韓國 컴퓨터情報學會 論文誌
第11卷 第4號, 2006. 9.

2006-11-4-1-3

강화학습 기반의 지능형 게임에 관한 연구

우 종 우*, 이 동 훈**

A Study on the Intelligent Game based on Reinforcement Learning

Chong-Woo Woo*, Dong-Hoon Lee**

요 약

인공지능 기법을 이용한 컴퓨터 게임에 대한 학술적 연구는 오랫동안 이루어져 왔으며, 주로 게임에 대한 숙련도를 높여서 인간에게 승리 하는 것이 주요 연구 목적이었다. 그러나 최근의 상업용 게임에서는 게임의 흥미를 제공하기 위해서, 사용자의 적응을 목적으로 개발하고 있다. 본 논문에서는 기존의 강화학습 알고리즘을 수정하여 사용자 적응에 중점을 둔 적응형 강화 학습 알고리즘을 제안하였다. 실험대상으로는 많은 상태공간을 가진 오델로 게임을 대상영역으로 하여 시스템을 설계 및 구현하였다. 시스템의 성능측정은 두개의 강화학습 알고리즘이 각각 Min-Max 알고리즘과 대결하는 방식으로 실험을 하였으며, 결과는 기존의 강화 학습 알고리즘과의 대결에서도 향상된 학습율을 나타내었다.

Abstract

An intelligent game has been studied for some time, and the main purpose of the study was to win against human by enhancing game skills. But some commercial games rather focused on adaptation of the user's behavior in order to bring interests on the games. In this study, we are suggesting an adaptive reinforcement learning algorithm, which focuses on the adaptation of user behavior. We have designed and developed the Othello game, which provides large state spaces. The evaluation of the experiment was done by playing two reinforcement learning algorithms against Min-Max algorithm individually. And the results show that our approach is playing more improved learning rate, than the previous reinforcement learning algorithm.

▸ Keyword : 지능형 게임 (Intelligent Game), 강화학습 (Reinforcement Learning), 기계학습 (Machine Learning)

• 제1저자 : 우종우
• 접수일 : 2006.07.07, 심사일 : 2006.08.16, 심사완료일 : 2006.09.23
* 국민대학교 컴퓨터학부 교수 ** 국민대학교 컴퓨터학부 대학원
* 본 논문은 2006학년도 국민대학교 교내연구비 지원을 받아 수행되었음

Ⅰ. 서 론

최근 컴퓨터 게임은 그래픽과 사운드를 중시하는 관점에서 벗어나, 보다 자연스럽고 재미있는 기능에 대한 관심이 고조 되고 있으며, 이러한 사용자들의 요구를 Finite State Machine이나 Decision Tree 등 인공지능의 여러 기법들을 적용함으로서 해결하려는 연구가 진행되고 있다 [1]. 이러한 전통적인 기법들은 게임 설계자의 경험적 규칙에 주로 의존하게 되어, 체계적이지 못한 점과 방대한 데이터의 처리 등이 문제로 제기 되고 있으며, 이에 따른 해결방안으로 최근에는 기계학습의 알고리즘들을 적용하여 성능향상을 하려는 연구가 진행되고 있다[2]. 기계 학습 알고리즘을 적용한 대부분의 지능형 게임관련 연구들은 게임의 재미보다는 인공지능 기법의 성능향상에 중점을 두는 연구이기 때문에 사용자들의 흥미를 유발시키기 위한 보다 다양한 방법이 요구된다.

본 논문에서는 기존의 강화학습 알고리즘을 수정하여 사용자 적응을 기반으로 접근한 강화 학습 알고리즘을 제안하고, 인공지능의 기법들을 비교적 쉽게 적용 시킬 수 있는 오델로 게임을 대상영역으로 선정하여 시스템을 설계 및 구현하였다. 본 연구의 실험은 Min-Max 알고리즘의 탐색 깊이를 조정하여 사용자의 게임 숙련도에 따른 적응성을 분석하고, 강화학습알고리즘과 본 연구에서 제안하는 알고리즘 각각 대결하여 승율을 비교 분석 하였다.

본 논문의 구성은 다음과 같다. 제 2장에서는 오델로 게임 및 강화 학습등 관련연구에 대하여 기술한다. 제 3장에서는 본 논문의 시스템에 대한 설계 및 구현에 대해 기술한다. 제 4장에서는 실험 방법 과 실험 후 결과의 분석에 대해 기술한다. 제 5장에서는 결론 및 향후 연구 과제를 기술한다.

Ⅱ. 관련연구

본 장에서는 시범영역인 오델로 게임에 관한 기존의 연구결과에 대하여 분석하고, 본 연구의 주요 접근기술인 강화 학습 및 Q-learning 알고리즘에 대하여 기술한다.

