3612/7204 ICT Database Systems

Graph Databases

David Meibusch
Principal Software Engineer
Oracle Labs Australia

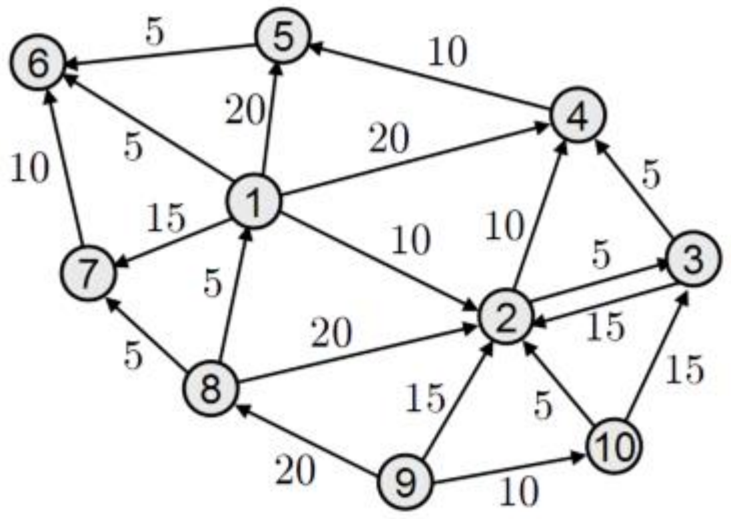
May, 2016

Disclaimer

The following is intended to provide some insight into a line of research in Oracle Labs. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Oracle reserves the right to alter its development plans and practices at any time, and the development, release, and timing of any features or functionality described in connection with any Oracle product or service remains at the sole discretion of Oracle. Any views expressed in this presentation are my own and do not necessarily reflect the views of Oracle.

Graph Theory

- Nodes (vertices)
- Relationships (edges)



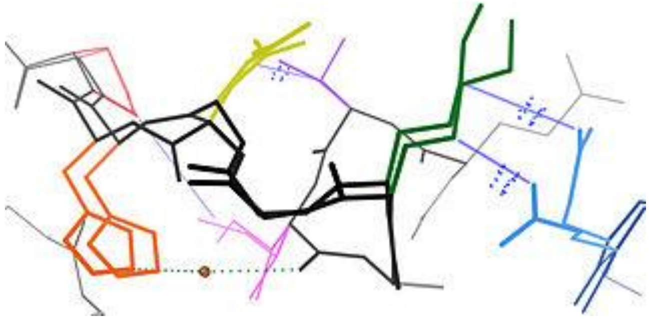
Mathematical properties

- Cyclic, acyclic
- Directed, undirected
- Reachability
- Transitive closures
- Shortest paths

