고차원에서 선택율 추정을 위한 블록 히스토그램 압축방법

이 주 홍[†] · 전 석 주[††] · 박 선[†††]

요 약

데이터베이스 질의 최적화기는 가장 효율적인 실행계획을 구하기 위해서 질의의 선택율을 추정한다. 일반적으로 애트리뷰트들은 서로 독립적이지 않기 때문에 여러 개의 애트리뷰트를 가지는 질의에 대해서는 다차원 선택율 추정 기법이 필요하다. 대부분의 상용 데이터베이스에서는 히스토그램이 계산 오버헤드가 많지 않고 작은 에러율로 데이터 분포를 근사 시킬 수 있기 때문에 실용적으로 많이 사용되고 있다. 그러나 여러 개의 애트리뷰트를 가진 다차원 질의의 경우에서는 차원이 높아 질수록 에러율을 낮추기 위해 많은 저장 공간을 필요로 하기 때문에 히스토그램 방법이 적합하지 않다. 이 논문에서는 다차원 선택율 추정을 위한 새로운 기법을 제안한다. 다차원 공간에서 크기가 작은 히스토그램 버켓을 많이 만들고 이 버켓의 정보를 DCT로 압축하여 선택율 추정에 사용함으로써 에러율을 작게 하고 저장 공간의 사용량도 줄인다. 폭 넓은 실험 결과는 본 논문에서 제시한 방법들의 타당성과 이점을 확인시켜 준다.

Block Histogram Compression Method for Selectivity Estimation in High-dimensions

Ju-Hong Lee[†] · Seok-Ju Chun[††] · Sun Park[†††]

ABSTRACT

Database query optimizer estimates the selectivity of a query to find the most efficient access plan. Multi-dimensional selectivity estimation technique is required for a query with multiple attributes because the attributes are not independent each other. Histogram is practically used in most commercial database products because it approximates data distributions with small overhead and small error rates. However, histogram is inadequate for a query with multiple attributes because it incurs high storage overhead and high error rates. In this paper, we propose a novel method for multi-dimensional selectivity estimation. Compressed information from a large number of small-sized histogram buckets is maintained using the discrete cosine transform. This enables low error rates and low storage overheads even in high dimensions. Extensive experimental results show advantages of the proposed approach.

키워드 : 질의 최적화기(Database Query Optimizer), 블록 분할 히스토그램(Block Partitioned Histogram), 다차원 히스토그램(Multi-dimension Histogram)

1. 서 론

데이터베이스에서 질의 최적화기는 질의의 가능한 실행 계획들의 비용을 추정하여 가장 효율적인 실행 계획을 선택한다. 실행 계획의 비용을 추정하기 위해서 가장 중요한 요소가 선택율이다. 선택율이 필요한 이유는 가장 간단한 질의도 실행 중에 중간 결과를 생성한다. 그러므로 최적화기는 질의 비용을 계산하기 위해 중간 결과의 크기를 알아야 한다. 이 크기는 실제 데이터 값에 많이 의존하므로 정확한 비용 산정이 어렵기 때문에 질의 처리 비용의 계산을 위해 선택율 추정이 필요하다. 선택율 추정의 정확도는 가장 효율적인 플랜을 선택하는 데 큰 영향을 미친다. 선택율 추정을 위한 통계 자료는 대개 데이타베이스에 있는 데이타의 분포를 근사적으로 나타낸다. 질의에 포함된 애트리뷰트의 수에 따라 일차원 선택율 추정과 다차원 선택율 추정으로 구분된다. 다차원 선택율 추정을 필요로 하는 응용은 많이 있다. 여러 애트리뷰트를 가진 질의의 최적화는 그 질의의 결과가 다차원 공간에서 표현되는 애트리뷰트들의 다차원 데이터 분포에 의존하므로 다차원 선택율 추정 기법이 필요하다. 멀티미디어 데이터 베이스에서는 멀티미디어 데이터의 특징 벡터가 다차원 색인 트리에 저장되므로 다차원 선택율 추정 기법이 필요하다[3].

지금까지 개발된 대부분의 다차원 선택율 추정방법은 히스토그램 방법에 기반하고 있다[14]. 그런데 히스토그램 방법에 기반한 다차원 추정방법은 차원이 높아지면 에러율이 높아질 수 밖에 없는 근본적인 문제가 있다. 그 이유는 히스토그램 방법은 버켓 안에서는 데이터의 분포가 일정하다고 가정하는데 실제로는 그렇지 않기 때문에, 질의 영역과

※ 이 논문은 2001학년도 인하대학교의 지원에 의하여 연구되었음(INHA-21858-01).
† 종신회원 : 인하대학교 컴퓨터공학부 교수
†† 정 회 원 : 안산1대학 인터넷정보과 교수
††† 준 회 원 : 인하대학교 대학원 전자계산공학과
논문접수 : 2002년 9월 11일, 심사완료 : 2003년 6월 18일

일부만 겹쳐있는 히스토그램 버켓은 에러를 발생시킨다. 그런데 질의 영역과 일부만 겹치는 히스토그램 버켓은 모두 질의 영역의 표면에 위치하고 있고, 전체 볼륨에서 질의 영역의 표면의 볼륨이 차지하는 비율은 차원이 증가함에 따라서 점점 커지기 때문에 히스토그램 방법의 에러율을 줄이기 위해서는 히스토그램 버켓의 크기를 줄여야 한다. 그러나 아래의 식에서 표현되는 것과 같이 데이터 공간이 정규화되어 있고 히스토그램 버켓의 한변의 평균 크기를 a라고 하고 dim을 차원이라고 한다면 전체 히스토그램 버켓의 개수는 a의 dim승에 반비례하므로 차원이 증가하면 히스토그램 버켓의 개수는 폭발적으로 증가하여 심각한 저장 공간 문제를 야기한다.

$$\#\ \text{of buckets} \propto \frac{1}{d^{\dim}},\ 0 < a < 1,$$

여기서 a는 버켓의 한 차원의 길이이다.