1. 사례 연구

기존의 지능형 게임 연구에서는 인공지능 기법의 수행능력에 대한 연구가 주로 수행되었기 때문에 알고리즘의 적용성이 편리한 오델로와 같은 보드게임에 관한 많은 연구가 수행되었다. 반면 재미를 목적으로 하는 상업용 게임에서는 사용자의 행동패턴을 수집한후 이를 기반으로 게임의 요소를 변경하는 방식으로 접근하고 있다. 대표적인 몇가지 연구결과는 다음과 같다.

1.1. BILL

BILL 프로그램은 베이지안 알고리즘을 이용한 오델로 게임 프로그램이다[3]. 주요관점은 평가 함수를 이용하여 오델로 게임의 승패에 큰 영향을 주는 중반부인 25번째부터 45번째 사이의 모든 행동에 대해 승리 확률을 평가하여, 최고의 확률을 가진 행동을 선택한다. 이 평가 함수는 3000번의 게임 수에 대해 데이터베이스를 구축하여 학습한다. BILL은 수집된 학습 데이터의 질에 의해 게임의 지능 수준이 달라지기 때문에, 만약 수집된 데이터가 초보자 수준의 데이터라면 BILL 프로그램의 지능성은 초보자 수준에 머무르게 된다.

1.2. Focus Neural Network

Focus Neural Network는 신경망 알고리즘의 뉴런을 Genetic Algorithm에 적용하여 학습을 한다[4]. 즉, 신경망의 구축을 평가하여 보다 나은 신경망 구축을 위해 학습을 한다. 그러나 학습을 할 때 다른 알고리즘의 도움을 받아야 하며, 학습된 인공지능 수준은 도움 받은 인공지능 알고리즘과 비슷한 수준으로 학습되는 단점이 존재 한다.

1.3. SIMs

가상의 아바타인 SIM이 다른 SIM과 어울려 살아가는 일상생활 모습에 관한 시뮬레이션 게임이다[5]. SIM은 처음에는 사용자의 제어가 필요하지만 게임이 진행되어 감에 따라 사용자의 제어를 학습하여 스스로 행동하게 된다. 예를 들면, 처음 SIM은 피로수치가 커도 사용자의 제어가 있어야 잠을 자지만, 게임이 진행됨에 따라 사용자의 제어를 학습하여 피곤하면 스스로 침대를 찾아 잠이 들게 된다.

2. 강화학습 알고리즘

2.1 알고리즘 개요

강화 학습은 환경과 에이전트와의 상호 정보교환을 통하여 에이전트의 행동을 학습해 나가는 학습방법이다. 일반적인 교사 학습은 예상되는 상황에 대해 예측하고, 예측된 상

황에 대한 대응방법을 제시하는 반면, 강화학습은 주어진 상태에 대해 자유로운 선택을 하고 선택의 결과를 제시함으로서 차후에는 보다 나은 선택을 할 수 있게 유도한다는 점이다[6][7].

주어진 환경과 에이전트간의 상화 작용을 위해서는 상태, 행동, 보상의 세 가지의 요소가 필요하게 되는데, 상태는 주어진 환경에 대한 정보, 행동은 적절한 정책에 따라 결정된 최적의 행동, 보상은 환경에 적용한 행동의 평가로 나타낸다.

[그림 1]는 강화 학습의 수행 과정을 간략히 도식화 한 것이다. 현재의 시간(t)에 대한 환경의 정보(st)를 통해 현재의 상황에서 행동 가능한 모든 행동(A(st))에 대한 정보를 에이전트에 전달한다. 에이전트는 이러한 정보를 바탕으로 최적의 행동(a)을 선택하고, 이것을 환경에 적용시킨다. 강화 학습 에이전트는 변경되는 새로운 상태 정보(st+1)와 환경에 적용시킨 행동(a)에 대한 평가 값인 보상(rt)을 받는다. 이러한 과정을 반복해서 수행하면서 에이전트가 점차 학습 되어 나간다. 이론 적으로는 학습한 지식과 받은 보상 간의 차이를 고려하여 학습하지만, 실제 구현된 강화 학습 에이전트의 대부분은 최적의 행동(a)을 선택할 때 보상(rt)을 미리 구하여 강화 학습 에이전트를 학습한다[8].

