강화학습을 통한 스타크래프트 빌드 최적화

김신영(18)1, 오상훈(18)2

*11Department of Statistics, Hankuk University of Foreign Studies, Yongin, Korea*

*21Department of Statistics, Hankuk University of Foreign Studies, Yongin, Korea*

cbrnt1210@naver.com1*, eric5050@hufs.ac.kr 2*

# *Abstract*

본 논문은 스타크래프트2를 활용한 강화학습의 적용성을 탐구한다. 이 연구의 주요 목적은 인공지능을 이용하여 게임 내에서 빌드 전략 최적화를 향상시킴과 동시에 사용자 편의성 증대에 중점을 두고 있다. 특히, 본 연구는 사용자가 제시한 특정 조건에 맞추어 최적화된 빌드 전략을 생성하는 인공지능 시스템을 개발하는 데 집중한다. PPO를 활용하여 사용자 명령 순서를 최적화함으로써, 게임 플레이어들이 효율적이고 효과적인 게임 경험을 할 수 있도록 돕는다. 연구 결과는 강화학습을 통한 게임 전략 개발의 가능성을 입증하며, 게임 인공지능 분야에 중요한 기여를 한다. 또한, 이 연구는 스타크래프트2가 강화학습 연구에 적합한 환경임을 보여주며, 복잡한 정보 아래에서의 최적 결정 모델링 능력을 입증하는 중요한 사례로 자리매김한다.

***Keywords:*** *강화 학습, PPO, RTS, 비동기 프로그래밍*

1. Introduction

컴퓨터 게임, 특히 스타크래프트2는 강화학습 연구에서 핵심적인 역할을 한다. 이 실시간 전략 게임은 광범위한 행동 공간과 다중 에이전트 환경을 제공하며, 다양한 종족, 유닛, 건물, 리소스가 상호작용하는 복잡한 환경을 통해 현실 세계의 다양한 문제를 모델링하는 데 이상적이다. 또한, 불완전한 정보를 기반으로 하는 게임 플레이는 강화학습 알고리즘의 일반화 및 의사결정 능력을 시험하는 데 중요한 도전 과제를 제공한다.

본 연구에서는 스타크래프트2의 이러한 독특하고 복잡한 환경을 활용하여 최적의 빌드 전략을 개발하고, 특정 유닛의 효율적인 생산을 통해 전투에서 승리하는 방법을 탐구한다. 본 연구는 스타크래프트2를 통한 강화학습의 적용 가능성을 탐색하며, 게임 전략 개발과 인공지능 분야에 새로운 기여를 목표로 한다. 스타크래프트2의 복잡한 환경을 이해하고 이를 바탕으로 한 강화학습 알고리즘의 효과를 검증함으로써, 게임 인공지능 발전에 중요한 통찰을 제공할 것이다.

**2. Background**

스타크래프트2는 강화학습 연구에서 중요한 환경으로 자리 잡았으며, 특히 이 게임의 다중 에이전트 환경은 복잡한 전략적 선택과 의사결정 기회를 제공한다. 다중 에이전트 환경에서 여러 독립적인 에이전트는 각자의 목표를 가지고 서로 상호 작용한다. 이러한 상호작용은 협력, 경쟁, 또는 이 둘의 조합을 포함할 수 있으며, 각 에이전트는 독립적인 의사결정을 내리면서 환경에 영향을 미친다. 스타크래프트2에서 플레이어는 이러한 다중 에이전트 환경에서 자신만의 전략을 수립하고 실행해야 한다. 이러한 환경에서 AlphaStar와 같은 혁신적인 프로젝트가 탄생했다.[1]

AlphaStar는 인공지능이 스타크래프트2와 같은 복잡한 실시간 전략 게임에서 전문가 수준의 플레이를 할 수 있음을 입증했다. 마르코프 결정 과정과 벨만 방정식과 같은 고급 수학적 프레임워크와 깊은 신경망을 활용하여 AI가 다양한 전략을 학습하고 적용하는 능력을 개발했으며, 국제 대회 수준의 선수들과의 대결에서 5-0으로 승리를 거두며 우수한 성과를 달성했다. 이 연구는 복잡한 전략 게임에서 AI가 인간과 유사한 수준으로 플레이할 수 있으며, 실세계 문제 해결에 AI를 적용할 수 있는 가능성을 보여주었다.

