시점 질의를 위한 선택율 추정
(Selectivity Estimation for Timestamp Queries)

신 병 철 † 이 종 연 ††
(Byoung-Cheol Shin) (Jong-Yun Lee)

요 약 최근 시간에 따른 대량의 공간 객체들의 효과적인 저장과 처리의 필요성이 요구되면서 시공간 데이타베이스에 대한 필요성이 증가하였다. 이러한 시공간 데이타베이스에서 효과적인 질의 처리를 위하여 여러 가지 질의 최적화 기법이 연구되었고 그중 질의의 근사적인 결과를 계산하는 선택도 추정 기법이 활발하게 연구되었다. 선택도 추정 기법에는 샘플링 기반 기법, 히스토그램 기반 기법, 웨이블릿 기반 기법 등이 있고 그중 히스토그램 기법은 현재 상용 데이타베이스에서 널리 사용되고 있다. 하지만 지금까지의 시공간 질의 최적화 연구는 이동 객체의 미래 위치에 대한 선택도 추정에 치중되어 왔다. 본 논문에서는 과거의 시공간 데이타의 질의 최적화를 위하여 새로운 히스토그램인 T-Minskew의 구축 방법을 제안한다. 또한 T-Minskew를 이용한 효과적인 선택도 추정 기법을 제안하고 임계치 기법을 이용한 히스토그램의 효과적인 유지 기법을 통해 잦은 히스토그램 재구축을 방지하고 작은 추정 오류율을 유지하는 방법을 제안한다.

키워드 : 시공간 데이타베이스, 질의 최적화, 선택도 추정, 히스토그램

Abstract Recently there is a need to store and process enormous spatial data in spatio-temporal databases. For effective query processing in spatio-temporal databases, selectivity estimation in query optimization techniques, which approximate query results when the precise answer is not necessary or early feedback is helpful, has been studied. There have been selectivity estimation techniques such as sampling-based techniques, histogram-based techniques, and wavelet-based techniques. However, existing techniques in spatio-temporal databases focused on selectivity estimation for future extent of moving objects. In this paper, we construct a new histogram, named T-Minskew, for query optimization of past spatio-temporal data. We also propose an effective selectivity estimation method using T-Minskew histogram and effective histogram maintenance technique to prevent frequent histogram reconstruction using threshold.

Key words : Spatio-temporal Databases, Query Optimization, Selectivity Estimation, Histogram

1. 서 론

최근 시공간 데이타에 대한 관심이 늘어나면서 이를 사용하는 여러 애플리케이션이 개발되었고 시공간 데이타를 효과적으로 저장, 관리 할 수 있는 시공간 DBMS의 필요성이 증가하였다. 이러한 시공간 DBMS에는 크게 시간에 따라 급격히 변화하는 이동 공간객체에 대한 영역(time-series data)과 장시간을 두고 서서히 변화하는 이력 공간객체에 대한 영역(sequence data)이 있다. 이동 객체에 대한 질의는 현재의 객체 정보에 기반하여 미래의 어느 시점에서 이 객체의 공간 정보가 어떻게 변화될 것인가를 예측하는 것이다. 예를 들면, "30분 동안 질의 영역 a와 겹치는 모든 이동 객체들을 찾아라." 와 같은 질의를 들 수 있다. 이력 공간 질의는 과거의 어떤 시점에서 질의의 공간 영역과 겹치는 객체들을 찾아내는 것이다. 예를 들면, "시간 t에서 질의 영역 A와 겹치는 객체들을 찾아라."와 같은 질의를 들 수 있다.

시공간 데이타베이스에서 사용되는 데이타들은 시간과 공간에 따른 데이타이므로 기본적으로 그 크기가 매우 큰 특징을 가지고 있다. 따라서 시공간 질의의 효과적인 처리를 위해 색인, 질의 처리, 질의 최적화와 같은 기법이 연구되었다. 시공간 데이타베이스는 시간과 공간의 결합이므로 두 분야의 결합에서 확장되어 접근되어

· 이 논문은 2005년도 충북대학교 학술지원사업의 연구비 지원에 의하여 연구되었음
† 정 회 원 : 충북대학교 컴퓨터교육과
suemirr@nate.com
†† 종신회원 : 충북대학교 컴퓨터교육과 교수
jongyun@chungbuk.ac.kr
(Corresponding author임)
논문접수 : 2005년 6월 29일
심사완료 : 2005년 12월 6일

져 왔다. 그에 대한 결과로 [1]이나 [2]와 같은 색인이나 질의 처리에 관한 연구가 활발하게 진행 되었다. 또한 시공간 데이타베이스에서 질의 최적화를 위해 중요하게 다루는 선택도 추정은 기존의 공간 선택도 추정의 연구 [3-7]에서 시공간 데이타베이스 선택도 추정 [8-11]로 확장되었다. 하지만 지금까지의 시공간 데이타베이스에서의 선택도 추정 연구는 이동 객체에 대한 연구로 치중되어 왔었다.

본 논문에서는 Minskew 히스토그램[3]을 확장하여 이력 공간객체의 선택도 추정이 가능하게 하는 T-Minskew 히스토그램의 구축 방법과 효과적인 히스토그램의 유지기법을 제안한다. 본 논문에서 제안하는 T-Minskew 히스토그램의 특징은 다음과 같다. (i) Minskew 히스토그램을 특정 timestamp에 구축하여 재구축이 일어나기 이전의 timestamp까지 유지한다. (ii) 히스토그램이 재구축 될 때 삭제되는 히스토그램을 저장하고 현재 시간을 기준으로 새로운 히스토그램을 생성한다. (iii) 너무 많은 히스토그램 재구축을 방지하기 위해 히스토그램 임계치를 두어 재구축 횟수를 줄이면서 만족스러운 선택도 추정율을 유지하도록 한다. 본 논문에서 제안하는 히스토그램은 기본적으로 2차원 객체의 특정 시점에 대한 질의를 주로 다룬다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 기술하고 3장에서는 본 논문에서 제안하는 T-Minskew 히스토그램 구축과 이를 이용한 선택도 추정 기법을 기술한다. 4장에서는 제안한 기술을 위한 실험 평가를 하고 5장에서 결론을 정리한다.

