Extended Classifier System을 이용한 파이팅 게임 인공지능의 개선

배청목 O 윤성훈 김경중*

세종대학교 컴퓨터공학과*

cjdahrl@gmail.com kiboyz2@naver.com kimkj@sejong.ac.kr

Improvement of Fighting Game Artificial Intelligence using XCS

Cheong-mok Bae O Seonghoon Yoon and Kyung-Joong Kim

Dept. of Computer Science and Engineering, Sejong University, Seoul.

요 약

Fighting Game AI Competition에 제출되는 AI들은 보통 규칙에 기반하여 패턴화된 행동을 반복하게 된다. 규칙에 기반한 AI들은 그 결과가 좋든 좋지 못하든 같은 상황에 대해서 같은 패턴만 반복하게 하므로, 상황에 대한 판단이 결여되어 있다고 볼 수 있다. 이런 점을 해결하기 위해 학습을 통해 여러 AI가 구사하는 다양한 패턴에 대응할 수 있는 능력을 가진 AI의 설계를 위해 Extended Classifier System 모듈을 도입한 AI를 구현하고 학습 성능을 평가한다.

1. 서 론

Fighting Game AI Competition(이하 FGAIC)은 2013년부터 시작한 국제 게임 인공지능 대회이다. FightingICE[1]라는 Player VS Player게임 에뮬레이터를 두고 참가자가 사전에 제출한 Java코드로 구현된 AI를 에뮬레이터에서 구동시켜 AI 대 AI로 대전하는 방식으로 AI간의 대결을 구현하였다.

그간 FGAIC에 제출되어온 AI코드를 보면 대체로 규칙에 기반한 하드코딩 AI가 상위권을 차지하는 모습을 볼 수 있었다. 이러한 AI들은 상대가 어떤 스타일로 행동하는지에 상관없이, 주어진 상황만을 분석하여 패턴화된 반응만을 보이게 된다. 임의의 상황에서 어떠한 반응이 불리하게 작용하더라도, 다음번에 같은 상황이 오면 같은 반응을 하게 된다는 것이다.

이러한 규칙 기반 AI의 패턴화된 전략의 한계점을 극복하기 위해, 규칙에 기반하지 않고 학습을 통해 상대하는 AI의 행동에 따라 최선의 선택을 하는 AI를 인공지능 구현을 위해 XCS를 이용한 AI를 구현하여 성능을 평가해보았다.

2. XCS의 개념

XCS는 보상에 의해 학습하는 강화 학습 방법과 유전 알고리즘을 결합한 방법이다.[1][2] 강화 학습은 어떠한 문제에 대한 행동을 취할 때, 환경으로부터 그 행동에 대한 보상을 받고 이러한 보상이 쌓인 경험을 통해서 더 나은 행동을 찾아가는 알고리즘이다. 기본적인 알고리즘은 환경 상태 집합 C(Condition)와 행동 집합 A(Action), 보상 집합 R(Reward)을 갖는 모델 구성을 가지고 있고, 매 시점에 에이전트는 자신의 상태와 가능한 행동을 가진다. 이 때, 어떤 행동을 취했을 경우, 변화한 환경으로부터 새로운 상태와 보상을 받게 되는데, 이러한 상호작용에 기반해 누적된 보상 집합 값 R을 최대화하는 방법이다. 강화학습에서 가장 중요한 부분은 다양한 상황을 시뮬레이션 할 수 있도록 하거나, 각 상황에 대해 평가를 받는 부분이다. 이는 게임에서 일어날 수 있는 다양한 상황에 대한 평가를 인간이 아닌 인공지능이 자체적으로 해야 하는 부분이기에 여러 인공지능을 통해 경험을 쌓아야 하고, 그러한 시간적인 부분 또한 고려해야 한다는 것은 쉽지가 않다.

유전 알고리즘은 자연세계의 진화과정에 기초한 계산 모델로서, 최적화 문제를 해결하는 기법이다.[3] 생물 진화 기법으로, 실제 진화의 과정처럼 돌연변이, 교배 과정이 있는 탐색 알고리즘으로써, 문제에 대한 가능한 해들을 점차적으로 변형함으로써 점점 더 좋은 해집단을 만들어 내는 알고리즘이다. 여기서 해들을 나타내는 자료구조는 유전자이며, 이들을 변형하여 좋은 해를 만들어 나가는 과정을 진화로 표현할 수 있다. 유전 알고리즘에서 해의 선택은 알고리즘의 성능에 큰 영향을 미치기 때문에 어떤 해를 선택하는 지는 유전 알고리즘의 성능에 큰 영향을 미치는 중요한 요소라고 할 수 있다.

