Double Vertex 그래프에 의한 궤도회로 토폴로지의 생성

황 종 규*, 이 종 우*, 정 의 진*, 김 태 진°
한국철도기술연구원*, 후지쯔°

Track Circuit Topology Design by Double Vertex Graph Algorithm

Jong-Gyu Hwang*, Jong-Woo Lee*, Eui-Jin Joung*, Tae-Sin Kim°
Korea Railroad Research Institute*, Fujitsu°

Abstract - A representation technique of a given track topology is required by many software applications in railway technology such as signalling system simulator. To achieve these, the concept of double vertex graph architecture is proposed. These are composed of pairs of vertices and node between the single vertices. Double vertex graph architecture can be understood as a extension of classical graphs. In developed railway signalling simulation software, it is shown that track topology can be represented by proposed algorithm in a efficient way. Especially it makes sure that these are suitable technique for representing and implementing of switch, routes which can be introduced some mistake in classical graph algorithm

1. 서 론

철도신호제어 시스템은 열차운행의 안전성을 보장하는 보안시스템으로 보장하는 최종적인 시스템으로 신호시스템의 설계 및 제작에 있어서 많은 노력들이 필요로 하게 된다.

이의 일환으로 신호시스템의 시뮬레이션을 위해서는 필수적으로 선로 프로파일을 소프트웨어 상으로 구현하여야 하며, 이때 실지 궤도회로처럼 이 선로 선형에서 진로 등의 정보를 가지고 신호시스템의 각종 로직에 따라서 열차의 운행이 시뮬레이션 될 수 있어야 한다 [3]-[5]. 즉, 물리적인 선로선형을 소프트웨어로 표현하여 적절한 시뮬레이션이 될 수 있도록 하여야 한다.

이러한 궤도회로의 모델링은 단순한 선로의 모델링 작업이 아니라 이들에게 진로정보 및 열차의 진행 정보 등이 포함되어야 한다. 본 논문에서는 이를 위해 일반 그래프 이론을 확장한 Double Vertex 그래프 알고리즘을 이용한 철도 선로선형의 모델링 방법을 연구하였다 [1][2]. 이 방법은 선로선형을 효과적으로 나타낼 뿐만 아니라 신호시뮬레이션에 필요한 진로설정 및 진로검색, 열차 주행 등이 효과적으로 구현될 수 있다.

2. 선로 선형 모델링

2.1 궤도선형 모델링

궤도선형은 철도의 선로 특성을 나타내는 부분이다. 이 궤도선형에는 궤도회로(Track Circuit), 분기기(Switch) 등 다양한 구성요소들이 존재한다. 또한 시뮬레이션을 위해서는 이러한 각각의 선형들이 서로 연결되어 전체적인 선로 선형을 이루어야 한다.

철도의 궤도회로 모델링에 있어서 대부분 분기기가 있어 이에 따라 진로가 달라지므로 이에 대한 고려가 반드시 있어야 한다. 일반적인 방법에 의해서는 이러한 분기기와 진로의 개념을 모델링 하기가 어렵다.

그림 1 (a)는 분기기가 포함된 전형적이면서 간단한 궤도회로 선형을 나타낸 것이다. 이 선형에서는 다음과 같은 진로들이 가능하다.

- X → A → B → Z
- X → A → C → Z
- Z → C → A → X
- Z → B → A → X

이러한 진로들의 구성은 가능하지만, 분기기의 특성 상 'Z → B → A → C → Z' 같은 진로는 구성이 되어서는 안된다.

(a)와 같은 실지의 선로 선형은 (b)와 같이 간단하게 모델링 되어질 수 있다. 이러한 모델링에서도 (a)와 마찬가지로 실지로 존재할 수 있는 모든 진로들을 나타낼 수 있지만 (a)와 마찬가지로 분기기의 특성을 나타낼 수는 없다. 즉, 실지로 경로가 구성되어서는 안되는 'Z → B → A → C → Z' 같은 경로도 (b)와 같은 모델에서는 성립되어질 수 있기 때문이다.

