The Many Uses of Rules in Ontology-Based Data Access

(Long version: cs.unb.ca/~boley/talks/RulesOBDA.pdf)

Harold Boley
University of New Brunswick
Faculty of Computer Science
Fredericton, NB, Canada

Department of Informatics Colloquium, University of Zurich, Switzerland, April 10, 2014
11th Annual CS Research Exposition, UNB Fredericton, NB, May 2, 2014
Seminar, IBM Research, Zurich, Switzerland, June 23, 2014
FE Colloquium, WSL Birmensdorf, Switzerland, June 30, 2014
Kolloquium Informatik / Semantic Web Meetup, Freie Universität Berlin, Germany, July 4, 2014
What is **Knowledge-Based Data Access**?

- Founded on *(logical) knowledge* representation for:
  - **Ontology-Based Data Access** (OBDA), e.g. data *integration/federation* and query optimization
  - **Rule-Based Data Access** (RBDA), e.g. [Datalog](https://en.wikipedia.org/wiki/Datalog) / *deductive databases* and query answering
- **Knowledge Base** as generalized *global* schema for data in *local* (e.g., relational or graph) DBs
- KB module amplifies data storage & query execution of *distributed, heterogeneous* (No)SQL DBs
- Provides **multi-purpose** knowledge level for data
Preview: Unified Architecture

Global schema → Rewriting → KB → Materialization → DB’ → DB → Folding Unfolding → Mappings → Q’ → Q_1' → Q_2' → ... → Q_n' → Local schemas

Q → DB → DB’ → Materialization → DB
Why Knowledge-Based Data Access?

- Domain knowledge utilized to deal with data torrent
  - Domain experts conceptually *fold* data / *unfold* queries via **Mappings** defined with IT (SQL, SPARQL, ...) experts
  - User concepts are captured in **Knowledge Base** for *domain-enriched* database materialization / querying **without** IT experts
- **Engines** use KB to deduce answers implicit in DBs
- **Analytics** enabled by queries exploring hypotheses
- KB as major organizational resource also for, e.g.:
  - Data validation (consistency, completeness, ...)
  - Schema-level query answering (even without DBs)
KB Contains Formal Knowledge as Ontologies and/or Rules

FormalKnowledge

OntologyKnowledge \quad \text{RuleKnowledge}

TaxonomyKnowledge \quad \text{FactKnowledge/Data}
Knowledge-Based Data Access, Using Strategy: 3-Dimensional KBDA_s

Knowledge

Rule

Ontology

Integration < Querying < Management

Access

Strategy

mediator

warehouse

bidirectional
RBDA Realizes Uniform KBDA – 1 of 3: Queries as Rules

1. a) A conjunctive *query* is a special Datalog *rule* whose body can be *rewritten* (see 2.) and *unfolded* (see 3.), and whose head instantiates the distinguished answer variables of the body

b) KBDA ontologies beyond RDF Schema (*RDFS*) often permit *Boolean* conjunctive queries corresponding to *integrity rules*

2. ...

3. ...
RBDA Realizes Uniform KBDA – 2 of 3: KBs as Rules

1. ...

2. KBDA *KB* supports, e.g., query *rewriting* through global-schema-level reasoning, including with RDFS *taxonomies* or Datalog *rule* axioms, and DL-Lite (*OWL 2 QL*) or *(head-)*existential *rules*; KBDA *rules* also permit Description Logic Programs (*OWL 2 RL*), *Datalog*\(^{±}\), and *Disjunctive Datalog*. [Semantics of *ontology languages* customizable for expressivity and efficiency requirements by adding/deleting *rules* (*SPIN*)]

3. ...
RBDA Realizes Uniform KBDA – 3 of 3: Mappings as Rules

1. ...

2. ...

