제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

AI 를 적용한 전략 게임에 관한 연구

김제민* 박영택*
*숭실대학교 컴퓨터학과
e-mail : kimjemins@hotmail.com

A Research About Strategy Game that Apply AI

Je-Min Kim* Young-Tack Park*
*Dept. of Computer Science, Soongsil University

요 약

요즘 사람들이 많이 즐기는 전략 게임은 전략 시뮬레이션 이라는 말이 무색할 정도로 장르가 가지는 특성을 이행하지 못하고 있다. 그래서 게이머들은 별다른 전략 없이 쉽게 컴퓨터를 상대로 쉽게 게임을 승리 할 수 있게 됐다. 이것은 게임의 재미를 크게 반감시키는 한 요인이 된다. 전략 게임의 컴퓨터 플레이어에게 상황 판단과 학습 능력을 갖게 하면, 게이머가 보다 재미있게 컴퓨터와 대전을 할 수 있다.

본 논문에서는 인공 지능을 가지는 컴퓨터 플레이어에 사용될 Default 추론 엔진과 컴퓨터 플레이어의 작전과 행동을 결정하기 위한 action & strategy generator 시스템을 연구한다. Default 추론 엔진은 귀납적 학습방법을 통해서 컴퓨터 플레이어가 추론 및 학습을 할 수 있는 정보를 생성하게 된다. 이렇게 생성된 정보를 바탕으로 컴퓨터 캐릭터의 행동과 전략을 결정한다. 이에 본 논문에서는 전략 게임에 인공 지능으로 machine leaning 기법 중의 하나인 decision Tree 를 사용하였다. decision Tree 를 적용하여 기존 컴퓨터 플레이어의 행위와 어떻게 다른지 차별성을 밝혀내고, 컴퓨터 플레이어가 향상된 전략을 구사할 수 있게 하는 것이 주된 목표다.

1. 서론

컴퓨터 플레이어가 자신의 행동과 전략을 결정하기 위해서는 행동과 전략 생성에 필요한 데이터가 있어야 한다. 이러한 데이터를 이용하여 컴퓨터 플레이어는 학습을 하게 되고, 그 결과에 따라 자신의 행동과 전략을 결정한다. 이렇게 컴퓨터 플레이어가 자신의 행위와 전략을 결정하기 위해 게이머의 행위에 대해 반응하는 action & strategy system 이 필요하다.

action & strategy system 은 user action/reaction 구조를 가지며, 매번 유저의 action 에 순차적으로 반응하는 stream 구조를 가진다. 이러한 action & strategy system 을 통하여 생성된 데이터를 가지고 캐릭터는 유저의 행위를 추론하며, 추론된 유저의 행위를 이용하여 캐릭터는 전략을 수립하고, 전략에 따라 유저와 대전한다.

2. 관련 연구

softmax 사는 1999 년 1 월에 출시한 자사 게임인 창세기전 3 편" 에 EGO 모드라는 지능형 어시스턴트 시스템을 탑재하였다. EGO 모드는 유저가 게임을 하면서 취했던 상황과 행동들을 컴퓨터 에이전트는 기억하게 된다. 이렇게 기억된 상황과 행동들을 일괄적인 데이터로 저장해 두었다가 게이머가 게임을 풀어가기 어려울 때,기억된 데이터를 미리 정리한 룰에 적용하여 결과 데이터를 출력 받은 후, 출력된 데이터를 바탕으로 게이머에게 적절한 조언을 해준다.

또 maxis 사는 2000 년도에 출시한 심즈에 supervised learning 기법을 이용하여 게임 속에 등장하는 심즈 캐릭터에게 기존의 성장시뮬레이션 게임에서 볼 수 없었던 인공지능 시스템을 적용하였다. 게임 속의 심즈 캐릭터는 적용된 인공지능 시스템을 바탕으로 게이머의 행위를 보고 배울 수 있다. 즉 처음에는 게이머가 심즈의 행동을 하나하나 결정하지만, 나중에는 심즈 캐릭터 스스로 처한 상황에 맞는 행동을 취한다.

이외에도 게임은 아니지만 clicker 의 carrot/stick 반응을 바탕으로 스스로 상황을 추론하여 적절한 행동을 하도록 만들어진 " 디지털 팻트" 가 있다.

