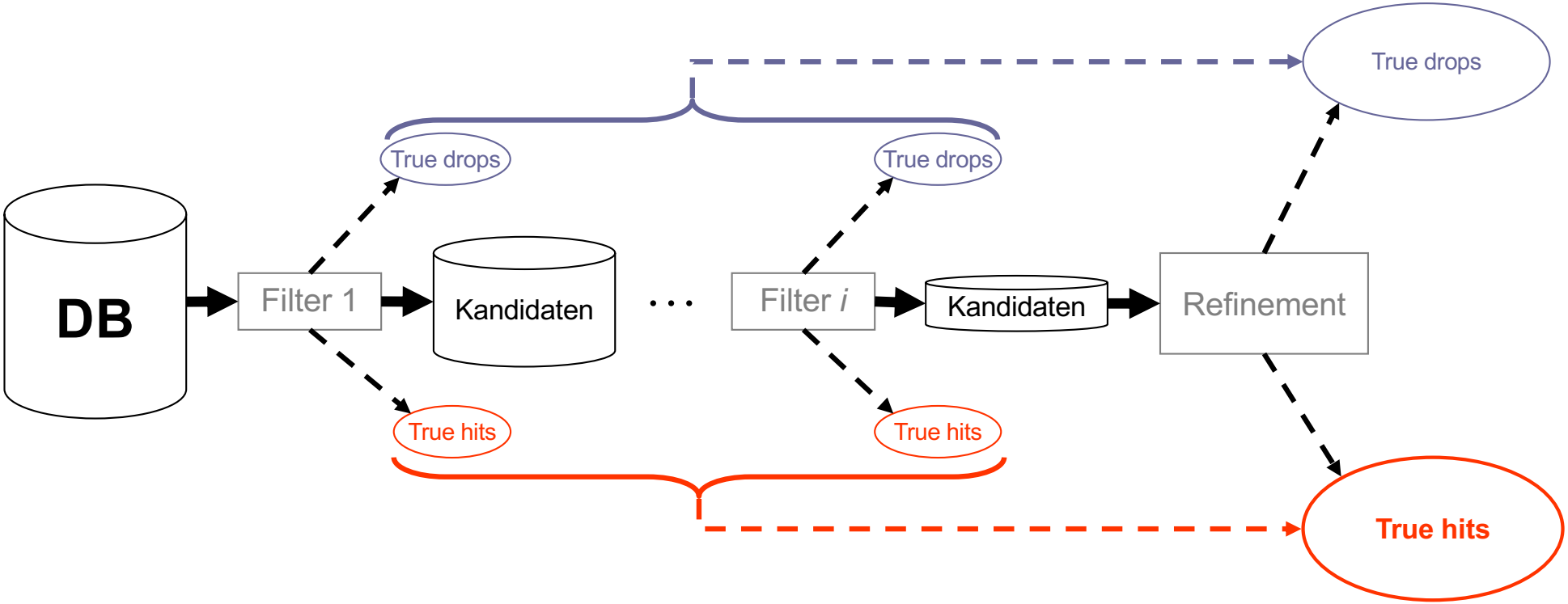
2.2.3 Mehrstufige Anfragebearbeitung

2.2.3 Mehrstufige Anfragebearbeitung

- Idee:
  - Verwendetes Distanzmaß ist sehr teuer (z.B. Edit-Distanz) oder nicht feature-basiert (z.B. Überlappungsfläche von Polygonen)
  - Vektorraum sehr hochdimensional (Curse of Dimensionality)
  - Benutze ein feature-basiertes (meist niedrig-dimensionaler Vektorraum) Distanzmaß als Filterschritt (Filterdistanz)
    - Filterdistanz sollte billiger sein als exakte Distanz (entsprechend niedrig-dimensional => Dimensionsreduktion?)
    - Werte Anfrage-Prädikat (Distanzberechnung) mit Filterdistanz aus
    - Ergebnisse sind noch keine exakten Treffer sondern Kandidaten
    - Kandidatenmenge sollte möglichst klein sein (Filterselektivität)
    - Filterselektivität $\sigma_F = \frac{\#\text{Kandidaten}}{n}$
    - Verfeinerung: für die Kandidaten wird das eigentliche Distanzmaß berechnet, was i.A. teurer dafür selektiver als der Filterschritt ist.

