VK Multimedia Information Systems

Mathias Lux, mlux@itec.uni-klu.ac.at

Dienstags, 16.oo Uhr c.t., E.1.42

Indexing

- Spatial Indexes
- MDS – FastMap
- Locality Sensitive Hashing
- Metric Indexes
- Inverted Lists

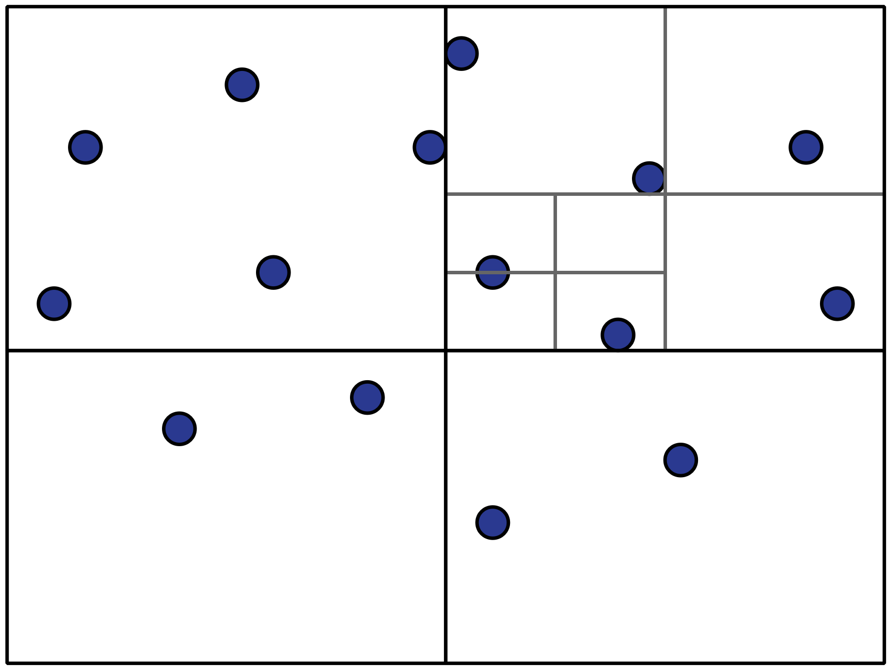
Indexing Visual Information

- Text is indexed in inverted lists
  - Search time depends on # of terms
- Visual information expressed by “vectors”
  - Combined with a metric capturing the semantics of similarity
  - Inverted list does not work here
  - An “index of vectors” is needed

Indexing Visual Information

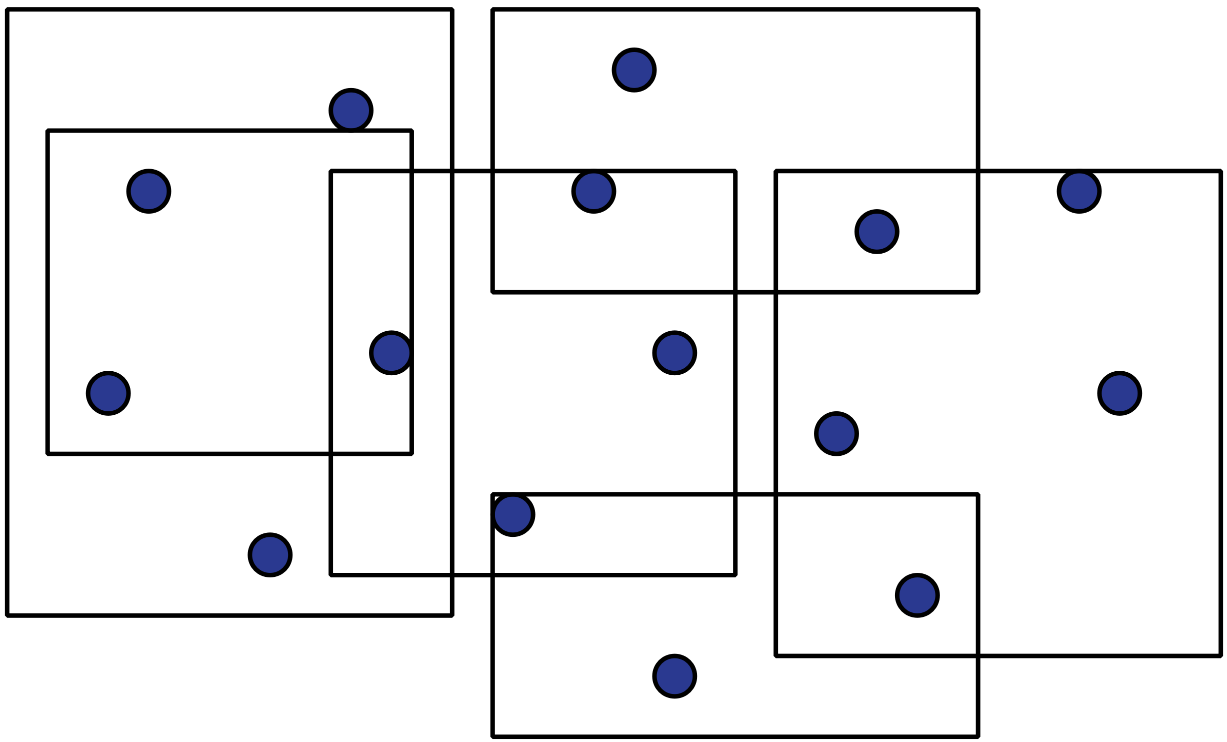
- Vectors describe “points in a space”
  - Space is n-dimensional
  - n might be rather big
- Distance (metric) between points
  - E.g. L1 or L2 …
- Query is also a vector := point
  - Searching for points (vectors) near to query
- Idea for index:
  - Index neighborhood …