따라서 고차원에서 적은 양의 저장 공간를 가지고 에러율을 낮추는 것은 불가능하다. 이 논문에서는 위와 같은 문제점들을 해결하기 위한 다음과 같은 새로운 방법을 제안한다. 많은 수의 크기가 작은 히스토그램 정보들을 이산 여현 변환(DCT : Discrete Cosine Transform)을 사용하여 압축하고 유지한다. 이것은 고차원에서 작은 크기의 히스토그램 버켓을 사용하기 때문에 선택율 추정의 에러율이 작아짐과 동시에 많은 양의 히스토그램 버켓 정보를 압축하므로 적은 양의 저장공간이 가능해 진다. 또한 이 방법은 블록 분할 방법을 사용하면 편중도가 높은 데이터 분포에서도 에러율이 현저히 낮아진다. 광범위한 실험을 통해서 이 논문에서 제시한 방법이 고차원에서 적은 저장 공간을 사용하고도 낮은 에러율을 보이고 선택율 추정을 빠르게 한다는 사실을 보여준다.

이 논문의 구성은 다음과 같다. 2절에서는 관련 연구에 대해서 개관하고 3절에서는 DCT를 이용하여 히스토그램을 압축하고, 압축된 자료로 선택율을 계산하는 방법과 데이터 공간을 분할 하여 압축하는 방법에 대해서 논한다. 4절에서는 저장공간 요구 사항과 계산 시간에 대해서 논한다. 5절에서는 실험 결과들을 보이고 상세히 토의한다. 6절에서는 결론을 보인다.

2. 관련 연구

일차원 선택율 추정 기법들은 다음과 같은 네가지 부류로 분류된다[2, 12, 13]. 모델 함수에 의한 모수적 방법[5], 일반적인 다항식을 이용한 커브 횟팅 방법[2, 17], 표본 추출에 의한 방법[6], 히스토그램에 의한 비모수적 방법[7-9, 13]등이 있다. 이 방법들 중에서 히스토그램 방법이 상대적으로 적은 양의 저장 공간을 가지고도 작은 에러율로 거의 모든 데이터 분포를 근사 시킬 수 있기 때문에 네 가지 부류의 방법 중에서 실용적으로 가장 선호되는 방법이다.

Chaudhuri[4]는 멀티미디어 데이터 베이스에서 퍼지 질의의 최적화를 위한 실험에서 다차원 선택율 추정 방법으로 상관 프랙탈 차원[1]을 사용하였다. 그러나 상관 프랙탈 차원을 사용한 방법은 같은 모양과 같은 크기를 가지는 모든 질의에 대한 평균적인 선택율을 계산해 준다. 여러 개의 애트리뷰트를 가지는 질의에 대한 다차원 선택율 추정 기법에 대해서는 여러 가지 방법들이 제안되었다[11, 14]. Wang 등은 웨이브렛을 이용한 방법을 제안하였다. Poosala 등은 SVD, 힐버트 넘버링, PHASED, MHIST 방법 등을 제안하였고 이 방법들은 히스토그램에 기반한 방법들이다. 따라서 이 방법들은 다차원 조인트 데이터 분포를 서로 겹치지 않는 버켓들로 분할한다. SVD 방법은 수치 해석 분야에서 이미 효율적인 SVD 방법이 많이 제공되고 있어서 상당히 효율적이지만 2차원으로 사용이 제한된다는 단점이 있다. 힐버트 넘버링 방법에 의해서 생성되는 버켓은 하이퍼 직각 다면체가 아닐 수 있기 때문에 질의와 겹치는 버켓을 찾는 것이 어렵고, 다차원의 인접성을 일차원에서 보존해 주지 못하기 때문에 이 방법을 적용한 선택율 추정 값이 부정확할 수 있다. PHASED 방법은 임의로 선택된 차원의 데이터 분포를 임의의 일차원 히스토그램 방법을 적용하여 분할하며 이러한 과정을 모든 차원에 대해서 반복 적용한다. MHIST 방법은 PHASED 방법을 개선한 것으로서 반복되는 각 단계에서 가장 중요한 차원을 선택하여 데이터 분포를 분할한다. Poosala의 실험에 의하면 [14] 다차원 방법들 중에서 MHIST가 가장 성능이 좋다고 알려져 있다. 그러나 MHIST도 2차원에서는 좋은 성능을 보이지만 3차원과 4차원에서는 상대적으로 높은 에러율을 보이고 있다. 이 사실은 다차원 데이터 공간을 서로 겹치지 않는 버켓 들로 분할하는 것이 쉽지 않다는 것을 보여준다. 따라서 이 방법들은 고차원에서는 사용하기가 어렵다. 웨이브렛 방법은 Harr 웨이블렛을 사용하는 경우 계산 속도가 빠르고 효율적이지만 기본적으로 이산적인 데이터에 잘 적용되는 방법으로서 연속적인 데이터에 대해서 적용하려면 미리 이산화 시켜야 되는 부담이 있다[16].

3. DCT를 이용한 히스토그램 압축 방법

3.1 DCT의 정의

연속된 데이터 $\vec{F} = (f(0), f(1), \cdots, f(N-1))$의 DCT 계수, $\vec{G} = (g(0), g(1), \cdots, g(N-1))$는 식 (1)과 같이 정의된다 [10, 15].

$$g(u) = \sqrt{\frac{2}{N}}\, k_u \sum_{n=0}^{N-1} f(n) \cos\left(\frac{(2n+1)u\pi}{2N}\right) \qquad (1)$$

$$k_u = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u = 0, \quad u = 0, \cdots, N-1 \\ 1 & \text{for } u \neq 0 \end{cases}$$

$\overrightarrow{F} = (f(0), f(1), \cdots, f(N-1))$는 다음과 같이 식 (2) 역 DCT에 의해서 복구된다.

$$f(n) = \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} k_u\, g(u) \cos\left(\frac{(2n+1)u\pi}{2N}\right), \quad n = 0, \cdots, N-1 \tag{2}$$

식 (1)의 일차원 DCT는 식 (3)과 같이 2차원 DCT로 확장된다. $[F]_2$를 2차원 데이터를 나타내는 $M \times N$ 매트릭스라고 하자. $[G]_2$를 $[F]_2$의 2차원 DCT 계수라고 하자. 그러면 $[G]_2$의 원소 (u, v)는 다음과 같이 주어진다.

$$g(u,v) = \frac{2k_u k_v}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \cos\left[\frac{(2m+1)u\pi}{2M}\right] \cos\left[\frac{(2n+1)v\pi}{2N}\right] \tag{3}$$

여기서 $u = 0, \cdots, M-1$ and $v = 0, \cdots, N-1$

이제 위 식을 다음과 같이 재귀적으로 k차원의 DCT로 일반화한다.

