Peer-to-Peer Similarity Search in Metric Spaces

Christos Doulkeridis, Akrivi Vlachou,
Yannis Kotidis, Michalis Vazirgiannis

http://www.db-net.aueb.gr/cdoulk/
cdoulk@aueb.gr

Department of Informatics
Athens University of Economics and Business (AUEB)
Athens, Greece

Motivation

- Similarity search in metric spaces
- Objects are represented in a high dimensional feature space
- Complex distance functions (e.g. text, multimedia)

- *Goal:* share the computational load over a set of computers
  → Peer-to-Peer

- DBISP2P'07 session on P2P similarity search
- Existing work
  - Centralized settings
  - Structured P2P systems (not preserving peer autonomy)

Outline

1. Preliminaries
   a. Metric spaces
   b. iDistance
2. SIMPEER
   a. Construction
   b. Range query processing
   c. *k*-NN query processing
3. Experimental results
4. Conclusions & further work

Metric Space

- Metric space *M=(D,d)*
  - *d(p,q) = d(q,p)* *(symmetry)*
  - *d(p,q) > 0, q≠p* and *d(p,p)=0* *(non negativity)*
  - *d(p,q) ≤ d(p,o) + d(o,q)* *(triangle inequality)*

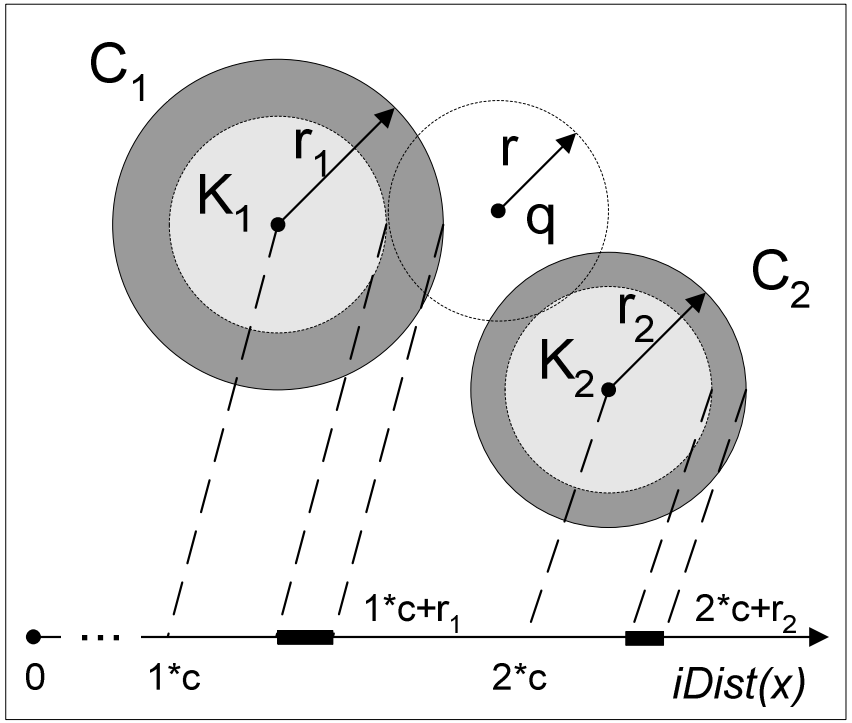
- Similarity queries
  - Range queries: $R(q,r) = \{ u \in D \mid d(q,u) < r \}$
  - *k*-NN queries: $NN_k(q)$

iDistance – Indexing the Distance

- Space partitioning into n clusters
- Reference points K_i
- Each cluster mapped to an interval
- Each object x mapped to 1-d

