Graph Pattern Matching in GQL and SQL/PGQ

SIGMOD 2022 (Industry Paper) + LDBC Presentation
Property Graphs

- A data model based on graphs where both nodes and relationships can have properties (attributes) and types (label)
Property Graphs in Industry

- Multiple vendors:
  - Neo4j
  - Oracle
  - TigerGraph
  - Amazon
  - SAP
  - Redis
  - DataStax, etc.

- Widespread: used by 75% of Fortune 100 companies

- Prediction (Gartner): in the next 5 years, up to **80% of all data analytics** tasks will involve graph databases

- Prediction (IDG): Graph database market will experience **600% growth** over the next decade
Querying Property Graphs

- Multiple declarative languages: Cypher, PGQL, GSQL, G-Core, etc...
  - They look like dialects of the same language rather than different ones
- New Standard: **GQL (Graph Query Language)**
  - in development since 2019
- Another standardization project: **SQL/PGQ** (SQL Property Graph Querying):
  - graphs are defined as views over a relational schema
  - in development since 2017
- The engine of a graph query language: **graph pattern matching language (GPML)**
  - shared by GQL and SQL/PGQ
Standards: Process

ISO/IEC JTC1 SC32 WG3

SQL Standard Committee:
9075-16 - SQL/PGQ
39075 - GQL Standard

Linked Data Benchmark Council

Formal Semantics Working Group (FSWG)

analyzes design decisions;
provides feedback
GPML: Nodes

Selecting nodes:

MATCH (x:Account)
WHERE x.isBlocked = 'no'
Selecting nodes:

MATCH (x:Account)
WHERE x.isBlocked = 'no'

- **owner:** Aretha
  - **isBlocked:** no
  - **date:** 2/1/2020
  - **amount:** 10M

- **owner:** Jay
  - **isBlocked:** yes
  - **date:** 3/1/2020
  - **amount:** 10M

- **owner:** Mike
  - **isBlocked:** no
  - **date:** 6/1/2020
  - **amount:** 10M

- **owner:** Dave
  - **isBlocked:** no
  - **date:** 4/1/2020
  - **amount:** 10M

- **owner:** Dave
  - **isBlocked:** no
  - **date:** 7/1/2020
  - **amount:** 5M
All nodes:

MATCH (x)
MATCH (x)-[el:Transfer]->(y)
WHERE x.isBlocked = 'no'
AND y.isBlocked = 'yes'
AND el.amount <= 5M
GPML: Paths

Multiple edge options:  ~, -, ->, <-

Longer paths are defined via ASCII art:

```
MATCH (x)-[:Transfer]->(y)<-[:Transfer]-(z)
WHERE y.isBlocked = 'yes'
```
Specifying graph traversal:

MATCH (x:Account)-[t:Transfer]->{2,5}(y:Account)
WHERE x.isBlocked = 'no' AND y.isBlocked = 'yes'
Specifying graph traversal:

MATCH (x:Account)-[t:Transfer]->{2,5}(y:Account)
WHERE x.isBlocked = 'no' AND y.isBlocked = 'yes'
GPML: Path traversal

Allowed quantifiers: \{m,n\}, *, +

Path conditions can be added:

```
MATCH (x:Account)
  -[t:Transfer WHERE t.amount > 7M]->{2,5}
(y:Account)
WHERE x.isBlocked = 'no' AND y.isBlocked = 'yes'
```
It’s not just single edges that can be repeated:

MATCH (x:Account)
  [ -[t:Transfer WHERE t.amount > 7M]->
    [tl: Transfer WHERE tl.amount > 3M] -> ]{2,5}
  (y:Account)
WHERE x.isBlocked = 'no' AND y.isBlocked = 'yes'
Issues when returning paths:

MATCH
    p = (x WHERE x.owner = 'Mike')
    -[:Transfer]>*
    (y WHERE y.owner = 'Jay')

To deal with this GPML allows restrictors and selectors:

- **restrictors** restrict the set of considered paths to be finite;
- **selectors** filter out the results to assure finiteness.
How do restrictors work?

MATCH SIMPLE
  p = (x WHERE x.owner = 'Mike')
    -[:Transfer]>*
    (y WHERE y.owner = 'Jay')

Also available: TRAIL, ACYCLIC
GPML: Simple, Trail, Shortest

How selectors work?

MATCH ALL SHORTEST

\[ p = (x \text{ WHERE } x.\text{owner} = 'Mike') \]
\[ -[:\text{Transfer}]->* \]
\[ (y \text{ WHERE } y.\text{owner} = 'Jay') \]

Also available: ANY SHORTEST

Can be combined with restrictors
Two types of union: set-based and multiset-based (SQL `UNION` vs `UNION ALL`)

Conditional matches:

```
MATCH (x)-[:Transfer]->(y)[[:Transfer]->(z)]
WHERE y.isBlocked = 'yes'
```

*Transfers to a blocked account, and, if available, all outgoing transfers.*
Finally, we can combine all these into a single query:

\[
\text{MATCH TRAIL } p = (x) -[:Transfer]-> (y), \\
(y) -[:Transfer]->+ (x), \\
(x:Account)-[:isIn]->(c1:City), \\
(y:Account)-[:isIn]->(c2:City) \\
\text{WHERE } c1.name = c2.name \text{ AND } y.isBlocked = 'yes'
\]

Accounts in the same city, with both a direct transfer between them, and also a path that links them in the other direction (i.e. Aretha is laundering money).
Money laundering scheme:

\[
\text{MATCH TRAIL } p = (x) -\text{:Transfer}\rightarrow (y),
\]
\[
(y) -\text{:Transfer}\rightarrow (x),
\]
\[
(x:\text{Account}) -\text{:isIn}\rightarrow (c1:\text{City}),
\]
\[
(y:\text{Account}) -\text{:isIn}\rightarrow (c2:\text{City})
\]

\text{WHERE } c1.\text{name} = c2.\text{name} \text{ AND } y.\text{isBlocked} = 'yes'
GPML: Joins

Money laundering scheme:

MATCH TRAIL  p = (x) -[:Transfer]->  (y),
            (y) -[:Transfer]->+ (x),
            (x:Account)-[:isIn]->(c1:City),
            (y:Account)-[:isIn]->(c2:City)
WHERE c1.name = c2.name AND y.isBlocked = 'yes'
Money laundering scheme:

MATCH TRAIL \( p = (x) -[:Transfer]-> (y), \\
(y) -[:Transfer]->+ (x), \\
(x:Account)-[:isIn]->(c1:City), \\
(y:Account)-[:isIn]->(c2:City) \\
WHERE c1.name = c2.name AND y.isBlocked = 'yes' \)
GPML: Output

GPML output: a data structure that combines paths in graphs with bindings of variables.
Can be embedded in GQL and in SQL/PGQ.

Graph pattern → GPML processor → output
Graph DB

SQL/PGQ
output

table
graph view
new graph

GQL
# Timeline to Standards

<table>
<thead>
<tr>
<th>Date</th>
<th>SQL/PGQ</th>
<th>GQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017</td>
<td>Work started</td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td></td>
<td>Work started</td>
</tr>
<tr>
<td>2021-02-07</td>
<td>CD Ballot End</td>
<td></td>
</tr>
<tr>
<td>2022-02-20</td>
<td></td>
<td>CD Ballot End</td>
</tr>
<tr>
<td>2022-12-04</td>
<td>DIS Ballot End</td>
<td></td>
</tr>
<tr>
<td>2023-01-30</td>
<td>Final Text to ISO</td>
<td></td>
</tr>
<tr>
<td>2023-03-13</td>
<td><strong>SQL/PGQ IS Published</strong></td>
<td></td>
</tr>
<tr>
<td>2023-05-21</td>
<td></td>
<td>DIS Ballot End</td>
</tr>
<tr>
<td>2023-07-30</td>
<td></td>
<td>Final Text to ISO</td>
</tr>
<tr>
<td>2023-09-10</td>
<td></td>
<td><strong>GQL IS Published</strong></td>
</tr>
</tbody>
</table>
Research Challenges

• Find a workable abstraction of GPML for research (systems and theory).

• Support aggregation

```
MATCH (x)-[e:Flight]->*(y)
WHERE x.name='Zembla'
AND y.name='Ankh-Morpork'
AND SUM(e.duration) < 24
```

• Optimize GMPL processing (vendors already working on it).