**Ludwig-Maximilians-Universität München**
**Institut für Informatik**
Dr. Matthias Renz
Tobias Emrich

München, 13.11.2012

# Neue Trends zur Suche in modernen Datenbanksystemen

Wintersemester 2012/13

## Übungsblatt 2: Distanzen / R-Bäume / Indexbasierte Anfragen / Multistep-Anfragen

Besprechung: 19.11.2011

### Aufgabe 2-1 *R-Bäume / Indexbasierte Anfragen*

(a) Gegeben sei eine räumliche Datenbank, indexiert durch einen R-Baum, illustriert in Abbildung 2. Füegen Sie zunächst in den R-Baum die folgenden Datenpunkte ein (Einfügestrategie: geringster Flächenzuwachs, Splitstrategie: geringster überdeckter toter Raum): $G = (14, 3), H = (7, 11), I = (9, 12)$.

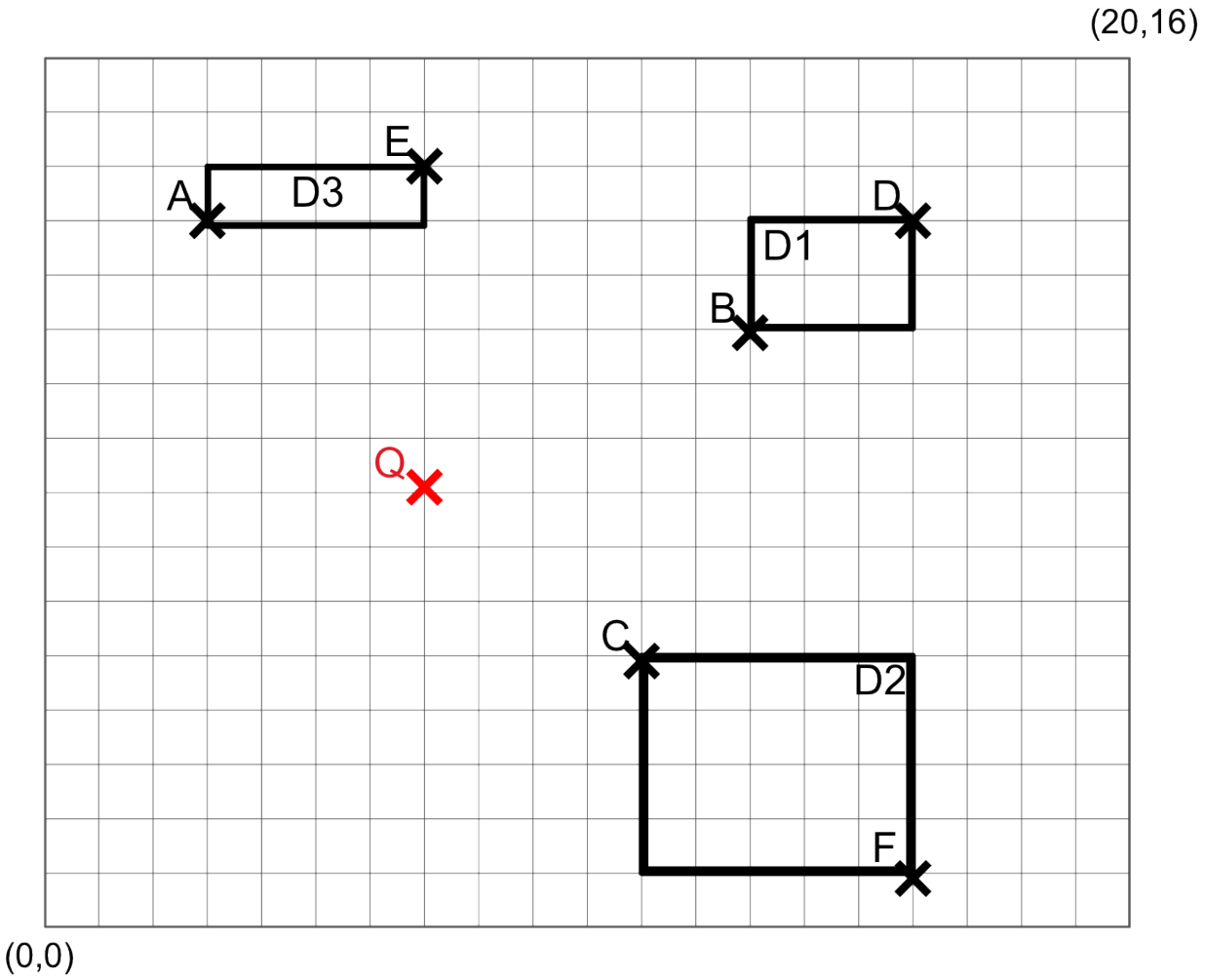


Abbildung 1: Illustration eines R-Baums.

