An Optimal and Progressive Algorithm for Skyline Queries

Presenter: Karan Ashar
Dimitris Papadias Yufei Tao Greg Fu Bernhard Seeger
SIGMOD, 2003

Contents
- Introduction
  - Skyline queries
  - Existing solutions
  - Motivation
- Algorithms BBS
- Other discussions
- Experiments
- Conclusion

Contents
- Introduction
  - Skyline queries
  - Existing solutions
  - Motivation
- Algorithms BBS
- Other discussions
- Experiments
- Conclusion

Finding the Cheapest & Closest Hotels

Skyline Objects
- A set of objects not dominated by any other objects.
- Dominating region

Existing Solutions
- Block Nested Loop (BNL)
- Divide and Conquer (D&C)
- Bitmap method
- Index method
- Nearest Neighbor (NN)

Existing Solutions
- Block Nested Loop (BNL)
  - Scan the dataset and keep a list of candidate skyline points.
  - Compare a point p with every other point in the list.
- Advantages
  - Wide applicability
- Disadvantages
  - Numerous comparisons, inadequacy for on-line processing
**Existing Solutions**

- **Divide- and- Conquer (D&C)**
  - Divide the dataset into several partitions.
  - Compute partial skylines in each partition.
  - Compute global skylines by merging them.

- **Nearest Neighbor (NN)**
  - Find nearest neighbor point.
  - Divide the space by the nearest neighbor point.
  - Compute recursively until empty space.

- **NN over three or more dimensions**
  - Has overlapped partitions in divided subspaces.
  - Needs duplicate elimination.

**Motivation**

- **Advantages of NN algorithm**
  - Fast running time to finding the first result
  - Progressiveness

- **Disadvantages of NN algorithm**
  - Redundant I/O computation
  - Explosive to do list size

*Do you think there exists more efficient and useful skyline algorithm?*

- **Goal**: Improve NN algorithm and offer useful variations.

---

**R-Tree: Clustering by Proximity**

- **Minimum Bounding Rectangle (MBR)**

---

**Contents**

- **Introduction**
- **Algorithms BBS**
  - Preliminary: R-Tree
  - How BBS works on
  - Example
- **Other Discussions**
- **Experiments**
- **Conclusion**
R-Tree

Branched and Bound Skyline (BBS)

• Assume all points are indexed in an R-tree.
• Top-down Approach
  • mindist = the L_1 distance between its lower-left corner and the origin.
  \[ f(x, y) = x + y \]
• Data structure
  • Heap by min distance
  • List to maintain the current skyline
• Dominance check condition
  • Before expanding or inserting, compare to current skylines.
  • Before inserting an object, also check for internal objects.
• Stop condition: empty heap

Example of BBS

• Each heap entry keeps the mindist of the MBR.
• Process entries in ascending order of their mindists.
Example of BBS

Example of BBS

Example of BBS

Example of BBS

Example of BBS

Contents

- Introduction
- Algorithms BBS
- Other Discussions
  - Constrained skyline queries
  - K-dominating queries
- Experiments
- Conclusion
Constrained Skyline Queries

Constrained Skyline Queries

Constrained Skyline Queries

Constrained Skyline Queries

Constrained Skyline Queries

K-dominating Queries

- Retrieve 3 points that dominate the largest number of other points.
  - num(i) = 9, num(a) = 2, num(k) = 2
  - 3-dominating result: {i}
- h and m may dominate at most 7 points. (num(k) - 2)
  - 3-dominating result: {i}
**K-dominating Queries**

- num(h)=7, num(m)=5, num(a)=2, num(k)=2
- c and g may dominate at most 5 points, (num(h) – 2)
- 2-dominating result: {i, h}, 3-dominating result: {i, h, m}

**Experiments (Comparing BBS with NN)**

- Datasets:
  - Independent (uniform), anti-correlated
- Dimensionality:
  - In range [2,5]
- Cardinality:
  - In range [100k,10M]
- Machine:
  - Pentium 4 CPU
  - 2.4 GHz
  - 512MB Ram

**EXP 1: Effect of dimensionality**

- Independent
- Anti-correlated

**EXP 2: Effect of cardinality**

- Independent
- Anti-correlated

**Contents**

- Introduction
- Algorithms BBS
- Other Discussions
- Experiments
  - Comparing BBS with NN
  - Comparing BBS with NN on Constrained Queries
- Conclusion
**EXP 3: Progressive behavior**

**EXP 4: Comparison between BBS and NN on constrained skyline query**

**Experiments - Key Notes**

- **In General BBS outperforms NN significantly**
- **For dimensionality**
  - NN does not terminate when \( d \) increases due to explosive to do list
- **For cardinality**
  - NN does not terminate here as well if \( N \geq 5M \) due to to do list
- **For Progressive behavior**
  - NN requires more node access and CPU time to return number of reported points

**Contents**

- **Introduction**
- **Algorithms BBS**
- **Other Discussions**
- **Experiments**
- **Conclusion**

**Conclusion**

- **Progressive:** The first results are given to user almost instantly.
- **Absence of false misses:** Given enough time, the algorithm will generate the entire skyline.
- **Fair:** The algorithm doesn’t favor points that are particularly good in one dimension.
- **Incorporation of preferences:** It allows users to determine the order according to which skyline points are returned (eg: Constrained skyline queries)
- **Universal:** The algorithm is applicable to any dataset distribution and dimensionality, using some standard index structure.
- **No pre-computation (except building the R-Tree)**
- **Future Work:** Investigate alternatives for high dimensional spaces where R-Trees are inefficient.

**Discussion**

- **Is this algorithm perfect?**
- **NO!!**
- **CPU cost?**
- **Number of dominance checks?**
- **Solution**
  - VS2 and B2S2 algorithms which utilize the geometric properties of skyline.
That’s it.

Q&A

Thank you!