2. 관련 연구

2.1 기존의 공간 선택도 추정

[1]에서는 공간 선택도 추정을 위한 히스토그램인 Minskew를 제안하고 있다. Minskew 히스토그램은 편중된 객체들의 버킷 분할을 통한 분배를 통하여 균일하게 만드는데 있다. 버킷 분할의 기준은 객체들의 편중도에 기반하고 있으며 분할 가능한 경우의 수중에 분할되는 두 버킷의 편중도의 가중치가 가장 낮은 분할을 선택하여 이진 공간 분할(BSP)을 한다. 각 축별로 분할 가능한 모든 경우를 검사하기에는 많은 복잡도를 가질 수 있으므로 분할 적합 축을 우선 선택한 후에 그 축을 기준으로 분할을 하는 것으로 분할 알고리즘의 효과를 높이고 있다. $B_i.num$은 i번째 버킷에 포함되는 점 객체의 수 또는 사각형 객체의 중심점이 버킷의 영역에 포함되는 수를 저장한다. C는 셀을 가리키며 $C_i.den$은 i셀에 겹치는 객체수를 저장한다. Avg(den)은 셀의 평균 den수를 말하고 |C|는 셀의 수를 가리킨다. 이 때 버킷의 편중도 $B_i.skew$는 식 (1)과 같이 표현하며 전체 편중도의 가중치는 식 (2)로 표현한다. 최종 분할 경우의 선택은 가중치가 가장 작은 것으로 한다.

$$B_i.skew = \frac{1}{|C|}\sum_{i=1}^{number\ of\ cell\ in\ Bi}(C_i.den - Avg(den))^2 \quad (1)$$

$$Minskew = \sum_{I=1}^{n}(B_i.num\ x\ B_i.skew) \quad (2)$$

그림 1은 Minskew 히스토그램에서 버킷 분할의 예를 보이고 있다. 각각의 사각형은 셀을 나타내며 그림 1(a)에는 9개의 셀을 포함하는 버킷이 하나인 상태이다. Dim을 각 차원이라 할 때 Dim 1의 분산은 6이고 Dim 2의 분산은 4임을 알 수 있다. 이것은 분산이 높은 축이 객체의 편중도가 높다는 것이고, 따라서 이러한 편중도를 낮추기 위하여 편중도가 높은 축을 기준으로 공간을 분할하는 것이 좋다는 것을 미리 알 수 있다. 그림 1(b)는 선택된 분할 축을 기준으로 가능한 분할 경우의 수에서 식 (1)을 이용하여 분할된 두 버킷의 편중도를 구하고 식 (2)를 이용하여 최종 가중치를 구한 뒤 가중치가 가장 작은 분할 선을 기준으로 공간을 두개의 버킷으로 분할한 것이다. 식 (2)를 이용하여 각 분할 가능한 경우의 가중치를 구하면 첫 번째 경우의 가중치는 28.14가 나오며 두 번째 경우는 37.38이 나온다. 따라서 첫 번째 분할이 편중도를 낮출 수 있는 방법임을 알 수 있고 그림 1(b)와 같다.

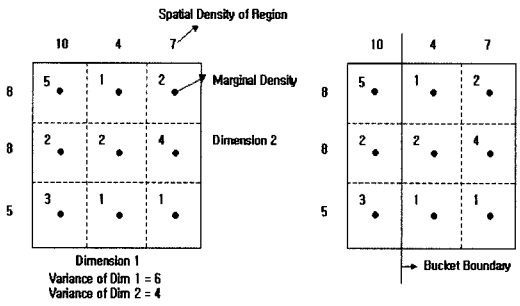


그림 1 Minskew 히스토그램의 공간 분할

2.2 기존의 시공간 선택도 추정

최초의 시공간 선택도 추정 기술인 [8]에서는 이동하는 객체가 어떤 시간동안 고정된 질의 영역에 겹칠 수 있는지 여부에 초점을 맞추고 있다. 선택도 추정을 위하여 우선 전체 공간을 Minskew 알고리즘을 사용하여 버킷으로 분할하고 각 버킷에 대한 선택도 추정을 한 후 이를 모두 합하여 전체 공간에 대한 선택도를 추정하는 기술을 제시하고 있다. 2차원 공간 선택도 추정은 각 차원 별로 질의와 이동 객체를 사상시켜 1차원 환경

에서 선택도를 추정한 뒤 각 차원별로 구해진 선택도를 곱하여 2차원 공간에서의 선택도 추정을 한다. 따라서 [8]에서 제시하는 선택도 추정 기술은 2차원 공간에서는 겹치지 않는 객체가 1차원 공간으로 사상하여 겹칠 수 있는 가능성을 가지기 때문에 다차원 공간에서의 선택도 추정에 좋지 못한 성능을 보인다. 만약 객체의 속도가 [0, V]에 존재하고, 전체 공간 [0, U]내에 균일하게 분포되어 있다면 질의는 T 시간동안 질의의 공간 영역과 겹쳐지는 객체들을 찾아내는 것이다. 이 때 실제적인 선택도 *Sel*은 [9]에서 다음과 같이 제시하고 있다.

$$Sel = \frac{VT}{U^2}[(q_{R.xmax} - q_{R.xmin}) + (q_{R.ymax} - q_{R.ymin})] + \frac{(q_{R.xmax} - q_{R.xmin})(q_{R.ymax} - q_{R.ymin})}{U^2} \quad (3)$$

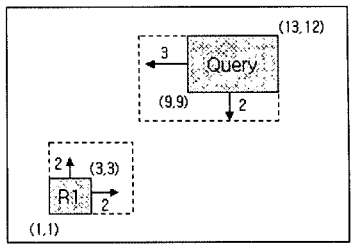
이 때 [8]에서 제시한 방법으로 선택도 추정을 한다면 각 차원 별로 식 (4)와 같은 선택도 추정이 가능하고 이를 2차원으로 생각하여 두 차원을 곱하면 식 (5)와 같아지므로 선택도가 원래보다 $\frac{V^2T^2}{4U^2}$만큼 높게 추정이 됨을 알 수 있다.

$$Seli = \frac{q_{Ri+} - q_{Ri-}}{U} + \frac{VT}{2U} \quad (4)$$

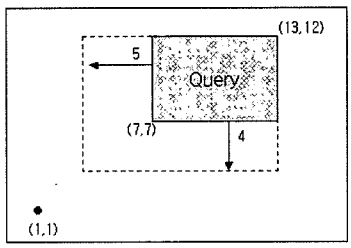
$$Sel = \frac{VT}{U^2}[(q_{Rxmax} - q_{Rxmin}) + (q_{Rymax} - q_{Rymin})] + \frac{(q_{Rxmax} - q_{Rxmin})(q_{Rymax} - q_{Rymin})}{U^2} + \frac{V^2T^2}{4U^2} \quad (5)$$

