Optimizing Multidimensional skyline queries

Sofian Maabout
University of Bordeaux
LaBRI
France
Overview

• Skyline queries?
• Multidimensional Skylines
• Problem definition
• The interplay between functional dependencies and skylines
• Our solution
• Some experimental results
Skyline query aka *Pareto front*

- Best hotels are those *not dominated*
- O in the skyline iff there is no other O’ better than O
- Skyline={a, b, c, d} not dominated by any hotel
Skyline of New York buildings
Basics

• O dominates O’ iff
  1. \( O[i] \leq O'[i] \) for every i and
  2. There exists at least one j such that \( O[j] < O'[j] \)

• \( O1=\langle 1, 3, 2 \rangle, O2<2, 3, 2>, O3<2, 3, 1> \)
  – \( O1 \) dominates \( O2 \)
  – \( O1 \) and \( O3 \) are incomparable
  – \( O3 \) dominates \( O2 \)
Complexity of skyline computation

• Time :
  – Naïve algorithm $O(n^2)$
  – «Sophisticated algorithm» : $O(n^*|\text{Skyline}|)$
    • Note that at worst, $|\text{Skyline}|=n$

• Space :
  – Naïve algorithm : $O(1)$
  – «Sophisticated algorithm» : $O(|\text{Skyline}|)$
Naïve Algorithm

For $i = 1$ to $n$
    $j=1$
    While $j<=n$ and $S[i]$ not dominated by $S[j]$
        $j=j+1$
    If $j>n$ then
        add $S[i]$ to result
Return result
More sophisticated

Let $Rank(O) = \sum O[i]$

Property: $Rank(O) < Rank(O') \Rightarrow O'$ cannot dominate $O$

Sort $S$ wrt $Rank$
Put $S[1]$ into the result
For $i=2$ to $n$
    For $j=1$ to $result$.size()
        if $result[j]$ dominates $S[j]$
            dominated=true
            break
    if not dominated
        add $S[j]$ to $result$
Multidimensional Skylines

- Users are allowed to ask queries using any combination of dimensions
  
  - Rich: Best hotels = closest to the beach and largest rooms, regardless the price
    
    • Note that we want to maximize the superficy of rooms
  
  - Student: Best hotels = cheapest and wifi included regardless rooms surfaces
Multidimensional skylines

\[ t_5 \text{ dominates } t_6 \text{ wrt } A \]
\[ t_5 \text{ doesn’t dominate } t_6 \text{ wrt } AB \]
Skylines are not monotone

<table>
<thead>
<tr>
<th>Att</th>
<th>Skyline</th>
<th>Att</th>
<th>Skyline</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABCD</td>
<td>{t_2, t_3, t_4}</td>
<td>ABC</td>
<td>{t_2, t_4}</td>
</tr>
<tr>
<td>ABD</td>
<td>{t_1, t_2, t_3, t_4}</td>
<td>ACD</td>
<td>{t_2, t_3, t_4}</td>
</tr>
<tr>
<td>BCD</td>
<td>{t_2, t_4}</td>
<td>AB</td>
<td>{t_1, t_2}</td>
</tr>
<tr>
<td>AC</td>
<td>{t_2, t_4}</td>
<td>AD</td>
<td>{t_1, t_2, t_3, t_4}</td>
</tr>
<tr>
<td>BC</td>
<td>{t_2, t_4}</td>
<td>BD</td>
<td>{t_1, t_2, t_4}</td>
</tr>
<tr>
<td>CD</td>
<td>{t_4}</td>
<td>A</td>
<td>{t_1, t_2}</td>
</tr>
<tr>
<td>B</td>
<td>{t_1, t_2}</td>
<td>C</td>
<td>{t_4}</td>
</tr>
<tr>
<td>D</td>
<td>{t_4}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sky(T, ABD) not included into Sky(T, ABCD)
Sky(T, AB) incomparable to Sky(T, ABC)
Optimizing multidimensional skylines

• Users can ask skylines wrt any dimensions combination \( 2^d \) possible queries

• 2 main directions so far:
  – Pre-compute all queries:
    - Large computation time  -- Large storage space
    + Perfect query response time
  – Pre-compute equivalent queries
    - - Large computation time  ± moderate storage space
    + Perfect query response time