3. KBDA data integration is centered on Global-As-View (GAV) *mappings*, which are Datalog *rules* for, e.g., *unfolding* each global head predicate to (a join, i.e. conjunction, of) local body predicates.
RBDA Realizes Uniform KBDA — All of: Queries, KBs, and Mappings as Rules

1. a) A conjunctive query is a special Datalog rule whose body can be rewritten (see 2.) and unfolded (see 3.), and whose head instantiates the distinguished answer variables of the body
   
   b) KBDA ontologies beyond RDF Schema (RDFS) often permit Boolean conjunctive queries corresponding to integrity rules

2. KBDA KB supports, e.g., query rewriting through global-schema-level reasoning, including with RDFS taxonomies or Datalog rule axioms, and DL-Lite (OWL 2 QL) or (head-)existential rules; KBDA rules also permit Description Logic Programs (OWL 2 RL), Datalog\textsuperscript{\pm}, and Disjunctive Datalog.

   [Semantics of ontology languages customizable for expressivity and efficiency requirements by adding/deleting rules (SPIN)]

3. KBDA data integration is centered on Global-As-View (GAV) mappings, which are Datalog rules for, e.g., unfolding each global head predicate to (a join, i.e. conjunction, of) local body predicates
Example: Forest/Orchard Knowledge
Subsumption axioms (in higher-order rule syntax):
EntityContainingAtLeastOneTree ← Forest.
EntityContainingAtLeastOneTree ← Orchard.
Forest ← Woodland.

Root of taxonomy tree of tree-containing entities to see the forest for the trees

“←” is taxonomy-style ‘subsumes’ infix
**EntityWithTree KB: Named Root Class (2)**

*Subsumption axioms (in higher-order rule syntax):*

EntityContainingAtLeastOneTree :- Forest.
EntityContainingAtLeastOneTree :- Orchard.
Forest :- Woodland.

```
“:-” is rule-style ‘if’ infix
```

Root of taxonomy tree of tree-containing entities to see the forest for the trees
**EntityWithTree KB: Constructed Root Class**

*Subsumption axioms (in higher-order rule syntax):*

- \( \exists \text{contains.Tree} : - \text{Forest.} \)
- \( \exists \text{contains.Tree} : - \text{Orchard.} \)

Forest :- Woodland.

\( \exists \text{contains.Tree} \)

Entities each having a `contains` property with at least one value in class `Tree`

Cf. ontology-style (description logic) axioms:

- \( \exists \text{contains.Tree} \equiv \text{Forest} \)
- \( \exists \text{contains.Tree} \equiv \text{Orchard} \)
Three Dimensions of KBDA$_s$: R,Q,m

Rule -Based Data Querying

RBDQ mediator
Query Rewriting and Unfolding

- Mediator strategy uses:
  - KB to *rewrite* $Q$ to $Q'$ and Mappings *(rules)* to *unfold* $Q'$ to $Q_i''$
  - KB can be *ontology*, e.g. in OWL 2 QL (DL-Lite), or *(rules)*
  - Abstract *(relational/graph/...)* queries $Q_i''$ -grounded *(to SQL/SPARQL/...)* for $DB_i$
  - Each *(relational/graph/...)* database $DB_i$ left as original; answers at ♦

(a) Two ‘backward’ transformations

```
Q    Rewriting
    KB
    Unfolding
    Mappings
    Q''
    DB1
    DB2
    DBn
```

```
(a) Two ‘backward’ transformations
```

```
Q    Rewriting
    KB
    Unfolding
    Mappings
    Q''
    DB1
    DB2
    DBn
```

```
(a) Two ‘backward’ transformations
```

```
Q    Rewriting
    KB
    Unfolding
    Mappings
    Q''
    DB1
    DB2
    DBn
```

```
(a) Two ‘backward’ transformations
```

```
Q    Rewriting
    KB
    Unfolding
    Mappings
    Q''
    DB1
    DB2
    DBn
```

```
(a) Two ‘backward’ transformations
```
Query Rewriting and Unfolding

