최적 동선을 고려한 MMORPG 퀘스트 보상 설계 기법

강신진*, 신승호+, 조성현*
*홍익대학교 게임학부, +고려대학교 정보통신대학 컴퓨터학과
directx@korea.ac.kr, winstorm@korea.ac.kr, scho@hongik.ac.kr

A MMORPG Quest Reward Design Technique By Considering Optimal Quest Play Paths

Shin Jin Kang, Seung Ho Shin, Sung Hyun Cho
School of Games, Hongik University
Department of Computer Science and Engineering, Korea University

요 약

퀘스트(Quest) 시스템은 MMORPG에서 콘텐츠를 제공하는 핵심 시스템 중 하나이다. 퀘스트 디자인 업무에서 퀘스트 보상 설정은 게임 내 작용하는 여러 구성 요소의 높은 조합 복잡도로 인해 적절한 보상 수준을 산출하기가 어려운 문제에 속한다. 본 논문에는 퀘스트 보상 문제를 순회 판매원 문제(Traveling Salesman Problem, TSP)로 모델링하여 해결함으로써 적절한 보상 수치를 자동적으로 산출해 낼 수 있는 기법을 제안하였다. 이를 통해 퀘스트 디자인 단계에서 퀘스트 보상 수치 확정을 위한 강도 높은 테스트 부담을 줄이고 정확한 보상 수치를 산출하는 데 도움이 될 것이다.

ABSTRACT

A quest system is one of the important parts in the MMORPG (Massive Multiplayer Online Role Playing Game) contents. Because of its complexity in combining various content components, quest reward design belongs to a complicated work in estimating quest reward levels correctly in the initial development stage. In this paper, we suggest a new quest reward design technique by considering optimal quest play paths. We model a quest reward problem as the TSP (Traveling Salesman Problem) and solve that by adopting genetic algorithms. With our system, game designers easily estimate the optimal quest play path and it can be useful in reducing the trial-errors in the initial quest design process.

Keyword : Genetic Algorithm, Evolutionary Computation, Quest Design, Traveling Salesman Problem

접수일자 : 2009년 06월 24일
심사완료 : 2009년 08월 07일
교신저자 : 조성현
※ 이 논문은 2009년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.

1. 서 론

MMORPG은 다수의 시스템으로 구성된 가상의 세계를 구성하여 이를 플레이어에게 제공하는 게임이다. 주요 MMORPG의 시스템들로는 전투, 성장, 생산, 채집, 커뮤니티 시스템 등이 존재한다. 이 중 퀘스트 시스템은 MMORPG 내에서 이야기를 전달하는 가장 유용한 시스템 중 하나이다. 플레이어는 퀘스트 시스템이 제공하는 퀘스트 수행을 통해 게임 내 주요 이야기를 전달받게 되고 플레이의 장단기 목표를 부여 받게 된다. 또한 퀘스트 수행 완료시 다양한 보상을 받음으로써 지속적인 게임 플레이에 대한 동기 부여를 받게 된다. 퀘스트 시스템은 플레이어가 직접적이고도 장기간 경험하게 되는 요소이므로 해당 콘텐츠 구성시 많은 노력을 기울여 설계하게 된다. 퀘스트 설계시 고려해야 할 내용들은 퀘스트의 주제, 내용, 이야기, 퀘스트 수행 내용, 퀘스트 동선 설계, 퀘스트 보상 수준 등이 존재하며, 이러한 모든 내용들이 복합적으로 조합되어 퀘스트를 구성하게 된다[1].

MMORPG 개발팀은 플레이어에게 끊임없이 연결되는 이야기, 플레이 요소 그리고 이에 따른 적절한 보상을 잘 조합된 퀘스트 제작을 목표로 한다. 그러나 주어진 시스템 내에서 최대한의 자유를 보장하는 MMORPG 특성상 플레이어의 행동은 예측하기 어려운 경우가 많아 개발팀에서 초기에 기획하고 예상했었던 많은 부분에 대해 예측 오차가 발생하게 된다.

퀘스트 콘텐츠에서 큰 예측 오차를 발생시키는 부분은 바로 퀘스트 보상 부분이다. 퀘스트 설계와 구성 등은 원래 개발팀의 의도에 맞게 구성될 가능성이 높으나, 플레이어가 어떠한 행동을 하여 완료할 것인지 예상하고 입력한 수치들이 플레이어의 자유 행동에 의해 어긋나게 될 경우가 많다. 이는 해당 퀘스트 보상을 너무 낮거나 혹은 너무 높게 만들게 됨에 따라 퀘스트 수행 의욕 저하 혹은 해당 퀘스트 남용(Abusing) 행위로 이어지게 만드는 요인이 되게 된다.

