도구이다[4]. Keepaway를 이용하여 학습을 하는 정책과 학습을 하지 않고 고정된 정책을 사용하는 것을 비교 실험함으로써 학습을 이용한 에이전트의 성능이 더 우수함을 보이고자 한다. 또

한 미지의 환경 요소로서 적을 고려하여 학습을 함으로써 적대적 학습을 수행하는 에이전트들 간의 비교를 통해 성능을 평가한다.

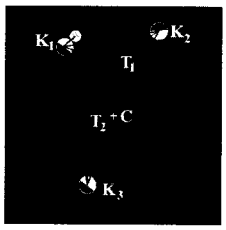
## 2. 다중 에이전트 환경

Keepaway는 로보컵 축구 시뮬레이션 리그를 이용한 기계 학습 테스트용으로 만들어진 도구이다. Keepaway는 Keeper와 Taker로 구성되며, 로보컵 축구 시뮬레이션 리그의 제한된 영역을 사용한다. Keeper의 목표는 공을 점유, 유지하는 것이고, Taker의 목표는 공을 뺏는데 있다. 파라미터는 영역의 크기, Keeper와 Taker의 개수이다. Keepaway는 Keeper가 공을 빼앗기거나 영역 밖으로 나갔을 때를 하나의 에피소드로 한다.




(그림 1) Keepaway 전체 구성도

그림 1은 본 논문의 실험 환경인 Keepaway의 전체적인 시스템 구조를 나타낸 것이며, 로봇 축구 시뮬레이션 리그 게임이 제공하는 실시간 환경을 기반으로 강화 학습을 실험하기 위한 환경으로 변형한 환경을 가지고 있다.




(그림 2) 3대2 Keepaway의 상태 변수들

Keepaway의 각 에이전트는 자신의 행동을 선택함으로써 전체 팀의 목표를 달성하기 위한 협력이 이루어지도록 하였다. 상태 변수는 Keeper를 중심으로 다음과 같은 기준에 의해 13개의 상태 변수를 정의한다.

- 중앙에서 모든 에이전트들까지의 거리들
- 공을 소유하고 있는 K1을 기준으로 다른 에이전트들까지의 거리들
- 근접한 적으로부터 동료들까지의 거리들
- 각각의 Keeper 에이전트들에 대해 적과 동료들 사이의 공을 소유하고 있는 K1을 바라 볼 수 있는 최소 각도들

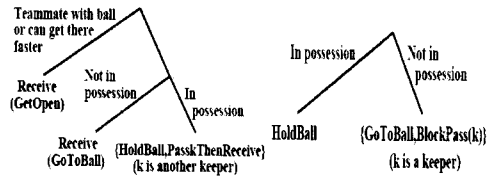
또한 Taker의 경우는 전체적으로는 Keeper의 상태 변수와 비슷하지만 공을 빼앗기 위해 공을 소유하고 있는 Keeper 에이전트를 중심으로 다른 Keeper 에이전트들과의 거리에 대한 중간 값을 정의함으로써 공을 빼앗는 행동을 하기 위한 좋은 위치로의 이동이 가능하다. Taker 에이전트는 다음의 기준에 따라 13개의 상태 변수를 정의한다.

- 중앙에서 모든 에이전트들까지의 거리들
- 공을 소유하고 있는 K1을 기준으로 다른 에이전트들까지의 거리들
- 각각의 Taker 에이전트들에 대해 공을 소유하고 있는 K1과 다른 Keeper들과의 거리에 대한 중간 값

연속적인 상태 공간을 간략화 하기 위해 유사한 상태들을 일반화 시키는 함수 근사 방법으로 타일 코딩 방법을 사용한다. 각 타일은 고정된 같은 크기를 가진다. 상태 공간의 각 차원에 타일링이 분배되고 각 타일링은 타일들로 분할된다. 각 타일링에서 그 상태-행동 쌍을 포함하는 타일이 활성화되면 활성화된 타일의 값의 합이 해당 상태-행동 쌍의 Q값이 된다[1]. 본 논문의 실험 환경인 Keepaway의 특징 집합(Feature Set)은 3대2 Keepaway의 경우, 13개의 상태 변수와 32개 타일링의 조합인 13 X 32 = 416 으로 구성된다.

본 논문에서 제안한 적대적 학습을 하는 에이전트들은 공을 점유, 유지하기 위해 3개의 Keeper 에이전트들과 2개의 Taker 에이전트들이 각자의 행동을 자율적으로 선택하여 실행하는 에이전트 특성을 가지도록 하였다. Keepaway에서 학습자는 로보컵 축구 시뮬레이션 리그의 기본적인 행동 뿐 만 아니

라 CMUnited99 팀에서 사용한 상위 레벨의 매크로 행동을 선택한다[3]. 상위 레벨의 매크로 행동에는 HoldBall(), PassBall(k), GetOpen(), GoToBall(), BlockPass(k)가 있으며 이를 이용한 Keeper와 Taker의 정책은 다음과 같다.