[9]에서는 [8]에서 제시한 이동 객체 표현을 2차원 공간으로 확장시켰다. 그리고 [8]에서의 초과된 선택도 추정에 대하여 2차원 공간을 1차원 공간으로 사상시키는 것을 회피함에 따라 선택도가 높아지는 현상을 해결하였다. 어떤 객체의 현재 시간 0에서의 위치를 (x, y)라 하고 각 축에 따른 속도를 (v_x, v_y)라 할 때 [9]는 Minskew 기술을 사용하여 4차원 점 (x, y, v_x, v_y)을 표현할 수 있는 4차원 히스토그램을 생성하였다. 이 때 히스토그램의 각 버킷은 공간 영역 MBR과 속도 MBR인 VMBR을 가지며 버킷안의 객체들은 영역 MBR과 VMBR안에서 균일하게 분포되도록 히스토그램을 구축한다. 이동하는 사각형 객체와 질의에 대해서는 그림 2(b)와 같이 객체가 가지는 이동 정보와 공간 정보를 질의에 포함시킴으로써 선택도 추정을 보다 쉽게 할 수 있게 하였다. 히스토그램의 재구축은 전체 데이타집합에서의 갱신률이 정해진 한계 값을 초과할 경우에만 일어나므로 재구축 빈도가 [8]에 비하여 줄어들었다.

[10]에서는 공간-시간 그래프를 Hough 변환을 통한 속도-절편 그래프로 변형한 뒤 MinSkew 기술을 적용하여 이동 객체에 대한 선택도를 추정한다. 공간-시간 그래프에서 점의 이동에 따른 궤적은 속도-절편 그래프

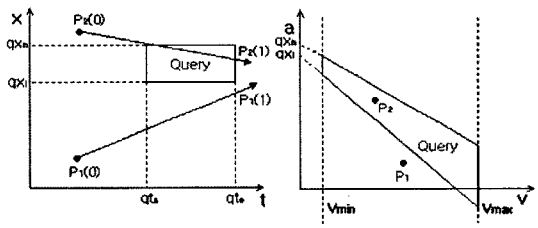


(a) 이동 객체와 질의



(b) 이동 객체의 간소화

그림 2 이동 객체의 간소화 과정



(a) 공간-시간 그래프 (b) 속도-절편 그래프

그림 3 Hough 변환을 사용한 그래프 변환

에서 하나의 점으로 표현될 수 있으며 질의 사각형은 어떤 공간 영역으로 표현된다. 만약 속도-절편 그래프에서 점이 질의 영역에 포함된다면 공간-시간 그래프에서 그 점에 해당하는 점의 궤적인 선은 질의 사각형을 지나간다고 할 수 있다. 그림 3은 공간-시간 그래프와 속도-절편 그래프의 예를 보인다. [10]에서의 선택도 추정은 점 객체들이 존재하는 전체 공간을 절편-속도 그래프에서 MinSkew 알고리즘을 사용하여 버킷들로 분해한 뒤 질의 영역과 겹치는 버킷들의 영역을 계산함으로써 구할 수 있다.

[11]에서는 클러스터링 기술을 기반으로 한 버킷 분할 히스토그램을 제시하고 있다. 이전까지의 이동객체를 위한 선택도 추정의 연구에서 대부분 Minskew 히스토그램을 확장한데에 반하여 [11]에서는 더 효율적인 공간 버킷 분할을 위해 클러스터링 기술을 이용했다. 클러스터링을 이용한 버킷 분할은 비슷한 성질을 가지는 객체는 가까운 거리에 존재하는 성질을 이용하여 버킷을 분

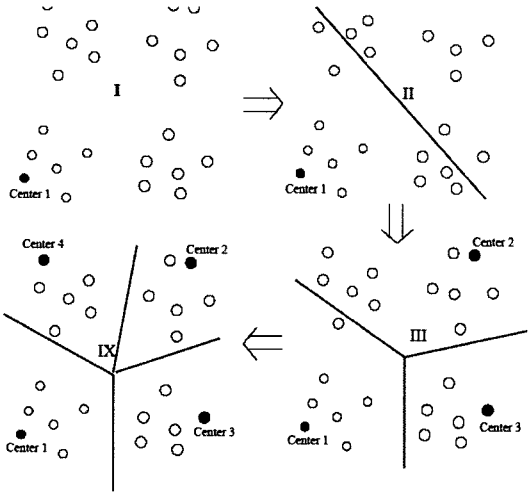


그림 4 Gonzalez Clustering

해한다. 다시 말해 두 객체가 가깝다면 초기 위치와 속도, 객체 크기가 비슷할 수 있고 이렇게 비슷한 성질을 가지는 객체들을 한 버킷에 둠으로써 버킷내의 객체들이 균일함이 보장된다는 것이다. 이동 객체간의 거리 계산을 위해 유클리드 계산법을 확장하여 적용하였으며 기본적인 클러스터링 방법을 위해 그림 4와 같은 방법을 사용하였다. 히스토그램의 정제를 위해 이미 구축된 히스토그램의 버킷에서 어떤 객체를 뽑아 다른 버킷에 넣어봄으로써 더 나은 히스토그램이 되게 하는 히스토그램 유지 기법도 제안하고 있다.

3. T-Minskew 히스토그램

이 장에서는 본 논문에서 제안하는 T-Minskew 히스토그램을 위해 개괄적인 히스토그램과 히스토그램에서 사용되는 셀과 버킷을 정리한다. 아울러 T-Minskew를 이용한 이력 공간 질의의 선택도 추정 방법을 제안한다. T-Minskew 히스토그램은 2차원 공간 구조에 시간을 고려한 히스토그램이다. 히스토그램에서 시간 정보를 가지고 있는 것은 timestamp에 따른 각각의 히스토그램 밖에 없고 히스토그램내의 버킷은 묵시적으로 공간 정보를 가지면서 히스토그램의 timestamp내에 존재한다는 것을 보장한다. 그림 5는 timestamp 0부터 12까지 간단한 T-Minskew 히스토그램의 예를 보이고 있다. *timestamp* 0에 히스토그램 H_0가 생성되어 5까지 지속되다가 timestamp 6에 새로운 히스토그램 H_1이 생성되고 *timestamp* 10에 히스토그램 재구축에 의해 다시 새로운 히스토그램 H_2가 생성되었음을 보이고 있다.

