휴리스틱 기반의 유전 알고리즘을 활용한 경로 탐색 알고리즘

고정운*, 이동엽**

공주대학교 게임디자인학과

nightofelf@nate.com*, kongjumail@gmail.com**

Path-finding Algorithm using Heuristic-based Genetic Algorithm

Jung-Woon Ko*, Dong-Yeop Lee**

Dept. of Game, Kongju National University

요 약

경로 탐색 알고리즘은 이동 가능한 에이전트가 게임 내의 가상 월드에서 현재 위치로부터 목적지까지 가는 경로를 탐색하는 알고리즘을 뜻한다. 기존의 경로 탐색 알고리즘은 A*, Dijkstra와 같이 비용 기반으로 그래프 탐색을 수행한다. A*와 Dijkstra는 월드 맵에서 이동 가능한 노드와 에지 정보들을 필요로 해서 맵의 정보가 다양하고 많은 온라인 게임에 적용하기 힘들다. 본 논문에서는 가변환경이나 맵의 데이터가 방대한 게임에서 적용 가능한 경로 탐색 알고리즘을 개발하기 위해 맵의 정보 없이 교배, 교차, 돌연변이, 진화 연산을 통해 해를 찾는 유전 알고리즘(Genetic Algorithm, GA)을 활용한 Heuristic-based Genetic Algorithm Path-finding(HGAP)를 제안한다. 제안하는 알고리즘은 Binary-Coded Genetic Algorithm을 기반으로 하며 목적지에 더 빨리 도달하기 위해 목적지로 가는 경로를 추정하는 휴리스틱 연산을 수행하여 경로를 탐색한다.

ABSTRACT

The path-finding algorithm refers to an algorithm for navigating the route order from the current position to the destination in a virtual world in a game. The conventional path-finding algorithm performs graph search based on cost such as A-Star and Dijkstra. A-Star and Dijkstra require movable node and edge data in the world map, so it is difficult to apply online games with lots of map data. In this paper, we provide a Heuristic-based Genetic Algorithm Path-finding(HGAP) using Genetic Algorithm(GA). Genetic Algorithm is a path-finding algorithm applicable to game with variable environment and lots of map data. It seek solutions through mating, crossing, mutation and evolutionary operations without the map data. The proposed algorithm is based on Binary-Coded Genetic Algorithm and searches for a path by performing a heuristic operation that estimates a path to a destination to arrive at a destination more quickly.

Keywords : Path-finding(경로 탐색), Genetic Algorithm(유전 알고리즘), Heuristic(휴리스틱)

※ 이 논문은 2017년 공주대학교 학술연구지원사업의 연구지원에 의하여 연구되었음

Received: Jul. 05. 2017 Accepted: Aug. 10. 2017
Corresponding Author: Dong-Yeop Lee(Kongju National Univ.)
E-mail: kongjumail@gmail.com

ISSN: 1598-4540 / eISSN: 2287-8211

© The Korea Game Society. All rights reserved. This is an open-access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0>), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. 서 론

게임 산업이 발전함에 따라 사람이 아닌 플레이어(Non Player Character, NPC)의 지능인 인공지능(Artificial Intelligence, AI) 연구가 활발하게 진행되고 있다. 게임 플레이어는 조금 더 지능화되고 똑똑한 인공지능과 게임을 하는 것을 원하고 있다. 이에 따라 게임 개발사도 인공지능의 중요성을 인식하고 게임에 성능 좋은 인공지능 NPC를 사용하기 위해 노력하고 있다[1,2].

특히 게임에서 가장 많이 사용되는 인공지능은 경로 탐색 알고리즘이다. 경로 탐색 알고리즘은 NPC뿐만 아니라 사용자의 편의를 위해 플레이어 캐릭터에게도 적용되는 게임들이 많기 때문에 많은 연구가 이루어지고 있다[3,4].