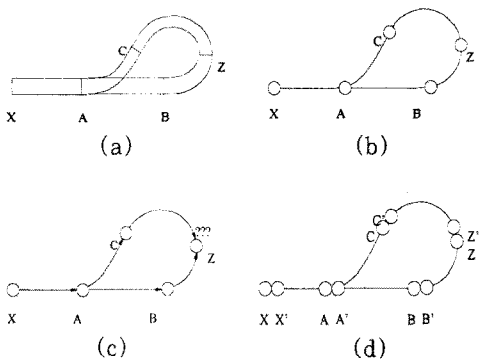


그림1. 궤도회로의 모델링

따라서 이러한 그림 (b)와 같은 모델에서 그림 1 (c)와 같은 방향성을 갖는 모델링을 생각할 수 있다(Directed Graph). 즉, 한 vertex에서 다른 vertex로의 경로가 항상 순방향 또는 역방향으로 표시될 수 있도록 함으로써 열차의 진로를 표시할 수 있도록 하는 방법이다. 이와 같이 vertex와 vertex 사이의 노드에 방향성을 줌으로서 분기기의 특성을 일부는 나타낼 수 있지만 또 다른 문제가 생기게 된다. 즉, 분기기의 특성을 나타내기 위해 그림 (c)와 같은 방향성을 갖는 노드로 표시함으로서 'C → A → B' 같은 경로는 금지되도록 표시될 수 있지만 'C → Z → B'와 같이 실지로는 표시되어야 하는 경로가 표시되어지지 않는 경우가 발생할 수 있다. 즉 vertex Z에서는 'B → Z'와 'C → Z'가의 서로 반대방향으로 인하여 'C → Z → B'와 같은 경로가 존재할 수 없게 된다.

따라서 철도신호시스템의 궤도회로 선형 모델링 특히 분기기가 포함된 경우는 새로운 방법에 의한 모델링이 필요하다.

2.2 Double Vertex Graph 개요[1][2]

그림 1의 (a), (b) 및 (c)의 궤도회로 선형 모델링은 분기기가 포함된 진로를 나타내기에는 적절치 못한 방법들이다. 본 연구에서는 이러한 분기기의 특성에 따른 경로의 표현이 가능하도록 궤도회로 선형을 그림 (1) (d)와 같이 모델링 하는 방법을 검토하였다.

기존의 그래프 이론에서는 각각 하나의 vertex와 하나의 노드로 구성되어져 있으나 이러한 그래프 이론을 바탕으로 하여 궤도회로의 선형을 모델링 할 경우 분기기 등에 대한 정확한 표현이 불가능하여 철도신호시스템의 각종 기능들의 시뮬레이션이 어렵게 된다. 따라서 본 연구에서는 하나의 노드에 두 개의 vertex를 가진 'Double Vertex Graph' 알고리즘을 사용하여 궤도회로 선형에 대한 모델링을 하였다.

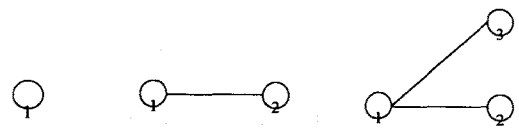
따라서 경로를 찾을 경우는 'vertex - vertex - node - vertex - vertex - node - vertex -....' 와 같은 조합에 의해 가능해 질 수 있다. 이는 그림 1의 (d)와 같이 표현될 수 있다. 이러한 이중 vertex 그래프에 의한 궤도회로의 표현에 있어서 경로는 다음과 같이 나타날 수 있다.