Applications of Graph Theory

© Koushikchakraborty13 - Wikimedia Commons CC-BY-SA-3.0

© SURENDAR KUMAR - Wikimedia Commons CC-BY-SA-3.0



© Daniel Keedy - Wikimedia Commons CC-BY-SA-3.0

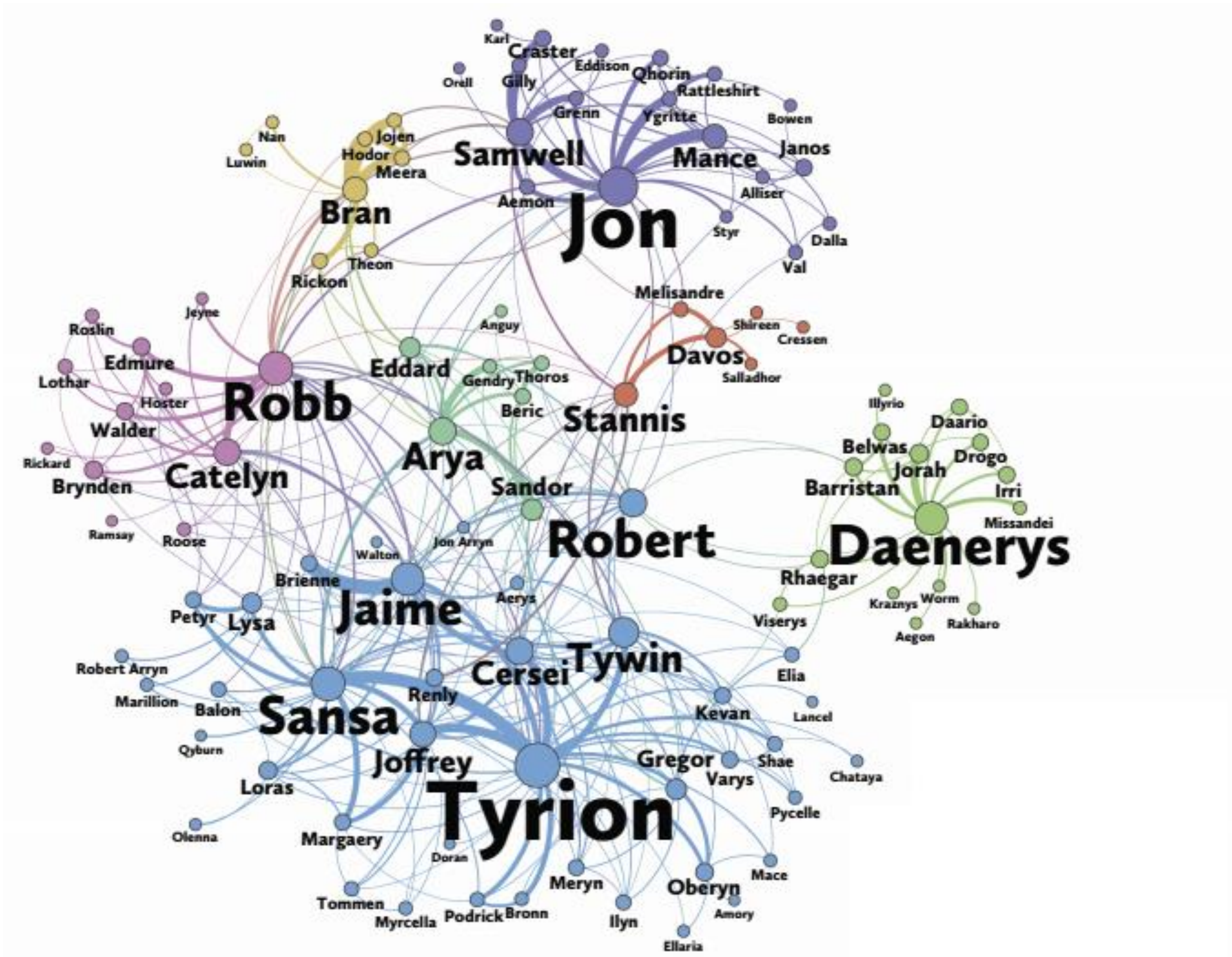
Frappé

© YYK - Wikimedia Commons CC-BY-SA-3.0

Graph Algorithms

Centrality

- Degree Centrality
- Closeness Centrality
- Betweenness Centrality
- PageRank
- …



Network of Thrones © The Mathematical Association of America, 2016 - http://dx.doi.org/10.4169/mathhorizons.23.4.18

Modelling Graph in Relational DB

Node and Edge tables

id	name
1	Tyrion
2	Cersei
3	Sansa
4	Robb
5	Stannis

type	start	end
talked-to	1	2
talked-to	1	3
talked-to	3	4
fought	1	4
fought	2	5

ORACLE

Modelling Graph in Relational DB

Tyrion has talked to…

id	name
1	Tyrion
2	Cersei
3	Sansa
4	Robb
5	Stannis

type	start	end
talked-to	1	2
talked-to	1	3
talked-to	3	4
fought	1	4
fought	2	5

```
SELECT n.name
FROM node n
LEFT JOIN edge e ON n.id = e.end
WHERE e.start = 1;
```

Modelling Graph in Relational DB

Tyrion conversations

id	name
1	Tyrion
2	Cersei
3	Sansa
4	Robb
5	Stannis

type	from	to
talked-to	1	2
talked-to	1	3
talked-to	3	4
fought	1	4
fought	2	5

```
SELECT n.name
WHERE id IN (
 SELECT start FROM edge WHERE end = 1
 UNION
 SELECT end FROM edge WHERE start = 1
);
```

Modelling Graph in Relational DB

Tyrion Chinese Whispers

id	name
1	Tyrion
2	Cersei
3	Sansa
4	Robb
5	Stannis

type	from	to
talked-to	1	2
talked-to	1	3
talked-to	3	4
fought	1	4
fought	2	5

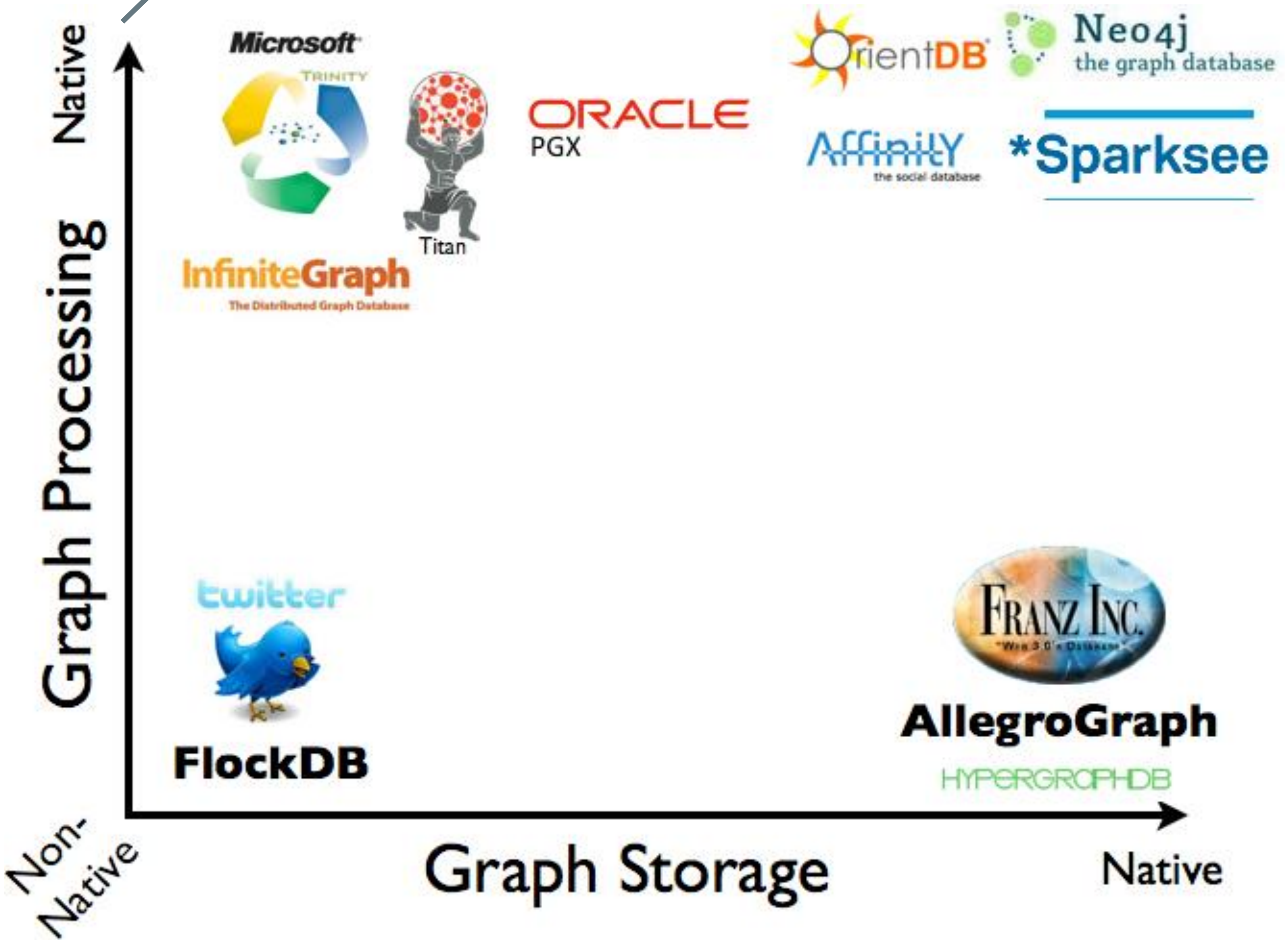
```
WITH RECURSIVE closure(id, name) AS
(
 SELECT n.id, n.name FROM node
 WHERE n.id = 1
UNION ALL
 SELECT n.id, n.name FROM node n
 LEFT JOIN edge e ON n.id = e.end
 WHERE e.start = id
)
SELECT * FROM closure;
```

