

Database and Information Systems

11. Deductive Databases
12. Data Warehouses and OLAP
13. Data Mining
- 14. Index Structures for Similarity Queries**
15. Semi-Structured Data
16. Document Retrieval
17. Web Mining
18. Content Extraction
19. Multimedia Data

Indexing Structures for High-Dimensional Data

Readings:

Schmitt, I.: Ähnlichkeitssuche in Multimedia-Datenbanken. Retrieval, Suchalgorithmen und Anfragebehandlung. Oldenbourg, München 2006.

Indexing Structures for High-Dimensional Data

- Queries in High-Dimensional Databases
- Boundary-Based Indexes
- Dimensionality Reduction

Queries in High-Dimensional Databases

- range queries: find all objects whose attribute values fall within certain given ranges
 - rectangular hyper-window (window query)
- similarity range queries: find all objects which are within a given distance from an object
 - hyper-sphere query
 - distance is defined based on an application specific metrics
- nearest neighbor query: find the object which is closest to a given object
- reverse nearest neighbor query: find all objects for which a given object would be a nearest neighbor
 - nn-relation is not symmetric
 - e.g. finding an optimal location for a meeting

Queries in High-Dimensional Databases

- k-nearest neighbor (KNN) queries: find the k-most similar objects which are closest in distance to a given object
 - high dimensional data have low contrast in distance
 - if more similar objects than required exist → random choice
- similarity join: find all pairs of objects which are similar enough
 - distance is smaller than a predefined threshold
- similarity queries can be emulated by range queries using a filter-and-refine approach
 - filter: find the candidates with a sufficiently large bounding box
 - refine: check the similarity criterion for each of them

Queries in High-Dimensional Databases

- requirements for index structures
 - soundness and completeness
 - suitability for high-dimensional problem spaces
 - suitability for spatially extended objects
 - retrieval efficiency
 - efficient update operations (insertion, deletion, update)
 - support for several distance metrics
 - minimal space requirements
- no one-fits-all solution → compromise needs to be found

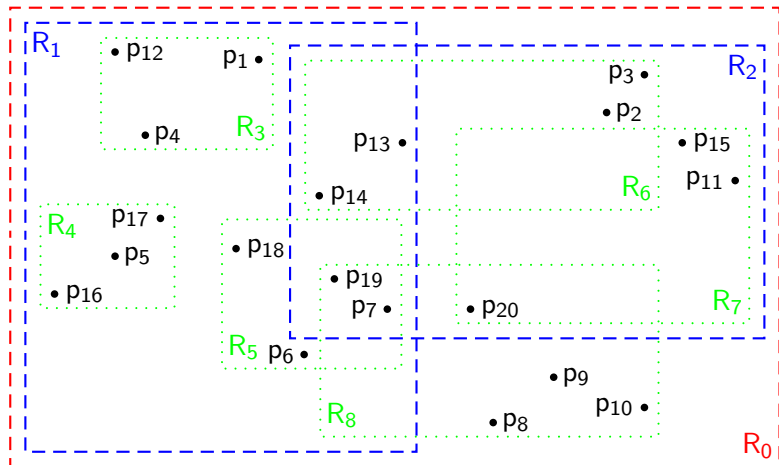
Queries in High-Dimensional Databases

- curse of high dimensionality
 - a high-dimensional data space is sparse
 - distance between data points increases
 - data contrast is low
 - distance between the nearest and the farthest data point is reduced
 - high number of almost equally similar data points
 - notion of similarity vanishes
 - approximate (probabilistic) methods suffice

