공간 데이타 관리

2000. 5. 15
송병호

공간 데이타(spatial data) 객체

- 공간상의 특정 영역(공간 익스텐트)을 차지
- 공간 익스텐트(spatial extent) 표현:
  - 위치(location)
  - 경계(boundary)
- 종류: 점 데이타와 영역 데이타
- 공간 질의의 대상

공간 데이타 객체: 종류

- 점 데이타 (point data)
  - 위치만 있는 공간 익스텐트: 공간을 차지하지 않는다.
  - 다차원 공간상의 점으로 표시
  - 직접 측정된 점: 래스터 데이타(raster data) 등
  - 변환으로 얻은 점: 특징 벡터(feature vector) 등

공간 데이타 객체: 종류 (계속)

- 영역 데이타 (region data)
  - 위치와 경계가 모두 있는 공간 익스텐트
  - 대개 실제 객체의 기하학적인 표현
  - 위치: 중심점 등 기준점의 역할
  - 경계: 2D는 선, 3D는 면

공간 질의 (spatial query)

- 공간 범위 질의 (spatial range query)
  - 질의에 해당하는 영역 존재
  - 영역 데이타는 일부 중첩 or 완전 포함
- 최근접점 질의 (nearest neighbor query)
  - ranking
  - 멀티미디어 DB에서 가장 '유사한' 객체 구하기

공간 질의 (계속)

- 공간 죠인 질의 (spatial join query)
  - 저장된 투플들 간의 비교
  - "호수와 가까운 도시들을 구하라" 등
  - 처리비용이 많이 들며, 영역간에는 기준(경계 or 중심)이 문제

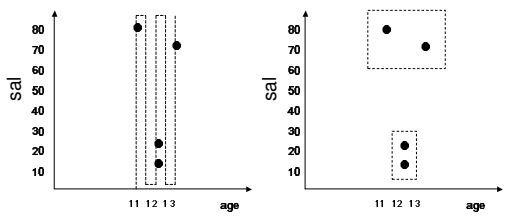
공간 데이타를 이용하는 응용

- 종래의 관계 DB: k개 필드 -> k차원
- 지리정보시스템(GIS)
- CAD/CAM
  - 공간 무결성 제약조건
- 멀티미디어 DB
  - 내용 기반 검색
  - 유사도에 부합하는 다차원 변환

공간 인덱스 소개

- 공간적 관련성에 따라 조직
- 탐색 키 필드가 k차원이면 k차원의 다차원 인덱스가 됨
- <나이, 급여>에 대한 1D/2D 인덱스 비교(그림 26.1)
- 점 데이타용: 그리드 화일, hB 트리, KD 트리, 점 사분 트리, SR 트리 등
- 영역/점 겸용: 영역 사분 트리, R 트리, SKD 트리 등

1D/2D 인덱스의 무리짓기 비교

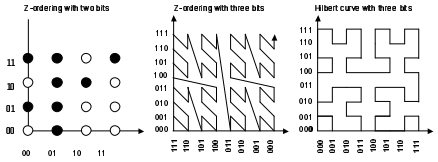


인덱싱 방안

- 공간 채움 곡선 이용법: 다차원을 1차원으로 배열
  - 영역 사분 트리 등
- 확장성 해싱법 이용: 디렉토리 구성
  - 그리드 화일 등
- 다차원 공간 반복 분할법
  - R 트리 등

공간 채움 곡선 (space filling curve)

- 다차원을 1차원(선형)으로 강제 순서화: Z순서 곡선(Z-ordering curve)이 대표적인 예 (그림 26.2)



공간 채움 곡선 (계속)

- Z값(Z-value)
  - 각 차원별로 차원값을 정해진 길이의 비트열로 표현한다. (가령 X=01, Y=11)
  - 각 차원값을 교차로 짜 집기(인터리빙)하면 Z값이 된다. (가령 0111=7)
  - 2D를 예로 든다면, 사분면 간에 Z 형태로 순서화되고, 각 사분면 안에서 다시 이 순서가 반복된다.