Graph Database

- Use nodes, edges, and properties to represent and store data
- Provide means to traverse the graph
- Allows fast implementation of graph algorithms

Graph Databases

Index free adjacency



Graph Databases © Neo Technology, 2015 - http://graphdatabases.com/

Neo4J

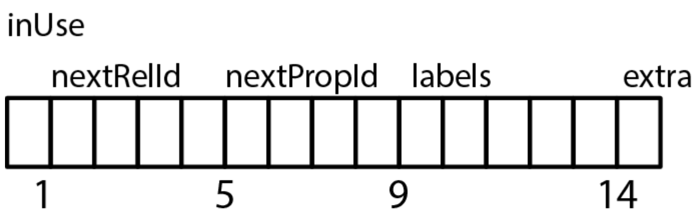
- Graph on-disk with in-memory cache
- CRUD
  - Create, Read, Update, Delete
- ACID transactions
- OLTP

- Labelled Property Graph Model
  - Node and relationships
  - Node/value pairs (properties)
  - Nodes can be labelled
  - Relationships are named and directed
  - Relationship/value pairs (properties)

Neo4J Implementation

- Node and Relationship store files (on disk)

Node (15 bytes)

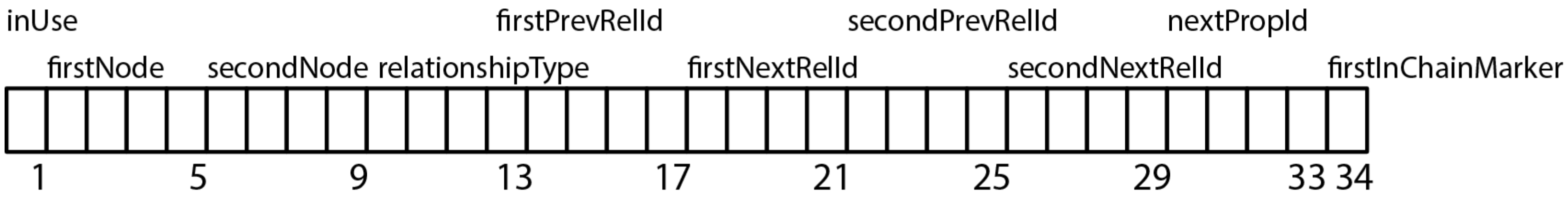


inUse

nextRelId nextPropId labels extra

1 5 9 14

Relationship (34 bytes)