본 논문에서는 기본적으로 사각형 질의 Qr에 대한 선택도 추정을 하고 있다. 또 특정 timestamp에 히스토그

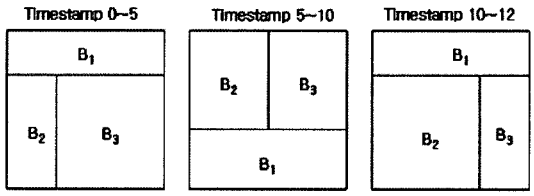


그림 5 T-Minskew 히스토그램

램을 재구축하므로 여러 개의 히스토그램이 존재하고 각 히스토그램은 유효 시간 $[t_{start}, t_{end})$를 가지고 있다. 각각의 히스토그램 내에는 여러 버킷이 존재하고 공간 영역 정보와 버킷이 포함하는 객체수를 저장한다. 셀은 전체 공간을 분할하기 위해 일정한 크기로 쪼개어 놓은 것을 말한다. 분할되는 셀의 형태는 2.1절의 그림 1과 같이 되고. 각각의 셀에는 객체의 정보가 수집되어 저장된다. 이 때 시공간 데이타베이스에서 사용되는 전체 시공간 영역은 매우 방대하므로 선택도 추정을 하기 위해서는 셀의 수를 줄여야할 필요가 있다. 따라서 셀들을 묶어 더욱 큰 단위의 형태로 만들어줄 필요가 있는데 이렇게 묶인 형태를 버킷이라고 한다. 하나의 버킷은 하나 이상의 셀을 포함하는 형태가 되며 히스토그램에 따라 버킷이 구성되는 알고리즘은 각각 다르다. 본 논문에서는 각 버킷 내에 존재하는 셀들의 객체 분포가 균일한 형태가 되도록 버킷을 구축한다. 이 때 버킷 구축 기준으로 사용하는 변수가 분산이다. 즉 그림 5와 같이 각 히스토그램은 3개의 버킷을 가지고 있고 분산 값을 이용하여 각각의 버킷내의 객체 분포가 균일하도록 구성이 되어 있다. 이러한 히스토그램을 이용한 전체 선택도

표 1 기호 테이블

기호	의미
Qr	윈도우 질의
$Qr.t$	윈도우 질의의 *timestamp*
$B_i.MBR$ = $[B_i.x_{min}, B_i.y_{min}, B_i.x_{max}, B_i.y_{max}]$	i번째 버킷의 2차원 공간 좌표
$B_i.num$	i번째 버킷이 포함하는 객체 수
H_i	i번째 히스토그램
$H_i.t$ = $[H_i.t_{start}, H_i.t_{end})$	i번째 히스토그램의 유효시간
$H_i.var$	i번째 히스토그램의 객체 변화량
$timestamp_{now}$	현재시간
Sel_i	i번째 버킷에 대한 선택도
Sel	전체 선택도
$celli$	i번째 셀
Bi	i번째 버킷
$OverlapArea(B_i.MBR)$	i번째 버킷과과 겹치는 질의 영역의 넓이
$area(Bi.MBR)$	i번째 버킷의 넓이
$avg(den)$	셀의 평균 밀도수 den의 평균
$MaxBucketNum$	최대 버킷 수

*Sel*은 각각의 버킷에 대한 선택도 추정 Sel_i을 합하여 구한다. 다음 표 1은 본 논문에서 사용하고 있는 기호들에 대한 정리를 하고 있다.

3.1 T-Minskew 히스토그램 구축

T-Minskew 히스토그램은 이력 데이타를 저장하는 히스토그램이다. 따라서 처음 히스토그램을 구축할 때 이미 저장되어 있는 이력 데이타를 기반으로 히스토그램을 구축하고, 이후에 새로운 데이타들이 데이타베이스에 저장될 때 현재 시간에 유지되고 있는 히스토그램을 갱신 시켜야 한다. 그림 6은 이미 저장되어 있는 이력 공간 객체들에 대한 히스토그램을 생성하는 알고리즘이다. 단계 1에서 이미 저장되어 있는 이력 공간 객체들의 timestamp를 충분히 포함할 수 있는 한계를 설정한 것이고, 단계 2에서는 검색되고 있는 시간 i에 대한 전체 공간 정보를 구축하기 위해 allocationData 프로시저를 호출한다. 이 때 allocationData 프로시저는 전체 공간을 셀로 나누고 검색되는 시간 i와 겹치는 데이타들을 공간 셀에 저장하는 역할을 한다. 단계 3부터 7까지 주어진 버킷 한계 MaxBucketNum까지 버킷을 생성하기 위해서 각 단계마다 분할될 최적 경우의 수를 Minskew 알고리즘을 이용하여 찾고 해당되는 버킷을 분할한다. 예를 들어, 버킷 분할 단계가 3이라면 j는 2이고 버킷은 현재 3개가 생성되어 있는 상태이다. 4번째 버킷을 생성하기 위해서 현재 존재하는 버킷 3개에 대한 각각의 최적의 분할 기준을 찾고 다시 이를 비교하여 그중 가장 나은 기준으로 버킷을 분할한다. 이러한 과정을 버킷이 MaxBucketNum 수만큼 될 때까지 반복하도록 한다.

```
Algorithm createHistogram
1   for(int i=0; i< MaxTimestamp; i++)
2       call allocationData procedure;
3       for(int j=0; j<MaxBucketNum; j++)
4           find bucket to be splitted
5               using Minskew algorithm;
6           split and adjust the bucket;
7       end for
8   end for
End createHistogram
```

그림 6 히스토그램 구축 알고리즘

```
Algorithm allocationData
1   each data[i] existed in timestamp j
2       find cells overlapped with extent of data[i];
3       cells.den++;
4       find a cell overlapped with
5           center point of data[i] in cells;
6       cell.num++;
7   end for
End allocationData
```

그림 7 데이타 할당 알고리즘

3.2 T-Minskew 히스토그램 재구축

가정 1. 히스토그램 H_i의 시간 간격은 H_i가 생성이 된 시점부터 i+1번째 재구축이 일어난 $timestamp_{now}$ - 1 까지로 한다.

가정 2. 히스토그램이 재구축 되지 않을 경우 객체 변화를 해당 버킷에 적용하기 변화된 객체 수를 시간과 함께 기록한다.

