Survey of RDF Storage Managers

Kiyoshi Nitta
Yahoo JAPAN Research
Tokyo, Japan
knitta@yahoo-corp.jp

Iztok Savnik
University of Primorska &
Institute Jozef Stefan, Slovenia
iztok.savnik@upr.si

The First International Workshop on Large-scale Graph Storage and Management (GraphSM 2014), April 22, 2014 - Chamonix, France.
Survey of RDF Storage Managers

Kiyoshi Nitta
Yahoo JAPAN Research
Tokyo, Japan
knitta@yahoo-corp.jp

Iztok Savnik
University of Primorska &
Institute Jozef Stefan, Slovenia
iztok.savnik@upr.si

RDF STORAGE MANAGERS

Subject

Predicate

Object

The First International Workshop on Large-scale Graph Storage and Management (GraphSM 2014), April 22, 2014 - Chamomix, France.
INTRODUCTION

Resource description framework (RDF) data are widely used in the Internet and their volume is growing steadily. The linked open data (LOD) project promotes the acceleration of the accumulation of RDF data to provide freely accessible on-line resources.

CLASSIFICATION OF RDF STORAGE MANAGERS

RDF storage managers in the local cache approach can be classified in accordance with several aspects.

RSM(S, M)

PROPERTIES OF RDF STORAGE MANAGERS

<table>
<thead>
<tr>
<th></th>
<th>T_e</th>
<th>I_e</th>
<th>Q_e</th>
<th>S_e</th>
<th>J_e</th>
<th>C_e</th>
<th>D_e</th>
<th>F_e</th>
<th>D_m</th>
<th>Q_m</th>
<th>S_m</th>
<th>A_m</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td>S</td>
<td>U</td>
<td>R</td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td>h</td>
<td>p</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td>R</td>
<td>T</td>
<td>A</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hexasto</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>U</td>
<td>l</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td>r</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td>U</td>
<td>l</td>
<td>o</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td>n</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>U</td>
<td>p</td>
<td>c</td>
<td>n</td>
<td>n</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AllegroGraph</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>U</td>
<td>p</td>
<td>c</td>
<td>h</td>
<td>p</td>
<td>m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hadoop/HiBase</td>
<td>h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

attributes of single process issues

attributes of multiple process issues

\[ S(T_e, I_e, Q_e, S_e, J_e, C_e, D_e, F_e) \]

\[ M(D_m, Q_m, S_m, A_m) \]
# RDF STORAGE MANAGERS

## 3store
- **Tx**: vertical, **Qs**: SPARQL, **Ss**: string id, **Jx**: conventional ordering, **Rims**: blank, **Rv**: RDB, **Tv**: Prow, **Dm**: none, **Qm**: none, **Sm**: none.
- It stored 500 classes and about 20 million triples in originally used semantic web applications.
- Store was implemented on top of the MySQL database management system.

## Virtuoso
- **Tx**: vertical, **Is**: SPARQL, **Qs**: SPARQL, **Ss**: string id, **Jx**: conventional ordering, **Rims**: blank, **Rv**: RDB, **Tv**: Neo4j, **Dm**: none, **Qm**: none, **Sm**: none.
- The most important aspects: extending SQL types with the RDF data type, dealing with unpredictable size of objects, providing efficient indexing and extending relational statistics.

## Hexastore
- **Tx**: horizontal, **Is**: string id, **Jx**: column store based, **Qs**: original customized, **Ss**: string id, **Jx**: column store based, **Rims**: blank, **Rv**: RDB, **Tv**: TESCA/Box, **Dm**: none, **Qm**: none, **Sm**: none.
- The proposed index provides natural representation of multi-valued properties and allows fine implementation of merge-join, intersection, and union.

## SW-Store
- **Tx**: horizontal, **Is**: string id, **Jx**: column store based, **Qs**: materialized path index, **Ss**: string id, **Jx**: column store based, **Rims**: blank, **Rv**: RDB, **Tv**: None, **Dm**: none, **Qm**: none, **Sm**: none.
- Storage systems based on columns can significantly improve some types of queries on RDF databases.

## AllegroGraph
- **Qs**: SPARQL, **Dm**: custom, **Rv**: hash, **Qm**: data parallel, **Am**: memory.
- It is a proprietary product of Franz Inc.
- A distributed version of AllegroGraph stored 1 trillion triples in about 15 days.
- In the scalability ranking of triple stores edited by VSC, AllegroGraph is ranked first as of August 2011.

