# Getting Started with Graph Databases

rik@neotechnology.com

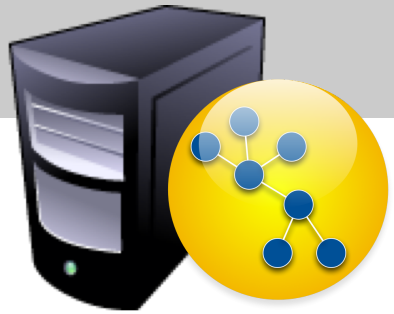

# Agenda

- Introduction
  - NO-SQL context
  - What is Neo4j?
  - When/why should I use it?
- Graph Queries
  - Cypher query language
  - Create and query data
- Graph Visualisations
- Technical Overview
  - Deployment modes
  - Java APIs
  - Other libraries
- Case Studies
- Q&A

neo technology
graphs are everywhere

goto; conference

# Introduction



neo technology
graphs are everywhere

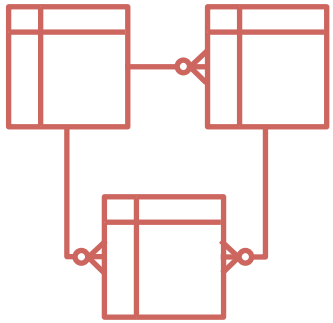
goto; conference

NOSQL is simply…

# Not Only SQL

# What's so bad about

## Relational

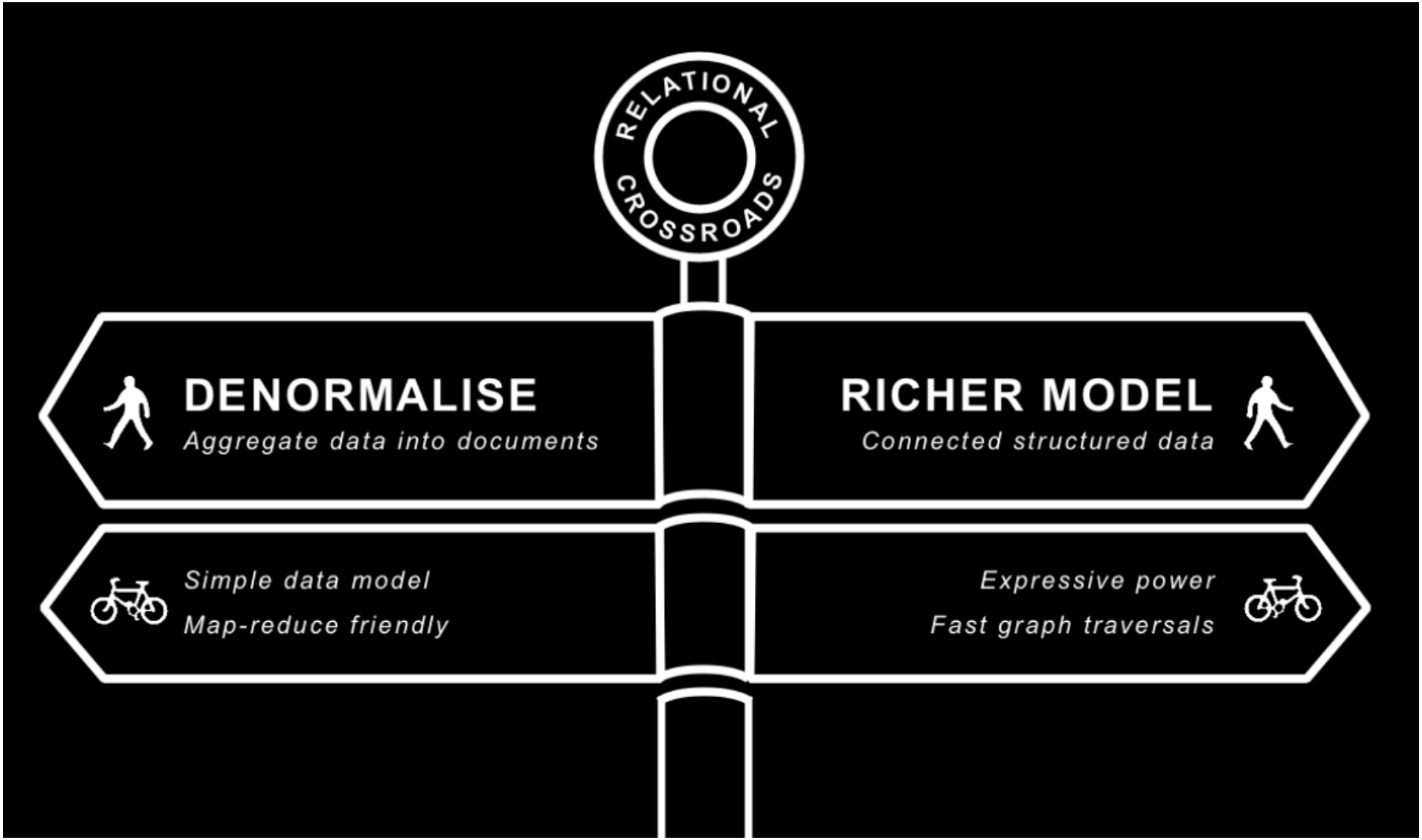


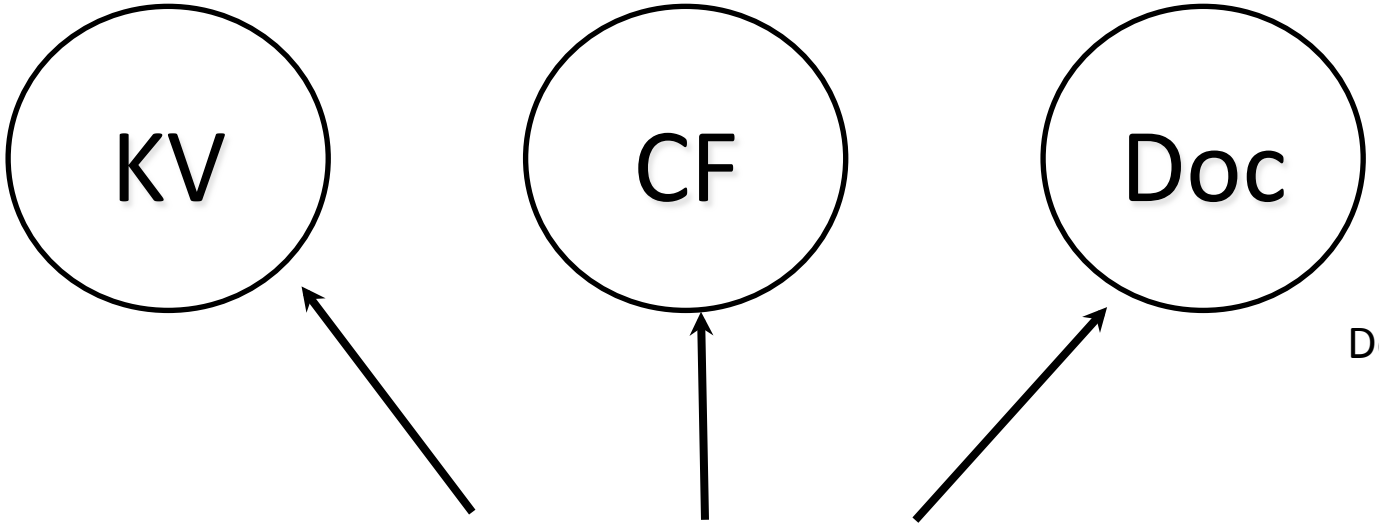
**VOLUME**

**Complexity**

neotechnology
graphs are everywhere

# The Relational Crossroads



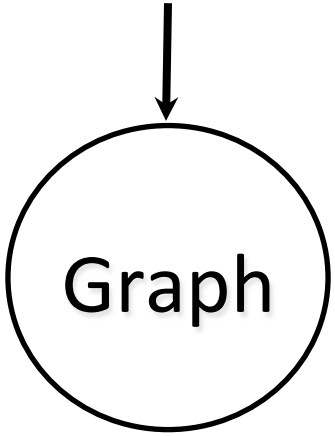



Denormalise

# Four NOSQL Categories

arising from the "relational crossroads"

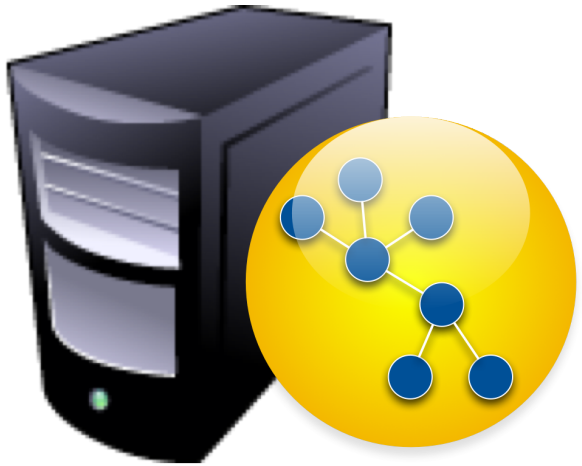
Normalise




neotechnology
graphs are everywhere

# So what is a graph database?

- OLTP database
  - "end-user" transactions
- Model, store, manage data as a graph