3. Background on Learning and strategy game

본 논문이 제안하는 action & strategy generator 시스템을 이해하기 위해서는 non-monotonic reasoning 중의 하나인 디폴트 추론과 귀납적 추론의 기반한 학습 중에 supervised learning 기법인 decision tree 을 잘 이해하여야 한다. 또한 action & strategy generator 시스템을 적

용할 strategy game 에 대해 역시 알아둘 필요가 있다

디폴트 추론

non-monotonic reasoning 란 하나의 논리식의 어떤 형식 이론에 대해 사실일 때, 이러한 논리식은 관계된 형식 이론이 확장된다고 해도 굳이 사실을 유지할 필요가 없는 논리를 말한다. non-monotonic logic 중 하나인 디폴트 추론은 " 어떤 필요조건 α가 사실이라고 증명할 수 있고 β가 사실로 증명된 α와 모순을 일으키지 않으면, β라는 결론을 내릴 수 있다." 라는 한 형태의 규칙이다.

예를 들어 " 엔지니어는 실용적이다." 라는 말이 일관적인 사실로 증명되면, " 엔지니어 홍길동씨는 실용적인 사람이다." 라는 일반적인 결론을 낼 수 있다.

Decision tree

decision tree 는 효과적인 데이터 저장과 질의에 적절한 대답을 하기 위해 설계된 supervised learning 기법중의 하나다. Decision tree 의 각 non-terminal node 에는 terminal node 에 저장된 객체들에 대한 질문이 들어있다.질문에 의한 대답에 의해서 그 질문이 소속된 node 의 child-node 로 연결된 가지를 따라 밑으로 내려가면서 tree 의 어느 위치에 새로운 객체를 저장할 것인가 결정하게 된다. Decision tree 를 구현하기 위해서는 node 의 저장될 데이터들을 추상적 자료 타입으로 정의해야 한다. 추상적인 자료 타입으로 표현된 객체는 각각의 속성과 속성값,질문에 대한 대답(yes/no)으로 분류되며, 이러한 속성과 속성값, 질문에 대한 대답(yes/no)을 바탕으로 평가함수인 gainratio 를 사용하여 들어온 질문에 대한 적절한 대답을 찾을 수 있는 데이터들을 분류해 낼 수 있다.

본 논문에서는 decision tree 를 구현하기 위해 가장 많이 쓰는 알고리즘 중에 하나인 c4.5 를 사용하였다.

strategy game

strategy game 은 컴퓨터와 게이머가 각자에게 할당된 캐릭터를 가지고 전투를 벌여서 최종 승자를 가리는 게임이다. 물론 게이머가 게임에서 지면 진행중인 게임은 끝나게 된다. strategy game 에서 캐릭터는 보통 5 종류로 나뉜다. strategy game 에서 나오는 일반적인 5 종류의 캐릭터는 이동 능력은 떨어지지만 기본적인 공격과 방어에서 우세함을 보이는 전사형 캐릭터, 빠른 이동 능력과 뛰어난 공수 능력을 지녔지만 장애물에 약한 기사형 캐릭터, 공격력은 약하지만 멀리 있는 적 캐릭터를 공격 할 수 있는 궁사형 캐릭터, 방어력은 아주 약하나 강력한 공격마법으로 다수의 적 캐릭터를 공격할 수 있는 마법사형 캐릭터, 약한 공수 능력을 지녔지만 자기 팀의 캐릭터를 보호하는 마법을 지닌 천사형 캐릭터이다.

이들 캐릭터는 서로 상성 관계를 가지고 있으며, 게이머는 자신이 컨트롤하는 캐릭터들의 수와 종류, 각 캐릭터간의 상성 관계를 고려하여 게임에서 제공하는 미션에서 컴퓨터 플레이어와 대전하게 된다

4. Key representation

본 논문이 제안하는 action & strategy generator 시스템은 state, action, strategy, result 로 표현된다. 이 네 가지 key word 는 action & strategy generator 시스템을 이해하는 핵심 단어다.

state

state 는 컴퓨터 플레이어의 캐릭터의 현재 상황을 나타낸다. 캐릭터의 현재 상황이란 필드에서의 자신의 위치, 에너지, 아군과의 거리, 적군과의 거리, 마법 기술 등을 나타낸다.

action

action 은 컴퓨터 플레이어의 캐릭터가 자신이 처한 상황에 맞는 행동을 실행하는 것을 의미한다. 캐릭터의 행동이란 이동이나 공격, 마법구사, 휴식 등을 나타낸다.

strategy

strategy 는 캐릭터의 현재 상황과 그 상황에 맞는 행동을 상호 연관 시켜주는 튜플이다. 캐릭터의 각각의 상황은 action & strategy generator 시스템에 의해 반복 학습되는 데이터로 제공되며, 그 학습 결과는 캐릭터의 행동으로 나타난다.