inUse firstPrevRelId secondPrevRelId nextPropId

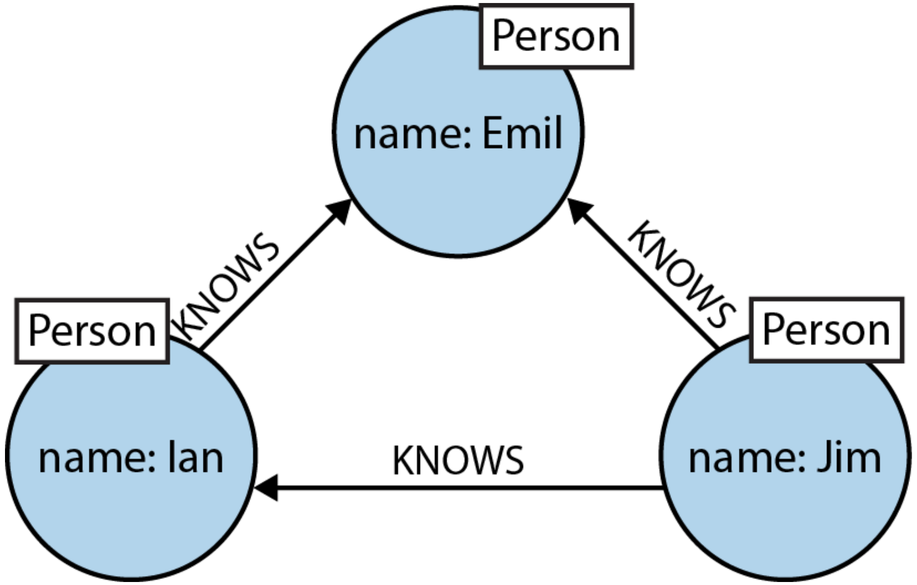
firstNode secondNode relationshipType firstNextRelId secondNextRelId firstInChainMarker

1 5 9 13 17 21 25 29 33 34

Accessing Neo4J graph

- Java core API, traversal API

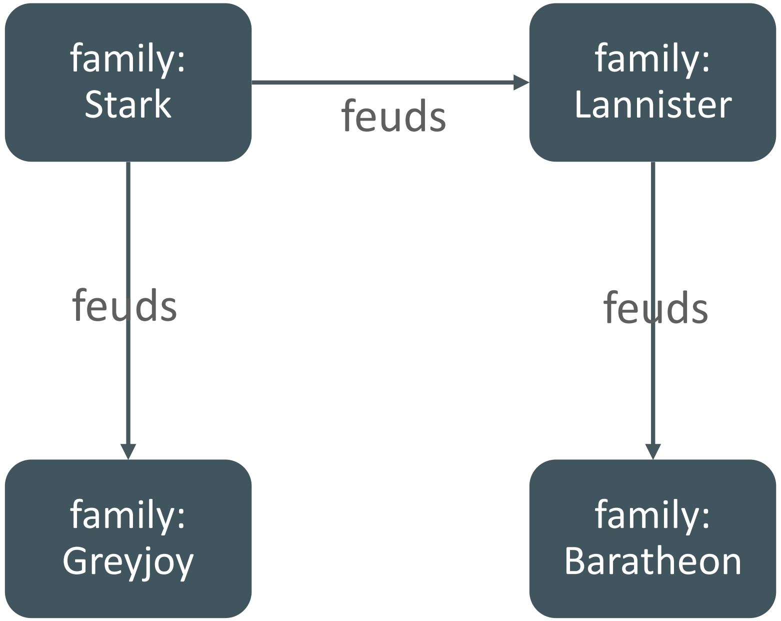
- Cypher query language
  - “ASCII art” for graph matching



Graph Databases © Neo Technology, 2015 - http://graphdatabases.com/

```
(emil)<-[:KNOWS]-(jim)-[:KNOWS]->(ian)-[:KNOWS]->(emil)
```

Cypher example



family: Stark

feuds

family: Lannister

feuds

feuds

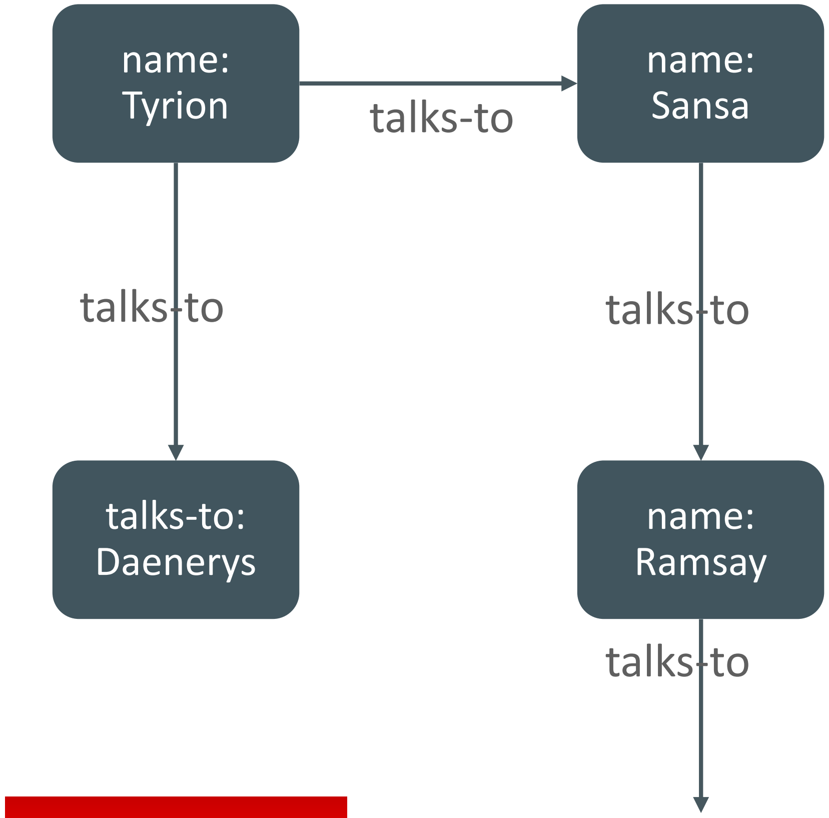
family: Greyjoy

family: Baratheon

Enemy of my enemy

```
MATCH (kin)-[:feuds]->(enemy)-[:feuds]->(friend)
WHERE kin.family = 'Stark'
RETURN kin, friend
```