[그림 1] 강화 학습 수행 과정
Fig 1. Process of Reinforcement Learning

2.2. Q-learning 알고리즘

2.2.1. Q-function

Q-learning 알고리즘은 Q(s,a)의 값을 통해 최적의 정책(policy)을 찾아내는 강화 학습 알고리즘이다. Q(s,a)의 값은 주어진 상황에서의 행동(a)로 인해 얼마나 좋은지를 수치화 한 것이다[8]. Q(s,a)는 Q-function으로 불리기도 한다. Q-function은 받은 보상에 의해 주어진 환경과 행한 행동에 맞는Q-function 값을 변경한다. [수식 1]은 보상에 의한 Q-function 값의 변화를 수식으로 나타낸 것이다. δ(s,a)는 환경의 상태 s에 행동 a를 적용시킨 후의 환경을 가리킨다.

$$Q(s,a) \equiv r(s,a) + \gamma V^*(\delta(s,a)) \quad \cdots\cdots 1$$

$$\pi^*(s) = \arg\max Q(s,a) \quad \cdots\cdots 2$$

[수식 2]는 Q-function 값에 의해 변경된 최적의 정책을 찾아내는 수식이다. Q-function은 가치 함수를 대체할 수 있다. 보상 함수 r(s,a)와 환경을 변경 시키는 함수 δ(s,a)가 필요하지 않게 된다. 즉 보상과 환경 변환을 한 상태 정보를 가지고 있지 않아도 Q-function을 값을 통하여 최적의 정책을 이용하여 행동을 선택할 수 있다는 것을 나타낸다.

2.2.2. Q-Learning 알고리즘

에이전트와 환경간의 상호작용이 반복 될수록 에이전트가 학습되기 위해서는 정책을 결정하는 주요 요소인 Q-function이 학습해야 하며, 학습을 하기 위해서는 학습 데이터가 필요하다. Q-Learning은 교사 학습과 같이 정형화된 학습 데이터를 제시하는 것이 아닌 수식을 통해 학습 데이터를 예측, 예측된 데이터를 이용하여 Q-function을 학습한다.

$$V^*(s) = \max_{a'} Q(s,a') \quad \cdots\cdots 3$$

$$Q(s,a) \equiv r(s,a) + \gamma \max_{a'} Q(\delta(s,a).a') \quad \cdots\cdots 4$$

[수식 3]는 최적의 정책에 대한 가치 함수 V^*과 Q-function과의 관계를 나타낸 수식이다. [수식 3]과 [수식 1]을 이용하면 Q-function은 [수식 4]로 정의할 수 있다. [수식 4]에 의해 정의된 Q-function은 재귀적인 성질을 가지고 있으며, 수식에서 나타내는 a'는 δ(s,a)에 의해 변화된 환경에서 실행 가능한 모든 행동을 나타낸다.

Q-learning은 [수식 4]에 정의한 Q-function을 기반으로 한다. 에이전트는 Q-learning 알고리즘을 이용하여 Q-function을 학습하면서 Q-function에 점차 근접해나간다. [수식 4]에 의해 나온 Q-function 값은 실제 Q-function의 값이 아닌 추정치이며 추정된 Q-function을 $\hat{Q}$ 또는 $\hat{Q}$(s,a)로 나타낸다. 에이전트는Q-function 학습 과정을 반복해서 수행하는데 각 반복단계에서현재의 환경을 s, 에이전트에 의해 선택된 행동을 a, 행동에 대한 보상 값을 r

= r(s,a), 환경에 행동을 적용한 결과는 s' = δ(s,a)로 정의한다면 [수식 6]을 기본으로 하여 [수식 5]의 룰을 정의한다.

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a') \quad \cdots\cdots 5$$

[그림 2] 는 [수식 5]을 기반으로 하여 만들어진 Q-learning 알고리즘을 나타낸다. Q-learning은 정책을 기반으로 하여 현재 상황에 적합한 행동을 선택하여 수행한 뒤에 보상 값을 받아 [수식 5]에 의해 Q-function 값을 학습하는 과정을 통해 Q-learning이 이루어진다.

2.2.3. 신경망을 이용한 Q-learning

Q-function은 Q-table을 가지고 있다. Q-table은 단순한 저장 공간으로 행동 가능한 모든 경우의 행동에 대한 Q-function의 값을 1:1 매칭되게 저장하고 있다. 이럴 경우 상태 공간이 큰 문제일 경우 저장 공간이 점차 커지는 문제를 가지고 있다. 또한 Q-table이 커질수록 조건에 맞는 Q-function 값을 검색하는 시간은 점차 많아진다.

본 논문에서 적용하는 오델로 게임의 경우 상태 공간이 약 1028정도를 가진다. 즉 1028개의 Q-function 값을 저장하는 Q-table을 만들어야 하기 때문에, 공간적인 측면이나 검색 시간적인 측면에서 매우 비효율적이다.