Spatial Indexes



Using equally sized rectangles (Optimal for L1 …)

Spatial Indexes

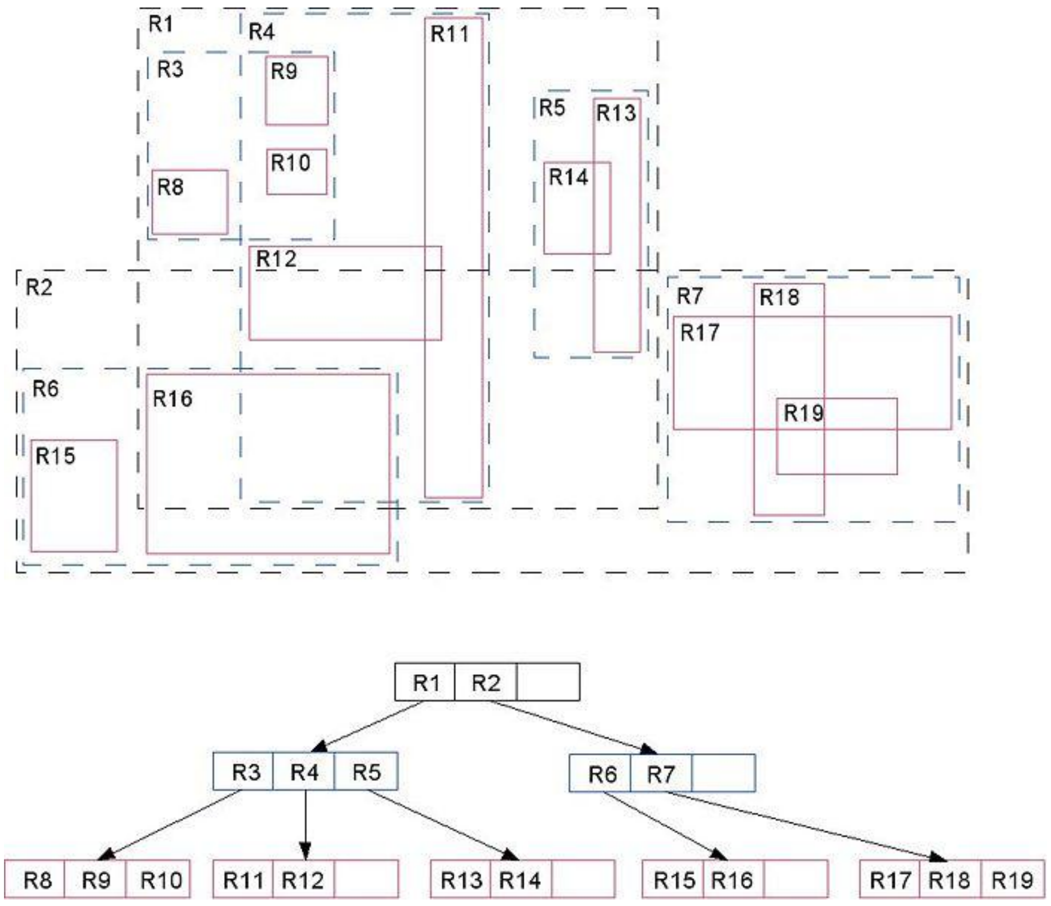


Using overlapping rectangles …

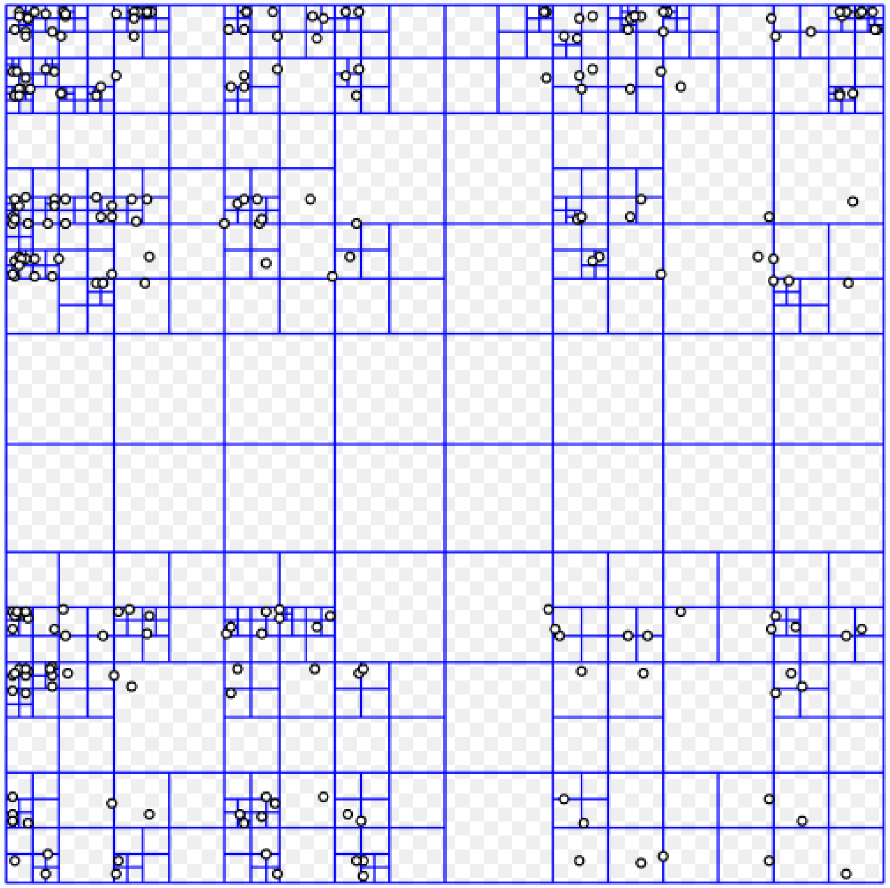
Spatial Indexes

- Common data structures
  - R Tree
    - R*, R+, ….
    - Overlapping rectangles
    - Search is a rectangle
  - Quadtree (Octtree)
    - Equally sized regions, subdivided
    - 4 quadrants or 8 octants
    - Search selects quadrants

R-Tree


R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
R17
R18
R19
R1 R2
R3 R4 R5
R6 R7
R8 R9 R10
R11 R12
R13 R14
R15 R16
R17 R18 R19

Quadtree



Spatial Indexes: Drawbacks

- Data structures must minimize
  - false negatives (-> maximizes recall)
  - false positives (-> search time)
- Features, distance function & parameters need to be selected at index time
  - Search combining multiple descriptors is complicated issue
- Works best for spaces with small dimension *n*
  - MDS has to be applied …

Indexing

- Spatial Indexes
- MDS – FastMap
- Locality Sensitive Hashing
- Metric Indexes
- Inverted Lists

Multidimensional Scaling (MDS)

- Reducing the dimensions of a feature space
  - E.g. From 64 dimensions to 8
  - Without loosing too much information about neighborhoods
- Applications in multimedia retrieval
  - Indexing based on coordinates
  - Spatial Indexes:
    - Data structures to find nearest neighbors fast