이후 Alphastar는 다중 에이전트 강화학습 알고리즘과 딥 뉴럴 네트워크를 사용하여 전문가 수준의 스타크래프트2 플레이어들과의 온라인 게임에서 그랜드마스터 레벨에 해당하는 성능을 보여주었으며, 이를 통해 인공지능이 실시간 전략 게임에서 인간과 유사한 수준으로 플레이할 수 있음을 입증했다.[2] 이 연구는 인공지능이 실시간 결정을 내려야 하는 복잡한 액션 공간에서 효과적으로 작동할 수 있음을 보여주었으며, 인공지능이 실제 세계 문제 해결에 적용될 수 있는 잠재력을 제시한다.

또한, 스타크래프트2를 인공지능 연구의 새로운 도전으로 제시한 연구도 나왔으며, 이를 통해 AI의 다양한 전략과 의사결정 능력을 평가했다.[3] 연구팀은 Python 인터페이스와 인간 플레이어의 데이터를 활용하여 AI를 훈련시키고, 미니 게임과 전체 게임에서 AI의 성능을 테스트했다. 현재 AI는 스타크래프트2의 기본 AI에도 도전이 되지만, 이 연구는 인간과 유사한 조건에서 AI가 경쟁할 수 있는 미래를 열어주고 있다.

**3.** Methodology

본 연구는 스타크래프트2내에서 사용자가 원하는 조건에 맞춰 최적화된 빌드 전략을 개발하는 데 중점을 두었다. 연구의 주 목적은 사용자 편의성을 증대시키고, 특정 유닛의 생산을 최적화하는 데 필요한 빌드 순서를 정교하게 계산하는 것이다. 이를 위해, 강화학습 알고리즘을 활용하여 사용자가 설정한 조건에 따라 최적화하였다.

본 연구는 멀티가 하나인 가정 하에서, 사용자가 원하는 특정 시점에 최적의 빌드 순서를 계산한다. 강화학습의 평가는 동일한 맵과 초기 상태에서 시작하여, MDP를 기반으로 한 이론적 계산과 강화학습 결과를 비교 분석했다. 이 과정에서 복잡한 게임 환경과 다양한 게임 상황에 대응하는 최적의 빌드 전략을 도출했다.

이 연구에서는 PPO 알고리즘을 사용하여, 사용자 명령 순서의 최적화에 중점을 두었다. 특히, 전투 전략이나 컨트롤은 고려 대상이 아니었으며, 오로지 빌드 순서의 최적화에만 집중했다.

**3-1. Bellman Equation, MDP, MRP**

벨만 방정식은 최적화 문제에서 각 상태의 최적 가치를 추정하는 데 사용된다. 이 방정식은 최적화 문제를 해결하기 위해 최적 가치 함수를 재귀적으로 규정한다. 이는 현재 상태와 그 상태에서 취해진 행동의 가치를 토대로 다음 상태의 가치를 예측하는 방식을 사용한다. 이러한 방식으로, 최적 가치 함수를 구성하고 이를 통해 최적의 행동을 결정할 수 있다.

마르코프 보상 과정(Markov Reward Process, MRP)은 각 상태의 장기적 가치를 평가하는 프레임워크로 상태(s), 보상 함수, 시간에 따른 보상의 감가율(), 그리고 상태 전이 확률()로 구성된다. MRP의 핵심은 각 상태가 게임 내에서 얼마나 유리한 위치를 제공하는지를 평가하는 것이다. 각 상태에 할당된 보상 함수는 해당 상태에 도달했을 때의 이익을 수치화한다. MRP의 수식은 벨만 방정식으로 다음과 같이 표현된다:

*v*(*s*) = *R*(*s*) + γ*P*(*s*′∣*s*)⋅*v*(*s*′)

여기서 *v*(*s*)는 상태 s의 가치, R(s)는 상태 s에서 받는 즉각적인 보상, *v*(*s*′)는 다음 상태 *s*′의 가치를 의미하며, *v*(*s*)는 기댓값을 수열의 합과 다음 상태에서의 가치 함수로 나타내었다.

