Foundations of accessing data through ontologies

Maria Keet
email: mkeet@cs.uct.ac.za
home: http://www.meteck.org

Department of Computer Science
University of Cape Town, South Africa

ISAO 2023, 10-14 July 2023, Sherbrooke, Canada
Outline

1. Approaches

2. Digging deeper to compare

3. Knowledge mapping data with OBDA: A system
   - The mapping layer
   - Query answering

4. A ‘simple’ example
Digging deeper to compare Knowledge mapping data with OBDA: A system
A 'simple' example
Approaches

Digging deeper to compare

Knowledge mapping data with OBDA: A system

A ‘simple’ example
Outline

1. Approaches

2. Digging deeper to compare

3. Knowledge mapping data with OBDA: A system
   - The mapping layer
   - Query answering

4. A ‘simple’ example
Why bother?

- Put the knowledge to use
- Add meaning to data
Why bother?

- Put the knowledge to use
  ⇒ For what use?

- Add meaning to data
  ⇒ What does that offer?
Why bother?

- Put the knowledge to use
  ⇒ For what use?
  e.g.: data integration across database, RDBMS back-end OO frontend, ...
- Add meaning to data
  ⇒ What does that offer?
  e.g.: easier and faster data access, infer more cf plain queries
An example

- Subsumption & equivalence for integration
An example

- Subsumption & equivalence for integration
- Data, say $\text{PersCustomer}(\text{Ndumi})$

*Customer*  

Integrated system

- **PersCustomer**  
  Purchased data source 3  
  $<$Ndumi, ... $>$

- **IndivCustomer**  
  Company 1, data source 2

- **OrgCustomer**  
  Company 1, system 1

$\text{etc}....$
An example

- Subsumption & equivalence for integration
- Data, say PersCustomer(Ndumi)
- Query: “retrieve all customers”
An example

- Subsumption & equivalence for integration
- Data, say PersCustomer(Ndumi)
- Query: “retrieve all customers”
- Plain DB answer: {}
An example

- Subsumption & equivalence for integration
- Data, say PersCustomer(Ndumi)
- Query: “retrieve all customers”
- Plain DB answer: {}

⇒ Ontology-enhanced DB system: \{Ndumi\}
Connecting the knowledge to the data

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Conceptual model shows what is stored in that particular application.

Implementation the actual information system that stores and manipulates the data.
Connecting the knowledge to the data

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation the actual information system that stores and manipulates the data.

Database

C++ application
Approaches

Digging deeper to compare Knowledge mapping data with OBDA: A system

A ‘simple’ example

Knowledge-to-Data Pipeline options

“Knowledge with data”

“Knowledge mapping data”

“Data transformation knowledge”

“Data with Knowledge”

Al-oriented

DB-oriented

Knowledge base \( K \) with instances \( D \)

Extended database with \( K+D \)

\( K+D \) stored as data

Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.
“Knowledge mapping data”: OBDA system EPnet [Calvanese et al. (2016)]

The federation engine operates at the physical or relational schema layer. Typically relational databases and RDF triple stores.

Linking elements from the ontology to queries over the data source(s).

Ontology or logic-based conceptual data model

Mappings

The federation engine operates at the physical or relational schema layer.

Data sources

Typically relational databases and RDF triple stores.
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation the actual information system that stores and manipulates the data.
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.
“Knowledge mapping data” : OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.
“Knowledge mapping data”: OBDA

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.
Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Mapping layer links each entity to a query over the data source(s).

Implementation the actual information system that stores and manipulates the data.