$$iDist(x) = i * c + dist(K_i, x)$$

- Values indexed in a B+-Tree



- Query R(q,r)
  - If a query intersects with a cluster

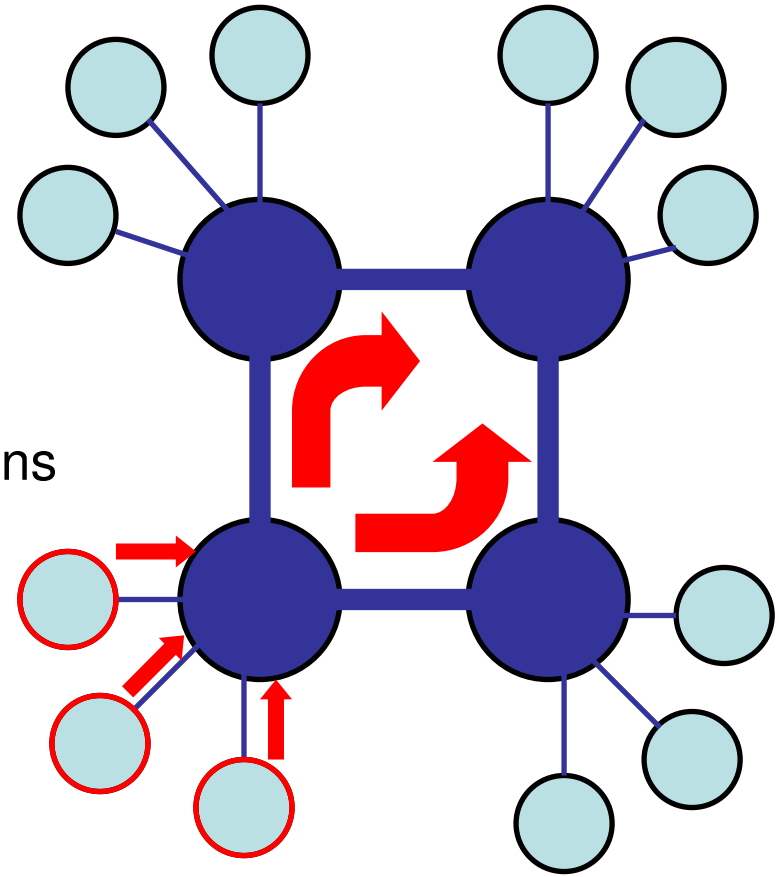
$$dist(K_i, q) - r \leq r_i$$

  - Scan the interval

$$[i * c + dist(K_i, q) - r, i * c + dist(K_i, q) + r]$$

SIMPEER
3-level Clustering Scheme

1. Each peer
   - clusters its own data
   - indexes local points using iDistance
2. Each super-peer
   - receives its peers' cluster descriptions
   - computes the ***hyper-clusters*** using our ***extension of iDistance***
3. Super-peers
   - exchange hyper-clusters
   - build a set of ***routing clusters***



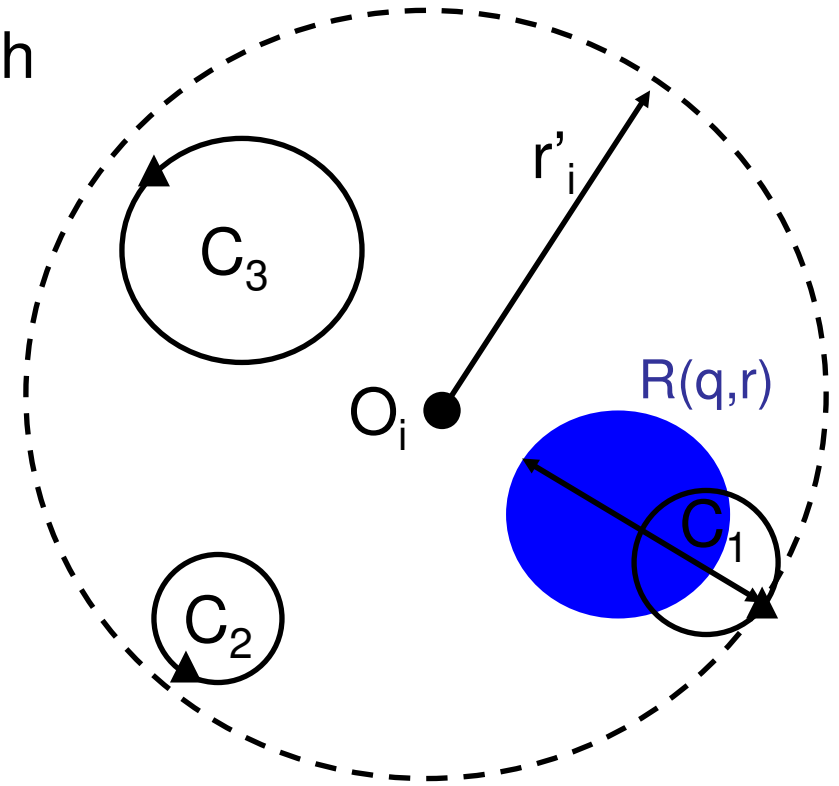
Super-peer architecture

iDistance Extension

- Map clusters, not points
- Index the furthest point of each cluster *only!*
- Each cluster C_j mapped to

