공간 네트워크 데이터베이스에서 공간 제약을 고려한 경로 내 최근접 질의처리 알고리즘† ††

In-Route Nearest Neighbor Query Processing Algorithm with Space-constraint in Spatial Network Databases

김용기* / Yong-Ki Kim, **김아름**** / Ah-Reum Kim, **장재우***** / Jae-Woo Chang

요 약

최근 공간 네트워크 데이터베이스를 위한 질의처리 알고리즘에 관한 연구가 많은 관심을 받고 있으나, 경로-기반 질의에 대한 연구는 매우 미흡한 실정이다. 공간 네트워크 데이터베이스에서는 이동객체가 공간 네트워크상에서만 이동하기 때문에, 위치기반 서비스 및 텔레매틱스의 응용을 지원하기 위해 경로 내 최근접(In-Route Nearest Neighbor: IRNN) 질의와 같은 경로-기반 질의에 대한 효율적인 질의처리 알고리즘 연구가 필수적이다. 그러나 기존 경로 내 최근접 질의처리 알고리즘은 도로내의 병목현상을 반영하지 못하는 문제점이 존재한다. 따라서 본 논문에서는 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘을 제안한다. 마지막으로, 기존 알고리즘과의 성능 비교를 통하여 제안하는 알고리즘이 우수함을 보인다.

Abstract

Recently, the query processing algorithm in the field of spatial network database(SNDB) has been attracted by many interests. But, there is little research on route-based queries. Since the moving objects move only in spatial networks, the efficient route-based query processing algorithms, like in-route nearest neighbor(IRNN), are essential for Location-based Service(LBS) and Telematics application. However, the existing IRNN query processing algorithm has a problem that it does not consider traffic jams in the road network. In this thesis, we propose an IRNN query processing algorithm which considers space restriction. Finally, we show that space-constrained IRNN query processing algorithm is efficient compared with the existing IRNN algorithm.

주요어 : 공간 네트워크 데이터베이스, 질의처리 알고리즘, 경로 내 최근접 질의

Keyword : Spatial network database, query processing algorithm, in-route nearest neighbor query

† 이 논문은 2008년도 정부(교육과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임 (NO. R01-2008-000-11002-0)

†† 본 논문은 2005년도 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단의 지원을 받아 연구되었음 (KRF-2005-041-D00656)

■ 접수일 : 2008.02.28 ■ 수정일 : 1차 2008.04.28 / 2차 2008.06.20 ■ 심사완료일 : 2008.06.20

* 전북대학교 전자정보공학부 컴퓨터공학과 박사과정(ykkim)@dblab.chonbuk.ac.kr

** 전북대학교 전자정보공학부 컴퓨터공학과 석사과정(arkim)@dblab.chonbuk.co.kr

*** 교신저자 전북대학교 전자정보공학부 컴퓨터공학과 교수(jwchang)@chonbuk.ac.kr

1. 서 론

최근 공간 네트워크 데이터베이스를 위한 질의처리 알고리즘에 관한 연구(좌표-기반 질의, 궤적-기반 질의, 경로-기반 질의)가 활발히 진행되어 왔다. 그러나 현재 좌표-기반 질의에 대한 연구는 활발히 진행 중이나, 경로-기반질의에 대한 연구는 매우 미흡한 실정이다[1,2,3,4,5,6,7]. 공간 네트워크 데이터베이스에서는 이동객체가 공간 네트워크상에서만 이동하기 때문에 경로에 기반을 둔 질의의 유용성이 매우 증대되며, 따라서 경로-기반 질의에 대한 효율적인 질의처리 알고리즘 연구가 필수적이다[8,9,10].

공간 네트워크 상에서의 경로-기반 질의는 연속 최근접(continuous nearest neighbor)질의, 연속 k-최근접(continuous k-nearest neighbor)질의, 경로 내 최근접(in-route nearest neighbor)질의로 분류할 수 있다. 첫째, 미국 Southern California 대학에서는 공간 네트워크상에서 보로노이 다이어그램(Voronoi network diagram)을 이용하여 미리 찾고자 하는 POI들의 네트워크 거리를 미리 계산하여 연속 최근접 질의 및 연속 k-최근접 질의를 수행할 수 있는 알고리즘을 제시하였다[9]. 둘째, 싱가폴의 SMU(Singapore Management University) 대학에서는 공간 네트워크상에서 R-tree를 이용한 연속 최근접 질의 및 연속 k-최근접 질의를 처리하기 위한 알고리즘을 제시하였다[10]. 이 연구에서는 연속 최근접 질의를 위하여, 점진적 모니터링 알고리즘(incremental monitoring algorithm)과 그룹 모니터링 알고리즘(group monitoring algorithm)을 제시하였다. 마지막으로, 미국 Minnesota대학의 S.Y. Jin 와 S. Shekhar는 경로 내 최근접(In-Route Nearest Neighbor: IRNN) 질의를 정의하였다[8]. 이는 사용자가 자신이 평소 이용하는 경로를 적게 벗어나면서 원하는 최근접점을 찾는 데에 초점을 맞추고 있다.