## 4store
- **Tx**: vertical, **Qs**: SPARQL, **Ss**: string id, **Jx**: conventional ordering, **Rims**: blank, **Rv**: RDB, **Tv**: none, **Dm**: hash, **Qm**: data parallel, **Am**: nothing.
- The requirements were to store and manage 10B RDF triples.
- The design of 4store is based on 3store especially in the way RDF triples are represented.

## RDF-3X
- **Tx**: vertical, **Is**: string id, **Qs**: SPARQL, **Ss**: string id, **Jx**: conventional ordering, **Rims**: blank, **Rv**: RDB, **Tv**: none, **Dm**: hash, **Qm**: none, **Sm**: none.
- It built six independent indexes of SPO, SAI, OPI, OSP, PNO, and PNS.
- The optimization also selectivity statistics.

## Apache Jena
- **Tx**: property, **Qs**: SPARQL, **Ss**: string id, **Jx**: conventional ordering, **Rims**: blank, **Rv**: RDB, **Tv**: none, **Dm**: hash, **Qm**: none.
- Jena is a database programming language environment based on RDF for Java.
- Jena supports MySQL, PostgreSQL, Oracle, and BerkeleyDB.

## BitMat
- **Tx**: horizontal, **Is**: matrix, **Qs**: SPARQL, **Ss**: string id, **Jx**: printing, **Dm**: custom, **Rv**: storage pipeline.
- It used a three-dimensional compressed matrix index named BitMat.
- It avoids maintaining materialized triples as much as possible.

## Hadoop/HBase
- **Tx**: horizontal, **Is**: custom, **Qm**: data parallel, **Am**: memory.
- HBase is column-oriented database system implemented on top of Hadoop.
- HBase is designed for horizontal distribution of tables into regions that are managed by one server. Map-reduce techniques can be employed to process table rows.
CHALLENGES

- More varied values with 5 properties than with 10 attributes
  - Research has so far succeeded in achieving good performance by developing single process systems.
  - Multithreading and parallel processing are still in the experimental stage.
- Caching techniques have not been researched that much
  - Only Apache Jena and SciVerse reported overcoming the efficiency of caching techniques.
  - Few techniques for automatically extracted and classification of processing queries are available.
- Many researches have been working on developing efficient SPARQL algorithms with index structures
  - SPARQL queries are expensive and may cause performance issues due to the existence of XML objects in most RDF repositories.

CONCLUSION

- Surveyed the RDF storage manager implementations based on the local cache approach by introducing the systematic classification structure RSM(S,M).
- This classification was applied to 3store, 4store, Virtuoso, RDF-3X, Hexastore, Apache Jena, SW-Store, BitMat, AllegroGraph, and Hadoop/HBase. The list will be expanded.
- There will be room for further improvement of the efficient query process by developing multi process technologies.
INTRODUCTION

Resource description framework (RDF) data are widely used in the Internet and their volume is growing steadily. The linked open data (LOD) project promotes the acceleration of the accumulation of RDF data to provide freely accessible on-line resources.

![Map showing 40 billion triples](image)

(D-1) Local Cache Approach
- Prefers accessing RDF data from local resources

(D-2) Federated Search Approach
- Fetches data from federated services

**plays an important role for query process efficiency**
accessible on-line resources

40 billion triples

(A-1) Local Cache Approach
(A-1) Local Cache Approach

gather a subset of RDF data on local computational resources
(A-2) Federated Search Approach

distribute sub-queries to several search services distributed over the Internet
(A-1) Local Cache Approach

gather a subset of RDF data on local computational resources

(A-2) Federated Search Approach

distribute sub-queries to several search services distributed over the Internet

plays an important role for query process efficiency
CLASSIFICATION OF RDF STORAGE MANAGERS

RDF storage managers in the local cache approach can be classified in accordance with several aspects.

\[ RSM(S, M) \]

PROPERTIES OF RDF STORAGE MANAGERS

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td>S</td>
<td>U</td>
<td>R</td>
<td>R</td>
<td>T</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>R</td>
<td>TA</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td>R</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td>Uo</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td>c</td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td>m</td>
<td></td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td></td>
<td>S</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>m</td>
</tr>
</tbody>
</table>

The table above classifies RDF storage managers based on various properties.
$S(T_s, I_s, Q_s, S_s, J_s, C_s, D_s, F_s)$

attributes of single process issues
$T_s$  triple table type (vertical | property | horizontal)

$I_s$  index structure type (6-independent | GSPO-OGPS | matrix)