$$key_j = i * c + [dist(O_i, K_j) + r_j]$$

- Values indexed in a B+-Tree

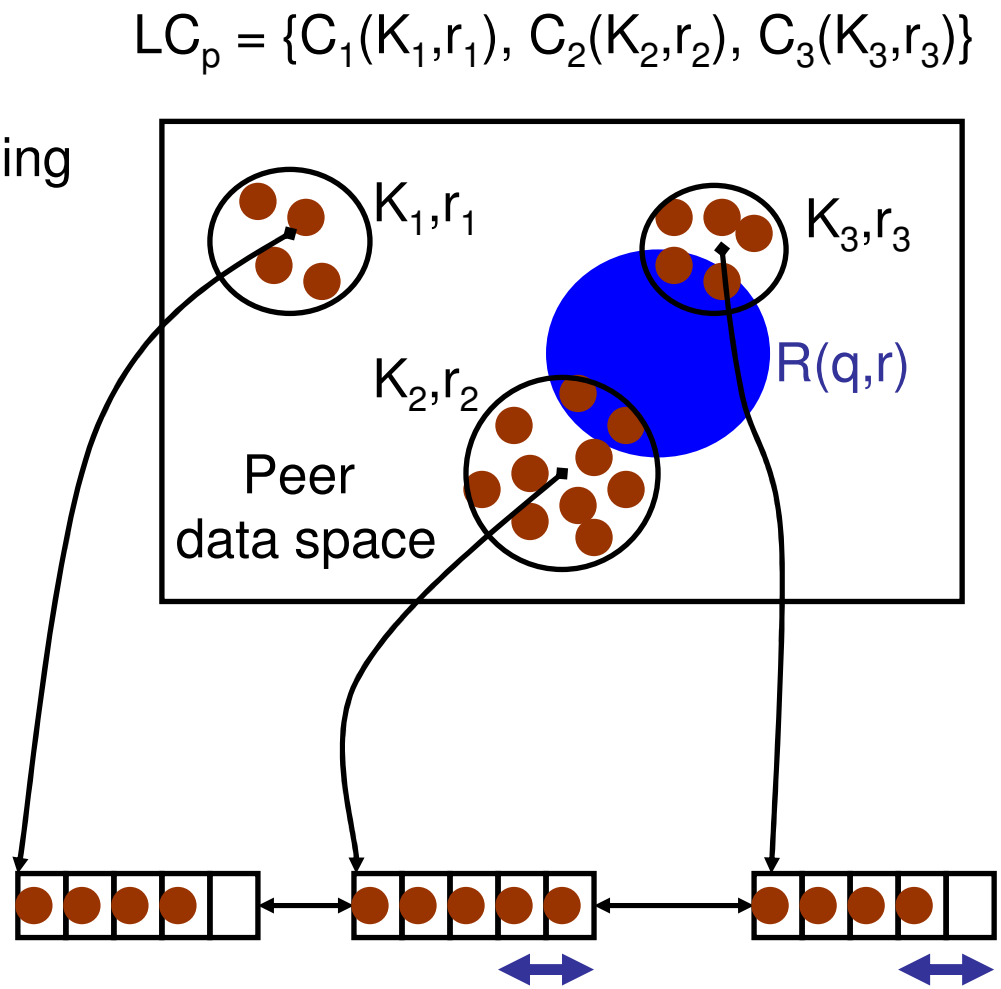


Hyper-cluster

- Query R(q,r)
  - Search region [d(O_i,q) - r, r'$_i$]

Peer Query Processing

- Data organization
  - Clustering / Space partitioning
  - iDistance
- LC_p sent to super-peer
  - $LC_p = \{ C_i : (K_i, r_i) \}$
- Range query processing
  - Scan intervals of B$^+$-tree

