# A Translator Framework for the Interoperation of Graph plus Relational Data and Rules on the Web

*AWoSS 2014*

*Saint John, NB, Canada, February 24, 2014*

Gen Zou, Harold Boley

Faculty of Computer Science,
University of New Brunswick, Fredericton, Canada

# Outline

# Outline

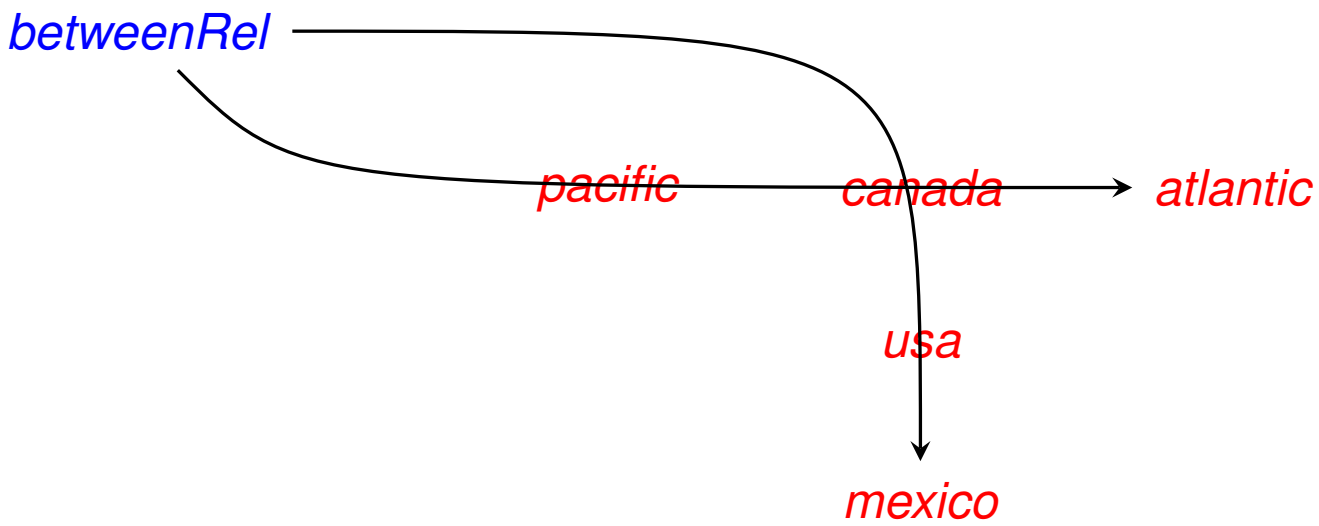
# Rule Languages

- Along with ontologies, rules provide foundation of knowledge representation and problem solving
- Used to express
  - Knowledge for ontology/rule-based data access
  - Domain-specific (e.g., biomedical) concept definitions
  - Associations among data
  - Business logics
  - Privacy/security/trust policies
  - Legal norms
  - ...
- Will enable **Semantic Analytics of Big Data**

# Rule Languages

- Paradigms for modeling entity dependencies:
  - Relational
  - Graph (Object-Centered)
  - Combined
- Since Knowledge Bases (KBs) have been developed in languages following all three paradigms, cross-paradigm translation, integration, and reuse is often necessary
- Need for an interoperation language and technology: Positional-Slotted Object-Applicative (PSOA) RuleML
- Naturally combinable with portability technology: Platform-independent implementation of PSOA RuleML

