# 공간 데이터베이스를 위한 효율적인 객체 교체 방법

민준기°
한국기술교육대학교 인터넷미디어 공학부
jkmin°@kut.ac.kr

# An Effective Object Replacement for Spatial Database Systems

Jun-Ki Min°
Korea University of Technology and Education, School of Internet-Media Engineering

요 약

버퍼의 효율적인 관리는 시스템의 성능과 밀접한 관련성을 가지고 있다. 따라서, 최근까지 다양한 버퍼 관리 기법들이 제안되었다. 그러나, 지금까지 알려진 기법들의 대부분은 시간 근접성만을 고려하고 있다. 그러나, 공간 데이터베이스와 같은 환경에서는 시간 근접성뿐 만 아니라, 유사한 위치에 있는 공간 객체들은 서로 같이 접근 될 가능성이 높다는 공간 근접성도 존재한다. 따라서, 본 논문에서는 공간 데이터베이스 환경에서 시간 근접성과 공간 근접성 같이 효과적으로 고려하는 버퍼 관리 기법을 제안하다. 또한, 실험을 통하여 제안 기법의 효율성을 보인다.

## 1. 서 론

버퍼의 효율적 관리는 데이터베이스 시스템의 성능에 매우 밀접하게 관련되어 있어서, 오랫동안 많은 연구자들에 의하여 다양한 버퍼 관리 기법들이 제안되어 왔다.

전통적인 데이터베이스 시스템은 숫자, 문자와 같은 정형된 구조의 데이터를 효율적으로 저장, 관리 할 수 있도록 되어 있다. 따라서, 빌딩, 도로 등과 같은 공간 객체를 효율적으로 저장 관리하기 위하여 공간 데이터베이스 관리 시스템이 등장하게 되었으며, 이러한 공간 데이터를 효율적으로 접근하기 위한 인덱스 기법 질의 처리 기법 등에 대한 연구가 지속적으로 진행되고 있다.

특히, 지금까지 잘 알려진 버퍼 관리 기법들은 현재 자주 접근되는 데이터 아이템은 앞으로도 자주 사용될 가능성이 높다라고 하는 시간 근접성 (temporal locality) 을 고려하고 있다. 그러나 공간 데이터베이스에서는 위치적으로 자주 접근 되는 데이터 아이템이 앞으로도 자주 접근될 가능성이 높다는 공간 근접성 (temporal locality) 이 존재한다.

본 논문에서는 기존의 버퍼 관리에서 유용한 척도로 사용되었던 시간 근접성과 공간 데이터베이스에서 발생되는 공간 근접성을 통합적으로 활용하여 공간 데이터베이스의 버퍼 관리를 효율적으로 처리할 수 있는 새로운 버퍼 관리 기법인 BEAST(Buffer rEplacement Algorithm using Spatial and Temporal locality) 기법을 제안한다.

## 2. 관련 연구

버퍼 관리 기법의 핵심은 버퍼 교체 기법 (buffer replacement) 으로 버퍼가 가득 차 있는 환경에서 새로운 데이터 아이템을 버퍼에 적재시키고자 할 때, 어떠한 데이터아이템을 희생자 (victim)로 삼아서 교체하는 가이다. 본 절에서는 기존에 제안된 다양한 버퍼 교체들에 대하여 살펴본다.

### 2.1 전통적인 버퍼 관리 기법

지금까지 가장 많이 알려진 버퍼 관리 기법이 LRU (Least Recently Used) [2] 이다. LRU 기법은 최근에 사용된 데이터 아이템을 앞으로 다시 사용된다 라고 하는 단순한 경험론적 규칙(heuristic rule)을 기반으로 하고 있어서 다양한 형태의 데이터 접근 패턴을 효과적으로 지원하지 못한다는 단점이 존재한다. 이러한 문제점을 해결하기 위하여 LRU-K [9] 기법이 제안되었다. LRU-K 기법은 각 데이터 아이템마다 k개의 접근 이력을 유지하고 희생자를 찾는데 많은 비용이 발생한다는 단점이 존재한다. 따라서, Johnson과 Shasha 는 LRU-2 기법처럼 동작하면서도 효율적인 2Q [4] 기법을 제안하였다. 2Q 알고리즘은 최근에 접근 된 적이 없는 데이터아이템은 FIFO 로 동작하는 A1IN 큐에 관리하고 최근에 접근 된 적이 있는 데이터 아이템은 LRU로 동작하는 AM 큐에서 관리한다. 여기서 최근에 접근 되었는지 아닌지에 대한 정보를 A1OUT 큐에서 관리한다. 2Q 기법은 각 큐의 크기 값에 따라서 그 성능 좌우된다는 단점이 존재한다.

