OLAP Query Personalization

6th Conference on Data Warehousing and On-Line Analysis
Djerba 11, June

Matteo Golfarelli
DEIS – University of Bologna

Summary

- OLAP query evolution and improvements
- Query personalization
  - Contextualization
  - Recommending
  - Visualization
- Preference queries
- myOLAP approach
  - myOLAP Algebra
  - WeSt algorithm
- myOLAP prototype
- Conclusions and future works
OLAP Queries

- OLAP analyses are at the core of DWing systems since they promise managers to autonomously carry out complex queries in real-time.
  - They could require a non-negligible effort to find out the useful information

- Since their birth (around 1995) commercial OLAP systems have grown from several point of views:
  - Human Computer Interaction
  - Add-ons: dashboard and analytics
  - Supported data: spatial, semi-structured

- … but almost retain the same expressivity in terms of basic operators
  - Drill-down
  - Roll-up
  - …..

In the past the existence of a stable set of operators favored the spread of OLAP, but it is time to make a step forward

- Many directions are possible:
  - OLAP over heterogeneous schemata and data
    - Peer-to-peer DW [KSC+02]
  - OLAP with uncertainty
    - On measures and facts [BDJ07]
  - Semantically enriched OLAP
    - Complex type of aggregate operators [HSC04],[GR00]
    - Advanced classification/aggregation semantics [EZ06]
  - Personalized OLAP queries

- The common goal is to increase expressivity and to reduce the effort in describing what would be returned
The goal of personalization is to deliver information that is relevant to an individual or a group of individuals in the most appropriate format and layout.

This goal can be achieved with different approaches:

- **Query recommendation**: the system, based on the navigation path and on the user profile, suggests new queries in order to help the user navigating the cube [GMN09]

  *If a user in session A issued one or more queries similar to those in session B he will probably issue more queries similar to those in session B*

- **Personalized visualization**: the user specifies a set of visualization constraints that are used to determine a preferred visualization [BGM+05] according to a user profile

  *Select a visualization that includes at most 10 cells and that includes events concerning European sales rather than Asian ones*

- **Result ranking**: query results are organized in a total or partial order so that the user visualizes only the “most relevant” tuples.

  *I prefer hotels that cost less than 100€ and as close as possible to the beach*

- **Query contextualization**: the query is enhanced adding predicates that depend on the query context [JRT+08]

  *The marketing executive is mainly interested in data aggregated by year then by quarter in the context of analysis of sales, but he may also wish to see data by month when analyzing sold quantities of Toshiba products*
**Context and User Profile**

- **Context**: any information that can be used to characterize the situation when the query is submitted. Common types of context include:
  - Computing context (e.g., network connectivity, resources)
  - Environment context (e.g., noise levels, temperature)
  - Time
  - User context (e.g., profile, location, role)

- **User profile**: a set of non-conflicting, possibly ordered, personalization criteria that are specific to a given user

- User profile and context allow preference criteria to be inferred and relieve the user of manually specifying them at query time

- When the applicability of a personalization criteria depends on context and user profile we have a **context-aware preference system** [JRT+08], [SPV06]

---

**OLAP Query Personalization**

- The previous personalization approaches differ in several aspects:
  - **Formulation effort**: some approaches require the user to manually specify preference criteria for each query, while in others the best personalization criteria are inferred from the context and the user profile.
  - **Prescriptiveness**: some approaches use personalization criteria as hard constraints that are added to a query while in other as soft ones: tuples that satisfy as much preference criteria as possible are returned even if no tuples satisfies all the preferences
  - **Proactiveness**: distinguishes the approaches that propose new queries based on the navigation log and on the context (but that does not execute them), with respect to those that change the current query or post process its results before returning them to the user.
  - **Expressiveness**: personalization criteria have different expressivities and can be differently combined.
Generalities on Preference queries

- A major classification distinguishes between: quantitative and qualitative preferences

- **Quantitative**: are indirectly defined through a scoring function \( f(t) \) that associates a numerical score to each tuple \( t \). They determine a total ordering of tuples, the preferred ones can be retrieved through a top-k query [BP09], [XHC+06]
  - Have a limited expressivity
  - Defining a good scoring function is hard and could determine a subjective result