경로 탐색 알고리즘을 사용해서 플레이어가 가고 싶은 위치를 선택하게 되면 캐릭터가 목표 위치까지 이동하는 동안 플레이어는 다른 행동을 할 수 있기 때문에 시간을 효율적으로 활용할 수 있다. 또한 NPC가 플레이어를 공격하기 위해 플레이어에게 다가오는데 제대로 경로를 탐색하지 못하고 장애물 앞에서 멈춰버리면 플레이어는 게임에 대한 흥미를 금방 잃게 될 것이다[5,6].

게임에서의 경로 탐색 알고리즘에 흔히 사용되는 방법은 경로에 대한 비용을 기반으로 최단거리를 탐색하는 A* 알고리즘이다. A* 알고리즘은 출발노드에서 현재노드까지의 거리 $g(goal)$ 와 현재노드에서 목표노드까지의 추정거리 $h(heuristic)$ 의 합이 가장 작은 경로를 선택하기 때문에 사용자에게 최단거리의 경로를 제공할 수 있다. 하지만 맵에 존재하는 노드와 에지 정보를 데이터 공간으로 구성하여야 하며, 맵의 크기가 크면 A*가 경로 탐색에 사용하는 열린 목록(Open List)과 닫힌 목록(Close List)에 수천에서 수만 개의 노드들을 저장하기 위한 데이터 공간을 요구하기 때문에 많은 메모리 용량을 필요로 하며, CPU에 큰 부담이 가서 게임의 다른 연산에 제약적일 수 있다[7,8].

본 논문에서는 제약적인 환경에서만 사용 가능

한 A* 알고리즘 대신 사용할 수 있는 경로 탐색 알고리즘을 제안한다. 제안하는 알고리즘은 교배, 교차, 돌연변이, 진화 연산을 통해 해를 찾는 유전 알고리즘(Genetic Algorithm, GA)에 휴리스틱 연산을 활용한 Heuristic-based Genetic Algorithm Path-finding(HGAP)을 개발하였다.

제안하는 알고리즘의 효율성을 검증하기 위해 20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 A* 알고리즘과 제안하는 알고리즘의 경로 탐색을 실험하였다. 실험에서는 탐색에 사용된 노드 수와 탐색에 소요된 시간, 탐색 후 나온 이동경로를 검사하였다.

2. 관련연구

2.1 A* 알고리즘(A-Star Algorithm)

A* 알고리즘은 월드 맵에 존재하는 에이전트가 현재 위치에서 목표지점으로 가는 경로 중 비용이 가장 적게 드는 경로를 탐색하는 알고리즘이다. A* 알고리즘은 비용 기반 그래프 탐색 알고리즘으로 월드에 있는 노드와 에지의 정보들을 비교하여 최적의 경로를 탐색한다. A* 알고리즘은 경로를 탐색하는 동안 출발 노드로부터 현재 머물고 있는 노드 n 까지의 경로 비용을 나타내는 $g(goal)$ 와 노드 n 으로부터 목표 노드까지의 경로 비용 추정치 $h(heuristic)$ 값의 합 $f(fitness)$ 를 끊임없이 계산하여 f 값이 가장 적게 나온 경로를 선택한다[9].

[Fig. 1] A* Algorithm Path-finding

2.2 유전 알고리즘(Genetic Algorithm)

유전 알고리즘은 생물의 진화와 유전자의 적자 생존 원리를 코드로 옮겨놓은 것이다. 유전 알고리즘에서는 여러 유전자들이 모여 하나의 세대를 이루게 된다. 유전 알고리즘의 세대들은 최적의 해를 찾기 위해 평가를 받게 되고, 최적의 해를 찾을 때까지 유전자의 교배, 교차와 돌연변이 연산을 통해 새로운 세대를 생성한다[10,11].

3. Heuristic-based Genetic Algorithm Path-finding

유전 알고리즘은 생물의 진화를 모방한 인공지능으로 유전자들로 구성된 세대를 생성하고, 세대가 해에 적합한지 평가하고 적합하지 않으면 현재 세대의 교배, 교차, 돌연변이 연산을 통해 새로운 세대를 생성해 최적의 해를 찾는 알고리즘이다.

