Building A **Conceptual Framework** for Network Analytics

Minjian Liu (u5506264)
Supervised by Dr. Qing Wang

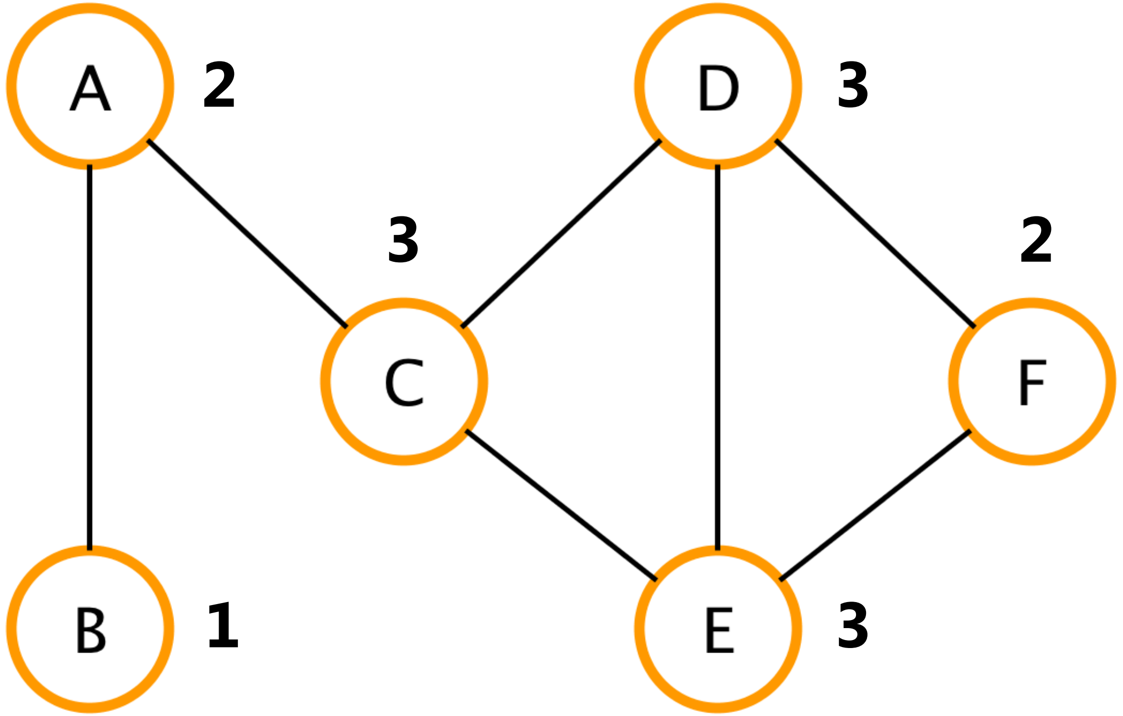
OUTLINE


Motivation
Data Model
Query Language
Query Engine
Performance

MOTIVATION

Ranking: the most important vertex

VertexID	Degree
C	***3***
D	***3***
E	***3***
A	2
F	2
B	1

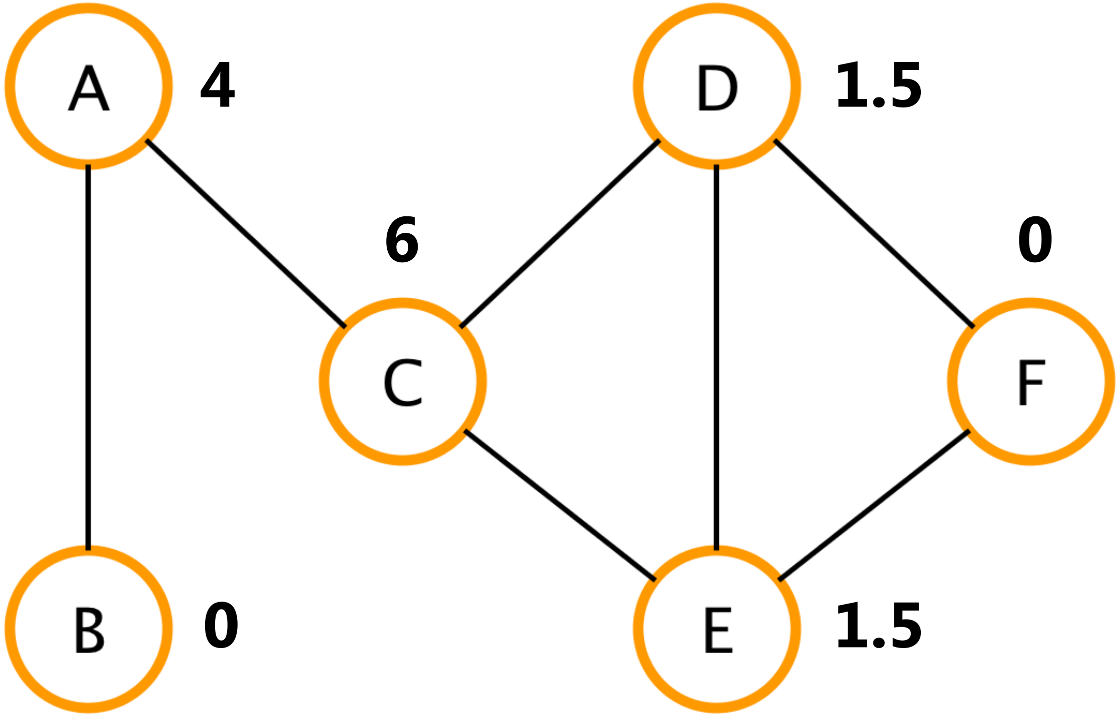


MOTIVATION

Ranking: the most important vertex

VertexID	Degree
C	***3***
D	***3***
E	***3***
A	2
F	2
B	1

VertexID	Betweenness
C	***6***
D	1.5
E	1.5
A	4
F	0
B	0

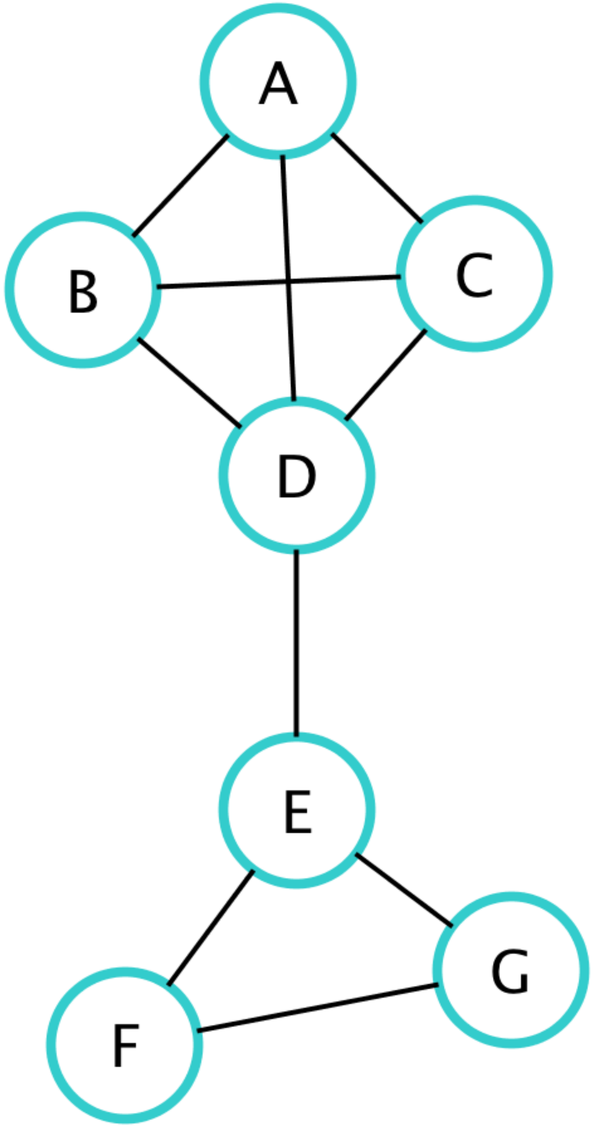


MOTIVATION

