Scientific Analysis
by Queries in Extended SPARQL
over a Scalable e-Science Data Store

Andrej Andrejev, Salman Toor, Andreas Hellander*, Sverker Holmgren, Tore Risch

Department of Information Technology, Uppsala University
* Department of Computer Science, University of California Santa Barbara

andrej.andrejev@it.uu.se

Andrei Andrejev, e-Science Conference - October 2013, Beijing
• Introduction
• SciSPARQL overview
• Evaluation
• RDF views over external storage systems
• Related approaches
• Summary
Motivation

Big data needs

• *scalable* data management

• *good* documentation

• *easy access*

• *reuse* of existing software packages
Motivation

Big data needs

• *scalable* data management
  – *standard* relational database management systems,
  – specialized e-Science data stores

• good *documentation*

• easy *access*

• *reuse* of existing software packages
Motivation

Big data needs

- *scalable* data management
  - **standard** relational database management systems,
  - specialized e-Science data stores

- good *documentation*
  - **standard** W3C representation for metadata: RDF

- easy access

- *reuse* of existing software packages
Motivation

Big data needs

- **scalable** data management
  - **standard** relational database management systems,
  - specialized e-Science data stores

- **good documentation**
  - **standard** W3C representation for metadata: RDF

- **easy access**
  - **standard** W3C query language for searching RDF databases: SPARQL

- **reuse** of existing software packages
Motivation

Big data needs

• *scalable* data management
  – *standard* relational database management systems,
  – specialized e-Science data stores

• *good* documentation
  – *standard* W3C representation for metadata: RDF

• *easy access*
  – *standard* W3C query language for searching RDF databases: SPARQL

• *reuse* of existing software packages
  – calling *standard* and custom libraries from queries
Problem with RDF databases

RDF (Resource Description Framework) – a W3C standard “metadata data model”

- RDF is very suitable for describing properties about scientific experiments (metadata) but:
  - Scientific data usually involves numerical arrays
  - Arrays are represented in a very inefficient way in RDF

- **Our approach:** Extend RDF with compact numerical array representation called *Numeric Multidimensional Arrays (NMA)*
SPARQL (SPARQL Protocol and RDF Query Language) – a W3C standard language for querying RDF

- SPARQL is very suitable for searching scientific RDF-based metadata, but:
  - SPARQL has no support for queries involving array operations

- Our approach: Extend SPARQL with common array operators => SciSPARQL
Reusing program libraries

- Often need for using existing program libraries when processing experiments data, **but:**
  - SPARQL has no standard way of plugging in external program libraries and algorithms

- **Our approach:** SciSPARQL provides a general mechanism to call functions in C, Java, Python, or MATLAB
Our System Architecture

**USER**

- SciSPARQL queries
- SciSPARQL results

**SSDM**

- SciSPARQL Database Manager
- In-memory database
- Turtle reader
  - binary reader (.mat)
- Standard API (JDBC)

**MySQL, MS SQL Server, ...**

- RDF Database
- Numeric arrays

**External functions**

- Python, Java, MATLAB, ...
- engines

Andrej Andrejev, e-Science Conference - October 2013, Beijing
• Introduction
• SciSPARQL overview
• Evaluation
• RDF views over external storage systems
• Related approaches
• Summary
Basic RDF experimental metadata

EXAMPLE

:Task1 :inExperiment :Experiment1
Basic RDF experimental metadata

**EXAMPLE**

```
:Task1 :inExperiment :Experiment1

:Experiment1 :responsibleName "Andrej"
```

Andrej Andrejev, e-Science Conference - October 2013, Beijing
Basic SPARQL metadata query

**EXAMPLE**

```sparql
prefix : <http://udbl.it.uu.se/bistab#>

:Task1 :inExperiment :Experiment1 .
:Experiment1 :responsibleName "Andrej" .
```

- **RDF database of triples:**

```sparql
prefix : <http://udbl.it.uu.se/bistab#>

:Task1 :inExperiment :Experiment1 .
:Experiment1 :responsibleName "Andrej" .
```
Basic SPARQL metadata query

EXAMPLE

• RDF database of triples:

    prefix : <http://udbl.it.uu.se/bistab#>

    :Task1 :inExperiment :Experiment1 .
    :Experiment1 :responsibleName "Andrej" .