본 논문에서는 이러한 퀘스트 수행 보상 설계에 있어 기존의 경험 혹은 직접적인 플레이 테스트에 의존하였던 방식에서 벗어나 수치적으로 최적화된 새로운 방법을 제안하고자 한다. 본 논문에서 해결하고자 하는 부분은 퀘스트의 수행 보상에 있어 큰 오차를 발생시키는 퀘스트 수행 순서 부분에 관한 것으로, 이를 순회 판매원 문제로 모델링하여 유전자 알고리즘을 적용하여 해결함으로써 최적화된 퀘스트 보상 수치를 산출하였다.

2. 퀘스트 시스템 개요

MMORPG에서 퀘스트 시스템은 크게 퀘스트 획득, 퀘스트 수행, 퀘스트 보상의 3단계로 이루어진다[2].

2.1 퀘스트 획득

MMORPG 게임 내에서 플레이어가 퀘스트를 받는 일련의 행위를 퀘스트 획득 단계라 한다. 모든 퀘스트는 해당 퀘스트를 플레이어가 획득한 후에 이를 수행할 수 있다. 퀘스트 획득은 해당 퀘스트를 제공하는 NPC와의 대화를 통하여 이루어지게 된다. 퀘스트 NPC는 플레이어가 현재 레벨에서 수행 가능한 퀘스트 항목들을 가지고 있고 플레이어가 이중 한 개를 선택함으로써 해당 퀘스트를 획득할 수 있다. 획득된 퀘스트는 플레이어의 퀘스트 리스트에 올라가게 되고 이를 수행 완료 시 리스트에서 삭제되게 된다[3]. 게임 내에서의 퀘스트 NPC의 위치는 퀘스트 수행 동선에 큰 영향을 미친다. 퀘스트 NPC의 배치가 플레이어의 진행 동선과 자연스럽게 연계되어 있을 때 플레이어는 기존의 동선에서 크게 벗어나지 않고도 퀘스트를 획득할 수 있지만 그렇지 않을 경우 퀘스트를 받기 위해 많은 거리를 움직이거나 해당 퀘스트 존재를 놓치는 경우가 발생한다. NPC와의 대화 이외에도 다른 다양한 방식으로 퀘스트를 습득할 수 있지만 기본적으로 특정 지역에 가서 특정 행위를 해야만

[그림 1] 월드 오브 워크래프트 퀘스트 대화창

얻을 수 있다는 점에서 획득 방식은 모두 지역 기반 퀘스트로 한정시킬 수 있다. [그림 1]은 블리자드(Blizzard)사의 MMORPG 게임인 월드 오브 워크래프트(World of Warcraft)[4]의 퀘스트 획득창으로서 일반적인 퀘스트 획득창의 모습을 보여주고 있다.

2.2 퀘스트 수행

퀘스트 수행 단계는 플레이어가 받은 퀘스트를 수행하는 단계이다. 획득한 퀘스트의 내용에 따라 퀘스트의 수행 내용도 바뀌게 된다. MMORPG에서 범용적으로 사용되는 퀘스트의 종류는 다음과 같다.

1. 대화 퀘스트: 특정 NPC와 대화를 하는 퀘스트, 지정된 퀘스트 NPC를 찾아 대화하면 수행이 완료된다.
2. 배달 퀘스트: 특정 오브젝트를 다른 NPC에게 전달하는 퀘스트, 특정 오브젝트를 지정된 NPC에게 전달하면 수행이 완료된다.
3. 사냥 퀘스트: 특정 몬스터를 일정 횟수 사냥하는 퀘스트, 지정된 몬스터를 사냥할 때마다 카운트가 올라가고 일정 카운트 이상이 되는 순간 퀘스트 수행 완료 조건을 만족한다.
4. 수집 퀘스트: 특정 아이템을 일정 횟수 모아오는 퀘스트, 특정 아이템을 채집 혹은 사냥 후 아이템 드랍을 통해 획득하는 퀘스트, 일정 카운트 이상이 되는 순간 퀘스트 수행 완료 조건을 만족한다.

대화 퀘스트나 배달 퀘스트는 해당 NPC들과의 대화만으로도 퀘스트를 수행 완료할 수 있는 반면에 사냥 수집 퀘스트는 퀘스트 획득 이후 목표 몬스터나 아이템이 있는 레벨로 직접 이동하여 요구조건(사냥 또는 수집)을 만족시켜야 퀘스트가 완료된다. 대화, 배달 퀘스트는 콘텐츠 상 주로 새로운 지역으로 플레이어의 이동을 유도할 때 쓰이고, 사냥, 수집 퀘스트는 콘텐츠의 반복적 소모를 통해 장시간 플레이를 유도하기 위해 사용된다. 블록버스터 급의 MMORPG일 경우 이와 같은 퀘스트들은 수백에서 수 천여 개 이상 제공하며 플레이어 캐릭터의 성장을 장기간 유도한다[3].

