More Branch and Bound Algorithms

Algorithmic Problems Around the Web #3

Yury Lifshits
http://yury.name

CalTech, Fall'07, CS101.2, http://yury.name/algoweb.html

Outline

Outline

1. Variations of Metric Trees
2. M-Trees

Outline

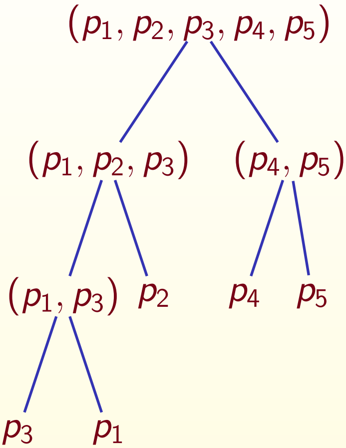
Part I

Variations of Metric Trees

Branch and Bound: Range Search

Task: find all $i \quad d(p_i, q) \leq r$:

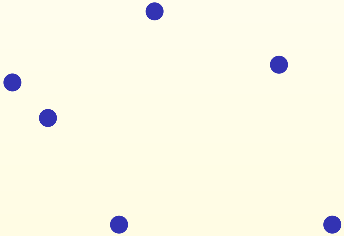
1. Make a depth-first traversal of search hierarchy
2. At every node compute the lower bound for its subtree
3. Prune branches with lower bounds above r



Vantage-Point Partitioning

Uhlmann'91, Yianilos'93:

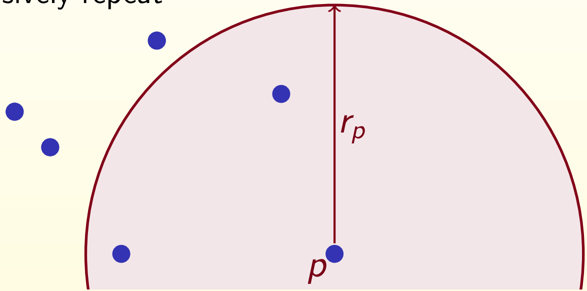
1. Choose some object p in database (called pivot)
2. Choose partitioning radius r_p
3. Put all p_i such that $d(p_i, p) \leq r$ into "inner" part, others to the "outer" part
4. Recursively repeat



Vantage-Point Partitioning

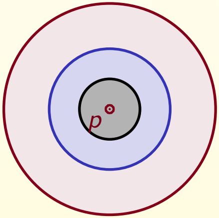
Uhlmann'91, Yianilos'93:

1. Choose some object p in database (called pivot)
2. Choose partitioning radius r_p
3. Put all p_i such that $d(p_i, p) \leq r$ into "inner" part, others to the "outer" part
4. Recursively repeat



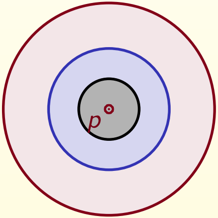
Variations of Vantage-Point Trees

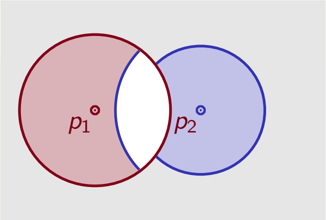
- **Burkhard-Keller tree:** pivot used to divide the space into m rings Burkhard&Keller'73
- **MVP-tree:** use the same pivot for different nodes in one level Bozkaya&Ozsoyoglu'97
- **Post-office tree:** use $r_p + \delta$ for inner branch, $r_p - \delta$ for outer branch McNutt'72



Variations of Vantage-Point Trees

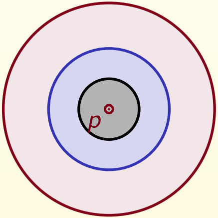
- **Burkhard-Keller tree:** pivot used to divide the space into m rings   Burkhard&Keller'73
- **MVP-tree:** use the same pivot for different nodes in one level   Bozkaya&Ozsoyoglu'97
- **Post-office tree:** use $r_p + \delta$ for inner branch, $r_p - \delta$ for outer branch   McNutt'72