(그림 3) Keeper(좌)와 Taker(우)의 정책 공간

그림3은 Keeper와 Taker의 행동을 선택하는 정책에 대한 그림이다. 크게 공을 점유하고 있을 경우와 점유하고 있지 않을 경우로 나누어 정책을 수립한다. 매크로 행동을 기반으로 Keepaway에서 수행하는 실제적인 행동은 공을 점유하고 있는가와 공을 누구에게 패스할 것인가로 전체 목표를 달성하기 위한 행동을 결정짓는다.

## 3. 다중 에이전트 행동 정책

### 3.1 Q 학습

각 에이전트($A_i$)는 공을 점유, 유지하기 위해 상태 공간으로부터 에이전트가 선택한 행동($a_t$)에 대한 보상으로 보상 값($r^i_{t+1}$)과 다른 에이전트들에 대한 상태 정보($s^i_t$)를 입력 받아 상태 전이 함수(h)에 의해 최적의 행동을 선택하여 다음 상태($s^i_{t+1}$)로 이동한다. 각 에이전트는 현재 상태($s^i_t$)에서 선택 가능한 모든 (상태-행동)쌍에 대한 평가 값을 [식 1]과 같이 갱신한다.

$$For\ all(s_t,a_t) \qquad a\in A(s_t) \qquad \text{[식 1]}$$
$$Q(s_t,a_t)=(1-\alpha)Q(s_t,a_t)$$
$$+\alpha[r_{t+1}+\gamma\max Q(s_{t+1},a_{t+1})-Q(s_t,a_t)]$$

[식 1]에서 α는 학습 율, γ는 할인 율, 그리고 r은 에이전트가 선택한 행동에 대한 평가를 나타내는 보상 값이다. 행동의 선택은 [식 2]와 같은 확률에 의해 확률적으로 결정된다.

$$P(a^i_t|s_t)=\frac{Q(s_t,a^i_t)}{\sum_{i=0}Q(s_t,a^i_t)} \qquad \text{[식 2]}$$

[식 2]의 $P(a^i_t|s_t)$는 상태 $s_t$에 행동$a^i_t$를 선택할 확률을 나타내며, 이 확률은 이 상태에 적용 가능한 모든 행동들의 Q값의 합에 대한 행동$a^i_t$의 Q값의 크기에 비례하여 결정된다. 강화학습 에이전트는 매 순간 적용 가능한 행동들에 대한 Q값을 기초로 이와 같은 확률을 계산하고 이 확률에 따라 행동을 선택한다. 이러한 방법은 일반적으로 룰렛 휠 방식이라 부르며, 단순히 Q값이 최대인 행동을 선택하는 방법보다 경험해보지 않은 낮은 Q값의 행동도 선택될 수 있도록 해주는 것이 특징이다.

### 3.2 Hand-Coded

그림 4에서 알 수 있듯이 이 정책은 공을 오래 가지고 있지 않고 바로바로 패스하지만 적이 가까이 와야지만(4 m) 패스하는 형태를 나타내도록 하였다. 이 정책의 경우 학습의 초기에 발생할 수 있는 탐험과 탐색의 문제를 해결한 것처럼 보이지만 주어진 상황이 계속적으로 발생한다면 계속적으로 같은 행동만을 보임으로써 효율적이지 못하다.

```
If no taker is within dist(K1,T1) > 4 Then
    HoldBall()
Else
    For i∈[2,n]
        Vi ← Min(ang(Ki,K1,T1), ang(Ki,K1,T2)) +
            α * Min(dist(Ki,T1), dist(Ki,T2))
    I ← argmaxiVi
    If VI > β Then
        PassBall(K1)
    Else
        HoldBall()
```