- Mediator strategy uses:
  - KB to rewrite $Q$ to $Q'$ and Mappings (rules) to unfold $Q'$ to $Q_i$''
  - KB can be ontology, e.g. in OWL 2 QL (DL-Lite), or rules
  - Abstract (relational/graph/...) queries $Q_i$” ♦-grounded (to SQL/SPARQL/...) for $DB_i$
  - Each (relational/graph/...) database $DB_i$ left as original; answers at ♦

![Diagram](image)
Query Rewriting Use of KB

- In Information Retrieval: Query expansion
  - With increased recall
  - Without loss of precision

- From Logic Programming (for Horn expressivity): Can use resolution method for KB-enrichment of a given (conjunctive) query with expanded (conjunctive) queries so that, for any DB, the answers to the enriched queries no longer using the KB are the same as the answers to the original query using the KB
Ontology-to-Rule Clausification of EntityWithTree KB Omitting Orchard

*Description Logic subsumptions (as higher-order rules):*

\[ \exists x \text{contains.Tree} : - \text{Forest}. \]
\[ \text{Forest} : - \text{Woodland}. \]

*Horn Logic rules (the first with conjunctive head):*

\[ (\text{contains}(\text{s}(\text{x})) \land \text{Tree}(\text{s}(\text{x}))) : - \text{Forest}(\text{x}). \]
\[ \text{Forest}(\text{x}) : - \text{Woodland}(\text{x}). \]

*Horn Logic rules (the first head split into two conjuncts):*

\[ \text{contains}(\text{x} \text{s}(\text{x})) : - \text{Forest}(\text{x}). \]
\[ \text{Tree}(\text{s}(\text{x})) : - \text{Forest}(\text{x}). \]
\[ \text{Forest}(\text{x}) : - \text{Woodland}(\text{x}). \]
KB Rules Perform Rewriting of Given Query

**KB with Horn rules (from above) for rewriting of query rules:**
- contains(?x s(?x)) :- Forest(?x).
- Tree(s(?x)) :- Forest(?x).
- Forest(?x) :- Woodland(?x).

**Rewriting Datalog query rule to obtain extra query rules:**

Given query rule: \( q(?z) \) :- contains(?z ?y) \( \land \) Tree(?y).

Expansion query rules:
- \( q(?z) :- Forest(?z) \land Tree(s(?z)) \)
- \( q(?z) :- \text{contains}(?z s(?x)) \land \text{Forest}(?x) \)
- \( q(?z) :- \text{Forest}(?z) \land \text{Forest}(?z) \)
- \( q(?z) :- \text{Forest}(?z) \land Woodland(?z) \).

**Q:** Given

**Q′:** Given \( \cup \) Expansion
Query Unfolding Use of Mappings to Original Database Sources

- Datalog rules **bridging** between:
  - KB
  - Distributed DBs
- Use **partial deduction**-like unfolding (and simplification) of (conjunctive) KB queries to (conjunctive) abstract DB queries
  - Abstract relational queries grounded to SQL, abstract graph queries grounded to SPARQL, etc.
  - Lower-level optimization and execution by SQL, SPARQL, etc. engines
- Generated queries distributed over multiple DBs as indicated by “source.” name prefixes
Sample Mapping Rules to Three Local Data Sources

Map KB predicates to locDB/regionDB tables for geo data:
contains(?x ?y) :- locDB.cnt(?x ?kind ?y).
contains(?x ?y) :- regionDB.sub(?x ?r) ∧ locDB.cnt(?r ?kind ?y).

Map KB predicates to locDB/ecoDB tables for forestry data:
Tree(?t) :- locDB.cnt(?plot "tree" ?t).
Tree(?t) :- ecoDB.Plant(?plot "tree" ?size ?t).
Forest(?x) :- ecoDB.Habitat(?plot "forest" ?size ?x).
Woodland(?x) :- ecoDB.Habitat(?plot "wood" ?size ?x).
Mapping Rules Perform Unfolding of Rewritten Queries

Union of conjunctive queries as Datalog rules (rewritten):

\[ q(z) : - \text{contains}(z, y) \land \text{Tree}(y). \]
\[ q(z) : - \text{Forest}(z). \]
\[ q(z) : - \text{Woodland}(z). \]