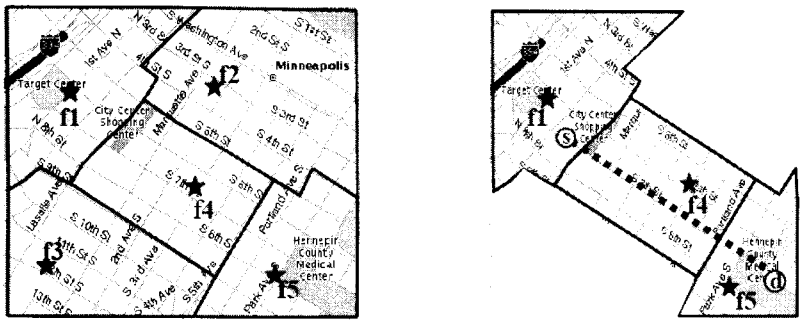
한편, 경로-기반 질의 중에서 경로 내 최근접 질의처리 알고리즘은 병목현상이 발생한 도로 내의 실시간 상황을 반영하지 못하는 문제점이 존재한다. 이러한 문제점을 극복하기 위해서 본 논문에서는 기존 경로 내 최근접 질의처리 알고리즘을 확장하여 공간제약을 고려한 질의처리 알고리즘을 제안한다. 제안하는 질의처리 알고리즘은 기존 경로 내 최근접 질의처리 알고리즘에서 고려하지 못한 공간제약을 이용하여 텔레매틱스(telematics), CNS(Car Navigation System), 자동항법장치, L-commerce 등과 같은 실제 응용에서 공간제약을 이용하여 최적 조건의 POI(Point of Interest) 및 경로를 탐색할 수 있도록 한다. 아울러 기존의 경로 내 최근접 질의처리 알고리즘과 본 논문에서 제안하는 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘에 대해 성능평가를 수행한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로써 공간 네트워크 데이터베이스를 위한 질의처리 알고리즘에 관한 연구들을 소개한다. 3장에서는 기존 경로 내 최근접 질의의 문제점에 기반하여 공간제약을 고려한 효율적인 경로 내 최근접 질의처리 알고리즘을 제시한다. 4장에서는 기존 경로 내 최근접 질의처리 알고리즘과 제안하는 공간제약 경로 내 최근접 질의처리 알고리즘의 성능평가를 수행한다. 마지막으로 5장에서는 결론 및 향후연구를 제시한다.

2. 관련 연구

본 장에서는 공간 네트워크 데이터베이스에서의 Minnesota대학의 S.Y. Jin 와 S. Shekhar가 제시한 공간 네트워크상에서 경로 내 최근접 질의에 대해 살펴본다[8]. S.Y. Jin 와 S. Shekhar는 경로 내 최근접 질의를 수행하기 위해 4가지 알고리즘을 제안하였다.

첫째, 단순 그래프(Simple graph) 기반 방법으로, 경로상의 각 노드마다 최근접점을 찾은 후 각 노드의 최근접점을 경유하여 목적지에 도달하는 최단 거리를 계산하여 가장 최단거리가 되는 최근접점을 찾는 방법이다. 이 방법은 노드가 많아지면 최근접점을 찾는 비용이 증가한다는 단점이 있다. 둘째, 반복 공간 범위(recursive spatial range) 질의 기반 방법으로, 현재 위치에서의 최근접점을 최단



(a) 미리 계산된 영역 및 POI　　(b) 질의 경로에 대한 후보 POI

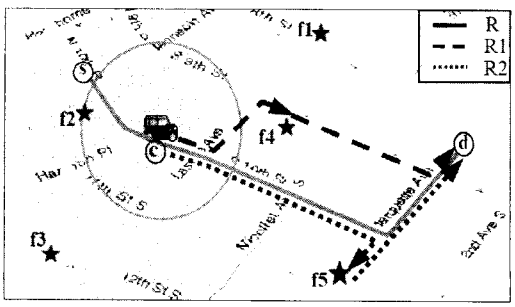
<그림 1> PCZ 기반 경로 내 최근접 질의

거리 알고리즘(shortest path algorithm)을 통해 찾은 후, 이미 찾은 최근접점까지의 거리를 이용하여 경로상의 다른 노드에서의 최근접점을 영역(range) 질의를 통해 찾는다. 만일 여기서 찾은 최근접점의 거리가 더 작다면 영역 범위를 이 거리로 조절한 후 다음 노드에서 영역질의를 수행한다. 이 방법은 첫 번째 방법에 비해 비용이 감소하나 각 노드에서의 영역질의를 수행하는 비용이 많이 든다. 셋째, 공간 거리 조인(spatial distance join) 기반 방법으로, 각 노드에서의 영역질의를 수행하는 비용을 줄이기 위해 제안되었다. 이 방법은 노드와 POI에 대한 두 가지의 트리를 가지고 경로 상에 존재하는 각 노드와 POI의 유클리디언 거리 조인을 통해 영역질의 범위를 구하여 최근접점을 구한다. 마지막으로, PCZ(Pre-Computed Zone) 기반 방법은 각 POI마다 서비스 지역(service zone)을 설정하여 노드에서 POI까지의 네트워크 거리를 미리 계산한다<그림 1(a)>. 그리고 각 노드에 대해 가장 가까운 POI를 B+-Tree에 저장한다. 이를 이용하여 질의 경로가 주어지면, 해당경로를 포함하는 서비스 지역을 검색하여 최근접점을 빠르게 찾을 수 있다<그림 1(b)>. 제안된 4가지 방법 중에서, PCZ 기반 방법은 노드에서 최근접 POI를 미리 계산하여 저장한 색인을 이용함으로써, 질의 경로상의 가까운 POI들을 네트워크 탐색 없이 찾을 수 있기 때문에 가장 우수한 성능을 보인다.

3. 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘

3.1 기존 경로 내 최근접 질의처리 알고리즘의 문제점

기존 경로 내 최근접 질의는 경로를 가장 적게 벗어나면서 최근접점을 찾는 데에 초점을 맞추고 있다. 이러한 질의의 예는 <그림 2>에서 경로 R2와 같이, "출발 지점(S)에서 목적지(D)까지 표시된 경로로 운행하고자 계획한 자동차가(경로 R), 그 경로 내에서 가장 적게 경로를 벗어나면서 방문할 수 있는 주유소를 찾아라." 이다. 이는 출발 지점에서 f4를 지나 목적지까지 운행(경로 R1)하는 것이 더 짧은 운행 결과이지만, 경로 내 최근접 질의는 사용자가 경로를 최소한으로 벗어나려는 경향을 반영하므로 f5를 거쳐 운행하는 경로(경로 R2)가 결과이다.