2.2.3 Mehrstufige Anfragebearbeitung

- Mehrstufige Anfragebearbeitung:
  - Ein oder mehrere (kaskadierende) Filterschritte schränken die Kandidatenmenge sukzessive ein
  - Verfeinerungschritt testet auf Korrektheit der Kandidaten



2.2.3 Mehrstufige Anfragebearbeitung

- Zusammenpassen von Filter und Refinement
  - Idealfall:
    Filterdistanz ist obere oder unter Schranke (upper/ lower bound) der exakten Distanz
    => es kann garantiert werden, dass keine exakten Treffer verloren gehen (no false dismissals/drops)
  - Sonst: Ergebnisse u.U. nicht vollständig!!!
  - Lower Bounding Filterdistanz $dist_{LB}$

$$\forall x, y \in DB: dist_{LB}(x, y) \leq dist(x, y)$$

    - Liefert konservative Approximation (d.h. Obermenge) der Ergebnismenge
    - Objekte können evtl. bereits aufgrund der Filterdistanz als exakte Fehltreffer (true drops) identifiziert werden

2.2.3 Mehrstufige Anfragebearbeitung

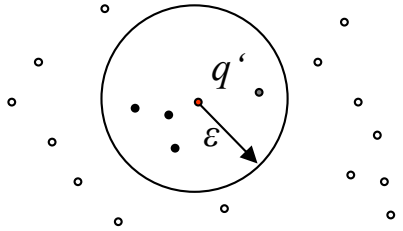
- Upper Bounding Filterdistanz $dist_{UB}$

$$\forall x, y \in DB: dist_{UB}(x, y) \geq dist(x, y)$$

  - Liefert progressive Approximation (d.h. Untermenge) der Ergebnismenge
  - Objekte können evtl. bereits aufgrund der Filterdistanz als exakte Treffer (true hits) identifiziert werden

2.3 Bereichsanfragen

2.3 Bereichsanfragen



- Allgemeines
  - Eigenschaften
    - Benutzer gibt Anfrageobjekt *q* und maximale Distanz ε vor
    - Ergebnis enthält alle Objekte, die höchstens eine Distanz von ε zu *q* haben
  - Formal

$$RQ(q,\varepsilon) = \{o \in DB \mid dist(q,o) \leq \varepsilon\}$$

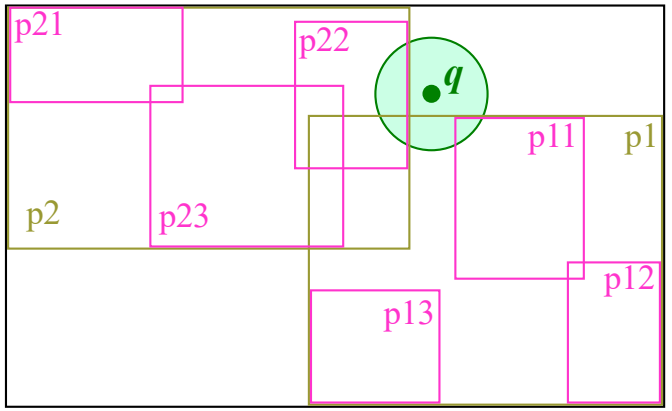
- Basisalgorithmus (sequential scan)