<table>
<thead>
<tr>
<th>Destination</th>
<th>Days to Start</th>
<th>Duration</th>
<th>Cost</th>
<th>( f(s,d,c) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ibiza</td>
<td>5</td>
<td>5</td>
<td>€ 1200</td>
<td>17.6</td>
</tr>
<tr>
<td>Paris</td>
<td>10</td>
<td>3</td>
<td>€ 500</td>
<td>4.3</td>
</tr>
<tr>
<td>Milan</td>
<td>10</td>
<td>5</td>
<td>€ 800</td>
<td>8.4</td>
</tr>
<tr>
<td>New York</td>
<td>20</td>
<td>9</td>
<td>€ 1500</td>
<td>7.3</td>
</tr>
<tr>
<td>Tokyo</td>
<td>30</td>
<td>7</td>
<td>€ 1000</td>
<td>5.2</td>
</tr>
<tr>
<td>Sidney</td>
<td>30</td>
<td>7</td>
<td>€ 1800</td>
<td>2.1</td>
</tr>
</tbody>
</table>

Travels

\[
f(s,d,c) = 20 \times \left( \frac{d}{s} \right) - 0.01 \times \left( \frac{c}{d} \right)
\]

\[
f(5,5,1200) = 20 \times \left( \frac{5}{5} \right) - 0.01 \times \left( \frac{1200}{5} \right) = 17.6
\]

- **Qualitative**: are directly expressed using preference relations \( \succ_P \)
  - Have a higher expressivity
  - Results are organized according to a strict partial order instead of a total order

- Preference relations can be specified using logical formulas

\[
P = \text{“I prefer trips that start shortly, with long duration and low cost”}
\]

\[
(s,d,c) \succ_P (s',d',c') \iff \begin{align*}
(c \leq 1000 \land c' > 1000) \lor \\
((c \leq 1000 \land c' \leq 1000) \lor (c > 1000 \land c' > 1000)) \land \\
((d > d' \land s \leq s') \lor (d > d' \land s < s'))
\end{align*}
\]

- A tuple \( t \) is preferred to (or dominates) a tuple \( t' \) according to \( \succ_P \) if \( t \succ_P t' \)
**Pareto optimality**

- Given a relation $r$ which are the best tuples according to a preference relations $P$?
- According to the pareto optimality criterion the tuples to be returned are those in $r$ for which no better alternative is available.

<table>
<thead>
<tr>
<th>Destination</th>
<th>Days to departure</th>
<th>Duration</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ibiza</td>
<td>5</td>
<td>5</td>
<td>1200</td>
</tr>
<tr>
<td>Paris</td>
<td>10</td>
<td>3</td>
<td>500</td>
</tr>
<tr>
<td>Milan</td>
<td>10</td>
<td>5</td>
<td>800</td>
</tr>
<tr>
<td>New York</td>
<td>20</td>
<td>9</td>
<td>1500</td>
</tr>
<tr>
<td>Tokyo</td>
<td>30</td>
<td>7</td>
<td>1000</td>
</tr>
<tr>
<td>Sidney</td>
<td>30</td>
<td>7</td>
<td>1800</td>
</tr>
</tbody>
</table>

**Preference composition**

- Preference relations are usually formulated through preference expressions that compose logical predicates on single attributes.

$I$ prefer trips that cost less than 1000 € $\quad P' = c \leq 1000 €$ and $c' > 1000 €$

$I$ prefer trips that start shortly $\quad P'' = s \leq s'$

- **Pareto composition $P' \otimes P''$:** $P'$ and $P''$ have the same relevance
  - "I prefer trips that cost less than 1000€ and those that start shortly"
  - If no tuples satisfy both the preference predicates I am equally interested in the tuples that satisfy either $P'$ or $P''$

- **Prioritization $P' \triangleright P'':** $P'$ is more relevant than $P''$
  - "I prefer trips that cost less than 1000€ and between those ones the trips that start shortly"
  - If no tuples satisfy both the preference predicates I am primarily interested in the tuples that satisfy $P'$
Preference in the OLAP context

Preference are particularly relevant in the OLAP context since:

- They enable users to focus on most interesting data: multidimensional database store huge amount of data while managers (and OLAP interfaces) can handle only a limited amount.

