WaterFowl: a Compact, Self-indexed and Inference-enabled immutable RDF Store
ESWC 2014

Olivier Curé\textsuperscript{a}, Guillaume Blin\textsuperscript{a}, Dominique Revuz\textsuperscript{a}, David Faye\textsuperscript{b}

\textsuperscript{a}: UPEM, LIGM UMR CNRS 8049, France  
\textsuperscript{b}: Univ. Gaston Berger de Saint Louis, LANI, sénégal

Wednesday, May 28th, 2014
Outline

- Motivation
- WaterFowl system
  - Presentation
  - Architecture
  - Two-layer storage approach
  - Dictionary encoding
  - Query processing
  - Inference
- Evaluation
- Conclusion
Motivation

- Facing increasing volumes of RDF triples
- The management issues (load balancing, SPOF) are usually addressed by distributing the workload (Virtuoso, OWLIM, bigdata, etc.)
- Requires a lot of work on replication.
- Few systems address high level compression, e.g., BitMat\textsuperscript{1}, TripleBit\textsuperscript{2}
- There is a need for highly compressed and distributed RDF Stores

\textsuperscript{1}Medha Atre et al (2010). Matrix "Bit" loaded: a scalable lightweight join query processor for RDF data, WWW, pp 41-50
\textsuperscript{2}Yuan, P.et al (2013). Triplebit: a fast and compact system for large scale rdf data. PVLDB,6(7):517528
Like HDT (Head Dictionary Triples)\textsuperscript{3}, WaterFowl aims at compressing the data using Succinct Data Structures (SDS), limits I/O by accessing the structures in directly in RAM.

Our contributions:

- even more compression (using SDS only) and experimenting with different implementations: compact, unique self-indexing
- propose algorithms for query processing and optimization
- clever encoding of dictionaries to support RDFS entailment regime

\textsuperscript{3}Miguel A. Martnez-Prieto, Mario Arias Gallego, Javier D. Fernandez: Exchange and Consumption of Huge RDF Data. ESWC 2012
Two-layer storage approach
Dictionary encoding
Query processing
Inference
Two-layer storage approach

- Uses two forms of SDS: Bitmaps and Wavelet trees
- Wavelet trees are binary balanced trees and enable to encode large alphabet.
- SDS are known for using an amount of space that is close to the information-theoretic lower bound.

Operations on SDS:

- $access(i)$: returns the elements stored at the $i^{th}$ position.
- $rank_a(i)$: returns the number of occurrences of $a$ in the binary substring ranging from position 0 to $i$.
- $select_a(i)$: returns the position of the $i^{th}$ occurrence of $a$. 
Motivation
WaterFowl System
Evaluation
Conclusion

Presentation
Architecture
Two-layer storage approach
Dictionary encoding
Query processing
Inference

Concept Encoding
ub:University = 00 110
ub:Department = 00 001
ub:AssociateProfessor = 01 010 10 11 010
ub:Course = 10 01

Property Encoding
rdf:type = 00
ub:name = 01 010
ub:subOrganizationOf = 10 00010
ub:teacherOf = 10 00101
ub:worksFor = 10 00111 1 0

Encoding of the rest
Unio = 0000
Dp0 = 0010
AP0 = 0011
C15 = 0100
C16 = 0110
"University" = 1000
"Department" = 1010
"Cure" = 1100
"Course15" = 1110
Motivation
WaterFowl System
Evaluation
Conclusion

Presentation
Architecture
Two-layer storage approach
Dictionary encoding
Query processing
Inference

Bp 101001000101

Uni0
Dpt0
AP0
C15
C16

rdf:type name rdf:type name subOrganizationOf rdf:type name teacherOf worksFor rdf:type name rdf:type

University "University0" Department "Department0" Uni0 AssociateProfessor "Cure" C15 C16 Dpt0 Course "Course15" Course
Motivation
WaterFowl System
Evaluation
Conclusion
Presentation
Architecture
Two-layer storage approach
Dictionary encoding
Query processing
Inference

```
rdf:type: 00..
ub:name: 01010..
ub.subOrganizationOf: 100010..
```

```
Uni0   Dpt0   AP0   C15   C16

uni0
```

```
Wtp: 000010011000
```

```
010101010  000
```

etc..
Concept Encoding
ub:University = 00 110
ub:Department = 00 001
ub:AssociateProfessor = 01 010 10 11 010
ub:Course = 10 01

Property Encoding
rdf:type = 00
ub:name = 01 010
ub:subOrganizationOf = 10 00 010
ub:teacherOf = 10 00 101
ub:worksFor = 10 00 111 1 0

Encoding of the rest
Uni0 = 0000
Dp0 = 0010
AP0 = 0011
C15 = 0100
C16 = 0110
"University" = 1000
"Department" = 1010
"Cure" = 1100
"Course15" = 1110
Dictionary encoding

