DOI: 10.3745/KIPSTB.2011.18B.3.157

유전자 알고리즘을 이용한 확장성 있고 빠른 경로 재탐색 알고리즘

이 정 규[†] · 김 선 호[††] · 양 지 훈[†††]

요 약

본 논문은 유전자 알고리즘을 이용해서 동적으로 변하는 네트워크상에서 빠르게 최단 경로를 재탐색할 수 있는 알고리즘을 제안한다. 제안 알고리즘은 다익스트라 알고리즘과 유전자 알고리즘을 통합한 형식의 알고리즘이다. 이 제안 알고리즘은 최초 탐색 시 다익스트라(Dijkstra) 알고리즘을 이용해서 유전자 알고리즘의 초기화 과정을 용이하게 하는 선행자 배열을 정의한다. 그 후 유전자 알고리즘은 적절한 유전 연산자를 통해 동적으로 변하는 트래픽 상황에서 최적의 경로를 재탐색한다. 실험 결과를 통해 제안 알고리즘이 거대한 네트워크 데이터에 대해서 다른 유전자 알고리즘 기반의 최단경로 찾기 알고리즘이나 다익스트라 알고리즘보다 적은 계산시간으로 더 짧은 주행시간의 경로를 제시한다는 것을 보였다.

키워드 : 최단 경로 찾기 문제, 유전자 알고리즘, 다익스트라 알고리즘, 라우팅 문제

Fast and Scalable Path Re-routing Algorithm Using A Genetic Algorithm

Jungkyu Lee[†] · Seonho Kim[††] · Jihoon Yang[†††]

ABSTRACT

This paper presents a fast and scalable re-routing algorithm that adapts to dynamically changing networks. The proposed algorithm integrates Dijkstra's shortest path algorithm with the genetic algorithm. Dijkstra's algorithm is used to define the predecessor array that facilitates the initialization process of the genetic algorithm. After that, the genetic algorithm re-searches the optimal path through appropriate genetic operators under dynamic traffic situations. Experimental results demonstrate that the proposed algorithm produces routes with less traveling time and computational overhead than pure genetic algorithm-based approaches as well as the standard Dijkstra's algorithm for large-scale networks.

Keywords : Shortest Path Problem, Genetic Algorithm. Dijkstra Algorithm, Routing Problem

1. 서 론

자동차를 이용하는 현대인에게 내비게이션은 이제 생활필수품이다. 최근에는 네비게이터에 기본적인 최단 경로 찾기 기능뿐만 아니라, TPEG(Tranport Protocol Experts Group)과 같은 운전자들에게 실시간으로 교통정보를 제공하는 서비스가 많이 제공되고 있다[1]. 하지만 불행히도, 이러한 실시간 교통정보를 효율적으로 이용하는 최단 경로 알고리즘의 개발은 미비하다. 실시간 정보를 이용하는 내비게이터의 최단 경로 알고리즘은 다음과 같은 두 가지 특징을 지녀야 한다. 첫째, 실시간 교통 정보가 전송되는 내비게이터에서 사용하는 최단 경로 탐색 알고리즘은 경로의 재탐색이 빨라야 한다. 왜냐하면 실시간 교통 정보가 전송될 때마다 다시 새로운 경로를 찾아야 하기 때문이다. 둘째, 최단 경로 찾기 알고리즘이 전체 지도 데이터의 정보를 전송 받는 것이 아니라, 필요한 지도 데이터의 정보만을 요청, 전송 받아야 한다. 본 논문에서는 위의 두 가지 성질을 가진 최단 경로 찾기 알고리즘을 제안한다. 우리가 제안하는 최단 경로 찾기 알고리즘은 다익스트라 알고리즘으로 최초 탐색, 선행자 배열을 찾고, 동적으로 변화하는 교통 상황에 적응하기 위해 유전자 알고리즘을 이용하여 재탐색한다. 2장에서는 제안 알고리즘의 바탕이 되는 Ahn이 제안한 유전자 알고리즘을

※ 이 연구는 한국연구재단 일반연구자 지원사업의 지원에 의한 것임.
† 준 회 원 : (주)사이람 연구원
†† 정 회 원 : 서강대학교 컴퓨터공학과 BK 연구 교수
††† 종신회원 : 서강대학교 컴퓨터공학과 부교수
논문접수 : 2010년 6월 7일
수 정 일 : 1차 2010년 7월 13일, 2차 2010년 8월 27일, 3차 2010년 9월 15일
심사완료 : 2011년 2월 11일

이용한 최단 경로 라우팅 알고리즘에 대해 알아보고[2], 3장에서는 제안 알고리즘을 설명한다. 4장에서는 제안 알고리즘에 대한 실험 결과를 제시하고 5장에서 마무리 한다.