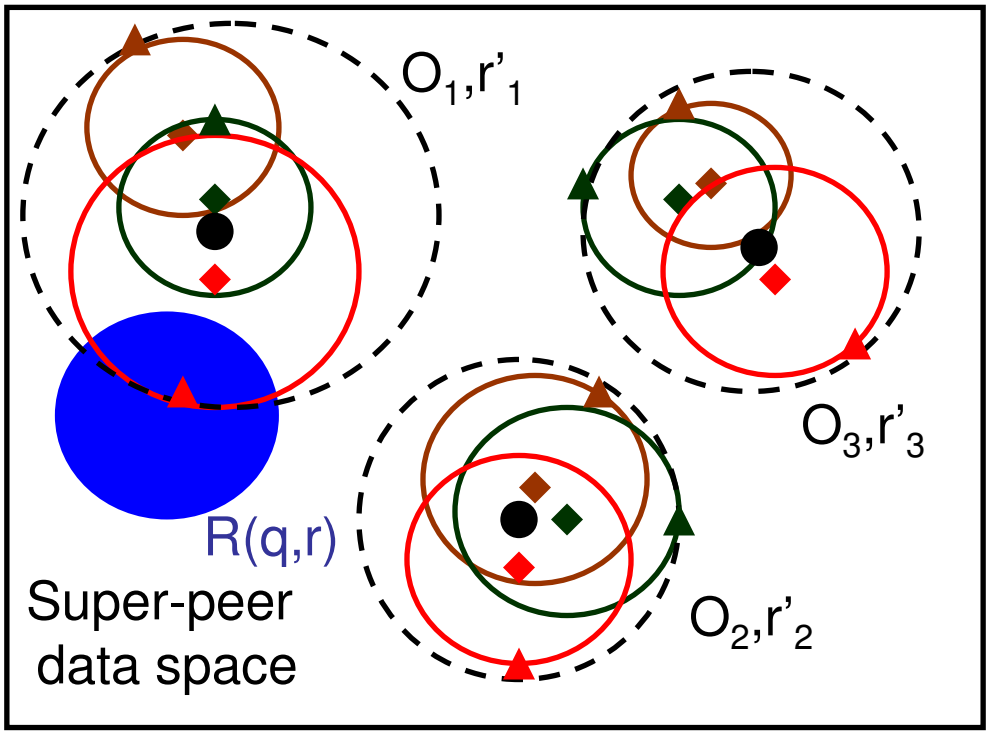


Super-Peer Query Processing

- Super-peer
  - Creates *hyper-clusters* based on peer clusters
  - Indexes the *furthest* point of each peer's cluster

$LHC_{sp} = \{HC_1(O_1,r'_1), HC_2(O_2,r'_2), HC_3(O_3,r'_3)\}$

- Range query processing
  - Find peers to forward the query
  - *Peer selection mechanism*



Routing Indices

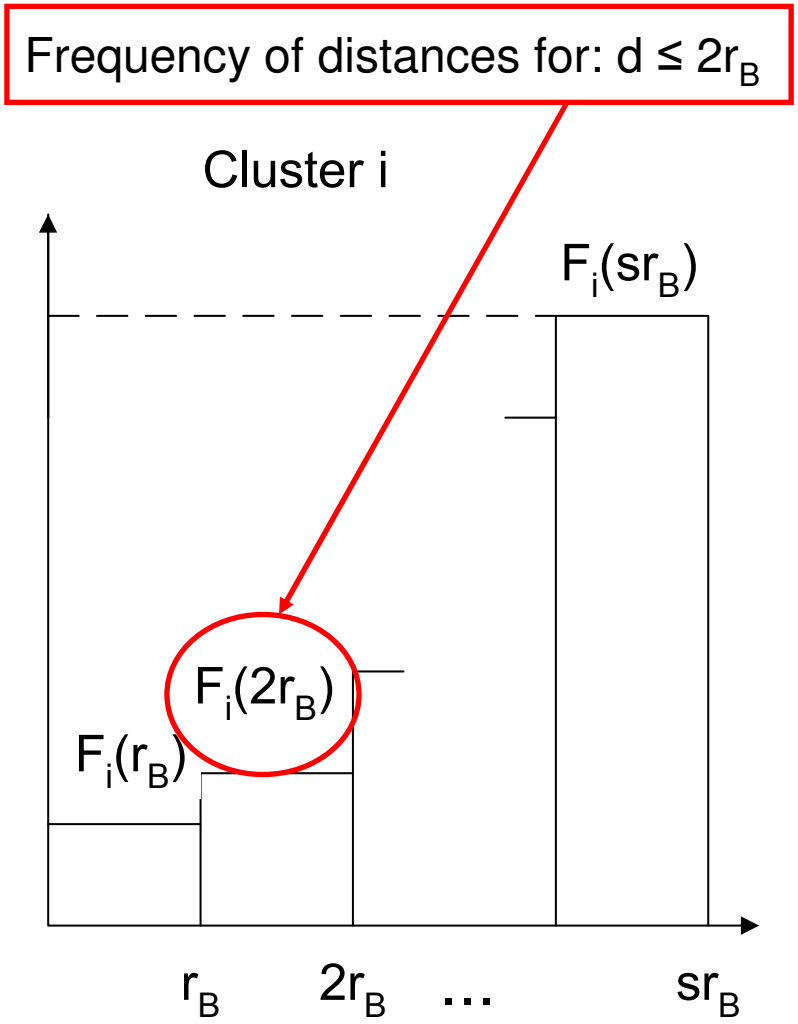
- Super-peers broadcast hyper-clusters
- Recipient super-peers
  - Treat hyper-clusters similarly to peer clusters
- Build **routing clusters** RC_i
  - Used to determine the neighbouring super-peer to forward the query
  - *Super-peer selection mechanism*