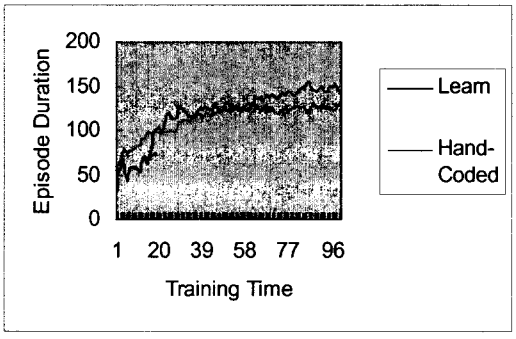
(그림 4) Hand-Coded 정책

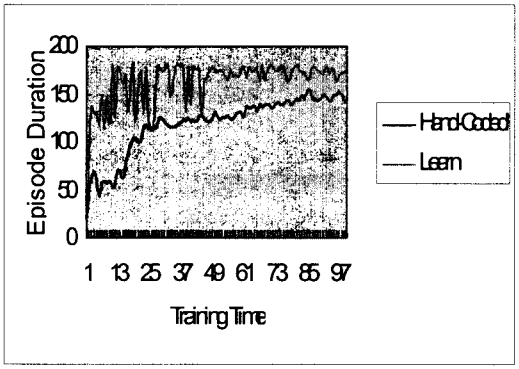
## 4. 실험

본 논문의 실험 환경인 Keepaway는 기계 학습을 실험하기 위한 테스트하기 좋은 환경을 제공한다. 특히 강화 학습을 테스트하기에 좋은 환경을 제공하는데, 그 이유는 상태 공간이 전체적으로 탐험하기에 매우 크며, 각 에이전트가 단지 부분적인 상태 정보만을 가지고 있기 때문이다. 또한, 행동 공간은 연속적인 값을 가지기 때문이다.

본 논문에서는 Q 학습을 기준으로 크게 Keeper를 기준으로 Taker가 고정된 학습 정책을 사용하는

경우와 Keeper와 마찬가지로 Q 학습을 사용하는 경우를 실험한 것과 다른 하나는 Keeper가 학습을 하지 않고 고정된 정책을 사용할 경우 Taker가 학습을 하는 경우와 학습을 하지 않는 경우로 나누어 비교 실험하였다.




(그림 5) Keeper의 학습과 비 학습간의 정책 비교

그림 5의 에피소드 기간은 Keeper의 공을 소유하고 있는 시간을 의미한다. 그림 5의 윗 그림은 Keeper가 학습을 하고 있는 경우 Taker가 학습을 하는 경우와 학습을 하지 않고 Hand-Coded 정책을 사용하는 경우를 비교한 그림이다. 이 경우는 학습 초기엔 공을 점유하는 시간이 다소 짧은 것을 볼 수 있으나 시간이 지날수록 최적의 행동 선택 정책에 의해 공을 소유하는 시간이 길어지고 있음을 볼 수 있다.

Keeper가 학습을 수행하는 경우 Hand-Coded는 최적의 행동 정책에 수렴해 가는 것을 볼 수 있지만 학습을 한 keeper 에이전트에 비해 수렴 속도가 늦는 것을 볼 수 있었다.

그림 5의 아래 그림은 Keeper가 학습을 하지 않고 Hand-Coded 정책을 사용한 경우이다. 여기에서는 Taker가 학습을 수행하는 경우가 학습을 하지 않는 경우보다 빠르게 수렴하는 것을 볼 수 있다. 이것은 Keeper가 고정된 정책만을 사용함으로써 항상 정해진 행동만을 수행하기 때문이다.

## 5. 결론

본 논문에서는 적대적 학습을 이용한 다중 에이전트들 간의 행동 정책을 실험하기 위해 Keepaway를 이용하여 성능을 평가하였다. 크게 학습을 수행하는 에이전트와 학습에 의한 정책을 수행하지 않는 에이전트로 나누어 실험하였다. 실험을 통해 학습을 수행한 에이전트가 학습을 수행하지 않은 에이전트보다 학습 초기에는 탐색의 기회를 더 많이 함으로써 학습 율이 낮은 것을 볼 수 있었지만 오랜 경험을 통해 최적의 행동을 선택함으로써 효과적인 학습이 이루어졌음을 확인 할 수 있었다.

## 참고문헌

[1] Alexander A. Sherstov and Peter Stone. Function Approximation via Tile Coding: Automating Parameter Choice, In Proc. Symposium on Abstraction, Reinforcement, and Approximation(SARA-05), 2005

[2] Bertsekas, Dimitri P., Dynamic Programming and Optimal Control, Athena Scientific, Belmont, Massachusetts. Vol.1 and 2, 1995

[3] David McAllester and Peter Stone. Keeping the ball from CMUnited-99, RoboCup-2000: Robot Soccer World Cup IV, Springer Verlag, Berlin, 2001

[4] Gregory Kuhlmann and Peter Stone, Progress in Learning 3 vs. 2 Keepaway, In RoboCup-2003: Robot Soccer World Cup VII, 2004

[5] Junling Hu, Michael P. Wellman, Multiagent Reinforcement Learning: Theoretical Frmaework and an Algorithm, In Proceedings of the 15th International Conference on Machine Learning, pp.242-250, Madison, WI, USA, July 1998