$[F]_k$를 $N_1 \times N_2 \times \cdots \times N_k$인 k차원의 데이터라고 하자. $u(t)$와 $n(t)$를 다음과 같이 정의하자. $1 < t \le k$인 t에 대해서, $u(t) = (u_1, \cdots, u_t), n(t) = (n_1, \cdots, n_t)$이고, $1 \le i \le k$인 i에 대해서 $u_i = 0, \cdots, N_i - 1, n_i = 0, \cdots, N_i - 1$이다. $[G]_k$를 $[F]_k$의 DCT 계수라고 하자. $G(u(t))$, $F(u(t))$를 식 (4)와 같이 정의한다.

$$G(u(t)) = \sqrt{\frac{2}{N_t}} k_{u_t} \sum_{n_t=0}^{N_t-1} G(u(t-1)) \cos\left(\frac{(2n_t+1)u_t\pi}{2N_t}\right) \tag{4}$$

$$G(u(1)) = \sqrt{\frac{2}{N_1}} k_{u_1} \sum_{n_1=0}^{N_1-1} f(n_1, \cdots, n_k) \cos\left(\frac{(2n_1+1)u_1\pi}{2N_1}\right)$$

$$F(n(t)) = \sqrt{\frac{2}{N_t}} \sum_{n_t=0}^{N_t-1} k_{u_t} F(n(t-1)) \cos\left(\frac{(2n_t+1)u_t\pi}{2N_t}\right)$$

$$F(u(1)) = \sqrt{\frac{2}{N_1}} \sum_{n_1=0}^{N_1-1} k_{u_1} g(u_1, \cdots, u_k) \cos\left(\frac{(2n_1+1)u_1\pi}{2N_1}\right)$$

그러면 k차원의 DCT 계수는 $g(u_1, \cdots, u_k) = G(u(k))$이다. 또한 그 역 DCT 변환은 $f(u_1, \cdots, u_k) = F(u(k))$이다.

3.2 DCT를 사용한 히스토그램 압축과 다차원 선택율 추정

1장, 2장에서 설명된 것과 같이 히스토그램 방법은 고차원의 선택율 추정에 사용할 수 없다. 대안으로 커브휘팅 방법을 생각 할 수 있다. 본 논문에서는 DCT를 사용하여 히스토그램 정보들을 압축하고 압축된 정보를 사용하여 다차원 선택율 추정을 하는 커브 휘팅 방법을 제안한다. 이 방법은 다차원 데이터 공간을 일정한 크기의 히스토그램 버켓으로 나눈다. 데이터 분포의 상관도가 매우 높은 경우에는 히스토그램 정보를 DCT로 효율적으로 압축하는 것이 가능하다. 그리고 역 DCT를 사용함으로써 원래의 데이터 분포를 심하게 왜곡하지 않고 복구할 수 있다. 이 방법은 크기가 작은 많은 개수의 균일한 히스토그램 버켓을 사용하는 반면에 이것들을 압축하여 저장하기 때문에, 다차원에서의 많은 저장 공간과 높은 에러율의 문제를 해결한다.

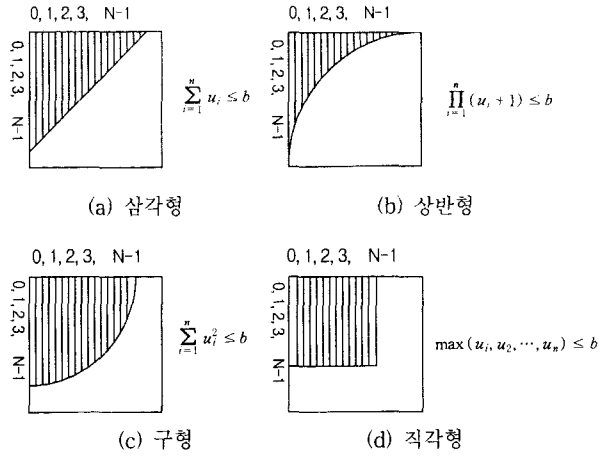
다차원 선택율 추정에 DCT를 사용하기위해서는 계수 샘플링, DCT 계수의 계산과 유지, 선택율 계산 방법을 고려하여야 한다.

3.2.1 기하 영역 샘플링

고차원에서도 낮은 에러율을 얻기 위해서 히스토그램 버켓의 크기를 충분히 작게 하여야 한다. 그러나 차원이 증가하면 DCT 계수의 수는 기하급수적으로 증가한다. 만약 모든 계수들을 계산하고 난 후 적절한 계수를 선택한다면 심각한 계산 부담이 발생한다. 따라서 큰 값을 가질 것으로 추정되는 계수들만을 선택하여 계산한다. 적절한 DCT 계수들을 선택하기 위해서 신호처리 분야에서 사용되는 저주파 필터링 기법을 다차원으로 확장하여 사용한다[10, 15]. 지정된 영역안에 있는 DCT 계수들만이 계산되고 나머지는 모두 0으로 간주된다. 이를 기하 영역 샘플링(Geometrical Zonal Sampling)이라고 부른다.

기하 영역 샘플링 방법에는 여러 가지가 있다. 삼각형, 상반형, 구형, 직각형 방법 등이다. 삼각형 방법은 인덱스 u_1과 u_2의 합이 주어진 값 b보다 작거나 같은 계수 $g(u_1, u_2)$를 선택한다. 즉, $u_1 + u_2 \le b$, $u_1 = 0, \cdots, N_1 - 1$, $u_2 = 0, \cdots, N_2 - 1$이다. 다차원인 경우에는 $\sum_{i=1}^{n} u_i \le b$, $u_i = 0, \cdots, N_i - 1$인 계수 $g(u_1, \cdots, u_n)$를 선택한다. 이 방법으로 샘플링 했을 때의 선택되는 계수의 개수는 만약 $b \le N_i$ 조건이 만족하면 ${}_{n+b}C_b$로 주어진다. 상반 방법은 계수의 인덱스의 곱이 주어진 값 b보다 작거나 같은 계수를 선택하는 것이다. 즉 $\prod_{i=1}^{n} (u_i + 1) \le b$, $u_i = 0, \cdots, N_i - 1$를 만족하는 계수를 선택한다. 이 방법은 각 차원의 고주파 성분이 앞의 방법에 비해 더 많이 포함되도록 하는 방법이다. 이 방법으로 샘플링 했을 때의 선택되는 계수의 개수는 대략 $O(n^{\lfloor \log_2 b \rfloor})$로 주어진다. 구형 영역 샘플링 방법은 인덱스의 제곱의 합이 정해진 값보다 작은 계수를 선택하는 방법이다. $\sum_{i=1}^{n} u_i^2 \le b$, $u_i = 0, \cdots, N_1 - 1$. 이 방법으로 샘플링 했을 때의 선택되는 계수의 개수는 근사적으로 $O\left(\frac{(\pi b)^{n/2}}{\lfloor n/2 \rfloor !}\right)$로 주어진다. 직각형 방법은 인덱스중의 최대값이 주어진 값 b보다 작거나 같은 계수를 선택한다. 즉 $\max(u_1, u_1, \cdots, u_1) \le b$인 계수를 선택한다. 이 방법으로 샘플링 했을 때의 선택되는 DCT 계수의 개수는 $O(b^n)$로 주어진다.