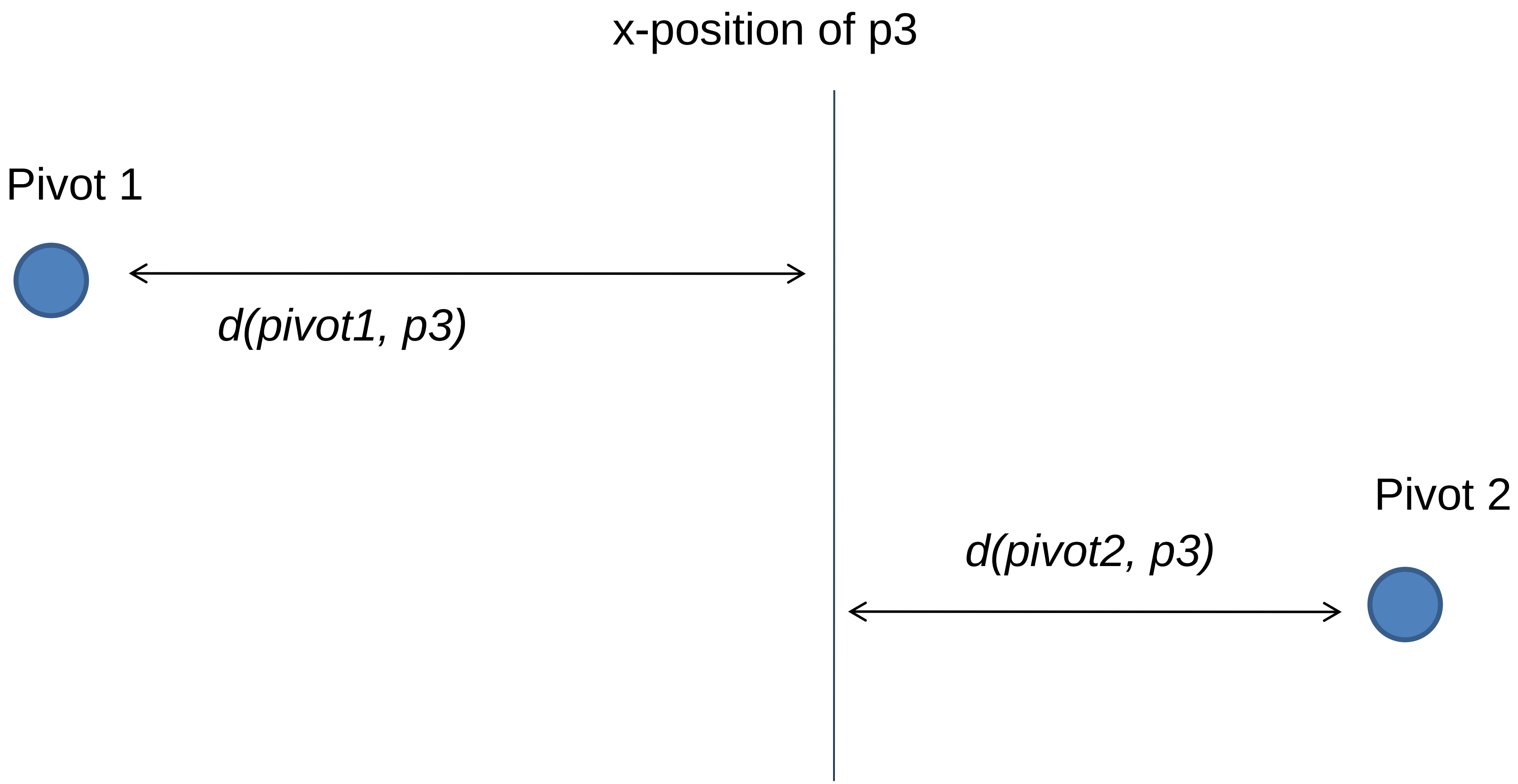
Multidimensional Scaling (MDS)

- Interpolation: FastMap
  - Linear in terms of objects
  - Used e.g. in IBM QBIC
- Iterative: Force Directed Placement
  - Iterative optimization of initial placement
  - Cubic runtime