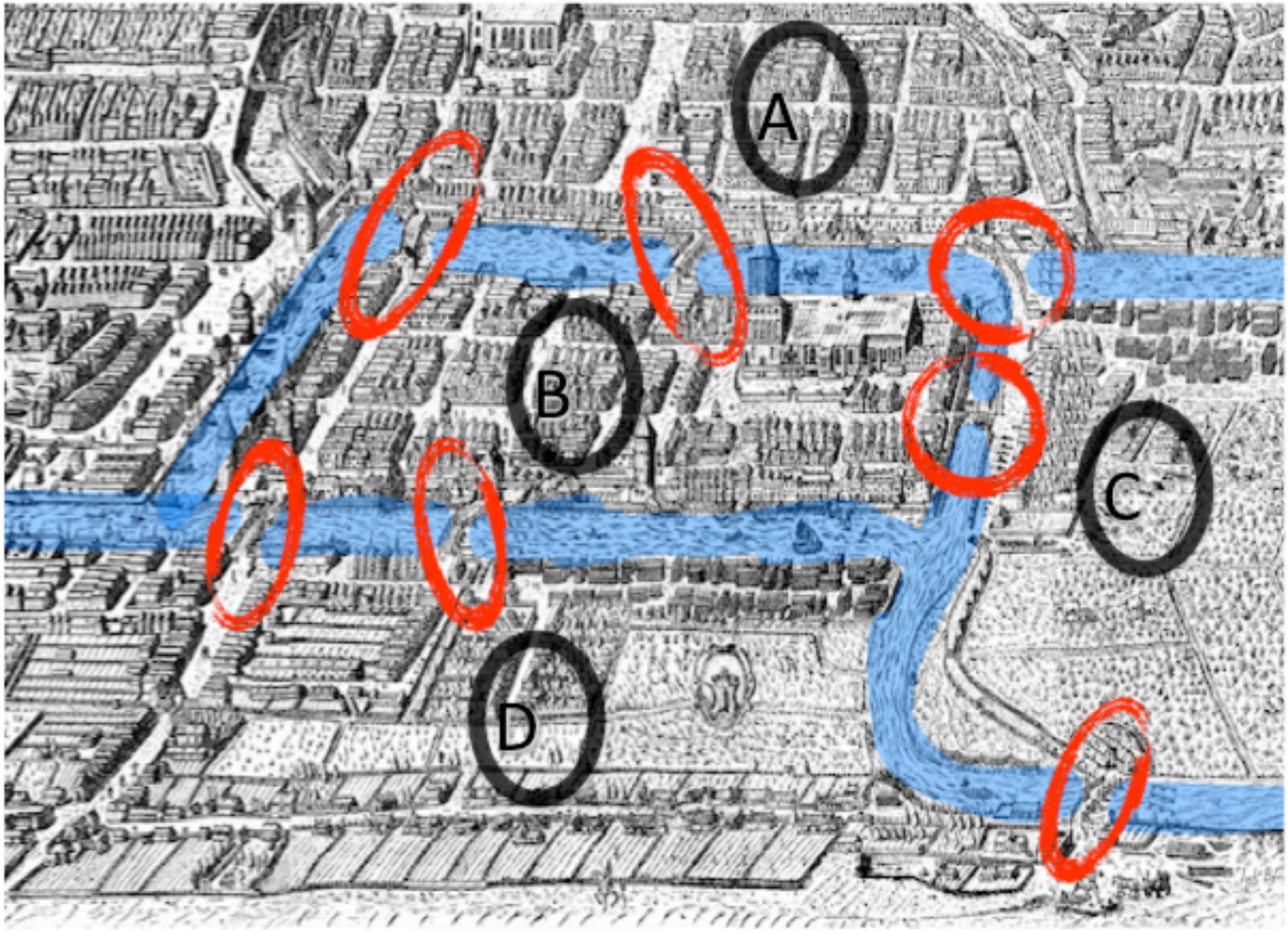


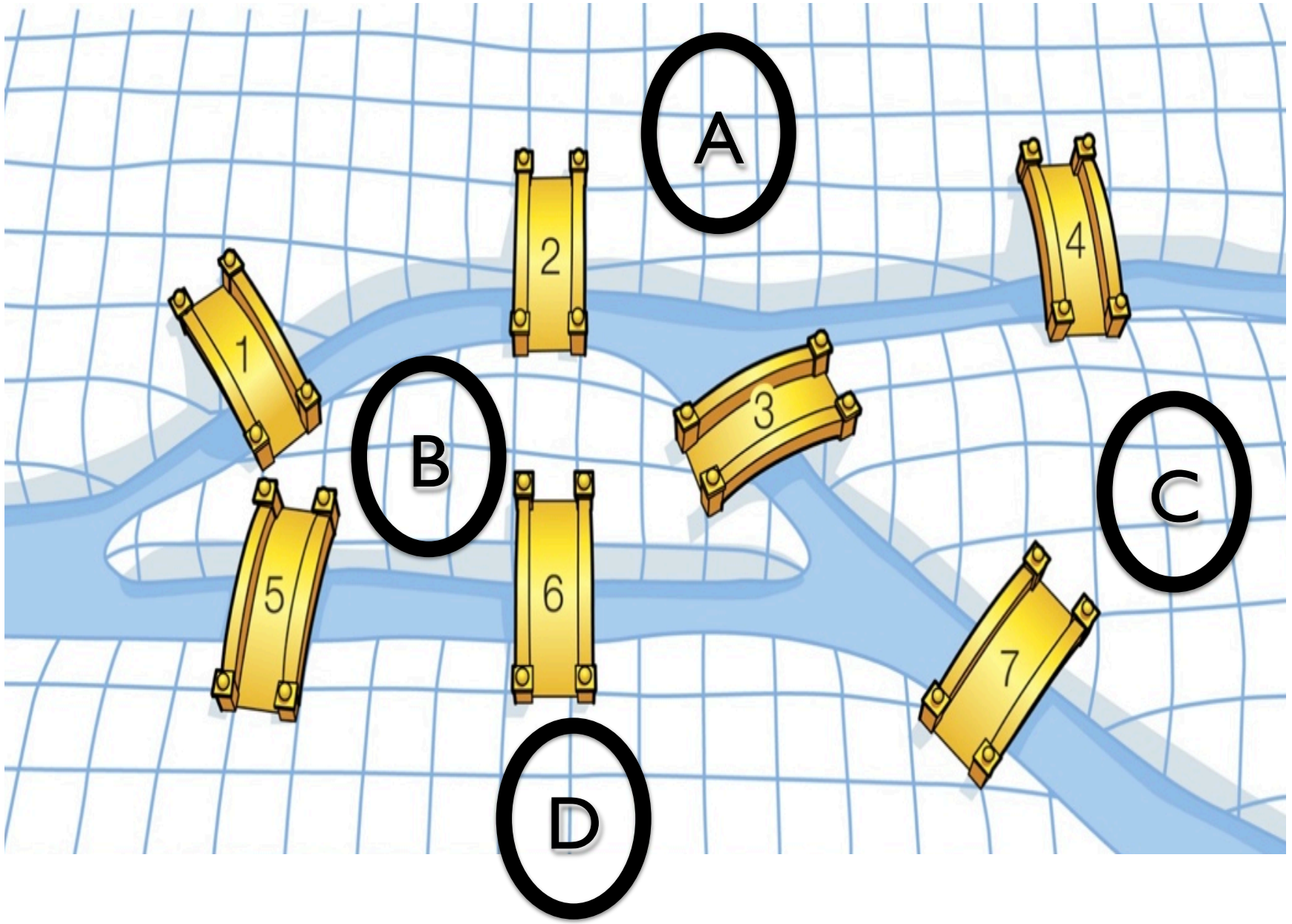
neotechnology
graphs are everywhere

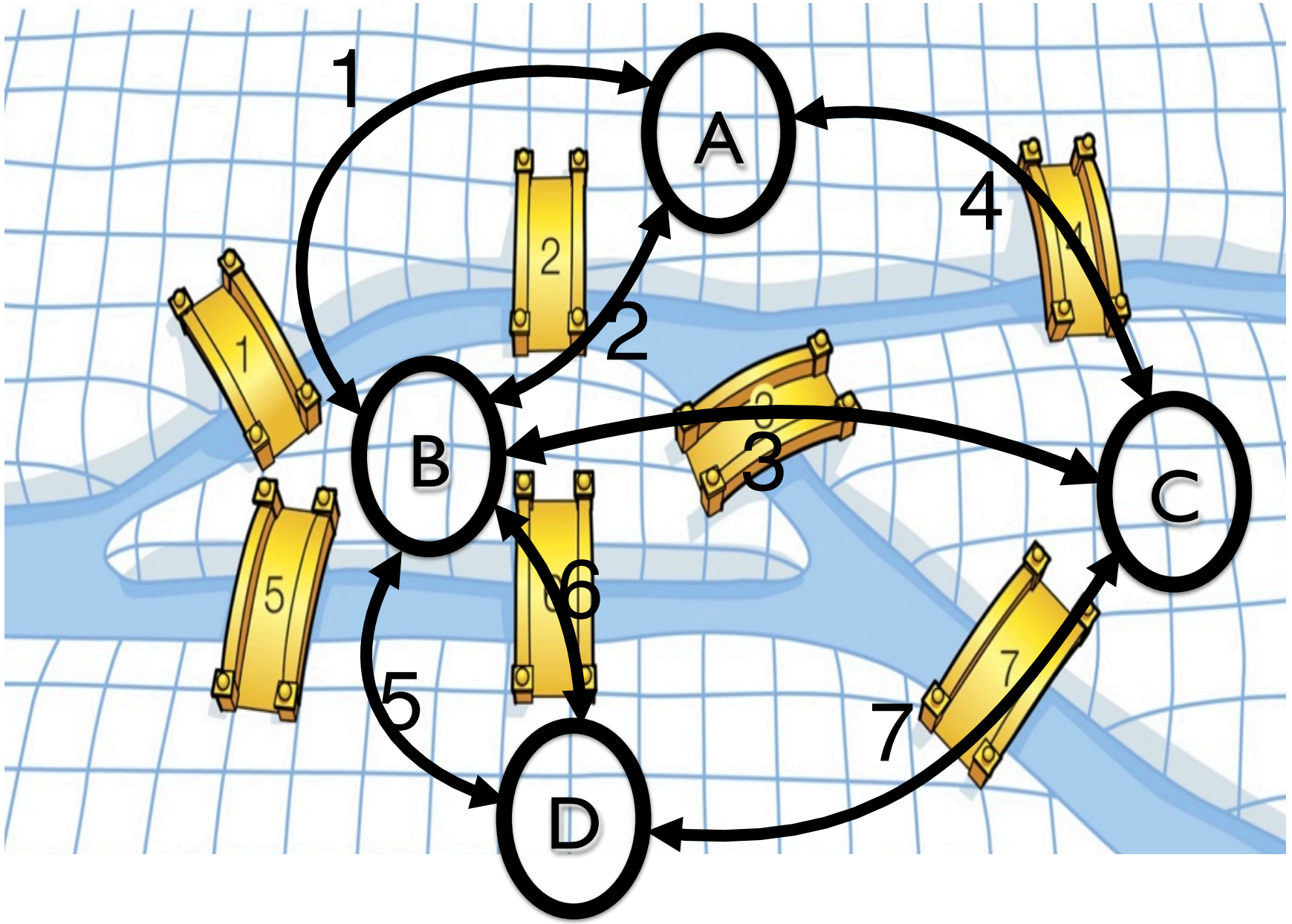
# Meet Leonhard Euler

- Swiss mathematician
- Inventor of Graph Theory (1736)