(그림 1)은 2차원에서의 기하영역 샘플링 방법에 관한 예를 보여준다.



(a) 삼각형 (b) 상반형

(c) 구형 (d) 직각형

(그림 1) 2차원 기하영역 샘플링의 예

3.2.2 영역 질의의 선택율 추정 방법

선택율은 cosine 함수가 연속 함수이므로 역 DCT의 적분을 이용하여 계산한다. 역 DCT 계산은 인접한 히스토그램 버켓 사이의 보간을 지원한다. 직각 영역 질의의 선택율을 계산하는 적분식을 구하기 위해, 먼저 2차원의 경우를 보이고 이것을 k차원의 것으로 일반화한다. q_2를 2차원의 직각 영역 질의라고 하고 그 영역이 $a \le x \le b, c \le y \le d$라고 하자. 데이터 공간은 $(0,1)^n$로 정규화 되어 있다고 가정한다. x축은 N개로 분할 되어 있고 y축은 M개로 분할 되어 있다고 하자. 그러면 x축과 y축의 i번째 값 x_i와 y_i는 다음과 같다.

$$x_i = \frac{2i+1}{2N}, \quad y_i = \frac{2i+1}{2M} \tag{5}$$

그러면 3.1절이 있는 역 DCT 함수 $f(m, n)$를 다음과 같이 다시 쓸 수 있다.

$$f(x,y) = \sqrt{\frac{2}{M}} \sum_{u=0}^{M-1} k_u \left\{ \sqrt{\frac{2}{N}} \sum_{v=0}^{M-1} k_v g(u,v) \cos(xv\pi) \right\} \cos(yu\pi) \tag{6}$$

직각 질의 q_2의 선택율 $= \int_c^d \int_a^b f(x,y)\,dx\,dy$ (7)

$$= \int_c^d \int_a^b \sqrt{\frac{2}{M}} \sum_{u=0}^{M-1} k_u \left\{ \sqrt{\frac{2}{N}} \sum_{v=0}^{M-1} k_v g(u,v) \cos(xv\pi) \right\} \cos(yu\pi)\,dx\,dy$$

$$= \int_c^d \sqrt{\frac{2}{M}} \sum_{u=0}^{M-1} k_u \left\{ \int_a^b \sqrt{\frac{2}{N}} \sum_{v=0}^{M-1} k_v g(u,v) \cos(xv\pi)\,dx \right\} \cos(yu\pi)\,dy$$

$$\approx \sqrt{\frac{2}{M}} \sqrt{\frac{2}{N}} \sum_{g(u,v)\in Z} k_u k_v g(u,v) \int_c^d \cos(u\pi y)\,dx \int_a^b \cos(v\pi x)\,dy$$

여기서 Z는 영역 샘플링으로 선택된 계수들의 집합이다.

이제, 위의 적분식을 k차원으로 일반화한다. q_k를 k차원의 직각 질의라고 하고 그 영역은 $a_i \le x_i \le b_i, 1 \le i \le k$이고 x_i 축은 N_i개로 분할되어 있다. 그러면 선택율은 식 (8)과 식 (9)로써 표현된다.

질의 q_k의 선택율

$$= \int_{a_k}^{b_k} \cdots \int_{a_2}^{b_2} \int_{a_1}^{b_1} f(x_1, x_2, \cdots, x_k)\,dx_1\,dx_2 \cdots dx_k \tag{8}$$

$$\approx \sqrt{\frac{2}{M}} \cdots \sqrt{\frac{2}{N_k}} \sum_{g(u_1,\cdots,u_k)\in Z} k_{u_1} \cdots k_{u_k} g(u_1, \cdots, u_k) \int_{a_1}^{b_1} \cos(u_1\pi x_1)\,dx_1 \cdots \int_{a_k}^{b_k} \cos(u_k\pi x_k)\,dx_k \tag{9}$$

3.3 정확도를 개선하기 위한 블록 분할 히스토그램 방법

실제로 DCT 방법을 적용하는 이미지 압축 방법인 JPEG에서는 블록으로 분할하여 압축하고 있다. 이미지의 각 블록은 영역의 범위가 작기 때문에 인접한 픽셀간의 값이 크게 다르지 않는 경향이 나타나는데 이를 이용하면 변화가 심한 이미지도 효율적으로 압축할 수 있다. 따라서 데이터 분포의 편중도가 높다면 여러 개의 블록으로 분할하여 압축하면 좀 더 좋은 압축율을 얻을 수 있을 것이다. 다음은 이러한 원리를 적용하는 다차원 선택율 추정 방법을 설명한다.

블록 분할 방법은 전체 다차원 데이터 공간을 일정한 수의 블록으로 먼저 분할하고 각 블록에 대해서 독립적으로 다차원 버켓으로 분할한 후 각 블록 별로 DCT 계수들을 구한다.

이 방법은 여러 개의 블록으로 데이터 공간을 나누기 때문에 전체 데이터 공간의 편중도가 높다고 하더라고 작은 블록내의 편중도는 작아지므로 에러율을 더 줄일 수 있다. 또한 이 방법은 블록내의 DCT 계수 만을 계산하기 때문에 계산 속도가 매우 빠르다. 그리고 블록내의 DCT 계수의 개수도 적으므로 원래 데이터로부터 DCT 계수를 계산하는 데 걸리는 시간이 현저히 작아진다는 장점이 있다. 그러나 블록 분할 방법은 전체 저장되는 DCT 계수의 수가 증가할 수 있다. 질의와 겹치는 블록만으로 질의에 대한 선택율을 계산하므로 질의의 크기가 작으면 계산 속도가 매우 빨라지고 질의 크기가 크면 그 질의와 겹치는 블록의 개수가 증가하므로 계산 속도가 느려진다.