2. 관련 연구: 유전자 알고리즘 이용한 최단 경로 탐색

유전자 알고리즘(Genetic Algorithm, GA)은 생물학에서 영감을 얻은 전역 탐색 휴리스틱 알고리즘(Global Search Heuristic Algorithm) 중 하나로서 여러 가지 최적화 문제들에 적용되어 왔다[3].

Ahn은 이 유전자 알고리즘을 이용한 최단 경로 라우팅 알고리즘을 제안하였다[2]. 하지만 그의 알고리즘을 내비게이터 알고리즘으로 사용하기에는 한계가 있다. Ahn은 그의 논문에서 약 노드의 개수가 50개 정도인 그래프에 대해서 그의 알고리즘이 잘 동작한다는 실험 및 검증을 하였는데, 우리는 실험을 통해서 그의 알고리즘이 노드 1000개 이상의 그래프에 대해서는 잘 동작하지 않는다는 것을 알아냈다. 대한민국 지도 데이터의 노드의 개수가 약 90만개정도인 것을 감안할 때 내비게이터의 최단 경로 찾기 알고리즘으로 사용하기에는 확장성 문제가 있음을 알 수 있다.

Kanoh는 자동차 내비게이션의 최단 경로 찾기 문제에 대한 유전자 알고리즘 방식의 접근 방식을 제안하였다[4, 5]. Kanoh의 방식은 동적으로 변화는 도로 상황을 고려하여 다익스트라 알고리즘을 통해 개체군 초기화를 한다는 점은 우리의 방식과 비슷하다. 하지만, Kanoh의 방식과 우리의 방식은 작동 방식이 완전히 달라, 비교할 수가 없다. 그의 방식은 해의 질을 다양한(multi-objective) 기준(예를 들어, 주행시간, 경로 길이, 신호등 개수 등등)을 이용하여 발전시키기 위해 유전자 알고리즘을 사용하였고, 우리의 방식처럼 경로의 재탐색을 위해 유전자 알고리즘을 사용한 것이 아니다.

본 연구에서 한 일을 한마디로 요약하면, Ahn의 알고리즘에 확장성을 부여하여, 빠른 재탐색을 통해 동적으로 변하는 네트워크에 대해서 적응성을 갖도록 했다는 것이다. 제안 알고리즘을 설명하기 전, Ahn의 알고리즘을 개략적으로 소개한다.

2.1 유전자 표현(Genetic Representation)

Ahn이 제안한 최단 경로 찾기 문제에 대한 유전자 알고리즘의 염색체는 가변 길이를 갖으며, 라우팅 경로가 지나는 노드들의 ID를 나타내는 일련의 번호로써 구성되고, 첫 번째 유전자는 항상 출발지 노드를 위해 할당된다.

2.2 개체군 초기화(Population Initialization)

Ahn은 개체군 초기화를 위해 무작위적 초기화를 사용하였다. 즉 각 염색체의 노드에서 그 노드와 물리적 링크가 형성된 노드들 중에 무작위적(random)으로 노드를 선택해 부호화한다.

2.3 적합도 함수(Fitness Function)

Ahn은 적합도 함수는 수식 (1)과 같이 정의하였다.

$$f_i = \left(\sum_{j=1}^{m_i - 1} C(g_i(j), g_i(j+1)) \right)^{-1} \quad (1)$$

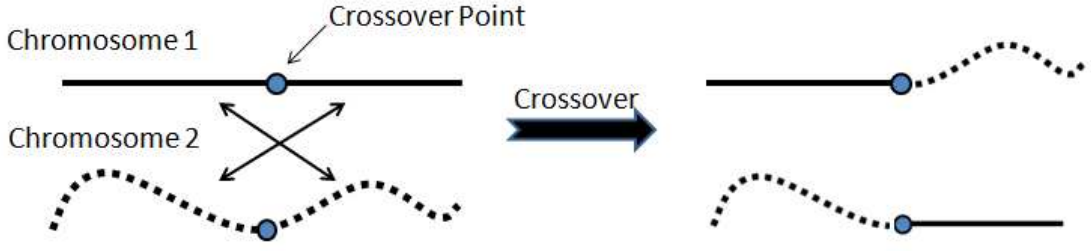
여기서 f_i는 염색체 i의 적합도 값을 나타내고, m_i는 염색체(경로) i의 길이를 의미하며, $g_i(j)$는 i번째 염색체의 j번째 위치의 유전자(노드)를 나타내고, $C(i,j)$는 노드 i와 j를 있는 간선의 간선 비용을 의미한다.

2.4 선택(Selection)

Ahn은 선택 방법으로 비복원 승자진출 선택(tounament selection without replacement)을 채택하였다. 비복원 승자진출 선택은 서로 다른 염색체를 개체군으로부터 n개 선택하여 가장 우수한 적합도를 갖는 염색체를 다음 세대의 부모 염색체로 선택하는 방식을 의미한다. 승자 진출전의 크기 n은 2로 고정한다.