For all states s ∈ S and all actions a ∈ A
initialize $\hat{Q}$(s', a) to an arbitrary value
Repeat (for each trial)
Initialize the current state s
Repeat (for each step of trial)
Observe the current state s
Select an action a using a policy π
Execute action a
Receive an immediate reward r
Observe the resulting new state s'
Update $\hat{Q}$(s',a) according to [수식 7]
s ← s'
Until s is a terminal state

[그림 2] Q-learning 알고리즘
Fig 2. Q-Learning Algorithm

본 논문은 이러한 문제를 해결하고자 신경망을 사용하였다. 그 이유는, Q-table에서 얻어지는 Q-function 데이터 값은 [수식 4]에 의해 추정된 Q-function 값이 아닌 추정된 Q-function 값에 대한 근사치이다. 따라서, 신경망은 학습을 통해 추정된 Q-function 값의 근사치를 구할 수 있다 [9].

[그림 3]은 신경망을 적용한 Q-learning 알고리즘에 대한 설명이다. Backpropagation 알고리즘을 이용하여 신경망을 통해 나온 Q-function에 대한 근사치와 Q-function의 추정 값의 차이를 이용하여 Q-learning을 수행한다.

III. 설계 및 구현

1. 설계원칙

본 연구에서는 게임의 학습효율 향상을 위해 다음 두 가지 요소를 기반으로 설계하였다. 첫째는, 학습 및 행동 선택에 있어 미래에 대한 예측 요소의 추가이며, 두 번째는 게임의 승패가 학습에 미치는 영향이다.

1.1 예측 정보

강화 학습은 현재 환경에 적용된 행동에 대한 보상을 이용하여 Q-learning을 수행한다. 즉 보상 값의 여부에 따라 학습이 효율에 영향을 미친다. 본 논문은 현재의 행동 외에 미래의 행동에 대해서도 학습에 고려한다는 점을 제안한다. 즉 게임을 할 때 승리를 위해서는 현재의 상황에 대해서만 고려하는 것이 아닌 미래의 상황에 대한 예측이 필요하다

본 논문에서는 기존의 강화 학습 모듈과 더불어 예측 모듈(Pre-selection Module) 을 추가함으로써 학습 효율의 향상을 도모하였다[그림 4 참조]. 예측 모듈은 게임 상대의 움직임에 대해 예측을 한다. 즉, 과거 게임을 통해 얻어지는 행동을 예측 모듈이 학습하여 학습된 결과를 바탕으로 상대방의 행동을 예측하는 방법을 사용하였으며, 이러한 학습을 위해 신경망 알고리즘을 사용하였다. 사용자가 기본적인 규칙만을 인지하여 게임을 하고 있다고 가정하였고 대부분의 사용자가 처음부터 게임에 능하지 않으며, 또한 인간은 학습 능력을 가지고 있기 때문에 신경망을 적용하였다.

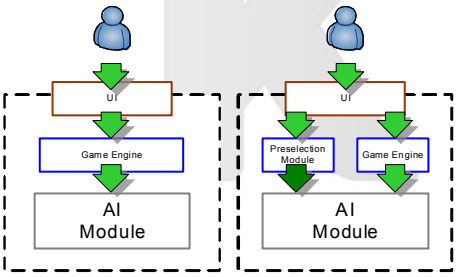
Initialize all neural network (NN) weights to small random numbers
Repeat (for each trial)
Initialize the current state st
Repeat (for each step of trial)
Observe the current state st
For all actions a' in s use the NN to compute $\hat{Q}$ (st, a')

Select an action a using a policy π
Qoutput ← $\hat{Q}$ (st, a)
Execute action a
Receive an immediate reward r
Observe the resulting new state s st+1
For all actions a' in st+1 use the NN to compute $\hat{Q}$ (st+1, a')
According to [수식 7] Compute Qtarget ← $\hat{Q}$ (s, a)
Adjust the NN by backpropagating the error (Qtarget-Qoutput)
st ← st+1
Until s is a terminal state

[그림 3] 신경망을 적용한Q-learning 알고리즘
Fig 3. Q-Learning Algorithm using Neural Net

$$\hat{Q}(s,a) = r + \gamma(r' + \max_{a'}(\hat{Q}(s',a')) \quad \cdots\cdots 6$$

[수식 6]은 [수식 5]를 기반으로 하여 변형된 식이다. [수식 5]와 다른 점은 r'이며, r'는 미래의 행동인 s'에 대해 예측 모듈이 제시한 행동에 대한 보상 값이다.