Clustering: a group of vertices

Connected Component

ClusterID	Size	Members
1	7	{A, B, C, D, E, F, G}



MOTIVATION

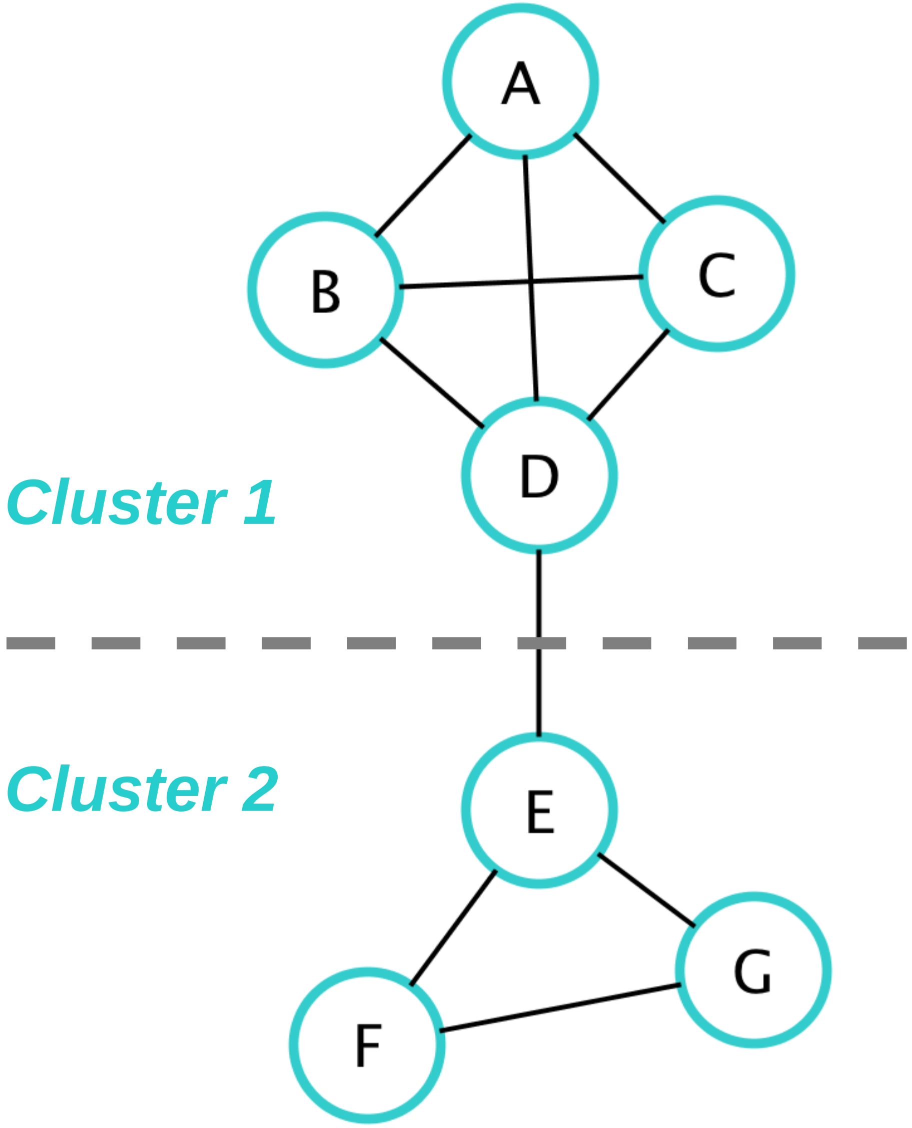
Clustering: a group of vertices

Connected Component

ClusterID	Size	Members
1	7	{A, B, C, D, E, F, G}

Community Detection

ClusterID	Size	Members
1	4	{A, B, C, D}
2	3	{E, F, G}

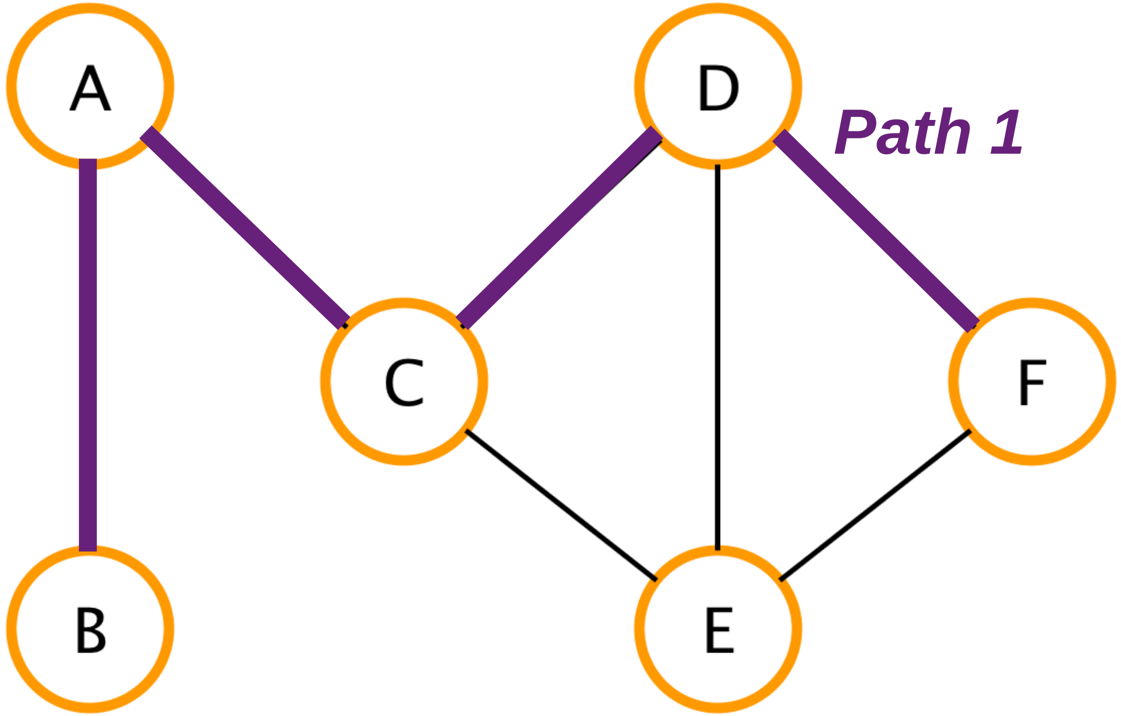


MOTIVATION

Path Finding: a sequence of vertices

A Path between B and F with length 4

PathID	Length	Path
1	4	<B, A, C, D, F>
2	4	<B, A, C, E, F>

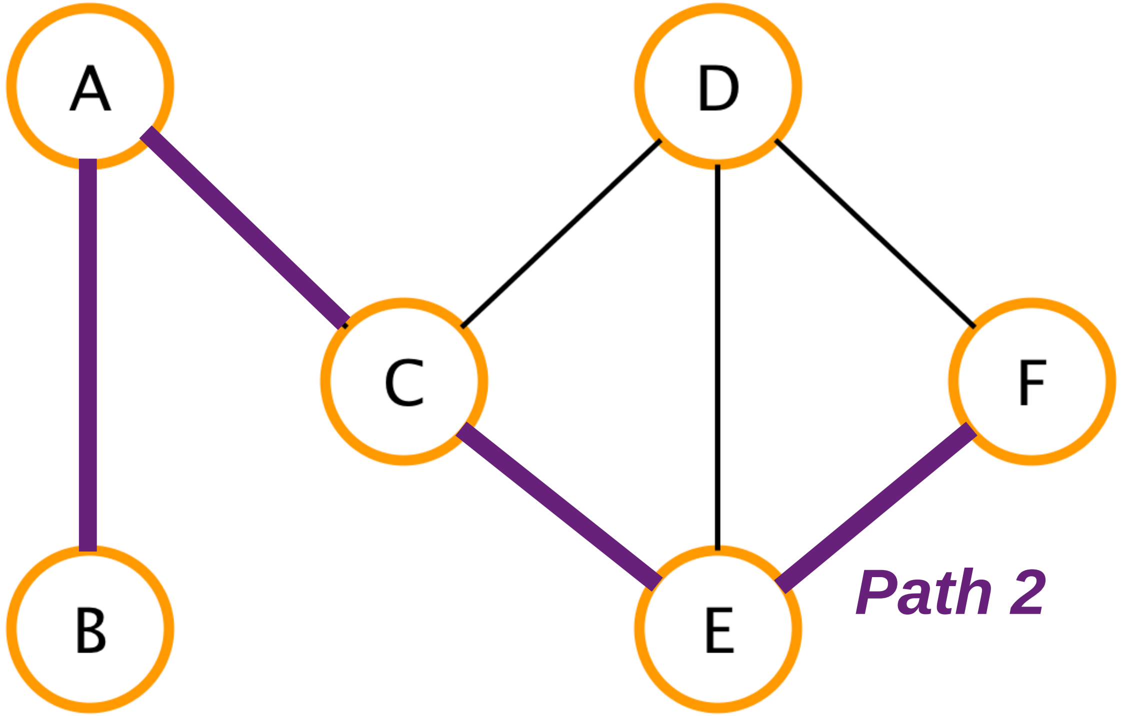


MOTIVATION