최근에 각광 받고 있는 데이터 마이닝 기법을 활용한 BROOM [13] 이라는 기법도 제안되었다. 이 기법은 offline 상에서의 학습 단계가 필요하다는 단점이 존재한다.

데이터 아이템에 대한 접근 이력(history)과 접근 횟수(frequency)를 통합적으로 고려하는 LRFU [7] 기법도 있다. LRFU는 조정 변수 λ 값에 따라서 LRU 또는 LFU와 유사하게 동작하게 된다.즉, LRFU의 성능은 λ 값에 결정된다.

최근에 조정 변수 없이 데이터 접근 패턴에 따라서 버퍼의 동작이 바뀌는 ARC [8] 기법이 제안되었다. ARC 기법은 버퍼를 LRU 버퍼 $B_1$과 LRU 버퍼 $B_2$ 로 나누어 관리한다. 여기서 버퍼 $B_1$과 $B_2$에 존재하지 않는 새로운 데이터 아이템이 접근되면 $B_1$에 적재하고, $B_1$ 또는 $B_2$에 존재하는 데이터아이템에 대한 접근이 발생하면 그 데이터아이템을 $B_2$에 적재한다. 여기서 버퍼 $B_1$, $B_2$의 크기는 변수 p에 의하여 지속적으로 변경된다.

여기서, ARC 기법은 학습률(learning rate) 에 따라서 p 값의 증감량을 조정할 할 수 있다. 즉 ARC의 기본 목표인 조정 변수의 제거를 이룩하지 못하고 있다.

또한, 총 접근 회수만을 고려하는 LFU의 일반화된 형태인 LFU-K [12] 기법 과 데이터 아이템의 참조 시간을 예측하고 그 정보를 활용하여 교체 대상을 선정하는 TNRP [5] 기법이 제안되었으며, 일반적인 데이터 아이템이 아닌 인덱스 아이템에 대한 버퍼 관리 기법으로 ILRU [11], GHOST [3] 등이 있다.

### 2.2 공간 데이터베이스를 위한 버퍼 관리 기법

기존의 시간 근접성만을 고려하는 기법들과 달리, 그러나, 공간 데이터베이스의 특성을 파악하고 이를 이용하여 공간 데이터베이스에 보다 적합한 버퍼 관리 기법들이 제안되었다.

Papadopoulos와 Manolopoulous는 LRD-Manhattan [10] 기법을 제안하였다. 공간 데이터베이스의 공간 객체의 MBR(minimum bounded rectangle)로 단순화 되어 표현된다. 또한 확률 상 MBR의 크기가 클수록 해당 데이터를 접근할 확률이 높아진다. 즉 단위 공간에서 임의의 점을 찍을 때 큰 공간 객체를 선택할 확률이 높아진다. 따라서, 이 기법은 각 공간 객체의 접근 밀도 (즉 전체 공간 데이터 접근 중 해당 공간 객체의 접근 비율)와 MBR의 정규화 크기의 산술 평균을 이용하여 최소값을 가지는 공간 객체를 교체 대상으로 삼는다.

또한, 최근에 LRU 규칙과 공간 데이터의 크기를 통합하여 고려하는 ASB [1] 기법이 제안되었다. 이 기법은 버퍼를 ARC 기법처럼 논리적으로 구분된 2개의 버퍼로 구성한다. 이중 B1 은 LRU특성을 이용하여 관리되며 B2에서는 공간 데이터의 크기와 같은 공간 데이터 특성을 이용하여 관리된다.

새로운 공간 객체는 B1 부분에 적재되며 이로 인하여 LRU 공간 부분이 모자라면 LRU 객체를 B2으로 넘긴다. 객체 교체가 필요한 상황에서는 버퍼의 B2부분에서 각 작은 크기의 공간 객체를 찾아서 희생자로 삼는다. 또한, 새롭게 접근되는 공간 객체의 특성을 파악하여 B1부분의 크기 및 B2 부분의 크기를 가변적으로 조정한다.