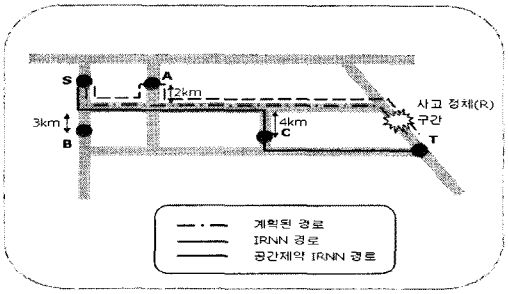
<그림 2> 경로 내 최근접 질의 개념

그러나 기존 경로 내 최근접 질의는 주유소와 같은 POI(Point of Interest) 검색 시, 현재 도로의 실제 상황을 고려하지 않는 문제점이 존재한다. 예를 들면, <그림 2>에서 "주어진 경로(S에서 D까지)로 운행하고자 계획한 자동차가, 목적지까지 가는 경로의 중간 구간에 사고로 인하여 정체현상이 발생하였을 때, 주어진 경로를 최대한 유지하면서 최적의 경로를 찾아라." 의 질의는, 우회하여 가는 것과 정체구간을 지나가는 것에 대한 비용을 고려하여 경로를 선택하여야 한다. 따라서 이러한 질의는 기존 경로 내 최근접 질의처리 알고리즘으로 처리하지 못하는 단점이 존재한다. 그러나 이러한 질의는 실제 응용에서는 매우 유용하게 사용될 수 있다. 따라서 텔레매틱스(telematics), CNS(Car Navigation System), 자동항법장치, L-commerce 등과 같은 응용을 효과적으로 처리하기 위해서는, 공간제약을 이용하여 최적 조건의 POI 및 경로를 탐색하는 공간제약을 고려한 경로 내 최근접 질의 처리 알고리즘에 대한 연구가 필수적이다.

3.2 공간제약 경로 내 최근접 질의처리 알고리즘

기존 경로 내 최근접 질의는 현재 도로의 상황을 고려하지 못하는 문제점이 존재하기 때문에, 이를 해결하기 위해 본 논문에서는 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘을 제안한다. 일반적으로 자동차가 예정된 경로를 운행할 때, 설정된 경로에 천재지변이 일어나 운행이 불가능한 경우 또는 교통사고 및 교통체증이 발생할 경우 사용자가 그 구간을 지나지 않고 새로운 길로 운행하고자 한다. <그림 3>을 통해 예를 들면, "현재 지점(S)에서 시청(T)까지 평소 운행하는 경로에서 사용자가 은행을 경유하고자 할 때, 사고가 발생하여 정체되는 R구간을 지나는 것보다 최적인 경로(평소 운행하는 경로를 가장 적게 벗어나는)를 찾아라." 와 같은 질의이다.

<그림 3>에서 A, B, C로 표시된 것은 은행 POI를 나타내며, 기존 경로 내 최근접 질의처리 알고리즘은 계획된 경로를 가장 적게 벗어나면서 최근접점을 찾는 데에 초점을 맞추기 때문에 은행 A를 경유하는 점선으로 표시된 경로를 선택한다.



<그림 3> 공간제약 경로 내 최근접 질의개념

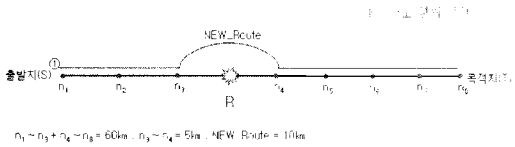
그러나 계획된 경로 중 사고가 발생하여 정체되는 구간(R구간)이 있다면, 이 구간(R구간)은 비용이 많이 소요되기 때문에 이를 제외시키고, R 구간을 지나지 않고 평소 운행하는 경로를 가장 적게 벗어나는 새로운 경로(실선으로 표시된 경로)를 선택하여야 한다. 또한, 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘의 설계 시, 공간제약 정도의 강약에 따른 경로 선택 연구가 필요하다. <그림 3>에서는 사용자가 해당 사고구간(R구간)을 경유하지 않고 다른 경로로 은행(POI)을 방문하였지만, 해당 구간에 반드시 방문해야할 POI가 존재하여 사용자가 해당 구간을 꼭 경유해야 하는 경우도 발생한다. 따라서 사용자의 선호도나 선택에 따라 공간제약 정도에 가중치를 두고, 최적 조건의 경로를 탐색하는 질의처리 알고리즘에 대한 연구가 필수적이다. 먼저, 사고정체구간(R)을 지나지 않는 최적의 새로운 경로를 구하기 위한 정의를 제시한다. 사용자로부터 계획한 길을 유지하고자 하는 정도에 대한 가중치(β)를 입력 받아, 최적 조건의 경로를 탐색하는 경로비용함수(route cost function)는 정의 1과 같다. 이 때, 계획한 길을 유지하고자 하는 정도에 대한 가중치 β값은 $0<\beta\leq 1$, 계획한길을 포함하지 않는 길에 대한 페널티의 가중치는 r 이다.