- XX′ → AA′ → BB′ →Z
- XX′ → AA′ → CC′ → Z′
- Z′Z → B′B → A′A → X′
- ZZ′ → C′C → A′A →X′
- CC′ → Z′Z → B′
- BB′ → ZZ′ → C′

여기에서 'B′B →A′ →C′와 같은 경로는 존재하지 않게 된다. 따라서 이러한 이중 vertex들과 노드들을 서로 링크시킴으로서 경로들을 표현할 수 있고, 또한 분기기의 특성을 적절히 나타낼 수 있다.

이러한 그래프 구조를 바탕으로 철도의 궤도회로 선형을 모델링 할 경우, 철도 궤도회로의 기본 요소들은 그림 2와 같이 크게 세 가지로 분류할 수 있다. 이들 기본 구성요소들을 바탕으로 실제 선로에 존재하는 여러 가지 다양한 형태의 궤도회로 선형이 구성되어질 수 있다.

물론 신호시스템의 시뮬레이션을 위해서는 이러한 궤도회로 선형을 이루는 구성요소 이외에, 신호기도 필요하지만 본 논문에서는 이 신호기 등 다른 신호장치 요소들이 필요하지만 본 논문에서는 이에 대한 설명을 생략한다.



(a) End (b) Track Circuit (c) Switch

그림 2. 궤도회로 선형의 구성요소

이러한 궤도회로 선형의 기본요소들을 이용해서 실제 선로들을 모델링 하게되며, 이들 기본 요소들이 서로서로 링크하여 열차가 이 선을에 따라 각종 신호로직에 의해 운행되는 시뮬레이션을 위한 기본 베이스가 구축되어질 수 있다. 이러한 이중 vertex 그래프 방법을 기반으로 그림 2와 같은 기본 구성요소를 바탕으로 선로선형을 구성한 예를 그림 3에 나타내었다.

그림 3의 선로선형 예를 앞의 기본 구성요소를 가지고 표현하면 다음과 같다.

P = {1, 2, 3, 4, ..., 30}
R = {2↔3, 4↔5, 6↔7, 6↔8, 9↔10, 11↔12, 15↔14, 13↔14, 26↔17, 20↔21 22↔23, 24↔25, 26↔27, 28↔29 }
E = { {1}, {2, 3}, {4, 5}, {6, 7, 8}, {9, 10}, {11, 12}, {13, 14, 15}, {16, 17}, {18}, {19}, {20, 21}, {22, 23}, {24, 25}, {26, 27}, {28, 29}, {30} }

where *P : The set of vertices*
R : The node relation
E : The equivalence classes of the track element

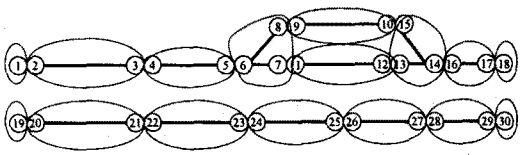


그림 3. 선로 선형의 예

실제로 소프트웨어로 구현 시에는 이러한 이중 vertex 그래프 구조를 바탕으로 한 변형된 방법을 사용하여 구현하였다.

3. 궤도회로 선형 편집 소프트웨어[4][5]

철도 신호시스템의 시뮬레이션을 수행하기 위해서는 소프트웨어로서 우선적으로 선로의 선형에 대한 모델링이 이루어져야 한다. 이러한 궤도회로의 모델링을 위해서 본 연구에서는 앞에서 설명한 이중 vertex 그래프 구조를 이용하였다.

개발한 소프트웨어에서는 궤도회로를 구성하는 가장 기본 단위인 레일을 클래스로 구성하였다. 이들 레일들이 몇 개가 묶여져서 하나의 궤도회로 클래스가 되게 된다.