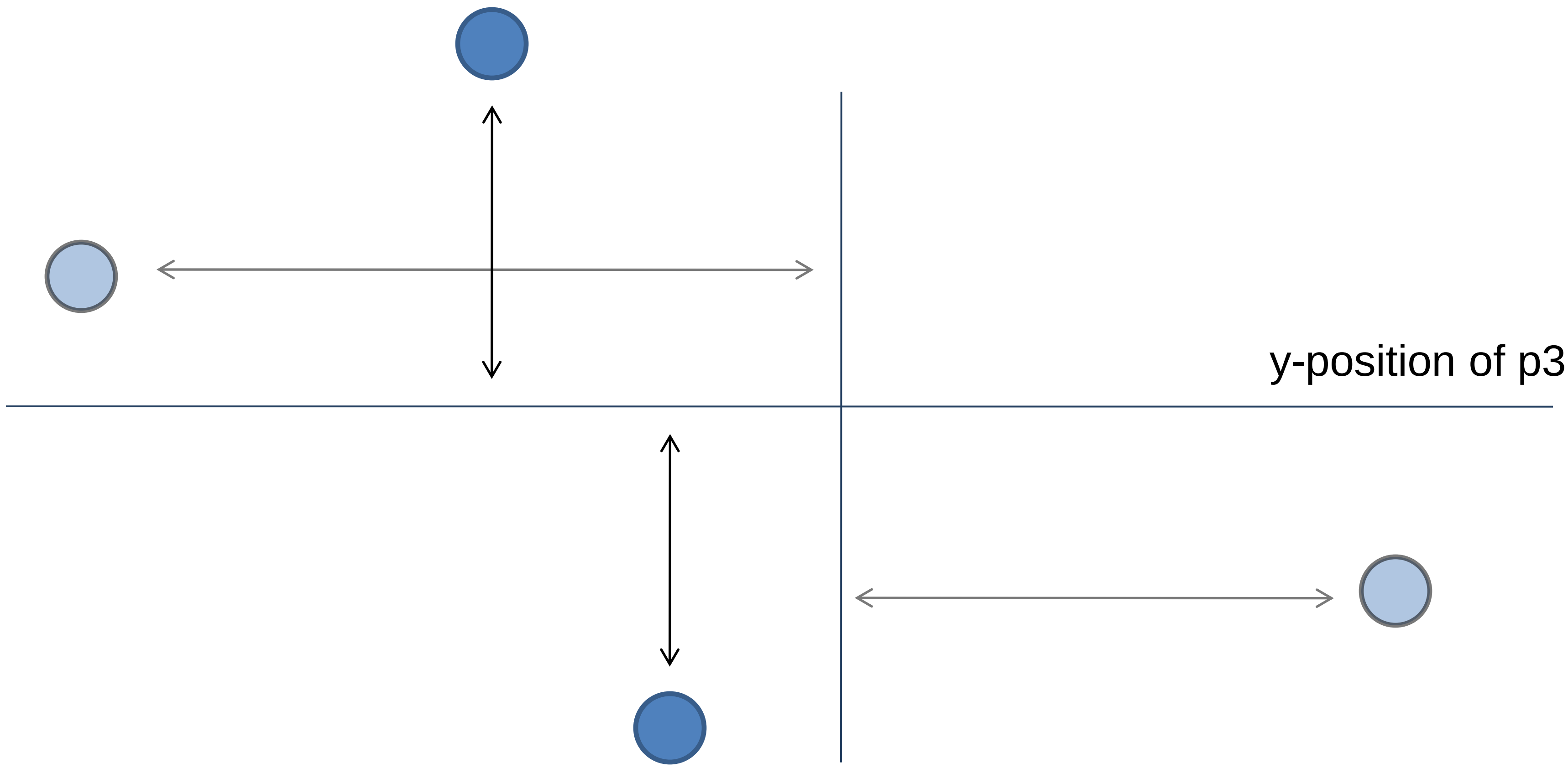
FastMap

- For Each dimension d
  - Find Pivots (the most distant objects)
  - For each object, which is not a pivot
    - Interpolate position between pivots in this dimension
  - Next object
- Next Pivot

FastMap



FastMap



FastMap: Pivots

How to find optimal pivots?

- Select one object randomly -> P_1
- Select Object P_2 with maximum distance from P_1 to P_2
- If $d(P_1, P_2) < t$
  - Set $P_1 = P_2$
  - Goto (2)

Normally no threshold is used but this is done x times.

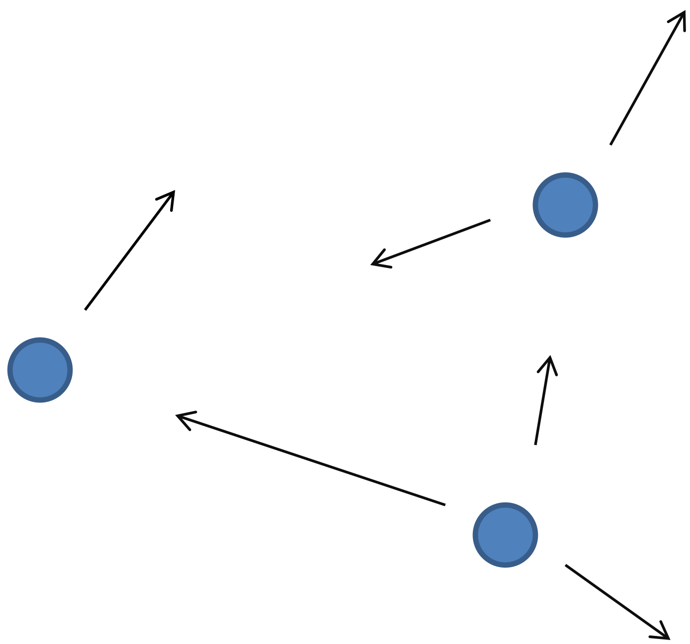
Force Directed Placement

1. All objects are assigned coordinates
2. For each object *o*
   – Movement vector $v = 0$
   – For each object *p*
     - Calculate repulsion & attraction forces between *o* & *p*
     - Compute movement vector $v(o, p)$ depending on the forces
     - $v = v + v(o, p)$
3. If overall movement is still high goto 2.

FDP: Parameters

- Gravity as overall attraction
  - Prevents uncontrolled spread
- Overall repulsion
  - Prevents coming objects from coming too close
- Minimum distance
  - If objects are on the coordinates
- Spring parameters
  - Repulsion stronger close up
  - Attraction stronger if far away

FDP



Demo

- Emir

Indexing

- Spatial Indexes
- MDS – FastMap
- Locality Sensitive Hashing
- Metric Indexes
- Inverted Lists

Locality Sensitive Hashing (LSH)