[Fig. 2] Heuristic-based Genetic Algorithm Path-finding

3.1 휴리스틱 기반의 초기 세대 생성

초기 세대의 유전자는 이동 가능한 경로에 대한 방향들을 염색체로 구성하게 된다. 각 이동 방향은 이진수(Binary-Code) 2자리가 한 쌍을 이룬다. 목표 지점까지 충분히 도착할 수 있도록 맵의 크기에 따라 적합한 염색체의 길이를 평가하여 생성된다.

Binary-Code	00	01	10	11
Direction	Up	Down	Left	Right

[Fig. 3] Binary-Coded Based Genome

염색체의 이동 방향은 휴리스틱(Heuristic) 연산을 활용하여 출발지점에서 목표지점으로 이어지는 직선 방향의 경로가 더 높은 확률로 선택될 수 있도록 설정하였다.

Heuristic Direction		00	01	10	11
X Axis	10	20%	20%	40%	20%
	11	20%	20%	20%	40%
Y Axis	00	40%	20%	20%	20%
	01	20%	40%	20%	20%
X Axis & Y Axis	00	35%	15%	-	-
	01	15%	35%	-	-
	10	-	-	35%	15%
	11	-	-	15%	35%

[Fig 4] Heuristic-based Chromosome Moving Direction Selection Probability

3.2 적응도 평가

적응도 평가 기준은 유전 알고리즘이 최적의 해를 찾는데 영향을 준다. 본 논문에서는 경로를 찾는 것에 중점을 두고 있기 때문에 적응도는 목표 지점에 얼마나 가깝게 도달했는지 평가한다. 적응도 평가에 사용되는 수식은 아래 그림과 같다. Vf

는 해 f 가 탐색을 종료한 시점에서 목표지점까지의 거리를 의미하고, P 는 사용자에게 의해 부여되는 상수를 의미한다.

$$Fitness(f) = \frac{P}{V_f} \quad (\text{eq. 1})$$

3.3 유전자 교배

유전자 교배는 현재 세대에서 최적의 해를 찾지 못했을 때 다음 세대의 유전자들을 생성하는 과정에서 하나이다. 본 논문에서는 현재 세대의 염색체들 중 적응도에 비례하여 염색체를 임의의 선택하는 룰렛 선택 방법을 사용하여 두 개의 부모 염색체를 선택한다.

[Fig. 5] Genome Mating: Roulette Wheel Selection

3.4 교차와 돌연변이

교차 연산은 선택된 두 개의 부모 염색체를 교차율에 의거하여 비트들을 치환하는 과정이다. 교차연산을 위해 염색체의 비트들은 이진수(Binary-Code) 형태로 생성하게 된다.

Genome Type	Genome Binary-Code
Genome:Mom	10101101 00110101
Genome:Dad	00110100 11100101
Genome:Baby1	10101101 11100101
Genome:Baby2	00110100 00110101

[Fig. 6] Genome Crossover Operation

돌연변이 연산은 변이율에 의거하여 염색체에서 임의의 비트가 0은 1로 1은 0으로 뒤집어지는 것을 의미한다.

4. 실험 및 결과

본 논문에서 제안하는 알고리즘의 효율성을 검증하기 위해 20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 A* 알고리즘과 제안하는 알고리즘의 경로 탐색에 사용된 노드의 수와 경로 탐색 시간, 최종 이동 경로의 거리를 비교하였다. [Fig. 7,8]은 20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 A* 알고리즘과 HGAP 알고리즘으로 경로 탐색을 수행한 결과이다.

[Fig. 7] A* Algorithm Path-finding

[Fig. 8] HGA Path-finding

4.1 경로 탐색에 소요된 시간

20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 A* 알고리즘과 제안하는 HGAP 알고리즘을 활용해 각각 1000회씩 경로 탐색을 수행했을 때의 경로 탐색에 소요된 시간이다.