Path Finding: a sequence of vertices

A Path between B and F with length 4

PathID	Length	Path
1	4	<B, A, C, D, F>
2	4	<B, A, C, E, F>



MOTIVATION

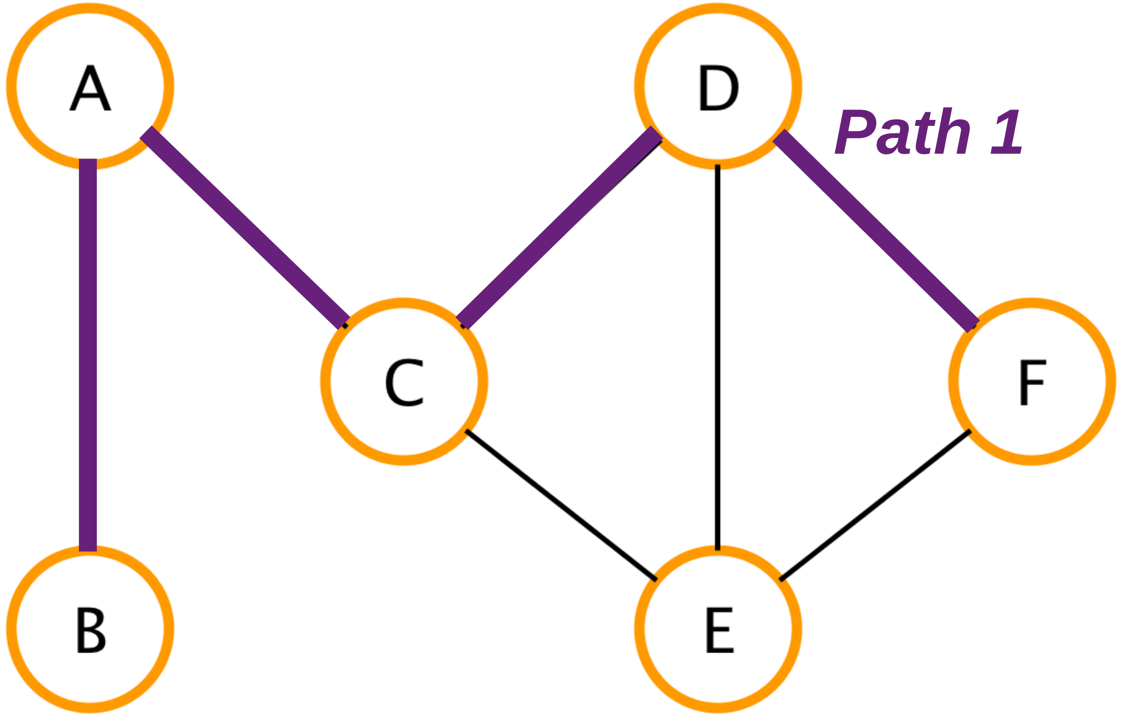
Path Finding: a sequence of vertices

A Path between B and F with length 4

PathID	Length	Path
1	4	<B, A, C, D, F>
2	4	<B, A, C, E, F>

A Path between B and F with D in the middle

PathID	Length	Path
1	4	<B, A, C, D, F>

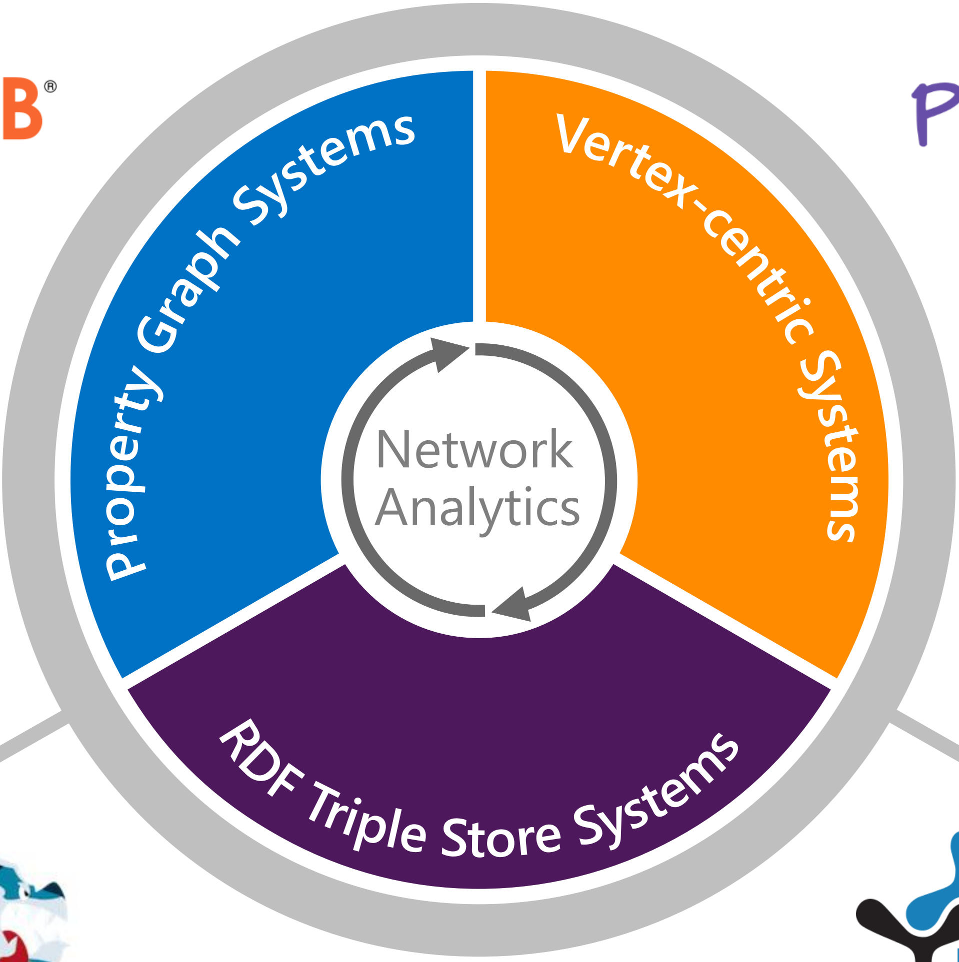


MOTIVATION

OrientDB

TITAN

neo4j



Pregelix

APACHE GIRAPH

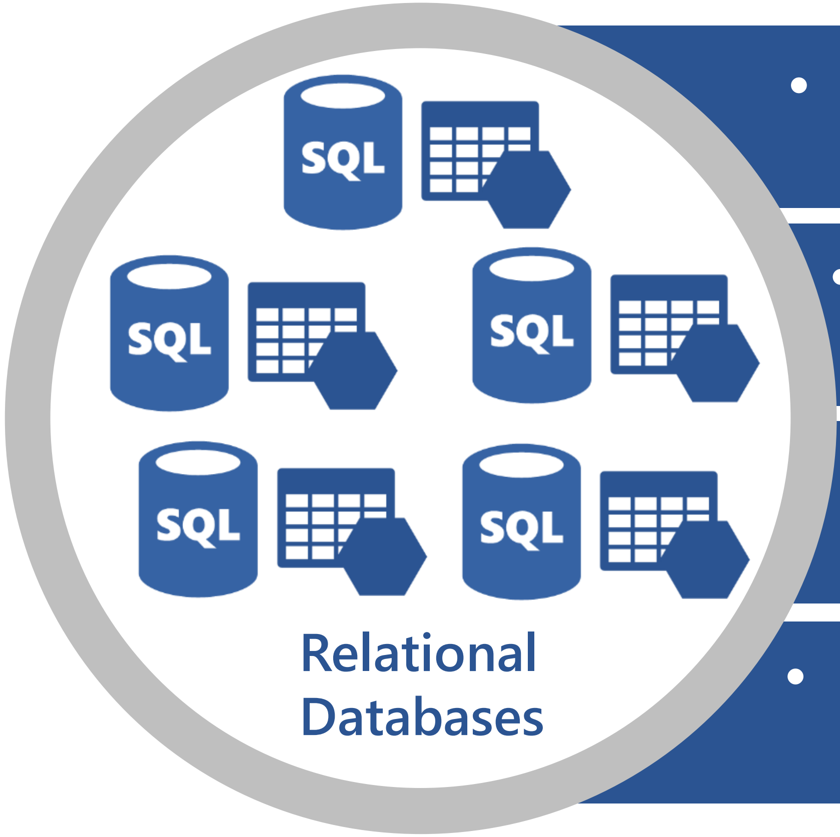
GraphLab

Stardog

AllegroGraph
Franz Inc.