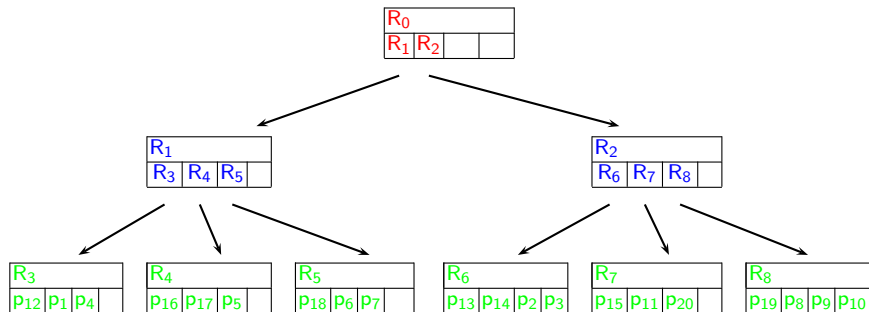
Boundary-Based Indexes

- range queries: R-Trees
 - multi-dimensional extension of B^+ -trees
 - preserves height balance
 - insertion: node splitting
 - deletion: node merging
- leaf nodes
 - object identifier
 - bounding box: Minimum Bounding Rectangle (MBR)
- non-leaf nodes
 - child-pointer
 - bounding box for the whole sub-tree

Boundary-Based Indexes



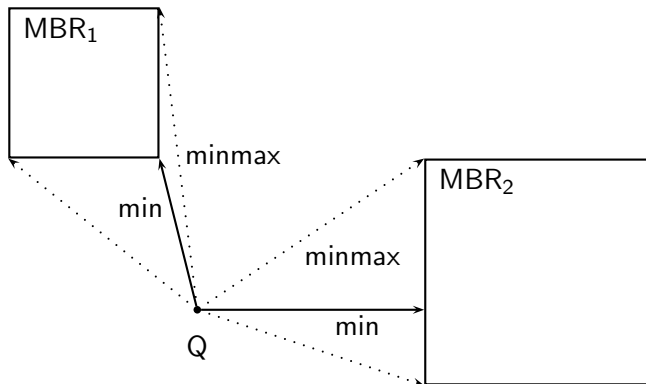
Boundary-Based Indexes



Boundary-Based Indexes

- range queries: recursive tree traversal
 - find all leaf nodes where the bounding box overlaps the query range
 - non-deterministic search: all non-leaf nodes with intersecting bounding boxes have to be considered
- nn-queries: objects in a node can be ordered
 - min-distance:
minimal distance between a query point and a point in an MBR
optimistic expectation for the distance to the nearest neighbor
 - minmax-distance:
maximal distance a query point can have to a nearest neighbor in an MBR
pessimistic expectation for the distance to the nearest neighbor

Boundary-Based Indexes



Boundary-Based Indexes

- search maintains an upper limit for the best result found so far
- $\min(Q, MBR_1) > \min(Q, MBR_2)$
 - MBR_1 need not be considered
- upper limit $> \min(Q, MBR_1)$
 - MBR_1 contains a closer neighbor
 - upper limit can be updated to $\min(Q, MBR_1)$
- upper limit $< \min(Q, MBR_1)$
 - MBR_1 need not be considered
- can be extended to deal with knn-queries
 - maintaining the candidates in a priority queue of length k
 - upper limit refers to the last entry in the priority queue

Boundary-Based Indexes

- node insertion
 - least coverage criterion: choose the branch which requires minimal enlargement to also accommodate the new object
 - in case of a tie: choose the smallest box
 - all traversed and splitted nodes are readjusted to a minimum bounding box
- node splitting: like in a B-tree
- node deletion:
 - changes the bounding box in ancestor nodes
→ adaptation needed
 - in case of underflow: delete the node and reinsert its remaining children from the root
 - might cause further node deletions

Boundary-Based Indexes

- problems:
 - the overlap of bounding boxes increases as the dimensionality grows
 - not well suited for multi-dimensional problems
 - for a large number of dimensions ($d > 10$) a sequential scan can be shown to be more efficient
 - overlap is sensitive to the order of insertion → reorganisation can give an advantage (e.g. node re-insertion from the root)
 - not well suited for KNN queries
 - only for vector spaces with a `EUCLIDEAN` distance