마르코프 결정 과정(Markov Decision Process, MDP)은 환경의 가치를 극대화하는 정책을 결정함으로써 최적의 정책을 결정하는 포괄적인 프레임워크로 상태(s), 행동(a), 보상 함수(), 감가율(), 정책 함수(), 그리고 상태 전이 확률()의 주요 요소로 구성된다. 본 연구에서 MDP는 스타크래프트2 게임의 각 상황을 상태로 모델링하고, 각 상태에서 취할 수 있는 가능한 행동들(빌드 순서)을 정의한다. 보상 함수는 각 행동의 결과로 얻어지는 게임 내 이익을 수치화한다. 이를 통해, 에이전트는 정책에 따라 다양한 상태와 행동을 경험하고 이를 통해 얻은 피드백으로 최적의 빌드 전략을 학습한다. MDP의 수식은 벨만 방정식으로 다음과 같이 표현된다:

= (*s*′∣*s*)(*s*′)

여기서 는 *v*(*s*)에서 가치를 계산할 때 정책 요소를 하나 더 고려한 것으로 *v*(*s*)와 동일한 값을 가지며, 에서 상태 S에서 정책을 고려한 직접적인 보상을 계산하고, (*s*′∣*s*)(*s*′)에서는 정책과 상태 전이 확률을 함께 고려해 수열의 합이 행동별 정책, 상태별 전이 확률 두 번 계산되었다.

**3-2. PPO**

Proximal Policy Optimization (PPO)는 Trust Region Policy Optimization (TRPO)의 후속작으로, 실용적이고 컴퓨터 연산에 용이하게 설계된 것이 특징인 알고리즘이다. 이 알고리즘은 특히 복잡한 환경에서의 정책 최적화 문제를 해결하고자 개발되었으며, 학습 과정에서 높은 안정성을 유지하면서 효율적으로 정책을 업데이트하는 것을 목적으로 한다. 이는 실시간 전략 게임인 스타크래프트2와 같은 복잡한 환경에서 특히 유용하다.

PPO는 TRPO의 아이디어를 기반으로 하지만, 구현의 복잡성을 크게 줄인 것이 특징이다. TRPO가 정책의 업데이트가 특정 범위 내에서 이루어지도록 제한하여 각 업데이트가 안정적으로 이루어지도록 설계되었으나, 계산적으로 복잡하고 구현이 어려운 단점이 있었다. 이에 비해 PPO는 이러한 TRPO의 제약을 완화하고, first-order optimization을 사용하여 계산 효율성을 높인다. PPO의 핵심적인 메커니즘 Clipped Surrogate Objective는 새로운 정책이 기존 정책으로부터 크게 벗어나는 것을 제한함으로써, 학습 과정의 안정성을 크게 향상시킨다. 이 메커니즘은 목표 함수의 최적화 과정에서 정책 비율의 변화를 일정 범위 내로 제한함으로써, 학습 과정의 안정성을 보장한다. 이는 학습 과정에서 발생할 수 있는 큰 정책 변화를 방지하고, 에이전트의 성능이 갑작스럽게 저하되는 것을 막으며, 높은 변동성을 크게 줄여 더 안정적이고 예측 가능한 학습 경로를 제공한다. 또한 PPO는 Actor와 Critic이라는 두 가지 네트워크를 사용하여 최적화를 수행하며, 정책과 가치 함수의 파라미터를 공유할 수 있는 구조를 가지고 있어 구현을 단순화하고 이론적 접근을 용이하게 한다.

PPO는 환경 모델을 필요로 하지 않고 게임 내 상황에 따라 병렬처리를 통해 여러 상태와 행동 조합을 동시에 학습할 수 있는 강화 학습 기법 중 하나이다. 이는 특히 복잡한 환경이나 다중 에이전트 시나리오에서 유리하며, 스타크래프트2와 같은 실시간 전략 게임 환경에서 효과적인 학습을 가능하게 한다. 병렬 처리를 통해, 알고리즘은 다양한 상황에서의 데이터를 동시에 수집하고 처리할 수 있으며, 이를 통해 더 빠르고 효율적으로 정책을 최적화할 수 있다. 따라서, 스타크래프트2와 같이 복잡한 게임에 적용하기에 적합하다 판단했다

**3-3. Environments**

본 논문에서는 AI봇 구현을 위해 python-sc2 라이브러리를 사용하였다.[5] 이 라이브러리는 스타크래프트2의 AI 개발을 돕기 위해 제공되며, Python 환경에서 비동기 프로그래밍의 형태로 게임 내 에이전트를 손쉽게 구현할 수 있게 해준다. 특히 본 연구는 스타크래프트2의 세 종족 중 하나인 프로토스에 초점을 맞췄다. 프로토스는 고유의 건물과 유닛을 보유하고 있으며, 이 연구에서는 '파일런 건설'과 같은 프로토스 종족 특유의 행동 요소를 시뮬레이션에 포함시켰다.

