Towards Semantic Network Models via Graph Databases for SDN Applications

Talita de Paula Cypriano de Souza and Christian Esteve Rothenberg
University of Campinas (UNICAMP)
{cypriano,chesteve}@dca.fee.unicamp.br

Mateus Augusto Silva Santos
Ericsson Research
Indaiatuba, SP, Brazil
mateus.santos@ericsson.com

Luciano Bernardes de Paula
Federal Institute of Sao Paulo (IFSP)
Braganca Paulista, SP, Brazil
lbernardes@ifsp.edu.br
Agenda

- Introduction
- Goals
- Related Work
- Proposal
- Experimental Evaluation and Results
- Conclusions and Future Work
Agenda

- Introduction
- Goals
- Related Work
- Proposal
- Experimental Evaluation and Results
- Conclusions and Future Work
Introduction & Motivation

- Rise of Software Defined Networking (SDN);
- Information models and data structures for topology (and more) needed in any network management / control technology;
  - Lots of related standardization efforts (IETF, YANG, ONF CIM, etc.)
- Evolution and Maturity of the Semantic Web (e.g., ontologies, RDF)
  - ... and success stories in niche application domains;
  - Networking? NML
- Popularity of NoSQL DB, e.g., graph databases such as Neo4j;
  - Scalability properties & Natural approach to graph/network problems
Software Defined Networking

- **Software Defined Networking (SDN):**
  - Clean / Programmatic Separation of control and data planes;
  - New abstractions in controlling and network forwarding;

- API for packet flow abstraction
  - OpenFlow Protocol;

- Network Topology Abstraction;
  - ?

KREUTZ, D. et al., 2015
Semantic Models

✓ Principles of the Semantic Web:
  ✓ Allows reuse of information;
  ✓ Data integration among organizations;
  ✓ Enhanced (rich) Web search;
  ✓ Guaranteed accessibility

✓ Ontology;
  ✓ “specification of a conceptualization”

✓ Web Ontology Language (OWL);

✓ Resource Description Framework (RDF):
  ✓ <subject, predicate, object>;
Databases

✓ Relational Database Model
  ✓ Consolidated; Well Documented;
  ✓ ACID Transactions (Atomicity, Consistency, Isolation and Durability)
  ✓ Limitations:
    ✓ Queries of highly interconnected data;
    ✓ Data Modelling is Adapted

✓ NOSQL (http://nosql-database.org/)
  ✓ Schema Free;
  ✓ Scalability;
  ✓ Disponibility;
  ✓ Response time;
  ✓ Horizontal Scalling;
Graph Databases (GDB)

✓ Graph:
  ✓ Nodes;
  ✓ Edges;

✓ Topology;

✓ Interconnected Data;
  (Relationship-centered);

✓ Natural modeling of problems, e.g:
  ✓ Semantic Web;
  ✓ Computer Networks;
  ✓ Recommendation Engines, etc;
Agenda

• Introduction

• Goals

• Related Work

• Proposal

• Experimental Evaluation and Results

• Conclusions and Future Work
Goals

- Apply a **semantic model** to describe network topologies (and complete network+compute infrastructures) in the context of SDN controllers leveraging **graph databases**;
- Map **SDN primitives** (in the literature) as graph database queries;
- Identify **limitations** of the chosen semantic language in support of SDN application primitives;
- **PoC system profiling**: Evaluate the performance of a prototype;
Agenda

- Introduction
- Goals
- Related Work
- Proposal
- Experimental Evaluation and Results
- Conclusions and Future Work
SDN Controllers

• ONIX Controller (Koponen et al., 2010):
  • Pioneer distributed control plane implementation of SDN;
  • Graph model to define the partitioning and state distribution/maintenance among different controllers;

• Graphs in SDN (Pantuza et al., 2014):
  • Support of dynamic network representation;
  • Minimum Spanning tree of network graph in a real time;

• NetGraph Library (Raghavendra et al., 2012):
  • Periodic updates of network state;
  • Queries results to SDN controller;