히스토그램 H_0가 실제 객체를 기반으로 timestamp 0에 구축되었을 때 유효 시간은 [0, *NOW*]을 가진다. 시간이 지나 timestamp가 1이 되었을 때 재구축을 해야 되는 지에 대한 판단을 하게 되는데 객체의 변화율과 임계치를 비교함으로써 히스토그램 재국축 여부를 결정한다. 만약 timestamp 1에서의 객체 변화율이 임계치를 넘어서지 않았을 경우는 timestamp 0에 구축된 히스토그램 H_0는 timestamp 1에도 유지가 되며 유효시간은 여전히 [0, *NOW*)로써 변화가 없게 된다. timestamp 4에서 객체 변화율이 임계치를 넘어 섰을 경우 히스토그램의 유효시간의 끝을 timestamp 4로 기록하여 유효시간을 [0, 4)으로 갱신하고 timestamp 4에는 이 시점에서 살아 있는 객체들을 기반으로 새로운 히스토그램 H_1를 생성한 뒤 유효시간을 [4, *NOW*)로 할당한다.

그림 8은 이러한 히스토그램 재구축 알고리즘이다. 단계 1에서 이전 timestamp까지의 객체 변화량에 현재 timestamp의 변화량을 더한다. 단계 2에서는 변화량에 전체 객체수를 나눈 값이 임계치를 넘어 섰는지를 검사한다. 만약 임계치를 넘어 선다면 단계 3에서 i번째 히스토그램의 유효 시간의 끝인 $H_i.t_{end}$를 현재 timestamp의 바로 이전으로 기록하고 단계 4에서 새로운 히스토그램 H_{i+1}을 생성한다. 단계 5와 6에서 각각 새로운 히

```
Algorithm rebuildHistogram
1   var += calculate variation of objects
           in universal space
           during [timestamp_now - 1, timestamp_now];
2   if (var / N) is over a given threshold
3       H_i.t_end = timestamp_now;
4       Create new histogram H_{i+1};
5       H_{i+1}.t_start = timestamp_now;
6       H_{i+1}.t_end = *;
7       var = 0;
8   else
9       call updateBucket procedure to apply
          changing information of objects at buckets;
11  end if
End rebuildHistogram
```

그림 8 히스토그램 재구축 알고리즘

```
Algorithm updateBucket
1   Find bucket Bi overlapped
2       with the space extent of objects;
    Calcuate the number of objects
3       changed within the bucket Bi;
4   if (Bi.num is changed)
        Append changed Bi.num
5           with timestamp t at the bucket Bi;
    end if
End updateBucket
```

그림 9 버킷 갱신 알고리즘

스토그램 H_{i+1}의 유효시간을 기록하고 단계 7에서 객체 변화량을 초기화 한 뒤 알고리즘을 종료한다. 만약 히스토그램의 객체 변화률이 정해진 임계치보다 높지 않다면 변경된 객체 정보를 히스토그램에 갱신해야하므로 단계 9에서 updateBucket 프로시저를 호출한다. 그림 9는 가정 2에 따른 updateBucket 프로시저의 알고리즘의 기술이다. updateBucket 프로시저의 단계 1에서 변경된 객체의 공간 영역과 겹치는 버킷 B_i를 찾는다. 단계 2에서는 버킷 B_i에서 변경된 객체수를 계산하여 만약 변화가 있다면 단계 4에서 버킷 B_i에 timestamp t와 함께 변경된 객체수를 추가한다.

히스토그램 재구축을 결정짓는 객체의 변화율은 히스토그램이 생성된 시점부터 실제 객체의 생성, 삭제, 갱신 횟수가 전체 객체 수에 비해 얼마나 일어났는지에 따른다. 예를 들어 전체 객체 수 N이 100이고 히스토그램 유지 기간 동안의 객체 변화 횟수가 20이라면 20%의 객체 변화율을 가지게 되며 만약 임계치가 15%라면 재구축이 일어나게 된다.

3.3 T-Minskew 히스토그램을 이용한 선택도 추정

시공간 이력 공간 질의에 대한 선택도 추정을 위해서는 우선 질의의 timestamp와 겹치는 히스토그램을 찾은 뒤 히스토그램 내에서 질의의 공간 영역과 겹치는 버킷들을 찾는다. 질의 공간 영역과 겹치는 각 버킷의 공간 영역을 해당되는 버킷의 전체 공간 영역으로 나누어 질의와 겹치는 공간 영역의 전체 공간 영역에 대한 비율을 구하고 이 비율에 버킷이 가지는 객체수를 곱함으로써 질의와 겹쳐지는 버킷내의 객체수를 추정할 수 있다. 마지막으로 각 버킷에서 추정된 객체수를 더함으로써 전체 선택도를 추정할 수 있게 된다. 식 (6)과 (7)은 이러한 과정을 수식으로 나타낸 것이다.

$$Sel_j = B_j.num * OverlapArea(B_j.MBR) \,/\, area(B_j.MBR) \quad (6)$$

$$Sel = \sum_{j=0}^{k} Sel_j \quad (7)$$

그림 10은 T-Minskew를 이용한 선택도 추정 방법의

```
Algorithm Selectivity
1  Find a histogram Hi that satisfy
     (Hi-t ⊃ Qp-t or Qr-t),
       where (0 ≤ i ≤ number of histogram)
2  Find buckets that overlap
     with query area within Hi
3  For j=0 to number of buckets overlapped with query
4    Selj = Bj-num * OverlapArea(Bj-MBR)
           / area(Bj-MBR)
   end for
5  Sel = Σj=0~k(Selj), where k is
         number of buckets overlapped with query
End Selectivity
```

그림 10 선택도 추정 알고리즘

알고리즘을 나타내고 있다. 단계 1에서 질의의 Timestamp를 포함하는 히스토그램을 찾고, 단계 2에서는 검색된 히스토그램 내에서 질의의 공간 영역과 겹치는 버킷들을 찾는다. 단계 3과 4에서는 질의와 겹치는 버킷의 선택도를 추정한다. 이 때 버킷 선택도를 추정 공식은 식 (6)과 같다. 단계 5에서 전체 선택도를 구하기 위해 각각 구해진 버킷의 선택도를 모두 합하는 것으로 알고리즘은 종료된다.

예를 들어 그림 11은 그림 5에서 질의의 timestamp와 겹치는 히스토그램 H_2를 가져온 것이다. 질의 사각형의 공간 영역과 각 버킷의 정보는 다음과 같다.

