Promotion Analysis in Multi-Dimensional Space

Tianyi Wu (UIUC)
Dong Xin (Microsoft Research)
Qiaozhu Mei (University of Michigan)
Jiawei Han (UIUC)
Outline

- Introduction
- Query execution algorithms
- Spurious promotion
- Experiment
- Conclusion
Outline

- Introduction
- Query execution algorithms
- Spurious promotion
- Experiment
- Conclusion
Promotion analysis: introduction

- Formulate and study a useful function
  - Promotion analysis through ranking
  - General goal: promote a given object by leveraging subspace ranking
- Motivating example
  - A marketing manager of a book retailer
  - Basic fact
    - Book sales: 30th out of 100 other retailers
    - Not particularly interesting!
  - After promotion analysis, he discovered:
    - Ranked 1st in the \{college students, science and technology\} area
    - Further advertising and marketing decisions
- Another example: person promotion

Let’s promote our brand!
# Promotion query

## Observation

<table>
<thead>
<tr>
<th>Global rank</th>
<th>May not be interesting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-space</td>
<td>Compare to all other objects in all aspects</td>
</tr>
<tr>
<td>Low cost</td>
<td>Single SQL query</td>
</tr>
<tr>
<td>Local rank</td>
<td>Can be more interesting</td>
</tr>
<tr>
<td>Subspaces</td>
<td>Compare objects in certain areas</td>
</tr>
<tr>
<td>High cost</td>
<td>Many subspaces</td>
</tr>
</tbody>
</table>

## The Promotion Query Problem

**Given:** an object (e.g., product, person)

**Goal:** discover the most interesting subspaces where the object is highly ranked
Subspace rank: why interesting

- Discover merit and competitive strengths
  - *E.g., a bestselling car model among hybrid cars*
- Enhance image
  - *E.g., fortune 500 company*
- Facilitate decision making
  - *E.g., marketing plan that focuses on college students*
- Deliver specific information
  - *E.g., “top-3 university in biomedical research” vs. “top-20 university”*
- Extensively practiced in marketing
  - *Market segmentation*
  - *Customer targeting and product positioning*
Challenges

- Current systems
  - Given a condition, find top-\(k\) objects
  - Sophisticated early termination and pruning algorithms
- Promotion query: not well-supported
  - User: manual search and navigation
  - Trial-and-error
- Computationally expensive
  - The rank measure: holistic
  - A blow-up of subspaces

It should be good at …
Let me try some queries…
Promotion analysis
Multidimensional data model

- Fact table

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Object</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lyon</td>
<td>July</td>
<td>T</td>
<td>0.5</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>T</td>
<td>0.8</td>
</tr>
<tr>
<td>Chicago</td>
<td>August</td>
<td>S</td>
<td>1.0</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>S</td>
<td>1.0</td>
</tr>
<tr>
<td>Lyon</td>
<td>August</td>
<td>V</td>
<td>0.3</td>
</tr>
<tr>
<td>Chicago</td>
<td>August</td>
<td>V</td>
<td>0.6</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>V</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Subspace dimensions
Object dimension
Score dimension
## Subspaces

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Object</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lyon</td>
<td>July</td>
<td>T</td>
<td>0.5</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>T</td>
<td>0.8</td>
</tr>
<tr>
<td>Chicago</td>
<td>August</td>
<td>S</td>
<td>1.0</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>S</td>
<td>1.0</td>
</tr>
<tr>
<td>Lyon</td>
<td>August</td>
<td>V</td>
<td>0.3</td>
</tr>
<tr>
<td>Chicago</td>
<td>August</td>
<td>V</td>
<td>0.6</td>
</tr>
<tr>
<td>Chicago</td>
<td>July</td>
<td>V</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Given a **target object** $T$

### Subspaces of $T$

{\{*\}}

- **SUM($T$) = 1.3**
- **Rank($T$) = 3rd / 3**

{\{Lyon\}}

- **SUM($T$) = 0.5**
- **Rank($T$) = 1st / 2**

{\{Chicago\}}

- **SUM($T$) = 1.8**
- **Rank($T$) = 3rd / 3**

{\{July\}}

- **SUM($T$) = 1.3**
- **Rank($T$) = 1st / 3**

{\{Lyon, July\}}

- **SUM($T$) = 0.5**
- **Rank($T$) = 1st / 1**

{\{Chicago, July\}}

- **SUM($T$) = 0.8**
- **Rank($T$) = 2nd / 3**

**{*}** is the special case: full-space

Aggregate and compute the target object’s rank in each subspace.
Query model

- Given a target object $T$, find the top subspaces which are promotive
- “Promotiveness”: a class of measures to quantify how well a subspace $S$ can promote $T$
  - $P(S, T) = f(Rank(S, T)) \times g(Sig(S))$
    - Higher rank $\sim$ more promotive
    - More significant subspace (e.g., more objects) $\sim$ more promotive
- Example instantiations
  - Simple ranking: $P(S, T) = Rank^{-1}(S, T)$
  - Iceberg condition: $P(S, T) = Rank^{-1}(S, T) \times I(ObjCount(S) > MinSig)$
  - Percentile ranking: $P(S, T) = ObjCount(S) / Rank(S, T)$
  - ...
Query model