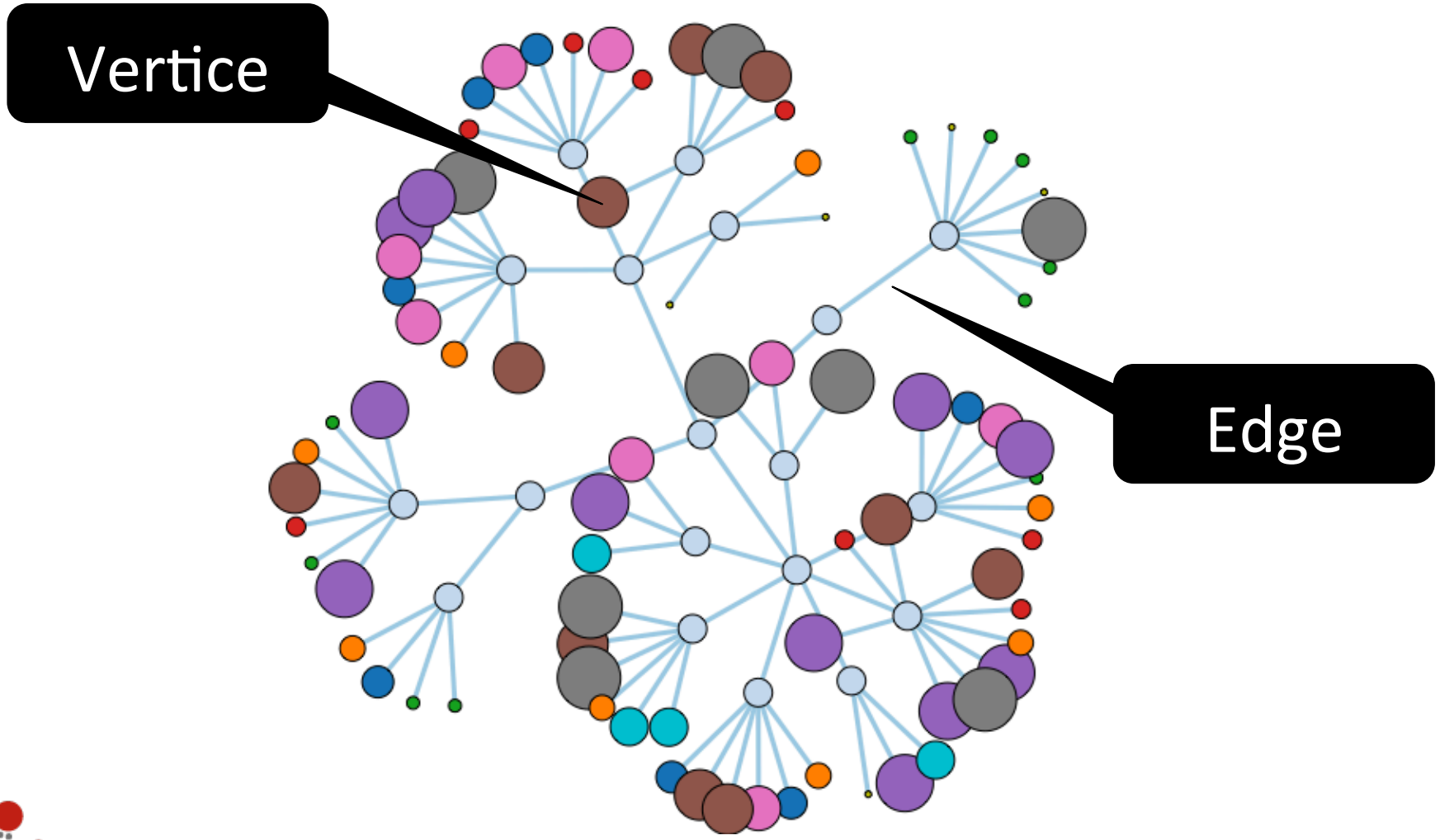
# Königsberg (Prussia) - 1736




A
B
C
D
1
2
3
4
5
6
7

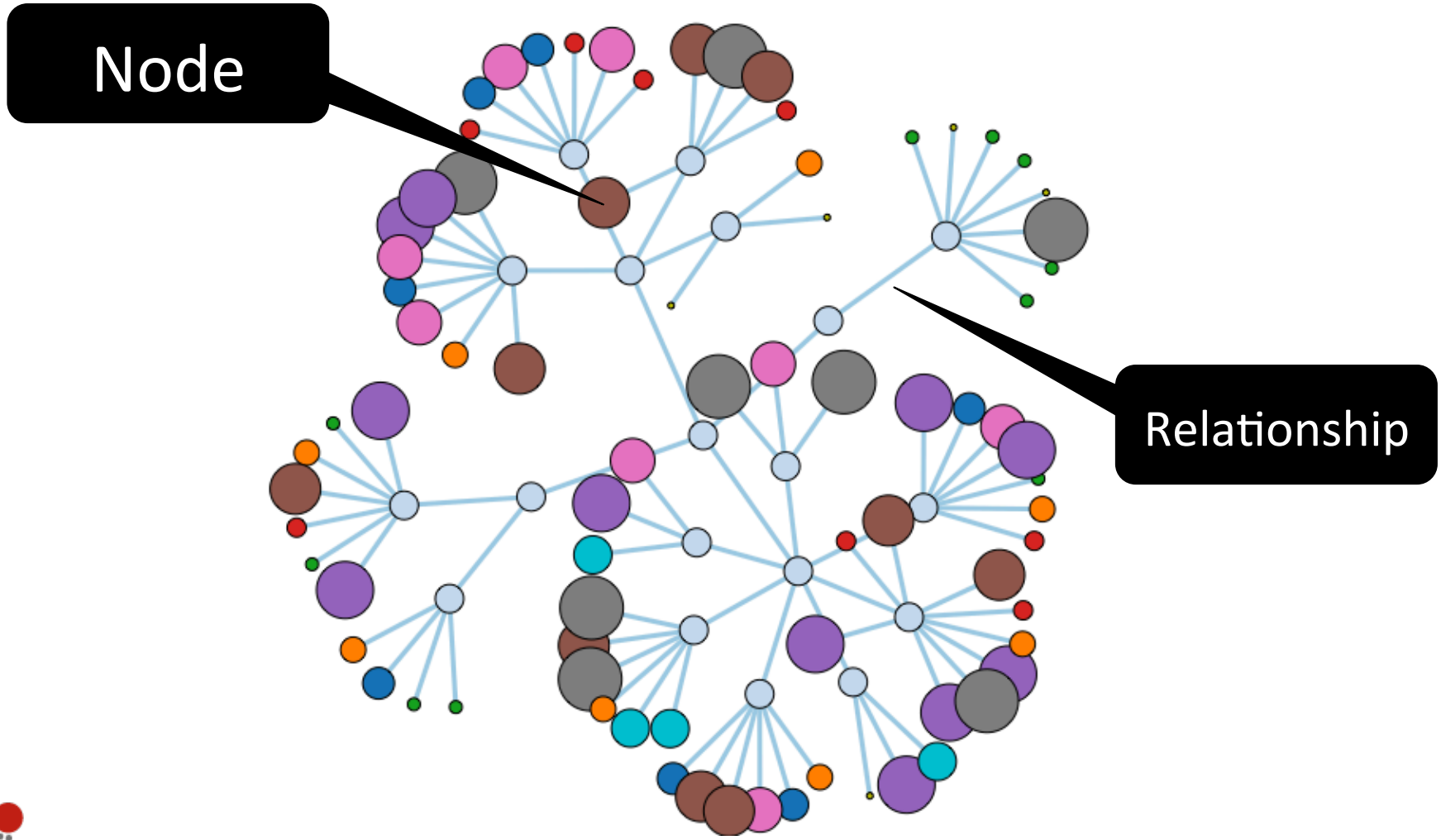

A
B
C
D
1
2
3
4
5
6
7
1
2
3
4
5
6
7

# What is a graph?




neotechnology
graphs are everywhere

# What is a graph?




neotechnology
graphs are everywhere

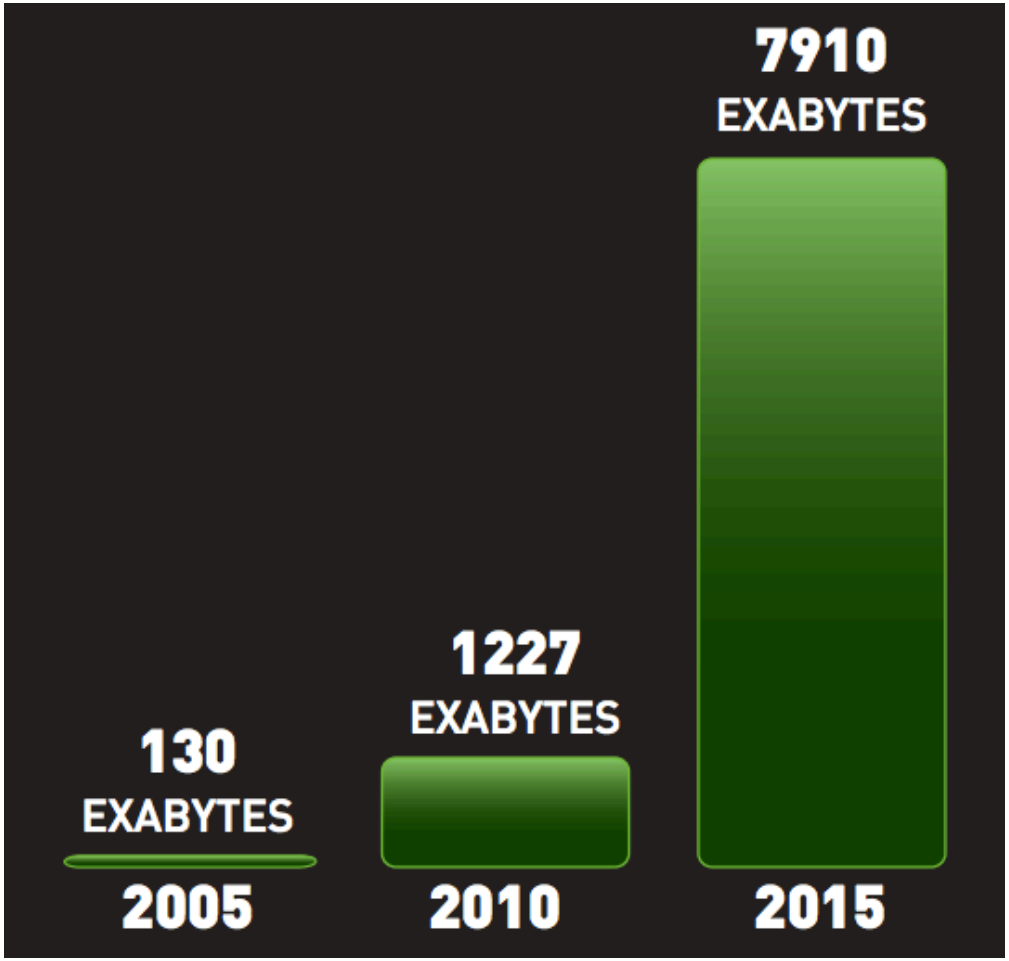
# What are graphs good for?