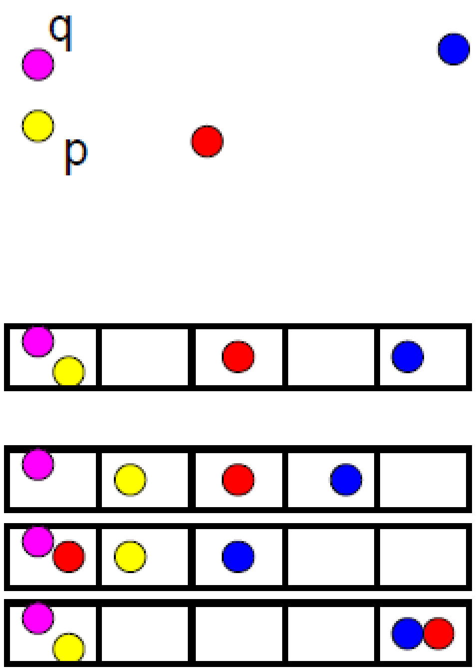
- Algorithm to determine the "Approximate Near(est) Neighbor"

- Given: a set P of points in R^d
- Nearest Neighbor: query q returns point $p \in P$ minimizing $|p-q|$
- r-Near Neighbor: query q returns point $p \in$ P so that $|p-q| \leq$ r

src. http://people.csail.mit.edu/indyk/mmds.pdf

LSH - Idea

- Construct hash functions $g: R^d \rightarrow U$ so that
  - If $|p\text{-}q| \leq r$ then Pr[g(p)=g(q)] is “not so small”
  - If $|p\text{-}q| > cr$ then Pr[g(p)=g(q)] is “small”



LSH - Process

- A family H of hash functions $h: R^d \rightarrow U$ is called (P1, P2, r, cr)-sensitive if
  - If $|p-q| \leq r$ then Pr[h(p)=h(q)] > P1
  - If $|p-q| > cr$ then Pr[h(p)=h(q)] <P2
- LSH uses functions $g(p)=<h_1(p), \ldots h_k(p)>$
  - Preprocessing
    - Select functions $g_1 \ldots g_L$
    - Hash all $p \in P$ to buckets $g_1(p) \ldots g_L(p)$
  - Query
    - Retrieve points from buckets $g_1(q) \ldots g_L(q)$

LSH

- LSH solves c-approximate NN with:
  - Number of hash functions: $L=n^\rho$, $\rho=\log(1/P1)/\log(1/P2)$
  - E.g., for the Hamming distance we have $\rho=1/c$
  - Constant success probability per query q
- LSH schemes
  - Extending beyond Hamming distance
  - Projection based, etc.

Indexing

- Spatial Indexes
- MDS – FastMap
- Locality Sensitive Hashing
- Metric Indexes
- Inverted Lists

Metric Indexes

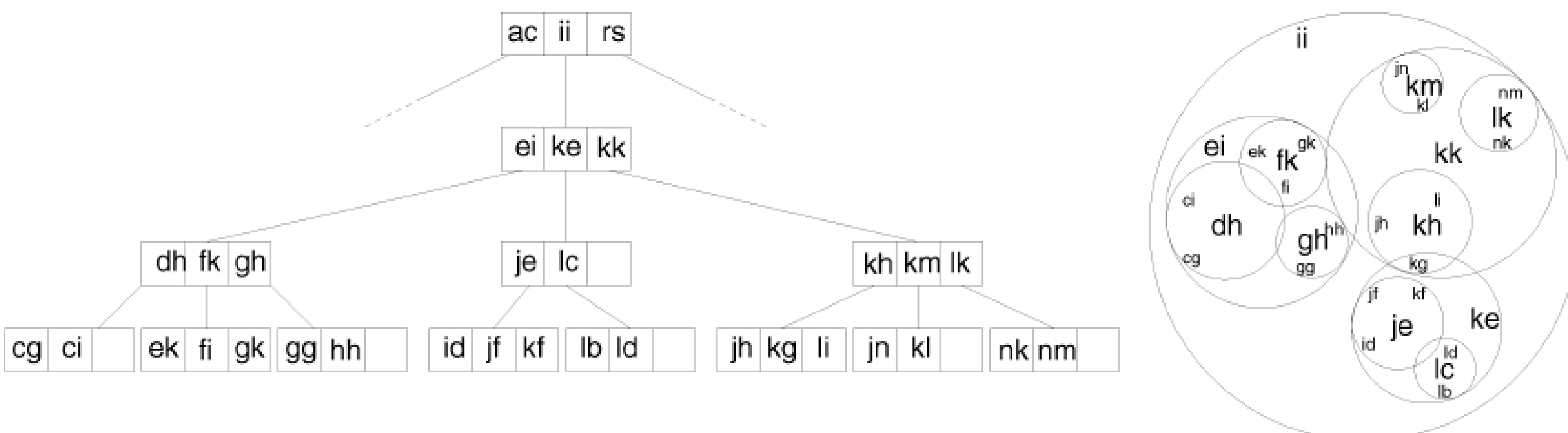
A metric index is a *tree of nodes*

- Each node containing a fixed maximum number of entries
- Each entry is constituted by a routing entry *D*

src. Berretti, S.; del Bimbo, A. & Vicario, E.
Efficient Matching and Indexing of Graph Models in Content-Based Retrieval
IEEE Transactions on Pattern Analysis and Machine Intelligence, **2001***, 23*, 1089-1105

