FinBench: The new LDBC benchmark targeting financial scenario

Shipeng Qi

(with contributions from members of the FinBench Task Force)
Benchmark Overview
FinBench Motivation

- **SNB**, Social Network Benchmark, is designed based on social network scenarios, which is limited when applied to the financial service industry.

- **FinBench** objective is to design a high-quality benchmark for evaluating the performance of graph database systems in financial scenarios, e.g. anti-fraud and risk control, based on financial data patterns and query patterns.
Key Features in FinBench

• Dataset
  • PowerLaw distribution
  • Multiplicity
  • Hub Vertex

• Transaction Workload
  • Read-write query
  • Special graph patterns
  • Time-window filtering
  • Recursive path filtering
  • Truncation
Brief of the initial version

- Standard Design: all key features in proposal implemented
- Workload: Transaction Workload, including 12 complex read queries, 6 simple read queries, 19 write queries and 3 read-write queries
- Dataset: Up to SF10 scale supported
- Implementation on 3 systems: TuGraph, Galaxybase, and UltipaGraph
- Collaboration: 9 vendors in Task Force and 6 developers
Data Design and Generated Datasets

- Data Schema
- Data Distribution
- Datasets Statistics
Data Schema

Person:
- id: ID
- name: String
- isBlocked: Boolean
- createTime: DateTime
- gender: String
- birthday: Date
- country: String
- city: String

Account:
- id: ID
- createTime: DateTime
- isBlocked: Boolean
- type: String
- nickname: String
- phoneNumber: String
- email: Long String
- freqLoginType: String
- lastLoginTime: DateTime
- accountLevel: String

Loan:
- id: ID
- loanAmount: 64-bit Float
- balance: 64-bit Float
- usage: String
- interestRate: 32-bit Float

Medium:
- id: ID
- type: String
- createTime: DateTime
- isBlocked: Boolean
- lastLoginTime: DateTime
- riskLevel: String

Edges:
- Single edges from src to dst
- Multiple edges from src to dst
Data Distribution: Transfer Edge

- Degree: PowerLaw Distribution
- Asymmetric directed graph
- Hub vertex: degree increases with scale
  - MaxDegree = 1000 in SF1
  - MaxDegree = 10000 in SF10
  - Larger scale to be supported

<table>
<thead>
<tr>
<th>toId_in_degree</th>
<th>fromId</th>
<th>toId_multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>14891906703010022601</td>
<td>483742894974347364</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14897383119788711667</td>
<td>483742894974347364</td>
<td>14975266456736143480</td>
</tr>
<tr>
<td>186260053134745075</td>
<td>483742894974347364</td>
<td>186260053134745075</td>
</tr>
<tr>
<td>990799191802151598</td>
<td>483742894974347364</td>
<td>990799191802151598</td>
</tr>
<tr>
<td>4868391197178506662</td>
<td>483742894974347364</td>
<td>4868391197178506662</td>
</tr>
<tr>
<td>14972347439735813091</td>
<td>483742894974347364</td>
<td>14972347439735813091</td>
</tr>
<tr>
<td>296393150476325373</td>
<td>483742894974347364</td>
<td>296393150476325373</td>
</tr>
<tr>
<td>14908642118857140591</td>
<td>483742894974347364</td>
<td>14908642118857140591</td>
</tr>
<tr>
<td>14865576447420410341</td>
<td>483742894974347364</td>
<td>14865576447420410341</td>
</tr>
<tr>
<td>14911456868624245691</td>
<td>483742894974347364</td>
<td>14911456868624245691</td>
</tr>
</tbody>
</table>

only showing top 10 rows

<table>
<thead>
<tr>
<th>toId_in_degree</th>
<th>fromId</th>
<th>toId_multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>14891906703010022601</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14897383119788711667</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>186260053134745075</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>990799191802151598</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>4868391197178506662</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14972347439735813091</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>296393150476325373</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14908642118857140591</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14865576447420410341</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
<tr>
<td>14911456868624245691</td>
<td>1457887621282584061</td>
<td>14891906703010022601</td>
</tr>
</tbody>
</table>

only showing top 10 rows

Num of accounts: 26347
Num of transfer edges: 138209
Average Degree: 5.245720575397579
Average Multiplicity: 1.616574068658986

Profiling of SF0.1
Transaction Workload

- Transaction Workload
- Time Window Filtering
- Recursive Path Filtering
- Read-Write Query
- Truncation
- Query Mix
- Transaction Workload Driver
Transaction Workload

Scenario: financial activities among accounts, persons, companies, loans and media

Queries:

• 12 complex reads: match exact patterns including cycles and trees (see next slide) starting from one or two vertices
• 6 simple reads: discover the neighbourhood of an Account node
• 19 write queries: inserts, updates, deletes (cascade deletion)
• 3 read-write queries: transaction-wrapped complex reads
Transaction Workload: Example Patterns

Cycle
[Ref: Transaction Complex Read 4]

Tree
[Ref: Transaction Complex Read 6]
Time Window Filtering

• Fact: queries only look back in a limited time window
• Filtering: filter edges between startTime and endTime in traversal

Blocked medium related accounts
[Ref: Transaction Complex Read 1]
Recursive Path Filtering


- Timestamp order: e1 < ... < ei
- Amount order: e1 > ... > ei

Transfer trace after loan applied
[Ref: Transaction Complex Read 8]
Read-Write Query

• Transaction-wrapped complex reads (risk control strategy)

• If the complex read matches, commit the transaction with write query. Otherwise, transaction abort
Truncation

- Truncate less-important edges to avoid complexity explosion when traversing.
- Truncating is actually sampling.
- TruncationLimit and truncationOrder is defined to ensure consistency of results.

For example, keep only the top 100 edges in order of timestamp descending.
Benchmark Suite
## Datasets Statistics

| Supported Scale Factor | |V|   | |E|         |
|------------------------|------------------|------------------|
| 0.01                   | 8663             | 61674            |
| 0.1                    | 64485            | 610658           |
| 0.3                    | 192971           | 1830891          |
| 1                      | 643241           | 6091820          |
| 3                      | 1928439          | 18243343         |
| 10                     | 6069955          | 51889416         |

FinBench datasets of SF0.01 to SF10 are published at the [Google Drive](https://drive.google.com). These datasets were all generated using csv serializers in the initial version.

*Note: please see the tables in Appendix A for detailed statistics*
Transaction Workload Driver

Inherited from SNB Interactive driver, the driver has 3 modes of operation, all starting with a database containing the initial data set.

1. Generate validation data set
   - single-threaded, sequential execution
   - output: validation results

2. Validate implementation
   - single-threaded, sequential execution
   - input: validation results
   - output:
     - passed/failed validation
     - if failed: expected vs. actual results

3. Execute benchmark
   - multi-threaded, concurrent execution
   - Use TCR to control the load scale
   - output:
     - passed/failed schedule audit
     - throughput (operations per second)
     - per-query performance results
Query Mix

Inherited from SNB design:

- **Write queries and read-write queries**: operations issue times generated by the data generator

- **Complex read queries**: complex reads times are expressed in terms of update operations (update frequencies)

- **Simple read queries**: a sequence of short reads follows each complex read instance
Implementations and Standard-establishing Audits
## Implementations and Standard-establishing Audits

<table>
<thead>
<tr>
<th>System</th>
<th>Data Model</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>TuGraph</td>
<td>graph</td>
<td>Cypher</td>
</tr>
<tr>
<td>GalaxyBase</td>
<td>graph</td>
<td>Cypher</td>
</tr>
<tr>
<td>Ultipa</td>
<td>graph</td>
<td>UQL</td>
</tr>
</tbody>
</table>

- Packages and Reports available at [https://drive.google.com/drive/folders/1OQXrz2CkQke7SE9KWBiMeEn0KYx-QCO1](https://drive.google.com/drive/folders/1OQXrz2CkQke7SE9KWBiMeEn0KYx-QCO1)
- All systems passed cross-validation
Roadmap and acknowledgement
## Roadmap

<table>
<thead>
<tr>
<th>Version</th>
<th>Estimated Time</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1.0</td>
<td>Mid of 2023</td>
<td>• Runnable and auditable</td>
</tr>
<tr>
<td>0.2.0</td>
<td>End of 2023</td>
<td>• Larger scale data generation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Optimize parameter curation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Query mix profiling and design</td>
</tr>
<tr>
<td>0.3.0</td>
<td>2024</td>
<td>• New workload: Analytics workload</td>
</tr>
</tbody>
</table>
Acknowledgement

Task Force Members

<table>
<thead>
<tr>
<th>Name</th>
<th>Affiliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shipeng Qi</td>
<td>Ant Group</td>
</tr>
<tr>
<td>Bing Tong</td>
<td>CreateLink</td>
</tr>
<tr>
<td>Changyuan Wang</td>
<td>Vesoft</td>
</tr>
<tr>
<td>Yang Bin</td>
<td>Ultipa</td>
</tr>
<tr>
<td>Shenghao Zhang</td>
<td>StarGraph</td>
</tr>
</tbody>
</table>

Developers
Appendix
Work Chart: Goals of FinBench

Intended output

• The intended output is LDBC FinBench, a precise specification for evaluating graph database query and computation performance based on financial scenarios. It is capable of independent implementations using various graph database products, intended for approval as one of LDBC Standards. [https://github.com/ldbc/ldbc_finbench_docs](https://github.com/ldbc/ldbc_finbench_docs)

Work product

• Software for data generation: [https://github.com/ldbc/ldbc_finbench_datagen](https://github.com/ldbc/ldbc_finbench_datagen)
• Software for query driver: [https://github.com/ldbc/ldbc_finbench_driver](https://github.com/ldbc/ldbc_finbench_driver)
• Reference implementation: [https://github.com/ldbc/ldbc_finbench_transaction_impls](https://github.com/ldbc/ldbc_finbench_transaction_impls)
Resources

• Specification: https://github.com/ldbc/ldbc_finbench_docs

• Benchmark Suite

  • https://github.com/ldbc/ldbc_finbench_driver
  
  • https://github.com/ldbc/ldbc_finbench_datagen
  
  • https://github.com/ldbc/ldbc_finbench_transaction_impls
  
  • https://github.com/ldbc/ldbc_finbench_acid

• Datasets: https://drive.google.com/drive/folders/1tURBIJE56ZNC9YvMtug31peYD5csizCa?usp=sharing

• Certification audit packages: https://drive.google.com/drive/folders/1OQXrz2CkQke7SE9KWBiMeEn0KYx-QCOI?usp=sharing
## Dataset statistics

<table>
<thead>
<tr>
<th>V/E</th>
<th>Entity</th>
<th>SF0.01</th>
<th>SF0.1</th>
<th>SF0.3</th>
<th>SF1</th>
<th>SF3</th>
<th>SF10</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>account</td>
<td>2633</td>
<td>26347</td>
<td>79199</td>
<td>264075</td>
<td>791769</td>
<td>1980883</td>
</tr>
<tr>
<td>V</td>
<td>company</td>
<td>2633</td>
<td>4000</td>
<td>12000</td>
<td>40000</td>
<td>120000</td>
<td>300000</td>
</tr>
<tr>
<td>E</td>
<td>companyApplyLoan</td>
<td>524</td>
<td>5332</td>
<td>15761</td>
<td>52820</td>
<td>158678</td>
<td>397060</td>
</tr>
<tr>
<td>E</td>
<td>companyGuarantee</td>
<td>248</td>
<td>2315</td>
<td>7123</td>
<td>23870</td>
<td>71716</td>
<td>179526</td>
</tr>
<tr>
<td>E</td>
<td>companyInvest</td>
<td>860</td>
<td>8639</td>
<td>25853</td>
<td>86092</td>
<td>259884</td>
<td>650190</td>
</tr>
<tr>
<td>E</td>
<td>companyOwnAccount</td>
<td>864</td>
<td>8805</td>
<td>26356</td>
<td>88119</td>
<td>264352</td>
<td>660625</td>
</tr>
<tr>
<td>E</td>
<td>deposit</td>
<td>5199</td>
<td>51686</td>
<td>153521</td>
<td>512680</td>
<td>1534595</td>
<td>3829905</td>
</tr>
<tr>
<td>V</td>
<td>loan</td>
<td>1597</td>
<td>16138</td>
<td>47772</td>
<td>159166</td>
<td>476670</td>
<td>1189072</td>
</tr>
<tr>
<td>E</td>
<td>loanTransfer</td>
<td>4886</td>
<td>49180</td>
<td>145679</td>
<td>484657</td>
<td>1453874</td>
<td>3625556</td>
</tr>
<tr>
<td>V</td>
<td>medium</td>
<td>1000</td>
<td>10000</td>
<td>30000</td>
<td>100000</td>
<td>300000</td>
<td>2000000</td>
</tr>
<tr>
<td>V</td>
<td>person</td>
<td>800</td>
<td>8000</td>
<td>24000</td>
<td>80000</td>
<td>240000</td>
<td>600000</td>
</tr>
<tr>
<td>E</td>
<td>personApplyLoan</td>
<td>1073</td>
<td>10806</td>
<td>32011</td>
<td>106346</td>
<td>317992</td>
<td>792012</td>
</tr>
<tr>
<td>E</td>
<td>personGuarantee</td>
<td>469</td>
<td>4694</td>
<td>14221</td>
<td>47935</td>
<td>144064</td>
<td>359283</td>
</tr>
<tr>
<td>E</td>
<td>personInvest</td>
<td>1650</td>
<td>17296</td>
<td>52002</td>
<td>174064</td>
<td>520584</td>
<td>1300980</td>
</tr>
<tr>
<td>E</td>
<td>personOwnAccount</td>
<td>1769</td>
<td>17542</td>
<td>52843</td>
<td>175956</td>
<td>527417</td>
<td>1320258</td>
</tr>
<tr>
<td>E</td>
<td>repay</td>
<td>5046</td>
<td>50495</td>
<td>149559</td>
<td>497033</td>
<td>1488916</td>
<td>3715487</td>
</tr>
<tr>
<td>E</td>
<td>signIn</td>
<td>4384</td>
<td>44540</td>
<td>134532</td>
<td>451362</td>
<td>1350759</td>
<td>8996781</td>
</tr>
<tr>
<td>E</td>
<td>transfer</td>
<td>14145</td>
<td>138209</td>
<td>411882</td>
<td>1379527</td>
<td>4136803</td>
<td>11005032</td>
</tr>
<tr>
<td>E</td>
<td>withdraw</td>
<td>20557</td>
<td>201119</td>
<td>609548</td>
<td>2011359</td>
<td>6013709</td>
<td>15056721</td>
</tr>
</tbody>
</table>
**ACID Test Suite**

- Based on the “ACID Test” work in LDBC SNB
- Atomicity and Isolation Test: Based on failing cases
- Consistency and Durability Test
  - Execute the benchmark workload for duration T
  - Inject failure (e.g., a power failure, software crash, reboot, etc) into tested system
  - After the restart of system, check if all the last committed data survive
  - Check if all the constraints (uniqueness, precomputed properties, indices) are not violated

```
ATOMICITY AND ISOLATION TEST

- Repeatable Read + WS (Q2-Item)
- Cursor Stability + G-Cursor(x), LU
- Read Committed + G1{a-c}
- Read Uncommitted G0

SNAPSHOT ISOLATION

- Snapshot Isolation + LU
- Read Atomic + PR
- Predicate Cut Isolation + PMP
- Item Cut Isolation IMP

SERIALIZABILITY

- Serializability
```

Atomicity and Isolation Test
Auditing rules

Audit workflow:

- Start from ACID to find problems earlier
- Contract -> Audit -> Review -> Publish

Audited benchmark results:

- Produced by an independent auditor
- Reviewed by Task Force Lead and LDBC
- Published as “LDBC benchmark results”

Auditor selection:

- Independent with no conflict of interest
- Provide COI if needed considering auditors are from vendors
New Chokepoint Example #1

Assuming: A -[e1]-> B -[e2]-> ... -> X

- Timestamp order: e1 < e2 < ... < ei
- Amount order: e1 > e2 > ... > ei
- Time window: ei-1 < ei < ei-1 + Δ

[LANG] Language Features: Recursive path filtering pattern
More flexible expression is wished to support this filtering pattern.
New Chokepoint Example #2

[STORAGE] Data Access Locality: Temporal access locality and performance
Boost the time-window filtering with well-sorted data in storage layer