# Hypergraph Example – Relational Betweenness

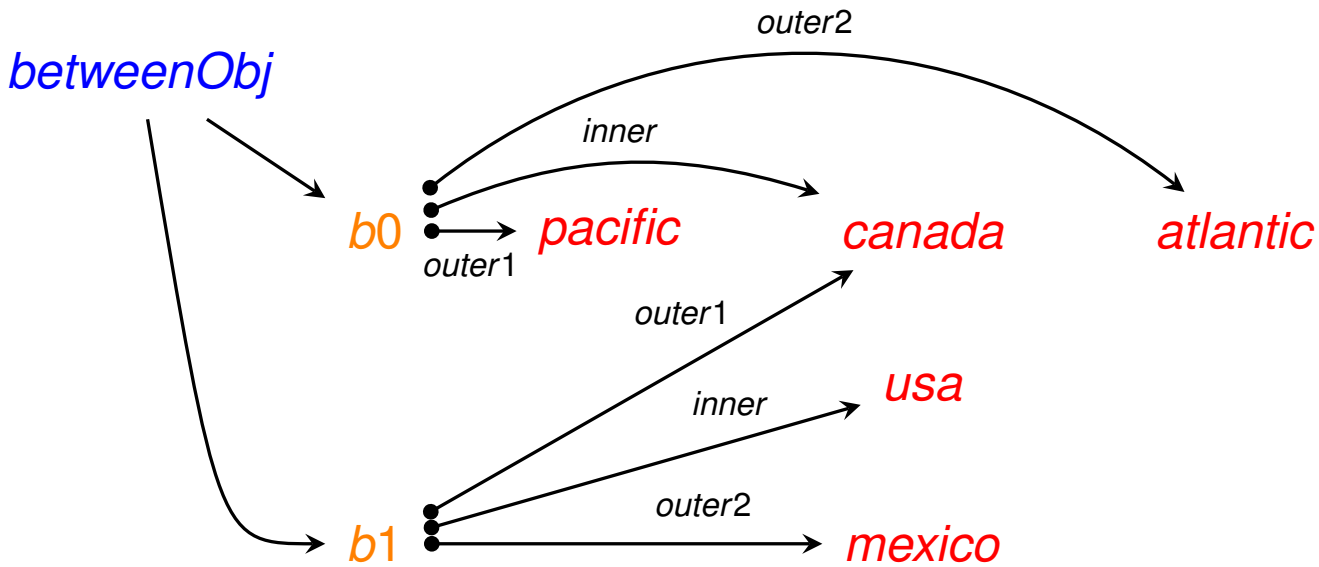



Directed hyperarcs cut through intermediate nodes (cf. Grailog)

## Facts

*betweenRel*(*pacific*, *canada*, *atlantic*)
*betweenRel*(*canada*, *usa*, *mexico*)