Cypher example 2



Chinese Whispers (transitive closure)

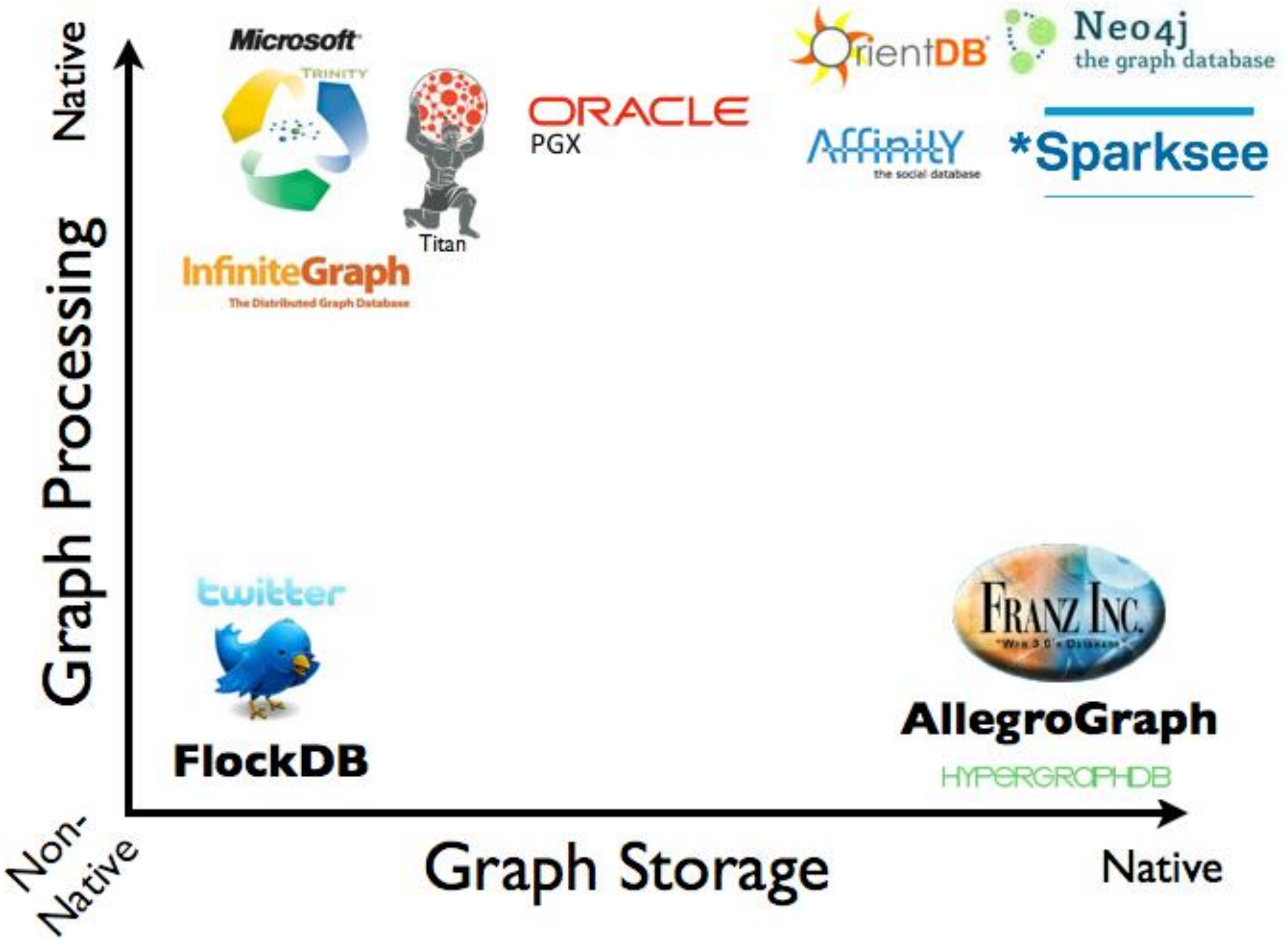
```
MATCH (tyrion: {name = 'Tyrion'})
       -[:talks-to*]->(people)

RETURN people
```

```
MATCH p=shortestPath(
  (tyrion: {name = 'Tyrion'})-[:talks-to*]->
  (ramsay: {name = 'Ramsay'}))

RETURN p
```

Graph Databases



Graph Databases © Neo Technology, 2015 - http://graphdatabases.com/

PGX

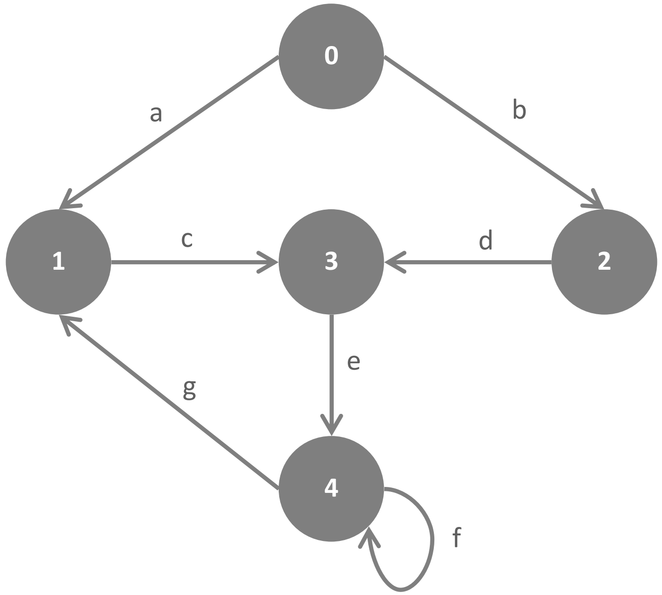
Oracle® Parallel Graph Analytics (PGX)