2.3 퀘스트 보상

플레이어가 퀘스트를 수행하여 수행 완료 조건을 만족시키게 되면 플레이어는 이에 대한 보상을 받을 수 있다. 퀘스트 보상은 일반적으로 퀘스트를 제공한 NPC에게 다시 대화를 하였을 경우 발생하거나 혹은 수행 완료시 자동으로 이루어지게 된다. 퀘스트 보상으로는 게임 내 화폐, 아이템, 경험치 등이 그 대상이 되게 된다. 어려운 퀘스트일수록 해당 퀘스트 완료시 더 좋은 보상을 받게 되는데 이러한 보상이 좋은 퀘스트 수행을 통해 플레이어는 게임 내 성장을 좀 더 단축시킬 수 있게 된다. 퀘스트를 디자인할 때 이러한 보상의 정도는 가장 중요한 기획 요소 중 하나로 보상 부분에 대한 기준이 충분하지 않거나 과도할 경우 지속적 게임 플레이에 악영향을 미치게 된다.

3. 퀘스트 보상 측정

MMORPG에서 퀘스트 보상 기준으로는 다양한 방법들이 사용되지만 가장 일반적인 것으로는 해당

퀘스트를 수행하는데 걸리는 수행 시간을 기준으로 한다. 즉 해당 퀘스트를 수행하는데 걸리는 시간이 길면 길수록 퀘스트 난이도가 높다고 판단하고 이에 대한 보상 수준을 높이게 된다. 즉, 퀘스트 전체 수행시간 t_{total}로 정의한다면 이는 퀘스트 수행을 위한 이동시간 t_{moving}과 퀘스트 수행시간 t_{action}의 합으로 [식 1]과 같이 계산되게 된다.

$$t_{total} = t_{moving} + t_{action} \quad [식\ 1]$$

이동시간 t_{moving}은 퀘스트를 제공하는 NPC의 위치 p_i 와 이를 수행하기 위한 사냥터의 위치 f_i 간의 유클리디언 거리의 합으로, 보상 획득을 위한 왕복 이동을 전제로 하며 이를 전체 퀘스트 횟수 i 만큼 누적시켜 [식 2]과 같이 계산한다. 퀘스트 수행시간 t_{action}은 단위 수행시간 w_i와 퀘스트에서 요구하는 수행 횟수 n_i의 곱의 총합으로 [식 3]과 같이 계산된다.

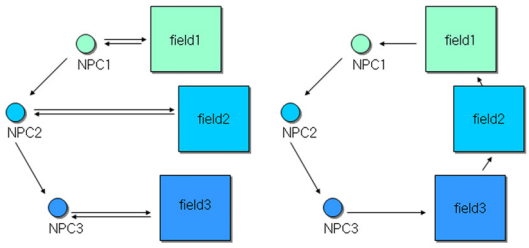
$$t_{moving} = 2\sum_{i=1}^{N} \left\| \mathbf{p}_i - \mathbf{f}_i \right\| \quad [식\ 2]$$

$$t_{action} = \sum_{i=1}^{N} \omega_i n_i \quad [식\ 3]$$

수집과 대화 퀘스트의 경우에는 t_{moving}값으로 전체 수행시간 예측이 이루어지게 되고, 사냥 및 수집 퀘스트의 경우에는 전체 퀘스트 수에 대한 t_{moving}과 t_{action}값의 합으로 시간 예측이 이루어지게 된다.

여기서 이동시간 t_{moving}은 복수 개의 퀘스트 수행 시에도 이론적으로 정확한 수치를 측정할 수 있을 것으로 보인다. 퀘스트를 제공하는 퀘스트 NPC들은 레벨 내 주요 플레이 거점 지역에 배치되고 이를 수행하기 위한 공간은 별도의 독립된 사냥터에서 이루어진다고 가정하였을 때, 각 퀘스트별로 이동하는 시간과 각 퀘스트 제공 NPC들 간의 이동거리의 합이 되기 때문이다. 일반적인 게임 실무에서는 이 부분의 계산을 플레이어가 순차적으로 퀘스트를 수행하는 것으로 가정하여 계산하게 된다. 즉 플레이어가 하나의 퀘스트 획득, 수행 그리고 보상 단계를 모두 마친 뒤 다음 퀘스트 수행을 진행할 것으로 예상하여 이동 거리 계산을 하게 된다.