k-NN Query Processing

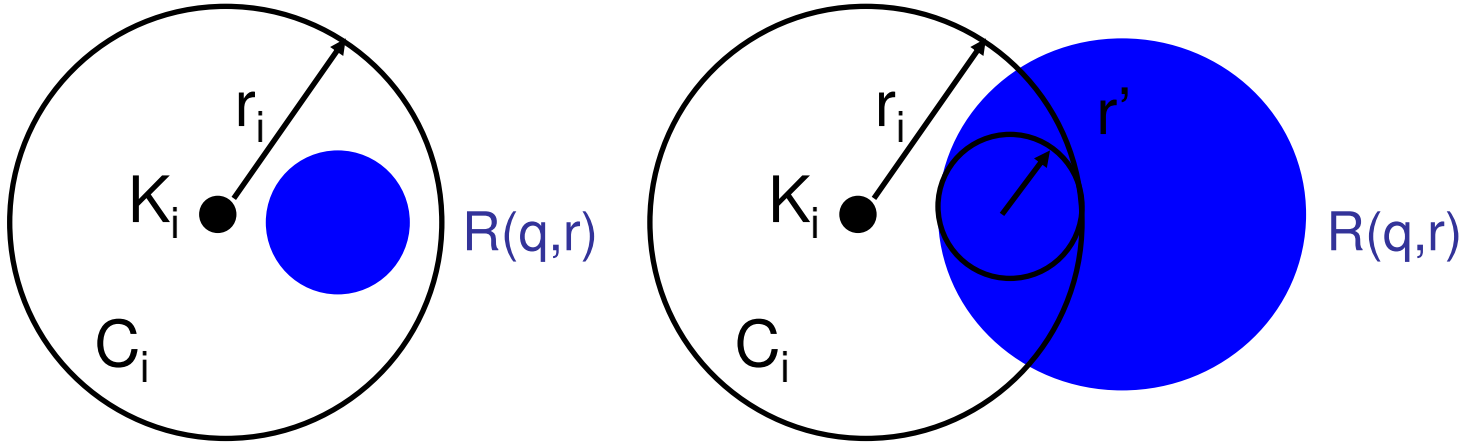
- Convert k-NN query to range query R(q,r)
  - Use *estimated* range r
  - Based on (peer) cluster information at a super-peer *local estimation*
  - Based on hyper-cluster information at a super-peer *global estimation*
  - No communication required for estimation!
- Maximum 2 round-trips required!
  - If less than k objects retrieved, cannot avoid second round-trip
  - Super-peer computes an upper bound for r, based on its peers data
- *Goal:* make a good estimation, such that
  - First round-trip is enough (overestimate r)
  - r is sufficient, but not too large (do not overestimate r too much)

Histogram Construction

- Distribution of distances
  - $F(r) = Pr\{d(q,p) \leq r\}$
- Expected number of retrieved objects by R(q,r)
  - $\#objs(R(q,r)) = n \times F(r)$
- Assumption
  - "high" homogeneity of viewpoints inside a cluster [Ciaccia, PODS'98]
  - Approximate F_q with a sampled distance distribution F



Local Estimation (LE)



Condition : $d(K_i, q) + r \leq r_i$ $d(K_i, q) + r > r_i$

Estimated #objects : $n_i \times F_i(r)$ $n_i \times F_i(r')$

where $r' = r_i + r - d(K_i,q)/2$

Binary search on $[0, sr_B]$ to find the smallest r for which the estimated number of objects ≥ k