[그림 4] 기존의 시스템(좌)과 본 논문의 시스템(우)
Fig 4. Previous System(left) and Our Approach(right)

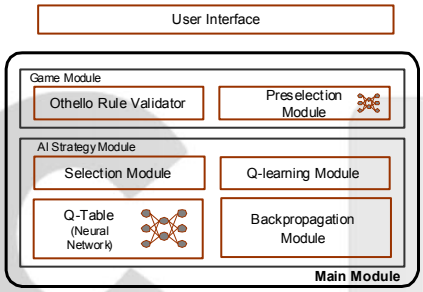
[그림 4]는 기존의 시스템과 본 논문에서 설계된 시스템 간의 차이를 나타낸다. 기존의 시스템에서는 AI module이 사용자와 진행하고 있는 현재 게임 상태만 가지고 적절한 행동을 선택한 반면, 본 논문에서 설계된 시스템은 현재 게임의 상태와 Pre-selection Module에서 예측한 게임 진행 상태 즉 두 가지 게임의 상태 정보를 이용하여 적절한 행동을 선택한다.

1.2 가중치

강화 학습은 게임의 각 상황에 대한 대처 방법에 대해 학습한다. 즉 게임의 승패는 학습에 영향을 미치지 않는다. 반면, 본 논문의 접근은 게임의 승패가 학습에 영향을 미쳐야 한다. 즉, 게임의 결과가 강화 학습 모듈의 승리 일 경우 반드시 혜택을 주어야 한다. 본 논문의 시스템에서는 게임 도중의 모든 행동에 대해 저장을 하고 게임이 승리했을 경우 저장된 로그를 바탕으로 다시 한번 학습을 하여 승리에 대한 혜택을 주었다.

2. 시스템 구조

[그림 5]는 본 연구에서 설계한 시스템의 구조이다. 시스템은 크게 User Interface, Main Module로 구분되며, Main Module은 여러 개의 세부 모듈로 구성된다. 시스템은 강화 학습 알고리즘의 기본 구조인 [그림 1]을 바탕으로 하여 설계 되었다. 즉, 게임의 상태 정보를 가지고 있는 User Interface는 Environment에 해당하고, 상태 정보를 바탕으로 게임을 하는 Main Module은 Agent에 해당된다.



[그림 5] 시스템 구조
Fig 5. System Structure

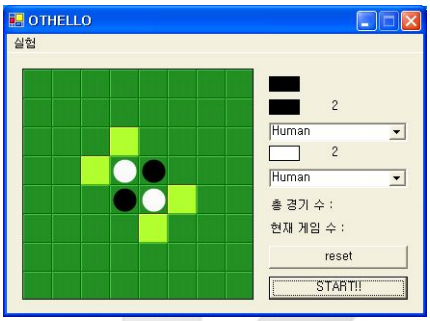
User Interface부분은 현재 진행 중인 게임의 상태 정보를 가지고 있다. 게임 정보를 사용자와 Main 모듈에게 제공하고 게임의 전반적인 일들을 제어한다. 또한 게임의 승리에 대한 혜택을 주기 위해 게임 내에서 일어나는 행동들을 기록한다.

Main Module은 User Interface 이외에 게임의 진행에 관련된 여러 모듈들을 포함한다. Main Module은 크게 Game Module과 AI Strategy Module로 구성되며 각각의 모듈은 여러 개의 세부 모듈로 구성되어 있다. 즉, Game Module은 Othello Rule Validator 와 Pre-selection Module로 구성되며, AI Strategy Module은 Selection 모듈, Q-learning 모듈 Q-table, Back-propagation 모듈들로 구성된다. 각 모듈 별 기능은 다음과 같다.

3. 시스템 구현

본 논문의 시스템은 Microsoft Windows XP 운영체제에서 Microsoft .NET 프레임 위에서 작동 되며, C# 으로 구현하였다.

[그림 6]은 게임 시작 버튼을 눌려 게임을 시작한 화면이다. 좌측 보드 판의 가운데에는 흰 색과 검은 색 디스크가 놓여져 있으며, 실행 가능한 영역은 보드 판 보다 밝은 색으로 표시를 하였다. 우측에는 지금 두어야 할 디스크를 색깔로 표시 하였으며, 각각 현재의 디스크 개수 및 현재 게임을 진행하고 있는 사용자 타입에 대해서 정의 한다.



[그림 6] 게임 시작 화면
Fig 6. Beginning of the Game

IV. 실험

1. 실험 방법

본 연구의 주요 알고리즘은 Q-learning알고리즘을 이용한 강화 학습 알고리즘이다. 본 논문의 시스템은 크게 두 부분으로 나누어 실험을 하였는데 예측 모델이 학습에 미치는 영향을 알기 위해 예측 모델이 초기화된 상태와 예측 모델이 학습된 상태 두 가지를 이용하여 실험 하였다. 또한 시스템의 성능측정을 위해서, 실험 방법은 사람 대신 Min-Max 알고리즘과의 게임을 통해서 학습의 여부를 보여주도록 하였다. Min-Max 알고리즘은 Depth 2와 Depth 3의 두 가지로 구현하여 각각 실험을 하였는데, 그 이유는 사용자의 게임 숙련도에 따른 학습의 차이를 알아보기 위해서이다. 실험은 10,000 경기씩 진행되었으며 학습 성과는 승률로 정의 하였다. 구체적인 실험의 과정은 다음과 같이 구분하였다.