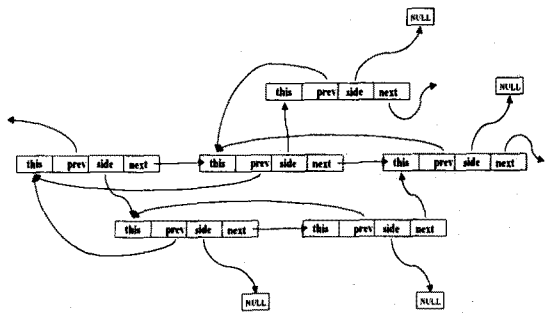


그림 4. 궤도회로 연결 원리

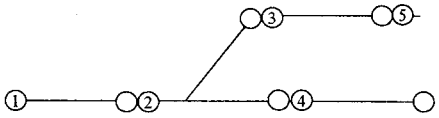
각 회로의 형식에 따라 달라지지만 기본적으로 오른쪽에 위치한 회로는 'next' 포인터로 연결되고, 왼쪽은 'prev' 포인터로 연결되고 측면에 위치하게 되면 'side' 포인터에 연결되게 된다. 이처럼 연결할 레일의 정보들을 포인터 정보로 가짐으로서 서로 링크되게 되는데, 이러한 과정이 편집된 모든 레일 인스턴스에서 수행됨으로서 전체적으로 선로가 링크되게 되고, 이로서 열차가 이 선로 위를 따라 주행 시뮬레이션을 할 수 있는 바탕이 되게 된다. 이렇게 연결된 레일 인스턴스들은 몇 개씩 하나의 궤도회로 클래스가 묶기게 된다.

이렇게 완벽하게 레일 인스턴스 서로간에 링크가 마무리되면 그 다음 그림 2의 (c)와 같은 분기기에서 열차의 주행 진로를 찾을 수 있도록 하여야 한다. 이때에 앞에서 설명한 이중 vertex 그래프 구조가 사용되어지게 된다.

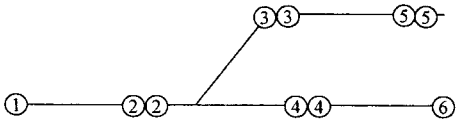
한 궤도회로에 소속된 회로의 포인터는 각 궤도회로의 맨 앞에 위치하는 회로가 Vertex_rail[0], 맨 뒤에 위치하는 회로가 Vertex_rail[1] 그리고 측면 회로가 존재하는 경우 Vertex_rail[2]에 저장된다. 이렇게 함으로서 Vertex_rail[ ] 의 배열에 들어 있는 포인터에 의해 시뮬레이션이 될 수 있도록 하였다. 시뮬레이션 시 마커는 Vertex_rail[ ]의 배열

에 들어 있는 포인터와 상호 연결되어져 있기 때문에 열차의 진입이 가능한지, 다음 궤도회로로 진입이 가능한지 등을 확인할 수 있다. 시뮬레이션 시 각 궤도회로의 내부에서는 레일의 포인터로 열차가 진행하지만 다음 궤도회로로 넘어갈 때는 Vertex 구조에 의해 정해진 다음 궤도회로로 진행하게 된다. 이렇게 해서 다음 궤도회로로 연결되어지면 또 다시 회로의 포인터로 연결되는 과정을 반복 수행한다. 즉 이러한 알고리즘에 의해 시뮬레이션 시 열차의 진행이 이루어지도록 하였다.

궤도회로가 연결되는 구조를 살펴보면 그림 5 (a)의 경우처럼 별개의 궤도회로로 구성되어 있다가 하나의 선로 프로파일로 링크시키면 그림 5 (b)와 같이 각각 궤도회로의 맨 앞과 맨 뒤, 그리고 측면 회로의 포인터와 연결되는 점들은 같은 번호를 가지게 된다. 이렇게 하여 임의로 편집된 선로 프로파일이 열차가 달릴 수 있는 실제 선로가 되게 된다.