- Given a target object $T$, find the top subspaces which are promotive
- "Promotiveness": a class of measures to quantify how well a subspace $S$ promotes $T$

**The Promotion Query Problem**

<table>
<thead>
<tr>
<th>Input:</th>
<th>a target object $T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output:</td>
<td>top-R subspaces with the largest $P(S, T)$ scores</td>
</tr>
</tbody>
</table>

/* assume simple ranking */

- More significant subspace (e.g., more objects) — more promotive

- Example instantiations
  - **Simple ranking**: $P(S, T) = \text{Rank}^{-1}(S, T)$
  - **Iceberg condition**: $P(S, T) = \text{Rank}^{-1}(S, T) * I(\text{ObjCount}(S) > \text{MinSig})$
  - **Percentile ranking**: $P(S, T) = \frac{\text{ObjCount}(S)}{\text{Rank}(S, T)}$
  - ...
Outline

• Introduction

• Query execution algorithms
  • (1) PromoRank framework
    • (a) Subspace pruning
    • (b) Object pruning
  • (2) Promotion cubes

• Spurious promotion

• Experiment

• Conclusion
The PromoRank framework

Idea: use a recursive process to partition and aggregate the data to compute the target object’s rank in each subspace

[Beyer99] The bottom-up method

Target object’s subspace lattice
Recursive process

Compute T’s rank in \{\star\}
Method: create a hash table:
HashTable\{object\} = AggregateScore

Partition the data based on A
Method: sorting
Compute T’s rank in \{A\}

Recursively repeat…

Top-R promotive subspaces: priority queue
(1.1) Subspace pruning

- **Idea**: reuse previous results
- **Goal**: prune out unseen subspaces by bounding their promotiveness scores

- \( \text{Sig}(S) \): bounded
- \( \text{Rank}(S, T) \): bounded
Subspace pruning

**Keys:**
- Compute T’s highest possible Rank: LBRank
- Use the monotonicity of the aggregate measure (e.g. SUM, MAX)

How to prune an unseen one?

Given a seen (aggregated) subspace

Any unseen subspace with low LBRank(T) can be pruned

Thus, LBRank(T) = |{V, S}| + 1 = 3rd

```
SUM(V) > SUM(T)
SUM(S) > SUM(T)
```

```
SUM(T) = 1.9
SUM(V) = 5.5
```

```
SUM(S) = 2.2
```

```
SUM(T) = 1.1
Rank(T) = 3rd / 3
```
(1.2) Object pruning

Idea: avoid computing objects which do not affect rank
Goal: reduce the partitioning and aggregation cost

Power-law distribution: objects at the long-tail can be pruned

SUM(S) = 6.5
SUM(T) = 2.2
SUM(U) = 1.5
SUM(W) = 1.0
SUM(Z) = 0.8

SUM(W) < MinScore(T)
SUM(Z) < MinScore(T)

W and Z can be pruned!

Seen (aggregated) subspace
{A}

Unseen subtree of subspaces
{AB}
{AC}
{ABC}
SUM(T) = 1.2
SUM(T) = 1.9
SUM(T) = 1.1

MinScore(T) = 1.1
(2) Promotion cubes

Observation:
(1) T: tends to be highly ranked in a top subspace;
(2) A top subspace is likely to contain many objects

- **Method**: promotion cube
  - Offline materialization
  - Structure
    - For each subspace with $\text{Sig}(S) > \text{MinSig}$
      - parameter: $\text{MinSig}$
      - Materialize a selected sample of top-$k$ aggregate scores in each subspace
        - Parameter(s): $k$ and $k'$
Promotion cell

- For each “significant” subspace $S$, create a “promotion cell”

> Promotion cell:
  - Store aggregate scores; no object IDs
  - Parameters $MinSig$, $k$, and $k'$: chosen to yield a space-time tradeoff; application dependent
  - Does not restrict query processing

- Passing the $MinSig$ threshold

- $k=9$, $k'=3$
Query execution using promotion cube

- Step 1: Compute T’s aggregate scores
- Step 2: Compute LBRanks and UBRanks and do pruning
  - Using the promotion cube
- Step 3: Call PromoRank
Query execution using promotion cube

- Step 1: Compute T’s aggregate scores
- Step 2: Compute LBRanks and UBRanks and do pruning
  - Using the promotion cube
- Step 3: Call PromoRank
Outline

- Introduction
- Query execution algorithms
- Spurious promotion
- Experiment
- Conclusion
The spurious promotion problem