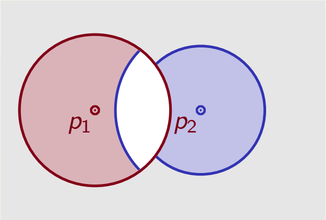


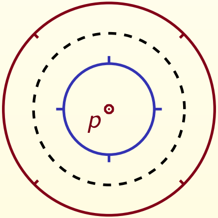


Variations of Vantage-Point Trees

- **Burkhard-Keller tree:** pivot used to divide the space into m rings   Burkhard&Keller'73
- **MVP-tree:** use the same pivot for different nodes in one level   Bozkaya&Ozsoyoglu'97
- **Post-office tree:** use $r_p + \delta$ for inner branch, $r_p - \delta$ for outer branch   McNutt'72







Generalized Hyperplane Tree

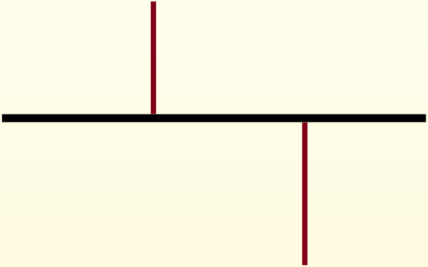
Partitioning technique (Uhlmann'91):

- Pick two objects (called pivots) p_1 and p_2
- Put all objects that are closer to p_1 than to p_2 to the left branch, others to the right branch
- Recursively repeat

Generalized Hyperplane Tree

Partitioning technique (Uhlmann'91):

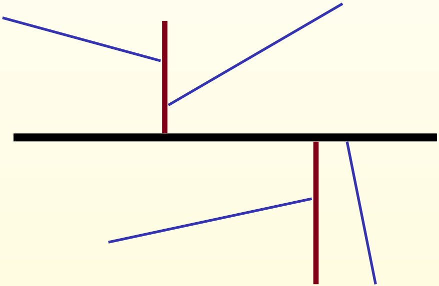
- Pick two objects (called pivots) p_1 and p_2
- Put all objects that are closer to p_1 than to p_2 to the left branch, others to the right branch
- Recursively repeat



Generalized Hyperplane Tree

Partitioning technique (Uhlmann'91):

- Pick two objects (called pivots) p_1 and p_2
- Put all objects that are closer to p_1 than to p_2 to the left branch, others to the right branch
- Recursively repeat

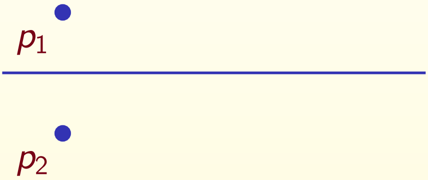


GH-Tree: Pruning Conditions

For r-range search:

If $d(q, p_1) > d(q, p_2) + 2r$ prune the left branch
If $d(q, p_1) < d(q, p_2) - 2r$ prune the right branch

For $|d(q, p_1) - d(q, p_2)| \leq 2r$ we have to inspect both branches

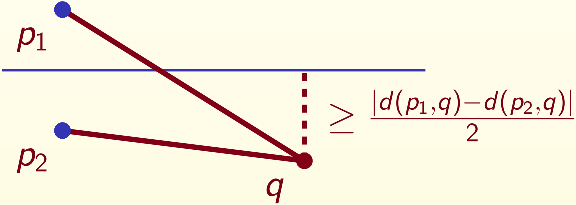


GH-Tree: Pruning Conditions

For *r*-range search:

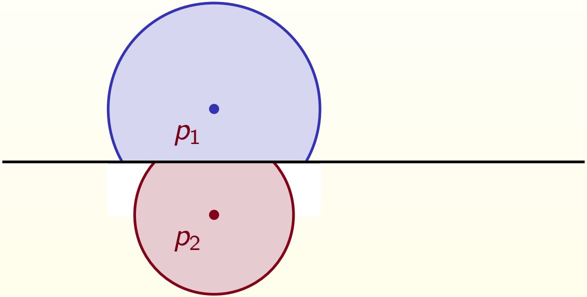
If $d(q, p_1) > d(q, p_2) + 2r$ prune the left branch
If $d(q, p_1) < d(q, p_2) - 2r$ prune the right branch