• Select all the tasks that Andrej is responsible for

    SELECT ?task
    WHERE { ?task :inExperiment ?experiment .
        ?experiment :responsibleName "Andrej" }

Andrej Andrejev, e-Science Conference - October 2013, Beijing
SSDM extends RDF with arrays

EXAMPLE

:Task1 :inExperiment :Experiment1 :result

:Experiment1 :responsibleName "Andrej"
SciSPARQL extends SPARQL with array access

**EXAMPLE**

- **One additional triple to store:**

  \[
  \text{:Task1 :result <file://task1.mat:matlab#result>} .
  \]
SciSPARQL extends SPARQL with array access

**EXAMPLE**

- One additional triple to store:
  
  :Task1 :result <file://task1.mat:matlab#result> .  
  or  
  :Task1 :result ((3 7 2 4) (8 0 1 0) ...) ...  ... .
SciSPARQL extends SPARQL with array access

EXAMPLE

- One additional triple to store:
  :Task1 :result <file://task1.mat:matlab#result> .
  or
  :Task1 :result ((3 7 2 4) (8 0 1 0) ...) ...

- Select 50-slice of "result" arrays of all tasks that Andrej is responsible for

SELECT (?result[50,:,:] AS ?slice50)
WHERE {
  ?task :result ?result ;
  :inExperiment ?experiment .
  ?experiment :responsibleName "Andrej"
}
### Experiment

<table>
<thead>
<tr>
<th>id</th>
<th>mesh</th>
<th>simulation algorithm</th>
<th># cells</th>
<th># species</th>
<th>specie ids</th>
<th>time points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>triangular #1</td>
<td>nsm</td>
<td>11107</td>
<td>4</td>
<td>0 1 2 3</td>
<td></td>
</tr>
</tbody>
</table>

0.0 0.5 1.0 ....

(sequence of 201 real numbers)

### Task

<table>
<thead>
<tr>
<th>id</th>
<th>experiment id</th>
<th>parameters</th>
<th>realization</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>k_1</td>
<td>k_a</td>
<td>k_d</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>32.159</td>
<td>79.279</td>
<td>782750669.857</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>19.151</td>
<td>39.044</td>
<td>300035857.676</td>
</tr>
</tbody>
</table>

(array of 11107 x 4 x 201 integers)
Relational Representation

### Experiment

<table>
<thead>
<tr>
<th>id</th>
<th>mesh</th>
<th>simulation algorithm</th>
<th># cells</th>
<th># species</th>
<th>specie ids</th>
<th>time points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>triangular #1</td>
<td>nsm</td>
<td>11107</td>
<td>4</td>
<td>0 1 2 3</td>
<td></td>
</tr>
</tbody>
</table>

- (sequence of 201 real numbers)

### Task

<table>
<thead>
<tr>
<th>id</th>
<th>experiment id</th>
<th>parameters</th>
<th>realization</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>k_1  k_a  k_d</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>32.159 79.279</td>
<td>53.286</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>19.151 39.044</td>
<td>73.445</td>
<td>1</td>
</tr>
</tbody>
</table>

- (array of 11107 x 4 x 201 integers)

Andrej Andrejev, e-Science Conference - October 2013, Beijing
SSDM Representation: RDF with Arrays

EXAMPLE
of an RDF database describing BISTAB scientific experiments

Andrej Andrejev, e-Science Conference - October 2013, Beijing
SSDM Representation: RDF with Arrays
More

Scientific SPARQL examples
SciSPARQL Query Language

```sparql
SELECT (AVG(?result[:,:,?s]) AS ?specAvarage)
WHERE { :Task1 :result ?result }
```

• Use free variable (?s) to generate a series of array slices
SciSPARQL Query Language

```
SELECT ?task ?s ?specAverage
WHERE { ?task :result ?result .
  BIND (AVG(?result[:,:,:,?s])) AS ?specAverage .
  FILTER (?specAverage > 5) }
```

- Filter data selection based on derived values
• Introduction
• SciSPARQL overview
• Evaluation
• RDF views over external storage systems
• Related approaches
• Summary
Our Contribution

SSDM shows performance on par with MATLAB, with added value of MATLAB SciSPARQL

<table>
<thead>
<tr>
<th>MATLAB</th>
<th>SciSPARQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programs implementing analysis algorithms</td>
<td>High-level queries</td>
</tr>
<tr>
<td>No metadata management user manually manages files</td>
<td>Uniform management of both data and metadata</td>
</tr>
</tbody>
</table>
SSDM shows performance on par with MATLAB, with added value of