BurnySc2 라이브러리 내에 제공되는 IncrediBot 클래스를 통해 강화학습의 환경이 구성된다. 매 프레임마다 on\_step함수를 통해서 게임의 상태를 확인하고 행동을 할 수 있다. 따라서 on\_step단위로 state\_rwd\_action.pkl파일을 통한 I/O방식으로 상태를 모델에 전달하고 액션을 결정하게 하였다.

본 연구의 목표는 Void Ray 5대의 생산하는데 필요한 빌드 순서를 최적화하는 것이다. 이를 위해 “일꾼 생산, 파일런 건설, 테크 트리 진행, Void Ray 생산 및 시간가속, 멀티 확장, 가스 건설, 가스 채취 일꾼 배치, 행동 없음”과 같은 총 7개의 행동을 설정하여 동작하게 하였다. 이 행동을 진행하는 데 필요한 값인 “미네랄, 가스, 현재 인구수, 최대 인구수, 일꾼의 수, 멀티의 수, 테크의 수준, iteration(시간값)”인 8개의 이산적인 값을 게임에서 추출하여 상태로 활용하였다. 또한 모델의 수렴 속도를 높이기 위해 프루닝 기법을 활용하여 행동과 상태에서 특정 제한 조건을 설정하였다. 가장 빨리 Void Ray 5대를 생산하는 것을 목표로 직접 계산을 해보니 대략 1000 iteration이라는 값이 나와서 이를 기준으로 하여 4배의 시간 공간으로 기준을 설정하였다. 게임 진행 시간이 4000 iteration을 초과할 경우 게임을 패배로 간주하여 환경을 제한하였다. 전체 게임 중 확장 행동은 한 번만 실행하는 것으로 설정하여 환경을 제한하였다. 또한 테크 트리 관련 행동은 게이트웨이, 사이버네틱스 코어와 같은 주요 건물을 단 한 번만 건설하도록 제한하였다.

기준이 4000 iteration이므로 목표를 달성한 시점의 iteration을 4000에서 빼서 최종 보상을 주었다. 방향성을 주어 좀 더 빠른 수렴을 하기 위해서 목표를 달성하는 데 필요한 테크 트리와 목표 유닛의 생산 행동이 이뤄질 때 중간 보상을 iteration 값에 따라 할당하여 중간보상을 제공하였다. 마지막으로 매 프레임마다 모델이 게임의 상태를 통해 행동을 결정하기 때문에 현재 불가능한 행동이면 -보상을 주고 행동이 이뤄질 수 있는 환경일 때 +보상을 주며 아무것도 하지 않는 행동을 경우에 추가함으로써 행동이 가능할 때만 행동을 선택할 수 있도록 유도하였다.

**3-4. Evaluation**

에이전트는 정책을 바탕으로 환경과 상호작용하며 상태, 행동, 보상 및 새로운 상태에 대한 데이터를 수집한다. 이 데이터는 PPO의 장점 추정과 정책 업데이트에 활용되며, 특히 클리핑 메커니즘을 통해 정책 업데이트의 크기가 조절되어, 에이전트의 효율적인 행동 전략 학습과 성능 개선을 지원한다.

일반적인 상황에서, 사용자가 임의로 정한 최적의 빌드를 따를 경우 Void Ray 5대를 생산하는 데 약 6분이 걸릴 것이라는 가정하에 시작하였다. 확장에 대한 행동 없이 PPO를 적용하였을 때, 일꾼을 처음에 생산하지 않고 테크를 먼저 타는 정책을 통해서 Fig 1과 같이 4분 51초 만에 Void Ray 5대를 생산하는 결과를 얻을 수 있었다.