	A1	A2	A3
S1	Q(1,1)	Q(1,2)	Q(1,3)
S2	Q(2,1)	Q(2,2)	Q(2,3)
S3	Q(3,1)	Q(3,2)	Q(3,3)

Set of all possible actions

Utility of taking action A3 in state S2

Set of all possible states of world

그림 1. strategy: 위 테이블에서, 왼쪽의 열은 캐릭터의 현재 상황을 나타내는 state, 맨 위의 행은 캐릭터가 취해야 할 각각의 행동을 나타내는 action 이다. 테이블의 각 개체는 state 가 주어졌을 때 캐릭터가 나타내는 행동들에 대한 결과값을 가진다.

result

result 는 캐릭터의 행동에 대한 결과를 나타낸다. 즉 strategy 가 적절히 구사되었다는 것을 판단하는 척도가 되며, 행동의 대한 결과는 데이터로 다시 저장되어 action & strategy generator 시스템에 의해 학습되는 자료로 쓰인다.

5. System description

5.1 시스템 구조

전체적인 시스템 구조는 아래의 그림 2 와 같다. 게임을 즐기는 유저는 자신이 제어권이 종료되면 컴퓨터 플레이어에게 제어권을 넘겨준다. 컴퓨터에 제어권이 넘어오면, scene modeler 는 컴퓨터 플레이어의 캐릭터 상황과 유저의 캐릭터 상황을 추상적인 데이터 형태로 정제하여, Default Inference Engine 에 넘겨준다. Default Inference Engine 에 저장된 디폴트 룰과 넘겨받은 추상적인 데이터는 Action Generator 에 넘겨지게 되고, Action Generator 는 데이터에 적합한 디폴트 룰을 찾아내어 컴퓨터 플레이어의 캐릭터가 실행 할

strategy 를 결정한다. 이렇게 결정된 strategy 는 Display Handler 에 의해 컴퓨터 플레이어의 다음 행동으로 표현된다. 만약에 결정된 strategy 가 올바르게 작용하지 않는 경우 Decision Tree Learning Engine 은 그 동안 저장해둔 data-example 을 바탕으로 학습을 실행하게 되고, 학습 결과는 Default Inference Engine 에 디폴트 룰 형태로 저장하게 된다.

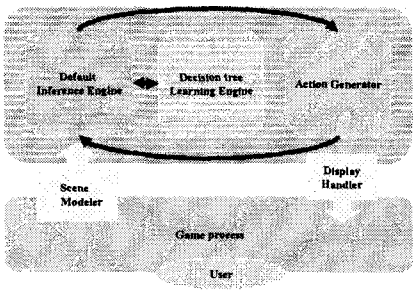


그림 2 action & strategy generator 시스템

5.2 scene modeler 구조

state-representation 은 각 캐릭터의 위치와 속성, 대치상황 등을 추출하여 변수에 담아내는 작업을 한다. 이렇게 변수에 담겨진 데이터들은 state-analysis 가 분석 한다. (예를 들어, 컴퓨터 플레이어의 캐릭터와 유저 캐릭터의 상대적인 거리나, 주변에 있는 적 캐릭터들의 숫자 등등) 분석된 데이터들은 abstraction 을 통하여 일정한 폼으로 vector 구조에 저장되며, 저장된 추상 데이터들은 default reasoning engine 에서 사용할 수 있는 law-data 로 정제된다. (예를 들어, [distance, warrior01, enemy warrior03, left02] 식으로 벡터에 저장된 추상적인 데이터는 distance(warrior01, enemy warrior03, left02)와 같은 law-data 로 변환된다.)

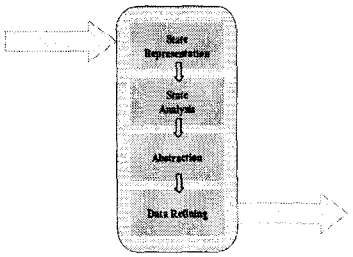


그림 3. scene modeler 의 구조

5.3 default inference engine 구조

scene modeler 로부터 정제된 low-data 들은 law-data process 에 의해 fact 로 변환되어 Rule selector 에 넘겨진다. Rule selector 는 넘겨받은 fact 를 knowledge Base 내에 저장하고, fact 와 fact 를 만족시켜주는 룰을 선택하여 Rule Handler 에 넘겨주면, Rule Handler 는 fact 를 추상적인 데이터로 변환하여 Action Generator 로 넘겨지게 된다.