- Users do not exactly know what they are looking for and manually finding reasons behind a specific phenomenon may require several navigation steps.
  - Preferences based on soft constraints allow to specify a pattern describing the requested information
  - We can think at preference queries as a mining process, thus giving concreteness to the OLAM idea

- Preferences can be useful in the context of federated and heterogeneous DW since allow partial or approximate information to be retrieved

myOLAP approach

- Preferences are formulated by the user for each single query (or for a group of queries) using a visual interface or using an extended version of MDX language

- Preferences are non-prescriptive, i.e. they are soft constraints

- Expressiveness is specifically tailored for the OLAP context and allows to specify criteria on:
  - Measures
  - Dimensional attributes
  - Group-by sets

- Both pareto composition and prioritization are supported

- The approach is not proactive
myOLAP approach: an example

A decision maker may want to analyze high average incomes for 2009.

Since she is not sure about the key factors of this phenomenon she will adopt a trial-and-error approach that requires a large set of query to be formulated…

… alternatively she can formulate one single query annotated with a set of preferences:

```
SELECT {AvgIncome} ON COLUMNS,
    CROSSJOIN(DESCENDANTS([RESIDENCE].[All].City, SELF_AND_BEFORE),
    CROSSJOIN(DESCENDANTS([RACE].[RaceGroup],SELF_AND_BEFORE),
    [OCCUPATION].[Occ].Members)) ON ROWS
FROM [CENSUS] WHERE [TIME].[Year].2009
PREFERRING AvgIncome BETWEEN 500 AND 1000
AND RESIDENCE CONTAIN State
```
Specificities of the OLAP context

- OLAP domain is representative of an unexplored class of preference queries since:
  - Preferences can be expressed not only on attributes, that have categorical domains, but also on measures that have numerical domains
    - All the known approaches focus on either categorical or numerical data
  - Preferences can be formulated on schema (the aggregation level of data) rather than on data
    - No existing approaches handles extrinsic preference [Cho03]
  - The search space is dramatically large since includes, beside elemental facts, also the aggregated ones: the whole data cube

myOLAP algebra [GR08]

- Extends the work by Kiessling to the OLAP domain [Kie02]

  Base constructors on attributes

- POS(h.a,c): facts are preferred when:
  - Their group-by set includes h.a and the value for h.a is c
  - Their group-by set does not include h.a but the attribute value for h.b maps on c
  - POS(State,'Florida'): preferred facts are those concerning
    - Florida
    - The cities in Florida
    - The South-Est region of USA (where Florida is located)

- NEG(h.a,c) behaves symmetrically
myOLAP algebra

Base constructors on measures

- **BETWEEN**$((m,v_{low},v_{high}))$: a fact $f$ is preferred to $f'$ if:
  - $f \cdot m \in [v_{low},v_{high}]$ and $f' \cdot m \notin [v_{low},v_{high}]$ independently from their group-by-set.
  - $f \cdot m$ is closer to $[v_{low},v_{high}]$ with respect to $f' \cdot m$ independently from their group-by-set.

- **HIGHEST**(m): a fact $f$ is preferred when:
  - $f \cdot m$ is higher than in other facts independently from its group-by-set.

- **LOWEST**(m): behaves symmetrically.

Base constructors on hierarchies

- **CONTAIN**$(h,a)$: facts are preferred when:
  - Their group-by set includes $h.a$.
  - **CONTAIN**(Residence, State) means that facts aggregated by residence state are preferred.

- **NEAR**$(h, a_{fine}, a_{coarse})$: facts are preferred when:
  - Their group-by set along $h$ is between $a_{fine}, a_{coarse}$.

- **FINEST**(h): finer facts along $h$ are preferred to coarser ones.

- **COARSEST**(h) behaves symmetrically.
Computing the BMO

- Our approach answers preference query on a data cube according to the best match only model (BMO) in which all and only the facts not worse than any other facts are returned.

