Branch-and-Bound Algorithm for Reverse Top-$k$ Queries

Akrivi Vlachou$^{1,2}$, Christos Doulkeridis$^{1,3}$, Kjetil Nørvåg$^1$, Yannis Kotidis$^4$

$^1$Norwegian University of Science and Technology (NTNU), Trondheim, Norway
$^2$Institute for the Management of Information Systems, R.C. “Athena”, Greece
$^3$University of Piraeus, Piraeus, Greece
$^4$Athens University of Economics and Business (AUEB), Greece
Outline

- Motivation
- Background
- Branch-and-Bound Reverse Top-$k$ Algorithm
  - Score Bounding Properties
  - INTOPk: Pruning and Result Inclusion
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
Rank-aware Query Processing

- Huge amount of available data
- Users prefer to retrieve a limited set of $k$ ranked data objects that best match their preferences (top-$k$ queries)
Reversing the Top-\(k\) Query

- From the perspective of manufacturers:
  - estimate the **impact** of a product compared to their competitors products
  - **advertise** a product to potential customers

- **Reverse top-\(k\) query:**
  Given a potential **product** \(q\) and a positive integer \(k\), which are the **user preferences** for which \(q\) is in the top-\(k\) query result set?
Outline

- Motivation
- **Background**
- Branch-and-Bound Reverse Top-$k$ Algorithm
  - Score Bounding Properties
  - INTOPk: Pruning and Result Inclusion
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
A database containing information about different cars
Different users have different preferences
Top-$k$ Query

- Given a scoring function $f()$, retrieve the $k$ objects that best match the user preferences.
- Linear scoring function
  $$f_w(p) = \sum w[i] * p[i]$$
  weight $w[i]$: relative importance of attribute $i$
- Definition $\text{TOP}_k(w)$: Given a weighting vector $w$ and a positive integer $k$, find the $k$ data points $p$ with the minimum $f(p)$ scores.
**Top-\(k\) Query Example**

- Bob prefers a cheap car, and does not care much about the age
  - the best choice (top-1) for **Bob** is the car \(p_1\) with score 2.5
- Tom prefers a newer car rather than a cheap car
  - the best choice for **Tom** and **Max** is the car \(p_2\)
- \(p_2\) is preferred by more users than \(p_1\)
Reverse Top-k Query

- Given a point \( q \), a positive number \( k \) and two datasets \( S \) and \( W \), a weighting vector \( w_i \) belongs to the result set \( (bRTOP_k(q)) \), if and only if there exists \( p \) in \( \text{TOP}_k(w_i) \) such that \( f_{wi}(q) \leq f_{wi}(p) \).
- Given a product \( q \) and a positive integer \( k \), which are the weighting vectors \( w_i \) for which \( q \) is in the top-\( k \) query result set?
Reverse Top-$k$ Query Example

- Query point $q=p_2$, $k=1$:
  - reverse top-$k$ set is ${(0.2,0.8), (0.5,0.5)}$
  - advertise product to Tom and Max

- Query point $q=p_3$, $k=1$:
  - empty result set

- Naïve approach:
  - for each weighting vector process the top-$k$ query
  - test if query point $q$ belongs to the top-$k$ set

<table>
<thead>
<tr>
<th>user</th>
<th>w[price]</th>
<th>w[age]</th>
<th>top-1(score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>0.9</td>
<td>0.1</td>
<td>$p_1 (2.5)$</td>
</tr>
<tr>
<td>Tom</td>
<td>0.2</td>
<td>0.8</td>
<td>$p_2 (2.2)$</td>
</tr>
<tr>
<td>Max</td>
<td>0.5</td>
<td>0.5</td>
<td>$p_3 (3.0)$</td>
</tr>
</tbody>
</table>
Goal:
- reduce the number of top-$k$ evaluations by discarding weighting vectors

Threshold-based Algorithm (RTA):
- sort the weighting vectors based on pairwise similarity
  - top-$k$ queries defined by similar vectors, have similar result sets
- evaluate the first top-$k$ query, calculate a threshold
- for each weighting vector
  - possibly prune based on threshold
  - refine threshold
RTA vs BBR

- **Threshold-based Algorithm (RTA)**
  - accesses all stored weighting vectors
  - processes at least as many top-$k$ evaluations as the cardinality of the result set

- **Branch-and-Bound Reverse Top-$k$ Algorithm**
  - a set of weighting vectors can be immediately added to the result set
  - a set of weighting vectors can be excluded from the reverse top-$k$ results
Outline

- Motivation
- Background
- Branch-and-Bound Reverse Top-$k$ Algorithm
  - **Score Bounding Properties**
  - INTOPk: Pruning and Result Inclusion
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
Score Bounds

- Score-lower-bound of $p$
  \[ \ell_V(p) = \sum m_{V1} \ell[i] * p[i] \]
- Score-upper-bound of $p$
  \[ u_V(p) = \sum m_{V1} u[i] * p[i] \]
- Score-lower-bound of $e_i$
  \[ \ell_V(e_i) = \sum m_{V1} \ell[i] * e_i \cdot \ell[i] \]
- Score-upper-bound of $e_i$
  \[ u_V(e_i) = \sum m_{V1} u[i] * e_i \cdot u[i] \]
Score Precedence