Do not use semantic notations (n)or store network graphs in a persistent way.
Semantic Model for Networking

*Network Markup Language – NML* (van der Ham et al., 2013)

- Single network description standard under guidance of the Open Grid Forum (OGF)
  
  “complex multi-layer path finding, with a technology independent algorithm”

- Supports description of multi-layer and multi-domain networks:
  - Virtualized networks;
  - Heterogeneous network technologies;

- Model extensibility as necessary, e.g.,
  - NOVI – Future Internet Platform:
    - [http://www.fp7-novi.eu/](http://www.fp7-novi.eu/)
  - GEYSERS – Virtualizing Optical Networks:
  - CINEGRID – Distribution of Digital Media:
    - [http://www.cinegrid.org](http://www.cinegrid.org)
Semantic Models

Main NML Classes and Properties

GHIJSEN, M. et al., 2013
Graph Databases

• Benchmark of GDBs (Jouili e Vansteenbergh, 2013):
  • Neo4j, OrientDB, Titan, DEX;
  • Neo4j obtained the best query time results;

• Auditing Cloud Architectures (Soundararajan and Kakaraddi, 2014):
  • Neo4j e Cypher:
    • Risk Analysis;
    • Simple Reporting;
    • Inventory Comparison;
Graph Databases

Neo4j

- Native Storage and Processing of Graph;
- Open Source (Community Version);
- Property Model:
  - Nodes and relationships have properties;
- Query Language:
  - Cypher;
  - Gremlin (TinkerPop);
Agenda

• Introduction
• Goals
• Related Work
• Proposal
• Experimental Evaluation and Results
• Conclusions and Future Work
Strawman Architecture

[Applications]
- Routing (RIP, OSPF, BGP)
- Security (IDS, Firewall)
- Analytics (flow statistics)

[SDN Controller]
- **Northbound Interface** (QoS, monitoring)
- **Centralized Controller** (events and connections)
- **Southbound Interface** (OpenFlow, SNMP)
- **Dynamic Topology**
- **Graph DB**
- **Parser** (semantic model)
- **East-west Interface**
## Primitives Analysis
### Compatibility of Primitives [NetGraph]

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Semantic Model</th>
<th>GBD</th>
<th>Read/Write</th>
</tr>
</thead>
<tbody>
<tr>
<td>setEdgeWeight</td>
<td>No</td>
<td>Yes</td>
<td>W</td>
</tr>
<tr>
<td>getEdgeWeight</td>
<td>No</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>countInDegree</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>countOutDegree</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>countNeighbors</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>computeMST</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>computeAPSP</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>computeSSSP</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>doesRouteExist</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>computeKSSSP</td>
<td>Yes</td>
<td>Yes</td>
<td>R</td>
</tr>
<tr>
<td>delete</td>
<td>Yes</td>
<td>Yes</td>
<td>W</td>
</tr>
<tr>
<td>insert</td>
<td>Yes</td>
<td>Yes</td>
<td>W</td>
</tr>
</tbody>
</table>
Agenda

• Introduction
• Goals
• Related Work
• Proposal
• Experimental Evaluation and Results
• Conclusions and Future Work
Data Modeling

✓ Example of Relationship Modeling between Nodes “9” and “0”
### Topologies

- Topology Generator: BRITE (Boston University);

<table>
<thead>
<tr>
<th>Topology</th>
<th>Nodes (BRITE)</th>
<th>Resultant Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tiny</td>
<td>10</td>
<td>76 nodes (160 relationships)</td>
</tr>
<tr>
<td>Small</td>
<td>100</td>
<td>640 nodes (1.760 relationships)</td>
</tr>
<tr>
<td>Medium</td>
<td>1.000</td>
<td>4.978 nodes (11.912 relationships)</td>
</tr>
<tr>
<td>Large</td>
<td>10.000</td>
<td>109.932 nodes (359.728 relationships)</td>
</tr>
</tbody>
</table>
Queries

