Self-Organization in Distributed Semantic Storage System

Anne Augustin, Sebastian Koske, Marko Harasic, Hannes Mühleisen, Philipp Obermeier, Kia Teymourian, Robert Tolksdorf
{firstname.lastname}@inf.fu-berlin.de

Presenter: Kia Teymourian
Freie Universität Berlin
Department of Mathematics and Computer Science
Networked Information Systems
http://digipolis.ag-nbi.de
Motivation

- Semantic Storage Service
  - General Requirements
    - *High Scalability*,
    - Reliability,
    - Resilience,
    - Inexpensiveness,
    - Simplicity,
    - Security,
    - etc ...
  - Semantic requirements
    - Reasoning
    - Optimized Metadata Storage

http://digipolis.ag-nbi.de
Storage Service

Client

Storage
## State of the Art

<table>
<thead>
<tr>
<th>Store</th>
<th>Description</th>
<th>#Triples</th>
</tr>
</thead>
<tbody>
<tr>
<td>centralised:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtuoso</td>
<td>No reasoning, proprietary</td>
<td>1.0 B [9]</td>
</tr>
<tr>
<td>AllegroGraph</td>
<td>RDFS supported, proprietary</td>
<td>1.1 B [16]</td>
</tr>
<tr>
<td>Jena TDB</td>
<td>No reasoning, open source</td>
<td>1.7 B [31]</td>
</tr>
<tr>
<td>OWLIM</td>
<td>RDFS / OWL-Lite supported, proprietary</td>
<td>12 B [26]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Store</th>
<th>Overlay Network</th>
<th>Routing Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>distributed:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edutella</td>
<td>Flooding</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>S-RDF</td>
<td>Restricted Flooding</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>RDFPeers</td>
<td>CHORD DHT</td>
<td>$O(\log(N))$</td>
</tr>
<tr>
<td>YARS2</td>
<td>unspecified DHT</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>GridVine</td>
<td>P-Grid</td>
<td>$O(\log(N))$</td>
</tr>
</tbody>
</table>

- Existing Distributed Storage Systems have some difficulties to scale out
- Billion triple challenge
  - DBpedia consists about 0.5 billion RDF triples
  - In future, billions, trillions, quadrillions, etc ...

http://digipolis.ag-nbi.de
The **Self-Organized Semantic Storage Service (S4)**
Swarms:
- Simplicity
- Dynamism
- Locality

http://digipolis.ag-nbi.de
Idea

- Operations are represented by ants
- Nodes can be ant colonies, ant hills
- Ants carry data chunks/queries

Data are considered as brood or food
Ants are collectively able to find the shortest path between a food source and the nest.
2. Ant Brood Sorting

Algorithms
• RDF (Resource Description Framework)
  - Language to represent knowledge in a formal way
  - RDF-statements (RDF-triples) are primitive which build a directed Graph

• RDF-triple $<\text{Subject}, \text{Predicate}, \text{Object}>$
  - Resource (Subject) has a property (Predicate) with a value (Object)
  - S and P have to be URIs; O can be an URI or a literal (e.g. integer)

http://www.w3.org/1999/02/22-rdf-syntax-ns#type

http://birds.org/description/onto.rdf#sparrow

• Basic Triple Pattern e.g. $<?, ?, O>$
  - Primitive for lookup operations
  - Finds all triples with same value in the field (e.g. all triples with O as object)
Write Triples

- Create three ants, one for each triple (S,P,O)
- Each ant carrying the triple and the corresponding field as ant-template

\[ \langle S, P, O \rangle \rightarrow S \rightarrow P \rightarrow O \rightarrow \]

- These ants search the cluster by following the pheromone which matches the triple
- After storing the triple, the ant returns home and amplifies the pheromone on its way.

http://digipolis.ag-nbi.de
initialize \( age \) to a given integer value \( > 0 \)

while \( age > 0 \) do

if (drop probability currently high) then

Drop triple, drop the scent of cluster-resource and in weaker quantity on the neighbor nodes and die

else

move to neighborhood node based on cluster-resource

\( age = age - 1 \)

end if

end while

Drop triple, drop the scent of cluster-resource and in weaker quantity on the neighbor nodes and die
Read Algorithm

- Create an ant carrying the fixed part of the Basic Triple Pattern as ant-template

\[<?, ?, O> \rightarrow O \rightarrow \text{ant} \]

- The ant follows the pheromone which matches the template and searches in the appropriate layer

- When it finds a result, returns home carrying the result and amplifies the pheromone
• Ants choose their next node
  - Only with local knowledge
  - At random
  - With the probability of each edge depending on the pheromone strength matching the carried template