2.5 교배(Crossover)

Ahn이 사용한 교배 방법은 간단하다. 그 개념은 (그림 1)에 기술하였다. 우선 선택된 두 염색체의 유전자 위치에 상관없이 동일한 노드를 포함하고 있는 유전자의 위치 쌍에 대한 집합을 형성하고, 그 후 집합 중 하나를 임의로 선택한다. 다시 말해, 두 염색체에 동시에 등장하는 유전자(노드)를 전부 찾아 그 중 하나를 랜덤하게 고른다. 이는 교배 동작을 수행할 때 교차점(crossover point) 된다.



(그림 1) 교배

(그림 2) 돌연변이

(그림 3) 복구 함수

2.6 돌연변이(Mutation)

일반적으로 돌연변이는 염색체의 유전자를 다른 대립형질로 돌연변이 시켜서 개체군의 유전적 다양성을 유지하는 역할을 수행한다. 최단 경로 찾기 문제에서의 돌연변이 연산의 개념은 (그림 2)에 기술하였다. 돌변 변이를 위해 선택된 염색체의 한 유전자를 임의로 선택하고 이러한 유전자를 돌연변이 점(mutation point)라고 지칭한다. 그 후 이 돌연변이 점으로부터 다음 노드의 선택을 랜덤하게 선택해 나간다. 이러한 절차를 목적지 노드가 선택될 때까지 반복한다.

2.7 복구함수(Repair Function)

교배의 결과로 나온 염색체는 경로 상에 루프를 포함할 수 있다. 염색체 상의 이러한 부적합(infeasible) 염색체를 제거하기 위해 Ahn은 복구 함수를 제안했다. 그 개념도은 그림 3에 기술했다. (그림 3)과 같이 루프의 만나는 노드를 찾아서 그 중간 노드를 제거함으로써 루프를 제거해 준다. 예를 들어 다음과 같은 염색체(경로)라면

$$1,2,3,4,5,6,3,7,8$$

루프의 만나는 노드인 3을 찾고 그 중간 노드들 4, 5, 6 제거하면 된다. 즉 결과는 다음과 같다.

$$1,2,3,4,5,6,3,7,8 \rightarrow 1,2,3,7,8$$

2.8 전체 알고리즘

이제 최단 경로 찾기에 대한 유전자 알고리즘을 다음과 같이 기술할 수 있다.

[Algorithm 1] Ahn's Genetic Algorithm
1: Initialize the population
2: **repeat until** convergence{
3: calculate the fitness of individuals in population
4: do selection
5: do crossover
6: remove loops by repair function
7: do mutation
8: **end**

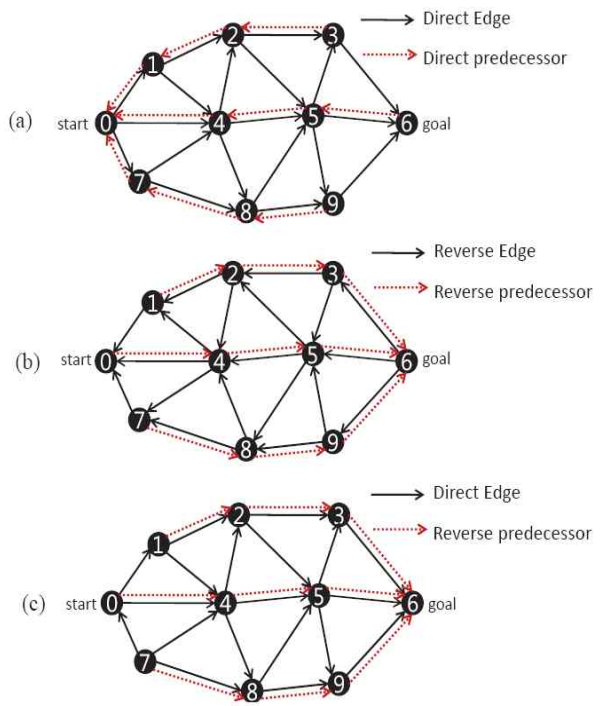
3. 제안 알고리즘

지금부터 제안 알고리즘을 설명한다. 제안 알고리즘의 목적은 유전자 알고리즘을 이용해서 최단 경로의 빠른 '재'탐색을 하는 것이다. 이를 위해서 초기 경로 탐색 시 다익스트라 알고리즘을 사용한다고 가정한다. 다익스트라 알고리즘은 단일 시작점과 모든 쌍(Single Source-All Pair)의 목적 지점에 대한 최단 경로를 찾는 알고리즘이다[6]. 다익스트라 알고리즘은 오랜 시간의 연구로 자료구조의 최적화에 대한 많은 연구가 있다. 본 연구에서는 Cherkassky et al.의 다익스트라 알고리즘의 실험 논문에서 버킷과 오버플로우 백(overflow bag)을 사용한 다익스트라 알고리즘을 사용하였다[7].