지금까지 살펴본 공간 데이터베이스를 위한 버퍼 관리 기법들은 공간 객체의 정적 특성 (즉, MBR의 크기)등 만을 고려함으로써, 동적인 객체 접근 패턴에 따른 효과적인 버퍼 관리 기법을 제공하지 못하고 있다.

## 3. 시간 및 공간 근접성을 고려한 버퍼 관리 기법

공간 데이터베이스에서는 시간 근접성뿐 만 아니라 특정 지역의 데이터가 자주 사용되는 공간 근접성 [6]이 나타난다. 즉, 임의의 영역이 자주 접근된다면, 해당 영 역 안에 존재하는 공간 객체들이 자주 사용될 확률이 높게 된다.

공간 데이터베이스에서 시간 근접성만을 고려한다면 자주 접근되는 영역 안에 존재하는 공간 객체를 희생자로 선정할 경우가 발생되어 버퍼의 효율성을 떨어뜨리게 된다. 또한, 공간 근접성만을 고려할 경우에는 무작위 접근과 같은 형태의 접근 패턴을 효율적으로 지원할 수 없게 된다.

따라서, 본 논문에서 제안하는 버퍼 관리 기법인 BEAST는 기존의 공간 데이터베이스를 위한 버퍼 관리기법과 달리, 공간 근접성과 시간 근접성을 통합적으로 고려하는 특징을 지닌다.

BEAST 관리 기법에서는 기본적인 경험적 규칙은 자주 접근되는 영역 안의 공간 객체들은 앞으로도 자주 사용된다 라는 것이다.

따라서, 가장 오랫동안 접근되지 않은 영역 내에서 가장 오랫동안 접근되지 않은 공간 객체를 우선 교체 대상으로 선정한다. 즉, 공간적으로나 시간적으로 재 참조될 가능성이 가장 적은 공간 객체를 교체 대상으로 삼음으로써 BEAST 기법이 기존의 버퍼 관리 기법들에 비하여 뛰어난 성능을 보여줄 것을 기대할 수 있다.

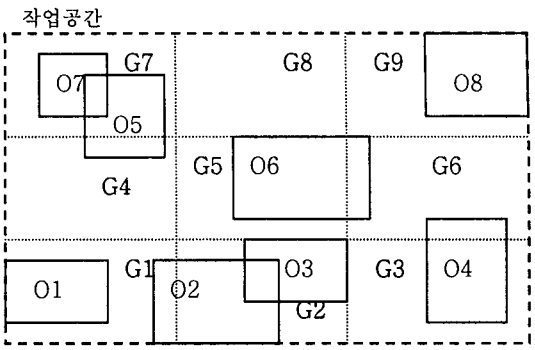


그림 1. 작업 공간 분할

BEAST 기법에서는 우선 그림 1와 같이 공간 객체들이 존재하는 작업 영역을 균등 크기의 NxN 격자(grid)들로 분할한다. 여기서, 각 공간 객체의 MRB의 중심점이 어는 격자에 안에 존재하는 가에 따라서 각 공간 객체가 소속된 영역이 결정된다.

예로써, 격자 G2에는 공간 객체 O2와 O3가 포함되며, G3에는 공간 객체 O4가 포함된다. 즉, 임의의 공간 객체가 어느 영역에 포함되는지를 손쉽게 파악할 수 있다. 이렇게 분할 된 공간 영역에서 어떠한 영역이 자주 사용되고 있는지를 파악하기 위하여 BEAST에서는 분할 된 공간 영역의 집합을 LRU 큐를 이용하 여 관리 한다.

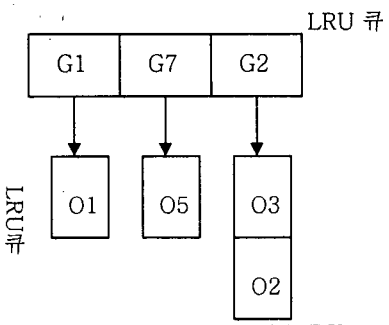


그림 2. BEAST 자료 구조

또한 각 영역 안에서 어떠한 공간 객체가 자주 사용되고 있는지를 파악하기 위하여 각 영역 안의 공간 객체들을 LRU 큐를 이용하여 관리한다. 즉, 그림 2에서 보이는 바와 같이, BEAST에서는 자주 사용되는 영역들의 정보를 관리 및 각 영역에서 자주 사용되는 공간 객체들의 정보 관리를 하는 2 단계 버퍼 관리 기법을 이용한다.