[Fig. 9] Time spent for Path-finding E20

[Fig. 10] Time spent for Path-finding S20

20x20 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 0.031, HGAP 알고리즘 0.0038로 HGAP 알고리즘이 평균 8.2배 빠른 것을 확인하였다.

20x20 크기의 장애물이 배치된 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 0.02, HGAP 알고리즘 0.01로 HGAP 알고리즘이 평균 2배 빠른 것을 확인하였다.

[Fig. 11] Time spent for Path-finding E50

[Fig. 12] Time spent for Path-finding S50

50x50 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 0.5, HGAP 알고리즘 0.005로 HGAP 알고리즘이 평균 100배 빠른 것을 확인하였다.

50x50 크기의 장애물이 배치된 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 0.2, HGAP 알고리즘 0.076로 HGAP 알고리즘이 평균 2.6배 빠른 것을 확인하였다.

[Fig. 13] Time spent for Path-finding E100

[Fig. 14] Time spent for Path-finding S100

100x100 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 7.2, HGAP 알고리즘 0.036으로 HGAP 알고리즘이 평균 200배 빠른 것을 확인하였다.

100x100 크기의 장애물이 배치된 맵에서 경로 탐색을 수행했을 때 평균 탐색 시간은 A* 알고리즘 4.86, HGAP 알고리즘 0.09로 HGAP 알고리즘이 평균 54배 빠른 것을 확인하였다.

4.2 탐색에 사용된 노드 수

20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 A* 알고리즘과 제안하는 HGAP 알고리즘을 활용해 각각 1000회씩 경로 탐색을 수행했을 때의 탐색에 사용된 노드 수이다.

[Fig. 15] Node count used in the search E20

[Fig. 16] Node count used in the search S20

20x20 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 322, HGAP 알고리즘 203.5로 HGAP 알고리즘이 평균 1.58배 적게 사용한 것을 확인하였다.

20x20 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 203, HGAP 알고리즘 171.6으로 HGAP 알고리즘이 평균 1.18배 적게 사용한 것을 확인하였다.

[Fig. 17] Node count used in the search E50

[Fig. 20] Node count used in the search S100

[Fig. 18] Node count used in the search S50

50x50 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 2302, HGAP 알고리즘 394.4로 HGAP 알고리즘이 평균 5.84배 적게 사용한 것을 확인하였다.

50x50 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 1442, HGAP 알고리즘 762.1로 HGAP 알고리즘이 평균 1.89배 적게 사용한 것을 확인하였다.

[Fig. 19] Node count used in the search E100

100x100 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 9602, HGAP 알고리즘 1554로 HGAP 알고리즘이 평균 6.18배 적게 사용한 것을 확인하였다.

100x100 크기의 장애물이 없는 빈 맵에서 경로 탐색을 수행했을 때 평균 사용 노드 수는 A* 알고리즘 7782, HGAP 알고리즘 1505로 HGAP 알고리즘이 평균 5.17배 적게 사용한 것을 확인하였다.

5. 결 론

본 논문에서는 맵의 데이터가 방대한 게임이나 가변적인 환경에서 사용이 어려운 A* 알고리즘 대신 활용할 수 있는 경로 탐색 알고리즘을 제안하였다.

제안한 알고리즘은 맵의 정보를 필요로 하지 않으며 교배, 교차, 돌연변이, 진화 연산을 통해 최적의 해를 찾는 유전 알고리즘에 목적지로 가는 경로를 추정하는 휴리스틱 연산을 활용하였다.

제안한 알고리즘의 효율성을 검증하기 위해 A* 알고리즘과 HGAP 알고리즘의 경로 탐색을 실험하였다. 실험은 20x20, 50x50, 100x100 크기의 장애물이 없는 빈 맵과 장애물이 배치된 맵에서 동일한 시작지점과 목표지점을 설정하여 A* 알고리즘과 HGAP 알고리즘의 경로 탐색을 1000번씩 수행하였으며, 탐색에 사용된 노드 수와 탐색 소요 시간, 결과로 나온 이동 경로의 거리를 비교하였다.