$Q_s$  query type (SPARQL | original)

$S_s$  translation method type of IRI and literal strings (URI | literal | long | none)

$J_s$  join optimization method type (RDBMS-based | column-store-based | conventional-ordering | pruning | none)

$C_s$  cache type (materialized-path-index | reified-statement (r) | none)

$D_s$  database engine type (RDB | custom)

$F_s$  inference feature type (TBox, ABox, and no)
attributes of multiple process issues

\[ M(D_m, Q_m, S_m, A_m) \]
$D_m$ data distribution method type (hash | data-source | none)

$Q_m$ query process distribution method type (data-parallel | data-replication | none)

$S_m$ stream process type (pipeline | none)

$A_m$ resource sharing architecture type (memory | disk | nothing)
### Properties of RDF Storage Managers

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>R</td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td>n</td>
<td>h</td>
<td>p</td>
<td>n</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>R</td>
<td>TA</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td>R</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td></td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td></td>
<td>Uo</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td>c</td>
<td>c</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td>m</td>
<td></td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td>h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
<td></td>
<td>m</td>
<td></td>
</tr>
</tbody>
</table>
RDF STORAGE MANAGERS

3store
- Tx = vertical, Qs = SPARQL, Ss = string id, Js = RDFS based, Ds = RDB, Fs = TriStore
- It stored 5000 classes and about 20 million triples in originally used semantic web applications.
- It was implemented on top of the MySQL database management system.

Virtuoso
- Tx = vertical, Js = SPARQL, Qs = SPARQL, Ss = string id, Js = RDFS based, Ds = RDB, Fs = TriStore
- It has a wide range of features that make it suitable for various applications.

Hexastore
- Tx = horizontal, Js = string id, Js = DB2, Ds = RDBMS based, Fs = TriStore
- It has a high performance and scalability for large datasets.

SW-Store
- Ts = horizontal, Js = string id, Js = DB2, Ds = RDBMS based, Fs = TriStore
- It has a high performance and scalability for large datasets.

AllegroGraph
- Ts = horizontal, Js = custom, Ds = hash, Ds = data parallel, Fs = memory
- It is a proprietary product of Franz Inc.
- A distributed version of AllegroGraph stored 1 trillion triples in about 30 days.
- In the scalability ranking of triple stores edited by W3C, AllegroGraph is ranked first as of August 2011.

4store
- Tx = vertical, Qs = SPARQL, Ss = string id, Js = conventional ordering, Ds = RDB, Fs = TriStore
- The requirements were to store and manage RDF triples.
- The design of 4store is based on 3store especially in the way RDF triples are represented.

RDF-3X
- Tx = vertical, Js = SPARQL, Ss = string id, Js = conventional ordering, Ds = RDB, Fs = TriStore
- It has six independent indexes of SPO, APO, OSP, OPO, PPO, and PPO.
- The optimization also generates statistics.

Apache Jena
- Tx = vertical, Qs = SPARQL, Ss = string id, Js = RDFS based, Ds = RDB, Fs = RDB, Fs = RDB
- It supports SPARQL and RDFS.
- Jena is a database programming language environment based on RDF for Java.
- Jena supports MySQL, PostgreSQL, Oracle, and BerkeleyDB.

BitMat
- Ts = horizontal, Js = matrix, Qs = SPARQL, Ss = string id, Js = printing, Ds = custom, Fs = pipeline
- It can be decomposed into a three-dimensional compressed matrix named BitMat.
- It avoids maintaining materialized triples as much as possible.

Hadoop/HBase
- Ts = horizontal, Js = custom, Ds = data parallel, Fs = memory
- HBase is a column-oriented database system implemented on top of Hadoop.
- HBase is designed for distributed storage of tables into regions that are managed by one server. Map-reduce techniques can be employed to process table rows.
3store

- $T_s =$ vertical, $Q_s =$ SPARQL, $S_s =$ string id, $J_s =$ RDBMS based, $D_s =$ RDB, $F_s =$ TBox, $D_m =$ none, $Q_m =$ none, $S_m =$ none.
- It stored 5,000 classes and about 20 million triples in originally used semantic web applications.
- 3store was implemented on top of the MySQL database management system.
4store

- $T_s =$ vertical, $Q_s =$ SPARQL, $S_s =$ string id, $J_s =$ conventional ordering, $D_s =$ RDB, $D_m =$ hash, $Q_m =$ data parallel, $A_m =$ nothing.
- The requirements were to store and manage 15x109 triples.
- The design of 4store is based on 3store especially in the way RDF triples are represented.
Virtuoso