그러나 실제로 게임 서비스 시에는 플레이어는 가까운 지역 내의 여러 개의 퀘스트를 한꺼번에 받은 다음 이를 한 번의 사냥터 이동을 통해 여러 개의 퀘스트를 한꺼번에 수행하는 경우가 많다. 수행 보상에 있어 오차가 발생하는 부분은 바로 이 부분으로, 실제 퀘스트 기획단계에서의 수치 산출 방식과 달리 플레이어가 빠른 퀘스트 수행을 위해 직관적으로 전체 최적 동선을 채택하려고 노력한다는 점에 있어 차이가 있기 때문이다. [그림 2]는 이러한 내용을 표현한 것으로 기획 단계에서 예상하는 1:1 대응 방식 동선은 좌측과 같고, 실제 플레이시 발생하는 플레이 동선은 우측과 같다. 정확한 퀘스트 보상 수식 산출을 위해서는 플레이어가 선택하는 이러한 최적동선을 예측하는 것이 필요하다. 이러한 고정된 위치에 대한 최적동선 예측은 바로 순회 판매원 문제로 알려진 NP-Hard문제로서 실제 게임 디자이너가 수작업을 통해 최단 경로를 도출하기가 매우 어려운 부분이다.



[그림 2] 기획 단계에서의 가정 경로와 실제 플레이 경로

4. 관련 연구

순회 판매원 문제 (TSP)[5]는 N개의 도시를 모두 연결하는 최단의 경로를 찾아내는 문제로, 계산 시간이 도시의 숫자 N의 지수 함수로 증가하는 NP-Hard 문제 중 하나이다. 순회 판매 문제를 해결하는 방법으로는 신경망(Neural Network), 담금질(Simulated Annealing), 그리고 유전자 알고리즘(Genetic Algorithm)등과 같은 기법들이 존재한다.

이중 유전자 알고리즘은 자연의 진화과정과 유전법칙에 영향을 받은 메타 휴리스틱 알고리즘이다. 유전자 알고리즘은 일반적으로 1) 개체의 모집단 정의, 2) 적합도(Fitness)에 따른 선택 과정, 3) 후손을 생성하기 위한 교배(Crossover), 4) 새로운 후손 발생을 유도하기 위한 돌연변이(Mutation) 단계를 거치며 최적의 해를 탐색한다. TSP는 유전자 알고리즘의 연구 초기 단계에서부터 적용이 되고 있는 대표적 분야로 지금까지 많은 해결책이 제시되어 왔다[6]. 이중 도시 간의 방문 순서 제약을 조건을 만족시키기 위한 유전자 알고리즘 기법은 Order Crossover(OX)[7], Cycle Crossover (CX)[7], Partially Matched Crossover(PMX)[8], Edge Recombination(ER)[9], Precedence 행렬[10], Union 연산자[11], 그리고 보로노이양자화 교차[12], 이 외에도 다양한 기법들[13]이 연구되고 있다.

본 논문에서는 퀘스트 보상 측정 부분에 있어 문제가 되는 퀘스트 수행 동선을 계산하기 위해 퀘스트 대상 요소들을 TSP로 대응시켜 해결하고자 하였다. 그 중에서도 퀘스트 시스템의 특성상 선행 관계가 존재하는 TSP 해를 찾기 위해 선행 관계 유지 교차연산자를 사용하는 유전자 알고리즘[14]을 사용하였다. 유전자 알고리즘은 항상 최적의 해를 보장할 수는 없지만 제한된 시간 내에 최선의 해 탐색을 보장한다. 본 논문에서는 이를 통해 최선의 해를 탐색한 다음 이들 수치를 퀘스트 보상 설정에 사용하는 방법을 퀘스트 보상 설계 단계에서 제안한다.

5. TSP 기반의 퀘스트 시스템 모델링

MMORPG 개발에 있어 많은 종류의 퀘스트 시스템이 존재한다. 본 연구의 결과가 이들에게 적용될 수 있는 범용성을 지닐 수 있도록 하기 위해 퀘스트 시스템을 사전에 정의할 필요가 있다. 본 논문에서 설정한 퀘스트 시스템은 아래와 같다.

1. 퀘스트는 고정된 자리에 배치되어 있는 퀘스트 NPC와 대화를 통해 획득한다.
2. 하나의 퀘스트 NPC는 한 개의 퀘스트를 제공한다.
3. 플레이어는 여러 개의 퀘스트를 한꺼번에 받아 그 리스트를 유지할 수 있다.
4. 플레이어가 받은 퀘스트 리스트를 유지하고 있을 때 관련 퀘스트를 수행할 수 있다.
5. 퀘스트들은 각각의 수행 사냥터가 1:1로 대응된다.
6. 퀘스트 수행 사냥터에는 사냥터에 필요로 하는 대상 몬스터가 충분히 많아서 사냥 대기 시간이 존재하지 않는다.
7. 퀘스트 보상은 퀘스트 수행 이후 원래의 퀘스트 제공 NPC에게 돌아가야 받을 수 있다.

본 논문에서 찾고자 하는 최단 거리 동선 비용을 산출하기 위해서는 기존의 비방향성 그래프를 기반으로 한 TSP 해를 찾음과 동시에 MMORPG 게임 특성상 다음과 같은 2개의 제약 조건을 만족시켜야 한다.