# Complexity

# Data Complexity

*complexity = f(**size**, semi-structure, connectedness)*

# Size

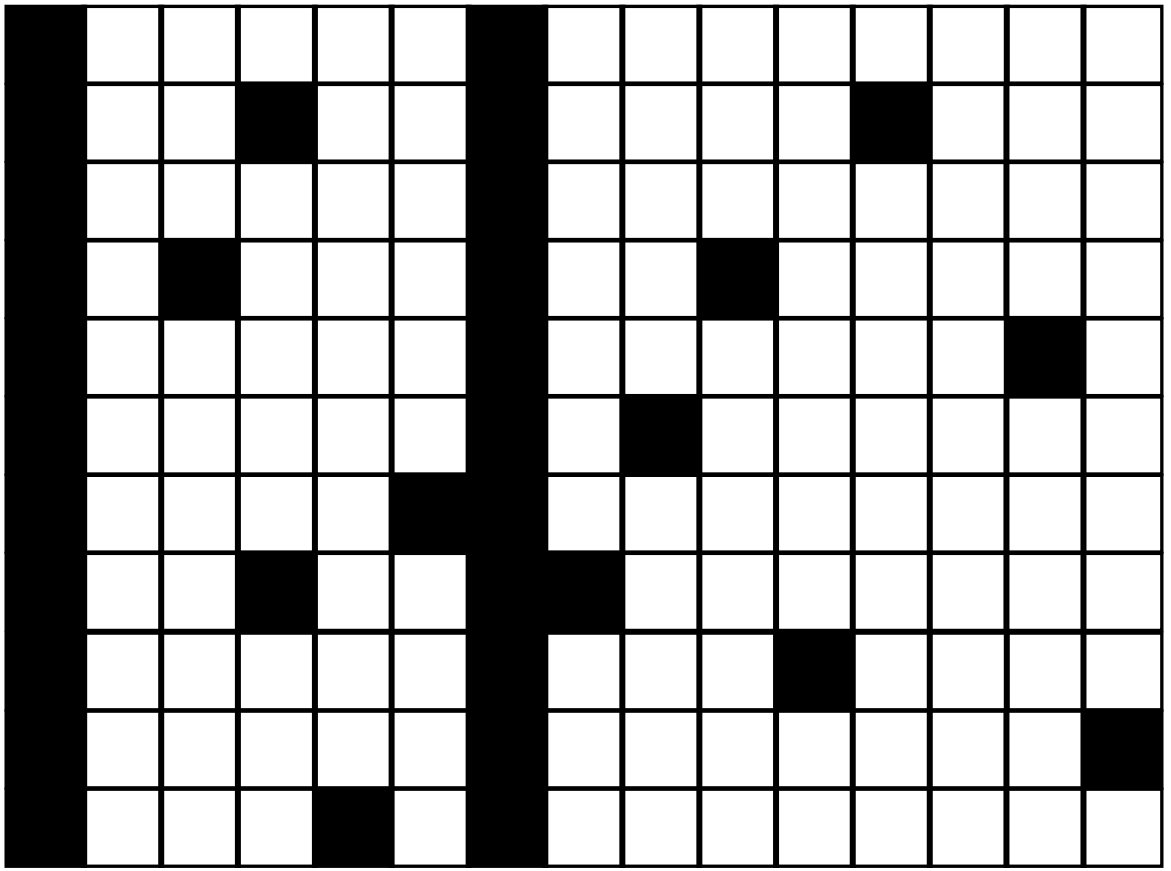



| Name (Symbol) | Value |
|---|---|
| kilobyte (kB) | $10^3$ |
| megabyte (MB) | $10^6$ |
| gigabyte (GB) | $10^9$ |
| terabyte (TB) | $10^{12}$ |
| petabyte (PB) | $10^{15}$ |
| **exabyte** (EB) | $10^{18}$ |
| zettabyte (ZB) | $10^{21}$ |
| yottabyte (YB) | $10^{24}$ |

# The Real Complexity

*complexity = f(size, **semi-structure**, **connectedness**)*

# Semi-Structure

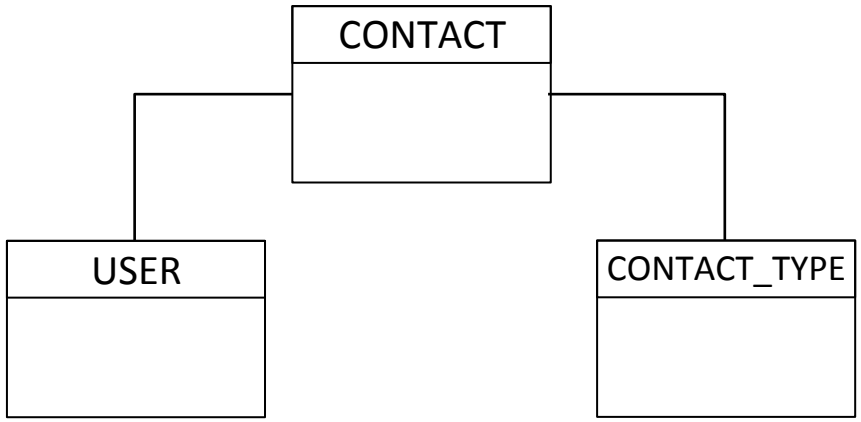


neotechnology
graphs are everywhere

# Semi-Structure

| USER_ID | FIRST_NAME | LAST_NAME | EMAIL_1 | EMAIL_2 | FACEBOOK | TWITTER | SKYPE |
|---|---|---|---|---|---|---|---|
| 315 | Rik | Van Bruggen | rik@neotechnology.com | rik@vanbruggen.be | NULL | @rvanbruggen | rvanbruggen |

Email: rik@neotechnology.com
Email: rik@vanbruggen.be
Twitter: @rvanbruggen
Skype: rvanbruggen