블록 분할 방법의 선택율 계산식은 아래와 같이 약간 변형된다. 비분할 방법은 블록 분할 방법에서 한 개의 블록만 있는 특수한 경우로 볼 수 있다. 한 블록내의 DCT 계수는 그 블록 안의 데이터 분포만을 근사 시키기 때문에 질의 영역과 겹치지 않는 블록에 대해서는 계산할 필요가 없다. 따라서 질의 영역과 겹치는 블록의 DCT 계수들 만으로 선택율을 계산하면 식 (10), 식 (11)과 같다.

질의 q_k의 선택율

$$= \int_{a_k}^{b_k} \cdots \int_{a_2}^{b_2} \int_{a_1}^{b_1} f(x_1, x_2, \cdots, x_k) dx_1\,dx_2 \cdots dx_k \tag{10}$$

$$\approx \sum_{j\in B} \sqrt{\frac{2}{N_{j1}}} \cdots \sqrt{\frac{2}{N_{jk}}} \sum_{g(u_{j1},v_{jk})\in Z_j} k_{u_{j1}} \cdots k_{u_{jk}} g(u_{j1}, \cdots, u_{jk}) \int_{a_{j1}}^{b_{j1}} \cos(u_{j1}\pi x_1)\,dx_1 \cdots \int_{a_{jk}}^{b_{jk}} \cos(u_{jk}\pi x_k)\,dx_k \tag{11}$$

여기서 B는 질의 영역과 겹치는 블록의 집합이고 Z_j는 j번

째 블록의 선택된 계수들의 집합이다. 적분 영역은 질의 영역이 블록과 겹치는 부분이므로 (a_{ji}, b_{ji})는 j번째 블록의 i번째 차원에서 겹치는 부분의 구간을 나타낸다.

4. 저장 공간 요구 사항과 계산 시간

제안된 방법은 요약 데이터를 저장하는 메모리 공간을 필요로 한다. 그 크기는 영역 샘플링 방법으로 선택된 DCT 계수의 수에 비례한다. 우리는 DCT 계수의 다차원 인덱스를 일차원의 값으로 변환했다. 그 역도 가능하다. DCT 계수의 값을 저장하는 데는 4바이트를 사용하고 인덱스를 저장하는 데는 8바이트 정수를 사용했다. 따라서 한 개의 DCT 계수를 저장하는 데는 12바이트의 저장 공간이 필요하다. 만약 n개의 DCT 계수를 사용한다면 12n바이트와 기타 정보를 저장하는 약간의 추가 바이트가 필요하게 된다.

계산 시간에는 선택율 계산시간과 DCT 계수 계산시간이 있다. 식 (9)의 계산 공식으로부터 다음과 같이 선택율 계산 시간을 추정할 수 있다. k는 차원이고 a는 sine을 계산하는 시간이라 하고 z은 영역 샘플링에 의해서 선택된 DCT의 개수라고 하면, 선택율을 계산하는 시간은 $2akz$으로 주어진다. a는 상수이므로 $O(kz)$이다. Intel Pentium Ⅲ 650MHz CPU에서 a는 약 0.2μsec로 측정되었다. 블록분할 방법에서는 질의 영역이 겹쳐지는 블록의 DCT 계수만으로 선택율을 계산한다. B는 질의 영역과 겹치는 블록의 집합이고 z_j는 j번째 블록의 선택된 계수의 수라고 하면, 선택율 계산시간은 $\sum_{j\in B} 2akz_j$으로 주어진다. 질의의 크기가 작으면 $\sum_{j\in B} z_j < z$이므로 블록 분할 방법의 선택율 계산 시간이 훨씬 적다.

3.1절에서와 같이 DCT계수를 계산하는 식은 2차원에서는

$$g(u,v)=\sqrt{\frac{2}{M}}k_u\sum_{m=0}^{M-1}\left\{\sqrt{\frac{2}{N}}k_v\sum_{n=0}^{N-1}f(m,n)\cos\left[\frac{(2n+1)v\pi}{2N}\right]\right\}\cos\left[\frac{(2m+1)u\pi}{2M}\right]$$

로 주어지고 다차원에서는

$$G(u(t)=\sqrt{\frac{2}{N_t}}k_{u_t}\sum_{n_t=0}^{N_t-1}G(u(t-1)\cos\left(\frac{(2n_t+1)u_t\pi}{2N_t}\right)$$

로 주어진다. 위 식에서 알 수 있듯이 한 개의 DCT 계수를 계산하려면, 차원이 k이고 i번째 차원의 분할 수가 N_i일때 $O(k\prod_{i=1}^{k}N_i)$의 cosine 계산이 필요하다. 그러나 $f(n_1,\cdots,n_k)$ 값이 0이 아닌(즉 데이터가 있는) 버켓의 개수는 $O(\prod_{i=1}^{k}N_i)$ 보다 훨씬 적다. 즉 데이터가 있는 버켓의 수를 s라 하면 $O(ks)$번의 cosine 계산이 필요하다. DCT 계수를 계산하는 방법은 두 가지를 적용하였다. 저 차원인 경우, 버켓의 개수가 전체 데이터의 개수에 비해 크지 않으면 버켓의 배열을 만들고 모든 데이터를 읽어서 버켓 안의 데이타 개수를 계산한다. 고차원인 경우, 전체 데이터의 개수보다 버켓의 개수가 더 많아진다. 이 경우에는 데이터가 있는 버켓을 유지하기 위해서 해싱 기법을 사용한다. 먼저 데이터를 읽고 그 데이터의 해당하는 버켓을 해싱으로 테이블에서 찾아보고 없으면 신규로 버켓을 테이블에 저장하고 1로 초기화한다. 만약 이미 테이블에 등록되어 있는 버켓이라면 단순히 1 증가시킨다. 그러면 데이터가 있는 버켓 만을 테이블에 저장할 수 있어서 저장 공간의 필요를 줄일 수 있다.

한개의 DCT 계수를 계산하는 시간이 $O(ks)$이므로 전체 DCT 계산 시간은, 영역 샘플링으로 선택된 DCT 계수의 개수가 z라면, $O(zks)$이다. 블록 분할 방법을 적용하는 경우를 보자. v는 블록의 수이고 j번째 블록의 i번째 차원의 분할 수가 N_{ji}이면, j번째 블록에서 한 개의 DCT 계수를 계산하는 시간은 $O\left(k\prod_{i=1}^{k}N_{ji}\right)$로 주어진다. 그런데 데이터가 전혀 없는 버켓에 대해서는 계산할 필요가 없으므로 s_j를 j번째 블록의 데이터가 있는 버켓의 수라고 하면, j번째 블록에서 한개의 DCT 계수를 계산하는 시간은 $O(ks_j)$로 주어진다. 따라서 j번째 블록의 전체 DCT 계산 시간은 $O(z_j ks_j)$으로 주어지고 데이터 공간 전체에 대한 DCT 계산시간은 $O\left(\sum_{j=1}^{v} z_j ks_j\right)$로 주어진다. 그런데 $s_j \approx \frac{s}{v}$이고 $z_j \approx \frac{z}{v}$이므로 $O\left(\sum_{j=1}^{v} z_j ks_j\right) \approx \frac{1}{v}O(zks)$이다. 따라서 블록 분할 방법의 DCT 계산 시간이 매우 적다는 것을 알 수 있다.