그림 2은 버퍼 크기가 4개의 공간 객체만 유지할 수 있다고 가정하였을 때의 BEAST의 구조이다. 그림 3에서 보게 되면 가장 최근에 접근된 영역은 G1이며, 이때 접근 된 공간 객체는 O1임을 파악할 수 있다.

그림 2과 같은 상태에서 새로운 공간 객체를 버퍼에 적재할 경우가 발생하게 되면 BEAST 기법에서는 우선 가장 오랫동안 접근되지 않은 영역 G2를 파악하고 버퍼 상에서 G2의 영역 안에 존재하면서 가장 오랫동안 접근되지 않은 객체 O2를 희생자로 선정하게 된다.

가장 오랫동안 사용되지 않은 영역을 파악하는 데 드는 비용은 LRU 기법과 같이 O(1)이다. 또한 가장 오랫동안 사용되지 않은

영역에서 가장 오랫동안 사용되지 않은 공간 객체를 찾는 비용도 O(1)이다. 즉, BEAST 기법의 희생자 선정 비용은 O(1)이다. 따라서, BEAST 기법은 시간 근접성과 공간 근접성을 통합적으로 고려하면서도 LRU 기법과 같이 효율적으로 공간 데이터베이스의 버퍼를 관리한다.

## 4. 실험과 분석

본 절에서는 제안된 BEAST 기법의 성능 실험을 다룬다. 실험에서는 합성 데이터와 실제 지도 데이터를 이용하여 BEAST 기법과 기존의 다양한 버퍼 관리 기법들과의 적중률을 비교하여 BEAST 기법의 우수성을 보였다.

| 객체 수 | 10000 |
|---|---|
| 평균객체 크기 | 80byte(최대 128byte) |
| 작업공간 크기 | 100000x100000 |
| 객체 분포 | 균등분포 |

표 1. 합성 데이터 속성

표 1은 본 실험에서 사용된 합성 데이터의 속성에 대하여 보여준다. 실제 데이터로서는 공간 데이터베이스 연구에서 많이 사용되고 있는 Tiger/Line[14]을 사용하였다.

본 실험에서 사용한 지역은 미국 캘리포니아 주 Kings 로 총 공간 객체 수는 21,853개이고 작업공간의 크기는 840681x7003660이다.

객체 접근 패턴으로는 총 세가지 접근 패턴을 이용하였다. 첫 번째는 모든 공간 개체의 접근 확률인 동일한 균등 접근이다. 두 번째는 Zipf 분포를 이용하여 총 객체들 중 20%에 총 접근 횟수 중 80%가 집중되도록 하여 시간 근접성이 발생하도록 하였다. 세 번째로는 작업공간 상의 10%의 영역 안에 존재하는 공간 객체들에 총 접근의 90%가 집중하도록 하여 시간 및 공간 근접성이 발생하도록 하는 접근 패턴을 만들었다.

본 실험에서는 지면의 제약으로 인하여 버퍼 크기가 전체 데이터 량의 10%일 때의 각 버퍼 관리 기법들의 적중률을 비교한다.