For $|d(q, p_1) - d(q, p_2)| \leq 2r$ we have to inspect both branches



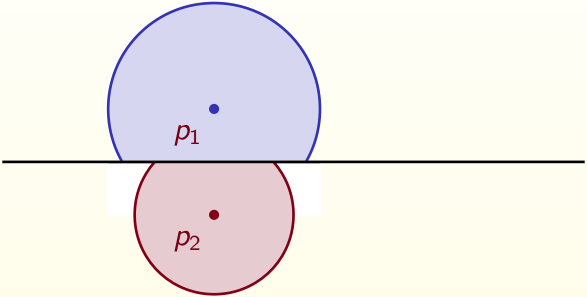
Bisector trees

Let's keep the covering radius for p_1 and left branch, for p_2 and right branch: useful information for stronger pruning conditions



Bisector trees

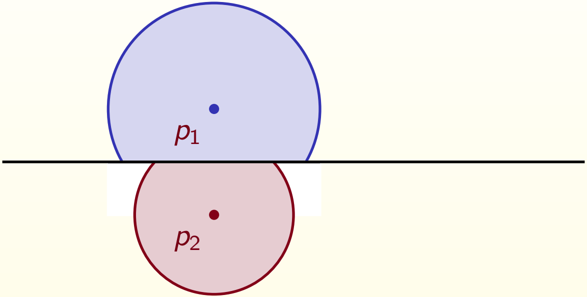
Let's keep the covering radius for p_1 and left branch, for p_2 and right branch: useful information for stronger pruning conditions



Variation: monotonous bisector tree (Noltemeier, Verbarg, Zirkelbach'92) always uses parent pivot as one of two children pivots

Bisector trees

Let's keep the covering radius for p_1 and left branch, for p_2 and right branch: useful information for stronger pruning conditions



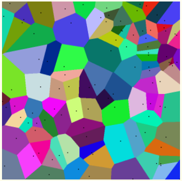
Variation: monotonous bisector tree (Noltemeier, Verbarg, Zirkelbach'92) always uses parent pivot as one of two children pivots

Exercise: prove that covering radii are monotonically decrease in mb-trees

Geometric Near-Neighbor Access Tree

Brin'95:

- Use m pivots
- Branch i consists of objects for which p_i is the closest pivot
- Stores minimal and maximal distances from pivots to all "brother"-branches



Part II

M-trees

M-tree: Data structure

Ciaccia, Patella, Zezula'97:

- All database objects are stored in leaf nodes (buckets of fixed size)
- Every internal nodes has associated pivot, covering radius and legal range for number of children (e.g. 2-3)
- Usual depth-first or best-first search

M-tree: Data structure

Ciaccia, Patella, Zezula'97:

- All database objects are stored in leaf nodes (buckets of fixed size)
- Every internal nodes has associated pivot, covering radius and legal range for number of children (e.g. 2-3)
- Usual depth-first or best-first search

Special algorithms for insertions and deletions a-la B-tree

M-tree: Insertions

All insertions happen at the leaf nodes:

1. Choose the leaf node using "minimal expansion of covering radius" principle

M-tree: Insertions

All insertions happen at the leaf nodes:

1. Choose the leaf node using “minimal expansion of covering radius” principle
2. If the leaf node contains fewer than the maximum legal number of elements, there is room for one more. Insert; update all covering radii

M-tree: Insertions

All insertions happen at the leaf nodes:

1. Choose the leaf node using “minimal expansion of covering radius” principle
2. If the leaf node contains fewer than the maximum legal number of elements, there is room for one more. Insert; update all covering radii
3. Otherwise the leaf node is split into two nodes

M-tree: Insertions

All insertions happen at the leaf nodes:

1. Choose the leaf node using "minimal expansion of covering radius" principle
2. If the leaf node contains fewer than the maximum legal number of elements, there is room for one more. Insert; update all covering radii
3. Otherwise the leaf node is split into two nodes
    1. Use two pivots generalized hyperplane partitioning

M-tree: Insertions

All insertions happen at the leaf nodes:

1. Choose the leaf node using “minimal expansion of covering radius” principle
2. If the leaf node contains fewer than the maximum legal number of elements, there is room for one more. Insert; update all covering radii
3. Otherwise the leaf node is split into two nodes
    1. Use two pivots generalized hyperplane partitioning
    2. Both pivots are added to the node’s parent, which may cause it to be split, and so on

Part III

k-d Trees, R-trees

Advantages of Euclidean Space

- Rich mathematical formalisms for defining a boundary of any set

  Examples: rectangles, hyperplanes, polynomial curves

- Easy computation of lower bound on distance between query point and any set boundary