5. 실 험

5.1 실험데이터와 에러 메트릭

이 논문에서 제안된 방법의 정확도를 측정하기 위해서 실제 데이터와 여러 가지 분포의 다양한 합성 데이터를 준비하고 다양한 실험을 하였다. 실험 환경으로는 256M의 주메모리, 20G 하드디스크를 가진 Windows98 운영체제하의 Intel Pentium Ⅲ 650MHz PC에서 실험하였다. 실제 데이터로는 비트맵 이미지 데이터를 사용하였다. 이미지의 수는 6만개를 사용하였고 각 이미지 데이터로부터 4×4 RGB 특징을 추출하였다. 특징의 차원은 4×4×3 = 48차원의 데이터가 된다. 48차원의 데이터를 KL 변환을 사용하여 3차원에서 6차원의 데이터로 차원 축소하였다. 이미지의 종류는 동식물, 패턴, 건축물, 풍경, 인물, 도형 등등의 각종 이미지를 사용하였다. 합성 데이터의 데이타의 개수는 10만개이며, 2~6차원의 데이터이며 정규분포와 Zipf 분포의 두가지를 사용하였다. 편중도의 정도를 변화시키기 위해서 정규분포의 σ값은 0.07~0.5의 범위를 사용하였고 Zipf 분포의 z 패러

미터는 0.2~0.6의 값을 사용하였다. z 패러미터는 큰 값일수록 데이터의 편중도가 커지는 것을 의미하고 작은 값일록 데이터의 편중도가 작아지는 것을 의미한다.

이 논문에서 제안된 선택율 추정 방법은 $(a_1 \le X_1 \le b_1)$ $\&\cdots\&(a_n \le X_n \le b_n)$, $0 \le a_i, b_i \le 1$인 영역 질의에 대하여 계산하였다. 실험 결과로서 30개의 같은 크기의 질의의 평균을 취하였고 계산으로 추정된 결과와 비교하였다. 추정치의 정확도를 측정하기 위해서 다음과 같이 정의되는 퍼센트 에러를 사용하였다.

$$\text{퍼센트 에러} = \frac{|\text{실제 질의 결과} - \text{추정된 결과}|}{\text{실제 질의 결과}} \times 100\%$$

블록 분할 방법을 위해서 다차원 데이터 공간을 <표 1>에서 보여준 대로 분할하였다.

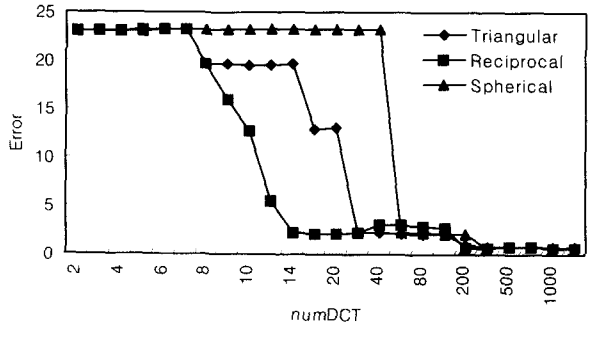
〈표 1〉 블록 분할 방법에서 차원별 블록 분할 수

	block 수	#DCT/block	전체 DCT 수
dim = 2	10×10 = 100	66	6600
dim = 3	5×5×4 = 100	74	7400
dim = 4	4×3×3×3 = 108	63	6804
dim = 5	3×3×3×2×2 = 108	61	6588
dim = 6	3×3×2×2×2×2 = 144	76	10944

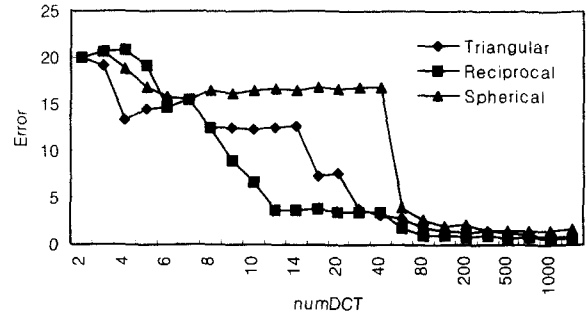
5.2 실험 결과

5.2.1 영역 샘플링의 효과

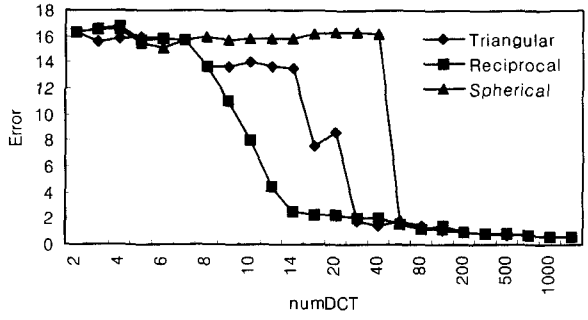
영역 샘플링은 저주파 필터링 역할을 하여 저주파의 계수들을 선택한다. 이 방법의 효율성은 최소 자승 에러(mean square error)로서 측정된다. 그러나 이것은 역 DCT에 의한 균일 히스토그램 버켓의 모든 값을 구해야 하므로 많은 시간이 소요된다. 그래서, 영역 샘플링의 효율성을 질의의 에러율로써 측정하였다. 30개의 질의에 대한 실험 결과들에 대한 평균을 구하였다. 영역 샘플링의 효율은 분포에 의해 영향을 받는다. 본 논문에서는 세 개의 다른 분포를 6차원으로 실험하였다. ① 정규분포 ② Zipf 분포 ③ 군집 15분포. 세 가지 영역 샘플링 방법을 이 데이터에 적용하였다. 직각형 샘플링으로 선택되는 DCT 계수의 개수는 b의 값에 따라