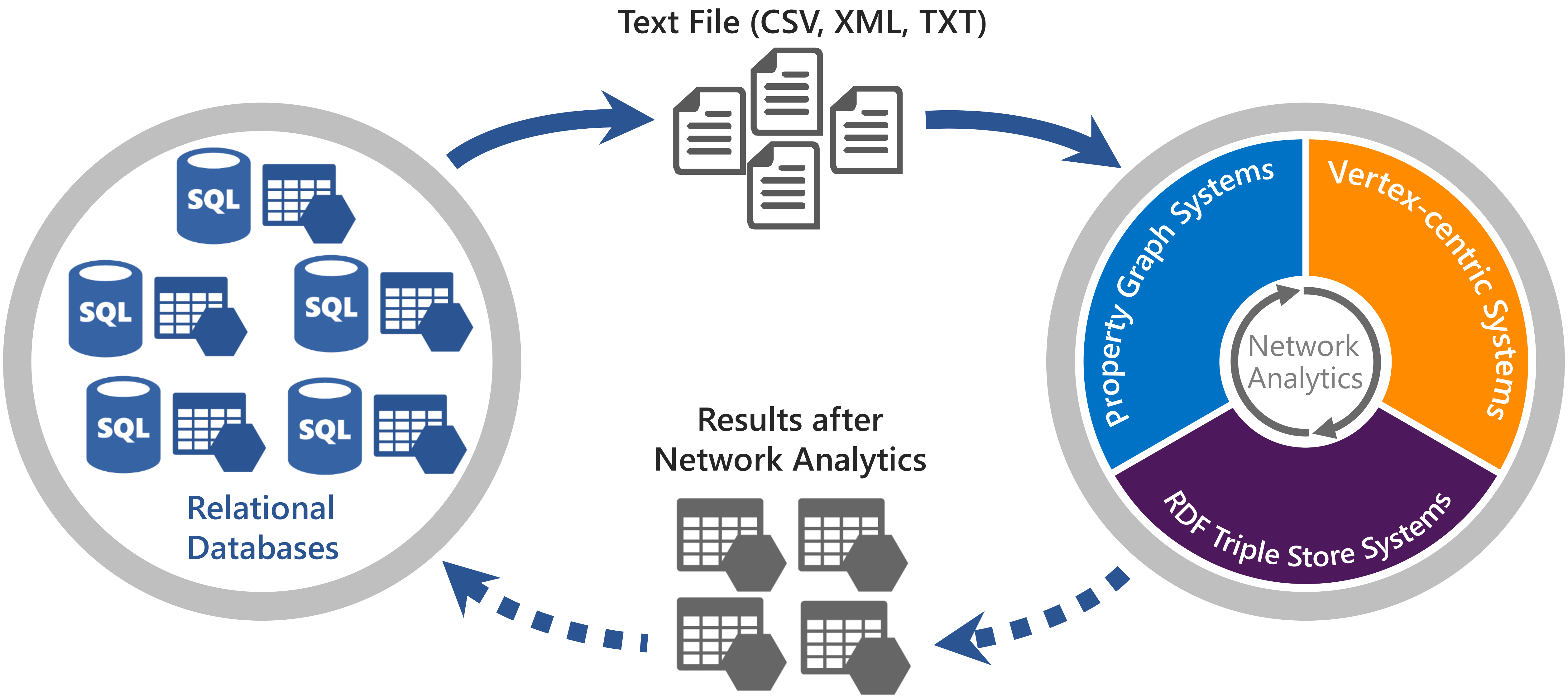
Jena

MOTIVATION



- They are still widely used by enterprises and organizations
- They are mature and battle-tested with a stable code base
- People are familiar with using SQL to manipulate data in relational databases
- They provide a number of sophisticated optimisation technologies (e.g. indexing, materialized views)

MOTIVATION



Text File (CSV, XML, TXT)
SQL
SQL
SQL
SQL
SQL
Relational
Databases
Property Graph Systems
Vertex-centric Systems
RDF Triple Store Systems
Network
Analytics
Results after
Network Analytics

MOTIVATION

Data Model

Extends the relational model with network analytics capability

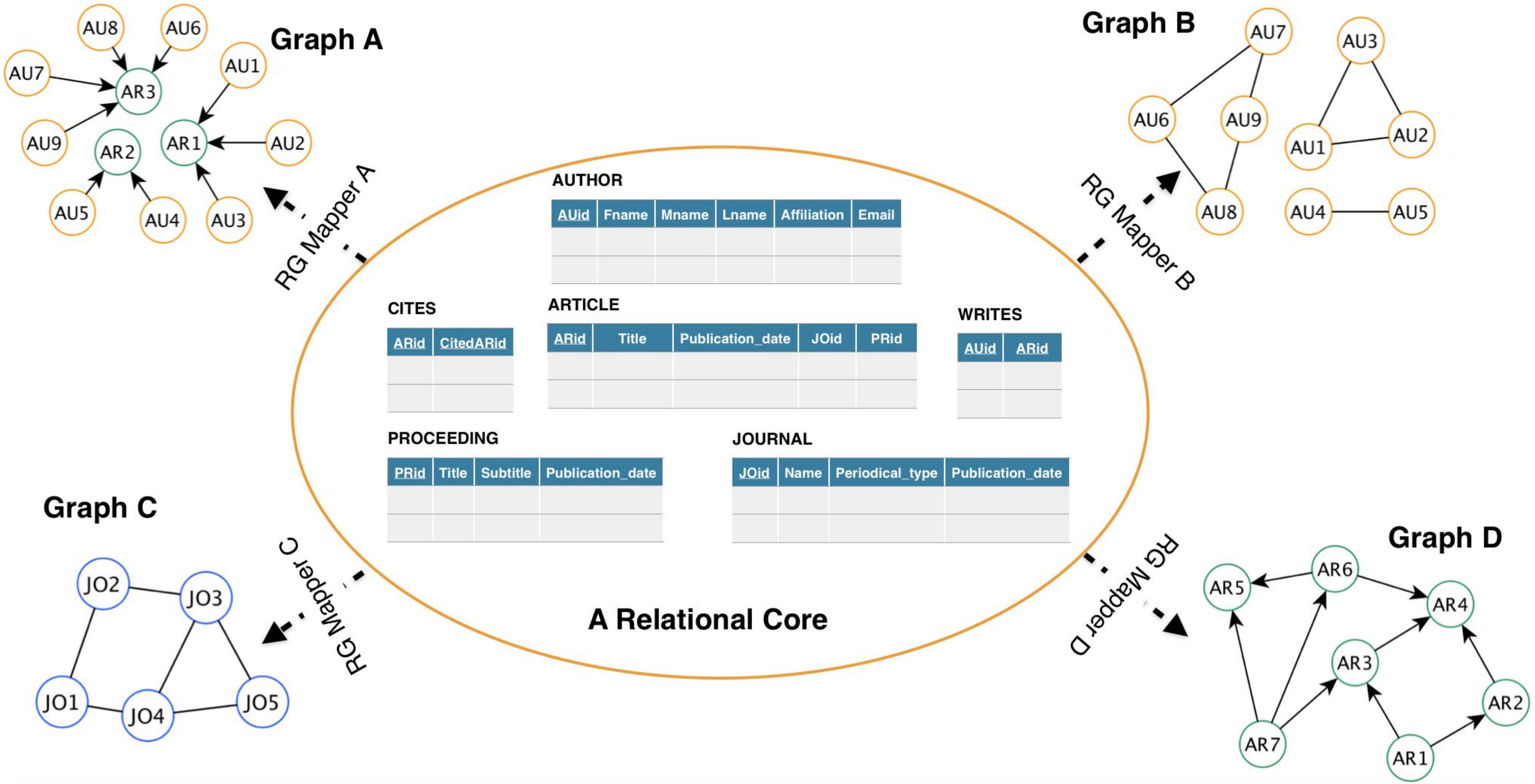
- Relational Core
- Graphical Views
- RG Mapper