(a) 연결되기 전 상태



(b) 연결된 후의 상태

그림 5. Vertex 자료구조에 의한 궤도회로 연결

본 연구에서는 앞에 설명한 알고리즘을 검증하기 위하여 그림 6과 같은 원형의 선로 프로파일을 편집하여 테스트하였다. 실지로 이러한 선로 프로파일은 존재하지는 않지만, 사용한 알고리즘의 타당성을 검증하고자 이러한 프로파일을 대상으로 선정하였다. 기존의 철도 시뮬레이션을 위한 대부분의 프로그램들은 선로들의 연결을 윈도우의 좌표 정보를 이용하여 링크시키고 있다. 따라서 이러한 방법을 사용할 경우 그림 6과 같은 원형 프로파일의 경우 구현이 불가능할 것이다.

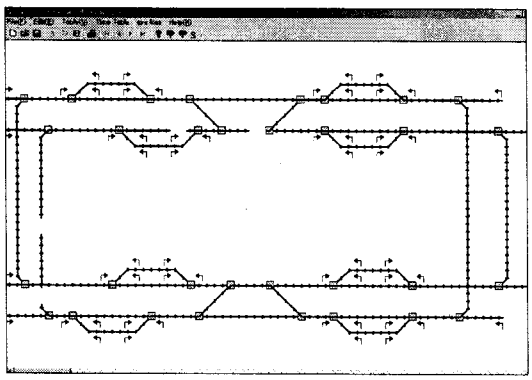


그림 6. 제안된 알고리즘의 테스트용 선로 프로파일

이와 같은 원형 선로 프로파일의 경우 윈도우의 좌표 정보에 의해서는 궤도회로들을 링크시킬 수 없고, 본 논문에서 사용한 알고리즘으로는 가능하다. 이러한 원형 프로파일에서의 주행 시뮬레이션 결과 정상적으로 수행됨을 확인할 수 있었다. 그림에서 보면 수평의 선로뿐만 아니라 수직의 선로도 정상적으로 열차가 주행하고 있음을 보여주고 잇다.

이러한 결과는 좌표정보에 의해 궤도회로들을 링크시키지 않고, 이중 vertex 그래프 구조, 연결될 정보를 Vertex[ ] 배열의 정보를 이용하여 링크시킴으로서 가능하게 되었다. 또한 분기기의 특성도 잘 나타낼 수 있다. 그림 7은 개발된 소프트웨어를 이용하여 궤도회로를 편집하는 화면 일부를 캡쳐한 것이다.

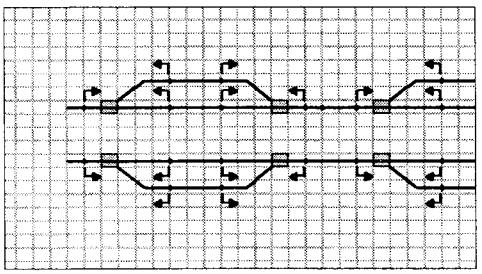


그림 7. 궤도회로 편집 프로그램

4. 결론

본 논문에서는 철도 신호시스템의 시뮬레이션을 위한 소프트웨어에 있어서, 철도만이 갖는 특성을 가지면서 선로의 선형을 표시할 수 있는 이중 vertex 그래프 구조에 대해 설명하였다. 이 알고리즘의 적용으로 철도 선로의 특성 중의 하나인 분기기의 특성을 나타낼 수 있고 또한 열차의 안전에 중요한 영향을 미치고, 시뮬레이션 시 가장 중요한 열차의 주행 진로의 표현 등도 이 알고리즘의 적용에 따라 쉽게 구현될 수 있다.

[참 고 문 헌]

[1] M. Montigel, 'Formal Representation of Track Topologies by Double Vertex Graphs', Proceedings of Computers in Railways, pp. 359-370, 1998.

[2] S. Axler, et al., 'Modern Graph Theory', Springer, 1998.

[3] http://www.azurenet.com/

[4] G7사업 1단계 연구보고서, '전기신호시스템 엔지니어링 기술개발', 한국철도기술연구원, 1999.

[5] 황종규 외, '신호제어시스템 시뮬레이터용 소프트웨어 설계', 한국철도학회 춘계학술대회, pp. 269-275. 2000.