Unfolding above queries via mappings from previous slide:

\[ q(z) : - \text{locDB.cnt}(z, \text{kind}, y) \land \text{locDB.cnt}(\text{plot}, \text{tree}, y). \]
\[ q(z) : - \text{locDB.cnt}(z, \text{tree}, y). \]
\[ q(z) : - \text{locDB.cnt}(z, \text{kind}, y) \land \text{ecoDB.Plant}(\text{plot}, \text{tree}, \text{size}, y). \]
\[ q(z) : - \text{regionDB.sub}(z, r) \land \text{locDB.cnt}(r, \text{kind}, y) \land \text{locDB.cnt}(\text{plot}, \text{tree}, y). \]
\[ q(z) : - \text{regionDB.sub}(z, r) \land \text{locDB.cnt}(r, \text{tree}, y). \]
\[ q(z) : - \text{regionDB.sub}(z, r) \land \text{locDB.cnt}(r, \text{kind}, y) \land \text{ecoDB.Plant}(\text{plot}, \text{tree}, \text{size}, y). \]
\[ q(z) : - \text{ecoDB.Habitat}(\text{plot}, \text{forest}, \text{size}, z). \]
\[ q(z) : - \text{ecoDB.Habitat}(\text{plot}, \text{wood}, \text{size}, z). \]

\( Q_i'': \) Bold-faced (1 ≤ i ≤ 3)
Three Dimensions of $\text{KBDA}_s$: $R, Q, w$
Database Materialization after Folding

- Warehouse strategy uses:
  Mappings (same rules as for unfolding) to fold $DB_i$ to $DB$ and $KB$ to materialize Database $DB$ to $DB'$
- $KB$ can be ontology, e.g. in OWL 2 RL (DLP), or rules
- Query is left as original; answers at solid triangular arrow head

(b)
Database Materialization after Folding

- Warehouse strategy uses:
  - Mappings (same rules as for unfolding) to fold $DB_i$ to DB and KB to materialize Database DB to DB'
- KB can be ontology, e.g. in OWL 2 RL (DLP), or rules
- Query is left as original; answers at solid triangular arrow head
Database Folding Uses Mappings to Translate and Integrate Databases

- Datalog rules bridging between $n$ local $DB_i$ and 1 global $DB_1$
- Usually simpler than under mediator strategy
  - Materialization often done for higher-level $DB_i$, e.g. with Prolog facts as relational $DB_i$ or RDF triples as graph $DB_i$
  - Even if folding needs no facts-to-table or triples-to-graph translation, it still needs to do $n$-to-$1$ integration
- For $n=1$ and higher-level $DB_1$, no folding is needed at all (as assumed here)
Database Materialization Use of KB

- In Database Systems: **Chase procedure**
- From Logic Programming (for Horn expressivity): Can use **bottom-up/forward-chaining method** and KB for enrichment of a given set of facts (DB) with derived facts so that, for any query, the answers against the enriched DB’ no longer using the KB are the same as the answers against the original DB using the KB
EntityWithTree (EWT) KB

Subsumptions (as higher-order rules):
∃contains.Tree :- Forest.
Forest :- Woodland.

∃contains.Tree

Forest

Woodland
EWT DB

Data (as higher-order facts):

\[ \exists \text{contains.Tree}(e). \]
\[ \text{Forest}(f). \]
\[ \text{Woodland}(w). \]

\[ \exists \text{contains.Tree} \]

Forest

\[ f \]

Woodland

\[ w \]
EWT KB and DB: ‘Populated’ KB

Subsumptions and Data (as higher-order rules and facts):

 existential contains.Tree :- Forest.
Forest :- Woodland.

 existential contains.Tree(e).
Forest(f).
Woodland(w).
Applying All KB Rules to All DB Facts (1)

Subsumptions and Data (as higher-order rules and facts):

\( \exists \)contains.Tree :- Forest.
Forest :- Woodland.