Global Estimation (GE)

- *Hyper-clusters* enhanced with 2 histograms: (hc_i) (hd_i)
  - Number of clusters intersecting the query (nc_i)
    - Distance distribution of clusters within a hyper-cluster
  - Number of data objects contained in the intersection (nd_i)
    - Superimpose cluster histograms, by keeping the minimum value of each bin
    - Also keep the minimum cardinality of all clusters

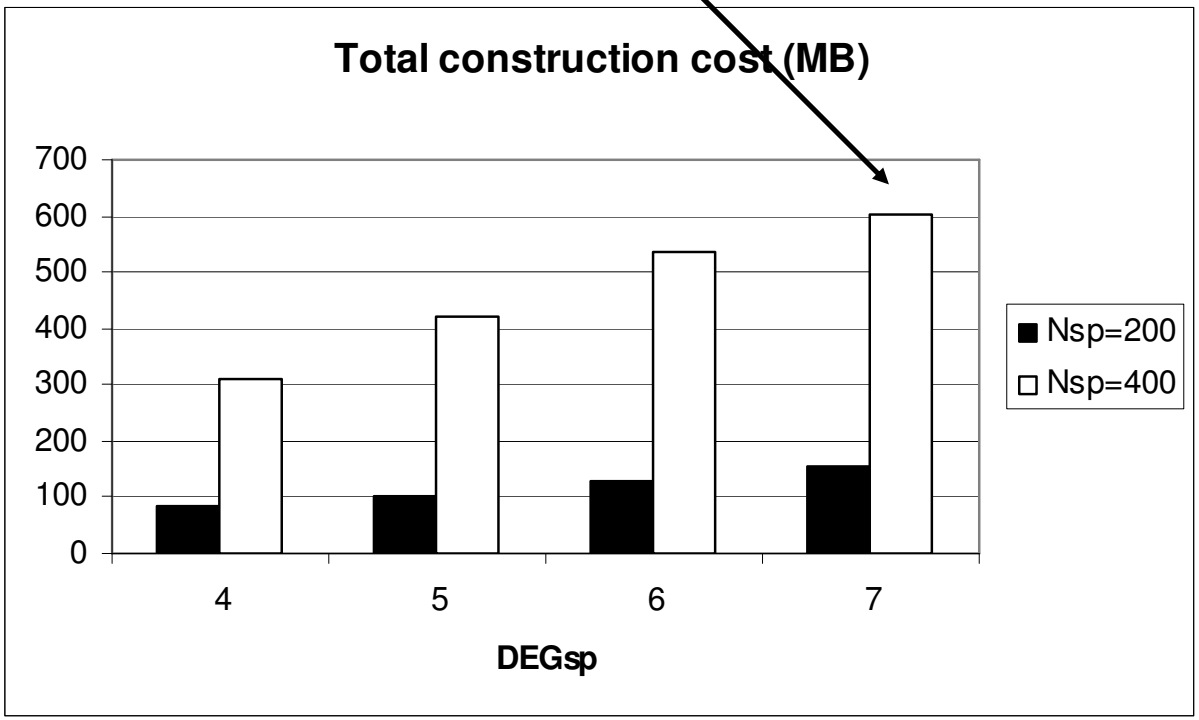
Estimated #objects : $nc_i(r) \times nd_i(r)$

Experimental Setup

- GT-ITM topology generator (4K-16K peers)
- #Super-peers={200,400}
- DEG_{sp}=4-7
- DEG_{p}=20-60
- k_p=10
- Sunthetic {uniform,clustered} datasets
  - 8-32d, 3M-12M objects
- Real datasets
  - VEC 1M 45-dim vectors of color image features
  - CovType 581K 54-dim instances of forest Covertype data