# Graph Example – Object-Centered Betweenness

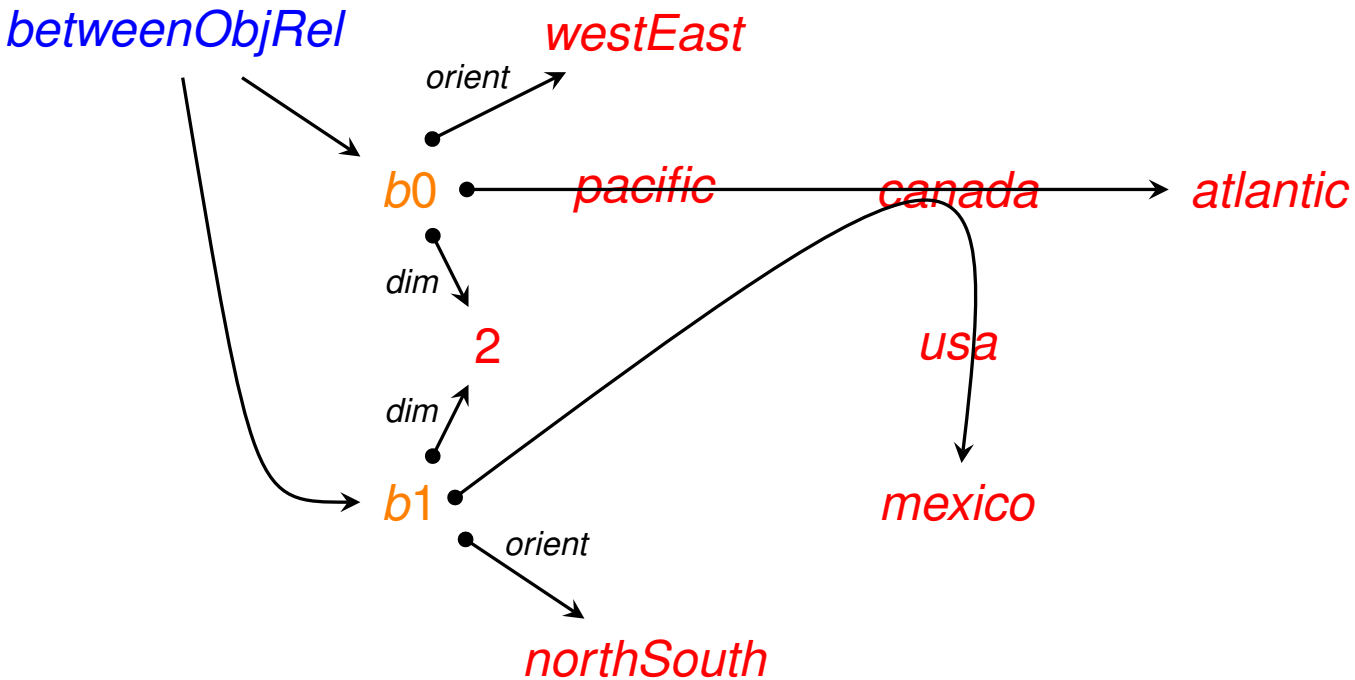



## Facts

"#" denotes "∈" for class membership; "→" associates a slot name with its filler

$b0\#betweenObj(outer1 \rightarrow pacific;\ inner \rightarrow canada;\ outer2 \rightarrow atlantic)$
$b1\#betweenObj(outer1 \rightarrow canada;\ inner \rightarrow usa;\ outer2 \rightarrow mexico)$

# Example – Integrated Betweenness (Enriched)




## Facts

$b0\#betweenObjRel(pacific, canada, atlantic;\ dim \rightarrow 2;\ orient \rightarrow westEast)$
$b1\#betweenObjRel(canada, usa, mexico;\ dim \rightarrow 2;\ orient \rightarrow northsouth)$

# Relational Rule Languages

- Widely used for relational DBs (SQL views) and KBs, representing information in classical logic
- Model dependencies among *n* entities as an *n*-ary **predicate** applied to an ordered sequence of *n* arguments, called **positional arguments**
- Languages: Common Logic, Prolog, TPTP-FOF, ...

# Graph (Object-Centered) Rule Languages

- Receive increasing attention because of expanding research and development in linked data on the Web, graph/'triple' stores, and big data in NoSQL DBs
- Each object is represented by a unique Object IDentifier (**OID**), typed by a **class**, and described by an unordered collection of **slots**, each being a pair of a name and a filler
- An OID-describing slotted term in AI is called a **frame** (represents a resource/'subject'-describing property list on the Semantic Web)
- Languages: RDF, N3, ...

# Object-Relational Rule Languages

- Combine the object-centered and relational paradigms, either in a heterogeneous or a homogeneous way
- Heterogeneous
  - Allow atomic formulas in both relational and object-centered forms, even mixed in the same rule
  - Languages: F-logic and RIF
- Homogeneous
  - Integrate relational and object-centered atomic formulas into a unified form
  - Language: PSOA RuleML

# PSOA RuleML

- Integrates relational and object-centered modeling
- Generalizes F-logic, RIF-BLD, and POSL
- Uses **p**ositional-**s**lotted **o**bject-**a**pplicative (**psoa**) terms, permitting a relation application to have an OID – typed by the relation – and, orthogonally, its arguments to be positional or slotted.

  General case (multi-tuple):

  $o\#f([t_{1,1} \ldots t_{1,n_1}] \ldots [t_{m,1} \ldots t_{m,n_m}]\ p_1->v_1 \ldots p_k->v_k)$

  Special cases (single-tuple brackets and zero-argument parentheses optional):

  Combined: $o\#f([t_1 \ldots t_n]\ p_1->v_1 \ldots p_k->v_k)$
  Positional: $o\#f([t_1 \ldots t_n])$
  Slotted: $o\#f(\ p_1->v_1 \ldots p_k->v_k)$
  Member-only: $o\#f()$

# Example of Querying a PSOA Fact and Rule

KB:

```
b1#betweenObjRel(canada usa mexico
                 dim->2 orient->northSouth)

Forall ?out1 ?in ?out2 ?b
(
   ?in#GeoUnit(neighborNorth->?out1
               neighborSouth->?out2) :-
      ?b#betweenObjRel(?out1 ?in ?out2
                       orient->northSouth)
)
```