\( \exists \)contains.Tree(e).
\( \exists \)contains.Tree(f).
Forest(f).
Woodland(w).

\( \exists \)contains.Tree

Forest

Woodland

w
Applying All KB Rules to All DB Facts (2)

*Subsumptions and Data (as higher-order rules and facts):*

\[ \exists \text{contains.Tree} \vdash \text{Forest}. \]
\[ \text{Forest} \vdash \text{Woodland}. \]
\[ \exists \text{contains.Tree}(e). \]
\[ \exists \text{contains.Tree}(f). \]
\[ \exists \text{contains.Tree}(w). \]
\[ \text{Forest}(f). \]
\[ \text{Forest}(w). \]
\[ \text{Woodland}(w). \]
Fixpoint with No New Rule-Derived Facts

*Subsumptions and Data (as higher-order rules and facts):*

\[
\exists \text{contains.Tree} :\text{Forest.}
\]

\[
\text{Forest} :\text{- Woodland.}
\]

\[
\exists \text{contains.Tree}(e).
\]

\[
\exists \text{contains.Tree}(f).
\]

\[
\exists \text{contains.Tree}(w).
\]

\[
\text{Forest}(f).
\]

\[
\text{Forest}(w).
\]

\[
\text{Woodland}(w).
\]
Querying by Lookup in Materialized DB’

Datalog rules for (conjunctive) queries, with answers:

\[
\begin{align*}
q_1(?z) & : - \text{Woodland}(?z). & ?z = w \\
q_2(?z) & : - \text{Forest}(?z). & ?z = f, w \\
q_3(?z) & : \exists \text{contains.Tree}(?z). & ?z = e, f, w \\
q_4(?z) & : \text{contains}(?z ?y) \land \text{Tree}(?y). & ?z = e, f, w
\end{align*}
\]

Separate conjunctive queries qi: After materialization, no need for query rewriting with the same KB.
Three Dimensions of KBDA<sub>s</sub>

Knowledge-Based Data Access Strategy
Unified Architecture

- Combines strategies (a)-(c) of earlier slides
- Meets the needs of ∆Forest case study
Unified Architecture

- Combines strategies (a)-(c) of earlier slides
- Meets the needs of ΔForest case study

![Diagram showing the unified architecture process](image)
Standard Language for $\text{KBDA}_s$ Rules

- Will permit (Web-based) interchange of at least the three kinds of rules in $\text{KBDA}_s$:
  - Sharing and Reuse of queries (+ integrity constraints), KBs, and mappings can save repeated work

- Language options include
  - ISO: Prolog (extra-logical features), Common Logic 2 (‘wild-west’ syntax)
  - W3C: OWL 2 RL (DLP: weak), RIF-BLD (Horn: no head-$\exists$)
  - OMG: PRR (metamodel only), SBVR (mostly Controlled English)
  - RuleML: Deliberation RuleML (customizable expressivity)
RuleML Family of Sublanguages (1)
RuleML Family of Sublanguages (2)

- RuleML family covers a wide rule spectrum, from **Deliberation** rules to **Reaction** rules
  - Rule condition part reused across the spectrum
  - Syntactic uniformity enables further reuse
- Family constitutes a deep sublanguage lattice
  - Major sublanguage inclusion path:
    
    Deliberation ⊃ HOL ⊃ **FOL** ⊃ Derivation ⊃ **Hornlog** ⊃ Datalog ⊃ ... 

- Naf mix-in customization of Hornlog RuleML (Naf Hornlog RuleML) leads to Logic Programs
**RBDA$_s$-Style KBDA$_s$ Architecture:** Expressivity of Rule Systems

- The language of the global schema can be generalized from unary/binary (OBDA$_s$) to n-ary predicates (RBDA$_s$).
- When decidability of querying is not required, RBDA$_s$ expressivity can be extended from Datalog, Datalog$^\pm$, and description logic to Datalog$^+$, Horn logic, and FOL, as enabled by Deliberation RuleML 1.01.
- Features customizable with the MYNG 1.01 GUI.
- Moreover, Reaction RuleML 1.0 can express updates, as needed for KBDM$_s$ (Ontology-based Data Management).
**RBDA$_s$-Style KBDA$_s$ Architecture: Uniformity via Rule Systems**