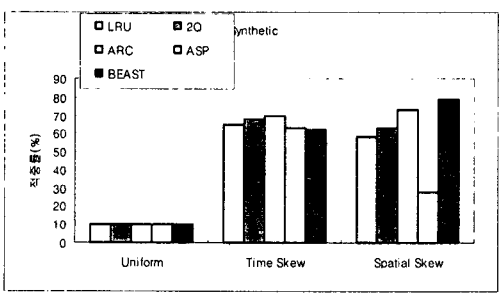


그림 3. 합성 데이터의 실험 결과

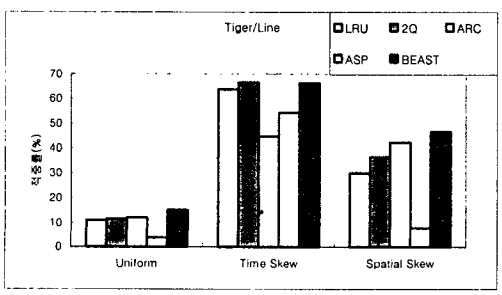


그림 4. 실제 데이터의 실험 결과

그림 3와 4에서 보듯이 균등 접근(Uniform) 및 시간 집중 접근(Time Skew)에서, 본 논문에서 제안하는 BEAST 기법은 가장 뛰어난 성능을 보이지는 않는다. 그러나, 기존의 시간 근접성 만을 고려하는 기법들인 LRU, 2Q, ARC에 비교하여, 시간 근접성 및 공간 근접성을 동시에 고려함에 있어서도 BEAST 기법은 뒤떨어지지 않는 우수한 성능을 보이고 있다. 또한 시간 및 공간 근접성이 동시에 발생하는 spatial skew 접근 패턴에서는 기존의 다른 기법들에 비하여 가장 높은 적중률을 보임을 알 수 있다. 따라서, 본 논문에서 제안하는 BEAST 기법은 공간 데이터베이스상에서 나타나는 다양한 접근 패턴에 대하여 효율적으로 버퍼를 관리할 수 있음을 알 수 있다.

## 5. 결론

데이터베이스에 있어서 효율적인 버퍼 관리는 디스크 I/O의 횟수를 최소하는 가장 중요한 요소로서 이에 대한 많은 연구가 있어왔다. 기존의 버퍼 관리 기법들은 시간 근접성만을 고려하여 버퍼의 교체 대상자를 선정하였다. 또한 최근에 각광 받고 있는 공간 데이터베이스를 위하여 고안된 기존의 버퍼 관리 기법들은 공간 객체의 크기와 같은 고정된 특성만을 고려한다. 따라서, 다양한 객체 접근 패턴을 효과적으로 지원하지 못하는 단점이 있다. 본 논문에서는 공간 객체 접근 시에 발생하는 시간 근접성과 공간 근접성을 통합적으로 고려하는 효율적인 공간 버퍼 관리 기법을 제안하였다. 또한 다양한 접근 패턴을 통한 실험을 통하여 제안된 버퍼 관리 기법의 효율성을 보였다.

## 참고문헌

[1] T. Brinkhoff, " A Robust and Self-tuning Page Replacement Strategy for Spatial Database Systems," EDBT Conference, pp. 533-552, 2002.

[2] W. Effelsberg, " Principles of Database buffer Management," ACM TODS, 9(4), pp.560-595, 1984.

[3] C. H. Goh, B. C. Ooi, D. Sim, K. Tan, " GHOST: Fine Granularity Buffering of Index," VLDB Conference, 1999.

[4] T. Johnson, and D. Shasha, " 2Q: a Low Overhead High Performance Buffer Management Replacement Algorithm, " VLDB Conference, pp.439-450, 1994.

[5] B. Juurlink, " Approximating the Optimal Replacement Algorithm," ACM CF Conference, 2004.

[6] L. Ki-Joune and L. Robert, " The Spatial Locality and a Spatial Indexing Method by Dynamic Clustering in Hypermap System," Advances in Spatial Databases, 1990, pp.207-223.

[7] D. Lee, J. Choi, J.-H. Kim, S. H. Noh, S. L. Min, Y. Cho, and C. S. Kim, " LRFU: A Spectrum of Policies that subsumes the Least Recently Used and Least Frequently Used Policies," IEEE Trans. Computers, v. 50(12), pp.1352-1360, 2001.

[8] N. Megiddo and D. S. Modha, " ARC: A Self-tuning, Low Overhead Replacement Cache, " USENIX FAST Conference, 2003.

[9] E. J. O' Neil, P. E. O' Neil, and G. Weikum, " The LRU-K Page Replacement algorithm for database disk buffering," ACM SIGMOD Conference, pp. 297-306, 1993.

[10] A. Papadopoulos, and Y. Manolopoulos, " Global Page Replacement in Spatial Databases," DEXA Conference, 1996.

[11] G. M. Sacco, " Index Access with a Finite Buffer," VLDB Conference, 1987.

[12] L. B. Sokolinsky, " LFU-K: An Effective Buffer Management Replacement Algorithm," DASFAA Conference, pp. 670-681, 2004.

[13] A.J. Tung, Y.C. Yay, and H. Lu, " BROOM: Buffer Replacement using Online Optimization by Mining, " CIKM conference, 1998, pp. 185-192.

[14] U.S. Census Bureau, " UA Census 2000 TIGER/Line Files," http://www.census.gov/geo/www/tiger/tigerua/ua_tgr2k.html.