<table>
<thead>
<tr>
<th>MATLAB</th>
<th>SciSPARQL Q2</th>
</tr>
</thead>
</table>
| \[ \text{sum}_\text{of}_A = []; \] \\
| \text{load('input.mat');} \& \& \text{parameters, tspan 'metadata'} \] \\
| \( t = \text{find(tspan==10)}; \) \\
| \( a = 1; \) \& \& \text{'this 'metadata' is not stored anywhere} \] \\
| \( \text{mspecies} = 8; \) \\
| \text{for ii=1:100 \& \& amount of files should be known!} \] \\
| \text{if parameters(1,ii) >= 50} \] \\
| \& \& \text{parameters(1,ii) <= 90} \] \\
| \& \& \text{parameters(3,ii) >= 1.0E8} \] \\
| \& \& \text{parameters(3,ii) <= 1.0E9} \] \\
| \text{realization = strcat('C:/DATA/bistab2f/realization_',int2str(ii),'_1.mat');} \] \\
| \text{load(realization);} \& \& \text{load matrices 1-by-1} \] \\
| \( \text{sum}_\text{of}_A = [\text{sum}_\text{of}_A \] \\
| \text{sum(UU(a:mspecies:end, t))];} \] \\
| \text{end} \] \\
| \text{end} \] \\
| \text{sum}_\text{of}_A; \] \\
| | \text{SELECT (array_sum(?U[a-1::?mspecies,?j]) AS ?res)} \] \\
| | \text{WHERE} { ?task :U ?U ; # retrieve data} \] \\
| | \( :k_a ?k_a ; # \text{retrieve metadata} \] \\
| | \( :k_d ?k_d ; \) \\
| | \( :\text{inExperiment} ?experiment . \) \\
| | \text{?experiment :A ?a ;} \] \\
| | \text{:MSpecies ?mspecies ;} \\
| | \text{:tspan ?tspan .} \\
| | \text{FILTER (?tspan[?j] = 10 \& \& \text{1.0E8 <= ?k_d \& \& ?k_d <= 1.0E9 \& \& 50 <= ?k_a \& \& ?k_a <= 90 })}; \]
SSDM shows performance on par with MATLAB, with added value of

```matlab
sum_of_A = [];
load('input.mat');  % parameters, tspan 'metadata'
t = find(tspan==10);
a = 1;  % this 'metadata' is not stored anywhere
mspecies = 8;
for ii=1:100  % amount of files should be known!
    if parameters(1,ii) >= 50
        && parameters(1,ii) <= 90
        && parameters(3,ii) >= 1.0E8
        && parameters(3,ii) <= 1.0E9
        realization = strcat('C:/DATA/bistab2f/realization_',int2str(ii),'_1.mat');
    end
    load(realization);  % load matrices 1-by-1
    sum_of_A = [sum_of_A
                sum(UU(a:mspecies:end, t))];
end
sum_of_A;
```
Our Contribution

SSDM shows performance on par with MATLAB, with added value of

```
SELECT (array_sum(?U[?a-1::?mspecies,?j]) AS ?res)
WHERE { ?task :U ?U ; # retrieve data
    :k_a ?k_a ; # retrieve metadata
    :k_d ?k_d ;
    :inExperiment ?experiment .
?experiment :A ?a ;
    :MSpecies ?mspecies ;
    :tspan ?tspan .
FILTER (?tspan[?j] = 10 &&
    1.0E8 <= ?k_d && ?k_d <= 1.0E9 &&
    50 <= ?k_a && ?k_a <= 90 ) };
```
## SSDM Performance

7GB database, query execution times (in seconds) with all data on disk

<table>
<thead>
<tr>
<th>Task</th>
<th>Data retrieved</th>
<th>SSDM with back-end</th>
<th>MATLAB script</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>MySQL</td>
<td>MS SQL Server</td>
</tr>
<tr>
<td>Q1: (selective query)</td>
<td>18MB</td>
<td>1.748</td>
<td>2.15</td>
</tr>
<tr>
<td>Compute an aggregate value over 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>big matrix, every 8th row</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q2: (SSDM worst case)</td>
<td>642MB</td>
<td>80.703</td>
<td>44.512</td>
</tr>
<tr>
<td>Select 36 matrices, access one</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>column × every 8th row</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q3: (database scan)</td>
<td>1785MB</td>
<td>187.073</td>
<td>192.365</td>
</tr>
<tr>
<td>Compute AGRMAX of Q1 across all</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrices, 25% rows</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

=> SSDM provides desired functionality with competitive performance
• Introduction
• SciSPARQL overview
• Evaluation
  • RDF views over external storage systems
  • Related approaches
  • Summary