Metric Indexes

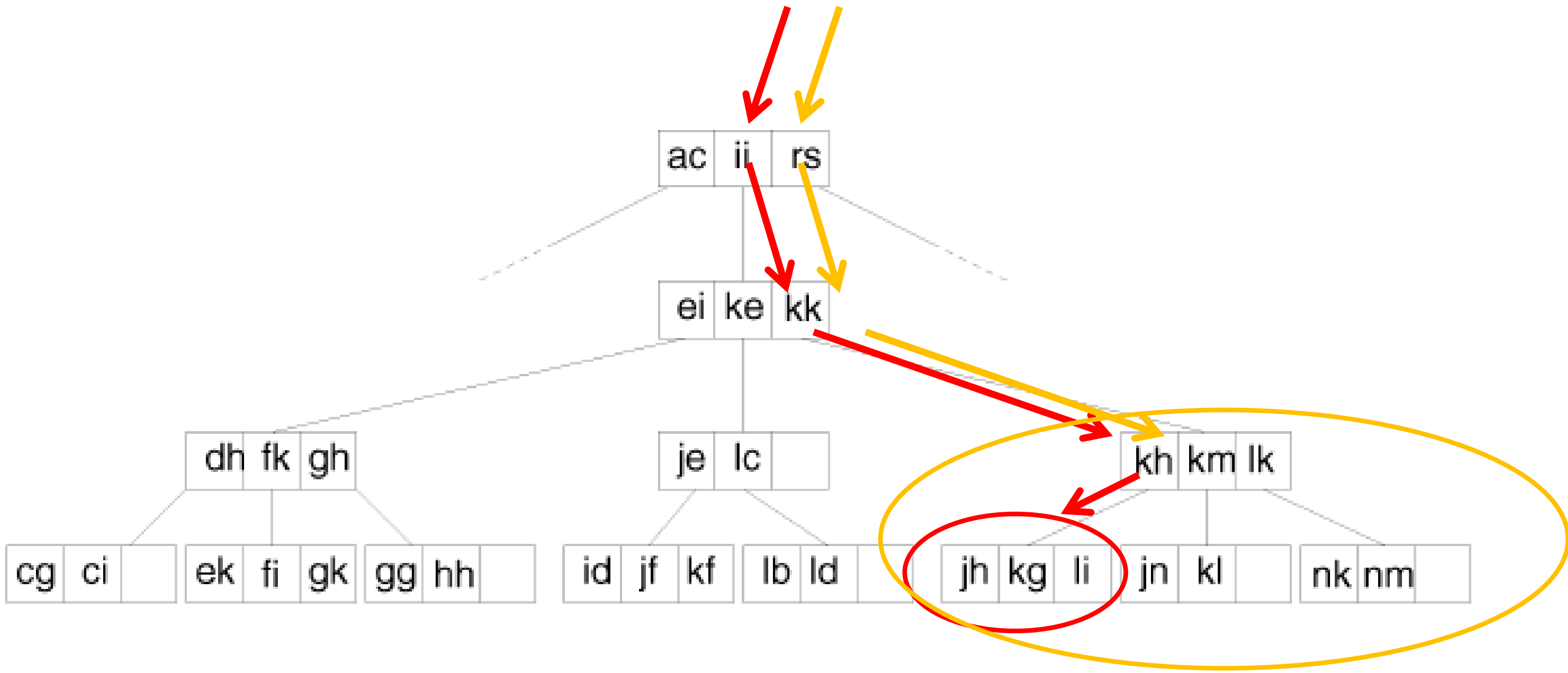
- *D* is the root of a sub index in the *covering region* of *D*
  – Also defines a radius r_D being the maximum distance from *D* to any entry in the covering



Metric Index: Construction

- Top-down: Indexing an entire archive at once
  - All documents to index are known
  - No iterative additions
- Bottom-up: Indexing on insertion
  - Documents are indexed as they are added to the collection
  - Optimizations (e.g. splitting) have to be done

Metric Index: Searching



Indexing

- Spatial Indexes
- MDS – FastMap
- Locality Sensitive Hashing
- Metric Indexes
- Inverted Lists

Metric Spaces

src. G. Amato & P. Savino, „Approximate Similarity Search in Metric Spaces Using Inverted Files ", Infoscale 2008