- Rule-based style of Unified Architecture (earlier slide)
- Presentation syntax (":-"), serialization ("<RuleML>"), and semantics approach (model theory) uniform from queries (+ integrity constraints) to KBs to mappings to abstract DBs
- Division of labor between KB rules and mapping rules can be modified *without crossing paradigm boundaries*
  - Allows KB- and mapping-directed normal forms
- Assumptions (unique-name and closed-world) of DBs accommodated by *default assumptions of rule systems*
ΔForest: Study Overview

- **RuleML** provides a family of rule (incl. fact/data & query) languages of customizable expressivity, a family-uniform XML format, and a suite of (MYNG) tools for processing.
- **WSL** creates knowledge and publishes data about Swiss forests, giving an integrated federal perspective on heterogeneous databases of various (geographically, thematically, ...) specialized sources.
- RuleML-WSL collaboration has brought RuleML technology to bear on WSL knowledge and DBs for RBDA.
- Prepared data sets, defined global schema and mappings, formalized knowledge, and specialized RBDA architecture in support of complex statistical data analysis.
Forest: Study Team

- Int’l collaboration between: RuleML, WSL, UNB
- Participants:
  - Harold Boley (UNB, RuleML, WSL[visiting Jan-Jun 2014])
  - Rolf Grütter (WSL)
  - Tara Athan (RuleML)
  - Gen Zou (UNB)
  - Sophia Etzold (WSL)
DeltaForest: Three Local Data Sources

- **Productivity Research Areas:** Ertragskundeflächen (EKF)
  - 83 areas
  - Pure stands
  - Approximate 10-year intervals (time series of different lengths)
  - Data: .dbf/.csv files (ca. 100 MB)
  - 3 tables

- **Natural Forest Reserves:** Naturwaldreservate (NWR)
  - 36 areas
  - Pure and mixed stands
  - 10-year intervals
  - Data: .txt files (ca. 65 MB)
  - 1 file per area

- **Long-term Forest Ecosystem Research:** Langfristige Waldökosystemforschung (LWF)
  - 18 areas
  - Pure and mixed stands
  - 5-year intervals (since 1995)
  - Data: Oracle DB (ca. 500 GB)
  - RDB tables
Forest: Schemas for Architecture (a)-(d)

Global schema

Local schemas

PlotsStatic
- plot
- source
- x
- y
- altitude
- class

PlotsDynamic
- plot
- year
- age
- dmg
- dg
- n

SGAbundance
- plot
- species-group
- percentage

RelMortality
- plot
- year1
- year2
- relmortality

EKF
- dom
  - FNUM
  - SPEC
  - BNR
  - PC
- trees
  - FNUM
  - SPEC
  - BNR
  - BA
- vfl
  - FNUM
  - BBG
  - BNR
  - HA
- dyn
  - FNUM
  - YEAR
  - N

NWR
- dom
  - F_ID
  - SPEC
  - PC
- wr
  - X
- F_ID
- B_ID
- BANR
- NINV
- dyn
  - F_ID
  - NAME
  - NINV
  - YEAR
  - N

LWF
- dom
  - FNUM
  - SPEC
  - PC
- trees
  - PLAC
  - CLNR
  - BANR
- plots
  - PLAC
  - CLNR
  - X
- dyn
  - PLAC
  - NINV
  - YEAR
  - N

Keys – three composite – in bold red
Eligibility Criteria

1. Impact of forest management on tree mortality is negligible at the investigated sites
   - NWR: impact is negligible by definition
   - EKF: impact of grades A, B is negligible (≠ C, D, H, P)
   - LWF: not recorded

2. Plots are statistically independent of each other
   - Plots that are located within a distance of 500 meters from each other are possibly dependent

3. Plots contain a time-averaged abundance greater than a threshold value for at least one of the target species
Questions Addressed

1. Are there sufficiently many eligible plots in order to perform an analysis per main tree species?

2. Are there sufficiently many eligible plots in order to perform an analysis per main tree species and climatic region?

3. Which eligible plots represent pure tree stands and which eligible plots represent mixed tree stands?
Query Rewriting

q(?plot) :- .EligiblePlot(?plot)
  .TreeStandKey(?id ?plot "oak")
  .TreeStandAbundance(?id ?pct)
?pct >= 15.