- no point $p \in m$ can affect the rank of $q$ for any weighting vector $w \in V$
- all points $p \in m$ have a better rank than $q$ for any weighting vector $w \in V$
- $q$ may have a better or worse score than some $p_i \in m$ for $w \in V$
Outline

- Motivation
- Background
- Branch-and-Bound Reverse Top-k Algorithm
  - Score Bounding Properties
  - \textbf{INTOPk: Pruning and Result Inclusion}
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
Pruning Property

- Pruning property: Given an MBR $m_V$, if $k$ data items (MBRs or data points) precede $q$ based on $V$ then $m_V$ can be safely pruned
  - no weighting vector $w \in V$ belongs to the reverse top-$k$ result of $q$
• Result inclusion: Given an MBR $m_V$, if fewer than $k$ data points $p_i$ exist such that $u_V(q) > \ell_V(p_i)$, then all weighting vectors $w \in V$ can be safely added to the reverse top-$k$ result of $q$. 
INTOPk Algorithm

- For a given MBR $m_V$
  - traverses the R-tree of data set $S$
  - decides if $m_V$ belongs to the result set or not
Outline

- Motivation
- Background
- Branch-and-Bound Reverse Top-$k$ Algorithm
  - Score Bounding Properties
  - INTOPk: Pruning and Result Inclusion
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
INTOPk is inconclusive for the root
INTOPk discards $e_1$
Optimizations of BBR

- **Basic BBR:**
  - for each processed $m_V$, an INTOPk query is posed
  - INTOPk queries (I/Os) should be avoided

- **BBR with Result sharing (BBR*):**
  - BBR discards an entry $m_V$ based on $k$ data items
  - these data items may also discard other (MBRs of) weighting vectors
  - BBR* maintains a set of data items in a list of bounded size $k$
• BBR with aggregate R-tree (BBRA):
  • aggregate R-tree: each entry is annotated with the number of all points contained in its subtree
  • BBRA counts the data points instead of data entries
Outline

- Motivation
- Background
- Branch-and-Bound Reverse Top-$k$ Algorithm
  - Score Bounding Properties
  - INTOPk: Pruning and Result Inclusion
  - Branch-and-Bound Algorithm
- Experimental Evaluation
- Summary
Experimental Setup

- Comparison between Branch-and-Bound algorithms (BBR, BBR*, BBRA) and RTA
  - varying dimensionality (n:2-9), cardinality (|S|:100K-5M, |W|:100K-1M), value of $k$ (10-50), data distribution (S:UN,CO,AC,CL, W:UN,CL) – real data (House: 127930 6d, Color: 68040 9d)

- Queries:
  - $k$-skyband
  - skyline points

- Metrics:
  - Time
  - I/Os
Comparison for Increased Dimensionality

Uniform distribution of $S$ and Uniform weights $W$

$|S|=100K$, $|W|=100K$, $top-k=10$, skyband query points

- BBR* and BBRA improve RTA by a factor of 4-8

- White bar: I/Os on $S$
- Colored bar: I/Os on $W$
- BBR* and BBRA: 4 times better than RTA
• BBR* and BBRA need fewer INTOPk evaluations than BBR
• Discarded MBRs
  • many due to result sharing, without invoking INTOPk
• Added MBRs
  • All algorithms add groups of vectors to the result set
Scaling with Number of Weighting Vectors

Uniform distribution of $S$ and Uniform weights $W$

$|S|=100K$, $n=4$, top-$k=10$, skyband query points

- The Branch-and-Bound algorithms scale nicely with increasing size of weights data set (*due to the efficient pruning*)
- RTA is more sensitive to increased values of $|W|$
Performance on Real Data

HOUSE consists of 127930 tuples, \(d=6\) (income spent on gas, electricity, water, heating, insurance, and property tax)
COLOR consists of 68040 9-dimensional tuples describing features of images in HSV color space
For varying different parameters, including:

- Higher dimensionality $n$ (up to 9d)
- Other data distributions for $S$ (AC, CO, CL)
- Other data distributions for $W$ (CL)
- Increasing the cardinality of data objects $|S|$
- Increasing the value of $k$
Outline

• Motivation
• Background
• Branch-and-Bound Reverse Top-$k$ Algorithm
  • Score Bounding Properties
  • INTOPk: Pruning and Result Inclusion
  • Branch-and-Bound Algorithm
• Experimental Evaluation
• Summary
Reverse top-$k$ queries are important for market analysis
- identify customers who are potentially interested in a product based on the customer preferences and the competitors’ products.

State-of-the-art algorithm (RTA)
- accesses each individual preference
- cannot add a preference function to the result set without evaluating a top-$k$ query

Our novel branch-and-bound algorithm (BBR)
- adds to the result set or discards sets of preferences instead of individual preferences

BBR* and BBRA outperform RTA and perform efficiently in all cases
Thank you!

More information: http://www.idi.ntnu.no/~vlachou/