① 강화학습 VS Min-Max Depth2 & Depth3
② 예측모듈기반의 강화학습 VS Min-Max Depth2 & Depth3 (not trained)
③ 예측모듈 기반의 강화학습 VS Min-Max Depth2 & Depth3 (trained)
④ 강화학습 VS 예측모듈 기반의 강화학습

2. 실험결과 분석

[표 1]은 본 실험의 전반적인 실험 결과를 정리한 데이터이다. ①~③은 Min-Max와의 대결을 통해 나온 결과로 기존의 강화 학습 알고리즘은 Min-Max 알고리즘과의 실험에서 본 논문의 시스템에 비해 깊이가 2일 경우에는 Pre-selection Module에서 학습이 안 된 경우 1.04%, Pre-selection Module에서 학습이 된 경우는 0.87%의 차이를 보여 큰 차이를 보이지 않는다. 그러나 깊이가 3인 Min-Max Search 알고리즘과의 실험에서는 Pre-selection Module에서 학습이 안된 경우는 8.69%, Pre-selection Module에서 학습이 된 경우는 8.72%의 큰 차이를 보인다.

Min-Max 알고리즘의 깊이가 2일 경우 기존의 알고리즘과 큰 차이를 보이지 않는 이유는 Min-Max 알고리즘 모듈의 게임 숙련도가 낮아 인공지능의 학습 여부를 판별하는 변별력이 없었다는 추측이 가능하다. 즉, 깊이가 3인 Min-Max 알고리즘 모듈과의 실험에서는 큰 차이가 난 것을 이유로 들 수 있다. 같은 이유로 깊이가 3인 Min-Max 알고리즘 모듈과 Pre-selection Module의 학습 여부에 따른 차이가 나지 않는 것에 대한 설명이 가능하다. Min-Max 알고리즘 모듈의 깊이를 더욱 깊게 한다면 다른 결과를 도출할 수 있을 것이다. 그 이유로 예측이 학습된 모듈은 깊이가 2 일 때 보다 깊이 3일 때가 승률이 좋아 진 것을 추측 근거로 들 수 있다.

[표 1] 실험 결과
Table 1. The Result of the Experiment

실험		승률 (%)	표준편차
①	Reinforcement Learning VS. Min-Max Depth 2	50.42 %	5.09
	Reinforcement Learning VS. Min-Max Depth 3	42.77 %	5.43
②	Reinforcement Learning with Pre-selection Module VS. Min-Max Depth 2(Not Trained)	51.97 %	4.99
	Reinforcement Learning with Pre-selection Module VS. Min-Max Depth 3(Not Trained)	51.46 %	5.23

실험		승률 (%)	표준편차
③	Pre-selection Reinforcement Learning VS. Min-Max Depth 2(Trained)	51.29 %	4.67
	Pre-selection Reinforcement Learning VS. Min-Max Depth 3(Trained)	51.49 %	5.20
④	Reinforcement Learning VS. Pre-selection Reinforcement Learning	51.1%	4.69

실험 결과를 보면 약 51% 정도의 승률을 계속 유지한다. 그 이유로 게임을 하면서 상대방에 대한 정보를 수집하는 Pre-selection Module과 직접 경기를 하면서 간접적으로 상대방의 정보를 수집하는 강화 학습만의 특성으로 보인다.

④의 실험의 경우 본 실험인 ①~③과의 결과와는 별도로 기존 강화 학습과의 직접 대결을 통해 비교 한 결과 이다. 본 논문의 시스템이 근소한 차이로 앞서는 것을 확인 할 수 있다.

2.1 세부 실험결과

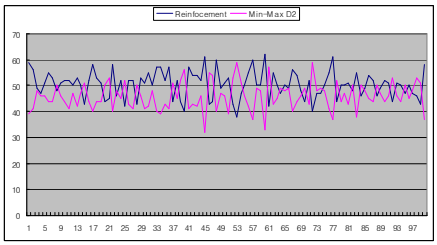
실험 결과 그래프는 100 게임당 승률로 결정 하였으며 총 100 경기의 승률을 이용하여 그래프를 구성하였다. x축은 경기 수(단위 100), y축은 승률을 나타낸다.

[그림 7]과 [그림 8]은 기존의 Q-learning을 이용한 강화 학습 알고리즘을 실험한 결과이다. 깊이가 2일 경우는 서로의 게임 숙련도가 비슷함을 볼 수 있으며, 깊이가 3일 경우 기존의 Q-learning 강화 학습은 전반적으로 지고 있다가 실험 후반에는 학습에 의해 점차 비슷해짐을 알 수 있다.