5.1 순서 제약 조건

퀘스트 플레이 동선상 획득-수행-보상의 3단계가 각각 순차적으로 이루어져야 한다. 후행 단계가 선행 단계보다 먼저 이루어질 수 없다. 이는 본 문제가 방문 순서 제약 조건이 있는 TSP 문제로 해결되어야 함을 의미한다.

5.2 이동 제약 조건

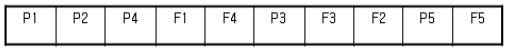
퀘스트 플레이 동선상 게임 내 레벨 지형 특성에 따라 이동 동선이 제한되는 경우가 발생한다. 지형으로 막혀 있는 거리를 플레이어가 통과할 수 없다. 이동 제한 조건이 있는 TSP 문제로 해결되어야 함을 의미한다.

6. 유전자 알고리즘의 적용

본 논문에서는 순서 제약 조건과 이동 제약 조건이 존재하는 TSP 문제 해결을 위한 알고리즘이 필요하다. 본 논문에서는 TSP 문제 해결을 위해 오랜 기간 그 성능이 검증된 유전자 알고리즘을 사용하였다. 본 논문이 사용한 세부 유전자 알고리즘 기법은 아래와 같다.

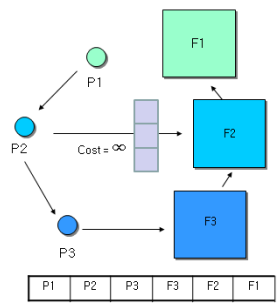
6.1 염색체 구성

본 논문에서 TSP 문제 해결을 위해 위치기반 형태의 염색체를 사용하였다. 염색체 상에서 유전자는 왼쪽에서 오른쪽으로 순차적으로 방문해야 하는 장소를 의미한다. 유전자의 종류는 퀘스트를 제공하는 NPC를 상징하는 p_i 유전자와 해당 퀘스트를 수행할 수 있는 사냥터를 상징하는 f_i 유전자 2종류로 구분하였다. 각각의 유전자는 해당 이벤트가 발생하는 위치 정보를 가지고 있다. 퀘스트 플레이 동선상 획득은 항상 수행보다 먼저 일어나야 하므로, 퀘스트 NPC의 위치 p_i 유전자는 사냥터 위치인 f_i 유전자보다 왼쪽에 배열되어야 획득-수행 조건을 만족시킬 수 있게 된다. p_i-f_i 간의 선행-후행 정보는 유저에 의해 직접 입력되게 된다. [그림 3]은 본 논문에서 사용한 염색체 구조로 p_i 유전자는 대응되는 f_i 유전자보다 선행하여 위치해 있음을 보이고 있다.



[그림 3] 염색체 구조

본 시스템은 사용자로부터 NPC 입력과 사냥터 입력을 직접 받을 수 있도록 구성되어 있다. 입력받은 NPC와 사냥터의 수에 맞도록 전체 염색체의 길이가 결정된다. 유전자들은 생성 단계에서 순서 제약 조건을 만족시킬 수 있도록 하였다. 사용자로부터 NPC 및 사냥터를 위치를 입력받을 때, 게임 리소스 중 해당 맵의 지형 특징을 가지고 있는 격자맵을 참조하여 이동 불가 지역을 검색하였고 이를 통해 이동 제약조건을 정보를 담고 있는 이동 불가능 경로 리스트를 생성하였다. 염색체 생성시 이 리스트를 참조하여 해당 염색체 구간의 비용에 무한값을 배분함으로써 이동 제약조건 만족을 거리 계산 단계에서 만족시킬 수 있도록 하였다. [그림 4]는 퀘스트 동선과 이에 대응되는 염색체의 구조를 보여주고 있다.



[그림 4] 퀘스트 플레이 동선과 대응 염색체

6.2 적합도 측정

유전자의 적합도를 측정하는 기준은 염색체 내의 조합된 유전자들 간의 거리의 합이다. 본 논문에서는 퀘스트 획득과 수행을 위해 조합된 N개의 유전자 g_i 들을 순서대로 방문하였을 때 길이의 합과 보상 이동을 위한 전체 거리의 합 d_{reward} 의 합, 즉 [식 4]의 값이 최소가 되는 염색체를 선택하도록 하였다. 보상 이동을 위한 전체 거리는

획득-수행 간의 최적화된 거리가 계산된 이후 마지막 f_i 와 모든 p_i 간의 TSP 문제가 된다. 이는 대상이 되는 위치들만 선별하여 선행 관계가 존재하지 않는 일반적인 유전자 알고리즘 적용을 통해 최단 거리를 계산해 낼 수 있다. 획득-수행 간의 최적화된 동선 산출을 위한 염색체 선택 방식은 순위 기반 선택 방법을 사용했으며 최상위 2개의 염색체가 교배를 위해 선택되도록 하였다.