- $T_s = \text{vertical, } I_s = \text{GSPO OGPS, } Q_s = \text{SPARQL, } S_s = \text{string id, } J_s = \text{RDBMS based, } D_s = \text{RDB, } F_s = \text{TBox, ABox, } D_m = \text{none, } Q_m = \text{none, } S_m = \text{none.}$
- The most important aspects: extending SQL types with the RDF data type, dealing with unpredictable sizes of objects, providing efficient indexing and extending relational statistics.
RDF-3X

- $Ts = \text{vertical}$, $Is = 6 \text{ independent}$, $Qs = \text{SPARQL}$, $Ss = \text{string id}$, $Js = \text{conventional ordering}$, $Ds = \text{RDB}$, $Dm = \text{none}$, $Qm = \text{none}$, $Sm = \text{none}$.
- It built six independent indexes of SPO, SOP, OSP, OPS, PSO and POS.
- The optimization uses selectivity statistics.
Hexastore

- $Ts = \text{vertical, } Is = 6 \text{ independent, } Qs = \text{original(customwrapped), } Ss = \text{string id, } Js = \text{unknown(itseemsnone), } Cs = \text{none, } Dm = \text{none, } Qm = \text{none, } Sm = \text{none.}$
- The proposed index provides natural representation of multi-valued properties and allows fast implementation of merge-join, intersection, and union.
Apache Jena

- $Ts = \text{property}, \: Qs = \text{SPARQL}, \: Ss = \text{string id}, \: Js = \text{RDBMS based}, \: Cs = \text{reified statement}, \: Ds = \text{RDB}, \: Dm = \text{none}, \: Qm = \text{none}, \: Sm = \text{none}.$
- Jena is a database programming language environment based on RDF for Java.
- Jena supports MySQL, Postgres, Oracle, and BerkeleyDB.
SW-Store

- Ts = horizontal, Ss = string id, Js = column store based, Cs = materialized path index, Ds = custom, Dm = none, Qm = none, Sm = none.
- Storage systems based on columns can significantly improve some types of queries on RDF databases.
BitMat

- Ts = vertical, Is = matrix, Qs = SPARQL, Ss = string id, Js = pruning, Ds = custom, Sm = pipeline.
- It used a three dimensional compressed matrix index named BitMat.
- It avoids maintaining materialized triples as much as possible.
AllegroGraph

- $Q_s =$ SPARQL, $D_s =$ custom, $D_m =$ hash, $Q_m =$ data parallel, $A_m =$ memory.
- It is a proprietary products of Franz Inc.
- A clustered version of AllegroGraph stored 1 trillion triples in about 14 days.
- In the scalability ranking of triple stores edited by W3C, AllegroGraph is ranked first as of August 2011.
Hadoop/HBase

- Ts = horizontal, Ds = custom, Qm = data parallel, Am = memory.
- HBase is column-oriented database system implemented on top of Hadoop.
- HBase is designed for horizontal distribution of tables into regions that are managed by one server. Map-reduce techniques can be employed to process table rows.
RDF STORAGE MANAGERS

**3store**
- **Ts**: vertical, **Qs**: string id, **Js**: SPARQL, **Es**: string id, **Ds**: RMMS-based, **Rms**: RDBMS-based, **Rs**: RDBMS-based, **Fs**: ne, **Dm**: name, **Qm**: none, **Sm**: none.
- **It stores 5,000 classes and about 10 million triples in originally used semantic web applications.**
- **It was implemented on top of the MySQL database management system.**

**Virtuoso**
- **Ts**: vertical, **Is**: GeoSPARQL, **Qs**: SPARQL, **Js**: string id, **Es**: string id, **Ds**: RDFMS-based, **Rds**: RDFMS-based, **Rs**: TBoxABox, **Dm**: none, **Qm**: none, **Sm**: none.
- **The most important aspects: extending SQL types with the RDF data type, dealing with unpredictable size of objects, providing efficient indexing and extending relational statistics.**

**Hexastore**
- **Ts**: horizontal, **Is**: string id, **Js**: string id, **Es**: string id, **Ds**: SPARQL, **Rds**: SPARQL, **Rs**: string id, **Dm**: none, **Qm**: none, **Sm**: none.
- **The proposed index provides natural representation of multi-valued properties and allows fine-tuned implementation of merge-join, intersection, and union.**