Boundary-Based Indexes

- variants
 - R^+ -Tree: no overlaps allowed \rightarrow too many splits
 - R^* -Tree: optimizes the margin of the bounding boxes
squarish boxes are preferred
 - X-tree: avoiding splits if they result in highly overlapping nodes
 \rightarrow supernodes
 - A-Trees: using virtual bounding boxes to approximate the minimal ones

Boundary-Based Indexes

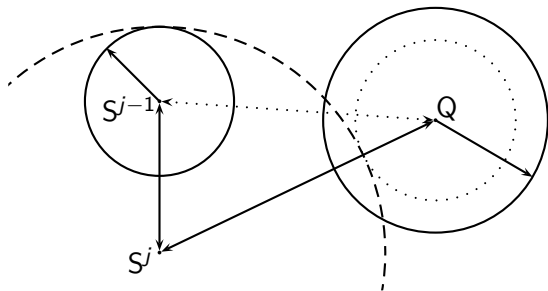
- alternative: using bounding spheres instead of bounding boxes (SS-Tree)
 - centroid: mean vector
 - radius: distance from the centroid to the farthest data point
- generalized version: metric tree (M-Tree)
 - data are clustered first
 - clusters are mapped to nodes in the tree
 - centroids are used as routing objects
 - triangle inequality is used to exclude subtrees from being searched
 - tree is balanced

Boundary-Based Indexes

- advantages
 - works with arbitrary metrics
 - lower dimensionality: $d + 1$ instead of $2d$
 - radius of the bounding sphere is determined by the distance
→ insensitive to the dimensionality while diagonal in a box increases with dimensionality
 - but: boxes can be better adapted to different value ranges along different dimensions
 - well suited for similarity search

Boundary-Based Indexes

- distance between a centroid and the query can be used to prune the search space



- there must be a subcluster in the closer hemisphere
- the true distance to the subcluster cannot be larger than the distance to the centroid it belongs to

Boundary-Based Indexes

- kd-Tree: binary search tree
- splitting on different levels of the tree can be done along different dimensions of the feature space
- partitions the feature space completely (no overlaps)
- tree is unbalanced (by definition)
- hyper-rectangles are usually larger than necessary but queried more often
 - worse performance

Boundary-Based Indexes

- no tree-based index is efficient for truly high-dimensional problems
- common assumption:
 - data points can be clustered
 - certain clusters can be excluded from search
- assumption is fundamentally wrong for similarity search in a space with many uncorrelated features

Dimensionality Reduction

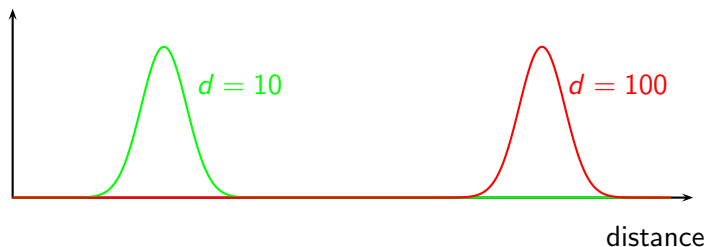
- dimensionality reduction techniques are always lossy
- indexing based on important attributes: TV-Tree (Telescopic-Vector Tree)
 - similar to an R-Tree, but nodes higher up in the tree use fewer features
 - features have to be selected according to some predefined ranking
- Principal Component Analysis
 - decorrelation of features
 - transforms the whole feature space
 - preserves similarity properties
 - using only a subset of the transformed feature space for indexing
 - efficient if data dimensions are globally correlated
 - but degree of correlation might change for dynamic data sets

Dimensionality Reduction

- alternative: Local Dimensionality Reduction (LDR)
 - detect local clusters in the data set
 - perform a LDR for the individual clusters
 - build a local, low dimensional index using the transformed feature space
 - build a global index for the clusters
 - data points which do not belong to any cluster are treated as outliers and cannot be indexed at all
 - user can determine the amount of information loss, which affects the query precision and the query costs
- general problems:
 - degree of correlation might change for dynamic data sets
 - approach is also based on the idea of clustering

The Curse of Dimensionality (Revisited)

- for high dimensional problems in a **EUCLIDEAN** space the
 - expected value for the distance between two data points grows
 - but the standard deviation remains constant