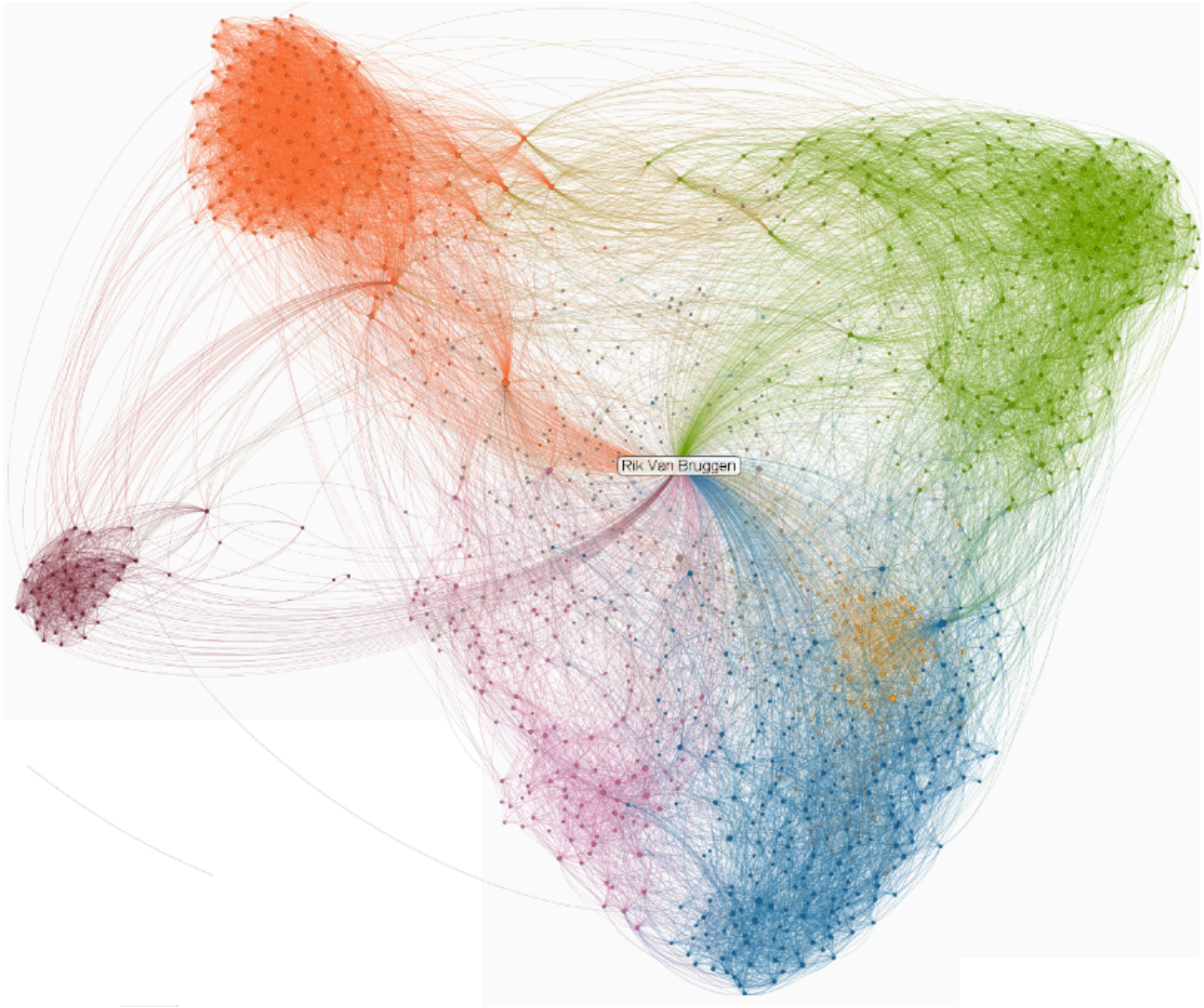



neotechnology
graphs are everywhere

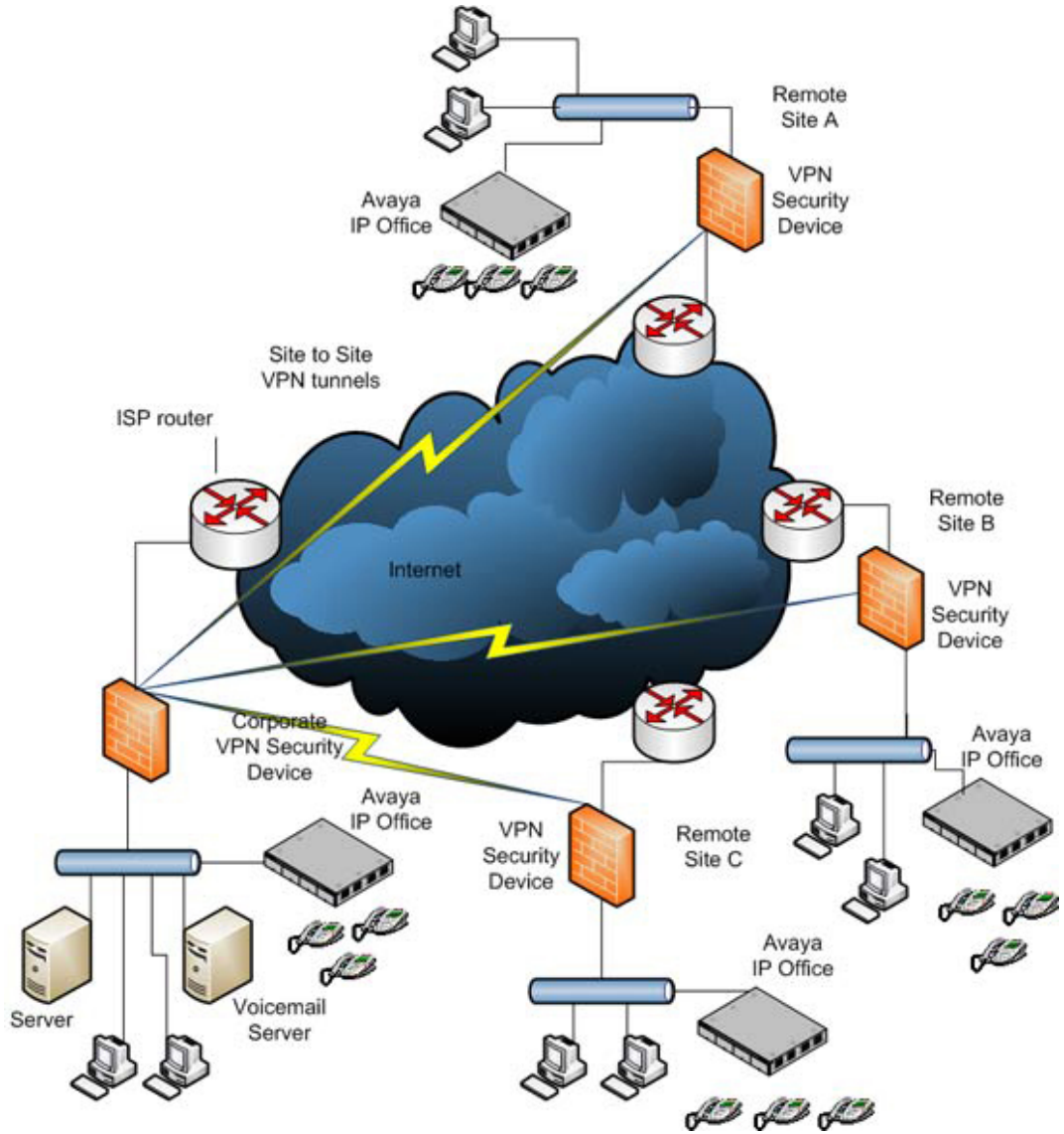
# The Real Complexity

*complexity = f(size,* ***semi-structure,*** ***connectedness****)*

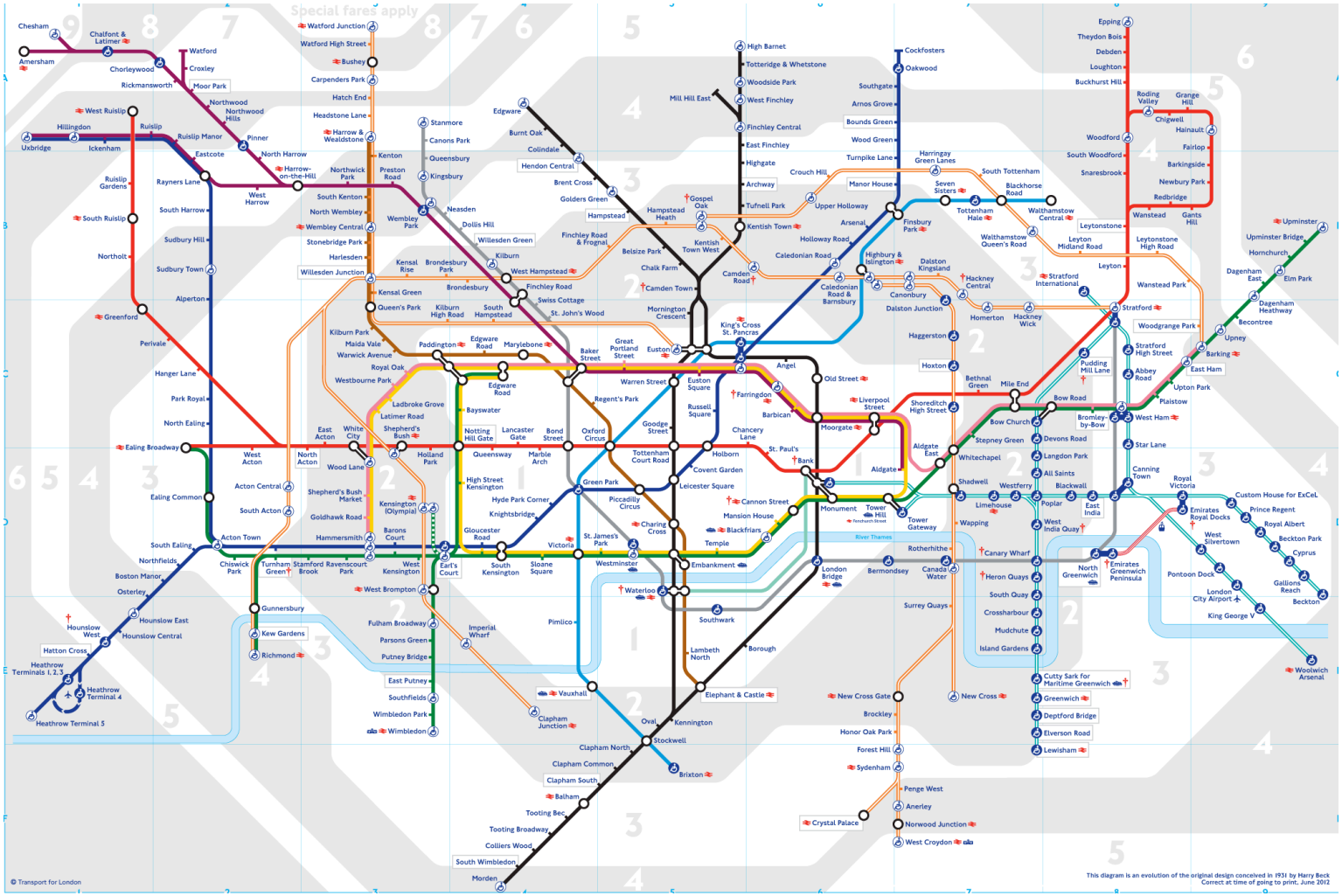
# Examples of Connectedness




neotec
graphs are ev




neotechnology
graphs are everywhere



## Frequently Bought Together



**Price For All Three: £69.50**

Add all three to Basket

Show availability and delivery details

- **This item:** Patterns of Enterprise Application Architecture (The Addison-Wesley Signature Series) by Martin Fowler Hardcover £24.00
- Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions (Addison-Wesley Signature Series (Fowler) Addison-Wesley Sign) by Gregor Hohpe Hardcover £22.00
- Domain-driven Design: Tackling Complexity in the Heart of Software by Eric Evans Hardcover £23.50

## Customers Who Bought This Item Also Bought

Enterprise Integration Patterns: Designing, …

Gregor Hohpe

(16)

Hardcover

£22.00

Service Design Patterns: Fundamental Design …

Robert Daigneau

(2)

Hardcover

£19.97

Design patterns : elements of reusable …

Erich Gamma

(51)