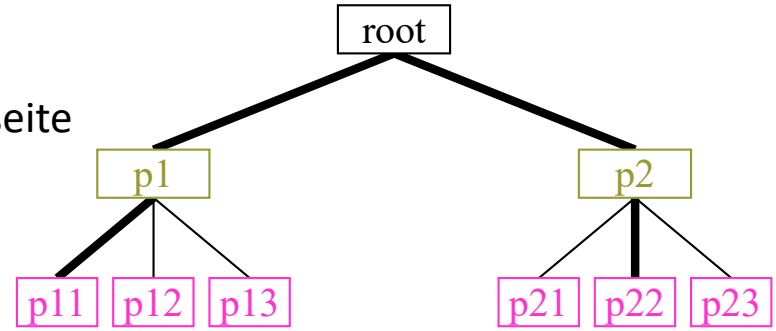
RQ-SeqScan(DB, *q*, ε)
  result = ∅;
  FOR *i*=1 **TO** n **DO**
    IF dist(*q*, DB.getObject(*i*)) ≤ ε **THEN**
      result := result ∪ getObject(*i*);
RETURN result;

2.3 Bereichsanfragen

- Algorithmus mit Index: Tiefensuche

```
RQ-Index(pa, q, ε)     // pa = Diskadress z.B. der Wurzel des Indexes
    result = ∅;
  p := pa.loadPage();
  IF p.isDataPage() THEN
    FOR i=0 TO p.size() DO
      IF dist(q, p.getObject(i)) ≤ ε THEN
        result := result ∪ getObject(i);
  ELSE                                  // p ist Directoryseite
  FOR i=0 TO p.size() DO
    IF MINDIST(q, p.getRegion(i)) ≤ ε THEN
      result := result ∪ RQ-Index(p.childPage(i), q, ε)
  RETURN result;
```

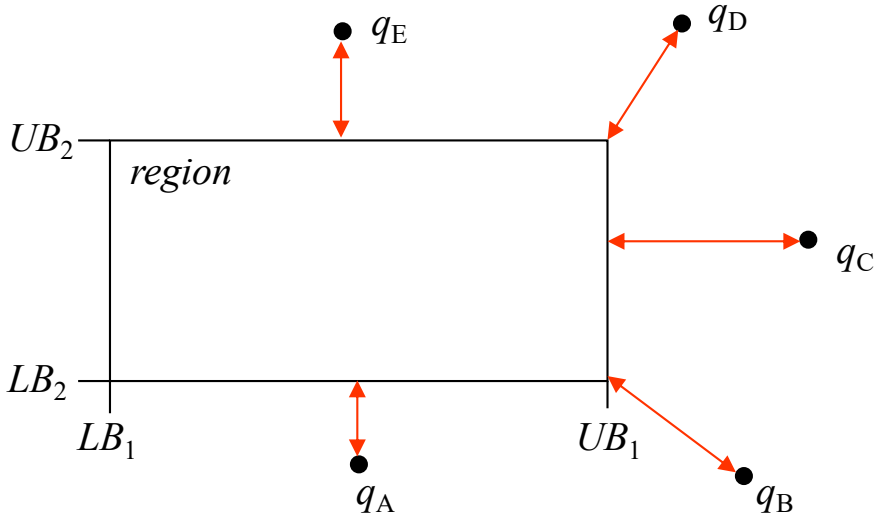




2.3 Bereichsanfragen

- MINDIST
  - Test ob Queryregion sich mit Seitenregion schneidet
  - Minimale Distanz zwischen Anfragepunkt und allen Punkten der Seitenregion (=> Lower Bound!!!)
  - Beispiel: Berechnung der MINDIST für L_2-Norm

$$\text{MINDIST}(region, q) = \sqrt{\sum_{0<i\leq d} \begin{cases} (region.LB_i - q_i)^2 & \text{if} \quad q_i \leq region.LB_i \\ 0 & \text{if} \quad region.LB_i \leq q_i \leq region.UB_i \\ (q_i - region.UB_i)^2 & \text{if} \quad region.UB_i \leq q_i \end{cases}}$$