3. XCS의 구현

[그림 1]은 파이팅 게임에 적용하기 위해 구현된 XCS의 구성도이다. XCS는 파이팅 게임에서 한 번의 게임에 대해 학습하는 것이 아닌 여러 번의 게임에 대한 학습 결과를 누적하여 진행하는 것으로, 상태집합과 행동집합, 보상집합이 학습의 과정에서 유지되어야 한다. 따라서 파이팅 게임내의 중요한 정보라 생각되는 요소들을 상태집합 C와 행동집합 A, 그리고 그에 해당하는 보상집합 R을 텍스트 파일로 저장해 학습의 과정에서 유지될 수 있도록 하였다.

*교신저자, 세종대학교 컴퓨터공학과 부교수

[1] FGAIC를 주최하는 Ritsumeikan대학 Intelligent Computer Entertainment 연구실에서 자체 제작한 2D 대전게임 플랫폼.

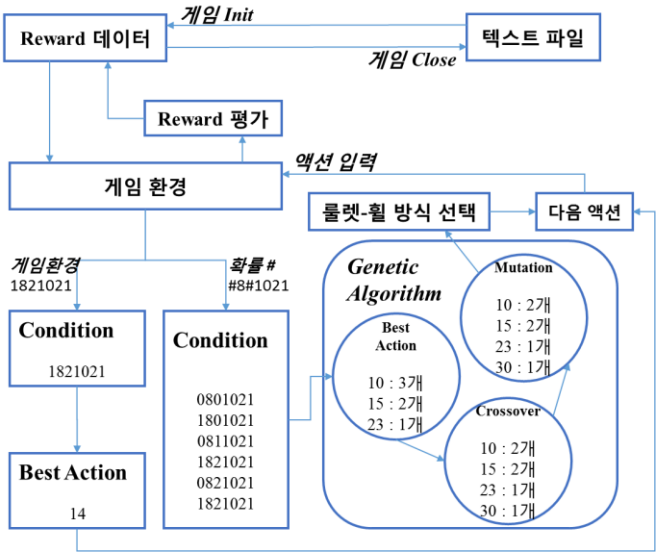

그림 1 구현된 XCS의 구조

게임 환경에서 추출한 C는 확률적으로 게임 환경 추출된 정보 그대로 다음에 취할 행동을 결정짓는 방식과, 확률에 의해 각 C요소를 #으로 대체하여 그에 해당되는 자식 C를 구성하고 유전 알고리즘을 통하여 다음 행동을 결정짓는 방식 두 가지가 있다.

4. 실 험

XCS를 이용한 AI의 성능이 어느 정도 개선되는지를 비교하기 위해 다음과 같은 방식으로 비교실험을 전개하였다. 학습 대상이 되는 AI는 게임 내 기술 중 장풍이라 불리는 원거리 기술만을 반복적으로 구사하며, 근접했을 경우 하단 발차기로 거리를 벌려놓고 다시 장풍을 쏘는 간단한 패턴을 가진 AI (이하 TestBot)를 사용하였다.

4.1 보상 테이블 단순 갱신 방식

실험 1에서는 요소 상태집합을

(1) 현재 내 캐릭터가 벽을 등지고 있는지 (2가지)
(2) 상대 캐릭터와의 거리가 어느 정도인지 (25단계)
(3) 내 캐릭터의 현재 상태가 무엇인지 (4가지)
(4) 내 캐릭터의 현재 에너지가 얼마인지 (7단계)
(5) 상대방의 직전 행동이 무엇인지 (56가지)