Query Language

Follows SQL-style syntax to manipulate both schema and network data

- Rank
- Cluster
- Path

DATA MODEL


Graph A
AU8
AU6
AU7
AR3
AU1
AU9
AR2
AR1
AU2
AU5
AU4
AU3
RG Mapper A
Graph B
AU7
AU3
AU6
AU9
AU1
AU2
AU8
AU4
AU5
RG Mapper B
AUTHOR
AUid | Fname | Mname | Lname | Affiliation | Email
CITES
ARid | CitedARid
ARTICLE
ARid | Title | Publication_date | JOid | PRid
WRITES
AUid | ARid
PROCEEDING
PRid | Title | Subtitle | Publication_date
JOURNAL
JOid | Name | Periodical_type | Publication_date
A Relational Core
Graph C
JO2
JO3
JO1
JO4
JO5
RG Mapper C
Graph D
AR6
AR5
AR4
AR3
AR2
AR7
AR1
RG Mapper D

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Find top 3 influential authors in the co-authorship network.

```
SELECT VertexID, Value
FROM RANK (coauthorship, betweenness)
LIMIT 3;
```

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Find top 3 influential authors in the co-authorship network.

```
SELECT VertexID, Value
FROM RANK (coauthorship, betweenness)
LIMIT 3;
```

Graph Operator

Graph Name

degree; indegree
outdegree; pagerank
betweenness; closeness

Ranking Measure

VertexID	Value
AU8	6
AU3	1.5
AU9	1.5

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Find the biggest community that consists of authors who collaborate with each other to publish articles together.

```
SELECT ClusterID, Size, Members
FROM CLUSTER (coauthorship, GN)
ORDER BY Size DESC
LIMIT 1;
```

QUERY LANGUAGE

Ranking:
Find the most
important vertex.

Clustering:
Find a group
of vertices.

Path Finding:
Find a sequence
of vertices.

Create Graph:
Use a RG mapper
to create graph.

Find the biggest community that consists of authors who collaborate with each other to publish articles together.

SELECT ClusterID, Size, Members
FROM CLUSTER (coauthorship, GN)
ORDER BY Size **DESC**
LIMIT 1;

Graph Operator

Graph Name

CC (Connected Component)
SCC (Strongly Connected Component)
GN (Girvan-Newman algorithm)
CNM (Clauset-Newman-Moore Algorithm)
MC (Monte Carlo Algorithm)

Clustering Algorithm

ClusterID	Size	Members
2	4	{AU1, AU3, AU8, AU9}

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Find shortest paths between two authors V1 and V2, author V1 is affiliated at ANU and author V2 is affiliated at NICTA.

```
SELECT PathID, Length, Path
FROM PATH (coauthorship, V1// V2)
WHERE V1 AS
(
    SELECT AUid FROM AUTHOR
    WHERE Affiliation = ‘ANU’
) AND V2 AS
(
    SELECT AUid FROM AUTHOR
    WHERE Affiliation = ‘NICTA’
)
ORDER BY Length ASC;
```

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Find shortest paths between two authors V1 and V2, author V1 is affiliated at ANU and author V2 is affiliated at NICTA.

```
SELECT PathID, Length, Path
FROM PATH (coauthorship, V1// V2)
WHERE V1 AS
(
    SELECT AUid FROM AUTHOR
    WHERE Affiliation = 'ANU'
) AND V2 AS
(
    SELECT AUid FROM AUTHOR
    WHERE Affiliation = 'NICTA'
)
ORDER BY Length ASC;
```

Graph Operator

Graph Name

Path Expression

V1 / . / . / V2
V1 // V2
V1 / . / V2 / . / V3
V1 // V2 // V3

PathID	Length	Path
1	1	<AU1, AU8>

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Create a materialized graph for co-authorship network.

UNGRAPH (Undirected Graph)
DIGRAPH (Directed Graph)

Graph Type

Graph Name

```
CREATE UNGRAPH coauthorship AS
(
    SELECT w1.AUid, w2.AUid
    FROM WRITES as w1, WRITES as w2
    WHERE w1.ARid = w2.ARid AND w1.AUid != w2.AUid
);
```

RG Mapper

QUERY LANGUAGE

Ranking:
Find the most important vertex.

Clustering:
Find a group of vertices.

Path Finding:
Find a sequence of vertices.

Create Graph:
Use a RG mapper to create graph.

Create a citation graph on-the-fly for ranking.

UNGRAPH (Undirected Graph)
DIGRAPH (Directed Graph)

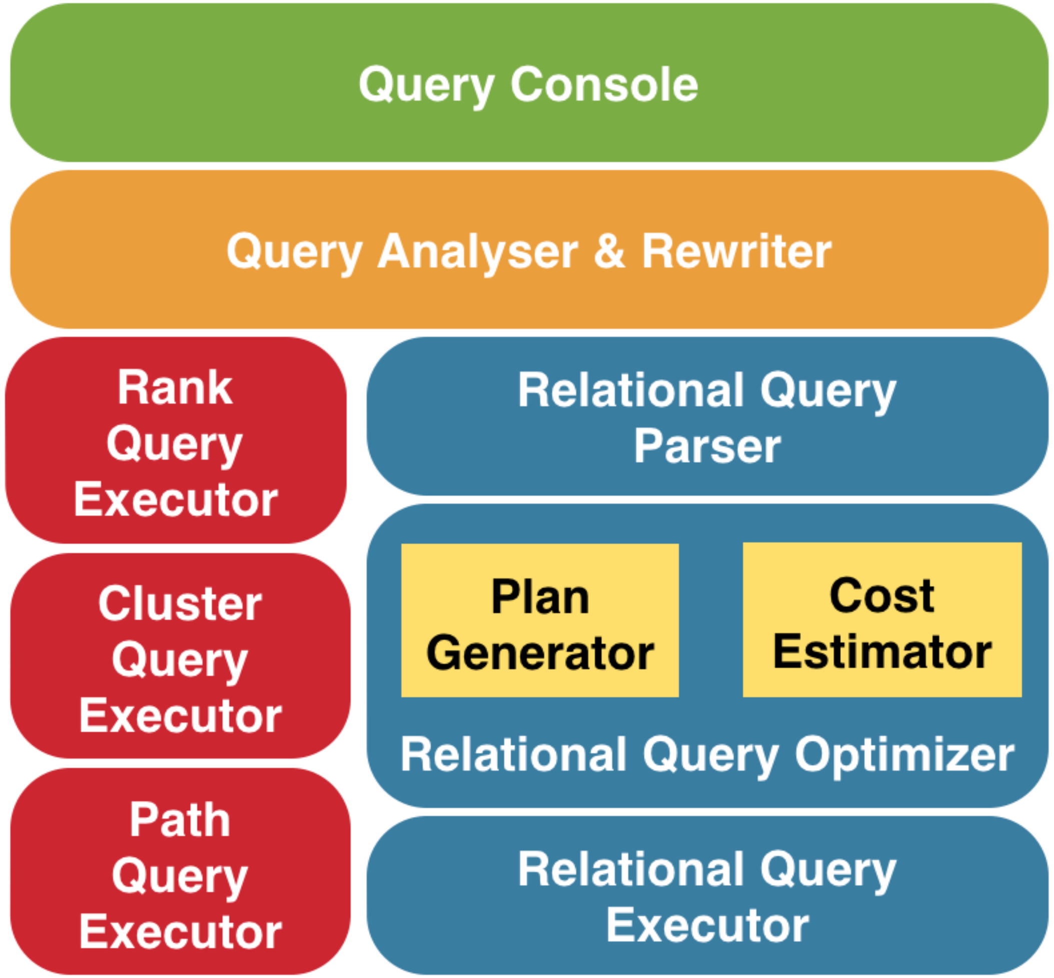
Graph Type

Graph Name

```
SELECT VertexID, Value
FROM RANK (citation, indegree)
WHERE citation IS DIGRAPH AS
(
    SELECT ARid, CitedARid FROM CITES
);
```

RG Mapper

QUERY ENGINE



Relation-Graph Engine (RG Engine)