End-user query “give me all red flowers” just click relevant elements in the diagram.
## “Knowledge mapping data”: OBDA

- **OBDA with Ontop**
  [Calvanese et al. (2017)](Calvanese, Cogrel, Komla-Ebri, Kontchakov, Lanti, Rezk, Rodriguez-Muro, and Xiao) now more elaborate and more robust

- **More recent case studies: Statoil, EPnet**
  [Calvanese et al. (2016)](Calvanese, Liuzzo, Mosca, Remesal, Rezk, and Rull) (early attempts: e.g., Genomics data with horizontal gene transfer
  [Calvanese et al. (2010)](Calvanese, Keet, Nutt, Rodríguez-Muro, and Stefanoni)
“Knowledge mapping data”: OBDA

- OBDA with Ontop

- More recent case studies: Statoil, EPnet
  [Calvanese et al.(2016)Calvanese, Liuzzo, Mosca, Remesal, Rezk, and Rull]
  (early attempts: e.g., Genomics data with horizontal gene transfer
  [Calvanese et al.(2010)Calvanese, Keet, Nutt, Rodríguez-Muro, and Stefanoni]

- Downsides
  - The mapping layer: cumbersome construction and maintenance
  - Low expressiveness for ontology language
  - Limitations on types of queries
An OBDA system with Ontop

[Calvanese et al. (2017)]
“Data-transformation-knowledge” example: KnowID

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Queries for decision-making formulate queries using the knowledge graph to retrieve data.

Implementation
the actual information system that stores and manipulates the data.

Database
C++ application
“Data-transformation-knowledge” example: KnowID

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers

Implementation: the actual information system that stores and manipulates the data.
“Data-transformation-knowledge” example: KnowID

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers.

Implementation the actual information system that stores and manipulates the data.
“Data-transformation-knowledge” example: KnowID

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

Transformation via abstract relational model with additional virtual identifiers.

Implementation: the actual information system that stores and manipulates the data.
Knowledge-driven Information and Data access (KnowID)

Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.
Fillottrani, P.R., Jamieson, S., Keet, C.M. Connecting knowledge to data through transformations in KnowID: system description. Künstliche Intelligenz, 2020, 2020, 34, 373-379.
Knowledge-driven Information and Data access (KnowID)

Fillottrani, P.R., Keet, C.M. KnowID: An architecture for efficient Knowledge-driven Information and Data access. Data Intelligence, 2020, 2(4): 487-512.
Fillottrani, P.R., Jamieson, S., Keet, C.M. Connecting knowledge to data through transformations in KnowID: system description. Künstliche Intelligenz, 2020, 2020, 34, 373-379.
Knowledge-driven Information and Data access (KnowID)

- There’s more on the ‘knowledge and information management’ module [Fillottrani and Keet(2020)]:
  - Swap between EER, UML, ORM  
    [Keet and Fillottrani(2015), Fillottrani and Keet(2014), 
    Braun et al.(2023)]Braun, Fillottrani, and Keet
  - DL (OWL) with reasoner at the back-end
  - Tool: crowd 2.0 (beta)  
    http://crowd.fi.uncoma.edu.ar:3335/  
    [Braun et al.(2020)]Braun, Gimenez, Cecchi, and Fillottrani
  - More in the pipeline, such as integrating NOMSA for summarisation and modularisation of ontologies
- Querying with SQLP: SQLP requires less time for understanding and authoring queries, with no loss in accuracy  
  [Ma et al.(2018)]Ma, Keet, Oldford, Toman, and Weddell
- Data Completion TBD
Outline

1. Approaches

2. Digging deeper to compare

3. Knowledge mapping data with OBDA: A system
   - The mapping layer
   - Query answering

4. A ‘simple’ example
Three key factors for choosing an approach

- **Stability of the data**: do they (i) continuously change a lot throughout the database or (ii) intermittently, rarely, or append-only?

- **Stability of the schemas**: do they (i) remain unchanged once the system is set-up or (ii) will they have to change based on changing business needs and usage optimisations?

- **Type of queries posed over the data**: are they (i) at most (unions of) conjunctive queries (UCQs) or (ii) also other types of SQL queries (with or without paths)?
Queries with OBDA models vs FO-inspired ontologies

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.
Queries with OBDA models vs FO-inspired ontologies

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

End-user query
“give me all red flowers”
just click relevant elements in the diagram.
Queries with OBDA models vs FO-inspired ontologies

Ontology (or controlled vocab, kg) provides the common vocabulary and constraints that hold across the applications.