• Our proposal: Precompute some queries
  ± moderate precomputation time, ± moderate storage space, ± moderate query response time
Problem statement

- Def: X is **ancestor** of Y iff
  (i) $X \supseteq Y$ and
  (ii) $\text{Sky}(X) \supseteq \text{Sky}(Y)$
- Fact: X ancestor of Y $\Rightarrow$ $\text{Sky}(T, Y) = \text{Sky}(\text{Sky}(T,X), Y)$

Pbm: select a minimal set of skylines sufficient to answer every skyline from a materialized ancestor

- Naïve solution:
  - Compute $S= \text{all skylines}$
  - For each $s1, s2$
    - If $s1$ is an ancestor of $s2$ then remove $s2$
Functional dependencies

• $X \rightarrow Y$ iff every value of $X$ is always associated to the same value of $Y$.

<table>
<thead>
<tr>
<th>RiD</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>$t_3$</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>$t_4$</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>$t_5$</td>
<td>3</td>
<td>9</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>$t_6$</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

A $\rightarrow$ B
BC $\rightarrow$ A
B $\not\rightarrow$ A

Theorem: If $X \rightarrow Y$ then $\text{Sky}(X) \subseteq \text{Sky}(XY)$
Ex: $\text{Sky}(A) \subseteq \text{Sky}(AB)$
Closed subspace

• X is closed iff $X \not\supset A$ for every A not in X

• The minimal FD’s satisfied by T are

$$A \rightarrow BD \quad BD \rightarrow A$$
$$BC \rightarrow AD \quad CD \rightarrow AB$$

C is closed
AB is not closed
Minimal set of Skylines

1. Find the closed subspaces
2. compute their skylines
3. test skylines inclusion between descendent/ancestor candidate pairs
Search space lattice

ABCD

ABC  ABD  ACD  BCD

AB  AC  AD  BC  BD  CD

A  B  C  D
Minimal solution

All closed subspaces are below minimal keys

Thm: Minimal solution is a subset of closed subspaces
Red : closed subspace
The minimal set of skylines to materialize is \{ABD, ABCD\}

Figure 3: The Partial Skycube
Experiments

• Our solution vs other proposals for fully computing the skycube

• Our solution vs a closed skycubes: a losseless compression technique

• Assess query evaluation time
Experiments: (1) compute all skylines

Algorithm 3: FMC algorithm

Input: Table $T$
Output: Skycube of $T$

1 $Closed = \text{ClosedSubspaces}(T)$;
2 foreach $X \in Closed$ do
3 \quad Compute $Sky(T, X)$;

4 foreach subspace $X$ do
5 \quad Compute $Sky(Sky(X^+), X)$;

6 Return $\bigcup_{X \in 2^D} Sky(X)$;
Experiments: (1) compute all skylines
Real data set. USCensus : \( n \approx 2 \times 10^6 \)

- For \( d > 14 \), others consume all available memory (32G)
Experiments: (1) compute all skylines
Synthetic data sets

Independent  Correlated  Anti-correlated
Experiments: (1) compute all skylines
Synthetic data sets

Figure 5: Speedup w.r.t dimensionality $d$ ($n = 100K$)
Experiments: (1) compute all skylines
Synthetic data sets

Figure 6: Speedup w.r.t. data size $n$ (d = 16)
Experiments: (2) query optimization
1000 random skyline queries

<table>
<thead>
<tr>
<th>n\d</th>
<th>16</th>
<th>18</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>500K</td>
<td>0.024/18.9 (1.19%)</td>
<td>0.026/22.54 (0.55%)</td>
<td>0.027/25.78 (0.13%)</td>
</tr>
<tr>
<td>1M</td>
<td>0.034/36.78 (2.197%)</td>
<td>0.036/44.41 (1.098%)</td>
<td>0.047/49.68 (0.274%)</td>
</tr>
<tr>
<td>2M</td>
<td>0.041/73.74 (2.22%)</td>
<td>0.044/87.92 (1.45%)</td>
<td>0.049/99.92 (0.31%)</td>
</tr>
</tbody>
</table>