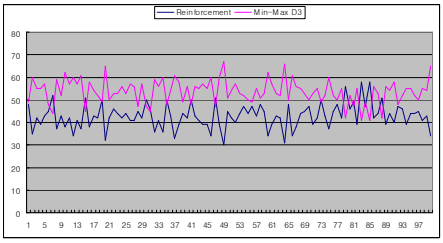
[그림 9]와 [그림 10]은 Pre-selection Module이 학습되지 않은 상태에서 실험을 한 결과이다. 두 그래프 모두 Min-Max Search 알고리즘에 비해 높은 게임 숙련도를 가지고 있으며, 후반으로 갈수록 안정적으로 변해 감을 확인 할 수 있다.

[그림 11]과 [그림 12]는 전 실험에서 수집하여 학습한 Pre-selection Module을 이용하여 실험한 결과이다. 두 그래프 모두 Min-Max Search 알고리즘에 비해 높은 게임 숙련도를 보여준다. 그러나 그래프에서 높은 승률이 나오다 갑자기 떨어지는 경우가 있다. 본 연구의 시스템의 게임 숙련도가 높아짐에 따라 상대방 게임 숙련도와 비슷해지려는 시도로 생각된다.

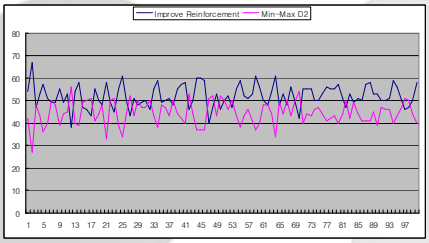
마지막으로, 직접적인 비교를 위해 기존의 강화 학습 알고리즘과 본 논문의 시스템을 직접 대결을 통한 실험을 하였다. [그림 13]은 이에 관한 실험 결과이다. 결과는 본 논문 시스템이 기존의 강화학습보다 더 우수한 능력을 보이고 있다. 전체 평균은 44.95% VS 51.1 %로 본 시스템이 6.15% 더 높았다.



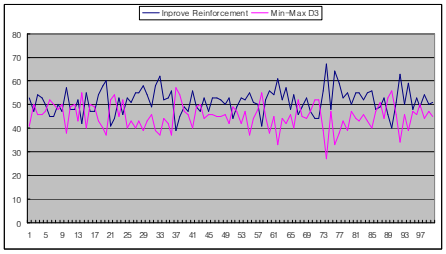
[그림 7] 강화학습 VS. 깊이2의 Min-Max
Fig 7. Reinforcement Learning VS. Min-Max Depth 2



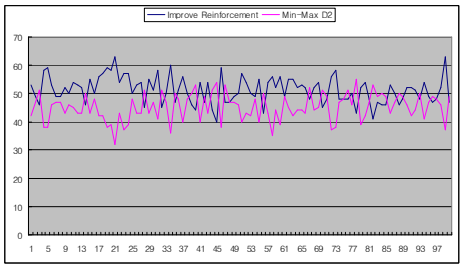
[그림 8] 강화학습 VS. 깊이 3의 Min-Max
Fig 8. Reinforcement Learning VS. Min-Max Depth 3.



[그림 9] Pre-selection의 강화학습 VS. 깊이2의 학습 안 된 Min-Max
Fig 9. Reinforcement Learning with Pre-selection VS. Min-Max Depth 2(Not Trained)

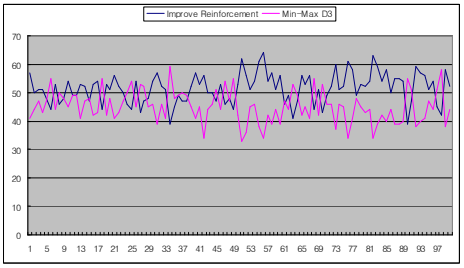


[그림 10] Pre-selection의 강화학습 VS. 깊이 3의 훈련 안 된 Min-Max
Fig 10. Reinforcement Learning with Pre-selection VS. Min-Max Depth 3(Not Trained)



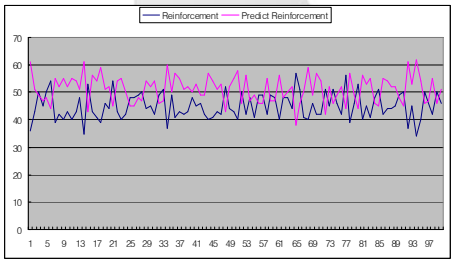
[그림 11] Pre-selection의 강화학습 VS. 깊이2의 훈련된 Min-Max

Fig 11. Reinforcement Learning with Pre-selection VS. Min-Max Depth 2(Trained)



[그림 12] Pre-selection의 강화학습 VS. 깊이3의 훈련된 Min-Max

Fig 12. Reinforcement Learning with Pre-selection VS. Min-Max Depth 3(Trained)



[그림 13] 기존의 강화학습 VS Pre-selection의 강화학습

Fig 13. Ordinary Reinforcement Learning VS Reinforcement Learning with Pre-selection

2.2. 실험 결과

실험 결과는 본 논문의 시스템이 기존의 시스템에 비해 보다 나은 게임 숙련도를 보이며 상대방의 게임 숙련도를 고려하여 게임의 난이도를 줄이려는 모습을 보였다.