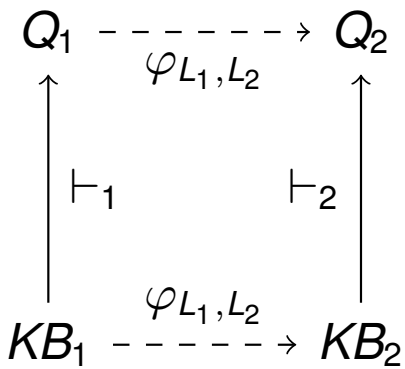
English Query: "*Which GeoUnit has Canada as its northern neighbor*?"
Query: `?X#GeoUnit(neighborNorth->canada)`
Answer: `?X=usa`

TPTP version in Common Logic would contribute to COLORE

# Semantics-Preserving Translation

For a translation $\varphi_{L_1,L_2}$ from the source language $L_1$ to the target language $L_2$:

- **Sound**: all entailments that hold after translation to $L_2$ already hold in $L_1$
- **Complete**: all entailments in $L_1$ still hold after translation to $L_2$
- **Semantics preserving** = sound + complete
- Semantics-preserving translation desired for **realizing** $L_1$ via $L_2$:

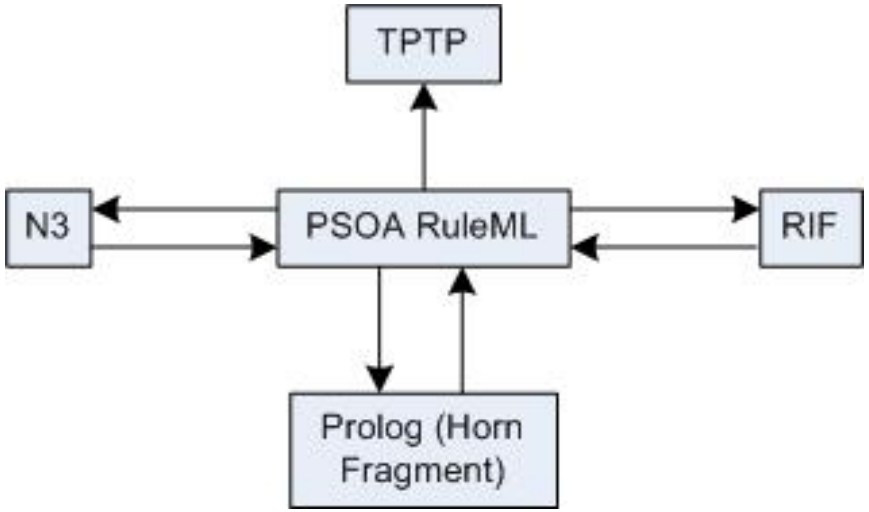


# Outline

# Outline

# System of Translation Pathways

- Use the homogeneous object-relational language PSOA RuleML as the interchange language
- Implement unidirectional or bidirectional translations between PSOA RuleML and
  - Relational languages: TPTP and the Horn subset of Prolog
  - Object-centered language: N3
  - Heterogenous object-relational language: RIF



# General Translation Steps

- Translations with PSOA RuleML as the **source language**
  - Transform the source KB into a normalized form with only elementary constructs, using a composition of transformations *staying within PSOA RuleML*
    - Objectification
    - Slotribution/tupribution
    - Skolemization
    - Unnesting
  - Map elementary constructs into the target language
- Translations with PSOA RuleML as the **target language**
  - Mostly syntactic translation, target different subsets of PSOA RuleML

# General Translation Steps

- Each translation is a recursive algorithm $\tau$ that can be specified as a mapping table