- Parallel, in-memory graph analytic framework
- Must fit in memory
- Immutable graph
  - Mutating creates a new graph
- No ACID, OLTP…

- Property Graph Model
- Execute graph algorithms
- Write custom graph algorithms in Green-Marl
- PGQL query language

PGX Implementation

Adjacency Matrix



	0	1	2	3	4
0		a	b		
1				c	
2				d	
3					e
4	g				f

- Edge iteration
  - loop through the array
- Large space requirements
  - Great for dense graphs, but most are very sparse

PGX Implementation

Compressed Sparse Row

	0	1	2	3	4
0		a	b		
1				c	
2				d	
3					e
4	g				f

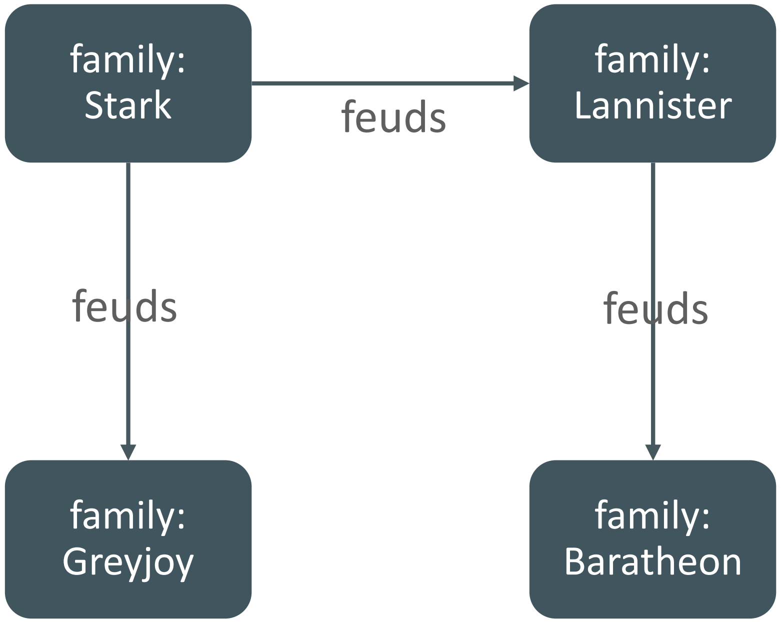
```
Array of nodes    [0, 1, 2, 3, 4]
Array of edges    [a, b, c, d, e, g, f]
Edge column       [1, 2, 3, 3, 4, 0, 4]
Row offsets       [0,    2, 3, 4, 5]
```

- Space dependent on the edge + node count

Accessing PGX Graph

- Java™ API
- Green-Marl algorithm
- PGQL query language

PGQL example



Enemy of my enemy

```
SELECT kin, friend
WHERE (kin WITH name = 'Stark')
 -[e1:feuds]->(enemy)
 -[e2:feuds]->(friend)
```