• Ants leave pheromones on their way back, which evaporate over the time

**Shortest paths to clusters emerge**
initialize \textit{age} to a given integer value \textgreater{} 0

\begin{verbatim}
while \textit{age} > 0 do
    add the current node to memory
    if (match found on current node) then
        make a copy and use memory to return to the origin. Leave scent of cluster-resource on each node on the way back with each hop in a weaker quantity.
        return the copy as result and die.
    else
        Move to neighborhood node based on cluster-resource
        \textit{age} = \textit{age} - 1
    end if
end while
die
\end{verbatim}
• Data is stored on a node
  - only with local knowledge
  - with a probability depending on the similarity of the data to the neighborhood
  - only if the similarity to the current node is over a threshold value

Creates clusters of similar data

• Clusters have to be maintained by moving misplaced data to its cluster
• Misplaced data („corpses“)

• Special ants without template roam the network randomly and clean it from corpses

• If a corpse is found, it will be moved by a spawned write-ant carrying the corpse

Leads to more homogeneous clusters and increases recall
Similarity Measurement
• All creatures are equal...
... but some are more equal than others

http://birds.org/description/onto.rdf#sparrow
http://birds.org/description/onto.rdf#duck
• **Idea:** Similar Things might have similar URIs

• **Algorithm:**
  1. Split path and domain components of the URIs
  2. Compare parts of both URIs pairwise with Levenshtein distance
  3. Sum up the weighted results
  4. Similarity between the URIs is the sum

- No Ontological Semantic
- Clusters based on namespace scheme
Similarity Measures
Fingerprint Similarity

• Idea: String Hashing

• Algorithm:
  1. Hash the URI Strings with \( h() \)
  2. Interpret the values as bitvectors
  3. Compare the vectors with the dot-product
  4. Similarity is the normalized result

+ Very fast to compute
- Lost of semantic information
• Similarity based on ontological knowledge

• Requirements and assumptions:
  - Only local decisions
  - No global ontology
  - Similarity measure operating on a fragment of an ontology
Semantic Clustering

- Extended behavior of out ants:
  - Carry triple and local type hierarchy
  - Type concentration on nodes determines responsibility for a certain type
  - Ants learn on their way and merge T-Boxes
  - Drop probability determined by similarity of type carried with type dominant on node
Simulation & Implementation
Evaluation
Evaluation
Test environment

- First Beta Implementation
- 10 computers simulating 10, 20, 30, 40 nodes
  - Nodes form a random graph
  - No connected nodes on the same computer
- Test system

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Pentium 4 2.4 Ghz</td>
</tr>
<tr>
<td>RAM</td>
<td>512 MByte</td>
</tr>
<tr>
<td>Network</td>
<td>100 MBit/s Ethernet</td>
</tr>
<tr>
<td>Operating System</td>
<td>Debian Linux Kernel 2.6.24</td>
</tr>
<tr>
<td>Java</td>
<td>1.5.0.17</td>
</tr>
</tbody>
</table>

- Test data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
<td>WordNet Ontology</td>
</tr>
<tr>
<td>Write</td>
<td>Sets of 1K, 10K, 100K, 1M triples</td>
</tr>
<tr>
<td>Read</td>
<td>10k random operations for each set</td>
</tr>
</tbody>
</table>

http://digipolis.ag-nbi.de
Evaluation
Write Operation

Average Write Time in ms

10 Nodes 20 Nodes 30 Nodes 40 Nodes

1000 10000 100000 1000000

http://digipolis.ag-nbi.de
Evaluation
Read Operation

Average Read Time in ms

10 Nodes 20 Nodes 30 Nodes 40 Nodes

- 1000
- 10000
- 100000
- 1000000
Future work

- Semantic Similarity
- Swarm-based Reasoning
- Efficient full SPARQL support
- Use geographical information to cluster even better

http://digipolis.ag-nbi.de
Conclusion

• We can use self-organization to cluster similar things, and build scalable storage service

• The evaluation of our initial Alpha implementation shows that such swarm-based storage system can scale out.
Thank You

Questions?
Image Reference

- http://1.bp.blogspot.com/_TbfuMu7x3_s/SNroUyUjxTI/AAAAAAAAAKA/f_s8I8bpyLI/s400/antclusterinicial.jpg
- http://www.answers.com/topic/ant-colony-optimization
- http://upload.wikimedia.org/wikipedia/commons/1/17/Aco_shortpath.svg
Extra Slides
URI Similarity

- URI Similarity:
- Take pairwise Levenshtein-distance in host and path parts eg:

\[ sim_{host} = \sum_{i=1}^{\min(k,l)} c_i \text{edit}(m_{k-i}, n_{l-i}) \]

- Weight components along their hierarchy, eg:

\[ c_i = \frac{2^{\max(k,l) - i}}{2^{\max(k,l)} - 1} \]

- Weight host and path importance, eg. 0.9/0.1

- Cats and dogs are quite similar:

\[ 1 \cdot 0.9 + \left( \frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 0 \right) \cdot 0.1 = 0.96 \]