**SW-Store**
- **Ts**: horizontal, **Is**: string id, **Js**: string id, **Es**: string id, **Ds**: column store based, **Rds**: materialized path index, **Rs**: none, **Dm**: custom, **Qm**: none, **Sm**: none.
- **Storage systems based on columns can significantly improve some types of queries on RDF databases.**

**AllegroGraph**
- **Ts**: SPARQL, **Is**: custom, **Js**: hash, **Es**: custom, **Ds**: hash, **Rds**: hash, **Rs**: hash.
- **It is a proprietary product of Franz Inc.**
- **A distributed version of AllegroGraph store 1 trillion triples in about 15 days.**
- **In the scalability ranking of triple stores edited by W3C, AllegroGraph is ranked first as of August 2011.**

**4store**
- **Ts**: vertical, **Qs**: SPARQL, **Js**: string id, **Es**: string id, **Ds**: hash, **Dm**: data parallel, **Am**: nothing.
- **The requirements were to store and manage 100GB of triples.**
- **The design of 4store is based on Store especially in the way RDF triples are represented.**

**RDF-3X**
- **Ts**: vertical, **Is**: 6 independent, **Qs**: SPARQL, **Js**: string id, **Es**: string id, **Ds**: conventional ordering, **Dm**: RDBMS-based, **Ds**: none, **Qm**: none, **Sm**: none.
- **It built six independent indexes of SPO, AOP, ODP, PEO, and PBO.**
- **The optimization also selects selective statistics.**

**Apache Jena**
- **Ts**: property, **Qs**: SPARQL, **Js**: string id, **Es**: string id, **Ds**: RDBMS-based, **Rds**: SPARQL-based, **Rs**: SPARQL-based, **Dm**: string id, **Qm**: none, **Sm**: none.
- **Jena is a database programming language environment based on RDF for Java.**
- **Jena supports MySQL, PostgreSQL, Oracle, and BerkeleyDB.**

**BitMat**
- **Ts**: horizontal, **Is**: matrix, **Qs**: SPARQL, **Js**: string id, **Es**: matrix, **Ds**: custom, **Dm**: pipeline.
- **It used a three-dimensional compressed matrix index named BitMat.**
- **It avoids maintaining materialized triples as much as possible.**

**Hadoop/HBase**
- **Ts**: horizontal, **Is**: custom, **Qs**: data parallel, **Ds**: memory.
- **Hbase is a column-oriented database system implemented on top of Hadoop.**
- **Hbase is designed for horizontal distribution of tables into regions that are managed by one server. Map-reduce techniques can be employed to process table rows.**
CHALLENGES

More varied values with S properties than with M attributes
- Researches so far have succeeded in achieving good performances by developing single process technologies.
- While practical semantic web applications tend to process large-scale data sets, solutions based on data distribution parallelism have become more popular.

Caching techniques have not been researched that much
- Only Apache Jena and SW-Store reported confirming the efficiency of caching techniques.
- Technologies for automatic investigation and classification of processing queries might become important to utilize caching technologies.

Many researches have been working on developing efficient join algorithms with index structures
- This area has a long history in the research of database management systems.
- While the accumulated RDF data-set is rapidly growing and SPARQL queries are basically constructed from joins of triple patterns, join operations will be applied more strongly in semantic web applications.

Most RDF storage managers can accept SPARQL queries
- SPARQL-based RDF storage managers rarely cause semantic mismatch due to the existence of RDF algebras described in the W3C recommendation.
- While OPTIONAL operator was introduced to make the query language convenient enough, efficient processing of such queries will be one of the most crucial challenges.
More varied values with S properties than with M attributes

- Researches so far have succeeded in achieving good performances by developing single process technologies.
- While practical semantic web applications tend to process large-scale data sets, solutions based on data distribution parallelism have become more popular.
# Properties of RDF Storage Managers

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>R</td>
<td></td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>o</td>
<td></td>
<td>R</td>
<td></td>
<td>h</td>
<td>p</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td></td>
<td>R</td>
<td>TA</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td></td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td></td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td></td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td></td>
<td>Uo</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td>m</td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td>h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td>p</td>
<td></td>
<td>m</td>
</tr>
</tbody>
</table>
More varied values with S properties than with M attributes

- Researches so far have succeeded in achieving good performances by developing single process technologies.
- While practical semantic web applications tend to process large-scale data sets, solutions based on data distribution parallelism have become more popular.
Caching techniques have not been researched that much

- Only Apache Jena and SW-Store reported confirming the efficiency of caching techniques.
- Technologies for automatic investigation and classification of processing queries might become important to utilize caching technologies.
## Properties of RDF Storage Managers

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>R</td>
<td></td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>o</td>
<td></td>
<td>R</td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td></td>
<td>R</td>
<td>TA</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td></td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td></td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td></td>
<td>Uo</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td>p</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td>h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
<td></td>
<td>m</td>
</tr>
</tbody>
</table>
Caching techniques have not been researched that much

- Only Apache Jena and SW-Store reported confirming the efficiency of caching techniques.
- Technologies for automatic investigation and classification of processing queries might become important to utilize caching technologies.
Many researches have been working on developing efficient join algorithms with index structures.