정의 1. 공간제약 구간을 지나지 않는 경로비용함수

$$CostDRoute = (1-\beta)\times IN\_IRRoute + \beta \times NEW\_Route + r \times OUT\_IRRoute$$

<표 1> 공간제약 구간을 지나지 않는 경로비용함수 용어

IN_IRRoute	계획한길을 포함하는 길의 길이
OUT_IRRoute	계획한길을 포함하지 않는 길의 길이
NEW_Route	계획한길을 벗어난 길의 길이
β	계획한 길을 유지하고자 하는 정도에 대한 가중치
r	계획한길을 포함하지 않는 길(OUT_IRRoute)에 대한 페널티의 가중치



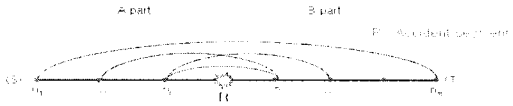
<그림 4> 최적조건의 경로비용함수 예

예를 들면, <그림 4>에서 n1에서 n8까지의 계획한 경로에서 n3 ~ n4구간에 사고가 발생하여 정체가 되었을 경우, 사고정체 구간(R)을 지나지 않는 ① 번 길을 구했을 때, 이때 계획한 길을 유지하고자 하는 정도에 대한 가중치(β)값이 0.9라 가정한다. 먼저, IN_IRRoute값 즉, 계획한 길을 포함하는 경로의 길이는 n1~n3 + n4~n8 = 60㎞이며, 계획한 길을 포함하지 않는 경로의 길이 OUT_IRRoute값은 n3~n4 = 5㎞이다. 또한, 계획한 길을 벗어난 경로의 길이 NEW_Route값을 10㎞라 할 때, CostDRoute값은 $(1-0.9)\times 60 + 0.9\times 10 + 1\times 5 = 62$이다.

아울러, CostDRoute (Detour Route) 비용을 구하기 위해 공간제약구간을 지나지 않고 평소 운행하는 경로를 가장 적게 벗어나는 최적의 새로운 경로를 구하기 위해, 모든 경로(Route)에 위와 같은 공식을 적용하여 가장 적은 값을 갖는 경로의 비용을 CostDRoute로 설정한다. 이때, 계획한길을 포함하지 않는 길에 대한 페널티의 가중치 r는 1로 설정한다. 정의 2는 CostDRoute 비용을 구하기 위해 공간제약구간을 지나지 않는 모든 경로에 대한 새로운 길을 탐색하는 방법에 대한 정의이다.

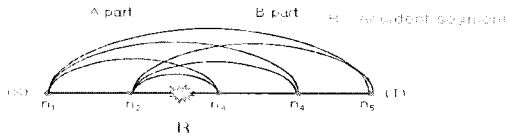
정의 2. 공간제약구간을 지나지 않는 새로운 길 탐색 방법 (가정, i<j)

n1에서 nm까지 계획한 경로에서 ni와 nj구간에서 사고가 발생하여 정체가 되었을 경우, 사고정체 구간을 시작으로 계획한 길을 포함하지 않는 구간(section)을 1부터 전체경로 범위(m)까지 확장시켜 새로운 길을 탐색한다<그림 5>.



<그림 5> 공간제약을 지나지 않는 모든 경로

이때, 각 구간별로 찾을 새로운 길의 개수는 1부터 노드가 적은 쪽 구간(A)의 노드 갯수 i만큼 증가한 뒤 일정해지고, 노드가 큰 쪽(B)의 노드 갯수 (m−i)이상의 구간에서는 1씩 감소한다. 예를 들면, <그림 6>과 같이 n1에서 n5까지의 계획한 경로에서 n2~n3구간에 사고가 발생하여 정체가 되었을 경우, <표 2>와 같이 R구간을 시작으로 계획한길을 포함하지 않는 구간을 1부터 전체경로 범위인 4까지 확장된다. 이때, 각 구간별로 찾을 새로운 길의 개수는 1부터 노드가 적은 쪽 구간(A)의 노드 갯수 2만큼 증가한 뒤 일정해지고, 노드가 큰 쪽(B)의 노드 갯수 3이상의 구간에서는 1씩 감소한다.



<그림 6> 새로운 길 탐색의 예

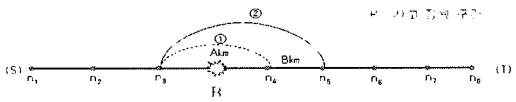
<표 2> 구간 별 새로운 길 개수

구간(section)	새로운 길 개수	새로운 길
1	1	n2~n3
2	2	n1~n3, n2~n4
3	2	n1~n4, n2~n5
4	1	n1~n5

한편 위와 같이 구간을 증가하면서 새로운 길을 탐색할 때, 효율적인 네트워크 확장을 위해 정의 3을 제시한다.

정의 3. 네트워크 확장 가지치기(Pruning) 방법

구간을 증가하면서 새로운 길을 탐색할 때, 정의2에 명시된 CostDRoute값이 이전 구간의 CostDRoute값보다 작아야 네트워크 확장을 하여 새로운 길을 찾는다. 그렇지 않으면, 네트워크 확장을 멈추고 가지치기 한다.

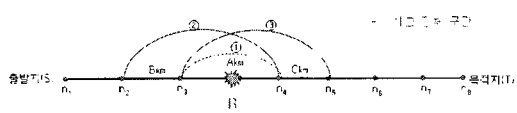


<그림 7> 네트워크 확장의 예

예를 들면, <그림 7>과 같이 n1에서 n8까지의 계획한 경로에서 n3~n4구간에 사고가 발생하여 정체가 되었을 경우, 구간이 1인 경우 찾은 경로①의 CostDRoute값보다 ②번 길의 CostDRoute값이 작아야 네트워크 확장을 하여 새로운 길을 찾는다. 따라서, ①CostDRoute값 > ②CostDRoute 공식을 이용하면 ②NEW_Route < A-B 의 공식을 도출해낸다. 그러므로 ②번 길은 A-B값보다 큰 경우가 되면 네트워크 확장을 멈춘다. 정의 4는 구간별 새로운 경로 간의 확장하는 순서에 대한 정의이다.