3.1 개체군 초기화(Population Initialization)

Ahn의 무작위적 초기화는 그래프의 크기가 커질수록 잘 동작하지 않는다. 그래프의 크기가 커질수록 각 노드에서 다음 노드를 랜덤으로 선택하는 방법으로는 실제로 목적지 노드에 도착할 확률이 지수적으로(exponential) 감소하기 때문이다. 본 연구팀은 실험적으로 약 1000개 이상의 노드 개수를 가진 그래프에 대해서 합리적인 시간에 개체군 초기화를 하지 못한다는 것을 밝혀냈다. 본 연구에서는 이를 극복하기 위해서 다익스트라 알고리즘을 이용한다.

다익스트라 알고리즘은 알고리즘의 실행 결과로써, 선행자 배열을 반환한다. 임의의 노드를 a라 하자. 선행자 배열은 (그림 4(a))의 그래프의 점선 화살표와 같이 노드 a에 대해서 시작점에서 노드 a까지의 최단 경로를 만드는 이전 노드를 가리키는 정보를 저장한다. 즉 노드 a에서부터 선행자를 따라 도착점까지 가면 도착점에서 노드 a까지의 최단



(그림 4) 선행자(predecessor) 배열의 구성

거리가 완성 된다. 이 선행자 배열을 이용하면 유전자 알고리즘의 개체군의 초기화를 빠르게 할 수 있다.

$G(N,E)$를 노드의 집합 N과 간선의 집합 E를 가진 방향 그래프라 하자. 그리고 모든 간선의 방향을 반전시킨 그래프를 $G_{reverse}(N,E)$라 하자. 예를 (그림 4(a))의 그래프를 반전 시키면 (그림 4(b))의 그래프처럼 되고 이 반전된 그래프의 선행자 배열은 역시 반전된다.

이 반전 그래프 $G_{reverse}(N,E)$에서의 반전된 선행자 배열(Reverse Predecessor)은 (그림 4(c))의 그래프처럼 반전 그래프가 아닌 원래의 그래프 $G(N,E)$에서 각 노드에서 목적지까지의 최적의 다음 노드를 가리킨다. 이 반전된 선행자 배열을 이용하면 몇 번 최단 경로에 포함되지 않는 노드를 선택해도 다시 선행자 배열의 점선 화살표대로 따라가면 결국 목적지에 도착할 수 있다. 예를 들어 (그림 4(c))의 그래프에서 시작점에서 차량이 반전된 선행자의 점선 화살표를 따라가지 않고 1번 노드로 간 후 선행자를 따라가면 목적지에 도착할 수 있다. 시작점에서 뿐만 아니라 차량 이동 중간 중간에 길을 잘 못 들어서 최단 경로에서 벗어났을 때, 반전된 선행자를 따라가면 결국에 목적지에 도착하게 된다. 반전된 선행자는 길을 잃은 차량의 가이드라인 같은 역할을 한다.

강화 학습 분야에서는 모든 각 노드에서 다른 노드로의 최적의 행동(한 노드에서 다른 노드로의 이동)의 사상 함수 $\pi^* : N \mapsto N$ 을 최적 정책(optimal policy)라 부른다[8]. 에이전트가 현재 자신이 있는 노드에서 다음 노드로의 행동을 선택 시 $1-\epsilon$의 확률만큼은 랜덤으로 선택하고 ϵ 확률만큼은 최적 정책에 따라 행동을 선택하는 방법을 $\epsilon-greedy$ 행동선택법이라 한다. 즉 수식 (2)와 같다.

$$\epsilon-greedy(s,pred) = \begin{cases} pred(s), & if\ \zeta < \epsilon \\ \text{다음 노드를 랜덤하게 고른다.} & otherwise \end{cases} \quad (2)$$

여기서 $pred$는 $G_{reverse}(N,E)$의 선행자 배열을 의미하고 s는 현재 노드, ζ는 $0<\zeta<1$의 난수를 의미한다. 이제 Routine 1과 같이 제안 알고리즘의 개체군 초기화 루틴을 기술할 수 있다.