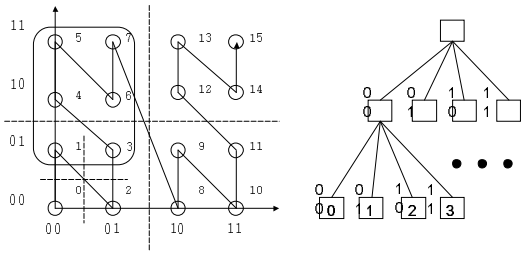
공간 채움 곡선 (계속)

- Z순서: 이러한 Z값 순서대로 나열하면, 공간적으로 인접한 점들이 가까이 무리짓게 된다. (가끔씩 대각선으로 멀리 '도약'하는 문제 있음)

영역 사분 트리 (region quad tree)

- 사분 트리: Z순서법이 공간을 반복 분할하듯이 분할해 나가는 트리 (그림 26.3)
- 영역 데이타 표현: 영역이 포괄하는 (세부)사분면에 모두 기록.
- 예-그림 26.3의 네모 부분 R:
  - <0001, R>, <0011, R>, <01, R>, 또는
  - <00, R>, <01, R>

영역 사분 트리의 모양



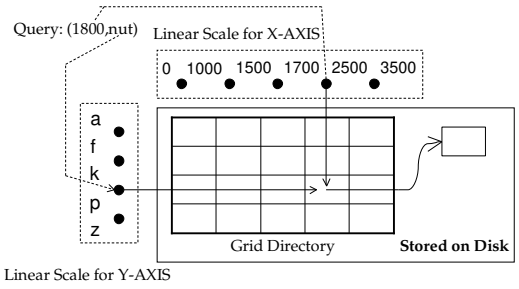
Z순서법에 따른 공간 질의

- 공간 범위 질의: Z값으로 표현되는 영역들에 대한 질의로 변환
- 최근접점 질의: 대각선으로 도약하는 문제를 고려할 것
  - 질의의 Z값에 가장 가까운 점(실제거리=r)을 찾는다.
  - 이 점보다 실제 공간에서 더 가까운 점이 없는가를 알아보기 위하여, 질의점을 중심으로 반경 r 인 구형 범위 질의 발부
  - 이 결과들의 실제 거리를 검사하여 가장 가까운 것을 선택.

그리드 화일 (Grid file)

- 어떤 점 질의도 최대 2번의 디스크 접근으로 처리 보장 (그림 26.4)
  - 각 축(차원)마다 하나의 선형 눈금자(linear scale): in-memory
  - 선형 눈금자 교차점마다 엔트리를 하나씩 가지는 그리드 디렉토리(grid directory)
  - 그리드 디렉토리 엔트리의 내용을 저장하는 디스크 페이지

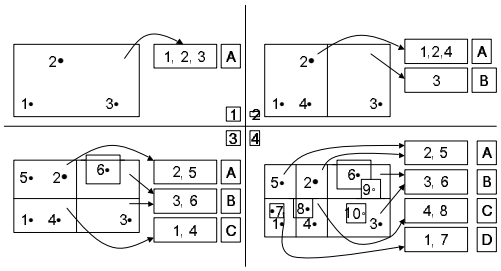
그리드 화일의 탐색 모양



그리드 화일: 분할과 합병

- 데이타 페이지가 가득 차면 분할 -> 선형 눈금자도 분할 (그림 26.5)
- 삭제로 점유율이 기준 이하로 떨어진 페이지는 합병: 복잡한 문제
  - 같은 페이지를 공유하는 디렉토리 엔트리들의 볼록꼴 요건

그리드 화일의 분할 모양



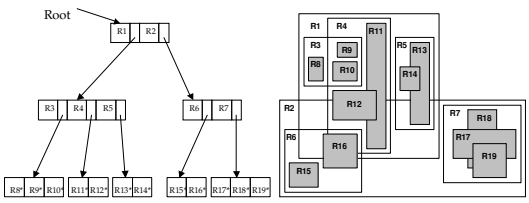
그리드 화일로 영역을 처리하려면

- 두 가지 방식이 있으나 비효율적
- 더 고차원 공간상의 점으로 표현하는 법 (예: 2차원 4각형을 4차원상의 한 점으로)
  - 원래 공간상의 거리와 변환 공간상의 거리가 무관함
- 해당 영역과 겹치는 모든 그리드 분할에 중복 저장하는 법