정의 4. 구간별 새로운 경로 간의 확장순서

같은 구간내의 여러 새로운 경로 중에서 길이가 짧은 쪽의 구간을 먼저 네트워크 확장을 하여 새로운 경로를 찾는다.



<그림 8> 구간별 새로운 길 간의 확장순서

<그림 8>을 통해 예를 들면, 만약, B의 길이가 C보다 작다면 B는 C에 비해 계획한 길을 포함하지 않는 경로에 대한 페널티값(r × OUT_IRRoute)이 작으므로 전체 CostDRoute값도 작기 때문에 n2에서 n4까지인 ②번 길을 네트워크 확장을 하여 새로운 경로를 찾는다.

정의 5는 사용자가 우회 경로가 없는 경우 또는 사고정체구간에 반드시 방문해야할 POI가 존재하여 해당 구간을 꼭 경유해야 하는 경우를 위한 정의이다. 아울러, 가중치 δ 값은 실험을 통하여 계산하였으며, 공간제약의 유형에 따라 표 4와 같이 모델링 할 수 있다.

정의 5. 공간제약 구간을 지날 때를 고려한 최소 경로 비용함수

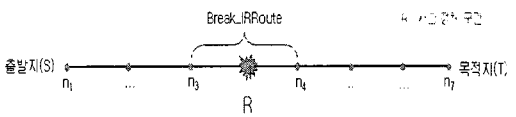
CostSCIRNN = MIN(CostDRoute, $(1-\beta) \times$ IN_IRRoute+ $\delta \times$ Break_IRRoute)

<표 3> 공간제약 구간을 지날 때의 경로비용함수 용어

IN_IRRoute	계획한길을 포함하는 길의 길이
Break_IRRoute	계획한길 중 사고정체구간
δ	계획한길 중 사고정체구간을 지날 때에 페널티의 가중치

<표 4> 공간 제약의 유형에 따른 가중치 δ의 모델링

가중치	공간 제약의 유형
$\delta = 1$	교통이 원활하게 소통되는 경우
$1 < \delta \leq 10$	도로 공사 또는 소형의 교통 사고 등으로 인해 약간의 교통 체증이 있을 경우
$10 \leq \delta < 1000$	대형의 교통 사고 등으로 인해 교통 체증이 매우 심할 경우
$1000 \leq \delta$	천재지변으로 인해 공간 제약 구간을 지나지 못하는 경우



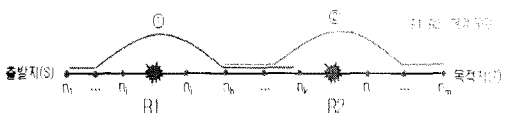
<그림 9> 공간제약 구간을 지날 경우의 예

<그림 9>와 같이, 계획한길 중 사고정체구간을 지날 때의 페널티의 가중치 δ 값은 사고정체구간(R)의 상황에 도로의 상황(1차선 도로, 2차선 도로) 및 공간 제약의 유형(교통 상황) 등에 따라 부여한다.

정의 6은 두 개의 제약구간이 있을 경우, 각각의 사고 정체 구간을 고려하여 계획한 경로를 두 개로 분할하여 새로운 길을 찾는 방법을 나타낸다.

정의 6. 두 개의 사고 정체 구간이 있을 경우의 경로 분할 방법

계획한 경로가 노드가 n1에서 nm 으로 구성될 때, 첫 번째 제약 구간(R1)을 포함한 양 노드를 ni, nj라 하고 두 번째 제약 구간(R2)를 포함한 양 노드가 nk, nl이라 하자. 이 때, 출발지(n1)에서 nk까지의 구간에서, 앞서 언급한 5개의 정의를 이용하여 최소 비용을 갖는 경로를 찾는다. i) R1 지나지 않는 경우, 첫 번째 제약구간을 우회하는 경로가 만나는 노드를 nh라 할 때, nh에서 목적지 nm까지 R2를 위한 우회경로를 찾는다. ii) R1을 지나는 경우, nj에서 목적지 nm까지 R2를 위한 우회경로를 찾는다. 이 때의 비용함수를 CostDoubleDivide라 한다.

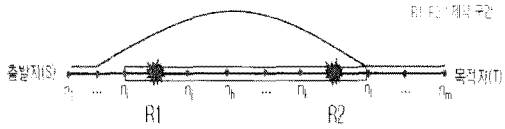


<그림 10> 두 개의 제약구간이 있을 경우 경로 분할 방법의 예

정의 7은 두 개의 제약구간이 있을 경우, 두 개의 사고 정체 구간 사이의 경로를 하나의 공간제약 구간으로 취급하여 처리하는 방법을 나타낸다.

정의 7. 두 개의 사고 정체 구간이 있을 경우의 공간제약 병합 방법

두 개의 제약구간(R1, R2)이 있을 경우, 두 개의 경로 사이를 하나의 공간제약으로 처리하여 새로운 길을 찾는다. 즉, 노드 n1~ni 에서 노드 nl~nm 까지 우회경로를 찾는다. 이 때의 비용함수를 CostOneDivide라 한다.



<그림 11> 두 개의 제약구간이 있을 경우 공간제약 병합 방법의 예

정의 6과 정의 7에서 구한 경로 비용함수 중에서 적은 비용함수를 갖는 경로를 선택하는 최소 비용함수는 정의 8과 같다.

