Game Balance

내용

[논문 이해하기 1](#_Toc435720290)

[4. 게임 언어 (Game Description Language) 1](#_Toc435720291)

[개념들 1](#_Toc435720292)

[언어 2](#_Toc435720293)

[5. 인공 지능으로 플레이 하기 3](#_Toc435720294)

[게임 상태 공간 3](#_Toc435720295)

[목표 기반 탐색과 게임 플레이 4](#_Toc435720296)

[연습문제 – 스킬 조합 패턴 파악 4](#_Toc435720297)

[결과 5](#_Toc435720298)

[정리 6](#_Toc435720299)

# 몬테 카를로 시뮬레이션

**Exploring Game Space Using Survival Analysis**

# 논문 이해하기

Game Balancing Using Game Descriptions and Artificial Intelligence

생각했던 아이디어와 동일한 접근이다. 2012년 논문으로 비교적 근래이며 석사학위 논문으로 보이고 social 게임을 대상으로 했다. 기본적인 아이디어를 정교하게 하고 방향을 수립해서 MMORPG에 적용하기 위해 진행한다.

Mini-max, alpha-beta, monte carlo tree search의 탐색 기법을 활용하여 AI를 구성하고 플레이 된 결과로 밸런스를 맞추는 데 활용한다. 게임 플레이를 일반화하기 위해 General Game Play와 같이 게임을 표현하는 언어 (Game Description Language)를 도입하고 그 기반 위에 AI를 구현한다. 핵심은 [5] Playing with Artificial Intelligence, [6] Experiments and Evaluation이다.

## 4. 게임 언어 (Game Description Language)

### 개념들

게임을 추상화하는 관점에서 개략적으로 이해하는 것이 좋겠다.

* Game State
* Game Object
  + Game Attributes
  + Game Actions
    - Condition
    - Consequence

게임 상태는 게임 오브젝트들의 상태이며 게임 오브젝트는 속성과 행동을 갖고 있다. 속성은 변수의 값이며 행동은 조건과 결과를 갖고 있다.

이는 Behavior Tree와 같이 행위들을 구성하는 방식이나 HFSM과 같이 상태에서 조건에 따른 처리를 하는 방식과 유사하다. 게임 오브젝트는 Entity나 Actor로 불리는 행위의 주체들과 연관된다.

Condition은 Expression (단위 타잎, 비교 연산자, 연산자, 속성, 오브젝트, 값들에 대한 참조)으로 서술한다.

Consequence (결과) 타잎은 direct (즉시 변경), conditional (특정 조건에 맞으면 적용), timed (일정 시간 후 적용), create (새로운 게임 오브젝트의 생성), create and assign (생성하고 해당 오브젝트의 참조를 얻음), game (게임의 값 변경)으로 나눈다.

게임의 미션이나 특정 조건들을 추적하기 위해 Game Value Store의 key, value 사전을 만들고 여기에 값을 얻거나 저장할 수 있게 한다.

### 언어

대략의 느낌을 얻는 정도면 충분하다.

//one build action for every game object description of a building, or its subclasses business and house, is created.

action globalClass build forEveryGOD building

condition build global.coins > building.coinsCost

//The class name building is replaced by the name of the gameobject description this action is supposed to create.

consequence createBuilding create building

hasConsequence build createBuilding

//Whenever a building is created the counter is increased

action create:building counterUpdate

consequence increaseBuildingCreatedCounter game increase String building, String built, this.name

hasConsequence counterUpdate increaseBuildingCreatedCounter

GDL (게임 언어)를 별도로 만들면 데이터 기반의 게임 구성을 보다 명확하게 이해할 수 있다. RPG의 경우 훨씬 복잡한 다양한 요소를 포함하고 있어 방대한 기획서와 코드를 작성해야 하나 여전히 데이터 기반, 게임에 대한 간결한 묘사 기반으로 서술할 수 있는 언어라는 개념은 유효하고 때로 강력할 수 있다.

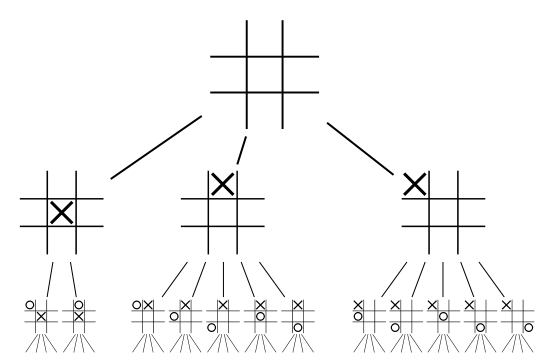
다른 측면에서 데이터 구성을 보다 잘 할 수 있는 어떤 방향을 제시하는 것으로 보인다.