$Qr.MBB$ = [5, 5, 10, 10]

$B_0.MBB$ = [0, 8, 10, 10], $B_0.num$ = 3

$B_1.MBB$ = [0, 0, 7, 8], $B_1.num$ = 9

$B_2.MBB$ = [7, 0, 10, 8], $B_2.num$ = 2

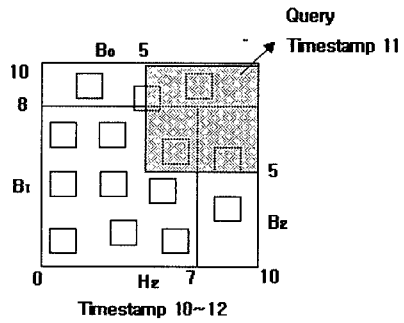


그림 11 T-Minskew를 이용한 선택도 추정

이 때 그림 11에서 각 버킷의 선택도를 추정과 전체 선택도 추정은 다음과 같이 계산된다.

Sel_1 = 1.5 = 3 * 10 / 20

Sel_2 = 0.96 = 9 * 6 / 56

Sel_3 = 0.75 = 2 * 9 / 24

$$Sel = \sum_{j=0}^{3} Sel_j = 1.5 + 0.96 + 0.75 = 3.21$$

4. 실험 평가

이번 절에서는 본 논문에서 제시한 T-Minskew 히스토그램의 성능 평가를 위한 실험 환경을 소개하고 이론적인 실험 모델로써 오류율 계산법과 실제 실험결과를 제시한다. 본 실험은 Intel Pentium4 northwood 2.8GHz CPU, 512MB RAM, 160GB HDD의 Windows XP 환경에서 수행하였으며 T-Minskew 히스토그램의 구현을 위해 Visual Studio 6.0을 사용하였다. T-Minskew 히스토그램의 생성을 위해 이력 정보를 가지는 공간 객체 10,000개를 무작위로 생성하여 히스토그램을 구축하였다. 표 2는 객체를 생성한 도메인 영역이다.

표 2 객체가 존재하는 전체 도메인

변수	값
공간 영역 X, Y	0 ~ 2000
시간 영역 T	0 ~ 999

4.1 이론적인 실험 모델

T-Minskew의 선택도 추정의 정확도를 판단하기 위해 여러 가지 실험 기준에 따른 상대적인 오류율을 측정하도록 한다. 실험을 위한 상대 오류율 계산법은 식 (8)과 같다. *Sel*은 T-Minskew를 이용하여 질의 결과를 추정한 값이고 *Sel'*는 실제 질의 결과 값이다.

$$Err = |Sel - Sel'| \ / \ Sel', \text{ 단 } Sel' > 0 \quad (8)$$

이 때 어떠한 질의의 실제 결과가 0가 되는 경우 오류율 측정이 불가능하게 되므로 1로써 대체하여 사전에 예외 상황에 대한 대처를 한다. 다수의 질의에 대한 오류율을 구한 뒤 이들의 평균을 구하기 때문에 강제로 0인 값을 1로써 대처한다고 하여도 질의의 수가 많으면 많을수록 실험 결과에 영향을 미치는 정도는 매우 작다. 그리고 특정 질의에 대한 편중된 결과를 해결하기 위해 Q_n개의 다수 질의에 대한 선택도 추정 오류율을 측정하고 그것의 평균을 구함으로써 실험에 보다 높은 신뢰도를 가지도록 한다. 따라서 최종 오류율은 식 (9)와 같다.

$$Avg(Err) = (\sum_{i=1}^{Qn} Err_i) \ / \ Qn \quad (9)$$

만약 Minskew 히스토그램의 평균 오류율이 M_{err}이라면 T-Minskew의 오류율 TM_{err}은 M_{err}보다 α만큼 증가한다. 왜냐하면 임계치 기법에 의하여 히스토그램을 유지할 때 객체의 변화에 따른 히스토그램의 재구축을 하지 않고 이전 단계에서 사용했던 히스토그램을 그대로 쓰기 때문이다. 또한 추가된 오류율 α는 높은 임계치일 수록 히스토그램의 재구축률이 감소하므로 증가하게 된다. 따라서 Minskew 히스토그램에 대한 T-Minskew의 오류율 TM_{err}은 식 (10)과 같다. 하지만 본 논문에서 제시한 updateBucket 알고리즘에서 보는 바와 같이 객체의 변화에 따른 버킷들의 정보 변화를 위해 히스토그램의 재구축을 통하지 않고 임계치 내에서 각각의 버킷에게 맡겨두기 때문에 임계치에 따른 히스토그램 재구축 횟수는 감소하고 오류율의 변화도 거의 없다.

$$TM_{err} = M_{err} + \alpha \quad (10)$$

4.2 실험 결과 분석

실제 실험은 다음의 변수에 따라 수행하여 T-Minskew 히스토그램에 대한 평가를 한다.

(1) 고정된 임계치와 고정된 버킷을 가지면서 변화하는 질의 크기에 대한 추정 오류율
(2) 고정된 임계치와 고정된 질의크기를 가지면서 변화하는 버킷 수에 대한 추정 오류율
(3) 고정된 임계치와 고정된 질의크기를 가지면서 변화하는 버킷 수에 대한 히스토그램 구축 시간
(4) 고정된 버킷 수와 고정된 질의 크기를 가지면서 변화하는 임계치에 대한 히스토그램 재구축 횟수
(5) 고정된 버킷 수와 질의 크기를 가지면서 변화하는 임계치에 대한 추정 오류율

4.2.1 변화하는 질의 크기에 대한 추정 오류율 평가

질의 크기에 대한 추정 오류율의 변화를 보기 위해 임계치를 30%로 고정하고 히스토그램의 버킷 수를 50으로 고정 한 뒤 각각 질의 크기에 따라 무작위로 만든 100개의 질의에 대해 실험하였다. 질의 크기는 각 축별로 전체 공간에 대한 비를 나타낸 것으로 만약 전체 공간의 x축 영역과 y축 영역이 1000이라고 하였을 때 질의 크기가 10%라면 각 축별로 질의 크기는 100의 크기를 가지게 되므로 2차원 공간이라면 전체 공간에 비해 1%의 크기를 가지게 됨을 알 수 있다. 실험의 결과 그림 12와 같이 질의 크기가 작으면 작을수록 오류율이 증가하였다. 이러한 실험결과를 가지는 이유는 질의 크기가 작으면 작을수록 추정된 결과값과 실제 질의 결과값이 차이가 많지 않음에도 불구하고 상대적인 오류율 계산법에 따라 오류율은 상대적으로 커질 수 있기 때문이다. 예를 들어 실제 질의 결과가 1이고 추정된 결과가

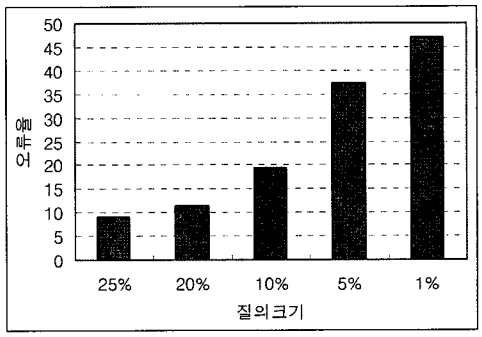


그림 12 질의 크기에 대한 추정 오류율

1.6이면 60%의 오류율을 가지는 반면 실제 질의 결과가 10이고 추정된 질의 결과가 10.6이라면 단순히 6%의 오류율을 가진다. 따라서 실험에서 사용한 10,000개의 객체 수 보다 더 많은 객체에 대한 실험을 하면 질의 크기가 작아도 질의 결과가 커지게 될 것이므로 이러한 현상은 줄어들 것이다.