2.3 Bereichsanfragen

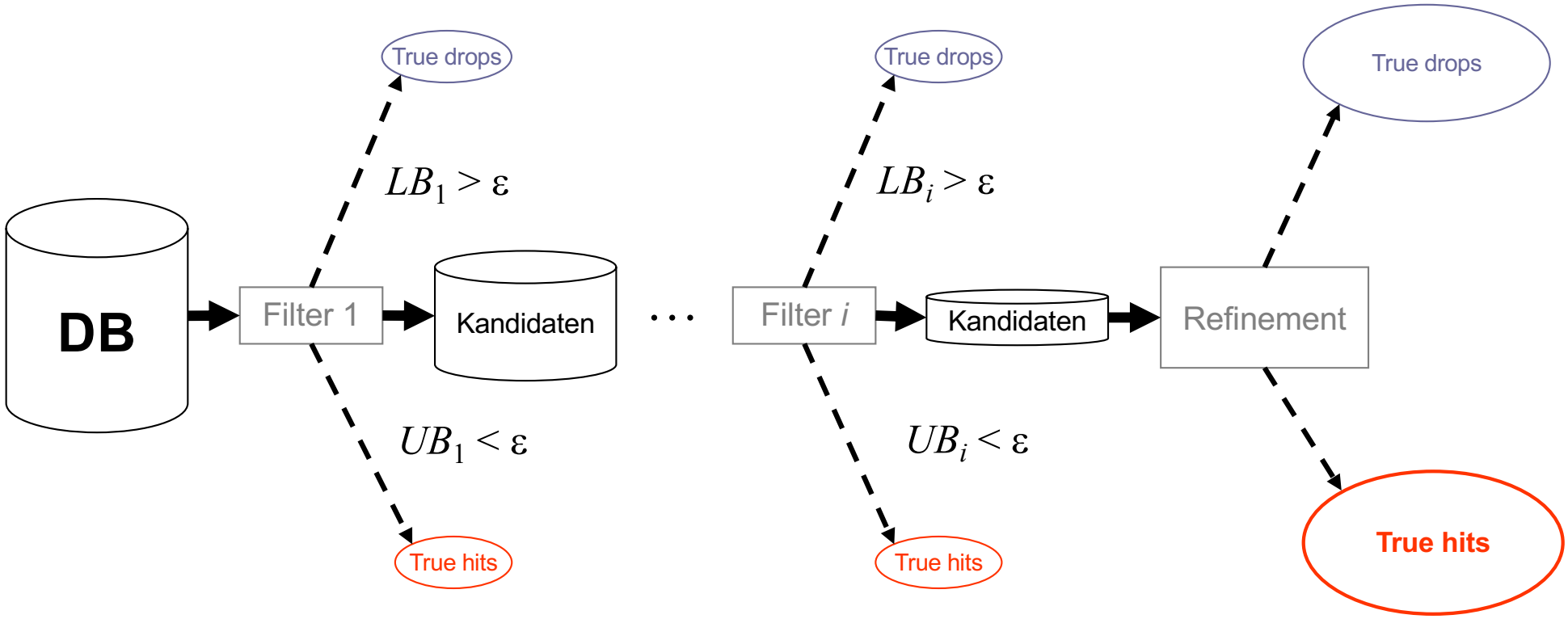
- Mehrstufiger Algorithmus: Filter-/Refinement
  - Lower Bounding Filterdistanz LB
  - Upper Bounding Filterdistanz UB

```
RQ-MultiStep(DB, q, ε)
        result = ∅;
    candidates = ∅;

    // Filter
    FOR i=1 TO n DO
        IF UB(q, DB.getObject(i)) ≤ ε THEN
            result := result ∪ getObject(i);
            ELSE IF LB(q, DB.getObject(i)) ≤ ε THEN
            candidates := candidates ∪ getObject(i);

    // Refinement
    FOR i=1 TO candiates.size() DO
        IF dist(q, candidates.getObject(i)) ≤ ε THEN
            result := result ∪ candidates.getObject(i);
        RETURN result;
```

2.3 Bereichsanfragen



- Oft nur Lower Bounding Distanzen

=> Anzahl der Kandidaten größer, da keine true hits