Semantic Integrator

UNIMAN-UPM Collaboration
8 June 2010

Research Goals - Integrator

Mapping & Rewriting

- Identify extensions to SPARQL algebra for streaming data
  - Windows
  - C-SPARQL has types issues
- Identify rewriting rules for the algebra
  - Use (Schmidt et al, 2010) as a starting point
- Generate source mappings from semantic data source property documents
- Incorporate semantic rewriting for SPARQL-STR queries (existing approaches available)
  - Use TBox assertions

C-SPARQL Model

RDF-Streams

- Pairs: RDF triple annotated with timestamp

(<subj, pred, obj>, ts)

Window

- Time-, or triple-, based definition
- Outputs a stream of triple-pairs

Aggregates (orthogonal to streaming work)

- Extends data with computed result

Evaluation Episodes

- State evaluation period for a query

1. D. F. Barbieri, D. Braga, S. Ceri, and M. Grossniklaus. An execution environment for C-SPARQL queries. In EDBT2010, pp 441–452, Lausanne, Switzerland, March 2010.
2. D. F. Barbieri, D. Braga, S. Ceri, E. D. Valle, and M. Grossniklaus. C-SPARQL: A continuous query language for RDF data streams. International Journal of Semantic Computing, 2010. To appear.

C-SPARQL Issues

Definition
- No declaration of stored/streaming parts

Windows
- Output type: bag of triples annotated with
  ts, index
- Physical window
  - Triple does not contain all information about an answer
  - Redefine to output number of answers?

Aggregates
- Normally used to condense information
- Not consistent with SPARQL 1.1 proposal
  - SPARQL 1.1 more "SQL-like"

C-SPARQL language
- Register query + evaluation periodicity
- See SPARQL 1.1 SERVICE additions

C-SPARQL Evaluation

Query Processing

- Stored RDF processed by triple store
- Steaming RDF processed by STREAM using CQL and mapping tuples to triples (Esper in fact, not STREAM/CQL)

Query Stack

- Generate Operator Graph
  - Places window immediately above stream source
  - Adds operators: stream, window, aggregate, filter
- Generate Denotational Graph
  - Distinguish static from streaming data
- Optimise
  - Push σ/π down – *not pushed all the way*

D. F. Barbieri, D. Braga, S. Ceri, and M. Grossniklaus. An execution environment for C-SPARQL queries. In EDBT2010, pp 441–452, Lausanne, Switzerland, March 2010.

SPARQL Optimisation

- Complexity of SPARQL arises from OPTIONAL clause
  - PTime or NP-complete without OPTIONAL
  - PSpace-complete with OPTIONAL
- 37 optimisation rules under set semantics
  - Not all carry through to bag semantics of SPARQL

Still looking into these rules.

M. Schmidt, M. Meier, and G. Lausen. Foundations of SPARQL query optimization. In ICDT2010, Lausanne, Switzerland, March 2010.

SPARQL and Relational Algebra

- Relational algebra semantics for SPARQL equivalent to mapping-based semantics
  Additional operators:
  - Outer union
  - Coalesce: $\dagger_{(a,b)->c}(R)$: $\forall t \in R$, if t(a) NOT NULL then t(c) <- t(a) else t(c) <- t(b).
- Semantics preserving query translation for RDBMS-based RDF stores
  - Storing RDF data in a RDBMS
  - Translation approach relies on RDF form

A. Chebotko, S. Lu, and F. Fotouhi. Semantics preserving SPARQL-to-SQL translation. Data & Knowledge Engineering, 68(10):973–1000, October 2009.

Research Idea

What do we need to do to C-SPARQL to make it work?

- *Improve type system*
  - *Window output timestamped bag of triples*
  - *Window-to-stream operators*
- *SPARQL 1.1 Aggregation*
- *Periodic evaluation*
  - *Do we need this?*
  - *What implications for SNEEql?*
    (This is what AG was discussing with Werner last year)

Can we extend the formalism of the translation approach of Chebotko et al for the integrator, i.e. formalise R2O/S2O?

- *Mappings that expose RDBMS data*
- *Streaming data*