[Routine 1] PopulationInitialize($pred$,$start$, goal)
1: **for** $i=0$ **to** PopulationSize - 1
2: set $s_{cur}=start$ and $chromosome[i]=[s_{cur}]$
3: **repeat**
4: $s_{cur}=\epsilon-greedy(s_{cur},pred)$
5: $chromosome[i]=[chromosome[i]\,s_{cur}]$
6: **until**($s_{cur}=goal$)
7: **end**
8: **return** $chromosome$

Ahn은 개체군 초기화를 위해 네트워크의 토폴로지정보가 저장되어 있는 DB를 직접 접근하여 개체군 초기화를 위해 방문한 노드의 DB 정보를 삭제함으로써 사이클 생성을 방지할 수 있다고 기술했다. 하지만 네트워크의 사이즈가 커지면 네트워크 DB에 접근하는 것 자체가 상당한 오버헤드(overhead)가 된다. 하지만 제안 알고리즘의 개체군 초기화는 단지 선행자 배열로만 개체군을 초기화함으로써 네트워크 DB로의 접근을 없애고 사이클은 개체군 초기화 후 복구 함수를 통해 제거하였다.

3.2 적합도 함수(Fitness Function)

제안 알고리즘은 매시간 동적으로 변하는 교통 네트워크에서의 최단 경로 재탐색이 목적이므로, 각 염색체의 적합도는 물리적인 거리의 합이 아니라 주행시간의 합이 될 것이다. 즉 수식 (1)에서 $C(x,y)$는 노드 x에서 노드 y로까지의 주행 시간이 된다.

그래프의 사이즈가 커지면 개체군의 각 염색체에 대해서 적합도를 구하는 과정도 계산 비용이 비싸진다(expensive). 왜냐하면, 사이즈가 큰 그래프 $G(N,E)$에 대해서는 비용 함수 C의 자료 구조를 $|N|\times|N|$의 행렬로 표현해 특정 노드 x,y의 간선 비용 $C(x,y)$를 메모리상에서 직접 접근(direct access)할 수 있는 것이 아니기 때문이다. 이를 최적화하기 위해서 3.1절에서 형성한 개체군의 염색체에 나타난 간선 "x,y"를 키(key)로 하는 해시 테이블을 구성하였다.

개체군에 등장한 간선으로만 테이블을 구성하기 때문에 작은 크기의 해시 테이블로도 충돌 없이 빠른 접근을 할 수 있다. 개체군의 염색체에 등장한 간선 (x,y)들의 집합 Ω을 실시간 교통 정보 시스템에 요구함으로써, 전체 도로에 대한 정보가 아니라 일부 정보만을 실시간 교통 정보 시스템으로부터 받게 되어 교통 정보 통신에 대한 비용도 낮출 수 있다.

3.3 선택(Selection)

Ahn이 사용한 비복원 승자진출 선택(tournament selection without replacement)은 비록 계산 비용은 비싸지 않지만($O(|Chromosome|)$, 여기서 $|Chromosome|$은 개체군의 수), 개체군에서 적합도가 높은 염색체끼리 만났을 때 승자진출전에 패배한 염색체는 높은 적합도를 가졌음에도 개체군에서 배제된다는 문제점을 가지고 있다. 본 연구에서는 이를 해결하기 위해 다음과 같은 선택 방법을 사용했다.

a) 개체군의 모든 염색체의 평균 적합도를 구한다.
b) 개체군에서 평균 적합도보다 높은 적합도를 가진 염색체를 다음세대의 염색체로 선택한다.
c) 개체군에서 평균 적합도보다 낮은 적합도를 가진 염색체는 b)에서 살아남은 염색체 중 하나로 대치한다.

본 연구에서는 개체군의 수는 10~50개 정도로 작은 수의 고정된 크기를 가지므로, 이러한 염색체 선택법은 조금 비싼 계산 비용을 가지지만, 선택의 정확도를 높일 수 있다. 위에 설명한 대로 염색체를 선택하면 $O(3\times|Chromosome|)$의 계산 복잡도를 가지지만(각 a), b), c) 단계가 각각 $O(|chromosome|)$의 시간 복잡도를 가진다), Ahn의 염색체 선택 방법의 단점을 극복할 수 있다.

3.4 교배(Crossover)

Ahn의 유전자 알고리즘은 두 염색체의 교배 시 두 염색체에 대해 동일한 노드를 포함하고 있는 유전자의 위치 쌍을 모두 찾아서 그 중 한 교차점을 랜덤하게 선택하였다. 두 염색체의 크기(경로의 길이)를 각각 α와β라고 하면 이 과정의 시간 복잡도는 $O(\alpha\beta)$가 되어서 작은 크기의 염색체에 대해서는 문제가 되지 않는다. 하지만 네트워크의 크기가 커져서 각 염색체의 크기도 커지게 된다면, 단순히 교차점을 찾는 루틴의 계산 비용이 비싸지게 된다. 이를 최적화하기 위해 [Routine 2]와 같은 교차점 찾기 루틴을 사용하였다.