Hardcover

£21.00

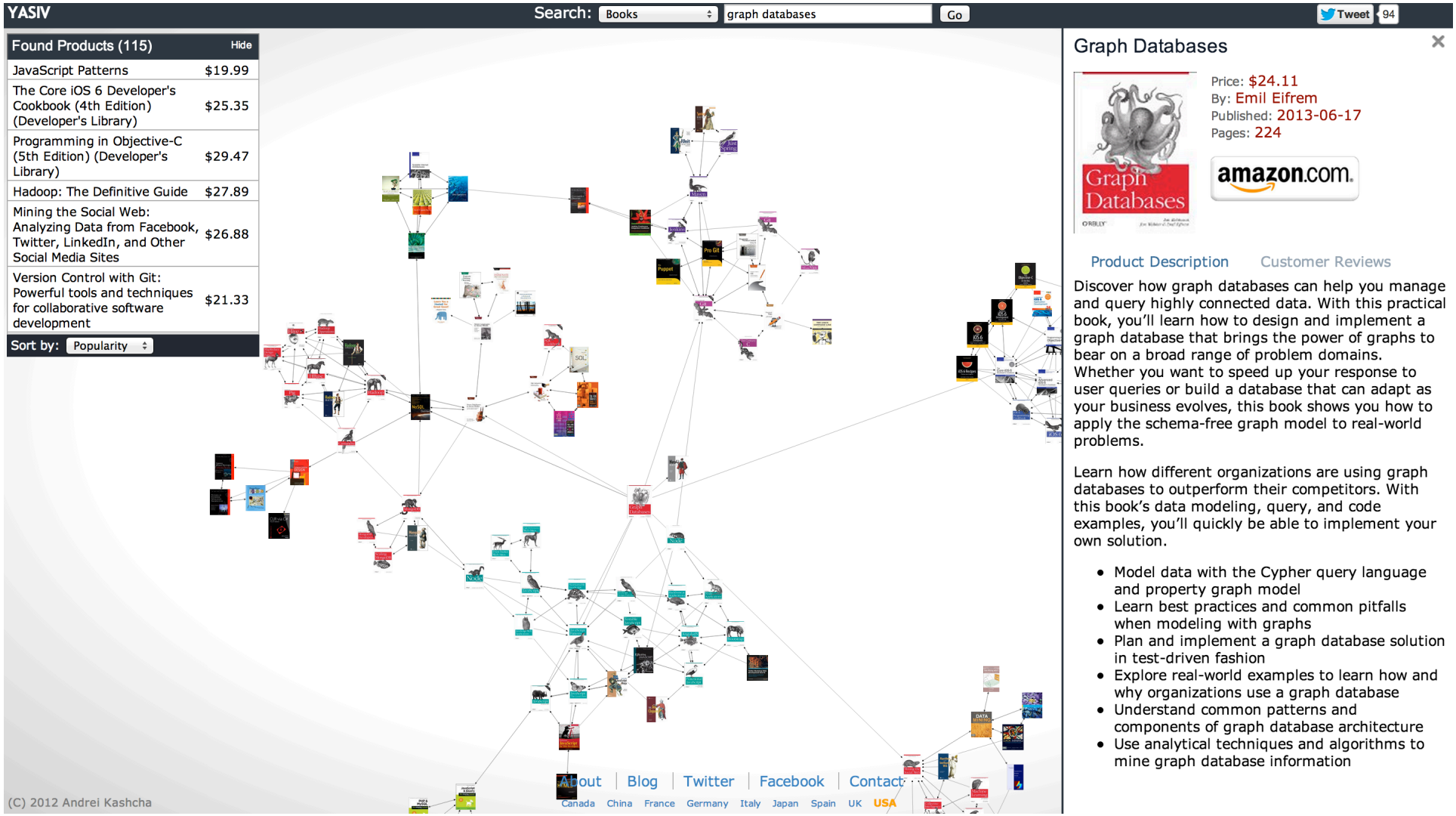
Domain-driven Design: Tackling Complexity in …

Eric Evans

(13)

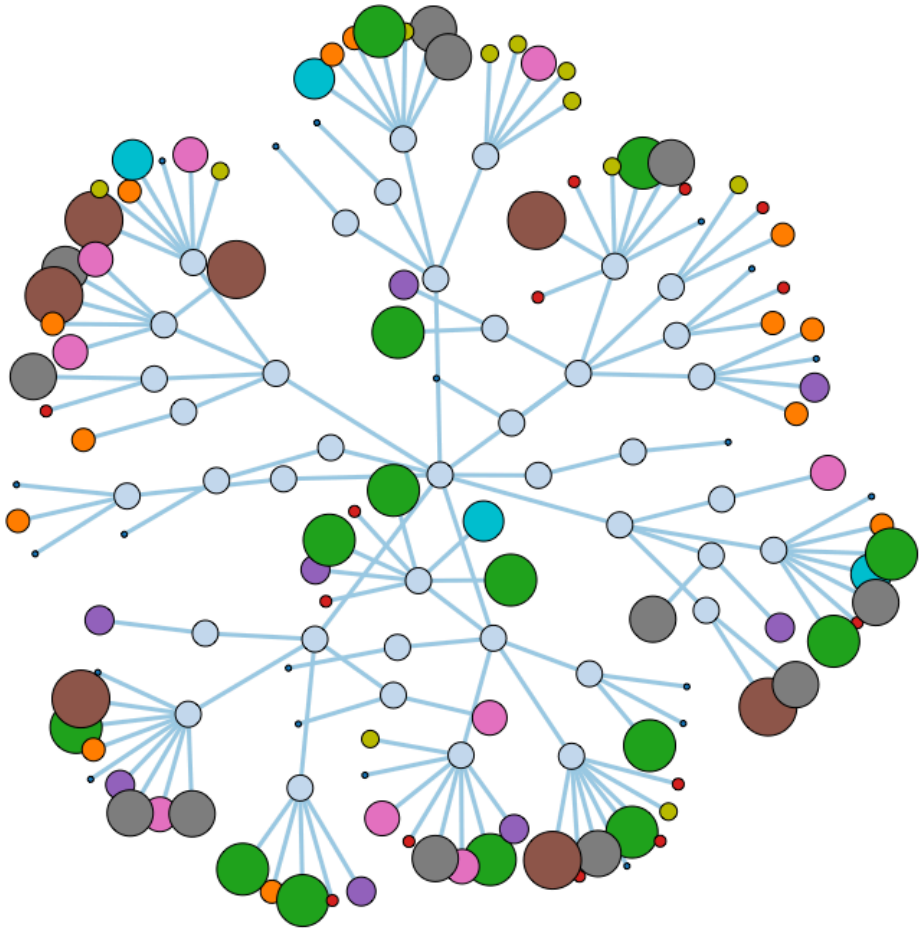
Hardcover

£23.50




neotechnology
graphs are everywhere

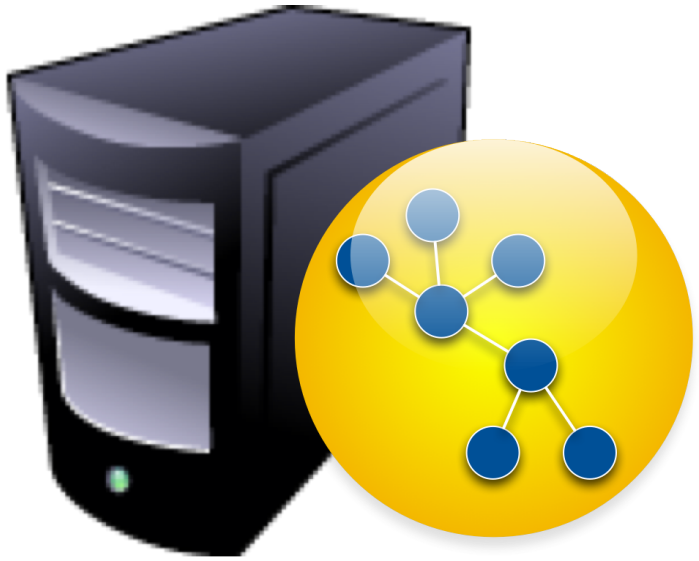
# Graphs Are Everywhere



neotechnology
graphs are everywhere

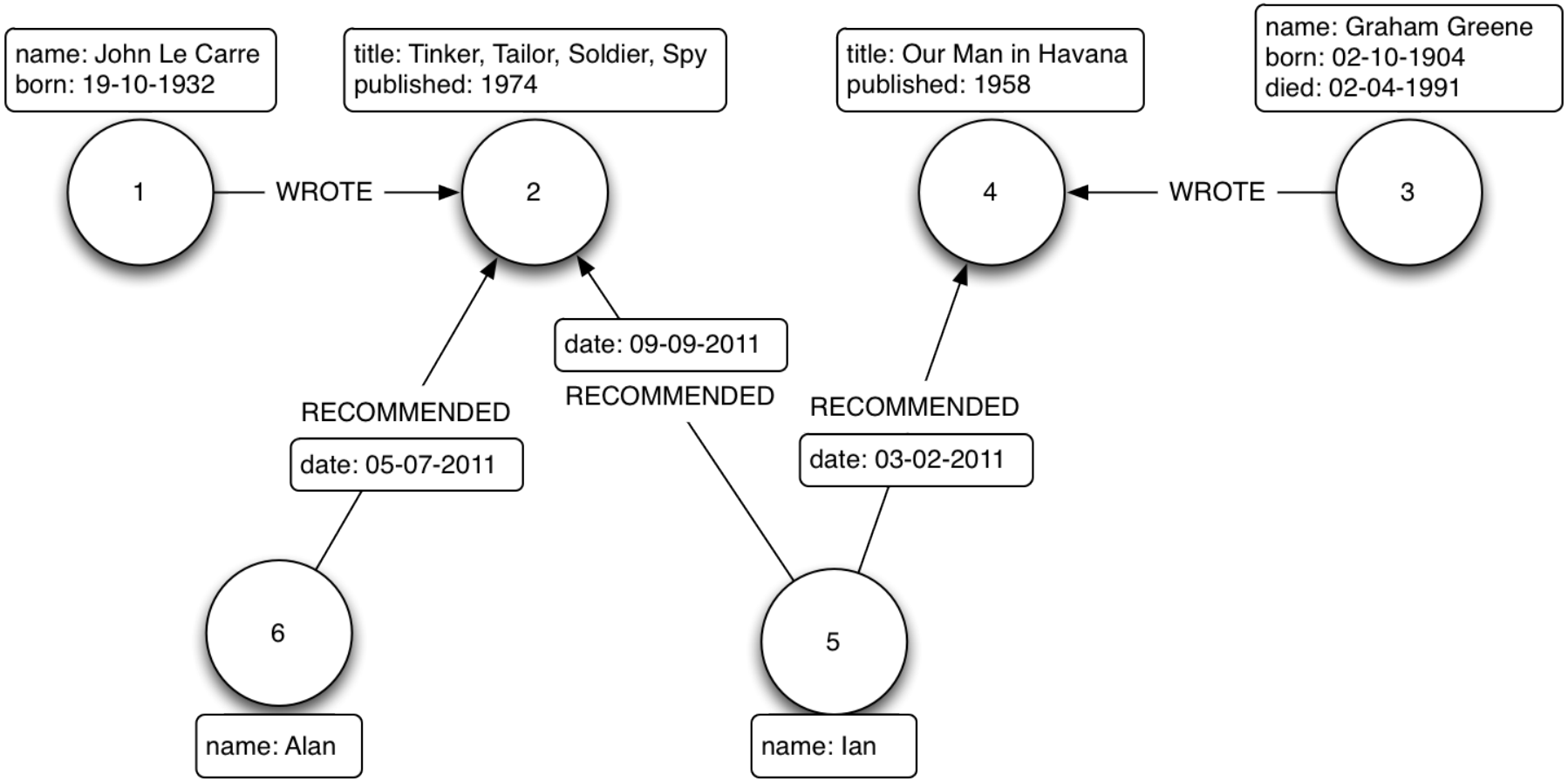
# Neo4j is a Graph Database

- JVM based
- ACID transactions
- Query language
- Rich Java APIs
- Using the Property Graph model



neotechnology
graphs are everywhere

# Property Graph Model




neotechnology
graphs are everywhere

# Property Graph Summary

- Nodes
  - Containers for properties
- Properties
  - Key-value pairs
  - Primitive and array values
- Relationships
  - Name
  - Direction
  - May also contain properties