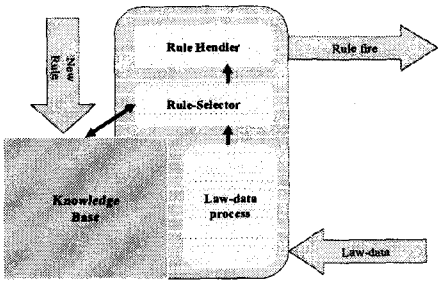


그림 4 default inference engine 구조

5.4 decision Tree learning engine 구조

컴퓨터 플레이어가 구사한 strategy 가 부적합하다고 판단 되었을 경우, 부적합한 strategy 를 야기한 데이터는 데이터 베이스에 새로운 data-example 로 저장되고, 기존에 저장되었던 데이터 data-example 들과 함께 decision tree learning 을 실행하게 되고, learning 결과는 Rule-define 에 의해 새로운 룰로 정의되어 knowledge Base 에 저장된다.

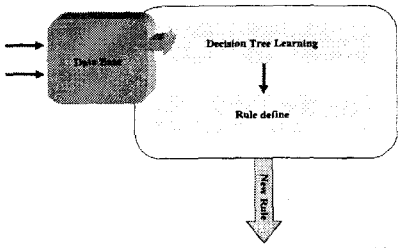


그림 5 decision tree learning engine 구조

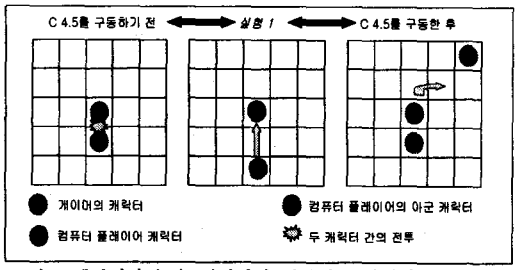
6. 실험 및 결과

java swing 사용하여, action & strategy generation 시스템을 적용한 데모 게임을 제작하여 가상 시나리오를 진행시켜 보았다. Strategy 를 생성하는데 중요한 Decision tree 는 가장 많이 사용되는 알고리즘 중의 하나인 c 4.5 를 사용 하였으며, 반복 진행된 가상 시나리오를 통해 추출된 data-example 들을 c4.5 에 구동시켜서 그 결과를 바탕으로 컴퓨터 플레이어의 strategy 구사 능력을 실험 해보았다.

6.1 실험 1

먼저 에너지가 높은 게이머의 캐릭터를 상대적으로 약한 컴퓨터 플레이어의 캐릭터에 접근 시켰을 경우를 실험해 보았다. 먼저 파랜드 택틱스와 팬저 제너럴 같은 기존 게임(decision tree 와 같은 인공지능 기법이 도입되지 않은 게임)에서 컴퓨터 플레이어의 캐릭터는 게이머의 캐릭터가 자신의 일정 시야 안에 들어오면 무조건 다가가서 전투를 벌인다. 즉 게이머가 컴퓨터 플레이어의 캐릭터 보다 월등히 강한 캐릭터를 컴퓨터 플레이어의 캐릭터에게 접근시킬 경우, 전투를 일으킨 컴퓨터 플레이어의 캐릭터는 게임에서 아웃 된다.

데모 게임 시스템의 경우 게임 초기에는 기존 게임과 같은 결과를 냈으나, 컴퓨터 플레이어는 게임의 결과를 데이터 베이스에 저장하여 c4.5 알고리즘을 통해 반복 학습한다. 그 결과, 자신 보다 강한 게이머의 캐릭터가 접근하면 뒤로 후퇴하거나, 아군 캐릭터가 있는 곳으로 후퇴하는 strategy 를 구사하였다.



.그림 6. 에너지가 높은 게이머의 캐릭터를 상대적으로 약한 컴퓨터 플레이어의 캐릭터에 접근 시켰을 경우

6.2 실험 2

다음은 게이머의 캐릭터가 주변에 동료 캐릭터를 매복시킨 후 컴퓨터 플레이어의 캐릭터를 유인해 봤을 경우를 실험해 보았다. 기존 전략게임에서 컴퓨터 플레이어의 캐릭터는 게이머의 캐릭터가 자신의 일정 시야 안에 들어오면 무조건 접근을 시도하였다. 즉 게이머가 컴퓨터 플레이어의 캐릭터에게 자신의 캐릭터를 접근시키면, 컴퓨터 플레이어의 캐릭터는 게이머의 캐릭터에 다가온다. 이런 식으로 게이머가 자신의 캐릭터 (이 경우 자신의 가장 약한 캐릭터를 이용한다)를 이용하여 컴퓨터 플레이어의 캐릭터를 아군 캐릭터가 배치되어 있는 곳까지 유인하면, 컴퓨터 플레이어의 캐릭터는 게이머의 캐릭터들이 배치된 곳까지 이동하여 전투를 벌인다. 이럴 경우 컴퓨터 플레이어의 캐릭터가 게이머의 캐릭터들을 확실히 제압할 수 있는 능력(기존 게임에서의 보스 급 캐릭터)을 갖추지 않은 이상 컴퓨터 플레이어의 캐릭터는 게임에서 아웃 된다.