정의 8. 두 개의 사고 정체 구간이 있을 경우의 최소 비용함수

CostTwoSCIRNN = Min(CostDoubleDivide, CostOneDivide)

셋 이상의 사고 정체 구간이 있을 경우를 정의 6, 7 ,8을 이용하여 처리할 수 있다. 앞서 언급한 정의들을 바탕으로 설계한 공간제약 경로 내 최근접 질의처리 알고리즘은 <그림 12>와 같다. 먼저 공간제약 구간(R구간)의 노드 쌍을 설정한다. 둘째, 공간제약구간을 지나지 않고 평소 운행하는 경로를 가장 적게 벗어나는 최적의 새로운 경로 SCRoute를 구한다. 이때, 정의 1에서 정의한 최적조건의 경로비용함수를 이용하여 모든 경로에 대해 정의 2, 3, 4를 이용하여 CostSCRoute를 계산한다. 이때, 주어진 경로를 공간제약 구간을 기준으로 왼쪽부분, 오른쪽부분으로 나누어 정의 2에서 정의한 구간별 새로운 경로 탐색 공식을 적용한다. 셋째, 최소의 경로비용함수를 가지는 경로를 설정한 다음, 새로운 경로내의 노드를 B+-트리에서 검색하여 각 노드마다 가장 가까운 POI를 테이블에 저장한다. 넷째, β가 $1>\beta>0$인 경우에는 공간제약 구간을 포함하지 않는 경로인 SCRoute의 SCPOI를 반환한다. 마지막으로, β가 0인 경우에는 공간제약을 고려하지 않는 의미이기 때문에 기존 경로 내 최근접 경로의 POI를 반환한다.

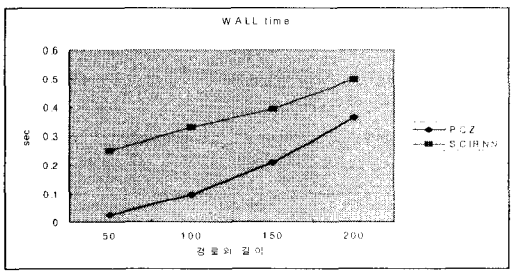
```
SCIRNN(BaseRoute, POI, Link, β, δ)
//BaseRoute: 기준경로, POI: 검색대상, Link:공간제약 구간, β:공간제약 가중치 , δ:사고정체구간을 지날 때에 페널티의 가중치
1. Cost=infinite;
2. LeftRoute=(BaseRoute, Link);
3. RightRoute=(BaseRoute, Link);
4. while (LeftRoute!=NULL)
   4.1 num=0;
   4.2 for(i=0; i<sizeof(LeftRoute); i++)
       4.2.1 NEW_Route=(BaseRoute[num], BaseRoute[num+i]);
       4.2.2 Cost =(1-β)×IN_IRRoute + β×NEW_Route + r×OUT_IRRoute;
       4.2.3 if(CostSCRoute>Cost)
             4.2.3.1 CostSCRoute=Cost;
             4.2.3.2 SCRoute =route;
   4.3 num++;
6. num=0;
7. while (num < sizeof(RightRoute))
   7.1 for(i=0; i<sizeof(LeftRoute); i++)
       7.1.1 NEW_Route=(BaseRoute[num], BaseRoute[num+i]);
       7.1.2 Cost =(1-β)×IN_IRRoute + β×NEW_Route + r×OUT_IRRoute;
       7.1.3 if(CostSCRoute>Cost)
             7.1.3.1 CostSCRoute=Cost;
             7.1.3.2 SCRoute =route;
   7.2 num++;
8. num=0;
9. while (RightRoute!=NULL)
   9.1 for(i=sizeof(RightRoute); i>0); i--)
       9.1.1 NEW_Route=(BaseRoute[num],BaseRoute[num+i]);
       9.1.2 Cost =(1-β)×IN_IRRoute + β×NEW_Route + r×OUT_IRRoute;
       9.1.3 if(CostSCRoute>Cost)
             9.1.3.1 CostSCRoute=Cost;
             9.1.3.2 SCRoute =route;
   9.2 num++;
10. if( β==0 )
   10.1 POI=IRNN(BaseRoute);
   10.2 return POI;
11. else
    11.1 SCPOI=findSpaceConstrainedPOI(SCRoute);
    11.2 return SCPOI;
End SCIRNN
```

<그림 12> 공간제약 경로 내 최근접 질의처리 알고리즘

이러한 공간 제약 경로 내 최근접 질의처리 알고리즘은 도로 내에서 발생할 수 있는 병목 현상을 실시간으로 고려하여 좀 더 효율적인 경로설정을 할 수 있는 장점이 있다. 그러나 기존 경로 내 최근접 질의처리 알고리즘에서 설정된 경로에서 병목현상이 발생된 구간을 제외하고 최적의 새로운 길을 설정해야하는 오버헤드가 발생한다.

4. 성능 평가

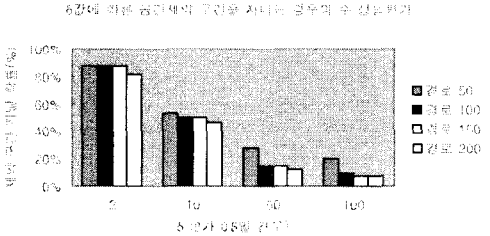
본 논문에서 제시한 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘과 기존의 연구인 경로 내 최근접 질의처리 알고리즘에서 가장 좋은 성능을 갖는 PCZ (Precomputed Zone) 방법과 성능평가를 수행한다. 성능평가에서 사용한 공간 네트워크 데이터는 Brinkhoff가 제안한 알고리즘에서 제공하는 공간 네트워크 데이터중의 하나인 샌프란시스코 만 데이터를 사용하였다[11]. 샌프란시스코 만 공간 네트워크의 에지의 수는 약 220,000개이며, 노드의 수는 약 170,000개를 가지고 있다. 아울러, RunTime21 알고리즘을 사용하여 임의로 생성한 여러 타입을 가진 10846개의 POI를 사용하였다[12]. 성능평가 항목은 성능평가 항목은 경로의 길이를 50, 100, 150, 200개로 증가하면서 각각 전체 응답시간(WALL_Time)을 측정한다.