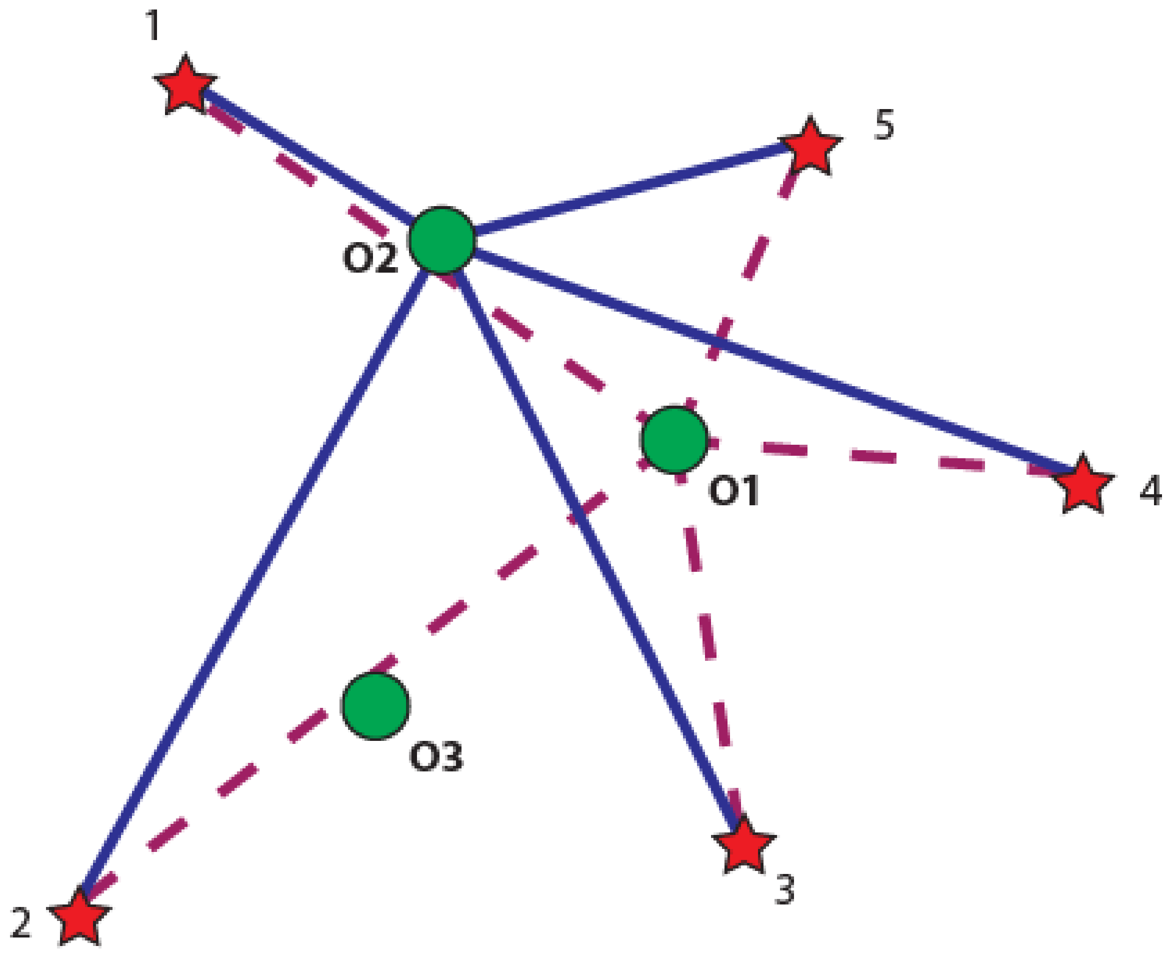
- $\mathcal{M} = (\mathcal{D}, d)$
  - Data domain $\mathcal{D}$
  - *Total (distance) function d*: $\mathcal{D} \times \mathcal{D} \rightarrow$ R (metric function or metric)
- The metric space postulates:
  - Non negativity $\forall x, y \in \mathcal{D}, d(x, y) \geq 0$
  - Symmetry $\forall x, y \in \mathcal{D}, d(x, y) = d(y, x)$
  - Identity $\forall x, y \in \mathcal{D}, x = y \Leftrightarrow d(x, y) = 0$
  - Triangle inequality $\forall x, y, z \in \mathcal{D}, d(x, z) \leq d(x, y) + d(y, z)$

Similarity Search in Metric Spaces

- Objects close to one another see the space in a “similar” way
- Choose a set of *reference objects RO*
- Orderings of *RO* according to the distances from two similar *data objects* are similar as well
  - Represent every data object *o* as an ordering of *RO* from *o*
  - Measure similarity between two data objects by measuring the similarity between the corresponding orderings

Similarity Search in Metric Spaces

O1 := <5, 3, 4, 1, 2>
O2 := <1, 5, 3, 4, 2>
O3 := …



Similarity Search in Metric Spaces

- Spearman Footrule Distance

$$SFD(S_1, S_2) = \sum_{ro \in RO} \left| S_2(ro) - S_1(ro) \right|$$

Similarity Search in Metric Spaces

[Slides G. Amato …]

Thanks …