데모 게임 시스템은 이 경우 에도 역시 게임 초기에는 기존 게임과 같은 결과를 냈으나, 컴퓨터 플레이어는 게임의 결과를 데이터 베이스에 저장하여 c4.5 알고리즘을 통해 반복 학습한다.

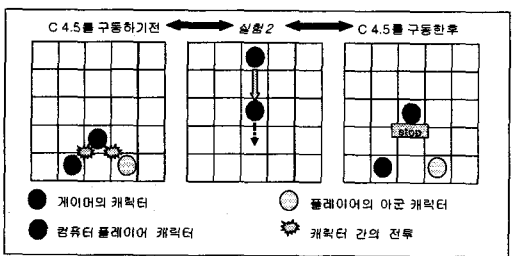


그림 7. 게이머의 캐릭터가 주변에 동료 캐릭터를 매복시킨 후 컴퓨터 플레이어의 캐릭터를 유인해 봤을 경우

그 결과, 자신 보다 약한 게이머의 캐릭터가 접근할 경우 일단, .접근을 시도하다가 게이머의 캐릭터가 자신의 아군 캐릭터들이 배치되어 있는 곳으로 유인하려고 하면 컴퓨터 플레이어의 캐릭터는 더 이상 게이머의 캐릭터가 있는 곳으로 이동을 하지 않는 strategy 를 구사 하였다..

7. 결론

전략 게임에서 컴퓨터 플레이어의 캐릭터가 strategy 를 가지고 게이머와 대전을 하게 되면, 게이머는 게임을 보다 흥미 있게 즐길 것 이다. 전략 게임은 현재까지 게이머가 보다 쉽게 게임을 즐길 수 있도록 하는 효과적인 인터페이스가 주로 연구되어 왔으나 현재에는 전략을 구사할 수 있는 컴퓨터 플레이어의 캐릭터들에 대한 연구도 필요하다.

본 연구에서는 전략 게임에서 컴퓨터 플레이어의 캐릭터가 현재의 state 를 바탕으로, strategy 를 생성하는 action & strategy system 을 연구 하였다. action & strategy system 은 캐릭터가 strategy 를 구사하기 위해 필요한 정보를 학습할 수 있는 구조를 가지며, 캐릭터가 stra-tegy 를 추론할 수 있게 하기 위해서 scene modeler, default inference engine, decision Tree learning engine 시스템과 생성된 strategy 를 저장하는 Knowledge Base 를 가진다. action & strategy system 을 이용하여 생성된 strategy 는 게임 중에 컴퓨터 플레이어 캐릭터 의 행동으로 표현되며. action & strategy system 을 적용한 데모 게임은 jdk 1.3 과 swing 으로 제작하였다.

제작된 데모 게임 시스템은 기존의 전략 게임에서 부족했던 컴퓨터 플레이어의 캐릭터가 전략을 구사하는 부분을 보완하였다. 즉, 기존 게임에서 보여 주었던 컴퓨터 플레이어 캐릭터의 단순한 행동 (적 캐릭터가 자신의 시야에 들어오면, 무조건 적 캐릭터에게 이동하여 전투를 벌이는 것)을 action & strategy system 에서 생성된 strategy 를 이용하여 보다 다양한 action (뒤로 후퇴, 아군 캐릭터가 있는 장소로 후퇴, 잠시 멈춤, 적절한 타이밍의 아군 캐릭터 치료) 을 보여줄 수 있도록 보완 되었다.

참고문헌

[1]Brodley, C.E. & utgoff, P.E(1995) Multivariate decision tree. Machine Learning, 19, 45-77.

[2]PRYOR, K.1999. Clicker training for dogs. Sunshine Books, Inc., Waltham, MA.

[3]Mitchell, K. 1997. Machine learning. McGraw Hill, New York,NY

[4]Thomas, D. 1995. Artificial Intelligence. Addison-Wesley Pte. Ltd 187-212

[5] http://www.digipen.edu/~tfolsom/cs381/lectures/black&white