실험의 수행 시간은 기존의 강화 학습 알고리즘의 실험은 2시간 30분 정도 소모된 반면 본 논문의 시스템은 3시간에서 3시간 30분 사이의 수행시간을 가졌다. Pre-selection Module의 행동을 학습하는 시간과 경기 결과 후 승리의 혜택을 주는 시간으로 인해 차이가 난 시간으로 실제 게임에 있어 진행에 불편을 느낄 수 있을 정도의 시간 소모는 일어나지 않았다.

V. 결론

본 연구에서는 사용자 적응성에 중점을 두는 강화 학습 알고리즘을 적용하여 보드게임을 설계 및 구현하였다.

본 연구의 특징은 다음과 같다. 첫째, 사용자 적응성을 높이기 위해 사용자의 행동을 수집하여 강화 학습의 의사결정 과정과 학습과정에 반영하였다. 둘째, 본 연구의 시스템을 분석하기 위하여 실험 대상으로는 적용이 쉽고 많은 상태 공간을 가진 오델로 게임을 선정하였고, 성능 측정을 위하여 기존의 강화 학습 알고리즘과 비교. 분석하였다. 두 알고리즘의 단순 비교가 어렵기 때문에 비교의 대상으로 Min-Max 알고리즘을 이용하여 대상으로 각각 대결하는 방식을 수행하였다. 셋째, 사용자간의 숙련도 차이를 고려하기 위해 Min-Max 알고리즘에 탐색 깊이를 변화 시켜 실험을 하였다. 실험의 결과 수정된 강화 학습 기반의 시스템은 사용자의 게임 숙련도의 차이를 주어 실험 했음에도 사용자의 게임 숙련도에 신속히 적응하는 모습이 보인다. 또한 기존의 강화 학습 알고리즘과의 대결에서도 보다 향상된 학습율을 을 나타내었다.

향후의 연구로는 오델로 이외의 다양한 게임에의 적용과 사용자 적응성에 대한 연구, 사용자의 행동을 미리 예측하는 Pre-selection에 대한 연구를 다양한 방법으로 연구할 필요성이 있다. 또한 게임의 승패에 대한 학습 가중치에 관한 연구도 진행되어야 할 것이다.

참고문헌

[1] 이만재, "게임에서의 인공지능 기술", 정보처리 학회지, 제 9권 3호, p69~p76, 2002.

[2] Michael Pfeiffer, "Reinforcement learning of strategies for settlers of CATAN" available at http://eprints.pascal-network.org/archive/00000425/01/ LAG-37_pfeiffer_ 2004.pdf, 2004.

[3] K.-F. Lee, S. Mahajan, "A Pattern Classification Approach to Evaluation Function Learning", Artificial Intelligence 36, pp.1-25,

Department of Computer Science, University of Texas at Austin, 1988.
[4] D. Moriarty, R. Miikkulainen, "Evolving Complex Othello Strategies Using Marker-Based Genetic Encoding of Neural Networks", Technical Report pp.AI93-206, 1993.
[5] Maxis Homepage, http://www.maxis.com/
[6] G.J. Tesauro, "Temporal Difference Learning and TD-Gammon", Communications of the ACM 38, 58-68, 1995.
[7] Imran Ghory, "Reinforcement learning in board games.", available at http://www.cs.bris.ac.uk /Publications/Papers/ 2000100.pdf, 2004.
[8] Tom M. Mitchell, "Machine Learning", ISBN 0070428077, 1997 .
[9] Nee Jan van Eck, Michiel van Wezel., "Reinforcement Learning and its Application to Othello", available at http://www.few.eur.nl /few/people/mvanwezel/rl.othello.ejor.pdf, 2004.

저 자 소 개

우 종 우

1983년 미네소타 주립대 전산학 석사
1991년 일리노이공대 전산학 박사
1994년~현재 국민대학교 컴퓨터학부 교수
관심분야: 인공지능, 에이전트, 정보보호

이 동 훈

2004년 국민대학교 컴퓨터학부 학사
2006년 국민대학교 대학원 전산과학과 석사
2006년 국민대학교 대학원 전산과학과 박사과정