- Fixed Internet-like topologies (BRITE) of different sizes;
- Random attributes;
- Each primitive executed 1,000 times in each topology;
- Cypher Language:
  - E.g.:

```clojure
MATCH (n:Node)-[:hasOutboundPort]->(p:Port)-[:isSource]->(l:Link)
WHERE n.name="A"
RETURN COUNT(l) AS CountOutDegree
```
## Results Analysis
### Large Topology (ms)

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Average</th>
<th>Standard Deviation</th>
<th>99 Percentile</th>
</tr>
</thead>
<tbody>
<tr>
<td>setEdgeWeight</td>
<td>162.33</td>
<td>9.46</td>
<td>205.01</td>
</tr>
<tr>
<td>getEdgeWeight</td>
<td>1.70</td>
<td>0.74</td>
<td>4.00</td>
</tr>
<tr>
<td>countInDegree</td>
<td>854.53</td>
<td>146.77</td>
<td>1.399.05</td>
</tr>
<tr>
<td>countOutDegree</td>
<td>425.17</td>
<td>68.36</td>
<td>699.02</td>
</tr>
<tr>
<td>countNeighbors</td>
<td>4.45</td>
<td>2.27</td>
<td>10.01</td>
</tr>
<tr>
<td>doesRouteExist</td>
<td>37.51</td>
<td>29.09</td>
<td>73.06</td>
</tr>
<tr>
<td>computeMST</td>
<td>1.44</td>
<td>1.25</td>
<td>3.02</td>
</tr>
<tr>
<td>computeSSSP</td>
<td>5.47</td>
<td>4.98</td>
<td>29.00</td>
</tr>
<tr>
<td>computeKSSSP</td>
<td>26.21</td>
<td>37.23</td>
<td>81.04</td>
</tr>
<tr>
<td>computeAPSP</td>
<td>1.04</td>
<td>0.68</td>
<td>3.01</td>
</tr>
<tr>
<td>delete</td>
<td>1053.89</td>
<td>162.55</td>
<td>1637.02</td>
</tr>
<tr>
<td>insert</td>
<td>3.57</td>
<td>3.21</td>
<td>16.01</td>
</tr>
</tbody>
</table>
Results Analysis

Insert Primitive

Time (ms)

small  medium  large

Topology

95 Percentile
5 Percentile
Results Analysis

ComputeKSSSP Primitive

Time (ms)

small  medium  large

Topology
Results Analysis

The graph shows a box plot for the "DoesRouteExist Primitive" across different topologies: small, medium, and large. The x-axis represents the topology, and the y-axis represents the time (in ms). The box plot indicates the distribution of times for each topology, with the central line within the box representing the median, and the ends of the box showing the interquartile range. The whiskers extend to show the range of the data, excluding outliers.

The large topology has a significantly higher time compared to the small and medium topologies, suggesting that the performance of the "DoesRouteExist Primitive" is more negatively affected by larger topologies.
Results Analysis
Primitives with large response time

- Count In Degree and Count Out Degree:
  - Number of hops (different relationships types):
    \[ \text{NodeA} \leftarrow \text{hasInboundPort} \leftarrow \text{Port} \leftarrow \text{isSink} \leftarrow \text{Link} \]

- Delete:
  - Number of hops;
  - Depends on connectivity of deleted node

- Set Edge Weight:
  - Read-Write Operation;
Results Analysis
Shortest Path Primitives

• All Pairs of Shortest Path (compute APSP) is faster than K-Shortest Path (compute KSSSP) and Shortest Path (compute SSSP);

• The GDB optimizes All Pairs of Shortest Path Computing, because during the traversal it computes the shortest path among the intermediates nodes;
Relational Model
Enhanced Entity Relationship Model (EER)
## Relational Model
### Results (MySQL) – Large Topology