## 5. 인공 지능으로 플레이 하기

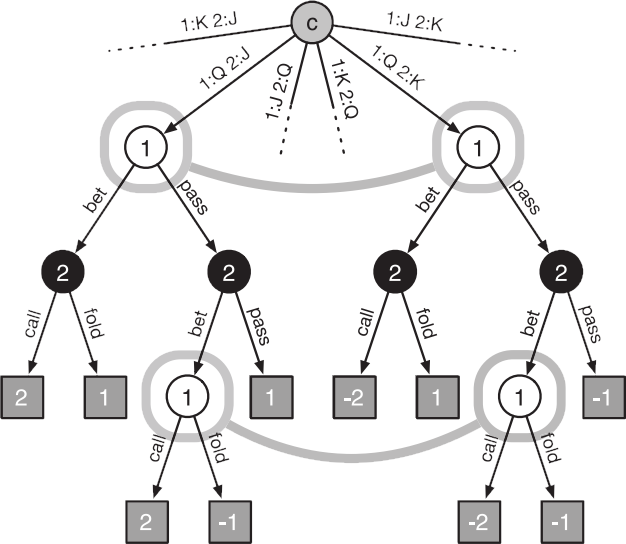
### 게임 상태 공간

선택 가능한 Action에 의해 게임 상태가 바뀐다는 관점으로 설명한다. 따라서, 한번에 선택 가능한 행동들이 상태 공간을 결정한다.

게임 트리는 아래 틱-택-토 그림에서처럼 선택 가능한 행위들을 결정해서 게임 상태를 바꿔 나가는 트리로 본다.



포커의 콜, 다이 (Fold), 베팅의 결정 트리 중 일부를 보여준다.



이 논문에서는 몬테카를로 트리 탐색 (MCTS)을 사용한다.

### 목표 기반 탐색과 게임 플레이

기본 아이디어를 게임 트리 탐색과 목표의 달성으로 잡고 있다. 따라서, 목표의 설정이 필요하다.

* 성장 (Progression)
* 밸런스 (Balance Game Features)

성장은 레벨과 같은 것 (재화의 획득이나 다른 것도 가능)이 된다.

밸런스는 게임 기능들 간의 밸런스를 의미한다. RPG에서는 퀘스트와 전투 또는 파티와 비파티, 던전과 에픽 던전 등으로 생각해 볼 수 있다. (강화에 투자할 것인가, 파밍에 투자할 것인가의 문제도 Feature 간 밸런스로 볼 수 있다)

#### 성장 (Progression) 측정의 예

result = 50% \* xp (경험치) + 30% \* coins + 20% \* missionsFinished (완료 미션 수)

게임 트리 탐색은 미리 최적의 방향을 찾는 것이므로 일정 깊이까지 트리를 따라 예측을 해야 한다. 선택할 수 있는 개수가 많으면 이런 과정이 많은 비용이 들기 때문에 하위 목표를 만족하는 행동을 선택하는 것으로 최적화 한다.

### 연습문제 – 스킬 조합 패턴 파악

RPG에서 한 클래스의 전투 패턴의 다양성을 찾는다. 클래스 A가 액티브 스킬과 패시브 스킬을 갖고 있다고 할 때 다양한 전투 상황에서 선택 가능한 스킬 조합과 자원 배분을 찾는다. 액티브 스킬과 패시브 스킬에 영향을 미치는 요소들이 있다면 이들의 영향에 따른 전투 패턴을 찾는다.

일정 시간 (10분) 동안 가장 많은 적의 HP를 0으로 만들어 사망 시키는 것을 목표로 한다. (아이템이나 경험치 획득 같은 것을 목표로 줄 수도 있다).

선택 가능한 행동을 정의해야 한다. 행동의 선택은 랜덤하게 한다. 이 랜덤한 선택의 트리가 갖고 오는 결과치의 분포를 살피면 선택 가능한 패턴을 알 수 있다.

어떤 패턴이 다른 패턴에 비해 훨씬 낫다면 게임 기능의 밸런스가 편중되었다고 할 수 있다.

만약 한번에 선택 가능한 행동의 개수가 n개이고 m번 진행한다면 번 시뮬레이션 해야 한다.

각 시뮬레이션은 행위들을 a={, , …, }이라고 하면 a의 원소들의 m번 나열이 된다.