End-user query “give me all red flowers” “just” process relevant elements in the diagram:

- Flower
- Height
- Colour
- Material entity
- Independent continuant
- SDC
- Quality
- Continuant

End-user query: “give me all red flowers”
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(\mathcal{I}\): then \(Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})\)
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(\mathcal{I}\): then \(Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})\)

   ... this is probably what you assume to be happening
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(\mathcal{I}\): then \(Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})\)
   
   ... this is probably what you assume to be happening

2. \((\mathcal{T}, \mathcal{A})\) have many models, say \(\mathcal{I}_j \ (j \in J)\):

   Option I: restrict \(\mathcal{T}\) to make it feasible: \((\text{simple})\) Horn theories
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(\mathcal{I}\): then \(Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})\)
   
   \[Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I}) \]  
   
   … this is probably what you assume to be happening

2. \((\mathcal{T}, \mathcal{A})\) have many models, say \(\mathcal{I}_j \ (j \in J)\):
   
   Option I: restrict \(\mathcal{T}\) to make it feasible: (simple) Horn theories
   
   \[\Rightarrow\] canonical models (and small ones!)
   
   \[\Rightarrow\] works well only for positive queries

slides 18, 19, 22 with thanks to David Toman
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(\mathcal{I}\): then \(Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})\)
   
   ... this is probably what you assume to be happening

2. \((\mathcal{T}, \mathcal{A})\) have many models, say \(\mathcal{I}_j\) \((j \in J)\):
   
   **Option I**: restrict \(\mathcal{T}\) to make it feasible: (simple) Horn theories
   
   ⇒ canonical models (and small ones!)
   
   ⇒ works well only for positive queries

   **Option II**: restrict \(Q\) to make it feasible: those for which it doesn’t matter which model is used

slides 18, 19, 22 with thanks to David Toman
How to answer queries efficiently?

1. \((\mathcal{T}, \mathcal{A})\) have exactly one model \(I\): then 
\[ Q(\mathcal{A}, \mathcal{T}) = Q(I) \]
... this is probably what you assume to be happening

2. \((\mathcal{T}, \mathcal{A})\) have many models, say \(I_j\) \((j \in J)\):
   - **Option I**: restrict \(\mathcal{T}\) to make it feasible: *(simple) Horn theories*
     \[ \Rightarrow \text{canonical models (and small ones!)} \]
     \[ \Rightarrow \text{works well only for positive queries} \]
   - **Option II**: restrict \(Q\) to make it feasible: those for which it doesn’t matter which model is used
     \[ \Rightarrow \text{e.g., safe queries in Codd’s relational model} \]

slides 18, 19, 22 with thanks to David Toman
Approaches
Digging deeper to compare
Knowledge mapping data with OBDA: A system
A ‘simple’ example

Option 1

\[Q\] 

\[\mathcal{A}\] \rightarrow \text{complete} \rightarrow \mathcal{A}' \rightarrow \text{evaluate} \rightarrow \mathcal{A}''

\[\text{rewrite}\]

[Calvanese et al.]

v1.0: rewrite; incorporate $T$ into $Q$; complete: an identity ($A' = A$).

v2.0: rewrite; rewrite independently of $T \cup A$; complete: incorporate $T$ into $A$. . .

[Lutz et al.]
Option I

### v1.0:
- **rewrite**: incorporate \( \mathcal{T} \) into \( Q \),
- **complete**: an identity \( \mathcal{A}' = \mathcal{A} \)

...[Calvanese et al.]

### v2.0:
- **rewrite**: rewrite independently of \( \mathcal{T} \cup \mathcal{A} \),
- **complete**: incorporate \( \mathcal{T} \) into \( \mathcal{A} \)

...[Lutz et al.]
An example – Revisited with query rewriting
An example – Revisited with query rewriting
An example – Revisited with query rewriting
An example – Revisited with query rewriting
An example – Revisited with data completion
An example – Revisited with data completion
An example – Revisited with data completion
An example – Revisited with data completion