Comparing

RDBMS

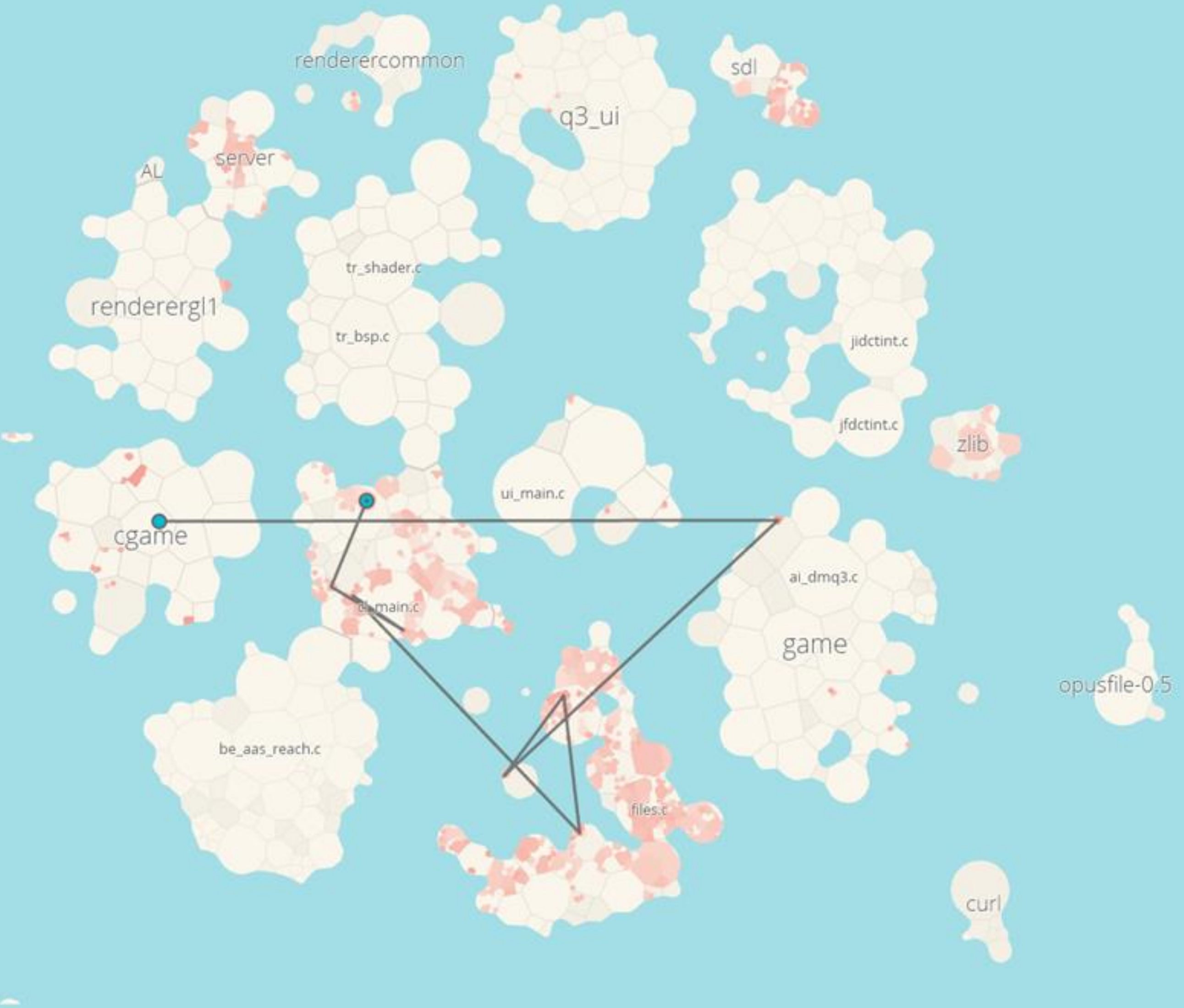
- Faster on large numbers of records
- Use less storage space
- Tortuous SQL for paths
- Very mature technology

Graph Database

- Faster on highly connected data
- Must store relationships
- Natural path queries
- Re-emerging technology

Frappé

Code Comprehension Tool



Current Tools for C/C++ Codebases

- IDEs
  - Issues with build integration
  - Use own indexer (not the compilers)
  - Instead…
- Fall back to text editors and text-search tools
  - Vim and emacs
  - Grep and cscope
- Fast and simple but imprecise

```
static VALUE mnew(…) {
    …
    data->id = rid;
    …
}
```

Find definition

```
  method.h:70
  node.h:244
  thread_pthread.c:594
  (+ 17 more)
```

Actual definition (14th)

```
  proc.c:21
```

Higher Level Task

- Tools provide defs/refs
  - Can go from a symbol to its definition or its references
- But usually have higher level task in mind
  - Where did this invalid value get introduced?
  - If I change this, what’s going to break?
  - How did the code reach this point?
- Can we answer these queries directly?

Frappé Graph Model

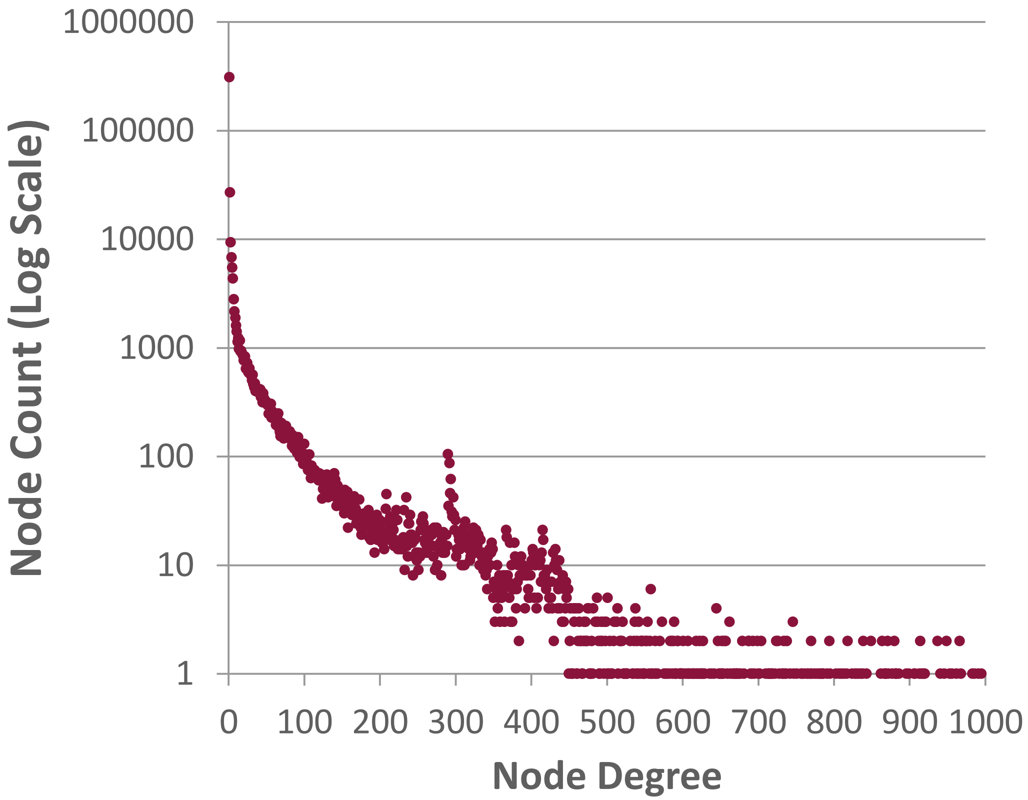
- Code entities are nodes
  - With properties: name, modifiers, etc.
- References are edges
  - With properties: source location, qualifiers, etc.
- Includes data from
  - File system: directory and file hierarchy
  - Preprocessor: includes, macros, their expansion and interrogations
  - AST: functions, locals, types, and relations between them
  - Build system: modules, libraries, executables, and linking information between them

Linux Kernel Graph

Oracle® Linux Kernel 2.6

- Approximately 11.4 million LOC
  - Around 1.6 million lines compiled

Node count	508 032
Edge count	3 991 063
DB Size	787 MB



Experiences and Challenges

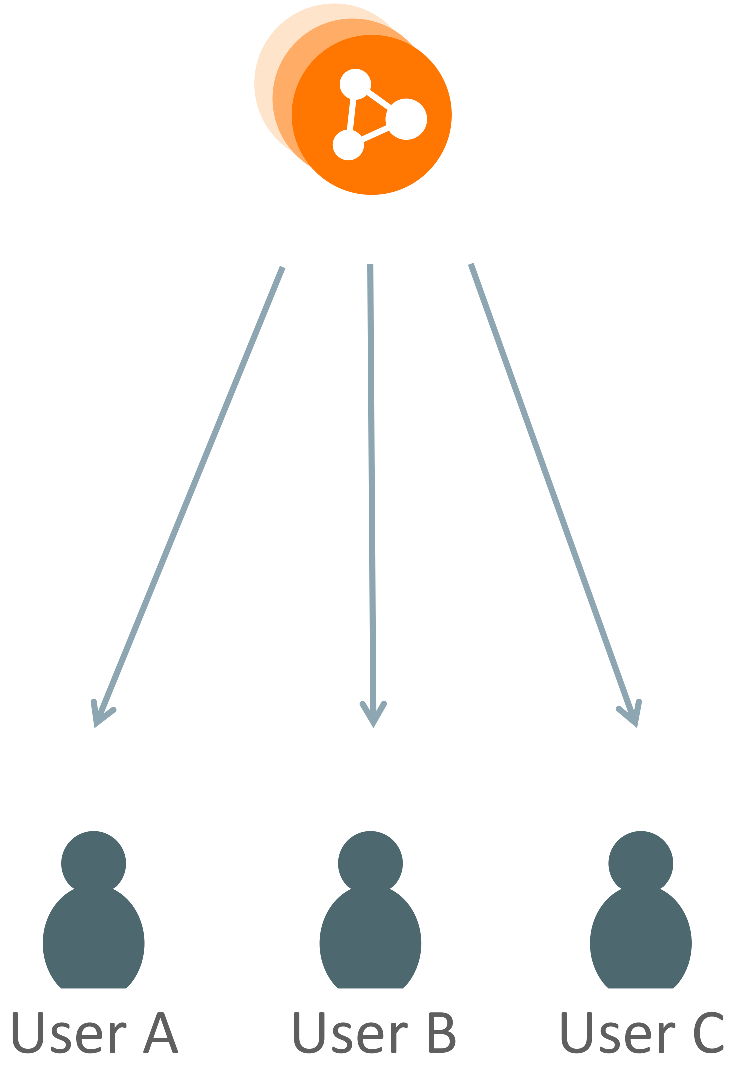
Temporal

- Developers work on multiple versions of the code
  - Hours to weeks ago if working on a bug fix
  - Weeks to months ago if working on a new feature
  - Months to years ago if backporting a bug fix
- Need to be able to query any of these versions

Experiences and Challenges

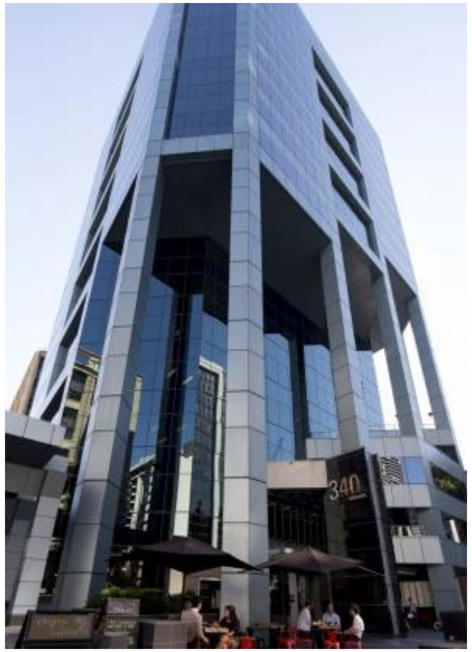
Temporal

- Could include temporal aspect in model
  - Complicates model and makes queries verbose
- Ideally the database layer would handle it
  - Efficient storage
  - Succinct cross-version queries
- Difficult, but has applications in many domains



Database Futures

- 1960s – Hierarchical
- 1970s, 1980s, 1990s – Relational
- 1990s –Object, Object Relational
- 2000s – NoSQL
  - Key-Value, Document, Distributed, **Graph**, …



Thank You!

David Meibusch
Oracle Labs, Australia
340 Adelaide St, Brisbane

ORACLE®