로 규정하였다. 위 5가지 요소에 대한 내 캐릭터가 취할 행동(56가지)에 대한 보상 테이블이 입출력 파일로 주어진다. 상태 테이블은 총 2 * 26 * 4 * 7 * 56 * 56 = 4,390,400개의 탐색 공간[2]을 가지며, 이는 파일로 대략 16MB정도의 크기이다. 게임을 시작할 때, 보상 테이블 파일을 읽어 들이고, 실시간으로 현재 상태에 대한 내 캐릭터의 행동을 56가지 중, 보상 값 비례 룰렛-휠 방식으로 선택하였다. (단, 첫 실험 시 사용된 보상 테이블은 프로그램에 의해 임의로 생성된 입력파일을 사용하였다.) 실험 4.1에서 적용한, 행동에 대한 보상 값 갱신은 수식 1과 같이 정의했다.

```
if (Rate > 0) {
   if (Rate > Rate') R(A) + 30
   else if (Rate < Rate') R(A) – 10
   else R(A) + 5
} else {
   if (Rate > Rate') R(A) + 30
   else if (Rate < Rate') R(A) – 30
   else R(A) – 5
}
```

수식 1 실험 4.1의 보상 값 갱신 방식

Rate는 캐릭터간의 우위를 간단히 비교하기 위해 고안한 값으로 Rate = (상대 캐릭터가 입은 피해 + 1) / (내 캐릭터가 입은 피해 + 1) 이다. Rate'는 A라는 행동을 취한 후의 Rate값이다. R(A)는 행동 A에 대한 보상 값이다. Rate와 Rate'를 비교하여 상황이 좋아졌을 경우와 나빠졌을 경우 R(A)값을 가감했다.[3]

4.2 XCS를 이용한 학습

실험 2에서는 요소 상태집합을

(1) 현재 내 캐릭터가 벽을 등지고 있는지 (2가지)
(2) 상대방의 기술이 장풍인지 (10가지)
(3) 내 캐릭터의 현재 상태가 무엇인지 (3가지)
(4) 상대방의 상태가 무엇인지 (3단계)
(5) 나와 상대방 캐릭터의 체력 비율이 얼마인지 (3가지)

로 규정하였다. 위 5가지 요소에 대한 내 캐릭터가 취할 행동(38가지)에 대한 보상테이블이 입출력 파일로 주어진다. 상태 테이블은 총 2 * 10 * 2 * 3 * 3 * 3 * 3 * 38 = 123,120 (개)의 탐색 공간을 가지며, 이는 파일로 대략 500KB정도의 크기이다.

```
if (Rate > 0) {
   if (Rate > Rate') R(A) + 5
   else if (Rate < Rate') R(A) – 3
   else {
      if (Dis < Dis') R(A) + 5
      else R(A) + 2 }
} else {
   if (Rate > Rate') R(A) + 30
   else if (Rate < Rate') {
      if (Dis < Dis') R(A) + 2
      else R(A) – 1 }
   else {
      if (Dis < Dis') R(A) + 2
      else R(A) – 1 }
}
```

수식 2 실험 4.2의 보상 값 갱신 방식

[2] 후술되는 실험 4.2와 탐색공간의 크기가 다른데, 실험4.1의 학습 정도가 너무나 저조하여 처음엔 같은 조건으로 시작하였으나 점진적으로 탐색공간을 늘인 결과이다.

[3] 후술되는 실험 4.2와 가감되는 값의 크기가 다른데, 앞선 2번과 비슷한 맥락으로, 실험 4.1의 저조한 학습 정도로 인해 학습 강도를 높이기 위해 보상 값을 점차 크게 한 결과이다.

수식 2의 보상 값 가감법도 수식 1에서와 크게 차이는 없다. 다만 Dis는 나와 상대의 거리를 나타나는 변수로 Rate값의 변동과 더불어 이후 상황에 대한 유불리함를 이끌어 갈 수 있다는 점에 크게 관여한다고 생각되어 넣게 되었다.

5. 결 과

5.1 보상 테이블 단순 갱신 방식 결과

4.1에서 제안된 방법으로 구현한 AI(이하 SimpleBot)와 TestBot과의 1,000회 시뮬레이션을 가질 결과는 [그림 2]와 같다.

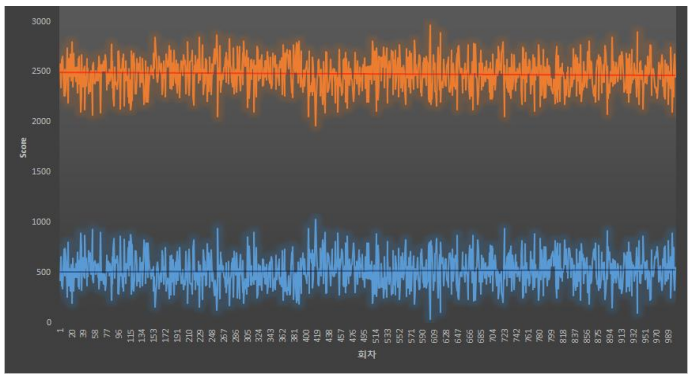

그림 2 SimpleBot vs TestBot 결과 (실험 4.1)