(그림 2) 정규분포, dim = 6, one-dim partition = 10, DCT 계수별 영역 샘플링 방법 비교



(그림 3) Zipf 분포, dim = 6, one-dim partition = 10, DCT 계수별 영역 샘플링 방법 비교

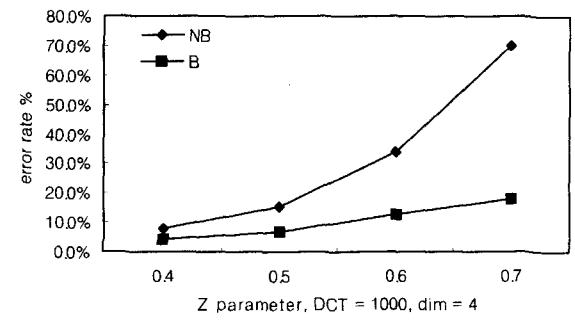


(그림 4) 군집 15 분포, dim = 6, one-dim partition = 10, DCT 계수별 영역 샘플링 방법 비교

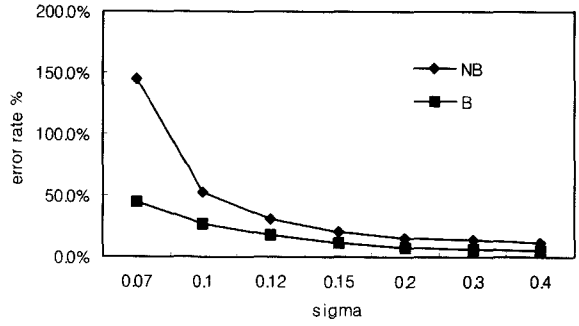
매우 빠른 속도로 증가하므로 제외하였다. (그림 2), (그림 3), (그림 4)에서 보는 바와 같이, 상반 샘플링이 모든 분포에서 제일 좋은 결과를 나타내었고 삼각 샘플링이 두 번째로 좋은 결과를 보여주며, 구형 샘플링이 가장 나쁜 성능을 보여 주었다. 그러나 어느 임계치를 넘으면 세 가지 방법 사이에 차이가 없었다.

5.2.2 블록 분할의 효과

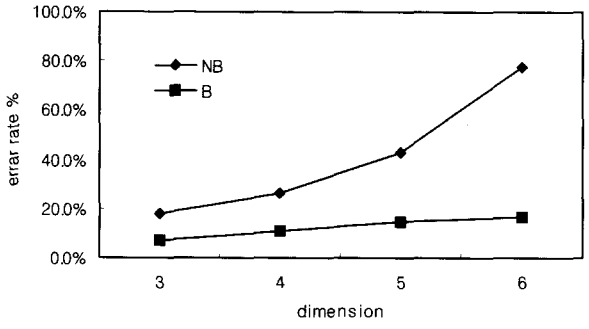
(그림 5), (그림 6), (그림 7)은 정규분포, Zipf 분포, 실제 데이터에 대한 블록 분할 효과를 보여주고 있다. Zipf는 z parameter의 값이 커질수록 편중도가 심해지는 것이고 정규분포는 σ값이 작을수록 편중도가 커진다. 실험의 결과는 편중도가 커질 수록 블록 분할 방법에 의한 에러율 감소효과가 크게 나타나고 있다는 것을 보여주고 있다.



(그림 5) DCT = 1000, 4차원, Zipf 분포의 z 패러미터별 비분할과 블록 분할의 비교



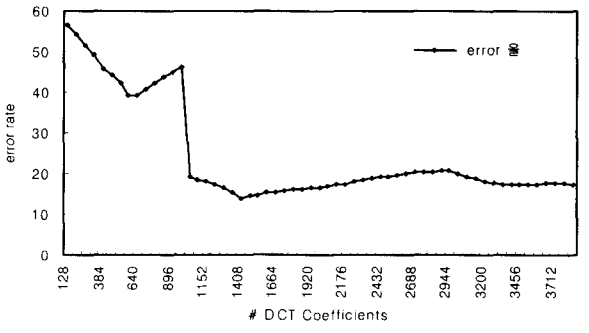
(그림 6) DCT = 1000, 4차원, 정규분포의 σ값에 따른 비분할과 블록 분할의 비교



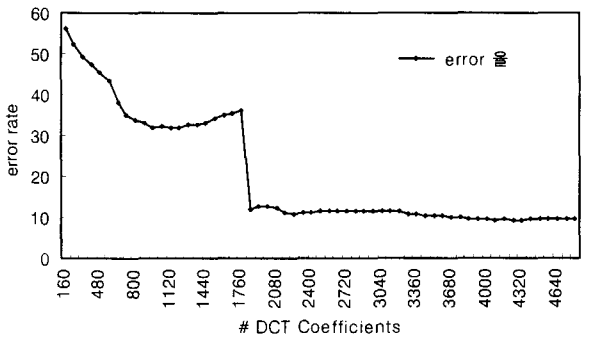
(그림 7) 실제 데이터에서 차원별 비분할과 블록 분할의 비교

5.2.3 블록 분할 규모의 효과

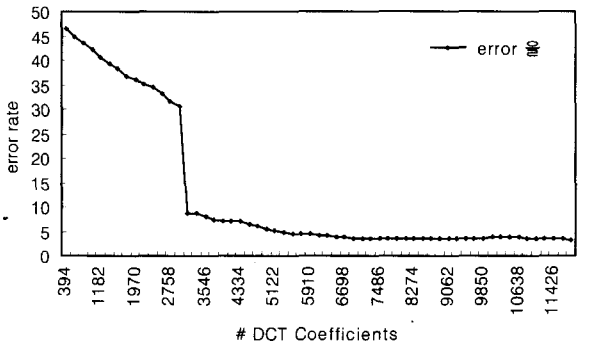
(그림 8), (그림 9), (그림 10)은 블록분할의 규모에 따른 에러율의 영향을 보여준다. 이미지 데이터의 6차원 특징 데이터를 사용하고 데이터 공간을 2×2×2×2×2×2, 3×3×3×2×2×2, 3×3×3×3×3×3로 나누었다. 질의 크기는 large query로 하였다. 실험 결과에서 보듯이 블록을 더 많이 나눌수록 에러율이 줄어드는 것을 알 수 있다. 또한 특이한 점은 DCT 계수의 수가 증가할수록 에러율이 줄어드는데 어느 시점에서 갑자기 떨어지고 그 이후부터는 DCT 계수의 수가 증가하여도 에러율은 더 이상 줄어들지 않는다는 사실이다. 이러한 현상은 DCT 계수의 크기가 큰 것부터 정렬되어 있고 어느 한계 위치 이후의 DCT 계수들은 더 이상 에러율에 영향을 미치지 못하기 때문인 것으로 해석된다.