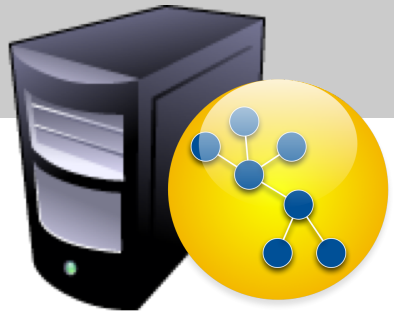
# More About Relationships

- Must have a start node and an end node (no dangling relationships)
- Start node and end node can be the same (e.g. 'self' relationships)
- Nodes can be connected by more than one relationship

# When Should I Use Graph Databases??

- Densely-connected, semi-structured domains
  - Lots of join tables? Connectedness
  - Lots of sparse tables? Semi-structure
- Data Model Volatility
- Easy to evolve
- Join Complexity and Performance
- Millions of 'joins' per second
- Consistent query times as dataset grows

# Let me SHOW you?

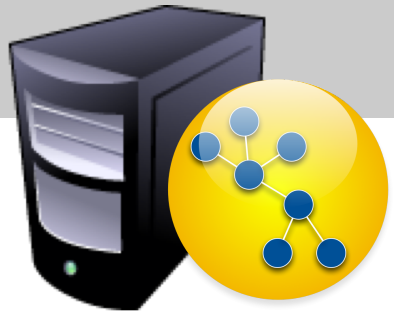


neo technology
graphs are everywhere

goto; conference


Neo
Beer
-[:love
Neo
Beer
-[:love]->gr
Neo
Beer
-[:love]->
er
]->graphs
eer
:love]->
raphs

# Graph Queries

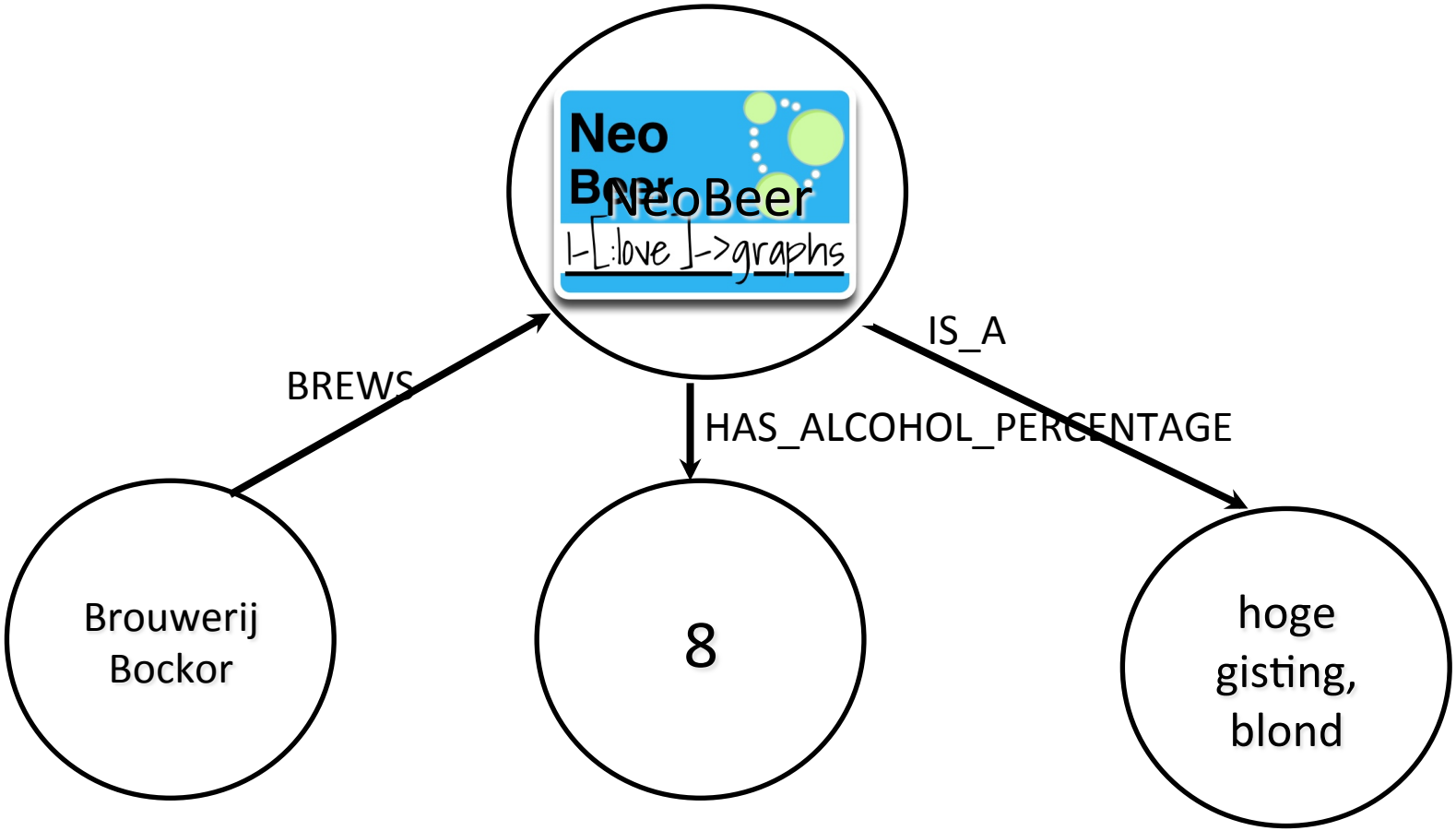


- A language for describing graphs
- Creating nodes, relationships and properties
- Querying data

neo technology
graphs are everywhere

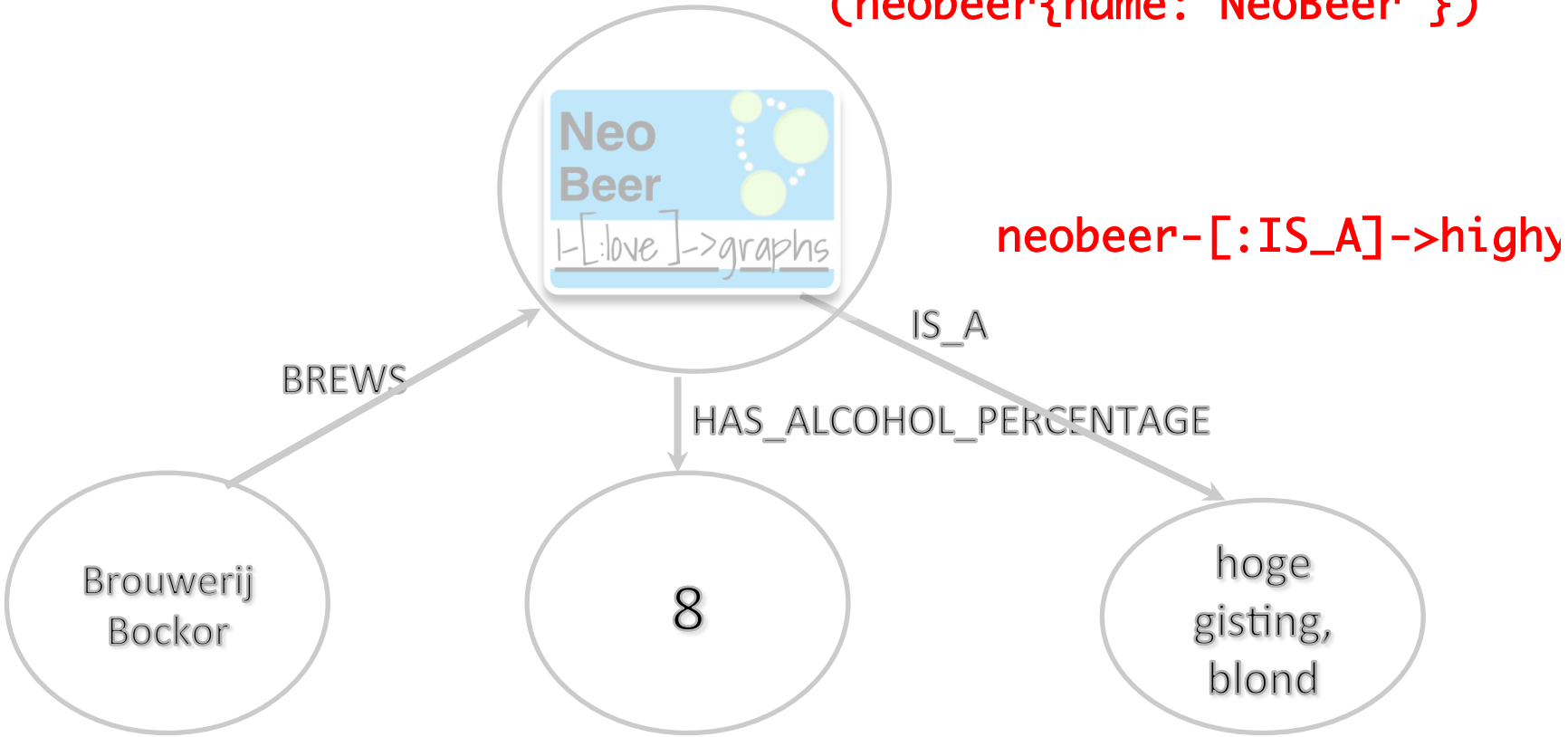
goto; conference

# Describing Graphs



# Cypher

(neobeer{name:"NeoBeer"})