- Distinction between ABox and TBox
- Automata-based data structure for the ABox
- Hash table-based for the TBox
  - two structures: one for the concept another one for the properties
  - key of the hash tables are binary strings representing the subsumption relationships
- Hence, all entries entries of a hierarchy have the same prefix.
(a) Concept hierarchy encoding

00 Organization
  000 self
  001 Department
  010 Institute
  011 Program
  100 ResearchGroup
  101 University

01 Person
  000 self
  001 Director
  010 Employee
    00 self
    01 AdministrativeStaff
      00 self
      01 ClericalStaff
      10 SystemsStaff

10 Faculty
  00 self
  01 Lecturer
  10 PostDoc
  11 Professor
    00 self
    001 AssistantProfessor
    010 AssociateProfessor
    011 Chair
    100 Dean
    101 FullProfessor
    110 VisitingProfessor

...  

10 Work
  00 self
  01 Course

...  

(b) Property hierarchy encoding

00 rdf:type
01 datatype property
  000 age
  001 emailAdress
  010 name
  011 officeNumber
  100 researchInterest
  101 telephone
  110 title

10 Object Property
  00000 advisor
  00001 affiliateOf
  00010 subOrganizationOf
  00011 degreeFrom
    00 self
    01 doctoralDegreeFrom
    10 mastersDegreeFrom
    11 undergraduateDegreeFrom
  00100 hasAlumnus
  00101 teacherOf
  00110 member
  00111 memberOf
    0 self
    1 worksFor
      0 self
      1 headOf
- SPARQL queries can be translated with a set of rank, select and access operations.
- We have designed a query optimization approach based on the:
  - identification of the cost of Basic Graph Pattern (BGP) in terms of access, rank and select
  - storing basic statistics about triple element distribution
  - design of heuristics based on these costs and statistics
subClassOf and subPropertyOf inferences are performed by using prefix versions of rank and select operations.

We support multiple inheritance using a mapping table storing the different identifiers of a concept.

For the domain and range inferences:
- adopt a materialization approach (types of objects and subjects)
- reduce of size of the materialization but storing the most general concept
Table 1. Size of database serialization (MB) and Time to prepare datasets

<table>
<thead>
<tr>
<th></th>
<th>Size in MB</th>
<th>Time in sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>univ100</td>
<td>univ1000</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>831,717</td>
<td>7,795,458</td>
</tr>
<tr>
<td>BigOWLIM</td>
<td>2,411,260</td>
<td>22,600,088</td>
</tr>
<tr>
<td>Jena TDB</td>
<td>1,492,057</td>
<td>13,984,467</td>
</tr>
<tr>
<td>WaterFowl Mode 1</td>
<td>91,539</td>
<td>922,106</td>
</tr>
<tr>
<td>WaterFowl Mode 2</td>
<td>71,064</td>
<td>720,396</td>
</tr>
<tr>
<td>WaterFowl Mode 3</td>
<td>77,351</td>
<td>798,829</td>
</tr>
</tbody>
</table>
### Table 2. Query answering times (sec) on univ1000

<table>
<thead>
<tr>
<th></th>
<th>LUBM QR#1</th>
<th>LUBM QR#2</th>
<th>LUBM QR#14</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDF-3X</td>
<td>1.65</td>
<td>14.88</td>
<td>1640</td>
</tr>
<tr>
<td>BigOWLIM</td>
<td>138</td>
<td>5.7</td>
<td>3320</td>
</tr>
<tr>
<td>Jena TDB</td>
<td>3.52</td>
<td>2.18</td>
<td>2998</td>
</tr>
<tr>
<td>WaterFowl Mode 2</td>
<td>1.80</td>
<td>10.18</td>
<td>1710</td>
</tr>
<tr>
<td>WaterFowl Mode 3</td>
<td>1.75</td>
<td>10.13</td>
<td>1680</td>
</tr>
<tr>
<td>System</td>
<td>QR#4</td>
<td>QR#5</td>
<td>QR#6</td>
</tr>
<tr>
<td>----------------------</td>
<td>------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>4.2</td>
<td>2.5</td>
<td>15.3</td>
</tr>
<tr>
<td>OWLIM-SE</td>
<td>705</td>
<td>16771</td>
<td>72</td>
</tr>
<tr>
<td>Jena TDB</td>
<td>4.85</td>
<td>6.3</td>
<td>30.7</td>
</tr>
<tr>
<td>WaterFowl Mode 2</td>
<td>3.66</td>
<td>2.3</td>
<td>13.4</td>
</tr>
</tbody>
</table>

Table 3. Inference-based query answering times (sec) on univ100
Conclusion

- On-going work on:
  - efficient distribution of triples and processing
  - parallelism in computing the dictionaries, query processing
  - Updates at the schema and data levels.

- Extending reasoning capabilities to OWL profiles
Thank you.

Questions?