- many data points have a similar distance
- any kind of clustering becomes impossible

The Curse of Dimensionality (Revisited)

- approximation error:
 - average distance between
 - a point query and the nearest data point of a cluster and
 - the point query and the cluster itself
- approximation error grows linearly with the expected value of the distance and
- eventually exceeds the greatest data point distance!
- → there is no method that reliably justifies to exclude a cluster from being considered in a high-dimensional data space

Signature-based Access

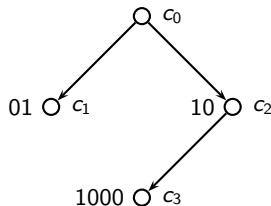
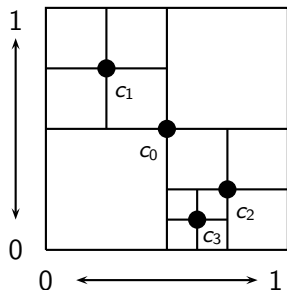
- vector approximation techniques (VA file)
- assumption: space can be partitioned into a finite set of discrete cells
 - partition the data space
 - number of cells per dimension is given
 - boundaries are chosen in a way that the number of data points in each interval is evenly distributed
 - store the cell boundaries
 - represent each cell by a signature
 - i.e. concatenation of a binary representation for the cell number in each dimension
 - store the signature together with the full vector for each data point
 - use the signature as a filter to eliminate the data points that are not in the answer set

Signature-based Access

- search:
 - determine the relevant cells by a sequential scan
 - fetch the data points within the cells from the data base
 - check the search conditions for the fetched items
- but number of bits needs careful tuning
- drawback: fixed number of bits to describe a data point

Signature-based Access

- modification: active vertex file (AV file)
 - partition the data space into a hierarchy of cells with different granularity



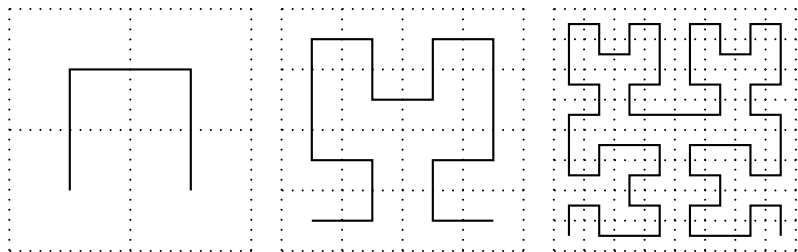
- a data point is assigned to a cell if it is closer than a given radius r from its centroid
- choice of r determines the efficiency

Space-filling Curves

- special case of vector approximation
 - locality preserving total ordering of points in a k -dimensional space
 - e.g. Hilbert-curve
 - special enumeration scheme for the cells, e.g. for a 2-dimensional space:

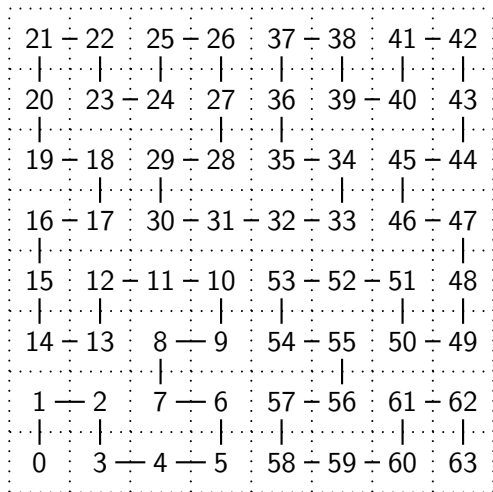
Space-filling Curves

- segment the space into 4 (2^k) cells
- order the cells according to a basic curve
- recursively call the algorithm on all the cells until a detailed enough segmentation has been reached



Space-filling Curves

- enumerate the cells along the path



Space-filling Curves

- not all regions in the original space can be represented as contiguous regions in the new one
- → decomposition into several regions necessary