Andrej Andrejev, e-Science Conference - October 2013, Beijing
RDF views over external storage systems

USER

SciSPARQL queries → SSDM Kernel

SciSPARQL results ← SSDM Kernel

SSDM Kernel

In-memory database

WRAPPERS

Chelonia RDF View

DATA SOURCE

Chelonia

Variable catalog

Numeric arrays

Andrej Andrejev, e-Science Conference - October 2013, Beijing
**Chelonia Native Schema**

<table>
<thead>
<tr>
<th>task id</th>
<th>var</th>
<th>k_1</th>
<th>k_a</th>
<th>k_d</th>
<th>k_4</th>
<th>realization</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>k_1</td>
<td>32.159</td>
<td>79.279</td>
<td>782750669.857</td>
<td>53.286</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>k_2</td>
<td>19.151</td>
<td>39.044</td>
<td>300035857.676</td>
<td>73.445</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
RDF views over external storage systems

USER

SciSPARQL queries

SciSPARQL results

SSDM Kernel

In-memory database

WRAPPERS

Relational to RDF View*

DATA SOURCE

Relational DB

RDF views over external storage systems

USER

SciSPARQL queries
SciSPARQL results

SSDM Kernel
In-memory database

 DATA SOURCE

.mat files

WRAPPERS

.mat reader

Andrej Andrejev, e-Science Conference - October 2013, Beijing
RDF views over external storage systems

USER

SciSPARQL queries

SciSPARQL results

SSDM Kernel

In-memory database

WRAPPER

... wrapper

DATA SOURCE

...DB

Andrej Andrejev, e-Science Conference - October 2013, Beijing
• Introduction
• SciSPARQL overview
• Evaluation
• RDF views over external storage systems
• Related approaches
• Summary
## Related approaches

<table>
<thead>
<tr>
<th>Databases</th>
<th>RDF</th>
<th>Files and programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>• High-level metadata descriptions (schemas)</td>
<td>• Designed for metadata in general</td>
<td>• No explicit metadata</td>
</tr>
<tr>
<td>• Scalable data representation</td>
<td>• Voluntary schema</td>
<td>• Many storage formats and APIs</td>
</tr>
<tr>
<td>• High-level query languages</td>
<td>• Weak support numeric applications</td>
<td>• Numerical libraries maintained since 1960:s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Extensively used in scientific computing</td>
</tr>
</tbody>
</table>

Andrej Andrejev, e-Science Conference - October 2013, Beijing
## Related approaches

<table>
<thead>
<tr>
<th>Databases</th>
<th>RDF</th>
<th>Files and programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>- High-level metadata descriptions (schemas)</td>
<td>- Designed for metadata in general</td>
<td>- No explicit metadata</td>
</tr>
<tr>
<td>- Scalable data representation</td>
<td>- Voluntary schema</td>
<td>- Many storage formats and APIs</td>
</tr>
<tr>
<td>- High-level query languages</td>
<td>- Weak support numeric applications</td>
<td>- Numerical libraries maintained since 1960:s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Extensively used in scientific computing</td>
</tr>
</tbody>
</table>

### SSDM and SciSPARQL

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Full database support</td>
<td>- Flexibility of RDF</td>
<td>- Reuse of existing libraries</td>
</tr>
</tbody>
</table>
Related query languages

Extending RDBMS with array semantics
  • AQuery [Lerner & Shasha, 2003]
  • SciQL [Kersten et.al, 2011]
  • Storing arrays as BLOBs
    • RasQL [Furtado & Baumann, 1999]
    • UDFs in T-SQL [Dobos et.al., 2011]
  • Specialized array databases and languages
    • AQL [Libkin et.al. 1996]
    • SciDB [Cudre-Mauroux et.al. 2009]
  • Foreign Functions in SPARQL
    • SESAME
    • CORESE
• Introduction
• SciSPARQL overview
• Evaluation
• RDF views over external storage systems
• Related approaches
• Summary
SSDM (SciSPARQL Database Manager) provides

- Efficient storage of RDF with arrays
- Back-end relational database storage and various data file formats
- Access to external databases

SciSPARQL provides

- support of numeric multidimensional arrays and operations
- extensibility with foreign functions in C, Java, Python, and MATLAB
The software, documentation, and examples are available at

http://www.it.uu.se/research/group/udbl/SciSPARQL

This work was supported by

esence
THE E-SCIENCE COLLABORATION