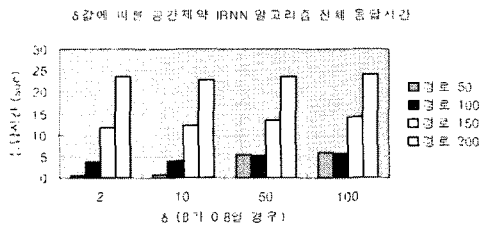
<그림 13> 공간제약 경로 내 최근접 알고리즘 전체 응답시간

<그림 13>에서 보듯이 공간 제약 경로 내 최근접 방법은 PCZ 방법에 비해 좋지 않은 성능을 보인다. 이는 공간 제약 경로 내 최근접 방법이 제시한 정의를 이용하여 설정된 경로에서 병목현상이 발생된 구간을 제외하고 최적의 새로운 길을 설정해야하는 오버헤드가 발생하기 때문이다. 하지만 경로의 길이가 증가할수록 응답시간의 폭이 좁혀지는 것을 알 수 있다. 이는 제시한 정의 4를 이용하여 네트워크 확장 시 가지치기 기법을 적용하기 때문에, 경로가 증가함에도 불구하고 탐색시간은 크게 증가하지 않기 때문이다. 공간 제약 경로 내 최근접 방법은 이러한 오버헤드가 발생하지만, 도로 내에서 발생할 수 있는 병목현상을 실시간으로 고려하여 설정된 경로를 적게 벗어나면서 지나는 경로설정을 할 수 있는 장점이 있다.

아울러, 정의 5에서 제안한 공간제약 구간을 지날 때의 경로비용함수에서 계획한길 중 사고정체구간을 지날 때에 페널티의 가중치 δ 값에 따라 공간제약 구간을 지나는 길을 선택하는 경우의 수와 전체 응답시간을 측정한다. <그림 14>에서 δ 값이 증가함에 따라, 공간제약 구간의 페널티의 가중치 값이 커지므로 공간제약 구간을 지나는 경우의 비용이 커지게 된다. 이는 δ 값이 1 이상 10 이하일 경우에는 약 50%에서 90%가량 사고 정체 구간을 지나게 되며, δ 값이 10 이상 100 이하일 경우에는 약 10%에서 50%가량 사고 정체 구간을 지나게 된다. 따라서 공간 제약의 유형에 따라 δ 값이 증가할수록 공간 제약구간을 지나는 경우의 수가 현저히 줄어드는 것을 알 수 있다. 또한 <그림 15>에서 δ 값이 증가할수록 공간제약 구간을 지나지 않는 새로운 경로를 구하는데 시간이 더욱 소모되므로 전체 응답시간이 커지는 것을 알 수 있다.

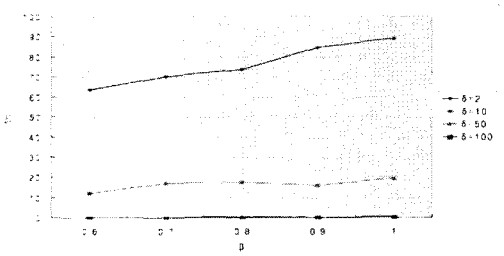


<그림 14> δ 값에 따른 공간제약 구간을 지나는 경우의 수 성능평가

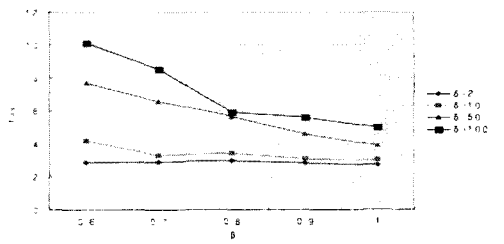


<그림 15> δ값에 따른 공간제약 경로 내 최근접 알고리즘 전체 응답시간

사용자로부터 입력받은 계획한길을 유지하고자하는 정도에 대한 가중치(β)에 따라 공간제약 구간을 지나는 경로를 선택하는 경우의 수와 전체 응답시간을 측정한다. <그림 16>에서 β값이 증가함에 따라, 즉, 사용자로부터 계획한길을 유지하고자하는 정도(β)가 커질수록, 공간제약 구간을 지나는 경우를 선택하는 것을 볼 수 있다. 그러므로 <그림 17>에서 나타나는 것과 같이 β값이 증가할수록 공간제약 구간을 지나는 경로를 선택하므로 새로운 경로를 구하는데 시간이 줄어들어 전체 응답시간이 감소하는 것을 볼 수 있다.



<그림 16> β값에 따른 공간제약 구간을 지나는 경우의 수 성능평가



<그림 17> β값에 따른 공간제약 경로 내 최근접 알고리즘 전체 응답시간

마지막으로, 표 5는 제시한 정의들의 사용 유무에 따라 우회 경로를 찾는 횟수를 나타낸다. 표에서 나타내는 바와 같이, 본 논문에서 제시한 모든 정의를 이용한 알고리즘이 효율적인 비용함수와 가지치기 기법을 이용한 정의를 제시함으로써 우회 경로를 찾는 경우의 수가 현격히 줄어든다.