**Fig 1.** 멀티 환경 없는 테크트리 빌드 과정 최적화 결과

스크린샷, 멀티미디어 소프트웨어, 그래픽 소프트웨어, PC 게임이(가) 표시된 사진

자동 생성된 설명

이후 추가적으로 멀티 환경을 추가함에 따라 스타게이트의 설치 수를 늘리거나 일꾼을 가스에 배치하는 시점 등의 다양한 환경 조치를 취했을 때의 복잡한 결과를 사람이 직접 예측하기는 어렵지만, PPO를 통해 분석함에 따라 Table 1과 같이 5분 40초만에 Void Ray 5대를 생산하는 최적화 결과를 얻을 수 있었다.

**Table 1.** 멀티 환경 추가에 따른 테크트리 빌드 과정 최적화 결과

|  |  |
| --- | --- |
| 순번 | 행동 |
| 1 | 넥서스 건설 |
| 2 | 파일런 건설 |
| 3 | 가스 건설 |
| 4 | 게이트웨이 건설 |
| 5 | 가스에 일꾼 이동 |
| 6 | 가스 건설 |
| 7 | 가스 건설 |
| 8 | 가스에 일꾼 이동 |
| 9 | 가스에 일꾼 이동 |
| 10 | 사이버네틱스코어 건설 |
| 11 | 가스 건설 |
| 12 | 스타게이트 건설 |
| 13 | 스타게이트 건설 |
| 14 | 스타게이트 건설 |
| 15 | 파일런 건설 |
| 16 | 파일런 건설 |
| 17 | 파일런 건설 |
| 18 | 가스에 일꾼 이동 |
| 19 | Void Ray 생산+시간가속 |
| 20 | Void Ray 생산+시간가속 |
| 21 | Void Ray 생산+시간가속 |
| 22 | Void Ray 생산+시간가속 |
| 23 | Void Ray 생산+시간가속 |
| 시간 | 05:40 |

결론적으로, PPO방법론은 복잡한 게임 환경인 스타크래프트2에서도 효과적으로 적용될 수 있음을 확인하였다. 특히, 주어진 제한 조건과 보상 시스템 하에서 에이전트가 최적의 전략을 성공적으로 학습하였으며, 이를 통해 게임 내에서의 성능을 효과적으로 향상시킬 수 있음을 보여주었다. 본 연구에서 사용된 모든 코드와 구현 과정은 GitHub에서 sc2ai 라이브러리를 통해 확인할 수 있다.[6]

**4. Conclusion**

본 연구에서는 스타크래프트2를 강화학습 연구의 중요한 대상으로 제시하고 게임 내 빌드를 최적화하여 최단 시간내에 목표를 달성하였다. 스타크래프트2 환경은 강화학습 알고리즘의 일반화 능력을 검증하고, 불확실하고 다양한 정보 아래에서 최적의 결정을 내리는 능력을 모델링하는데 매우 적합한 것으로 나타났다.

이러한 연구를 통해 게임 환경에서의 강화학습 연구의 중요성과 가능성을 강조하며, 게임 인공지능 분야에 새로운 기여를 제공함을 확인했다. 사용자가 생각한 빌드에 따라 미네랄, 가스 수집의 비율과 빌드를 자동으로 최적화하여 게임에서의 인공지능 성능을 향상시킬 수 있음을 입증하였으며, 앞으로의 연구에서는 이러한 결과를 발전시켜 최종 목표를 계속해서 변경해 나감에 따라서 사용자가 생각한대로 작동하는 게임 인공지능 분야에 더 큰 기여를 할 것으로 기대한다. 이를 통해 RTS게임의 진입장벽인 빌드와 컨트롤부분 중 빌드에 대한 부분을 좀 더 완화할 수 있을 것이다. 최종적으로는 모바일 환경과 같이 많은 조작이 어려운 환경 속에서도 RTS게임을 즐길 수 있을 것이다.

**References**

1. The AlphaStar team (2019). AlphaStar: Mastering the real-time strategy game StarCraft II.
2. Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature volume 575, 350–354
3. Oriol Vinyals, Timo Ewalds, Sergey Bartunov (2017). StarCraft II: A New Challenge for Reinforcement Learning.
4. John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov(2017). Proximal Policy Optimization Algorithms
5. <https://github.com/BurnySc2/python-sc2/tree/develop>
6. [https://github.com/sinyeong10/sc2ai](https://github.com/sinyeong10/sc2ai%20)