(b) Geben Sie die Resultatmenge einer *RQ-Index-LB*-Anfrage (Algorithmus siehe unten) um $Q$ mit $\varepsilon = 8$ und der von der Manhattan-Norm induzierten Metrik $dist(x, y) = d_{L_1}(x, y) = \sum_{i=1}^{d} |x_i - y_i|$ (dabei sind $x, y \in \mathbb{R}^d$ Vektoren) an.

**Algorithm 1** RQ-Index-LB

```
Require: page, q, ε
    result = ∅
    p = page.loadPage();
    if p.isDataPage() then
        for all objects in p do
            if dist(q,p.getObject(i))≤ ε then
                result = result ∪ getObject(i);
            end if
        end for
    else
        for all regions in p do
            if MINDIST(q,p.getRegion(i))≤ ε then
                result = result ∪ RQ-Index(p.childPage(i),q,ε);
            end if
        end for
    end if
    return result;
```

Für die nächsten Aufgaben verwende wir folgende Notation:
Betrachte Distanzmaß, Filterdistanz(en) und Anfragepunkt $Q$ als gegeben und fest. Für ein Datenbankobjekt $P$ bezeichne $d_{LB}(P)$ die LB-Filterdistanz von $P$ zu Anfragepunkt $Q$ (analog für die UB-Filterdistanz, falls vorhanden), $d_{EX}(P)$ stehe für die exakte Distanz von $P$ zum Anfragepunkt $Q$. (Alle Distanzen beziehen sich auf $Q$, deshalb verzichten wir auf die Erwähnung.)
Außerdem stehe ein Ranking der Objekte gemäß ihrer LB-Filterdistanz zum Anfragepunkt zur Verfügung (vorstellbar als Heap, von dem die Objekte gepoppt werden, hier organisiert als sortierte Liste). Dabei sei $d_{LB}^{next}$ die Filterdistanz des gemäßdes Rankings nächsten Objekts. Schließlich bezeichne $d_{LB}^{k}$ die k-kleinste $d_{LB}$-Filterdistanz der Kandidatenmenge, und $d_{UB}^{k}$ die k-kleinste (!) $d_{UB}$-Filterdistanz der Kandidatemenge.

(c) Geben Sie die verschiedenen Zustände der APLbei einer *kNN-Index-LB-HS*-Anfrage (Algorithmus siehe unten) um $Q$ mit $k = 3$ und der von der Manhattan-Norm induzierten Metrik an.

**Algorithm 2** kNN-Index-LB-HS

```
Require: page, q
  result = LIST OF (dist : Real, obj: Object) OF LENGTH k ORDERED BY dist ASC;
  stopdist = pruningdist = ∞;
  APL = LIST OF (dist : Real, da : DistAddress) ORDERED BY dist ASC;
  APL = [0, page)]
  while ¬ APL.isEmpty() ∧ APL.getFirst().dist ≤ pruningdist do
    p = APL.getFirst().da.loadPage();
    APL.deleteFirst();
    if p.isDataPage() then
      for all objects in p do
        if dist(q,p.getObject(i)) ≤ stopdist then
          result.insert((dist(q,p.getObject(i)),p.getObject(i)));
          //element with largest distance is removed, if result.size = k
          if result.get(k).isDefined() then
            stopdist = result.get(k).dist;
          end if
        end if
        if stopdist < pruningdist then
          pruningdist = stopdist;
        end if
      end for
    else
      for all regions in p do
        if MINDIST(q,p.getRegion(i))≤pruningdist then
          APL.insert((MINDIST(q,p.childPage(i)),p.childPage(i)));
        end if
      end for
    end if
  end while
  return result;
```