R 트리: 개요

- 일차원 B+ 트리를 공간 데이타에 적응시킨 형태: 높이-균형 구조(그림 26.6)
- 탐색 키: 각 차원당 한 구간씩, 여러 구간의 모임
  - 각 구간으로 둘러싸인 사각형 개념: 외접 사각형(bounding box)

R 트리를 보는 두가지 관점



R 트리의 노드 구성

- 단말 노드: 데이타 엔트리들로 구성
  - 데이타 엔트리 구조: <n차원 외접사각형, rid>
- 비단말 노드: 인덱스 엔트리들로 구성
  - 인덱스 엔트리 구조: <n차원 외접사각형, 자식 포인터>
- 인덱스 엔트리의 외접사각형들은 겹칠 수 있어도, 데이타는 한 단말 노드에만 들어간다!

R 트리: 질의

- 점 질의
  - 질의의 외접 사각형 B를 구한다.
  - 루트로부터 시작하여, B와 조금이라도 겹치는 경로를 모두 따라간다.
  - 어떤 단말 노드도 방문하지 않으면, 결과가 없다는 의미
- 영역 객체/범위 질의
  - 점 질의와 비슷
  - 단말 노드에서는, 착오 중첩(외접 사각형으로는 겹치지만 실제 객체는 겹치지 않는 경우)을 검사해 주어야 한다. (예: R8에 해당하는 범위 질의)

R 트리: 질의 (계속)

- 최근접점 질의
  - 질의 점에 대한 질의처럼 진행
  - 해당 단말 노드의 모든 점 중에서 최근접점 검사
  - 단말 노드가 나오지 않으면 질의점의 외접 사각형을 점차로 키워서 재시도

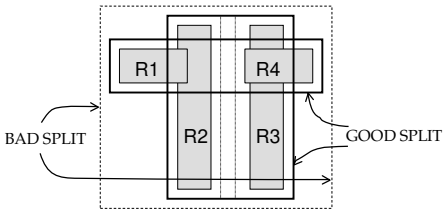
R 트리: 삽입과 삭제

- 데이타 객체 삽입: <외접사각형 B, rid>를 적당한 단말에 삽입
- 루트로부터 시작하여 B를 포함할 수 있는 가장 작은 자식으로, 단일 경로로 내려간다.
- 단말의 크기가 작은 경우에는 B를 포함할 수 있도록 외접 사각형 재조정: 상위로 파급

R 트리: 삽입과 삭제 (계속)

- 단말 노드 포화
  - 분할 및 재분배 - 상위로 파급: 노드간의 겹침을 최소화하도록 재분배 (그림 26.7)
  - R*트리: 일정 분량(30% 정도)을 삭제했다가 강제 재삽입
  - R+트리: 가능한 모든 단말에 삽입하고, 단일 경로로 검색
- 삭제:
  - 질의처럼 진행
  - 이론상, 단말의 외접사각형을 줄여주면 좋다

R 트리의 재분배 모양



일반형 탐색 트리

- generalized search tree: GiST
- 트리 인덱스 구조의 필수적인 형태를 추상화
- 삽입, 삭제, 탐색 알고리즘에 대한 틀(templet) 제공
- 시스템 코드를 고치지 않고도 B+, R 등 여러 트리 구조를 확장, 구현할 수 있음

고차원 인덱싱의 문제점

- 탐색 성능 저하
  - 수십 차원의 경우, 순차 검색이 더 효율적
- 유사도 변별력 저하
  - 차원이 높아질수록 최근접점의 거리와 최원격점의 거리간의 대비도(contrast)가 떨어짐
  - 대비도가 좋은 응용인지 확인할 필요가 있음