$$f_{fitness} = \sum_{i=1}^{N-1} \| g_{i+1} - g_i \| + d_{reward} \quad [식\ 4]$$

교배는 2개의 개체 간에 염색체를 부분적으로 서로 바꿈으로써 새로운 개체를 생성하는 단계이다. 본 시스템에서는 선행 조건을 만족시킬 수 있도록 한 선행관계 유지 교차연산자 (Precedence-Preserving Crossover: PPX)[14]를 사용하여 교배가 이루어지도록 하였다. 이는 부모염색체가 이미 선행관계를 만족하고 있을 때 사용할 수 있으며 다음과 같은 방법으로 새로운 염색체를 생성한다. 새로 생성된 염색체는 기존의 선행 관계를 유지하게 된다. [그림 5]는 선행관계 유지 교차연산자가 적용된 염색체의 예이다[14].

부모1: (1 2 3 | 4 5 6 | 7 8)
부모2: (2 **4 6** | 8 7 **5** | 3 1)

자손1: (1 2 3 **4 6 5** 7 8)
자손2: (2 4 6 5 7 8 3 1)

[그림 5] 교차연산자의 적용 사례

선행관계 유지 교차연산자는 다음과 같은 방법으로 새로운 염색체를 생성한다. (1) 두 개의 부모 염색체 중 한 개의 염색체 1에 대해서 두 개의 절단 위치를 랜덤으로 선택한다. (2) 부모 염색체 1의 첫 번째 절단 위치 왼쪽 부분과 절단 위치의 오른쪽 부분을 자손 염색체 1의 같은 위치에 복사한다. (3) 부모 염색체 1의 절단 영역 내에 있는 유전자들은 부모 염색체에 2에 나타난 순서대로 다시 정렬하여 자손 염색체 가운데에 복사한다. 자손 염색체 2를 생성할 때도 같은 방법을 적용한다.

7. 최적동선을 고려한 퀘스트 보상 설계

TSP 해결을 통한 생성된 퀘스트 플레이 동선은 유전자 알고리즘에 의해 제한된 시간 내에 최적화된 동선이다. 생성된 동선 길이는 다수의 퀘스트들 수행에 대한 누적 보상으로서 이를 퀘스트 각각에 대한 보상값으로 분배해야 한다. 이를 위해 본 논문에서는 [식 5]와 같은 보상 수식을 제안한다. 수식에서 R_k는 k번째 퀘스트 보상값, R_d은 해당 퀘스트에 대한 기본 보상값, $d_{optimal}$은 해당 퀘스트를 포함하고 있는 전체 최적화된 동선 길이 그리고 d_{direct}은 NPC(p)와 사냥터(f)간의 직선 왕복 거리이다. 결과적으로 $d_{optimal}/d_{direct}$의 비율은 기존의 보상에서 적용되는 최적화된 동선을 고려한 보정률 σ이 된다.

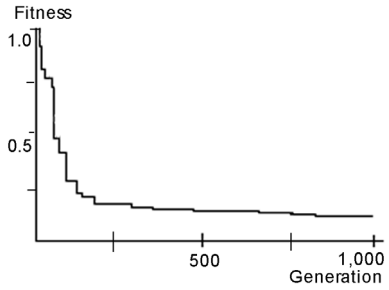
$$R_k = \sigma R_d \quad (\sigma = d_{optimal} / d_{direct}) \quad [식\ 5]$$

본 논문에서 제안된 최적화된 동선을 플레이어가 선택할 가능성은 플레이어가 퀘스트 시스템에 대한 기본적인 경험이 있고 퀘스트 제공 NPC들과 사냥터의 위치가 공개되어 있다고 가정하였을 때, 퀘스트 제공 NPC들과 사냥터의 밀집도와 연관된다. 즉 해당 레벨에서 퀘스트 제공 NPC들이 특정 지역(거점 또는 도시)에 몰려 있고 대상이 되는 사냥터의 밀집도가 높을수록 본 논문에서 제안한 동선을 따를 확률이 높으며 이는 퀘스트 NPC 위치들의 분산값 기준으로 예측할 수 있다.

8. 실험 결과

본 시스템은 Windows XP 운영 체제, Intel

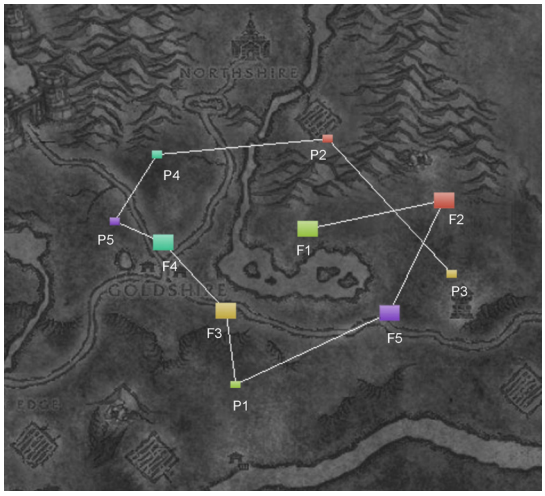
Core2 Duo 1.83GHz, 1GB 메모리의 환경에서 C++와 DirectX 9.0을 사용하여 구현되었다. 이를 위해 본 논문에서 사용한 기준 파라미터 값들은 최대 세대수는 1,000회, 염색체의 길이는 10, 집단의 크기는 4로 설정하였고, 최적해는 고정된 세대 동안 생성된 조합 중 최단 거리 값을 가지고 있는 염색체를 선택함으로써 이루어지도록 하였다. [그림 6]은 본 논문에서 사용된 유전자 알고리즘의 세대별 수렴 경과를 보여준다.