Query Console

- Allows user enter queries
- Shows the query results

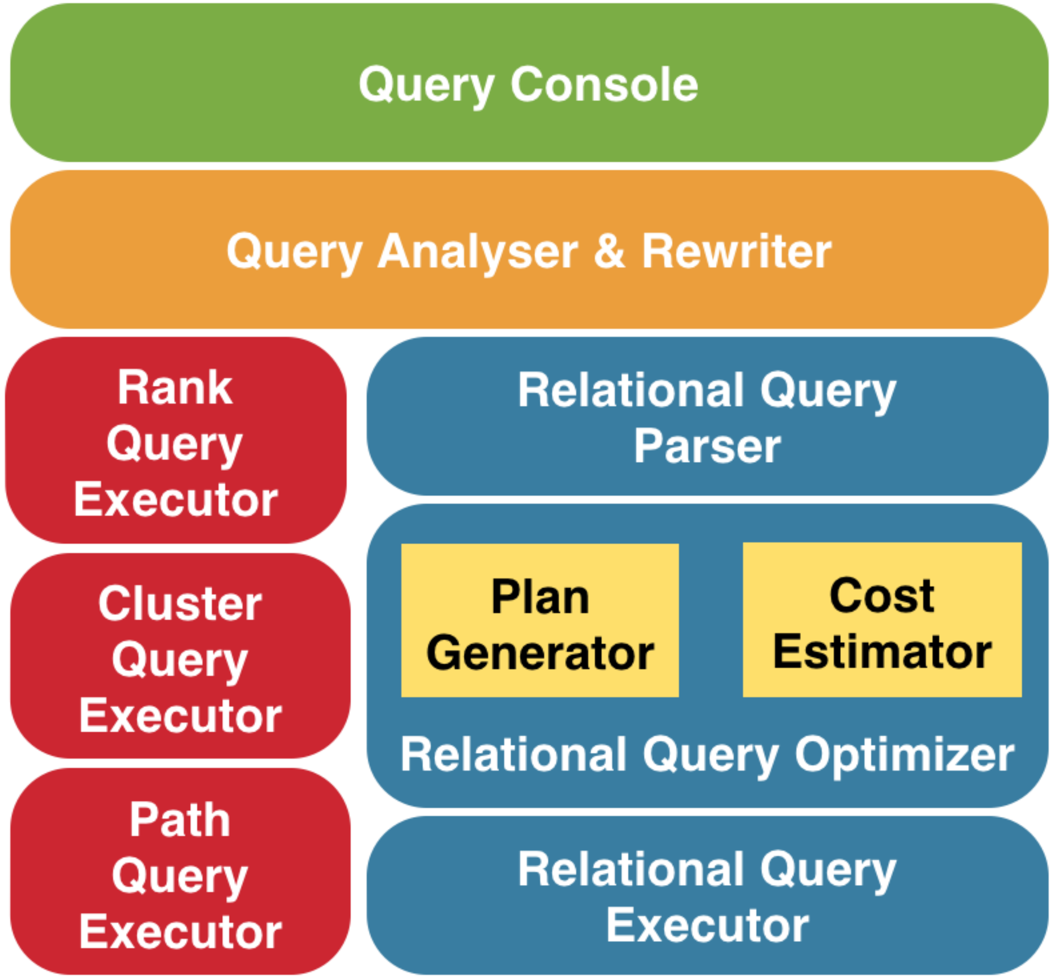
Query Analyser & Rewriter

- Checks whether or not the query syntax is correct
- Differentiates graph sub-queries and relational sub-queries
- Rewrites graph sub-queries

Graph Query Components

- Three query executors execute three types of network analytics tasks (ranking, clustering, path finding).
- They transform the results into temporary tables stored in data storage.

QUERY ENGINE

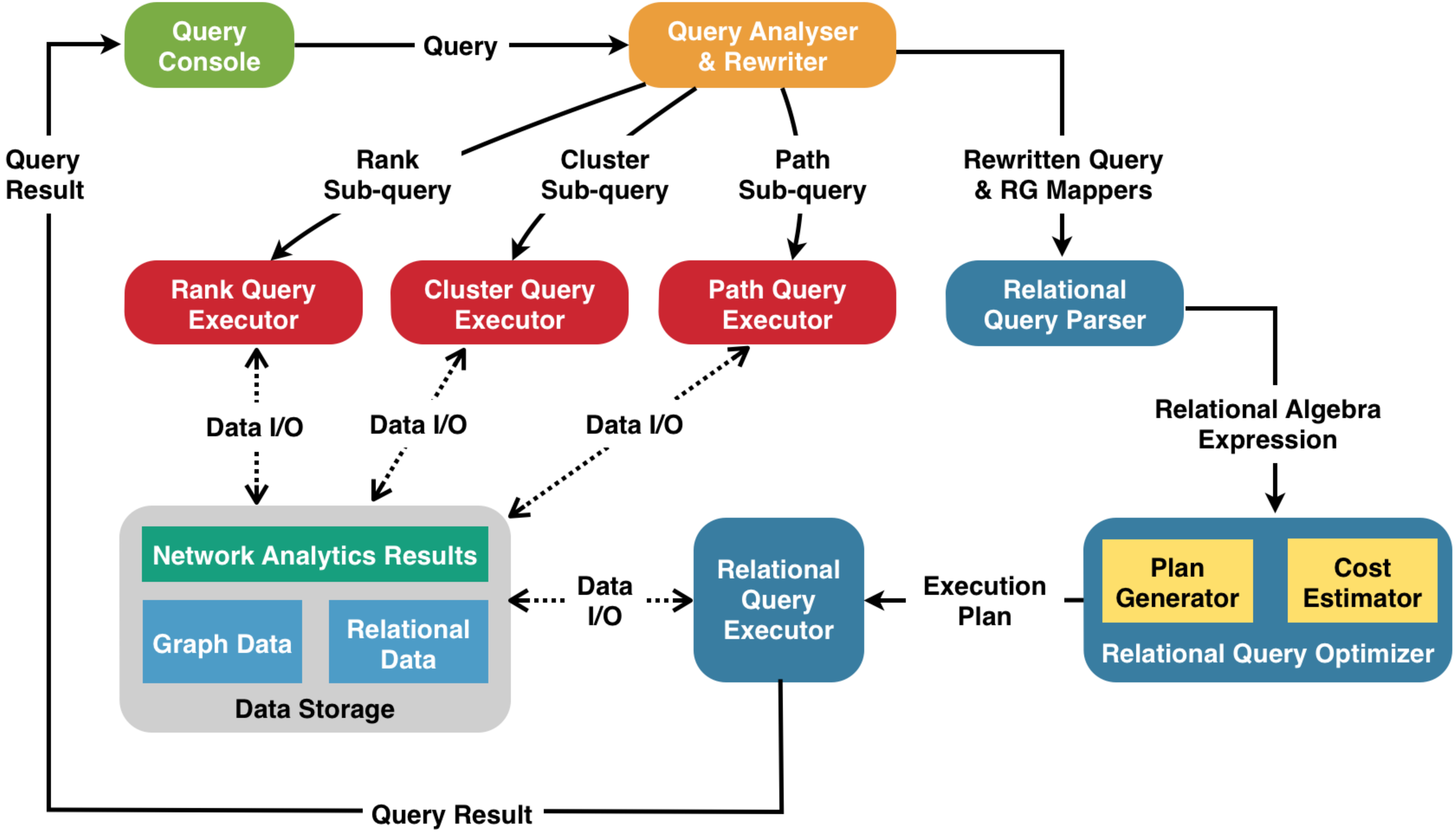


Relation-Graph Engine (RG Engine)

Relational Query Components

- **Parser** checks if operators are consistent with data types and converts the query into relational algebra expression.
- **Optimiser** generates alternative execution plans based on the relational algebra expression, estimates the cost for a subset of plans and chooses the best plan.
- **Executor** executes the best plan and returns results to the query console

QUERY ENGINE



PERFORMANCE

- We adopt 3 graph analysis tools including SNAP, NetworkX, Graph-tool as algorithm support for our query executors.

Functions of Graph Analysis Tools

	Algorithms	**SNAP**	**NetworkX**	**GraphTool**
Ranking	Degree	✓	✓	✓
	Pagerank	✓	✓	✓
	Betweenness	✓	✓	✓
	Closeness	✓	✓	✓
Clustering	Connected Component(CC)	✓	✓	✓
	Strongly Connected Component(SCC)	✓	✓	✓
	Girvan-Newman (GN)	✓	–	–
	Clauset-Newman-Moore (CNM)	✓	–	–
	Monte Carlo (MC)	–	–	✓
Path Finding	Shortest Path	–	✓	–
	Path with Specific Length	–	✓	–