Exists ?id (.TreeStandKey(?id ?plot ?sp) .TreeStandAbundance(?id ?pct)) :-

q(?plot) :- .EligiblePlot(?plot)
  .TreeStandMerged(?plot "oak" ?pct)
?pct >= 15.
Query Unfolding

\[ q2(?plot) : - \text{TreeStandMerged}(?plot \text{ "oak" } ?pct) \]
\[ \quad ?pct \geq 15. \]

\[ \text{TreeStandMerged}(?plot \text{ "oak" } ?pct) : - \text{EKF.dom}(?plot \text{ "Quercus petraea" } ?pct1) \]
\[ \quad \text{EKF.dom}(?plot \text{ "Quercus robur" } ?pct2) \]
\[ \quad ?pct = ?pct1 + ?pct2. \]

\[ q2(?plot) : - \text{EKF.dom}(?plot \text{ "Quercus petraea" } ?pct1) \]
\[ \quad \text{EKF.dom}(?plot \text{ "Quercus robur" } ?pct2) \]
\[ \quad ?pct1+?pct2 \geq 15. \]
Conclusions

- Ontology-Based Data Access (OBDA) founded on three kinds of rules: *Query rules* (including integrity rules), *KB rules* (for query rewriting and DB materialization), as well as *mapping rules* (for query unfolding and DB folding)
- OBDA complemented by Rule-Based Data Access (RBDA) and generalized to Knowledge-Based Data Access (KBDA)
- Specified an RBDA-uniform KBDAₚ architecture with unified mediator, warehouse, and bidirectional strategies
- RuleML used for XML-serialized rules, MYNG-customized rule expressivity, and platform-independent RBDA
- Introduced ΔForest specialization of RBDA architecture for statistical data analysis in ecosystem research at WSL
Future Work (1)

- Incorporate feedback from [Public Review](#) of [Deliberation RuleML 1.01](#) / [MYNG 1.01](#) into release and test Rulebases.
- Support implementations of specified architecture reusing (open source) KBDA technology (cf. [RBDA wiki page](#)).
- For high-precision $\text{RBDQ}_s$ language support, complement current techniques of Datalog$^+$ RuleML for Datalog$^\pm$ RuleML using context-sensitive/semantic validators for “-” constraints.
- Evaluate (mediator/warehouse, relational/graph, ...) trade-offs for $\text{KBDQ}_s$ in [PSOA RuleML](#) as executed in [PSOATransRun](#).
- Develop $\Delta$Forest study at WSL for extended and new data sources of big volume, variety, and velocity (e.g., about climate change).
- Augment geospatial $\text{KBDQ}_s$ mappings with [Optique](#) mapping (bootstrapping, repair, ...) techniques.
Future Work (2)

• Compare engines for OBDO \(_s\) and RBDQ \(_s\), including HYDRA and RDFox, w.r.t. expressivity and efficiency

• Adapt Semantic Automated Discovery and Integration (SADI) test cases for KBDQ \(_s\) experiments in PSOA RuleML querying

• Evaluate Abstract Logic-based Architecture Storage systems & Knowledge base Analysis (ALASKA) for RBDA \(_s\)

• Extend the KBDA \(_s\) architecture with semantic annotation rules for (Deep) Web data extraction (Deep Web Mediator)

• Use Grailog for KBDA \(_s\) data and knowledge visualization

• Explore synergies between the logical KBDA \(_s\) approach with statistical approaches, e.g. from Statistical Relational AI