(그림 8) 블록 = 2×2×2×2×2×2, 이미지데이터, Large query, dim = 6



(그림 9) 블록 = 3×3×2×2×2×2, 이미지데이터, Large Image, dim = 6



(그림 10) 블록 = 3×3×3×3×3×3, 이미지데이터, large query, dim = 6

6. 결 론

이 논문에서 우리는 다차원 선택율을 추정하기 위한 새로운 방법을 제안하였다. 히스토그램 방법은 에러율을 낮추려면 버켓의 크기를 줄여야 한다. 따라서 히스토그램 방법은 고차원에서는 엄청난 양의 저장 공간을 필요로 하기 때문에 적합하지 않다. 이런 문제의 해결을 위해 압축율이 뛰어난 DCT를 사용하였다. 본 논문에서 제시한 방법은 작은 크기의 버켓을 사용함으로써 높은 정확도를 달성하였고 적은 양의 압축 정보만을 가지므로 저장 공간문제를 해결하였다. 실험 결과는 다음의 잇점을 입증해 주고 있다. ① 이전의 방법들은 고차원 선택율을 지원하지 못하였으나, 이 논문의 방법은 고차원 선택율을 높은 정확도로 지원할 수 있다. ② 블록분할방법을 적용하여 압축정보를 계산하는 시간을 현저히 줄인다.

향후의 연구과제로서 멀티미디어 질의와 같이 질의의 형태가 구형인 질의의 선택율 추정에 관한 연구가 있다.

참 고 문 헌

[1] A. Belussi, C. Faloutsos, "Estimating the Selectivity of Spatial Queries Using the 'Correlation' Fractal Dimension," *21th VLDB Conference*, 1995.

[2] C. Chen. N. Roussopoulos, "Adaptive Selectivity Estimation Using Query Feedback," *ACM SIGMOD Conference*, pp. 161-172, 1994.

[3] W. Chang, G. Sheikholeslami, A Zhang, T. Syeda-Mahmood, "Efficient Resource Selection in Distributed Visual Information Systems," *ACM Multimedia Conference*, pp. 203-213, 1997.

[4] S. Chaudhuri, L. Gravano, "Optimizing Queries over Multimedia Repositories," *ACM SIGMOD Conference* pp.91-102, 1996.

[5] S. Christodoulakis, "Estimating record selectivities," *Information Systems Journal*, Vol.8, No.2, pp105-115, 1983.

[6] P. J. Haas, J. F. Naughton, S. Seshadri and L. Stokes, "Sampling based estimation of the number of distinct values of an attribute," *21th VLDB Conference*, 1995.

[7] Y. Ioannidis, "Universality of Serial Histograms," *19th VLDB Conference*, pp.256-267, 1993.

[8] Y. Ioannidis, V. Poosala, "Balancing Optimality and Practicality for Query Result Size Estimation," *ACM SIGMOD Conference*, pp.233-244, 1995.

[9] H. Jagadish, N. Kouda, S. Muthukrishnan, V. Poosala, K. Sevcik, T. Suel, "Optimal Histograms with Quality Gurantees," *24th VLDB Conference*, pp.275-286, 1998.

[10] J. S. Lim, "Two Dimensional Signal And Image Processing," *Prentice Hall*, 1990.

[11] Vitter, J. S., Wang, M. and Iyer, B., "Data Cube Approximate and Histograms via Wavelets," *In Proceedings of seventh International Conference on Information and Knowledge Management*, ACM Press, Washington D.C., pp. 96-104, 1998.

[12] M. V. Mannino, P. Chu and T. Sager, "Statistical profile estimation in database systems," *ACM Computing Surveys*, Vol.20, No.3, 1988.

[13] V. Poosala, Y. E. Ioannidis, P. J. Haas, E. J. Shekita, "Improved Histograms for Selectivity Estimation of Range Predicates," *ACM SIGMOD Conference*, pp.294-305, 1996.

[14] V. Poosala, Y. E. Ioannidis, "Selectivity Estimation Without the Attribute Value Independence Assumption," *23th VLDB Conference*, pp.486-495, 1997.

[15] K. R. Rao, P. Yip, "Discrete Cosinc Transform Algorithms, Advantages, Applications," *Academic Press*, 1990.

[16] Shanmugasundaram, J., Fayyad, U. and Bradley, P. S., "Compressed Data Cubes for OLAP Aggregate Query Approximation on Continuous Dimensions," *In the proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, San Diego, pp.223-232, 1999.

[17] W. Sun, Y. Ling, N. Rishe and Y. Deng, "An Instant and accurate size estimation method for joins and selections in a retrieval-intensive environment," *ACM SIGMOD Conference*, 1993.

이 주 홍

e-mail : juhong@inha.ac.kr

1983년 서울대학교 컴퓨터공학과(학사)

1985년 서울대학교 컴퓨터공학과(석사)

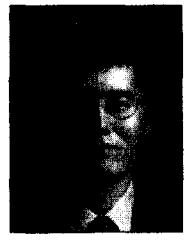
2001년 한국과학기술원 정보및통신공학과(박사)

1985년~1989년 한국통신 사업지원단 전임연구원

1989년~1993년 한국아이비엠 소프트웨어 연구소 선임프로그래머

2001년~현재 인하대학교 컴퓨터공학부 조교수

관심분야 : 소프트컴퓨팅과 데이터마이닝, 데이터 웨어하우스와 OLAP, 데이터베이스, 웹 서비스, 정보검색



전 석 주

e-mail : chunsj@ansan.ac.kr

1987년 경북대학교 전자공학과 컴퓨터공학전공(학사)

1989년 경북대학교 대학원 전자공학과 컴퓨터공학전공(석사)

2002년 한국과학기술원 정보및통신공학과(박사)

1989년~1995년 현대중공업 중앙연구소 주임연구원

1997년~현재 안산1대학 인터넷정보과 조교수

관심분야 : 데이터마이닝, 데이터 웨어하우스와 OLAP, 멀티미디어 데이터베이스 등

박 선

e-mail : sunpark@inha.ac.kr

1996년 전주대학교 전자계산학과 전공(학사)

2001년 한남대학교 정보산업대학원 정보통신과 정보통신전공(석사)

2002년 인하대학교 대학원 전자계산공학과 박사과정

관심분야 : 데이터마이닝, 데이터 웨어하우스, 분산병렬처리, 정보검색