Integrated system

PersCustomer
Purchased data source 3
<Ndumi, … >

IndivCustomer
<Ndumi, … >
Company 1, data source 2

OrgCustomer
Company 1, system 1

Customer
1

etc….
One more note on rewriting vs data completion

<table>
<thead>
<tr>
<th>V1.0 (query rewriting)</th>
<th>V2.0 (data completion)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queries</td>
<td>rewriting is exponential in $</td>
</tr>
<tr>
<td></td>
<td>applies to original data</td>
</tr>
</tbody>
</table>

we always can devise a way where one system wins over the other
# Key distinguishing features of varying computational cost

<table>
<thead>
<tr>
<th>Feature</th>
<th>$K\odot D$</th>
<th>$K \Leftrightarrow D$</th>
<th>$D \bowtie K$</th>
<th>$D\odot K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>World</td>
<td>OWA</td>
<td>OWA+CWA</td>
<td>CWA</td>
<td>CWA</td>
</tr>
<tr>
<td>Language for $\mathcal{K}$</td>
<td>OWL</td>
<td>OWL</td>
<td>relational, DL</td>
<td>relational</td>
</tr>
<tr>
<td>Language for $\mathcal{D}$</td>
<td>OWL</td>
<td>relational</td>
<td>relational</td>
<td>relational</td>
</tr>
<tr>
<td>Query language</td>
<td>SPARQL</td>
<td>SPARQL + SQL (fragment)</td>
<td>SQLP</td>
<td>SQL</td>
</tr>
<tr>
<td>Automated reasoning</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>depends on system</td>
</tr>
<tr>
<td>Reasoning w.r.t. data</td>
<td>no separate approach</td>
<td>query rewriting</td>
<td>data completion</td>
<td>data completion</td>
</tr>
<tr>
<td>Mapping layer</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Transformations</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Entity recasting</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Syntactic sugar</td>
<td>available</td>
<td>available</td>
<td>possible</td>
<td>possible</td>
</tr>
</tbody>
</table>
Outline

1. Approaches
2. Digging deeper to compare
3. Knowledge mapping data with OBDA: A system
   - The mapping layer
   - Query answering
4. A ‘simple’ example
Summary

1. Approaches

2. Digging deeper to compare

3. Knowledge mapping data with OBDA: A system
   - The mapping layer
   - Query answering

4. A ‘simple’ example
References I

G. Braun, P. R. Fillottrani, and C. M. Keet.
A framework for interoperability between models with hybrid tools.

Germán Braun, Christian Gimenez, Laura Cecchi, and Pablo Fillottrani.
crowd: A Visual Tool for Involving Stakeholders into Ontology Engineering Tasks.
*Künstliche Intelligenz, 2020.*

D. Calvanese, P. Liuzzo, A. Mosca, J. Remesal, M. Rezk, and G. Rull.
Ontology-based data integration in epnet: Production and distribution of food during the roman empire.
*Engineering Applications of Artificial Intelligence, 51:212–229, 2016.*

Diego Calvanese, C. Maria Keet, Werner Nutt, Mariano Rodríguez-Muro, and Giorgio Stefanoni.
Web-based graphical querying of databases through an ontology: the WONDER system.

Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao.
Ontop: Answering SPARQL queries over relational databases.

P. R. Fillottrani and C. M. Keet.
KnowID: An architecture for efficient knowledge-driven information and data access.
References II

Pablo Rubén Fillottrani and C. María Keet.
Conceptual model interoperability: a metamodel-driven approach.
August 18-20, 2014, Prague, Czech Republic.

S. Garcia-Vallvé, E. Guzman, M.A. Montero, and A. Romeu.
HGT-DB: a database of putative horizontally transferred genes in prokaryotic complete genomes.

C. María Keet and Pablo Rubón Fillottrani.
An ontology-driven unifying metamodel of UML Class Diagrams, EER, and ORM2.

Weicong Ma, C. Maria Keet, Wayne Oldford, David Toman, and Grant Weddell.
The utility of the abstract relational model and attribute paths in sql.
12-16 Nov. 2018, Nancy, France.
Thank you!

Questions?

- My textbook on ontology engineering (aimed at computer scientists)
- Also available in paperback: