Benchmarking
Graph Data Management Systems
Peter Boncz
boncz@cwi.nl

www.ldbccouncil.org
Graph Data Management Scenarios

• Creating and Maintaining Graphs
  – e.g. social networks, but typically the graph is derived from raw tabular data
  – computing similarities between entities

• Analyzing interactions
  – what about indirect (multi-step) relationships?
  – learning about network structure
Graph Data Management Systems

- **Graph database** systems
  - e.g. Neo4j, InfiniteGraph, Sparksee, Titan
- **Graph programming frameworks**
  - e.g. Giraph, GraphLab, Signal/Collect, SNAP
- **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
- **MapReduce** cluster programming systems
  - E.g. Flink, Hadoop, Pig, and Spark (GraphFrames)
Why Benchmarking?

- make competing products comparable
- accelerate progress, make technology viable
What is the LDBC?

Linked Data Benchmark Council = LDBC

• Industry entity similar to TPC (www.tpc.org)
• Focusing on graph and RDF store benchmarking
LDBC Organization (non-profit)

+ non-profit members (FORTH) & personal members
+ **Task Forces**, volunteers developing benchmarks
+ **TUC**: Technical User Community (8 workshops, ~40 graph and RDF user case studies, 18 vendor presentations)
LDBC Task Forces

• Semantic Publishing Benchmark Task Force
  – Develops industry-grade RDF benchmark

• Social Network Benchmark Task Force
  – Develops benchmark for graph data management systems
  – Broad coverage: three workloads

• Graph Analytics Task Force
  – Spin-off from the SNB task force (third workload)

• Graph Query Language Task Force
  – Studies features of graph database query languages
LDBC benchmarks consist of..

• Four main elements:
  – *data schema*: defines the structure of the data
  – *workloads*: defines the set of operations to perform
  – *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, ...
Semantic Publishing Benchmark (SPB)
Social Network Benchmark

- **Interactive Workload**: tests throughput running short queries while consistently handling concurrent updates
  - *Show all photos posted by my friends that I was tagged in*

- **Business Intelligence Workload**: consists of complex structured queries for analyzing online behavior
  - *Influential people the topic of open source development?*

- **Graph Analytics Workload**: tests the functionality and scalability on most of the data as a single operation
  - *PageRank, Shortest Path(s), Community Detection*
## SNB Workloads: target systems?

<table>
<thead>
<tr>
<th>System Type</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>No</td>
<td>Maybe</td>
<td>Yes</td>
</tr>
<tr>
<td>RDF databases</td>
<td>Yes</td>
<td>Yes</td>
<td>No</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>No</td>
<td>No – unless they support MapReduce</td>
</tr>
<tr>
<td>NoSQL MapReduce</td>
<td>No</td>
<td>Maybe</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Interactive (On-line) Workload

• Test online ACID features and scalability
• The system under test is expected to run in a steady state, providing durable storage
• Updates are typically small
• Updates will conflict a small percentage of the time
• Queries typically touch a small fraction of the database
The LDBC Social Network Benchmark: Interactive Workload

Orri Erling
OpenLink Software, UK
oerling@openlinksw.com

Alex Averbuch
Neo Technology, Sweden
alex.averbuch@neotechnology.com

Josep Larriba-Pey
Sparsity Technologies, Spain
larri@sparsity-technologies.com

Hassan Chafi
Oracle Labs, USA
hassan.chafi@oracle.com

Andrey Gubichev
TU Munich, Germany
gubichev@in.tum.de

Arnaud Prat*
Universitat Politècnica de Catalunya, Spain
aprat@ac.upc.edu

Minh-Duc Pham
VU University Amsterdam,
The Netherlands
m.d.pham@vu.nl

Peter Boncz
CWI, Amsterdam, The
Netherlands
boncz@cwi.nl

ABSTRACT

The Linked Data Benchmark Council (LDBC) is now two years underway and has gathered strong industrial participation for its mission to establish benchmarks, and benchmarking practices for evaluating graph data management systems. The LDBC introduced a new choke-point driven methodology for developing benchmark workloads, which combines user input with input from expert systems architects, which we outline. This paper describes the LDBC Social Network Benchmark (SNB), and presents database benchmarking innovation in terms of graph query functionality, structural complexity, and data size.

A key characteristic of the SNB is that it focuses on graph queries, which are typically harder to evaluate than traditional table, for instance as a table where every row contains an edge, and the start and end vertex of every edge are a foreign key reference (in SQL terms). However, what makes a data management problem a graph problem is that the data analysis is not only about the values of the data items in such a table, but about the connection patterns between the various pieces. SQL-based systems were not originally designed for this – though systems have implemented diverse extensions for navigational and recursive query execution.

In recent years, the database industry has seen a proliferation of new graph-oriented data management technologies. The SNB focuses on this important area by providing a solid benchmark for evaluating these systems.
Business Intelligence Workload

• The workload stresses query execution and optimization
• Queries typically touch a large fraction of the data
• The queries are concurrent with trickle load
• The queries touch more data as the database grows
Graph Analytics Workload (Graphalytics)

• The analytics is done on most of the data in the graph as a single operation
• The analysis itself produces large intermediate results
• The analysis transactional: no need for isolation from possible concurrent updates
Graphalytics Algorithms

• general statistics (STATS)
  – counts the numbers of vertices and edges in the graph and computes the mean local clustering coefficients
• breadth-first search (BFS)
  – traverses the graph starting from a seed vertex, visiting first all the neighbors of a vertex before moving to the neighbors of the neighbors.
• connected components (CONN) algorithm
  – determines for each vertex the connected component it belongs to.
• community detection (CD) algorithm
  – detects groups of nodes that are connected to each other stronger than they are connected to the rest of the graph
• graph evolution (EVO)
  – predicts the evolution of the graph according to the “forest fire” model
VLDB2016 paper

Google: “Peter Boncz” ➔ click: publications ➔ find: LDBC Graphalytics
http://oai.cwi.nl/oai/asset/24634/24634B.pdf

LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis on Parallel and Distributed Platforms


Oracle Labs, Intel Labs, IBM Research, Huawei Research America, Delft University of Technology, UPC Barcelona, Georgia Tech, CWI Amsterdam

ABSTRACT

In this paper we introduce LDBC Graphalytics, a new industrial-grade benchmark for graph analysis platforms. It consists of six deterministic algorithms, standard datasets, synthetic dataset generators, and reference output, that enable the objective comparison of graph analysis platforms. Its test harness produces deep metrics that quantify multiple kinds of system scalability, such as horizontal/vertical and weak/strong, and of robustness, such as failures and performance variability. The benchmark comes with open-source software for generating data and monitoring performance. We describe and analyze six implementations of the benchmark (three from the community, three from the industry), providing insights into the strengths and weaknesses of the platforms. Key to our contribution, vendors perform the tuning and benchmarking of their platforms.
Graph Query Language Task Force

- Renzo Angles, Universidad de Talca
- **Marcelo Arenas, PUC Chile - task force lead**
- Pablo Barceló, Universidad de Chile
- Peter Boncz, Vrije Universiteit Amsterdam
- George Fletcher, Eindhoven University of Technology
- Irini Fundulaki, Foundation for Research and Technology - Hellas (FORTH)
- Claudio Gutierrez, Universidad de Chile
- Tobias Lindaaker, Neo Technology
- Marcus Paradies, SAP
- Raquel Pau, UPC
- Arnau Prat, UPC
- Tomer Sagi, HP Labs
- Oskar van Rest, Oracle Labs
- Hannes Voigt, TU Dresden
- Yinglong Xia, Huawei Research America
**More Information**

- [http://www.ldbcouncil.org](http://www.ldbcouncil.org)
- [http://wiki.ldbcouncil.org](http://wiki.ldbcouncil.org)
- [http://github.com/ldbc](http://github.com/ldbc)

**Blogs**
**Specifications**
**Early Result FDRs**
**Videos of TUC talks**
**Developer info**
**Code, Issue Tracking**

---

**LDBC**

- The graph & RDF benchmark reference
- [Benchmarks](https://www.ldbcouncil.org/benchmarks)
- [Developer](https://github.com/ldbc)

---

**Benchmarks**

Here you may find the results for different benchmarks, i.e. the Social Network Benchmark (SNB) and the Semantic Publishing Benchmark (SPB), their definitions and best practices, the repositories where to find the data generators and the query implementations, an access to the intranet for the LDBC Industry partners and a list of the LDBC member vendors.

---

**LDBC official benchmarks for industry**

- [Semantic Publishing Benchmark (SPB)](https://www.ldbcouncil.org/benchmarks/spb)
  - What are Graph Database systems?
  - What are RDF Database systems?
  - Why is benchmarking valuable?
  - What is the mission of LDBC?

---

**The benchmarking community**

Test the SPB and/or contribute to it.
Test the SNB and/or contribute to it.
Provide feedback to the community.
Talk Structure

• Introduction of LDBC and its benchmarks
  – SPB and SNB (Interactive, BI, Graphalytics)

• SNB datagen
  – how to generate an interesting social network

• “Choke Point”–based benchmark design
  – Examples in SNB Interactive and Graphalytics

• Conclusion / Hard Stop
Talk Structure

• Introduction of LDBC and its benchmarks
  – SPB and SNB (Interactive, BI, Graphalytics)

• SNB datagen
  – how to generate an interesting social network

• “Choke Point”–based benchmark design
  – Examples in SNB Interactive and Graphalytics

• Conclusion / Hard Stop
DATAGEN: social network generator

advanced generation of:

• network structure
  – *Power law* distributions, small diameter
Friendship Degree Distribution

• Based on “Anatomy of Facebook” blogpost (2013)
• Diameter increases logarithmically with scale factor
  – New:
    function has been made pluggable
DATAGEN: social network generator

advanced generation of:

• network structure
  – Power law distributions, small diameter

• property values
  – realistic, correlated value distributions
Data correlations between attributes

```
SELECT personID from person
WHERE firstName = 'Joachim' AND addressCountry = 'Germany'
```

```
SELECT personID from person
WHERE firstName = 'Cesare' AND addressCountry = 'Italy'
```

- Query optimizers may underestimate or overestimate the result size of conjunctive predicates

Anti-Correlation
Data correlations **between attributes**

```sql
SELECT COUNT(*)
FROM paper pa1 JOIN conferences cn1 ON pa1.journal = jn1.ID
    paper pa2 JOIN conferences cn2 ON pa2.journal = jn2.ID
WHERE pa1.author = pa2.author  AND
    cn1.name = 'VLDB'  AND  cn2.name = 'SIGMOD'
```
Data correlations over joins

```sql
SELECT COUNT(*)
FROM paper pa1 JOIN conferences cn1 ON pa1.journal = cn1.ID
    paper pa2 JOIN conferences cn2 ON pa2.journal = cn2.ID
WHERE pa1.author = pa2.author AND cn1.name = 'VLDB' AND cn2.name = 'SIGMOD'
```

- A challenge to the optimizers to adjust estimated join hit ratio depending on other predicates

**Correlated predicates are still a frontier area in database research**
Realistic Correlated Value Distributions

- Person.firstname correlates with Person.location
  - Values taken from DBpedia
- Many other correlations and dependencies..
  - e.g. university depends on location
- In forum discussions, people read DBpedia articles to each other (= correlation between message text and discussion topic)
  - Topic = DBpedia article title
  - Text = one sentence of the article
Correlated Value Property in LDBC SNB

- Main source of dictionary values from DBpedia ([http://dbpedia.org](http://dbpedia.org))

<table>
<thead>
<tr>
<th>Correlated Value Property</th>
<th>Dictionary Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>person.location,</td>
<td>person.firstName (typical names)</td>
</tr>
<tr>
<td>person.gender</td>
<td>person.interests (popular artist)</td>
</tr>
<tr>
<td>person.location</td>
<td>person.lastName (typical names)</td>
</tr>
<tr>
<td></td>
<td>person.university (nearby universities)</td>
</tr>
<tr>
<td></td>
<td>person.company (in country)</td>
</tr>
<tr>
<td></td>
<td>person.languages (spoken in country)</td>
</tr>
<tr>
<td>person.language</td>
<td>person.forum.message.language (speaks)</td>
</tr>
<tr>
<td>person.interests</td>
<td>person.forum.post.topic (in)</td>
</tr>
<tr>
<td>post.topic</td>
<td>post.text (DBpedia article lines)</td>
</tr>
<tr>
<td></td>
<td>post.comment.text (DBpedia article lines)</td>
</tr>
<tr>
<td>person.employer</td>
<td>person.email (@company, @university)</td>
</tr>
<tr>
<td>(friendship.userId1,</td>
<td>friendship.terminator (=one of the two)</td>
</tr>
<tr>
<td>friendship.userId2)</td>
<td></td>
</tr>
<tr>
<td>message.photoLocation</td>
<td>message.latitude (matches location)</td>
</tr>
<tr>
<td></td>
<td>message.longitude (matches location)</td>
</tr>
<tr>
<td>friendship.requestDate</td>
<td>friendship.approveDate (&gt;)</td>
</tr>
<tr>
<td></td>
<td>friendship.deniedDate (&gt;)</td>
</tr>
<tr>
<td>person.birthDate</td>
<td>person.createdDate (&gt;)</td>
</tr>
<tr>
<td>person.createdDate</td>
<td>person.forum.message.createdDate (&gt;)</td>
</tr>
<tr>
<td></td>
<td>person.forum.createdDate (&gt;)</td>
</tr>
<tr>
<td>forum.createdDate</td>
<td>message.photoTime (&gt;)</td>
</tr>
<tr>
<td></td>
<td>forum.post.createdDate (&gt;)</td>
</tr>
<tr>
<td></td>
<td>forum.groupmembership.joinedDate (&gt;)</td>
</tr>
<tr>
<td>message.createdDate</td>
<td>message.comment.createdDate (&gt;)</td>
</tr>
</tbody>
</table>
DATAGEN: social network generator

advanced generation of:

• network structure
  – Power law distributions, small diameter

• property values
  – realistic, correlated value distributions
  – temporal correlations / “flash mobs”

• correlations between values and structure
  – 2 correlation “dimensions”: location & interests
Correlated Edge Generation

- Student: "Anna" (University of Leipzig, Germany, 1990)
- Student: "Laura" (University of Leipzig, 1990)
- Student: Britney Spears (University of Leipzig, 1990)
- Student: Britney Spears (University of Amsterdam, Netherlands, 1990)
Simple approach

- Compute similarity of two nodes based on their (correlated) properties.
- Use a probability density function wrt to this similarity for connecting nodes.

Danger: this is very expensive to compute on a large graph! (quadratic, random access)
Our observation

Probability that two nodes are connected is skewed w.r.t the similarity between the nodes (due to probability distr.)

Trick: disregard nodes with too large similarity distance (only connect nodes in a similarity window)
MapReduce data generation: one map pass per Correlation Dimension

“S3G2: A Scalable Structure-correlated Social Graph Generator”
DATAGEN: social network generator

advanced generation of:

• network structure
  – Power law distributions, small diameter

• property values
  – realistic, correlated value distributions
  – temporal correlations / “flash mobs”
Temporal Effects (Flash Mobs)

- Forum posts generation spikes in time for certain topics:
DATAGEN: Scaling

- Scale Factor (SF) is the size of the CSV input data in GB
- Some Virtuoso SQL stats at SF=30:

<table>
<thead>
<tr>
<th>SFs</th>
<th>Number of entities (x 1000000)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nodes</td>
</tr>
<tr>
<td>30</td>
<td>99.4</td>
</tr>
<tr>
<td>100</td>
<td>317.7</td>
</tr>
<tr>
<td>300</td>
<td>907.6</td>
</tr>
<tr>
<td>1000</td>
<td>2930.7</td>
</tr>
</tbody>
</table>

- Table | Size (MB) | Largest Index (MB) |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>post</td>
<td>76815</td>
<td>ps_content (41697)</td>
</tr>
<tr>
<td>likes</td>
<td>23645</td>
<td>l_creationdate (11308)</td>
</tr>
<tr>
<td>forum_person</td>
<td>9343</td>
<td>fp_creationdate (5957)</td>
</tr>
</tbody>
</table>
DATAGEN: Graph Characteristics

Livejournal  | LFR3 (synthetic)  | LDBC DATAGEN

(a) Clustering Coefficient  | (a) Clustering Coefficient  | (a) Clustering Coefficient
(b) TPR  | (b) TPR  | (b) TPR

(c) Bridge Ratio  | (c) Bridges Ratio  | (c) Bridges Ratio
(d) Diameter  | (d) Diameter  | (d) Diameter

(e) Conductance  | (e) Conductance  | (e) Conductance
(f) log10(Size)  | (f) log10(Size)  | (f) log10(Size)

GRADE2014 “How community-like is the structure of synthetically generated graphs” - Arnau Prat(DAMA-UPC); David Domínguez-Sal (Sparsity Technologies)
Talk Structure

• Introduction of LDBC and its benchmarks
  – SPB and SNB (Interactive, BI, Graphalytics) ↔

• SNB datagen
  – how to generate an interesting social network

• “Choke Point”–based benchmark design
  – Examples in SNB Interactive and Graphalytics

• Conclusion / Hard Stop
Talk Structure

• Introduction of LDBC and its benchmarks
  – SPB and SNB (Interactive, BI, Graphalytics)

• SNB datagen
  – how to generate an interesting social network

• “Choke Point”–based benchmark design
  – Examples in SNB Interactive and Graphalytics

• Conclusion / Hard Stop
Database Benchmark Design

Desirable properties:
• Relevant. ➔ “Choke Points”
• Representative.
• Understandable.
• Economical.
• Accepted.
• Scalable.
• Portable.
• Fair.
• Evolvable.
• Public.


Multiple TPCTC papers, e.g.
Karl Huppler (2009) *The Art of Building a Good Benchmark*
CP1.4 Dependent GroupBy Keys

```
SELECT c_custkey, c_name, c_acctbal,
    sum(l_extendedprice * (1 - l_discount)) as revenue,
    n_name, c_address, c_phone, c_comment
FROM customer, orders, lineitem, nation
WHERE c_custkey = o_custkey and l_orderkey = o_orderkey
    and o_orderdate >= date '[DATE]' 
    and o_orderdate < date '[DATE]' + interval '3' month 
    and l_returnflag = 'R' and c_nationkey = n_nationkey
GROUP BY 
    c_custkey, c_name, c_acctbal, c_phone, n_name, 
    c_address, c_comment
ORDER BY revenue DESC
```
CP1.4 Dependent GroupBy Keys

```
SELECT c_custkey, c_name, c_acctbal, 
    sum(l_extendedprice * (1 - l_discount)) as revenue, 
    n_name, c_address, c_phone, c_comment 
FROM customer, orders, lineitem, nation 
WHERE c_custkey = o_custkey and l_orderkey = o_orderkey 
    and o_orderdate >= date '[DATE]' 
    and o_orderdate < date '[DATE]' + interval '3' month 
    and l_returnflag = 'R' and c_nationkey = n_nationkey 
GROUP BY 
    c_custkey, c_name, c_acctbal, c_phone, 
    c_address, c_comment, n_name 
ORDER BY revenue DESC
```
Q1. Extract description of friends with a given name. Given a person’s firstName, return up to 20 people with the same first name, sorted by increasing distance (max 3) from a given person, and for people within the same distance sorted by last name. Results should include the list of workplaces and places of study.

Q2. Find the newest 20 posts and comments from your friends. Given a start Person, find (most recent) Posts and Comments from all of that Person’s friends, that were created before (and including) a given Date. Return the top 20 Posts/Comments, and the Person that created each of them. Sort results descending by creation date, and then ascending by Post identifier.

Q3. Friends within 2 steps that have recently traveled to countries X and Y. Find friends and friends of friends of a given Person who have made a post or a comment in the foreign CountryX and CountryY within a specified period of DurationInDays after a startDate. Return top 20 Persons, sorted descending by total number of posts.

Q4. New Topics. Given a start Person, find the top 10 most popular Tags (by total number of posts with the tag) that are attached to Posts that were created by that Person’s friends. Only include Tags that were attached to Posts created within a given time interval, and that were never attached to Posts created before this interval.

Q5. New groups. Given a start Person, find the top 20 Forums which that Person’s friends and friends of friends became members of after a given Date. Sort results descending by the number of Posts in each Forum that were created by any of these Persons.

Q6. Tag co-occurrence. Given a start Person and some Tag, find the other Tags that occur together with this Tag on Posts that were created by start Person’s friends and friends of friends. Return top 10 Tags, sorted descending by the count of Posts that were created by these Persons, which contain both this Tag and the given Tag.

Q7. Recent likes. For the specified Person get the most recent likes of any of the person’s posts, and the latency between the corresponding post and the like. Flag Likes from outside the direct connections. Return top 20 Likes, ordered descending by creation date of the like.

Q8. Most recent replies. This query retrieves the 20 most recent reply comments to all the posts and comments of Person, ordered descending by creation date.

Q9. Latest Posts. Find the most recent 20 posts and comments from all friends, or friends-of-friends of Person, but created before a Date. Return posts, their creators and creation dates, sort descending by creation date.

Q10. Friend recommendation. Find a friend of a friend who posts much about the interests of Person and little about topics that are not in the interests of the user. The search is restricted by the candidate’s horoscopeSign. Returns 10 Persons for whom the difference between the total number of their posts about the interests of the specified user and the total number of their posts that are not in the interests of the user, is as large as possible. Sort the result descending by this difference.

Q11. Job referral. Find a friend of the specified Person, or a friend of her friend (excluding the specified person), who has long worked in a company in a specified Country. Sort ascending by start date, and then ascending by person identifier. Top 10 result should be shown.

Q12. Expert Search. Find friends of a Person who have replied the most to posts with a tag in a given TagCategory. Count the number of these reply Comments, and collect the Tags that were attached to the Posts they replied to. Return top 20 persons, sorted descending by number of replies.

Q13. Single shortest path. Given PersonX and PersonY, find the shortest path between them in the subgraph induced by the Knows relationship. The weight of the path takes into consideration amount of Posts/Comments exchanged.

Q14. Weighted paths. Given PersonX and PersonY, find all weighted paths of the shortest length between them in the subgraph induced by the Knows relationship. The weight of the path takes into consideration amount of Posts/Comments exchanged.
Choke-Point: **shortest paths**

Q14. *Weighted paths.* Given PersonX and PersonY, find all weighted paths of the shortest length between them in the subgraph induced by the Knows relationship. The weight of the path takes into consideration amount of Posts/Comments exchanged.

- compute weights over a **recursive forum traversal**
  - on the fly, or
  - materialized, but then maintain them under updates
- compute **shortest paths** using these weights in the friends graph
Q3. Friends within 2 steps that have recently traveled to countries X and Y. Find friends and friends of friends of a given Person who have made a post or a comment in the foreign CountryX and CountryY within a specified period of DurationInDays after a startDate. Return top 20 Persons, sorted descending by total number of posts.

Q4. New Topics. Given a start Person, find the top 10 most popular Tags (by total number of posts with the tag) that are attached to Posts that were created by that Person's friends. Only include Tags that were attached to Posts created within a given time interval, and that were never attached to Posts created before this interval.

Q5. New groups. Given a start Person, find the top 20 Forums which that Person's friends and friends of friends became members of after a given Date. Sort results descending by the number of Posts in each Forum that were created by any of these Persons.

Q6. Tag co-occurrence. Given a start Person and some Tag, find the other Tags that occur together with this Tag on Posts that were created by start Person's friends and friends of friends. Return top 10 Tags, sorted descending by the count of Posts that were created by these Persons, which contain both this Tag and the given Tag.

Q7. Recent likes. For the specified Person get the most recent likes of any of the person's posts, and the latency between the corresponding post and the like. Flag Likes from outside the direct connections. Return top 20 Likes, ordered descending by creation date of the like.

Q8. Most recent replies. This query retrieves the 20 most recent reply comments to all the posts and comments of Person, ordered descending by creation date.

Q9. Latest Posts. Find the most recent 20 posts and comments from all friends, or friends-of-friends of Person, but created before a Date. Return posts, their creators and creation dates, sorted descending by creation date.

Q10. Friend recommendation. Find a friend of a friend who posts much about the interests of Person and little about topics that are not in the interests of the user. The search is restricted by the candidate's horoscopeSign. Returns 10 Persons for whom the difference between the total number of their posts about the interests of the specified user and the total number of their posts that are not in the interests of the user, is as large as possible. Sort the result descending by this difference.

Q11. Job referral. Find a friend of the specified Person, or a friend of her friend (excluding the specified person), who has long worked in a company in a specified Country. Sort ascending by start date, and then ascending by person identifier. Top 10 result should be shown.

Q12. Expert Search. Find friends of a Person who have replied the most to posts with a tag in a given TagCategory. Count the number of these reply Comments, and collect the Tags that were attached to the Posts they replied to. Return top 20 persons, sorted descending by number of replies.

Q13. Single shortest path. Given PersonX and PersonY, find the shortest path between them in the subgraph induced by the Knows relationship. The weight of the path takes into consideration amount of Posts/Comments exchanged.

Q14. Weighted paths. Given PersonX and PersonY, find all weighted paths of the shortest length between them in the subgraph induced by the Knows relationship. The weight of the path takes into consideration amount of Posts/Comments exchanged.
Choke-Point: outdegree correlation

Q3. Friends within 2 steps that recently traveled to countries X and Y. Find top 20 friends and friends of friends of a given Person who have made a post or a comment in the foreign CountryX and CountryY within a specified period of DurationInDays after a startDate. Sorted results descending by total number of posts.

- Travel is correlated with location
  - People travel more often to nearby countries
- Outdegree after \((\text{country}X,\text{country}Y)\) selection varies a lot
  - \((\text{Australia,NZ})\): high outdegree (“join hit ratio”)
  - \((\text{Australia,Belgium})\): low outdegree \(\iff\) different query plan, or navigation strategy likely wins
SNB Interactive: Experiment Setup

MapReduce-base data generation

• Generate 3 years of network activity for a certain amount of persons
  – 33 months of data ➔ bulk load
  – 3 months of data ➔ insert queries

• Scalable (SF1000 in one hour on 10 small compute nodes)
  – can also be used without a cluster (pseudo-distributed)

Query Driver fires off in parallel queries

• Combination of complex-reads, simple-reads, and inserts
• Server has to keep up inside a SLA
SNB Query Driver

• Window-based parallel query generation
  – Problem: friends graph has complex dependencies (non-partitionable). Could cause large checking overhead.
  – Solution: Window based approach for checking dependencies (Global Completion Time)
Problem: Parameter Sensitivity

SNB Interactive Q5:

explores the 2-hop friend neighbourhood, of one start person

Observation: depending on the start person, there is a large runtime variance
Parameter Curation

- **Example: Q3**
  - Problem: value correlations cause very large variance
  - Solution: data mine for **stable** parameter **equivalence classes**

TPCTC2014 “Parameter Curation for Benchmark Queries” Andrey Gubichev (TUM) & Peter Boncz (CWI)

- **form sliding windows of rows**
- **pick sub-window with the smallest variance in the next column**
Query Mix & Metric

Query Mix

• Insert queries (~10% of time):
  ➔ challenge: *execute parallel but respect data dependencies in the graph*

• Read-only Complex Queries (~50% of time)
  ➔ challenge: *generate query parameters with stable query behavior*
  Parameter Curation to find “equivalence classes” in parameters

• Simple Read-only Queries (~40% of time)
  – Retrieve Post / Retrieve Person Profile

Metric

• **Acceleration Factor (AF)** that can be sustained (+ AF/$ weighted by cost)
  – with 99th percentile of query latency within maximal query time
SNB Query Driver

- Dependency-aware parallel query generation
  - **Problem**: friends graph is non-partitionable, but imposes ordering constraints.
    
    *Could cause large checking overhead, impeding driver parallelism.*
  
  - **Solution**: Window-based checking approach for keeping driver threads roughly synchronized on a global timestamp.
    
    *Is helped by DATAGEN properties that ensure there is a minimal latency between certain dependencies (e.g. entering the network and making friends, or posting on a new friend’s forum). This minimal latency provides synchronization headroom.*
Graphalytics Choke Points

- Excessive network utilization
- Large graph memory footprint
- Poor Access Locality
- Skewed Execution Intensity
Talk Structure

• Introduction of LDBC and its benchmarks
  – SPB and ➔ SNB (Interactive, BI, Graphalytics) ⇐

• SNB datagen
  – how to generate an interesting social network

• “Choke Point”–based benchmark design
  – Examples in SNB Interactive and Graphalytics

• Conclusion / Hard Stop
Summary

LDBC

• Graph and RDF benchmark council
• Choke-point driven benchmark design (user+system expert scenarios)

Social Network Benchmark

• Advanced social network generator
  – skewed distributions, power laws, value/structure correlations, flash mobs
• 3 workloads
  – Interactive Workload
    • Parallel Query Driver that respects dependencies efficiently
    • Parameter Curation for stable results
  – Business Intelligence Workload
  – Graphalytics Workload
Thank you! - Questions?

- **LDBC Interactive**
  - SIGMOD 2015, *Erling et al.* The LDBC Social Network Benchmark: Interactive Workload

- **LDBC Graphalytics**
  - VLDB 2016, *Hegeman et al.* LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis on Parallel and Distributed Platforms

- **LDBC Datagen**
  - TPCTC 2012, *Pham et al.* S3G2: a Scalable Structure-correlated Social Graph Generator

- **Parameter Curation**
  - TPCTC 2014, *Gubichev et al.* Parameter Curation for Benchmark Queries

- **Every question you always wanted to ask about TPC-H, but where afraid to ask**
  - TPCTC 2013: *Boncz et al.* TPC-H Analyzed: Hidden Messages and Lessons Learned from an Influential Benchmark

- **Distributions of real-word and synthetic graphs**
  - GRADES 2014: *Prat et al.* How community-like is the structure of synthetically generated graphs

Google “Peter Boncz” then click “Publications”