Grouping and Aggregation in the Concept-Oriented Data Model

Alexandr Savinov
Fraunhofer Institute for Autonomous Intelligent Systems
Knowledge Discovery Team
Germany
savinov@conceptoriented.com
Outline

- Introduction
- Physical and Logical Structures
- Model Dimensionality
- Projection and De-projection
- Multidimensional Analysis
- Conclusions
Introduction
Concept-oriented paradigm

- **Duality:** any element is a collection of other elements and a combination of other elements, for example:
  - references vs. properties
  - entity modeling vs. identity modeling
- **Order:** order of elements determines most of syntactic and semantic properties
- **Representation and access (RA)** is the main concern.

![Diagram showing Concept-oriented paradigm, Concept-oriented model (COM), and Concept-oriented programming (COP)]
Physical and Logical

Physical structure

- At physical level an element of the model is a collection of other elements
- Physical structure is used for representation and access
- Physical structure is used to implement reference
- Physical structure is hierarchical where each element has only one parent
Physical and Logical
Logical structure

- Each element is a combination of other elements (by reference)
- Logical structure is used to represent data semantics (properties)
- Logical collection is a dual combination
- Each element has many parents and many children

Diagram:
- Customers
- Countries
- Germany
- France
- CompanyX
- Orders
- #23

Logical structure:
- customer
- date
- order
- part1
- part2

AND
OR
Physical and Logical
Two level model

- [Root] One root element is a physical collection of concepts,

- [Syntax] Each concept is
  - (i) a combination of other concepts called superconcepts (while this concept is a subconcept),
  - (ii) a physical collection of data items (or concept instances),

- [Semantics] Each data item is
  - (i) a combination of other data items called superitems (while this item is a subitem),
  - (ii) empty physical collection,
Physical and Logical
Two level model

- **[Special elements]**
  - Top and bottom concepts
  - Primitive concepts
  - Null item

- **[Cycles]** Cycles in subconcept-superconcept relation and subitem-superitem relation are not allowed,

- **[Syntactic constraints]** Each data item from a concept may combine only items from its superconcepts.
Model Dimensionality
Multidimensional space

- Superconcept is a domain of a dimension
- A common subconcept is a multidimensional space
- More levels can be added to the multidimensional space
Model Dimensionality
Hierarchical space

- It is one-dimensional space with many levels of details
- Subconcepts are alternative views on their common superconcept
Hierarchical multidimensional space

- Both structures are combined in one concept graph
- The concept graph possesses both multidimensional and hierarchical properties
Model Dimensionality

Dimensions

- **Dimension** is a named position of superconcept
- Superconcept is referred to as the domain
- Dimensions of higher rank consists of many (local) dimensions
- Dimension with the domain in a primitive concept is a *primitive dimension*
- The number of primitive dimensions is the model *primitive dimensionality*
Model Dimensionality

Inverse dimensions

- **Inverse dimension** has an opposite direction
- Inverse dimension identifies a subconcept
- Inverse dimensions are multi-valued (while dimensions are one-valued)
- The number of primitive dimensions is equal to the number of primitive inverse dimensions
- `{AuctionBids.auction.product.category}`
Projection and De-projection

Two retrieval operations

- Two ways to retrieve related items: projection and de-projection
- These two ways are supported by the model structure and correspond to moving up and down in the concept graph
- These two retrieval operations need only dimension names – no complex joins anymore
- These operations are analogous to the corresponding geometrical operations
Projection and De-projection

Projection

- Projection operator returns a set of superitems along some dimension
- Projection operator -> is followed by a dimension: 
\[ \text{OrderParts} \rightarrow \text{product} \rightarrow \text{category} \]
De-projection operator returns a set of subitems

De-projection operator \( \rightarrow \) is followed by an inverse dimension:
\[\text{Category} \rightarrow \{\text{product} \rightarrow \text{category}\}\]
Projection and De-projection

Access path

- Access path is a sequence of projections and de-projection where each next operator is applied to the result of the previous operator
- Category.getOrders = this->
  {OrderParts->product->category}->
  order;
- Category.getOrders = this->
  {OrderParts->product->category}->
  order->customer->country;
- Zigzag paths are possible
- Aggregation can be applied to sets of items
- Category.meanPrice = avg(
  this->getOrders->price );
Multidimensional Analysis

Multidimensional de-projection

- More than one bounding dimension
- Multidimensional de-projection returns a set of subitems referencing source items along all bounding dimensions:

One-dimensional de-projection

Multi-dimensional de-projection
Multidimensional Analysis
Steps of analysis

1. Choose dimension paths along which we want to view our data S
2. Choose the levels along these dimensions
3. Universe of discourse is the Cartesian product of the chosen levels
4. Each point from UoD is de-projected onto the target subconcept S
5. De-projection is aggregated using some property (measure)
**Multidimensional Analysis Example**

- Choose the target concept `OrderParts` and two dimensions leading to concepts `Countries` and `Categories`.

- De-project each pair of customer and product to `OrderParts`:
  
  \[
  \langle c, p \rangle \rightarrow \{ \text{OrderParts} \rightarrow \text{order} \rightarrow \text{customer}, \text{OrderParts} \rightarrow \text{product} \}
  \]

- Aggregate and return average price:
  
  ```latex
  \text{FORALL} (c \text{ Customers}, p \text{ Products}) \{ \\
  \text{tmp} = \langle c, p \rangle \rightarrow \{ \\
  \text{OrderParts} \rightarrow \text{order} \rightarrow \text{customer}, \text{OrderParts} \rightarrow \text{product} \} \\
  \text{RETURN} (c, p, \text{avg}(\text{tmp}.\text{price}));
  \}
  ```
Multidimensional Analysis
Change the level of details

- Choose other domains along dimension paths and apply the same query:

  $\text{FORALL}(c \text{ Countries}, \ p \text{ Categories}) \ {\ \}
  \begin{align*}
  & \text{tmp} = <c,p>->
  \\
  & \text{OrderParts}->\text{order}->\text{customer}->\text{country},
  \\
  & \text{OrderParts}->\text{product}->\text{category}
  \\
  \end{align*}

  $\text{RETURN}(c,p,\text{avg}(\text{tmp}));$

roll up

Drill down
Multidimensional Analysis

Example 2

- **FORALL** \{ d Dates, c Categories \} {
  \[ \text{tmp = this->}{ \]
  Auctions.date,
  Auctions.product.category
  \} \]
  \[ \text{RETURN} (d,c, \text{avg(tmp->meanPrice)} ) \]
}
Conclusions

- Features:
  - Canonical semantics
  - Logical navigation via access paths, dimensions and inverse dimensions
  - Multidimensional aggregation and analysis
  - Constraint propagation and inference (not described in this presentation)

- Advantages:
  - Grouping and aggregation is integral part of the model
  - Combination in one model hierarchical and multidimensional properties
  - Formal syntax and semantics
  - Simple query language -- no joins anymore