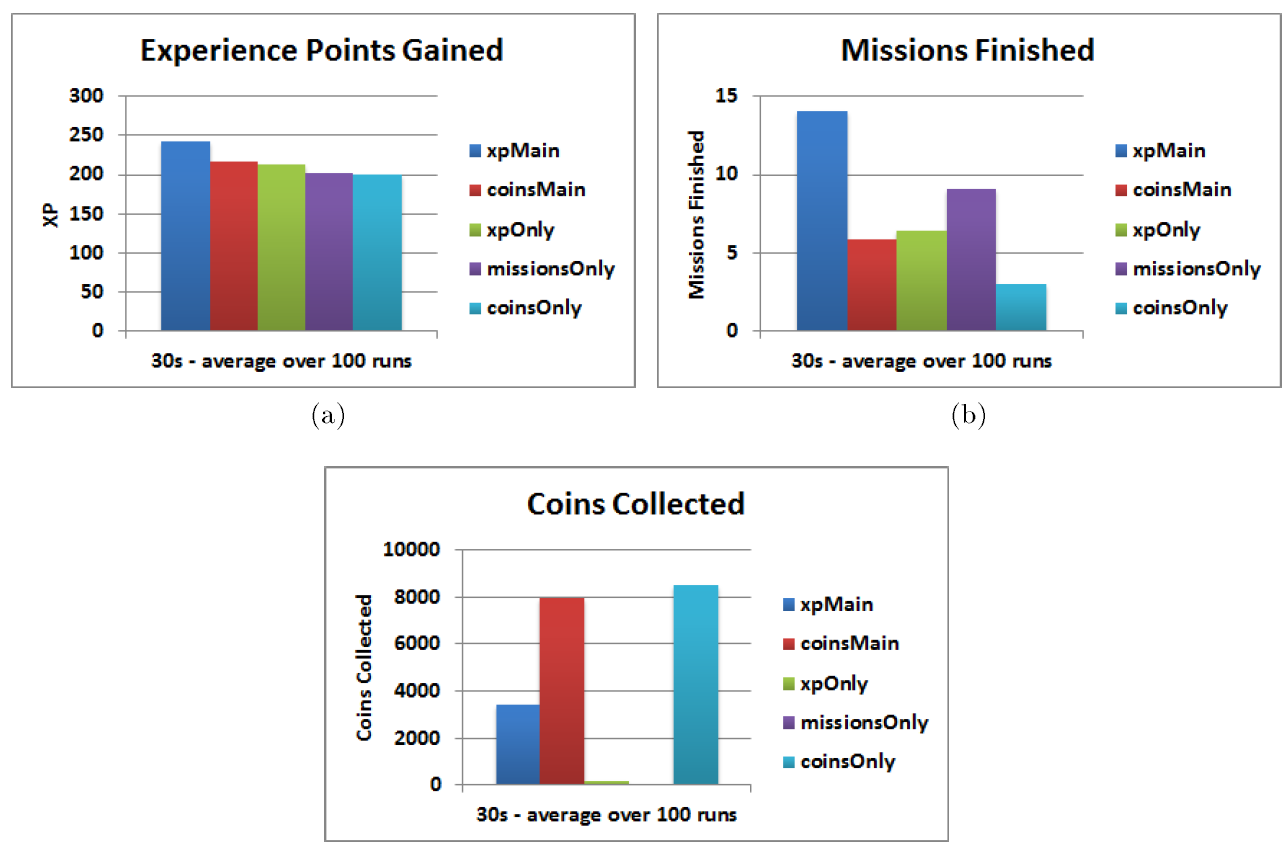
대상이 사람인 PvP의 경우는 훨씬 복잡하긴 하지만 상대방의 플레이 경로와 나의 플레이 경로의 조합으로 결과를 시뮬레이션 할 수 있다. (물론 엄청나게 큰 상태 공간이다).

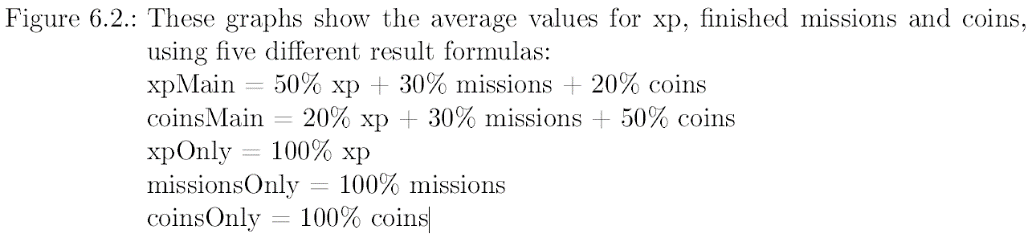
파티 플레이나 길드전과 같이 여러 클래스가 맞붙게 되면 더욱 복잡해진다. 상태 공간이 기하 급수적으로 증가하기 때문이다. 공간 상의 위치 선택도 중요해지기 때문에 더욱 어려워진다. 따라서, 전부를 시뮬레이션 하는 건 가능하지 않다.

혼자 몬스터를 상대로 하는 경우에도 고려해야 할 것들이 많다. 계산량 때문에 단순화 시키는 과정에서 원래 시뮬레이션 결과와 오차가 커질 수 있다.

이런 문제는 매우 일반적으로 보이며 경제학이나 금융 공학 등에서 이미 많이 다루었을 것으로 보인다.

### 결과





목표 값에 따라 다른 플레이 경로를 보여준다. 미션에서 경험치도 많이 준다는 걸 알 수 있고 동전만 모으려면 동전을 모으기 위한 활동을 해야 한다는 걸 보여준다.

### 정리

몬테카를로 트리 서치로 주어진 목표를 만족하는 게임 플레이를 AI가 진행하는 방식으로 게임의 밸런스를 맞추려는 노력을 했다. 하지만, 전체적으로 기술적인 관점에 집중하여 게임 디자인 측면의 의미 있는 결과들은 별반 나오지 않은 것으로 보인다.

80여 개의 선택 가능한 액션이 있는 큰 게임 공간을 효율적으로 시뮬레이션 하기 위한 다양한 기법들의 개발에는 어느 정도 성공한 것으로 보인다.

아이디어는 유효하나 여전히 해결되지 않은 문제로 남아 있고 아마도 작은 문제들을 제외하고는 앞으로도 해결이 쉽지 않을 듯 하다.