Construction Cost

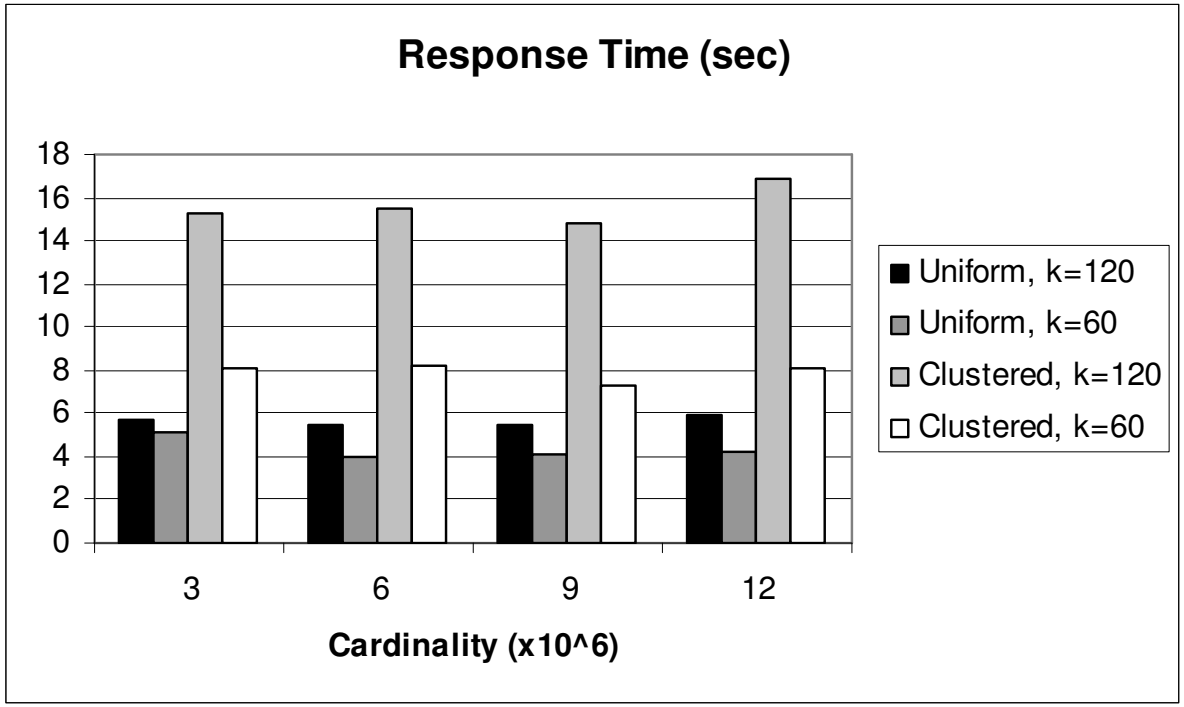
- Mainly depends on super-peer topology
- One-time cost!
- Approx. 1.5MB per super-peer



Range Queries – Response Time

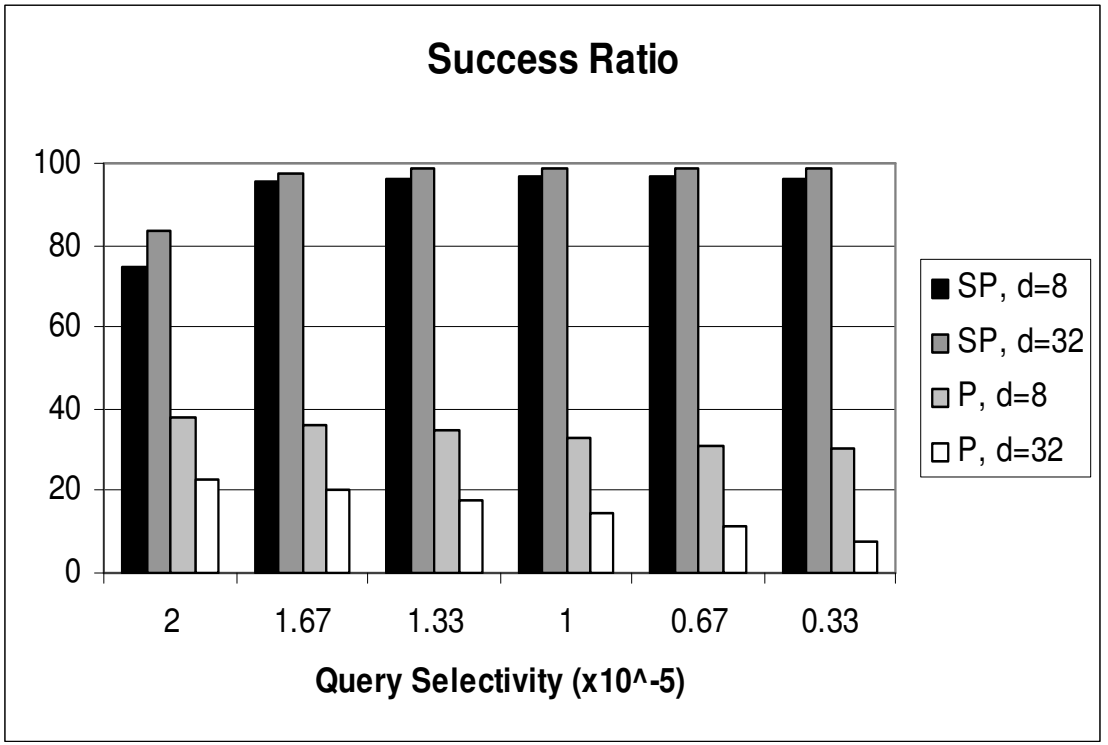
- (N_{sp}=200, N_p=2000, n=1M, d=16)
- Increases only slightly with cardinality
- Higher response time in clustered dataset
- Most results come from the same network paths, causing delays

