Stardog Position Statement: Path queries in SPARQL

Evren Sirin, Chief Scientist, Stardog Pavel Klinov, Vice President (VP), Research & Development, Stardog

Background

Stardog is a knowledge graph platform that allows large enterprises to connect, query, and retrieve data of all structures, schemas, and velocities. The core of the platform is based on W3C specifications RDF, OWL, SPARQL and R2RML and being used many Fortune 500 companies around the world. Evren Sirin and Pavel Klinov have been working on knowledge graphs and semantic technologies since early 2000s.

Summary

RDF and various W3C specifications that build on top of RDF provide a powerful ecosystem to solve challenging data problems. There are certainly some usability issues (namespaces, lists, bnodes) that make RDF harder for newcomers but most of them are/can be addressed by tool support and documentation without fundamental changes to the standards.

In this position statement we present an extension to SPARQL that addresses the problem of finding and retrieving shortest paths in RDF graphs.

Position

RDF is a flexible and powerful graph data model allowing people to represent and easily connect heterogeneous data both on the Web as Linked Data but also as importantly in the enterprise to solve the data silo problem. W3C specifications that build on top of RDF provide schema languages (RDFS and OWL), expressive constraints (SHACL), mappings to relational data sources (R2RML) and a comprehensive query language (SPARQL). This set of standards makes it possible to address many challenging data management and analysis problems in a way that is not otherwise possible.

SPARQL is a very expressive query language yet there is one major limitation. When we query RDF graphs with SPARQL we get the results either as a table (SELECT) or as a graph with a fixed template (CONSTRUCT). SPARQL has a property path feature that can be used for queries that recursively traverse the RDF graph and find two nodes connected via a complex path of edges. But the result of a property path query is only the start and end nodes of the path and does not include intermediate nodes. To find intermediate nodes additional or more complex queries are needed which is not the case in other graph query languages.

In order to address this problem in Stardog we have extended SPARQL with a path query form that can be used for finding and retrieving paths of arbitrary complexity. Path queries use SPARQL Basic Graph Patterns (BGP) as the main building block and provide additional syntactic elements for features like shortest paths, limiting the length of paths, ordering paths based on a condition, etc. The syntax for path queries is as follows:

```
PATHS \[SHORTEST|ALL\] [CYCLIC] [<DATASET>]
START ?s [ = <Term> | <GRAPH PATTERN> ]
END ?e [ = <Term> | <GRAPH PATTERN> ]
VIA <GRAPH PATTERN> | <VAR> | <PATH>
[MAX LENGTH <int>]
[ORDER BY <condition>]
[OFFSET <int>]
[LIMIT <int>]
```

The following query returns all paths starting from Alice to any node in the graph by traversing any edge:

```
PATHS START ?x = :Alice END ?y VIA ?p
```

The path query results can be retrieved in many different formats including a property graph like syntax:

```
(:Alice)-[p=:knows]->(:Bob)
(:Alice)-[p=:knows]->(:Bob)-[p=:knows]->(:David)
(:Alice)-[p=:knows]->(:Bob)-[p=:worksWith]->(:Charlie)
(:Alice)-[p=:knows]->(:Bob)-[p=:worksWith]->(:Charlie)-[p=:parentOf]->(:Eve)
```

Sometimes a relationship between two nodes might be implicit and there might not be an explicit link between those two nodes in the graph. For the well-known Six Degrees of Bacon game where we are looking for paths between movie stars we might need to traverse implicit co-star relationships based on explicit links between the stars and the movies. Such a path query would look like this:

```
PATHS START ?x = :Kevin_Bacon END ?y = :Robert_Redford
VIA { ?movie a :Movie ; :starring ?x , ?y  }
```

The BGP in the VIA pattern defines the kind of connections between nodes `?x` and `?y` that we are interested in when finding the paths.

There are many problems addressed by the path query extension and the details can be found in the Stardog documentation [1] and our blog posts [2, 3].

References

[1] Stardog property path documentation - https://www.stardog.com/docs/#_path_queries [2] A path of our own - https://www.stardog.com/blog/a-path-of-our-own/ [3] GraphQL and paths - https://www.stardog.com/blog/graphql-and-paths/