- Easy definable mappings to smaller spaces

k-d Tree

Preprocessing: Bentley, 1975

Top-down partitioning

On level l: split the current set

by hyperplane orthogonal to $l \bmod k$ axis

k-d Tree

Preprocessing: Bentley, 1975

Top-down partitioning

On level l: split the current set

by hyperplane orthogonal to $l \bmod k$ axis

Query processing:

Standard branch and bound

k-d Tree

Preprocessing: Bentley, 1975

Top-down partitioning

On level l: split the current set

by hyperplane orthogonal to $l \bmod k$ axis

Query processing:

Standard branch and bound

k-d Tree

Preprocessing: Bentley, 1975

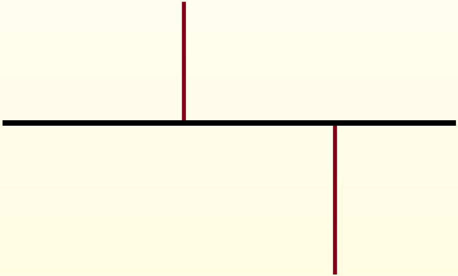
Top-down partitioning

On level l: split the current set

by hyperplane orthogonal to $l \bmod k$ axis

Query processing:

Standard branch and bound



k-d Tree

Preprocessing: Bentley, 1975

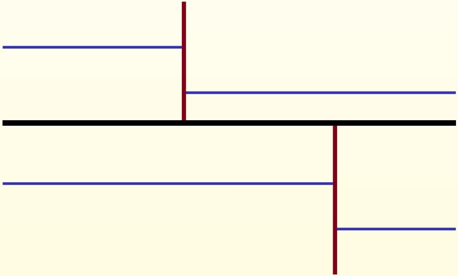
Top-down partitioning

On level l: split the current set

by hyperplane orthogonal to $l \bmod k$ axis

Query processing:

Standard branch and bound



R-Tree

Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

R-Tree

Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

Query processing:

Standard branch and bound

R-Tree

Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

Query processing:

Standard branch and bound

Insertions/delitions: similar to M-tree, B-tree

R-Tree

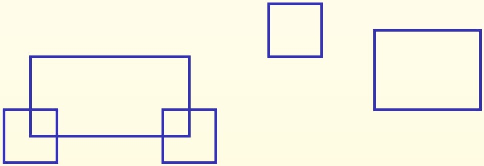
Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

Query processing:

Standard branch and bound

Insertions/delitions: similar to M-tree, B-tree



R-Tree

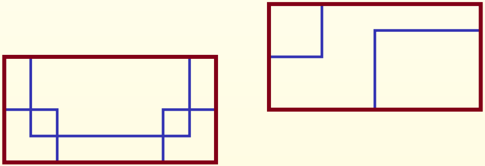
Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

Query processing:

Standard branch and bound

Insertions/delitions: similar to M-tree, B-tree



R-Tree

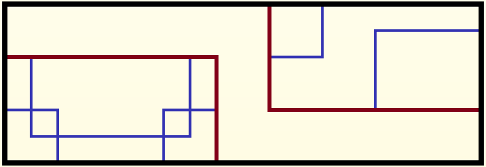
Preprocessing: Guttman, 1984

Bottom-up partitioning
Keep bounding rectangles
Every time: merge current rectangles
and compute bounding rectangle for every group

Query processing:

Standard branch and bound

Insertions/delitions: similar to M-tree, B-tree



Thanks for your attention! Questions?

References

Course homepage http://yury.name/algoweb.html

P. Zezula, G. Amato, V. Dohnal, M. Batko
Similarity Search: The Metric Space Approach. Springer, 2006.
http://www.nmis.isti.cnr.it/amato/similarity-search-book/

E. Chávez, G. Navarro, R. Baeza-Yates, J. L. Marroquín
Searching in Metric Spaces. ACM Computing Surveys, 2001.
http://www.cs.ust.hk/~leichen/courses/comp630j/readings/acm-survey/searchinmetric.pdf

G.R. Hjaltason, H. Samet
Index-driven similarity search in metric spaces. ACM Transactions on Database Systems, 2003
http://www.cs.utexas.edu/~abhinay/ee382v/Project/Papers/ft_gateway.cfm.pdf