실험 결과 본 논문에서 제안하는 HGAP 알고리즘이 항상 최단거리를 경로로 선택하지는 않지만 탐색에 사용된 노드 수와 탐색에 소요되는 시간은 A* 보다 월등히 적게 나온 것을 확인할 수 있다.

또한 맵에 노드와 에지에 대한 정보를 따로 저장하지 않아도 되고, A* 알고리즘에서 열린 목록, 닫힌 목록에 사용되는 추가적인 데이터 공간을 필요로 하지 않기 때문에 제안하는 알고리즘이 가변적인 환경이나 맵의 데이터가 방대한 게임에서 활용하기 좋다.

향후에는 최단거리 경로에 더 가깝게 탐색할 수 있도록 연구를 진행하여 HGAP 알고리즘의 성능을 향상시킨다.

ACKNOWLEDGMENT

This research was financially supported by Kongju National University.

REFERENCES

- [1] J. E. Laird, "Using a computer game to develop advanced AI", 2001 IEEE Computer, July 2001.
- [2] Hyung-il Kim, "State based Context Awareness Method for Non-Player Character", Journal of Korea Game Society 2014, pp.93-102, Feb 2014.
- [3] Sung-lok Choi, "Fast Any-angle Path Planning on Grid Maps with Non-collision Pruning", 2010 IEEE International Conference, Dec 2010.
- [4] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids", in Proceedings of AAAI Conference on Artificial Intelligence(AAAI), 2007.
- [5] Myoun-jae Lee, "An Artificial Intelligence Evaluation on FSM-Based Game NPC", Journal of Korea Game Society 2014, pp.127-136, Oct 2014.
- [6] Eun-sol Kim, Hye-yeon Kim, Kyeon-ah Yu, "Implementation of Adaptive Navigation for NPCs in Computer Games", Journal of Korea Information Science Society 2016, pp.222-228, Feb 2016.
- [7] Sung-hyun Cho, "Reducing Search Space of A* Algorithm Using Obstacle Information", Journal of Korea Game Society 2015, pp.179-188, Aug 2015.
- [8] Abhilash Sreeramaneni, "A Real-Time Centralized Multi-Agent Pathfinding System using Systemized Iterative Deepening A*", The Graduate School of Seoul National University of Science and Technology, July 2015.
- [9] Jeong-won Ko, Sung-jun Park, "Path-Plan Algorithm using Collision Line Search Process", Proceedings of Korea Game Society Spring Conference 2012, pp.249-254, June 2012.
- [10] Myun-sub Lee, Byeong-heon Cho, Sung-hoon Jung, Yeong-rak Seong, Ha-ryoung Oh, "Performance Evaluation of Intelligent Characters for Fighting Action Games using Genetic Algorithms", Journal of Institute of Electronics Engineers of Korea 2004, pp.119-128, Dec 2004.
- [11] Hak-su Lee, Don-jung Choi, Hye-wuk Jung, Jee-hyong Lee, "Intelligent Tutoring System based on Genetic Algorithm", Proceedings of Korea Institute of Intelligent Systems Fall Conference 2010, pp67-69, Nov 2010.

고 정운(Ko, Jung Woon)

약 력 : 2012 호서대학교 게임공학과(공학사)
2014 호서대학교 게임학과(공학석사)
2017~현재 공주대학교 게임디자인학과 박사과정

관심분야 : 게임 프로그래밍, 인공지능, 기능성 게임

이 동 엽(Lee, Dong Yeop)

2002년 10월 De Montfort University(Master of Arts)
2013년 8월 상명대학교 감성공학과 (감성공학박사)
2014년 3월- 현재 공주대학교 게임디자인학과 조교수

관심분야 : 게임디자인, 게임 모델링, 멀티미디어