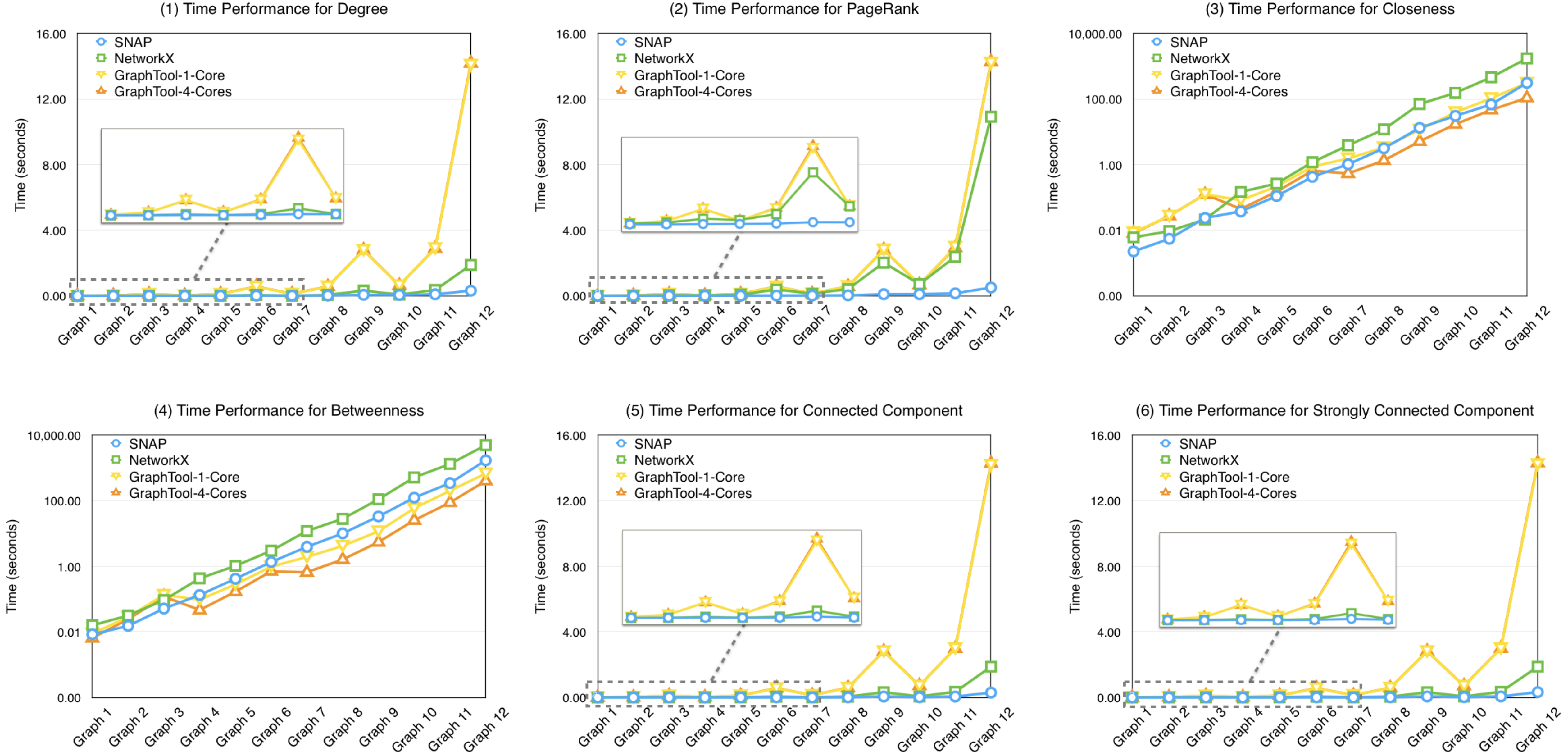
PERFORMANCE

- We adopt 3 graph analysis tools including SNAP, NetworkX, Graph-tool as algorithm support for our query executors.
- For the first part of experiment, we use Erdos-Renyi method to create random graph as the experiment input.

Erdos-Renyi Random Graph

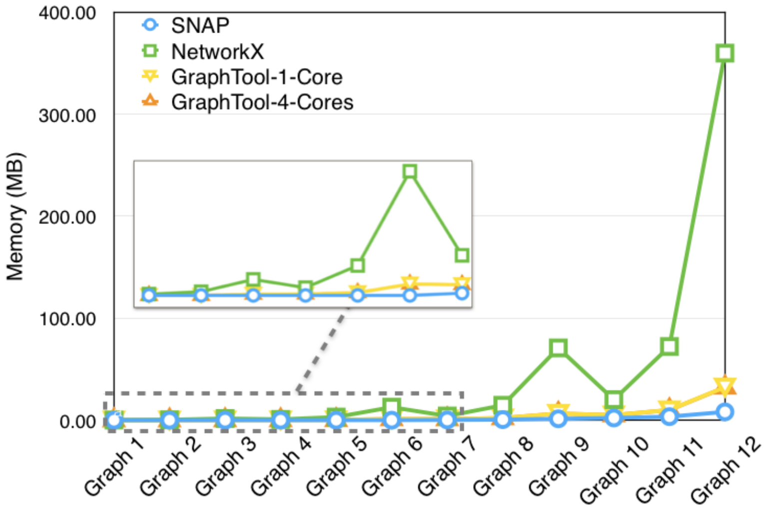
	Number of Vertices	Number of Edges	Size(KB)
Graph 1	100	200	1
Graph 2	100	1,000	6
Graph 3	100	5,000	29
Graph 4	500	1,000	8
Graph 5	500	5,000	38
Graph 6	500	25,000	189
Graph 7	2,500	5,000	46
Graph 8	2,500	25,000	228
Graph 9	2,500	125,000	1,100
Graph 10	12,500	25,000	256
Graph 11	12,500	125,000	1,300
Graph 12	12,500	625,000	6,400

