Social Network Benchmark Task Force

4th TUC Meeting
Amsterdam - April 3, 2014
Task Force

• University
  – VUA - The Vrije Universiteit Amsterdam
  – UPC - Universitat Politècnica de Catalunya
  – TUM - Technische Universität München

• Industry
  – RDF
    • OpenLink Software (Virtuoso)
  – Graph Databases
    • Neo Technology (Neo4J)
    • Sparsity Technology (DEX)
Social Network Analysis

• Intuitive: everybody knows what a SN is
  – Facebook, Twitter, LinkedIn, ...
• SNs can be easily represented as a graph
  – Entities are the nodes (Person, Group, Tag, Post, ...)
  – Relationships are the edges (Friend, Likes, Follows, ...)
• Different scales: from small to very large SNs
  – Up to billions of nodes and edges
• Multiple query needs:
  – interactive, analytical, transactional
• Multiple types of uses:
  – marketing, recommendation, social interactions, fraud detection, ...
Audience

• For **end users** facing graph processing tasks
  – recognizable scenario to compare merits of different products and technologies
• For **vendors** of graph database technology
  – checklist of features and performance characteristics
• For **researchers**, both industrial and academic
  – challenges in multiple choke-point areas such as graph query optimization and (distributed) graph analysis
Workloads

- **Interactive**: tests a system's throughput with relatively simple queries with concurrent updates
  - *Show all photos posted by my friends that I was tagged in*

- **Business Intelligence**: consists of complex structured queries for analyzing online behavior
  - *Who got the most replies during 1st month of participation?*

- **Graph Analytics**: tests the functionality and scalability on most of the data as a single operation
  - *PageRank*
• **Graph database** systems
  – e.g. Neo4j, InfiniteGraph, DEX, Titan
• **Graph programming frameworks**
  – e.g. Giraph, Signal/Collect, Graphlab, Green Marl, Grappa
• **RDF** database systems
  – e.g. OWLIM, Virtuoso, BigData, Jena TDB, Stardog, Allegrograph
• **Relational** database systems
  – e.g. Postgres, MySQL, Oracle, DB2, SQLServer, Virtuoso, MonetDB, Vectorwise, Vertica
• **noSQL** database systems
  – e.g. HBase, REDIS, MongoDB, CouchDB, or even MapReduce systems like Hadoop and Pig
# Workloads by system

<table>
<thead>
<tr>
<th>System</th>
<th>Interactive</th>
<th>Business Intelligence</th>
<th>Graph Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph databases</td>
<td>Yes</td>
<td>Yes</td>
<td>Maybe</td>
</tr>
<tr>
<td>Graph programming frameworks</td>
<td>-</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>RDF databases</td>
<td>Yes</td>
<td>Yes</td>
<td>-</td>
</tr>
<tr>
<td>Relational databases</td>
<td>Yes</td>
<td>Yes</td>
<td>Maybe, by keeping state in temporary tables, and using the functional features of PL-SQL</td>
</tr>
<tr>
<td>NoSQL Key-value</td>
<td>Maybe</td>
<td>Maybe</td>
<td>-</td>
</tr>
<tr>
<td>NoSQL MapReduce</td>
<td>-</td>
<td>Maybe</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Expected Results

• Four main elements:
  – *data schema*: defines the structure of the data
  – *workloads*: defines the set of operations to perform
  – *test driver*: to execute the workloads
  – *performance metrics*: used to measure (quantitatively) the performance of the systems
  – *execution rules*: defined to assure that the results from different executions of the benchmark are valid and comparable

• Software as Open Source (GitHub)
  – data generator, query drivers, validation tools, ...
Data Schema

• Structure of the Social Network / Graph:
  – Entities (nodes)
  – Relationships between entities (edges)
  – Attributes for entities and relationships

• Some of the relationships represent dimensions (for BI analysis)
Data Generation Process

• Produce synthetic data that mimics the characteristics of real SN data

• Graph model:
  – correlated property (directed labeled) graph

• Based on SIB–S3G2 Social Graph Generator
  – property dictionaries extracted from DBPedia with specific ranking and probability density functions
  – subgraph generation: new nodes and new edges in one single pass
  – MapReduce for scalability
DBGen improvements

• Schema updates
  – hasTag & likes relationships
  – knows creationDate attribute
• Deterministic
• Facebook-like knows distribution
• New distributions to rebalance the size of the user activity w.r.t. the graph size
  – e.g. number and size of posts/comments
• Quantization of population (categories of country populations)
• Compressed output and serialization enhancements
Interactive Workload

• Tests system throughput with relatively simple queries and concurrent updates
• Current set: 12 read-only queries + 1 proposal of shortest path
• For each query:
  – Name and detailed description in plain English
  – List of input parameters
  – Expected result: content and format
  – Textual functional description
  – Relevance:
    • textual description (plain English) of the reasoning for including this query in the workload
    • discussion about the technical challenges (Choke Points) targeted
  – Validation parameters and validation results
  – SPARQL and SQL examples
Example: Q3

Name: Friends within 2 hops that have been in two countries

Description:
Find Friends and Friends of Friends of the user A that have made a post in the foreign countries X and Y within a specified period. We count only posts that are made in the country that is different from the country of a friend. The result should be sorted descending by total number of posts, and then by person URI. Top 20 should be shown. The user A (as friend of his friend) should not be in the result.

Parameter:
- Person
- CountryX
- CountryY
- startDate - the beginning of the requested period
- Duration - requested period in days

Result:
- Person.id, Person.firstname, Person.lastName
- Number of post of each country and the sum of all posts

Relevance:
- Choke Points: CP3.3
- If one country is large but anticorrelated with the country of self then processing this before a smaller but positively correlated country can be beneficial
## Interactive: Choke Point Coverage

<table>
<thead>
<tr>
<th>Group</th>
<th>Choke Point</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
<th>Q10</th>
<th>Q11</th>
<th>Q12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation Performance</td>
<td>1.2</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.6</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.7</td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Join Performance</td>
<td>2.3</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.4</td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.6</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Access Locality</td>
<td>3.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expression Calculation</td>
<td>4.2a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correlated Subqueries</td>
<td>5.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Parallelism and Concurrency</td>
<td>6.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RDF and Graph Specifics</td>
<td>7.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7.3</td>
<td></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>
Interactive Workload Improvements

- 12 queries
  - tested in SPARQL and SQL
  - validation parameters
- Update streams
  - analysis and definition of the update events
- Substitution parameters
  - Mining data
  - Query parameters based on distributions and correlations
- Query mixes
- Test driver
- First draft of execution rules
Scale Factors

- **DBGen parameters:**
  - fixed by default
    - distributions
    - quantizations
    - 3 years of activity
  - variable parameter: number of users

- **Validation scale factor:** 100K users
  - 53M nodes, 284M edges, 384M attribute values
  - more than 720M triples
  - 12GB data
Future Work

• First release of the Interactive workload
  – End April 2014
  – DBGEN, QGEN and test driver
  – Validation, execution and auditing rules

• Second draft of BI queries
  – analysis of new requirements to schema and data

• First draft of analytical workload
Thank you!