Aligning RDF and LPGs

A Status Report of the OneGraph Initiative

Dr. Ora Lassila

Principal Technologist, Amazon Neptune
Co-chair, W3C RDF-star WG

Outline

1. On interoperability: RDF vs. LPGs – or RDF and LPGs?
2. OneGraph – introduction & status
3. W3C RDF-star – introduction & status
4. Other considerations and challenges

aws

Won’t get fooled again…

“Meet the new silos… just like the old silos”

- **Old silos:** Single-application -controlled data, at best behind a bespoke API
- **New silos:** Single-purpose (knowledge) graphs built without interoperability and interlinking in mind

Oxford English Dictionary says interoperability is “the ability of computer systems or programs to exchange information”

- (this is vague and/or limited)

aws

Our view of “graph interoperability”

Good: Link from one graph to another

Better: Link from an RDF graph to an LPG (and vice versa)

Even better: Freely intermix RDF and LPG data

Good: Use SPARQL on LPGs

Better: Use Gremlin, openCypher, etc., on RDF

Even better: Use any graph tooling on any graph data

aws

Project OneGraph

We think debates about “RDF vs. LPG” are pointless

We much better like the idea of “RDF and LPG”

OneGraph is an effort to provide graph interoperability and remove the adversarial thinking from graph choice

aws

Motivation for Project OneGraph

Amazon Neptune DB supports both RDF graphs and LPGs, but users have to choose

This choice determines which query language(s) they can use

Users are often confused

We'd like to see better alignment between the two graph models

Making RDF == LPG is not a goal

aws

The OneGraph Model

Our graph (meta)model accommodates various existing types of graphs

- RDF, RDF-star (as it is emerging)
- LPGs
- possibly others in the future (e.g., the WikiData flavor of RDF)

The goal is to have a single graph which can be “seen as” one of the constituent types of graphs

- in other words, we do not translate between graph models
- each of the constituent models is a projection from the OneGraph model, with potential loss of data visibility and expressiveness (e.g., classic RDF cannot do edge properties, so those are not visible)

aws

Challenges

Most challenges are rather simple (syntax, scalar datatypes, identifiers)

We have a proposal for how to use typed RDF literals to encode instances of composite data types (lists and maps)

Main challenge (“multiple edge instance problem”)

- an RDF graph is a mathematical set of triples, thus the same triple cannot occur more than once in the graph
- this makes it hard to say **:Liz :married :Dick**, annotate it with a date from 1964, and then realize they married again in 1975
- LPGs do not have this issue, but we cannot simply break one of the most fundamental features of RDF (other things in RDF rely on it)

aws

Implementing OneGraph

Neptune Analytics (our new service) already implements the OneGraph metamodel “behind the scenes”

Only openCypher is supported (so far) on Neptune Analytics

We recently introduced the ability to ingest RDF graphs and use openCypher over RDF data

- this mitigates some well-known shortcomings of SPARQL

The schedule for the remaining parts of OneGraph is yet to be fixed; also, no definite plans yet about “back-porting” Neptune Analytics’ features to Neptune DB

aws

Examples of openCypher over RDF

SPARQL

```
PREFIX nepo: <http://neptune.aws.com/ontology/…
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?name ?iata_code {
    ?airport a nepo:Airport ;
        nepo:ICAO "KMHT" ;
        nepo:IATA ?iata_code ;
        rdfs:label ?name
}
```

openCypher

```
PREFIX nepo: <http://neptune.aws.com/ontology/…
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

MATCH (airport: nepo::Airport)
WHERE airport.nepo::ICAO = "KMHT"
RETURN airport.rdfs::label, airport.nepo::IATA
```


aws

Examples of openCypher over RDF

Data modification

```
PREFIX nepo: <http://neptune.aws.com/ontology/…
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

MATCH (n1: nepo::Airport)
      <-[:nepo::source]-(r: nepo::Route)-
        [:nepo::destination]->
      (n2: nepo::Airport)
CREATE (n1)-[:nepo::hasOutboundRouteTo]->(n2)
```

Calling graph algorithms

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nepo: <http://neptune.aws.com/ontology/…

MATCH (airport)
CALL neptune.algo.pageRank(
       airport, {
         edgeLabels: [nepo::hasOutboundRouteTo],
         vertexLabel: nepo::Airport
       }
     )
YIELD rank WHERE rank > 0.0035
RETURN airport.rdfs::label, rank
ORDER BY rank
```


aws

W3C RDF-star

Initially, an effort to make it easier to use reification in RDF

- effectively, make it possible to easily use edge properties in RDF graphs

W3C RDF-star Community Group issued its findings in December 2021

Subsequently, a Working Group was formed; its work is still ongoing

The end goal is to produce specifications for

- RDF 1.2 (semantics, various serializations, etc.)
- SPARQL 1.2

aws

Challenges

RDF-star fixes the “multiple edge instance problem” via individually identified “triple terms”

- you can have a separate identifier for the same triple term, and use this identifier as any RDF resource (i.e., a node in the graph)
- an RDF triple with a triple term identifier (called “reifier”) as its subject is the equivalent of an LPG edge property

RDF-star goes well beyond LPG expressivity, some of which is a point of contention in the current W3C working group

- using a single identifier to reify multiple triples breaks the view that statements about (reified) statements are like edge properties

aws

What about schemas and ontologies?

Basic interoperability provided by OneGraph would make it possible to use RDF's schema mechanisms for LPGs → ontologies for LPGs!

However, no good understanding of how to use edge properties (or more generally, the RDF variant thereof) in ontologies

- some people feel this is unnecessary (they would be correct, but pragmatic considerations still say edge properties are desirable)

Exploratory work: PG-Schema for RDF [Lassila 2024]

Semantics?

RDF-star WG has defined formal semantics for RDF 1.2

- compatible with RDF 1.1, and allows RDFS & OWL to be built on top
- formalizes semantics for reification (something we never did earlier)
- (we are not addressing named graph semantics this time)

What about semantics for LPGs?

Using RDF mechanisms on LPGs (e.g., reasoning) still needs to be explored and experimented on

aws

Summary

OneGraph will provide graph interoperability

Current status: openCypher over RDF now available, work continues

RDF-star WG is working to complete RDF 1.2 and SPARQL 1.2

RDF 1.2 will improve RDF/LPG alignment, but some issues remain

Use of schemas, ontologies, reasoning, etc., still needs work

aws

More information

Questions? Contact: ora@amazon.com

AWS Database Blog post on openCypher-over-RDF

aws.amazon.com/blogs/database/build-and-deploy-knowledge-graphs-faster-with-rdf-and-opencypher/

Semantic Web Journal article introducing OneGraph

www.semantic-web-journal.net/content/onegraph-vision-challenges-breaking-graph-model-lock-0

ESWC 2024 paper on composite datatypes

www.lassila.org/publications/2024/HartigEtAl_SPARQLCDTs_PosterPaper2024.pdf

Our work on PG-Schema for RDF

github.com/aws-samples/amazon-neptune-samples/tree/master/pg-schema-for-rdf

aws