4.2.2 변화하는 버킷 수에 대한 추정 오류율 평가

버킷 수에 따른 선택도 추정 오류율에 대한 평가를 위해 임계치를 30%로 고정하고 질의 크기는 10%로 고정한 뒤 무작위로 질의를 100개 생성하여 실험하였다. 버킷의 수가 작으면 작을수록 하나의 버킷이 포함하는 공간 영역 또한 증가하고 최적의 편중도를 만족할 수 없기 때문에 오류율이 증가할 수 있다. 다시 말해 버킷의 크기가 크면 클수록 편중된 객체 분포를 버킷으로 분할 시켜 균일하게 바꿀 수가 있게 된다. 하지만 너무 많은 버킷의 수는 오히려 최적의 히스토그램 상태를 억지로 분할 할 수도 있기 때문에 오히려 오류율이 증가할 수도 있다. 이러한 실험의 결과가 그림 13에 나타나 있다. 버킷의 개수 80까지는 오류율이 하락하지만, 그 이후에는 오히려 오류율이 증가한다. 따라서 히스토그램의 상태에 따라 최적의 버킷 수를 결정하는 일은 선택도 추정 오류율을 낮추는 대에 있어 중요하다.

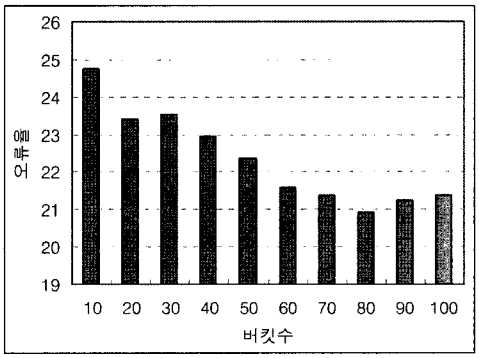


그림 13 변화하는 버킷 수에 따른 추정 오류율

4.2.3 변화하는 버킷 수에 대한 히스토그램 구축 시간 평가

실험 항목(3)을 평가하기 위해 실험항목(2)에서의 임계치와 질의 크기를 사용하여 버킷의 변화에 따른 전체 히스토그램 구축 시간을 실험하였다. 실험 시간은 CPU 클럭 횟수를 이용하여 측정하였다. 히스토그램을 구축할 때 주어진 버킷 수가 되거나 더 이상 분할 이득이 없는 경우 버킷 분할을 멈추게 된다. 이 때 히스토그램에 구축에 필요한 시간은 대부분 버킷 분할을 하는 과정에서 발생하며 주어진 버킷의 수가 많으면 많을수록 히스토그램 구축 시간이 증가하게 된다. 그림 14는 이러한 결과를 나타내고 있으며 주어진 버킷수가 증가함에 따라 히스토그램 구축 시간도 함께 증가함을 알 수 있다.

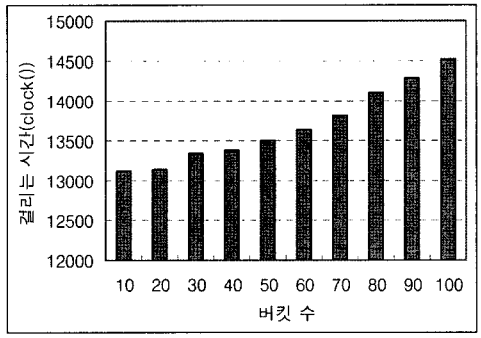


그림 14 변화하는 버킷 수에 따른 히스토그램 구축 시간

4.2.4 변화하는 임계치에 대한 히스토그램 재구축 횟수 평가

변화하는 임계치에 대한 히스토그램의 재구축 횟수를 평가하기 위해 버킷 수는 50개로 고정시키고 4.2.2절에서 사용하였던 10%의 크기를 가지는 100개의 질의를 사용하였다. 이 때 임계치가 크면 클수록 시간에 따른 객체의 변화량이 많아도 히스토그램을 유지하고 갱신하기 때문에 재구축 횟수가 줄어든다. 그림 15는 임계치를 5부터 100까지 5단위로 변화시키며 재구축 횟수를 측정한 것이다. 실험 결과 역시 임계치의 증가에 따라 재구축 횟수가 점점 줄어들고 있다.

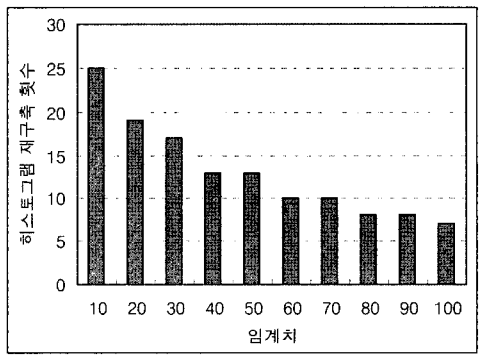


그림 15 변화하는 임계치에 대한 히스토그램 재구축 횟수

4.2.5 변화하는 임계치에 대한 추정 오류율 평가

변화하는 임계치에 대한 선택도 추정 오류율 평가를 위해 버킷 수는 50개로 고정 시키고 4.2.2절에서 사용하였던 10%의 크기를 가지는 100개의 질의를 사용하였다. 이러한 실험의 결과가 그림 16에 나타나 있다. 4.1절과 같이 임계치가 변화하더라도 선택도 추정 오류율의 변화는 거의 없음을 알 수 있다. 이것은 히스토그램의 작은 재구축 횟수로도 낮은 오류율을 가지는 선택도 추정

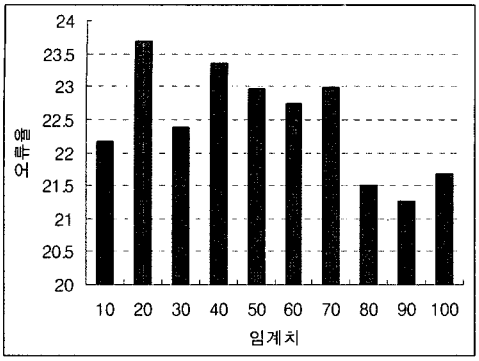


그림 16 변화하는 임계치에 따른 추정 오류율

을 할 수 있음을 의미한다.