주황색 선이 회차별 TestBot이 획득한 점수, 하늘색 선이 SimpleBot이 각 회차별로 획득한 점수이다. 붉은색, 푸른색 선은 각각의 AI가 획득한 점수를 1회에서 1,000회까지 선형회귀 분석한 결과이다. 결과적으로, 4.1의 방법 구현된 AI는 1,000회 시뮬레이션 해 보았을 때, 개선의 모습의 모습을 관찰할 수 없었다. 선형회귀를 통한 분석 또한 개선되는 점을 관찰할 수 없다.

5.2 XCS를 이용한 학습 결과

4.2에서 제안된 방법으로 구현한 AI(이하 XCSBot)와 TestBot과의 1,000회 시뮬레이션을 가질 결과는 [그림 3]과 같다.

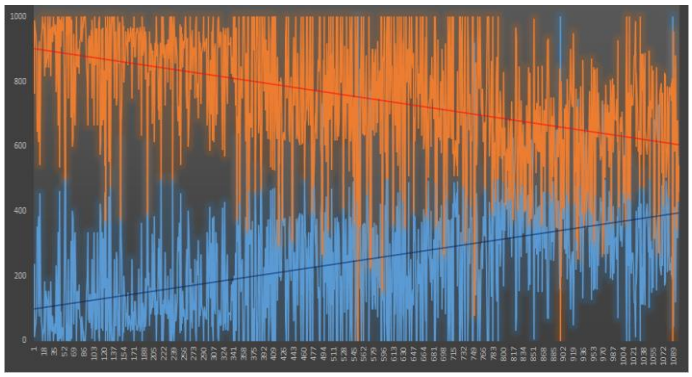

그림 3 XCSBot vs TestBot 결과 (실험 4.2)

주황색 선이 회차별 TestBot이 획득한 점수, 하늘색 선이 회차별 XCSBot이 각 회차별로 획득한 점수이다. 붉은색, 푸른색 선은 각각의 AI가 획득한 점수를 1회에서 1,000회까지 선형회귀 분석한 결과이다. 결과적으로 XCSBot은 1,000회까지 학습을 진행했을 때, 점진적으로 점수가 증가하는 것을 관찰할 수 있었다. 각 회차별 총점인 1,000점을 획득하는 횟수 또한 늘어나는 것을 관찰할 수 있었다.

6. 결 론

본 실험을 통해, 보상 테이블 단순 갱신 방식을 구현한 AI와 XCS를 이용한 학습을 구현한 AI, 두 가지 모델의 학습 경과를 관찰했다. 아직 두 가지 방식 모두 대전 상대 AI를 압도하는 모습을 볼 수 없었지만, XCS를 이용한 회차가 진행됨에 따라 획득하는 점수가 점차 증가하고 있다는 점에서, 단순한 패턴을 갖는 AI의 경우 학습을 통해 점진적으로 좋은 성적을 거둘 수 있다는 점을 확인할 수 있었다. 다만, 짧은 회차 내에서 극적인 학습효과를 얻을 수 없기 때문에, 각 AI간 1회 3라운드의 대결만을 진행하는 대회에서는 좋은 성적을 낼 것이라고 기대하기는 어렵다.

7. 감사의 글

본 연구는 2013년도 미래창조과학부의 재원으로 한국 연구재단의 지원을 받아 수행된 중견연구자지원사업 (2013 R1A2A2A01016589) 및 미래창조과학부 및 정보 통신 산업진흥원의 개방형 ICT융합과정 지원사업의 연구결과로 수행되었음. (NIPA-2014-과제번호H1822-14-1003)

Reference

[1] Butz, Martin V., and Stewart W. Wilson. "An algorithmic description of XCS."*Advances in Learning Classifier Systems*. Springer Berlin Heidelberg, 2001. 253-272.

[2] Lanzi, Pier L., Wolfgang Stolzmann, and Stewart W. Wilson. *Learning classifier systems: from foundations to applications*. No. 1813. Springer Science & Business Media, 2000.

[3] 문병로, *쉽게 배우는 유전 알고리즘: 진화적 접근법*, 한빛미디어 (2014)