PERFORMANCE

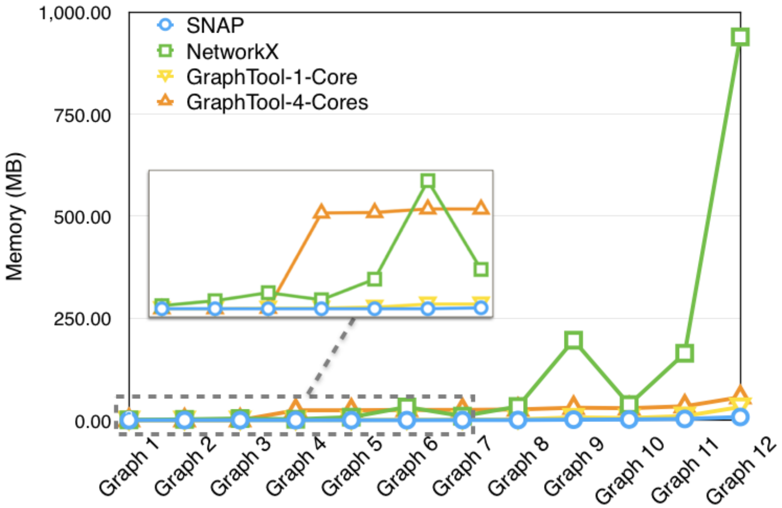


PERFORMANCE

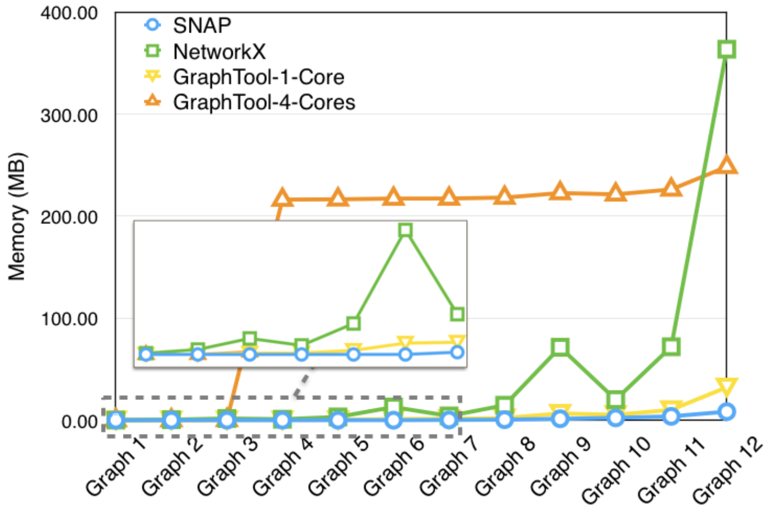
(1) Memory Performance for Degree



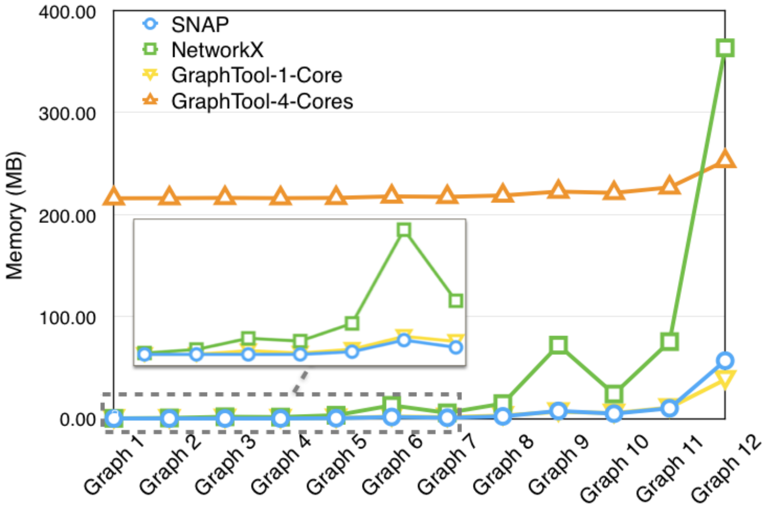
(2) Memory Performance for PageRank



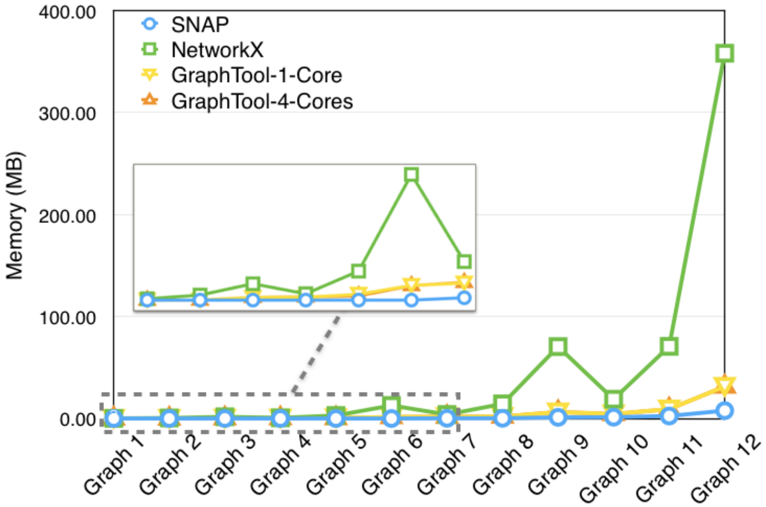
(3) Memory Performance for Closeness



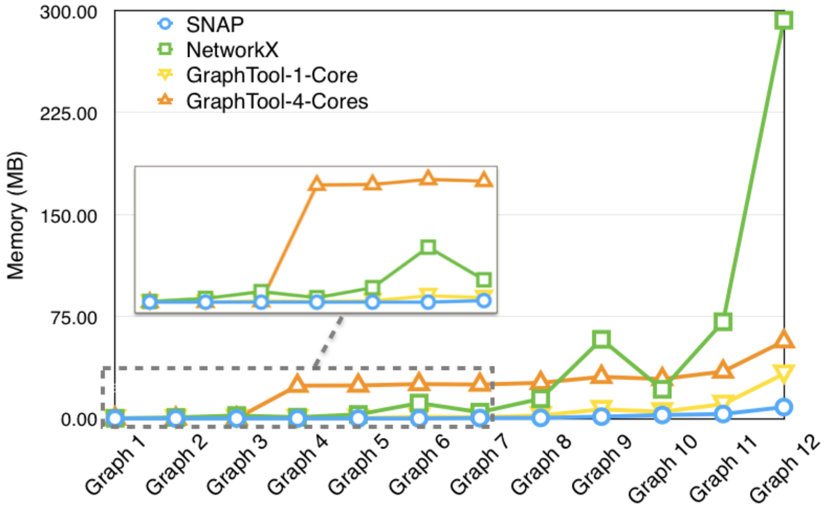
(4) Memory Performance for Betweenness



(5) Memory Performance for Connected Component



(6) Memory Performance for Strongly Connected Component

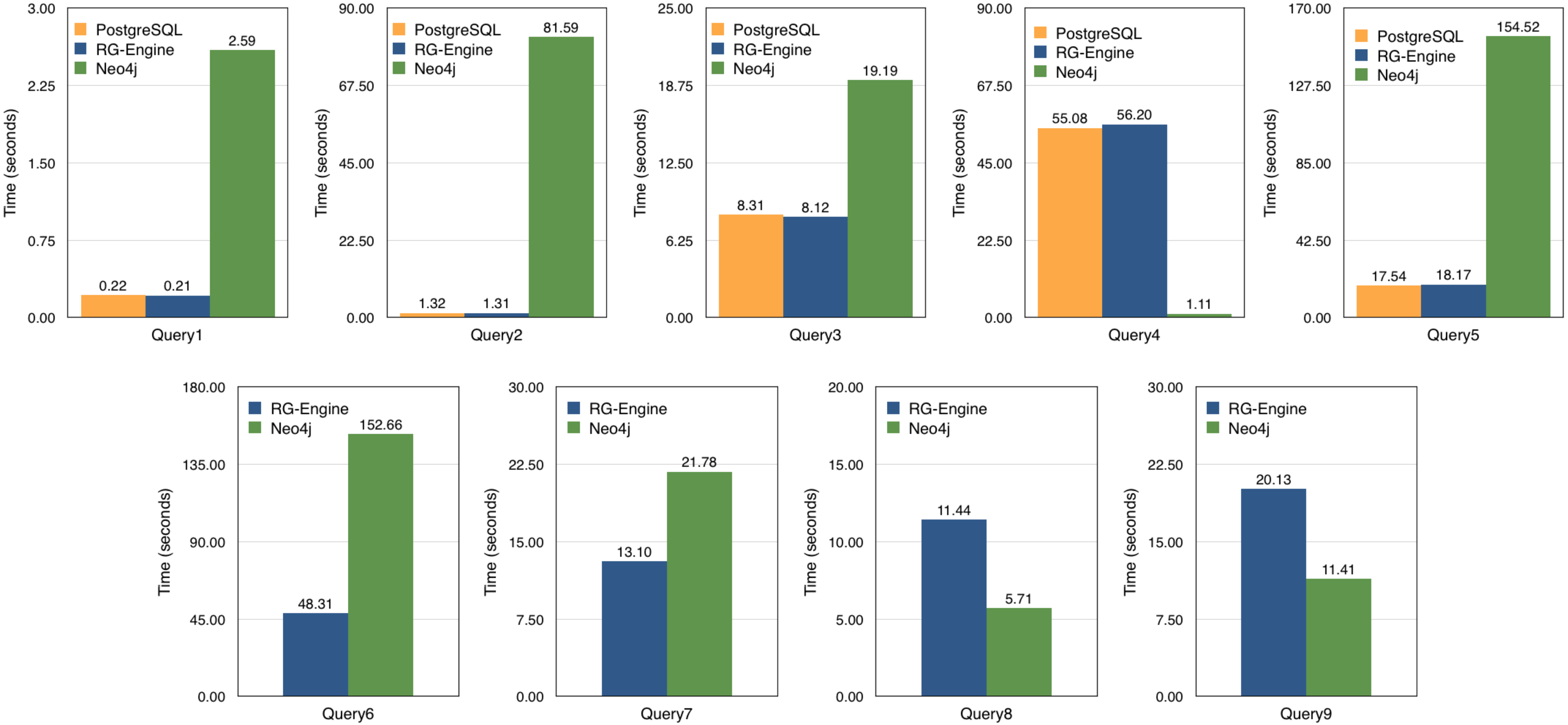


PERFORMANCE

- **Queries 1 – 3** are traditional relational queries including join operations and aggregate operations.
- **Queries 4 – 10** are about some typical network analytics tasks including pattern matching, triangle counting, pagerank centrality, connected component, path finding and community detection
- **Queries 11 – 12** are advanced queries that are nested with two different types of network analytics tasks.

		PSQL	RG-Engine	Neo4j
Query 1	Aggregate Operation + Set Operation	✓	✓	✓
Query 2	Aggregate Operation + Sorting Operation	✓	✓	✓
Query 3	Aggregate Operation + Sorting Operation + Join Operation	✓	✓	✓
Query 4	Pattern Mathcing	✓	✓	✓
Query 5	Triangle Counting	✓	✓	✓
Query 6	PageRank Centrality	—	✓	✓
Query 7	Connected Component	—	✓	✓
Query 8	Path Finding	—	✓	✓
Query 9	Shortest Path	—	✓	✓
Query 10	Community Detection	—	✓	—
Query 11	PageRank Centrality + Connected Component	—	✓	—
Query 12	PageRank Centrality + Path Finding	—	✓	—

PERFORMANCE



Thank You !

Q & A

Contact:

Minjian Liu

0426839321

378610682

u5506264@anu.edu.au

Some of icons are downloaded from www.rapidbbs.cn through its membership services.

The source code and query samples are available at https://gitlab.com/Minken/COMP8800_Computing_Research_Project.git