[Routine 2] Search Crossover Point
//x_1, x_2 are two chromosomes to crossover.
1: $s_1 = rand\%size(x_1)$
2: **if** $(s_1 == 0)$ **then** $e_1 = size(x_1)-1$
3: **else** $e_1 = s_1 - 1$ **endif**
4: $s_2 = rand\%size(x_2)$
5: **if** $(s_2 == 0)$ **then** $e_2 = size(x_2)-1$
6: **else** $e_2 = s_2 - 1$ **endif**
7: **for** $(i = s_1;\ s_1 \neq e_1;\ ++i)$
8: **for** $(j = s_2;\ s_2 \neq e_2;\ ++j)$
9: **if**$(x_1[i] == x_2[j])$ **then return** i,j **endif**
10: **if**$(j \equiv size(x_2)-1)$ **then** $j = 0$ **endif**
11: **endfor**
12: **if**$(i \equiv size(x_1)-1)$ **then** $i = 0$ **endif**
13:**endfor**

의사 코드의 1, 3번째 줄은 각각 염색체 x_1, x_2에서 랜덤하게 탐색의 시작점 s_1, s_2을 고른다. 의사 코드의 7번째 줄에서 13번째 줄은 그렇게 랜덤하게 고른 시작점 s_1, s_2에서 시작해서 교차점 탐색을 시작하고 최초로 찾은 교차점을 알고리즘의 결과로 돌려준다. 끝까지 찾아도 교차점을 찾지 못할 경우 다시 처음으로 돌아가서 찾는 순환식 방법을 사용한다. 교차를 하는 방식과 교배의 결과로 생기는 부적합한 염색체들의 복구는 Ahn의 유전자 알고리즘과 같은 방식을 사용하였다

3.5 돌연변이(Mutation)

3.2에 기술한대로 개체군의 염색체(경로)에는 개체군의 염색체에 등장한 간선 (x,y)들의 집합 $(x,y)\in\Omega$만이 등장해야 한다. 만약 염색체의 돌연변이로 인해서 집합 Ω의 원소가 아닌 간선 $(x',y')\notin\Omega$이 염색체에 등장하게 된다면, 실시간 교통 정보 시스템과의 추가적인 정보교환을 통해 간선(도로) $(x',y')\notin\Omega$의 실시간 정보를 가져와야(fetch) 한다. 이를 막기 위해 제안 알고리즘에서는 돌연변이의 과정을 생략했다. 본 연구에서는 실험을 통해 돌연변이 과정이 생략이 제안 알고리즘의 수렴 시간에 아무런 영향을 주지 않는다는 것을 알아냈다.

3.6 전체 알고리즘

이제 제안 알고리즘의 전체적인 의사 코드를 다음과 같이 기술할 수 있다.

[Algorithm 2] Proposed Algorithm(K,$pred$)
//K is the size of population
//$pred$ is obtained in initial path search using //Dijkstra''s algorithm
1: construct the population using Routine 1 (3.1절)
2: remove loops in chromosomes of population Y by repair function (2.7절)
3: construct hash table using real time traffic information (3.2절)
4: **Repeat until** convergence {
5: calculate the fitness of population Y (3.2절)
6: do selection (3.3절), crossover(3.4절)

7: remove loops in Y (2.7절)
8 **end**
9 **return** Y

알고리즘의 수렴은 염색체의 개체군이 모두 같아지는 것으로 체크한다.

4. 실험 및 결과

제안 알고리즘을 성능을 측정하기 위해서 노드 크기 50부터 120000개까지의 강력하게 연결된(strongly connected) 네트워크를 랜덤하게 생성하였다. 각 네트워크에 대해서 간선의 개수는 노드의 2배를 부여하였다. 예를 들어 노드의 개수가 50이라면 간선의 개수는 100이다. 간선의 비용으로는 1~9999까지의 난수를 부여하였다.

4.1 제안 알고리즘의 매개 변수(parameter)

제안 알고리즘에는 $\epsilon - greedy$의 ϵ과 개체군 수라는 두 개의 매개 변수가 존재한다. ϵ이 감소할수록 네트워크에 대해 다양성이 높은 개체군을 형성하기 때문에 평균 주행시간은 감소하지만 평균 계산시간은 증가한다. 개체군 수도 마찬가지로 개체군 수가 증가할수록 개체군의 다양성이 좋아지므로 평균 주행시간은 감소하지만 평균 계산시간은 증가한다. 본 연구에서는 실험에 사용한 네트워크에 맞게 ϵ은 0.5에서 0.7까지 개체군의 수는 20에서 50까지 사용하였다. 네트워크의 노드 개수가 커질수록 더 높은 ϵ과 개체군의 수를 사용하였다. <표 1>은 노드 크기별 사용한 매개변수를 나타낸다.