- Spurious promotion
  - The target object is highly ranked in a subspace due to random perturbation: not meaningful
- Example: Michael Jordan (NBA player)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Subspace</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>{Year = 1995}</td>
<td>OK</td>
</tr>
<tr>
<td>#1</td>
<td>{MonthOfBirth = February}</td>
<td>Spurious</td>
</tr>
<tr>
<td>#1</td>
<td>{Weather = Sunny}</td>
<td>Spurious</td>
</tr>
</tbody>
</table>
Avoid spurious promotion

- How to avoid such meaningless subspaces?
- **Observation**
  - Spuriously promotive dimension: mean aggregate scores tend to be similar across different dimension values

![Mean aggregate score vs. dimension](image1)

**BirthMonth**

![Mean aggregate score vs. dimension](image2)

**position**
Preprocessing to filter out spurious dimensions

Method:
- **ANOVA (analysis of variance) test**
- Given a subspace dimension $A$
  - $|A|$ groups of scores
  - Between-group sum of squared deviation
    $$SS_B = \sum_i \frac{\sigma_i}{size_i} - \left(\frac{\sum_i \sigma_i}{n}\right)^2$$
  - Within-group sum of squared deviation
    $$SS_W = \sum_i \sum_j (s^i_j - \mu_i)^2$$
- **F-ratio** ($A$) = $SS_B / SS_W$
- **F-ratio** too small: $H_0$ rejected; no correlation with score.

For each subspace dimension

ANOVA test

Spurious? Yes Remove

No

Query execution

Top-R non-spurious subspaces
Outline

• Introduction
• Query execution algorithms
• Spurious promotion
• Experiment
• Conclusion
Experiment

- Evaluation
  - Effectiveness (case study)
  - Efficiency (space-time tradeoff)
- Data sets
  - NBA
  - DBLP
  - TPC-H
- Methods
  - PromoRank
  - PromoRank++ (with the pruning methods)
  - PromoCube
- Implementation
  - Pentium 3GHz CPU / 2G memory
  - WinXP / Microsoft Visual C# 2008 (in-memory)
DBLP data set

- Subspace dimensions
  - Conference (2,506)
  - Year (50)
  - Database (boolean)
  - Data mining (boolean)
  - Information retrieval (boolean)
  - Machine learning (boolean)

  From title

- Object dimension: Author (450K)

- Score dimension: Paper count

- Base tuples (1.76M)
## A case study on DBLP

<table>
<thead>
<tr>
<th>Query object</th>
<th>Top-3 subspaces</th>
<th>Rank</th>
<th>Authors</th>
<th>Top-%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{*}, {Database}</td>
<td>376th</td>
<td>451,316</td>
<td>0.08%</td>
</tr>
<tr>
<td>David Dewitt</td>
<td>{1990}</td>
<td>16th</td>
<td>65,321</td>
<td>0.02%</td>
</tr>
<tr>
<td></td>
<td>{SIGMOD}</td>
<td>2nd</td>
<td>13,170</td>
<td>0.02%</td>
</tr>
<tr>
<td></td>
<td>{*}, {Database, 2003}</td>
<td>3325th</td>
<td>451,316</td>
<td>0.74%</td>
</tr>
<tr>
<td>Yufei Tao</td>
<td>{Database, 2004}</td>
<td>11th</td>
<td>6,707</td>
<td>0.16%</td>
</tr>
<tr>
<td></td>
<td>{ICDE}</td>
<td>18th</td>
<td>8,877</td>
<td>0.20%</td>
</tr>
<tr>
<td></td>
<td>{*}, {ICDE}</td>
<td>30th</td>
<td>4,822</td>
<td>0.62%</td>
</tr>
</tbody>
</table>

Promotiveness measure decided by rank and a penalty for small subspace.
Query execution time (DBLP)

Promotion cube = 310KB
(most aggregate scores are small integers)
ANOVA test: effectiveness

- **NBA data**
  - 3,460 *players* (objects)
  - *Rebounds* (score)
  - 18,050 base tuples
TPCH benchmark

- 6M tuples
- 6 subspace dimensions
- 10,000 objects
- Promotion cube
  - \( k = 1000, k' = 8 \)
  - Size < 1MB

---

**Query execution time vs. top-R**

- PromoRank
- PromoRank++
- PromoCube

---

**Query execution time vs. # objects**

- PromoRank
- PromoRank++
- PromoCube
Outline

- Introduction
- Query execution algorithms
- Spurious promotion
- Experiment
- Conclusion
Conclusion

- Promotion analysis: a new direction
  - Search-based advertising
    - [Borgs WWW 07] Dynamics of bid optimization in online advertisement auctions
  - Data mining for marketing
    - [Kleinberg DMKD 98] A microeconomic view of data mining
  - Finding top-\(k\) attributes
    - [Das SIGMOD 06] Ordering the attributes of query results
    - [Miah ICDE 08] Standing out in a crowd: Selecting attributes for maximum visibility
- Skyline queries

- Future
  - Application: social networks, recommender systems, …
  - Data model: links, textual data, numerical, …
Thank you!

Any questions?