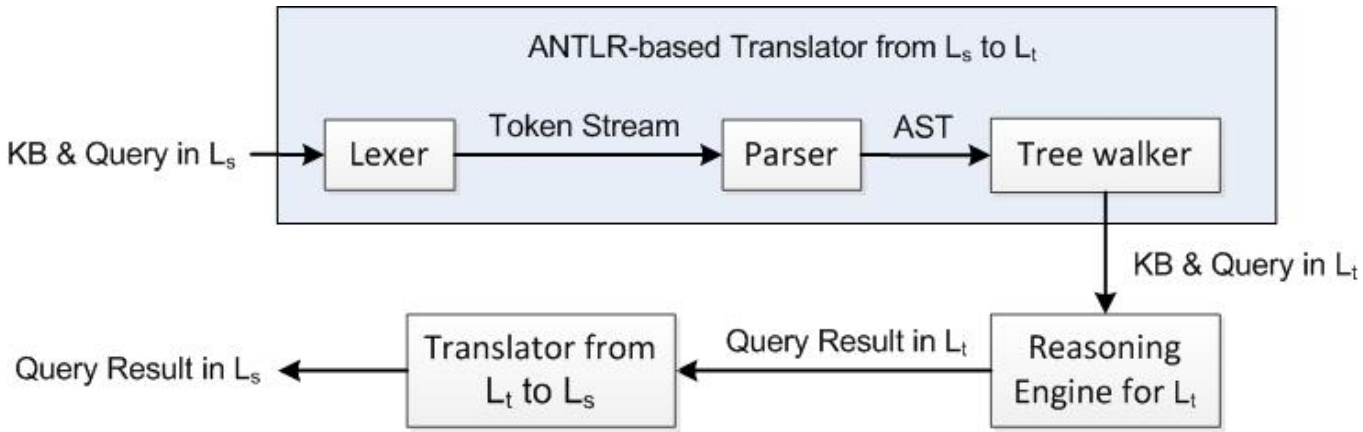
## Example of $\tau_{psoa}$: Mapping from PSOA RuleML to TPTP

| PSOA/PS Constructs | TPTP Constructs |
|---|---|
| `o # f()` | `member(`$\tau_{psoa}$`(o), `$\tau_{psoa}$`(f))` |
| `AND(`$f_1$ `...` $f_n$`)` | `(`$\tau_{psoa}(f_1)$ `&` `...` `&` $\tau_{psoa}(f_n)$`)` |
| $\varphi$ `:-` $\psi$ | $\tau_{psoa}(\psi) => \tau_{psoa}(\varphi)$ |
| ... | ... |

# Outline

# Portability Architecture

- Realizing increasing subsets of PSOA RuleML semantics in portablility architecture: Framework for translators and translator-based reasoning systems




- Instantiated in ANTLR (ANother Tool for Language Recognition) for executable translators implementing mapping tables, e.g. targeting $L_2$ = TPTP

# Portability Architecture

- ANTLR-based Java library of translators
  - Parsers/Generators: between concrete-syntax strings and abstract-syntax trees (ASTs)
  - Transformers: from ASTs to ASTs
- Service-based composition technology for translators along the pathways

# Outline

# Evaluation

- Evaluation metrics for soundness and completeness, analogous to, respectively, precision and recall in information retrieval
- Test Suite
  - Contains existing independent test cases, e.g. from `http://www.w3.org/2005/rules/wiki/Category:Test_Case`

    and test cases developed by ourselves, at `http://wiki.ruleml.org/index.php/PSOA_RuleML`
  - Also contains new use cases (following slides)
  - Covers different language and translator features

# Outline

# Ongoing: Geospatial Rules

- Extend our running example by formalizing and integrating further geospatial knowledge
- For each relation/class, determine most suitable paradigm (relational-only, object-centered-only, or object-relational combined) along with its argument treatment
- Enhance geospatial data with knowledge in a Rule-Based Data Access (RBDA) scenario, e.g. for WSL forestry data

# Planned: Financial Business Rules

- Develop KB with business rules in PSOA RuleML for the financial management of organizations, e.g. RuleML Inc.
- Data from financial documents and electronic financial statement spreadsheets will be mapped to PSOA facts
- OIDs are used to connect metadata (KB) and paper documents. Can also be used as dereferenceable URLs for scanned versions of documents.
- A systematics of financial business rules will be explored (e.g., validation rules for transaction plausibility checks, transformation rules for currency conversion, heuristic rules for checking-to-savings transfers)

# Demo

- PSOATransRun available online:
  `http://psoa.ruleml.cloudbees.net`
- Try test cases and send feedback
  `http://wiki.ruleml.org/index.php/PSOA_RuleML`