- 0.31% out of the $2^{20}$ queries are materialized.
- 49 ms to answer 1K skyline queries from the materialized ones instead of
- 99.92 seconds from the underlying data.
- Speed up > 2000
Experiments: (3) comparison with closed skycubes

- Identify equivalent skylines and store just one copy ➔ compression of the whole skylines set
- E.g., Sky(C), Sky(D) and Sky(CD) are equivalent

<table>
<thead>
<tr>
<th>Att</th>
<th>Skyline</th>
<th>Att</th>
<th>Skyline</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABCD</td>
<td>{t_2, t_3, t_4}</td>
<td>ABC</td>
<td>{t_2, t_4}</td>
</tr>
<tr>
<td>ABD</td>
<td>{t_1, t_2, t_3, t_4}</td>
<td>ACD</td>
<td>{t_2, t_3, t_4}</td>
</tr>
<tr>
<td>BCD</td>
<td>{t_2, t_4}</td>
<td>AD</td>
<td>{t_1, t_2, t_3, t_4}</td>
</tr>
<tr>
<td>AC</td>
<td>{t_2, t_4}</td>
<td>BD</td>
<td>{t_1, t_2, t_4}</td>
</tr>
<tr>
<td>BC</td>
<td>{t_2, t_4}</td>
<td>AB</td>
<td>{t_1, t_2}</td>
</tr>
<tr>
<td>CD</td>
<td>{t_4}</td>
<td>AD</td>
<td>{t_1, t_2, t_3, t_4}</td>
</tr>
<tr>
<td>B</td>
<td>{t_1, t_2}</td>
<td>A</td>
<td>{t_1, t_2}</td>
</tr>
<tr>
<td>D</td>
<td>{t_4}</td>
<td>C</td>
<td>{t_4}</td>
</tr>
</tbody>
</table>
Experiments: (3) comparison with closed skycubes

Storage space: 2 Skylines vs. 6
Query response time: Closed skycubes are better
Experiments: (3) comparison with closed skycubes

<table>
<thead>
<tr>
<th></th>
<th>NBA</th>
<th>IPUMS</th>
<th>MBL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MICS</td>
<td>5304 (12.8 sec.)</td>
<td>11 (2.1 sec.)</td>
<td>29155 (172 sec.)</td>
</tr>
<tr>
<td>Closed</td>
<td>5304 ( 9 sec.)</td>
<td>738 (322 sec.)</td>
<td>43075 (11069 sec.)</td>
</tr>
</tbody>
</table>

Number of materialized skylines (time to find and materialize them)

Synthetic correlated data: n=100K, d=20: MICS=20sec, Closed didn’t finish after 36 hours
Trends: fixed #tuples

Number of distinct values/dimension vs. # FD’s vs. # closed subspaces
Trends: fixed number of dimensions

Worst situation: all subspaces are closed !!

But there is a hope 😊
Trends: fixed number of dimensions

Intuition: the more we have tuples, the more we have chances to have the smallest tuples
Case where skylines are « small »

Property: Let $X \subseteq Y$. Then $t \in \text{Sky}(T, X)$ iff there exists $t' \in \text{Sky}(\text{Sky}(T, Y), X)$ such that $t[X]=t'[X]$

$\Rightarrow$ We can « easily » recover $\text{Sky}(X)$ from $\text{Sky}(Y)$
Example

Sky(ABCD)={ t2, t3, t4}

Sky(Sky(ABD), AB)={t2<1,3>}

⇒ t1 is also in Sky(AB) since t1[AB]=<1,3>
Ongoing and future works

• Deal with data insertion/deletion

• When data are distributed, are local or/and global FD’s helpful?

• Approximate FD’s for soft skylines
  – A room whose price 30$ doesn’t clearly dominate a room whose price is 30.1$

• Reduce the size of a skyline
  – From each skyline, keep those that dominate the largest number of objects
Ongoing and future works

• Given a storage space threshold $\tilde{S}$ ($\geq |\text{MICS}|$) find the best skylines set $S$ to materialize in order to optimize all skylines queries while storage$(S) \leq \tilde{S}$

• Moving reference vs fixed reference
  – Apps: Best restaurant in the neighborhood

• Communication cost with cell phones
  – Once sky(ABCD) is received, sky(ABC) doesn't need communication if ABC->D $\Rightarrow$ local computation