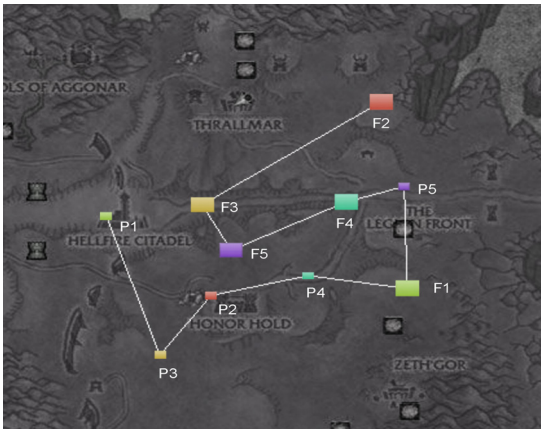
[그림 6] 세대별 최적해 수렴 과정

[그림 7]부터 [그림 9]는 본 논문의 구현 결과로써 실제 MMORPG 게임의 레벨에 적용해 보았다. 작은 사각형은 NPC의 위치를 나타내고 있고 같은 색깔의 큰 사각형은 이에 대응하는 사냥터의 위치를 나타낸다. 플레이어가 5개의 퀘스트를 획득하여 이를 수행하였을 때 발생할 수 있는 동선을 시뮬레이션 해 본 결과이다. 결과를 살펴보았을 때 본 시스템에서 생성된 동선은 유저가 퀘스트 획득 NPC들을 우선적으로 방문하여 해당 레벨에서 수행할 수 있는 관련 퀘스트들을 일괄적으로 취합한 뒤 대응하는 사냥터를 방문하는 동선을 생성하게 된다. 본 논문의 결과에 의해 생성된 퀘스트 동선의 거리와 기존의 1:1 대응 방식으로 계산된 퀘스트 동선 거리와의 차이와 이에 따른 보정률의 값은 [표 1]과 같다.

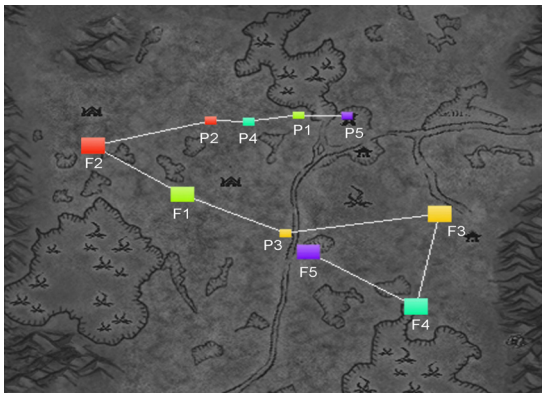
결과를 분석해보면 퀘스트 제공 NPC들이 좀더 가까이 모여 있을 경우 기존 대비 전체 이동 거리가 더욱 단축되는 것을 알 수 있으며 이는 자연스럽게 더 낮은 값의 보정률을 발생시키는 것을 알 수 있다. 즉 밀집도가 높은 퀘스트일 경우 본 논문에서 산출한 보정률을 적용하지 않았을 때 최악의 경우 41% 수준의 과보상이 발생할 수 있음을 보이고 있다. 본 논문에 의해 제안된 퀘스트 보상 수치로 계산된 값은 기본 퀘스트 보상값에 대한 주요 반영 수치로 사용될 수 있으며 퀘스트 제작자가 그 반영 여부를 NPC 밀집도의 수준을 고려하여 결정하게 된다.



[그림 7] Case 1 (NPC 밀집도: 하)



[그림 8] Case 2 (NPC 밀집도: 중)



[그림 9] Case 3 (NPC 밀집도: 상)

[표 1] 구현 결과 비교 (단위: 픽셀)

거리 계산 / 결과	본 논문의 기법에 의해 산출된 거리	1:1 대응 모델에 의한 거리	보정률
Case1	1,726	2,251	76.7
Case2	2,490	3,964	62.8
Case3	3,763	6,359	59.2

9. 결 론

퀘스트 보상 기획 업무는 그 동안 기획자가 수작업을 통해 단순한 거리 산출 모델로 계산되었다. 본 논문에서는 기존의 수작업에서 벗어나 유전자 알고리즘 도입을 통한 TSP문제 해결 방법을 퀘스트 동선 설계에 도입하였고, 그 결과는 퀘스트 동선 설계시 최선의 케이스로 구분하여 사용될 수 있는 값을 제공할 수 있음을 보였다.