<표 5> 우회 경로를 찾는 횟수 (β = 0.8, δ = 100)

사용한 정의 / 경로 길이	정의를 사용하지 않은 경우	정의 1, 5만을 사용한 경우	모든 정의를 이용한 경우
50	452.350	255.341	3.383
100	1988.193	726.155	8.266
150	4373.855	1379.391	8.840
200	6766.609	1933.471	11.398

5. 결 론

본 논문에서는 공간 네트워크상의 공간제약을 고려한 경로기반 질의처리 알고리즘을 제안하였다. 공간 네트워크 데이터베이스에서는 이동객체가 공간 네트워크상에서만 이동하기 때문에 경로에 기반을 둔 질의의 유용성이 매우 증대되고 있다. 따라서 경로-기반 질의에 대한 효율적인 질의처리 알고리즘 연구가 필수적이다. 대표적인 경로-기반 질의로써 경로 내 최근접 질의가 제시되었으며[8], 이는 경로를 가장 적게 벗어나면서 최근접점을 찾는 데에 초점을 맞추고 있다. 그러나 경로 내 최근접 질의처리 알고리즘은 공간 제약을 고려하지 못하기 때문에 병목현상이 발생한 도로 내의 실시간 상황을 반영하지 못하는 문제점이 존재한다. 이러한 문제점을 극복하기 위해서 본 논문에서는 기존 경로 내 최근접 질의처리 알고리즘에서 가장 좋은 성능을 보인 PCZ 방법을 기반으로 하여, 공간제약을 고려한 경로 내 최근접 질의처리 알고리즘을 개발하였다. 또한, 기존의 방법과 비교성능평가를 통하여 기존 경로 내 최근접 방법에 비해 탐색시간이 증가하지만, 제안하는 공간 제약 경로 내 최근접 방법은 도로 내에서 발생할 수 있는 병목현상을 실시간으로 고려하여 설정된 경로를 적

게 벗어나면서 경로설정을 할 수 있으므로 텔레매틱스(telematics), CNS(Car Navigation System), 자동항법장치, L-commerce 등의 응용에 적용 가능하다.

향후 연구로는 제안하는 공간제약의 알고리즘에 시간제약을 추가하여 실세계에서 적용 가능한 시공간 제약을 고려한 경로 내 최근접 질의처리 알고리즘을 개발하는 것이다.

참 고 문 헌

1. P. Sistla, O. Wolfson, S. Chamberlain, and S.Dao, "Modeling and Querying Moving Objects", Proc. of IEEE ICDE, 1997, pp. 422-432.
2. Z. Song, and N. Roussopoulos, "K-Nearest neighbor Search for Moving Query Point", Proc. of SSTD, 2001, pp. 79-96.
3. Y. Tao, and D. Papadias, "Time Parameterized Queries in Spatio-Temporal Databases", Proc. of ACM SIGMOD, 2002, pp. 334-345.
4. Y. Tao, D. Papadias, and Q. Shen, "Continuous Nearest Neighbor Search", Proc. of VLDB, 2002, pp. 287-298.
5. Y. Hsueh, R. Zimmermann, and M. Yang, "Approximate Continuous K Nearest Neighbor Queries for Continuous Moving Objects with Pre-defined Paths", Proc. of COMOGIS, 2005, pp. 270-279.
6. H. Jung, S. Kang, M. Song, S. Im, J. Kim, and C. Hwang, "Towards Real-Time Processing of Monitoring Continuous k-Nearest Neighbor Queries", Proc. of ISPA, 2006, pp. 11-20.
7. J. Feng, L. Wu, Y. Zhu, N. Mukai, and T. Watanabe, "Continuous k-Nearest Neighbor Search Under Mobile Environment", Proc. of WAIM, 2007, pp. 566-573.
8. S. Shekhar, and J.S. Yoo, "Processing In-Route Nearest Neighbor Queries: A Comparison of Alternative Approaches", Proc. of ACM GIS, 2003, pp. 9-16.
9. M.R. Kolahdouzan, and C. Shahabi, "Continuous K Nearest Neighbor Queries in Spatial Network Databases", Proc. of STDBM, 2004, pp. 33-40.
10. K. Mouratidis, M. L. Yiu, D. Papadias, N. Mamoulis, "Continuous Nearest Neighbor Monitoring in Road Networks", Proc. of VLDB, 2006, pp. 43-54.
11. http://www.fh-oow.de/institute/ iapg/personen/ brinkhoff/generator/
12. T. Brinkhoff, "A Framework for Generating Network - Based Moving Objects", GeoInformatica, 2002, pp. 153-180.

김용기
2002년 전북대학교 컴퓨터공학과(공학사)
2005년 전북대학교 대학원 컴퓨터공학과(공학석사)
2006년~현재 전북대학교 대학원 컴퓨터공학과 박사과정
관심분야 : 공간 네트워크 데이터베이스, 질의처리 알고리즘, 센서 네트워크, 정보 보호
e-mail : ykkim@dblab.chonbuk.ac.kr

김아름
2008년 전북대학교 컴퓨터공학과(공학사)
2008년~현재 전북대학교 대학원 컴퓨터공학과 석사과정
관심분야 : 공간 데이터베이스, 질의처리 알고리즘, 센서 네트워크
e-mail : arkim@dblab.chonbuk.co.kr

장재우
1984년 서울대학교 전자계산기공학과(공학사)
1986년 한국과학기술원 전산학과(공학석사)
1991년 한국과학기술원 전산학과(공학박사)
1996년~1997년 Univ. of Minnesota, Visiting Scholar

2003년~2004년 Penn State Univ., Visiting Scholar
1991년~현재 전북대학교 컴퓨터공학과 교수
관심분야 : 공간 네트워크 데이터베이스, 상황인식, 하부저장구조
e-mail : jwchang@chonbuk.ac.kr