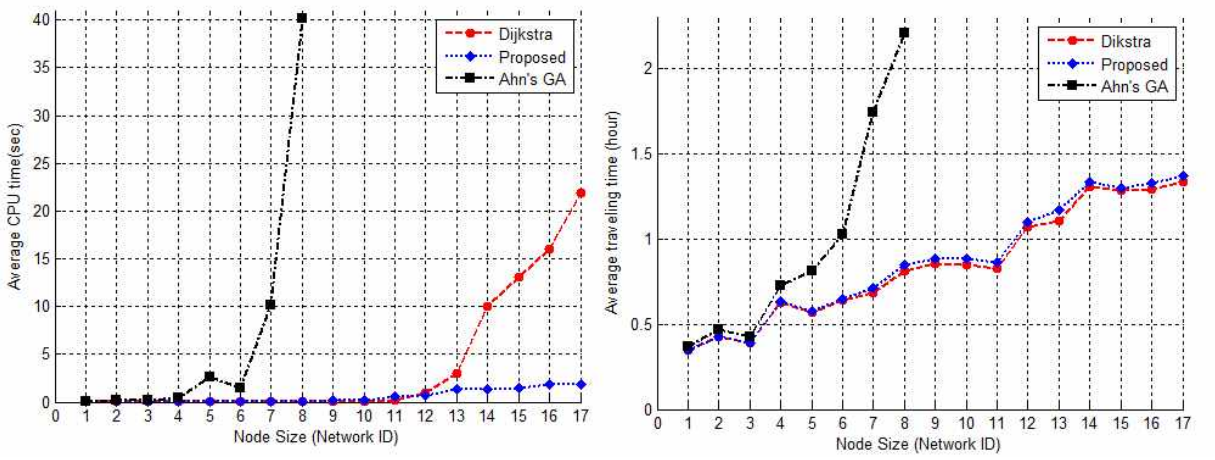
4.2 제안 알고리즘의 성능 비교

실시간 도로 상황을 시뮬레이션하기 위해 모델을 구성하였다. 랜덤으로 생성된 노드 크기별 네트워크에 대해서 각 간선의 비용을 거리로 놓고, 간선 마다 현재 속도라는 필드를 추가했다. 간선의 현재 속도는 각 차량이 한 노드에서 다른 노드로 이동 시 마다 동적으로 변경하였다. 변경되는 속도는 평균 속도 80km/h에 표준편차 20km/h의 정규 분포에서 추출된 난수를 이용해서 변경하였다.

300개의 (시작점, 도착점) 쌍을 생성하여 각각 Ahn의 유전자 알고리즘(Ahn's GA), 제안 유전자 알고리즘(proposed), 다익스트라 알고리즘에 대해서 성능 평가 실험하였다. 차량이 노드에서 노드로 이동시 마다 각 알고리즘은 현재의 네트워크 상황에 맞게 다시 계산되어 경로를 재설정한다. 다익스트라 알고리즘으로는 앞서 언급한 대로 Cherkassky et al.의 다익스트라 알고리즘의 실험 논문에서

<표 1> 실험에 사용된 네트워크 크기 및 매개 변수

Network ID	Network Size	Epsilon	Population Size
#1	50	0.5	20
#2	100	0.5	20
#3	200	0.5	20
#4	400	0.5	20
#5	800	0.5	20
#6	2000	0.5	20
#7	4000	0.5	20
#8	8000	0.5	20
#9	20000	0.5	30
#10	40000	0.6	30
#11	80000	0.6	40
#12	160000	0.6	40
#13	320000	0.6	40
#14	640000	0.7	40
#15	800000	0.7	40
#16	1000000	0.7	40
#17	1200000	0.7	50



(그림 5) 각 알고리즘 별 성능 비교

버킷과 오버플로우 백(overflow bag)을 사용한 다익스트라 알고리즘을 사용하였다[7]. (그림 5)는 각각 노드 크기별로 세 개의 알고리즘에 대해서 평균 주행시간과 평균 계산 시간을 측정한 그래프이다. 그래프에 x축은 <표 1>의 네트워크 ID를 의미한다. 실험 결과를 보면 Ahn의GA는 약 노드 개수 10000개 이상의 네트워크에 대해서는 합리적인 평균 계산 시간(Average CPU time)에 해를 구할 수 없었다. 게다가 해를 구하더라도 해의 질(quality) 즉 평균 주행 시간(Average traveling time)이 최적인 다익스트라 알고리즘의 해의 질과 차이가 컸다. 제안 알고리즘은 노드 크기 40000개 이하의 네트워크에 대해서는 다익스트라 알고리즘과 비슷한 평균 계산 속도를 보였지만 네트워크의 사이즈가 커질수록 다익스트라의 계산 속도를 큰 차이로 앞서게 된다는 것을 알 수 있다. 제안 알고리즘의 평균 주행 시간도 최적인 다익스트라와 차이가 크지 않고 거의 최적의 경로를 제시해 준다는 것을 알 수 있다.