- This area has a long history in the research of database management systems.
- While the accumulated RDF data-set is rapidly growing and SPARQL queries are basically constructed from joins of triple patterns, join operations will be applied more strongly in semantic web applications.
## Properties of RDF Storage Managers

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>R</td>
<td></td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>o</td>
<td></td>
<td>R</td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td>n</td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td></td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td></td>
<td>TA</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td></td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td></td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td></td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td></td>
<td>Ulo</td>
<td></td>
<td>c</td>
<td>m</td>
<td>c</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td></td>
<td></td>
<td>m</td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td>h</td>
<td></td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td>p</td>
<td></td>
<td></td>
<td>m</td>
</tr>
</tbody>
</table>
Many researches have been working on developing efficient join algorithms with index structures.

- This area has a long history in the research of database management systems.
- While the accumulated RDF data-set is rapidly growing and SPARQL queries are basically constructed from joins of triple patterns, join operations will be applied more strongly in semantic web applications.
Most RDF storage managers can accept SPARQL queries

- SPARQL-based RDF storage managers rarely cause semantic mismatch due to the existence of RDF algebras described in the W3C recommendation.
- While OPTIONAL operator was introduced to make the query language convenient enough, efficient processing of such queries will be one of the most crucial challenges.
# PROPERTIES OF RDF STORAGE MANAGERS

<table>
<thead>
<tr>
<th></th>
<th>$T_s$</th>
<th>$I_s$</th>
<th>$Q_s$</th>
<th>$S_s$</th>
<th>$J_s$</th>
<th>$C_s$</th>
<th>$D_s$</th>
<th>$F_s$</th>
<th>$D_m$</th>
<th>$Q_m$</th>
<th>$S_m$</th>
<th>$A_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>R</td>
<td></td>
<td>R</td>
<td>T</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>4store</td>
<td>v</td>
<td></td>
<td>S</td>
<td>U</td>
<td>o</td>
<td>R</td>
<td></td>
<td></td>
<td>h</td>
<td>p</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Virtuoso</td>
<td>v</td>
<td>G</td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td></td>
<td>R</td>
<td>TA</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>RDF-3X</td>
<td>v</td>
<td>6</td>
<td>S</td>
<td>Ul</td>
<td>o</td>
<td>R</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Hexastore</td>
<td>v</td>
<td>6</td>
<td>o</td>
<td>Ul</td>
<td></td>
<td>n</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Apache Jena</td>
<td>p</td>
<td></td>
<td>S</td>
<td>Ulo</td>
<td>R</td>
<td>r</td>
<td>R</td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>SW-Store</td>
<td>h</td>
<td></td>
<td>Uo</td>
<td>c</td>
<td>m</td>
<td>c</td>
<td></td>
<td></td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>BitMat</td>
<td>v</td>
<td>m</td>
<td>S</td>
<td>Ul</td>
<td>p</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
</tr>
<tr>
<td>AllegroGraph</td>
<td></td>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>h</td>
</tr>
<tr>
<td>Hadoop/HBase</td>
<td>h</td>
<td></td>
<td></td>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
</tr>
</tbody>
</table>
Most RDF storage managers can accept SPARQL queries

- SPARQL-based RDF storage managers rarely cause semantic mismatch due to the existence of RDF algebras described in the W3C recommendation.
- While OPTIONAL operator was introduced to make the query language convenient enough, efficient processing of such queries will be one of the most crucial challenges.
CONCLUSION

• Surveyed the RDF storage manager implementations based on the local cache approach by introducing the systematic classification structure RSM(S,M).
• This classification was applied to 3store, 4store, Virtuoso, RDF-3X, Hexastore, Apache Jena, SW-Store, BitMat, AllegroGraph, and Hadoop/HBase. The list will be expanded.
• There will be room for further improvement of the efficient query process by developing multi process technologies.