- State of the art of the algorithms
  - Approaches based on sorting: (e.g. SALSA[BCP08]) exploit tuples sorting in order to find out a stop point: none of the tuples behind such point can belong to the BMO and should not be accessed
    • Suitable for numerical attributes (measures)
    • Require presorting: impracticable for the whole data cube
    • Not enough selective when categorical attributes are involved
  - Approaches based on partitioning: (e.g. LBA [GKC+08]) partition the search space in S-classes (i.e. group of tuples that fulfill preferences in the same way) build a preference graph (BTG) between S-classes and access only nodes corresponding to undominated S-Classes
    • Suitable for categorical attributes and hierarchies
    • Unsuitable for numerical ones that would determine too many nodes

The WeSt algorithm [BGR10]

- Our idea is to get the best from both the previous approaches by creating a new type of partitioning graph whose nodes collapse several S-classes into one node whose processing is delayed
  - Collapsed S-classes are those concerning domination between measures (HIGHEST, LOWEST, BETWEEN), the corresponding nodes are called dotted

- We need a new type of domination called weak domination
  - Given two nodes $s_1$ and $s_2$, $s_1$ weakly dominates $s_2$ iff:
    • each class in $s_1$ dominates at least one S-Class in $s_2$ and is not dominated by any other S-class in $s_2$
    • Each S-class in $s_2$ is dominated by at least one S-class in $s_1$

$G(f).\text{RESIDENCE}=\text{State} \land f.\text{AvgInc} \in [500,1000]$

$G(f).\text{RESIDENCE} \neq \text{State} \land f.\text{AvgInc} \notin [500,1000]$

$G(f).\text{RESIDENCE}=\text{State} \land f.\text{AvgInc} \notin [500,1000]$

CONTAIN(RESIDENCE,State) $\otimes$ BETWEEN(AvgIncome,500,1000)
The WeSt algorithm

1. Access \( s_1 \) and return its facts if they exist
2. If at least one tuple is retrieved the algorithm terminate...
3. else \( s_2 \) and \( s_3 \) must be accessed
4. tuples in \( s_3 \) must be further compared each other in order to verify which ones are non-dominated
5. If \( s_2 \) is not empty \( s_4 \) must not be accessed and the algorithm terminate
6. else also \( s_4 \) must be accessed and its tuples must be compared each others and with the undominated tuples in \( s_3 \)

CONTAIN(RESIDENCE,State) \( \odot \) BETWEEN(AvgIncome,500,1000)
WeSt performances

- We created a benchmark for the CENSUS fact
  - Up to $2.5 \times 10^7$ events in the data cube
  - 50 queries with different combinations of base constructors

- WeSt always outperforms LBA since LBA does not directly exploit preference on hierarchies: tuples group by set is modeled as extra attributes

WeSt performances

- We created a benchmark for the CENSUS fact
  - Up to $2.5 \times 10^7$ events in the data cube
  - 50 queries with different combinations of base constructors

- Salsa outperforms WeSt when queries mainly includes preference on measures
- Salsa does not support prioritization
myOLAP in action

- We created a tool for handling OLAP preferences
  - It is based on Java technology
  - It builds on JPivot and Mondrian
  - It allows both graphical and textual query formulation
myOLAP in action
myOLAP in action
myOLAP in action

Conclusions and future works

- Personalization represents an interesting direction of research for increasing OLAP effectiveness and for reducing user efforts
  - No commercial solutions for OLAP currently implement any type of personalization features

- OLAP domain introduces a new class of preference queries that cannot be satisfactorily managed by existing approaches

- myOLAP approach represents a complete solution to OLAP preferences, but many extensions are possible:
  - Using the context to formulate the preference in order to reduce the formulation effort
  - Optimizing the execution of OLAP queries
  - Exploiting preferences for specifying preferred data in a federated DW
BIN: functional architecture

- A Business Intelligence Network is composed by a set of autonomous peers, one for each company, that expose BI functionalities described by ontologies owned by peers.

- Peers build up a P2P network...
  - ...defined by semantic mappings
  - ...characterized by sharing policies and different degrees of trust between peers.
The user formulates a BI query through his peer ontology. The query is sent on the network exploiting semantic paths defined through the mappings that connect the peer ontologies.

“I wish to know the number of persons affected by heart disease. The data should be grouped by sex, type of illness, and region of residence. If region is not available I prefer data grouped by city.”

“I wish to know the number of adults affected by cardiopathy. The data should be grouped by sex, category of illness, and city of residence.”

Each contacted peer locally answers the query and returns its results according to the preference expressed by the requesting peer. The results, even partial or approximate, are integrated and returned to the user based on his local ontology vocabulary and using a friendly interface.
References I


References II