5. 결론 및 향후 과제

지금까지 유전자 알고리즘을 이용한 확장성 있고 빠른 경로 재탐색 알고리즘을 제시하였다. 이 방법은 최초 탐색 시 다익스트라 알고리즘을 이용해야 한다는 단점이 있으나, 동적으로 변하는 네트워크에 대해서 재탐색 시, 기존 Ahn이 제안한 유전자 알고리즘과 비교해서 빠르고 큰 네트워크에 대해서도 잘 작동한다는 것을 확인하였다.

실시간 교통 정보를 이용하는 내비게이션의 최단 경로 탐색 알고리즘은 실시간 교통 정보가 내비게이션 클라이언트에 전달될 때마다 재탐색을 해야 하기 때문에 재탐색이 빠르다는 장점을 가진 제안 알고리즘의 좋은 응용 분야이다. 실시간 교통정보를 이용하는 내비게이터는 서버와 클라이언트 간에 실시간 교통 정보를 통신해야 한다. 제안 알고리즘은 개체군에 나타나는 네트워크 정보만을 통신함으로써 이러한 서버와 클라이언트 간의 통신을 효율화할 수 있다.

GA는 하나의 큰 개체군을 작은 여러 개체군으로 나누어, 작은 여러 개의 개체군에 대해서 독립적으로 진화할 수 있기 때문에 병렬화(Parallelization)에 대한 많은 연구가 있다[9-11]. 이러한 병렬한 기법을 이용하면 제안 알고리즘을 병렬화할 수 있는데, 이것은 다익스트라 알고리즘이 갖지 못하는 장점이 될 수 있을 것이다. 이에 대한 연구는 향후 과제로 남겨 둔다.

참 고 문 헌

[1] TPEG, website, http://www.tpeg.org

[2] Ahn, C. W. and Ramakrishna, R. S. 2002. "A genetic algorithm for shortest path routing problem and the sizing of populations." IEEE Trans. Evol. Comput., Vol.6, pp.566 - 579, Dec.

[3] Goldberg, D. E. , "Genetic algorithms in Search, Optimization and Machine Learning", Addison-Wesley Longman, Inc, Boston, 1989.

[4] Kanoh, H., "Dynamic route planning for car navigation systems using virus genetic algorithms," International Journal of Knowledge-based and Intelligent Engineering Systems, Vol.11, pp.65-78, 2007.

[5] Kanoh, H., and Hara, K. ""Hybrid genetic algorithm for dynamic multiobjective route planning with predicted traffic in a real-world road network,"" in Proceedings of the 10th annual conference on on Genetic and evolutionary computation. ACM, 2008, pp.657-664.

[6] Dijkstra, E. W., "A note on two problems in connexion with graphs." Numerische Mathematik 1: 269 - 271, 1959.

[7] Cherkassky B. V., A. V. Goldberg and T. radzik, "Shortest Paths Algorithms: Theory and Experimental Evaluation",Technical report 93-1480, Computer Science Department, Stanford University, 1993.

[8] Sutton, R.S., and Barto, A. G."Reinforcement Learning." MIT Press, 1997.

[9] Cant-paz, E. "A survey of parallel genetic algorithm", Calculateurs Paralleles, Vol.10, pp.141-171, 1998.

[10] Yi, W., Q. Liu, and Y. He, "Dynamic distributed genetic algorithm", Proceedings of the 2000 Congress on Evolution Computation, Vol.2, pp1132-1136, 2000.

[11] Alba, E., "Parallel Metaheuristics : A New Class of Algorithms", JohnWiley & Sons, Inc., New Jersey, 2005.

이 정 규

e-mail : sweaterr@gmail.com

2008년 서강대학교 수학과(학사)

2010년 서강대학교 컴퓨터공학과(석사)

2011년~현 재 (주)사이람 연구원

관심분야 : 기계학습, Collaborative Filtering, 소셜네트워크 분석, System Biology

김 선 호

e-mail : shkim@lex.yonsei.ac.kr

2002년 8월 연세대학교 컴퓨터과학과 (박사)

2002년 9월~2003년 8월 연세대학교 언어정보원 전임연구원

2003년 9월~2006년 8월 고려대학교 컴퓨터공학과 Post-doc

2006년 9월~현 재 서강대학교 컴퓨터공학과 BK 연구 교수

관심분야 : 자연어처리, 기계학습, 생물정보학, 정보검색, 데이터 마이닝, 정보추출, 기계번역

양 지 훈

e-mail : yangjh@sogang.ac.kr

1987년 서강대학교 전자계산학과(학사)

1989년 ISU Department of Computer Science(석사)

1999년 ISU Department of Computer Science(박사)

1999년~2000년 HRL Laboratories, LLC, Malibu, CA 연구원

2000년~2002년 SRA International, Inco, Fairfax, VA 연구원

2002년~현 재 서강대학교 컴퓨터공학과 부교수

관심분야 : 기계학습, 바이오인포메틱스 등