5. 결 론

본 논문에서는 이력 공간객체의 선택도 추정을 위한 T-Minskew 히스토그램을 제안하였다. 히스토그램의 효과적인 유지를 위하여 임계치 기법을 이용하였다. 그에 대한 효과로써 히스토그램의 재구축 판단 여부를 객체의 변화율이 주어진 임계치를 넘어설 경우에만 발생하므로 잦은 히스토그램 재구축 횟수를 줄이고 적절한 추정 오류율을 유지할 수 있었다. 히스토그램을 유지하는 동안의 객체 정보를 갱신하기 위하여 히스토그램 전체를 갱신시키지 않고 변화한 객체 정보를 포함하는 히스토그램 내의 버킷만을 갱신하는 기법을 사용하였다. 이 기법을 통하여 이전 시간의 객체수를 해당 버킷에 저장하고 변화된 객체수를 시간과 함께 저장함으로써 효율적인 히스토그램 유지를 할 수 있다. T-Minskew 히스토그램은 편중된 객체들을 균일한 구조로 바꾸기 위해 Minskew 히스토그램 기법을 사용하였다.

실험결과로써 질의 크기가 작을수록 선택도 추정 오류율이 증가함을 알 수 있었다. 그리고 히스토그램이 가질 수 있는 버킷의 개수가 많을수록 추정 오류율이 감소하였다. 그러나 최적의 버킷 개수를 넘어 서는 경우는 오히려 오류율이 증가했다. 임계치가 크면 클수록 히스토그램의 재구축 횟수는 줄어들었는데 이때 히스토그램의 재구축 횟수가 작더라도 특정 시점에서의 공간 선택도 추정의 오류율이 높아지지 않았다. 이러한 실험 결과는 T-Minskew 히스토그램이 효과적인 히스토그램 유지 기법을 통해 재구축 횟수는 줄이면서 선택도 추정 오류율을 유지할 수 있기 때문이다.

앞으로의 연구 과제는 시공간 범위 질의의 선택도 추정이 가능한 히스토그램을 구축하고 유지하는 기술을 개발하는 것이다.

참 고 문 헌

[1] Tao, Y., Papadias, D., and Sun, J., "The TPR*-tree: An Optimized Spatio- Temporal Access Method for Predictive Queries," In Proceedings of the 29th Very Large Data Bases Conference, Berlin, Germany, pages 790-801, 2003.

[2] Tao, Y. and Papadias, D., "Time-Parameterized Queries in Spatio-Temporal Databases," In Proceedings of ACM SIGMOD international conferences on Management of data, pages 334-345, 2002.

[3] Acharya, S., Poosala, V., and Ramaswamy, S., "Selectivity Estimation in Spatial Databases," In ACM SIGMOD, USA, pages 13-24, 1999.

[4] Aboulnaga, A. and Naughton, J. "Accurate Estimation of the Cost of Spatial Selections," In ICDE, pages 123-134, 2000.

[5] Poosala V., Yannis E., Ioannidis, Peter J., Haas., and Eugene J. Shekita, "Improved Histograms for Selectivity Estimation of Range Predicates," In ACM SIGMOD, NY, USA, pages 294-305, 1996.

[6] Wang, M., Vitter, J., S., Lim, L., and Pdmanabhan, S., "Wavelet-Based Cost Estimation for Spatial Queries," In The 7th International Sysposium on Spatial and Temporal Databases(SSTD), CA, USA, pages 175-196, July 2001.

[7] Nikos Mamoulis and Dimitris Papadias, "Selectivity Estimation of Complex Spatial Queries," In The 7th International Sysposium on Spatial and Temporal Databases (SSTD), CA, USA, pages 156-174, July 2001.

[8] Choi, Y. and Chung, C., "Selectivity Estimation for Spatio-Temporal Queries to Moving Objects," In ACM SIGMOD, pages 440-451, 2002.

[9] Tao, Y., Sun, J., and Papadias, D., "Selectivity Estimation for Predictive Spatio-Temporal Queries," ICDE, pages 417-428, 2003.

[10] Hadjieleftheriou, M., Kollios, H., and Tsotras, V J., "Performance Evaluation of Spatio-temporal Selectivity Estimation Techniques," In The 15th Int. conference on Science and Statistical Database Management (SSDBM), pages 202-211, 2003.

[11] Zhan, Q. and Lin, X., "Clustering Moving Objects for Spatio-temporal Selectivity Estimation," In ADC, pages 123-130, 2004.

[12] Yossi Matias, Jeffrey Scott Vitter, and Min Wang, "Wavelet-Based Histogram for Selectivity Estimation," In Proceedings of ACM SIGMOD international conferences on Management of data, pages 448-459, 1998.

[13] Lee, J., Kim, D., and Chung, C., "Multi-dimensional selectivity estimation using compressed histogram information," In Proceeding of ACM SIGMOD international conferences on Management of data, pages 205-214, 1999.

신 병 철

2004년 충북대학교 컴퓨터교육과(이학사). 2006년 충북대학교 대학원 컴퓨터교육과(교육학석사). 관심분야는 질의 최적화, 시공간 데이타베이스, GIS, Ubiquitous Computing, u-learning, 데이타 마이닝

이 종 연

1985년 충북대학교 전자계산기공학과(공학사). 1987년 충북대학교 대학원 전자계산기공학과(공학석사). 1999년 충북대학교 대학원 전자계산학과(이학박사) 1989년 비트컴퓨터(주) 개발부. 1990년~1994년 현대전자산업(주) 소프트웨어연구소 주임연구원. 1994년~1996년 현대정보기술(주) CIM사업부 책임연구원. 1999년~2003년 삼척대학교 정보통신공학과 조교수. 2003년~현재 충북대학교 컴퓨터교육과 부교수. 관심분야는 질의 최적화, 시공간 데이타베이스, 데이타 마이닝, Bioinformatics, Ubiquitous Computing, u-learning, GIS.