Network transfer rate 4KB/sec



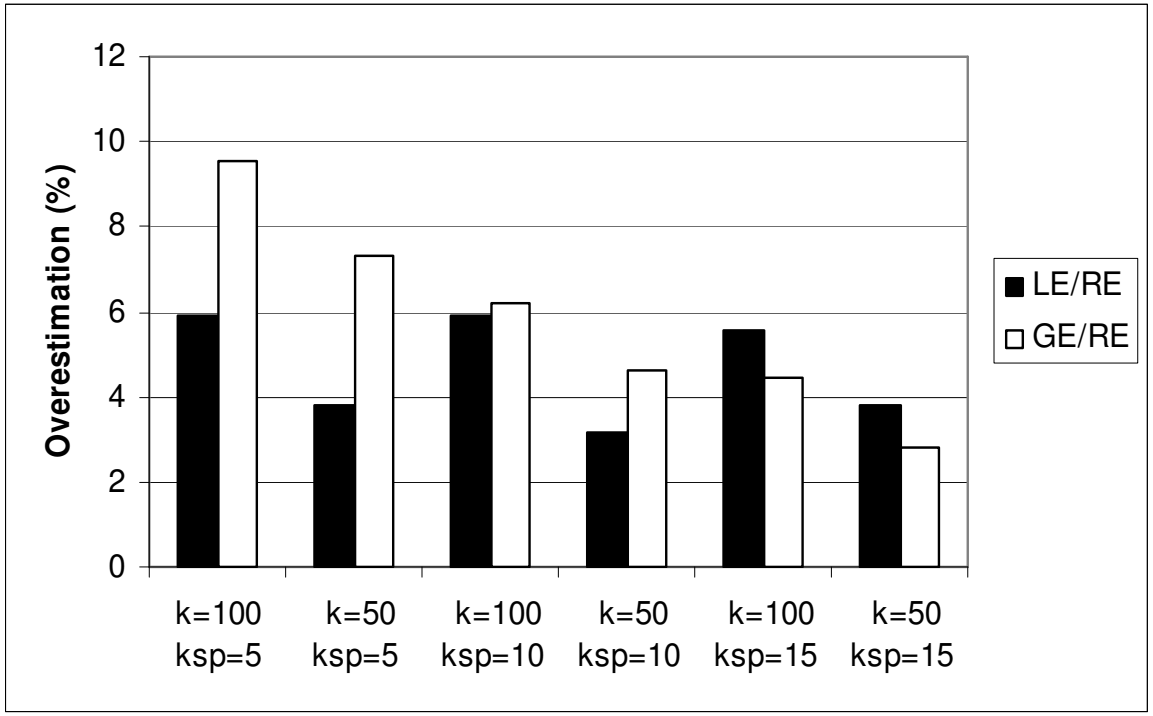
Range Queries – Success Ratio

- Clustered dataset (N_{sp}=200, N_p=2000, n=1M)
- Success ratio = how many of the contacted peers (super-peers) returned results



k-NN Queries – Overestimation(%)

- VEC dataset (N_{sp}=200, N_p=2000)
  - LE better (initially)
  - GE becomes better with increasing k_{sp}



Conclusions & Further Work

- SIMPEER
  - A metric-based framework for P2P similarity search
  - Utilizes a three-level clustering scheme
- Support for range and *k*-NN query processing
- Distributed statistics

- Further work
  - Extension for non-vector-based data representations
  - Devise an approach that deals with uniform data distributions in a better way

Thank you for your attention !

More info:

http://www.db-net.aueb.gr/cdoulk/

cdoulk@aueb.gr