본 논문에서 제안하는 퀘스트 보상 기법은 바로 이러한 최적화된 퀘스트 동선을 기반으로 한 것으로 NPC들의 밀집도가 높을수록 그 적용 효과가 크고 과보상을 방지하는데 효과가 있다. 또한 생성된 최적화된 동선은 퀘스트 제작자로 하여금 사전에 퀘스트 동선 흐름을 파악할 수 있도록 도와줌으로써 기획 단계에서 퀘스트 구성 전반적으로 좀 더 정확한 예측을 할 수 있도록 하는데 기여할 수 있다.

본 논문에서 제안한 퀘스트 보상 기법 및 시스템을 실제 제작 프로세스에 적용 시 기존의 수작업으로 하였을 경우 발생하였던 장기간의 실제 플레이 테스트 시간을 사전에 단축시킬 수 있고 최적 동선에 기반한 퀘스트 동선 흐름을 사전에 파악할 수 있도록 하여 기획 단계에서 좀 더 정확한 동선 추정 및 퀘스트 보상 수치를 산출하는데 도움이 될 수 있을 것으로 기대한다.

참고 문헌

[1] A. Rollings and D. Morris, “Game Architecture and Design”, Coriolis, 2000.
[2] 송현주, 이대웅, “스크립트 DB를 이용한 MMORPG의 게임 시나리오 개발”, 한국게임학회논문지, 제6권, 제4호, pp. 89-95, 2006.
[3] Jeff Howard, “Quests: Design, Theory, and History in Games and Narratives”, A. K. Peters, 2008.
[4] Blizzard, World of Warcraft, 2004.
[5] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnnooy Kanand, and D. B. Shmoys, “The Travel Salesman Problems,” Wiley, 1985.
[6] 문병로, 쉽게 배우는 유전 알고리즘, 한빛미디어, 2008.
[7] J. M. Oliver, D. J. Smith and J. R. C. Holland, “A Study of Permutation Crossover Operators on the Travel Salesman Problems”, Genetic Algorithms and Their Applications: Proceeding of the Second ICGA, pp. 224-230, 1987.
[8] D. E. Goldberg and R. Lings, “Alleles, Loci and the Travel Salesman Problems”, Proceeding of an International Conference on GA and Their Applications, pp. 154-162, 1985.
[9] D. Whitley, T. Starweather and D. Fuqua, “Scheduling Problems and the Traveling Salesman: The Genetic Edge Recombination Operator”, Proceeding of the third ICGA, pp. 133-141, 1989.
[10] G. J. E. Rawlins (ed.), Foundation of Genetic

Algorithms, Morgan Kaufmann, 1991.
[11] 이강구, 한승기, “TSP에서 Union 연산자를 이용한 유전자 알고리즘”, 한국정보과학회 학술발표논문집, 제21권, 제1호, pp. 213-216, 1994.
[12] 서동일, 문병로, “서열순서화문제를 위한 상위정보를 이용하는 혼합형 유전알고리즘”, 퍼지 및 인공지능시스템학회 논문지, 제15권, 제6호, pp. 661-667, 2005.
[13] 이혜리, 이건명, “Sequential ordering problem과 job shop scheduling problem에 적용 가능한 선행관계유지 유전 연산자의 비교”, Journal of the Research Institute for Computer and Communications, 제7권, 제2호, pp. 563-570, 1999.
[14] 이경미, 이건명, “방문순서 제약이 있는 순회세일즈맨 문제를 위한 유전자 알고리즘”, 정보과학회 논문지(A), 제25권, 제4호, pp. 362-368, 1998.

강신진(Kang, Shin Jin)

2003년 고려대학교 정보통신대학 컴퓨터학과 이학석사
2009년 고려대학교 정보통신대학 컴퓨터학과 박사수료

관심분야 : 게임 기획, 컴퓨터 그래픽스

신승호(Shin, Seung Ho)

2007년 고려대학교 정보통신대학 컴퓨터학과 이학석사
2009년 고려대학교 정보통신대학 컴퓨터전파통신학과 박사수료

관심분야 : 게임 프로그래밍, 컴퓨터 그래픽스, 가상현실

조성현(Cho, Sung Hyun)

1978년 서울대학교 계산통계학과 이학사
1980년 서울대학교 계산통계학과 이학석사
1995년 UCLA 컴퓨터과학과 이학박사
1996년~현재 홍익대학교 게임학부 교수

관심분야 : 게임 프로그래밍, 게임 그래픽스, 게임 물리, 분산 시스템