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Average</th>
<th>Standart Deviation</th>
<th>Percentile 99</th>
</tr>
</thead>
<tbody>
<tr>
<td>countInDegree</td>
<td>1,39</td>
<td>4,57</td>
<td>22,02</td>
</tr>
<tr>
<td>computeSSSP</td>
<td>18,13</td>
<td>3,82</td>
<td>26,00</td>
</tr>
<tr>
<td>computeAPSP</td>
<td>2,11</td>
<td>1,39</td>
<td>7,00</td>
</tr>
<tr>
<td>Delete</td>
<td>162,86</td>
<td>79,93</td>
<td>405,00</td>
</tr>
<tr>
<td>Insert</td>
<td>137,36</td>
<td>43,80</td>
<td>300,00</td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>Graph Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Adapted modelling to tables;</td>
<td>• Natural modelling;</td>
</tr>
<tr>
<td>• For computing shortest paths it was necessary to</td>
<td>• Native functions to compute shortest paths;</td>
</tr>
<tr>
<td>implement/adapt an algorithm</td>
<td></td>
</tr>
<tr>
<td>• Lower response time:</td>
<td>• Lower response time:</td>
</tr>
<tr>
<td>• CountInDegree</td>
<td>• ComputeSSSP</td>
</tr>
<tr>
<td>• Delete</td>
<td>• Insert</td>
</tr>
</tbody>
</table>
Agenda

• Introduction
• Goals
• Related Work
• Proposal
• Experimental Evaluation and Results
• Conclusions and Future Work
Conclusions

Feasibility of indexing a network topology following a semantic model (NML) in a graph database (Neo4j) in the context of SDN primitives;

- Integration Architecture (NML-to-Neo4j parser) proposed;
- Basic SDN control application primitives reproduced;
- Some limitations of the semantic model were identified;
- Neo4j graph DB technology choice compatible (property graphs) with the network modeling problem
  - Promising performance and scalability
  - Cypher language exhibited good flexibility;
Future Work

• Evaluate the performance with dynamic workloads and applications on OpenDaylight controller using REST APIs;

• Develop extensions of the Semantic Model (NML) to meet SDN (and NFV) applications needs;
  • Use cases under investigation include SDN eXchanges, east/west interfaces, controller platform for multiple applications

• Develop new graph-oriented primitives;

• Explore system optimizations to reduce latency and increase scalability;
Thanks! Obrigado! (More) Questions?

https://github.com/intrig-unicamp/NML-Neo4j

cypriano@dca.fee.unicamp.br
Backup
## Results Analysis
### Small Topology (ms)

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Average</th>
<th>Standard Deviation</th>
<th>99 Percentile</th>
</tr>
</thead>
<tbody>
<tr>
<td>setEdgeWeight</td>
<td>8,78</td>
<td>3,23</td>
<td>23,02</td>
</tr>
<tr>
<td>getEdgeWeight</td>
<td>1,73</td>
<td>0,76</td>
<td>3,00</td>
</tr>
<tr>
<td>countInDegree</td>
<td>17,94</td>
<td>11,36</td>
<td>65,01</td>
</tr>
<tr>
<td>countOutDegree</td>
<td>8,35</td>
<td>3,46</td>
<td>23,00</td>
</tr>
<tr>
<td>countNeighbors</td>
<td>6,16</td>
<td>22,43</td>
<td>14,07</td>
</tr>
<tr>
<td>doesRouteExist</td>
<td>6,55</td>
<td>3,82</td>
<td>15,02</td>
</tr>
<tr>
<td>computeMST</td>
<td>1,12</td>
<td>0,66</td>
<td>2,00</td>
</tr>
<tr>
<td>computeSSSP</td>
<td>1,34</td>
<td>1,38</td>
<td>4,00</td>
</tr>
<tr>
<td>computeKSSSP</td>
<td>2,94</td>
<td>3,44</td>
<td>12,00</td>
</tr>
<tr>
<td>computeAPSP</td>
<td>1,04</td>
<td>0,84</td>
<td>4,01</td>
</tr>
<tr>
<td>delete</td>
<td>20,71</td>
<td>7,20</td>
<td>48,01</td>
</tr>
<tr>
<td>insert</td>
<td>3,66</td>
<td>3,26</td>
<td>15,02</td>
</tr>
</tbody>
</table>
Results Analysis

ComputeSSSP Primitive

- Small
- Medium
- Large

Time (ms)