neobeer-[:IS_A]->highy

neobeer<-[:BREWS]-bockor

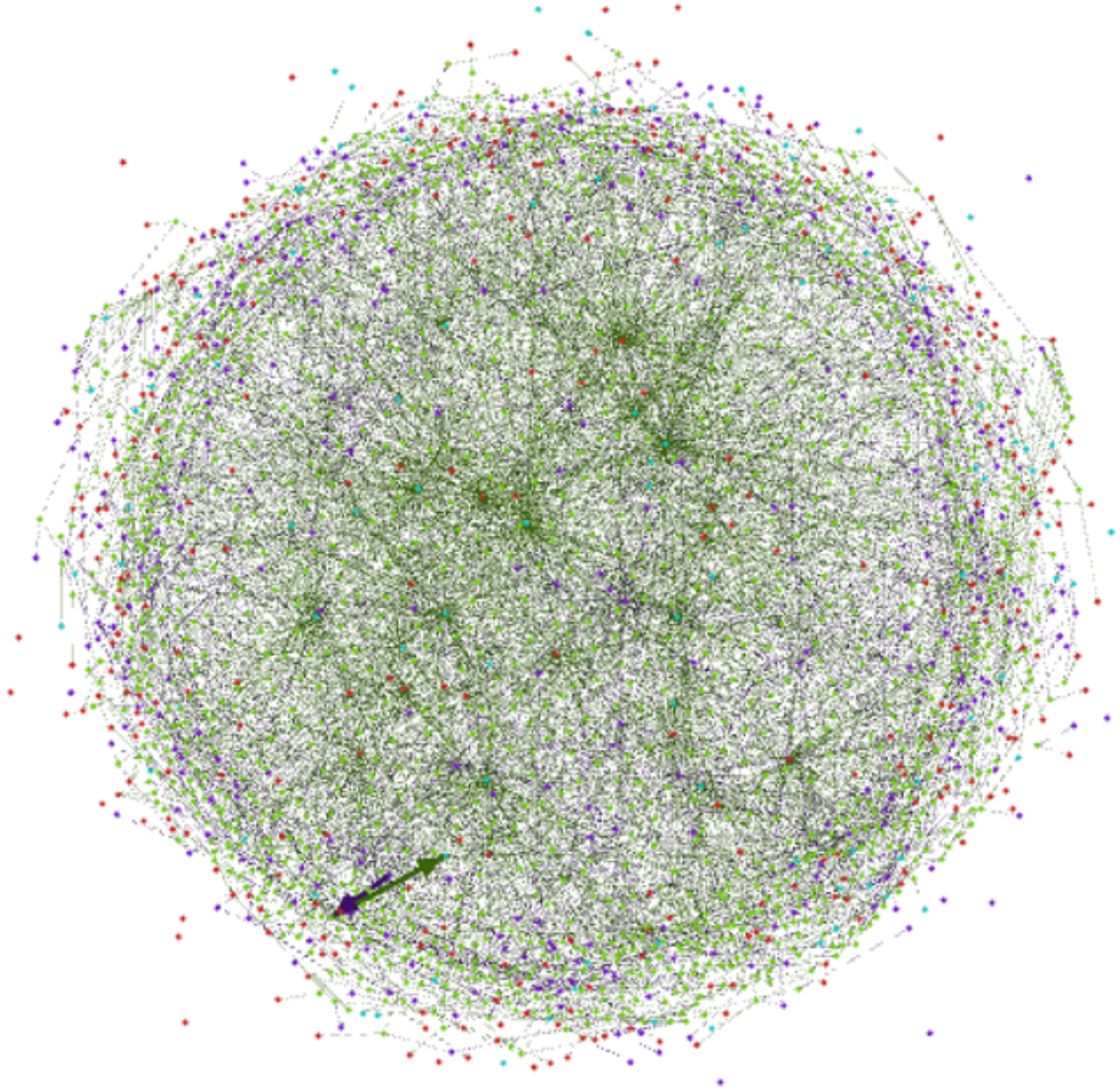
neobeer-[:HAS_ALCOHOL_PERCENTAGE]-8

neotechnology
graphs are everywhere

# Starting with “Omer”

```
START
        omer=node:node_auto_index(name="Omer")
MATCH
        omer-[:IS_A]->beertype,
        omer-[:HAS_ALCOHOL_PERCENTAGE]->alcperc,
        brewery-[:BREWS]->omer
CREATE
        (neobeer{name:"NeoBeer"})-[:IS_A]-beertype,
        neobeer-[:HAS_ALCOHOL_PERCENTAGE]-alcperc,
        brewery-[:BREWS]-neobeer
RETURN neobeer;
```

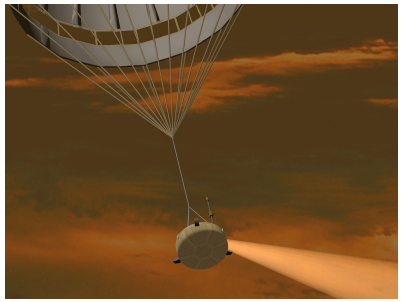
# A Hairball!

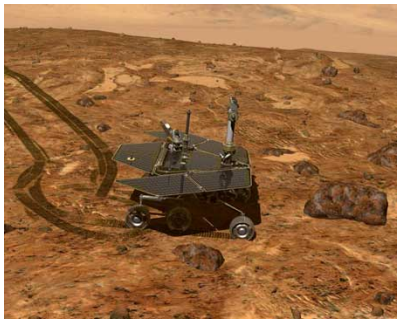


neotechnolo
graphs are everywhere

# Querying a Graph

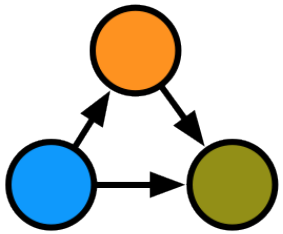
- Graph local
  - Contextualized "ego-centric" queries
- "Parachute" into graph
  - Start node(s)
    - Found through Index lookups
- Crawl the surrounding graph
  - 2 million+ joins per second
    - No more Index lookups: Index-free adjacency



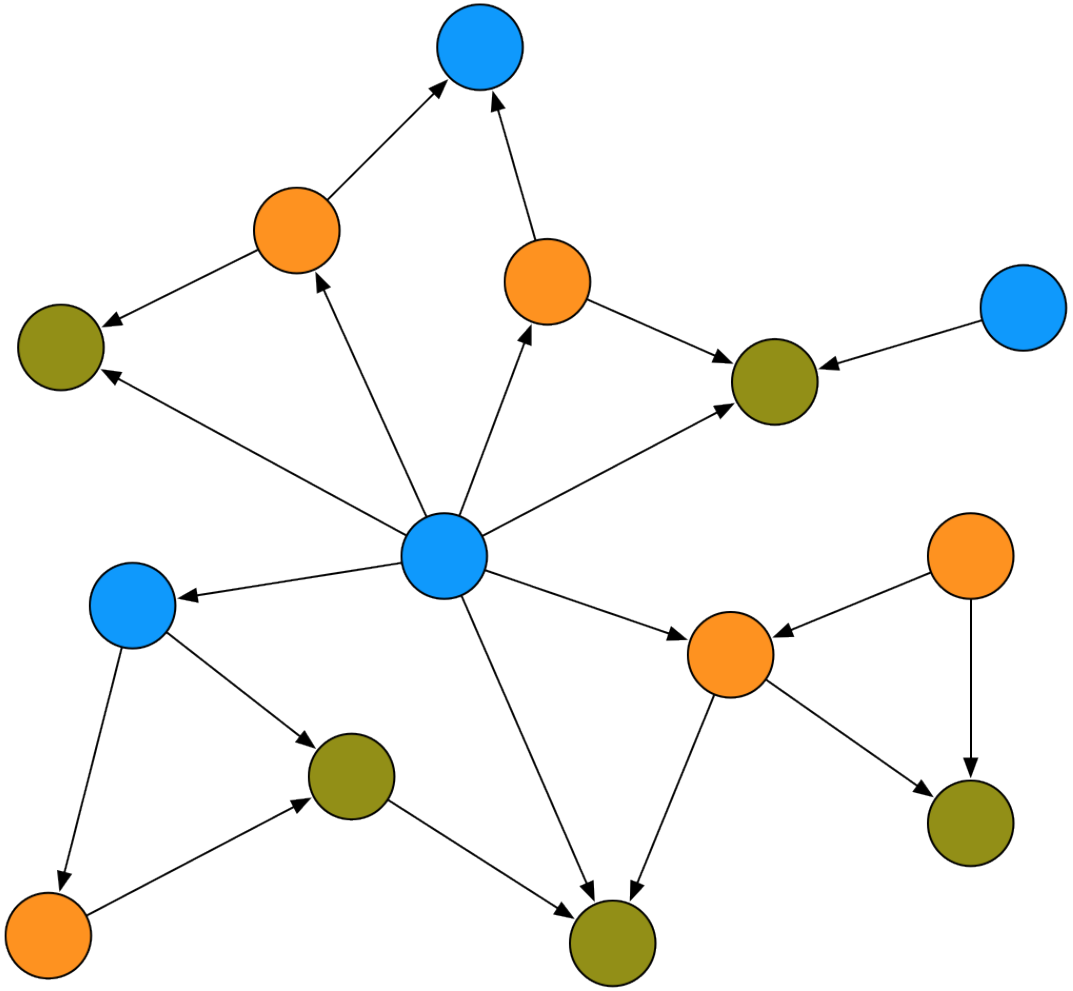


neotechnology
graphs are everywhere

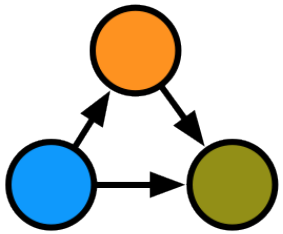
# Queries: Pattern Matching



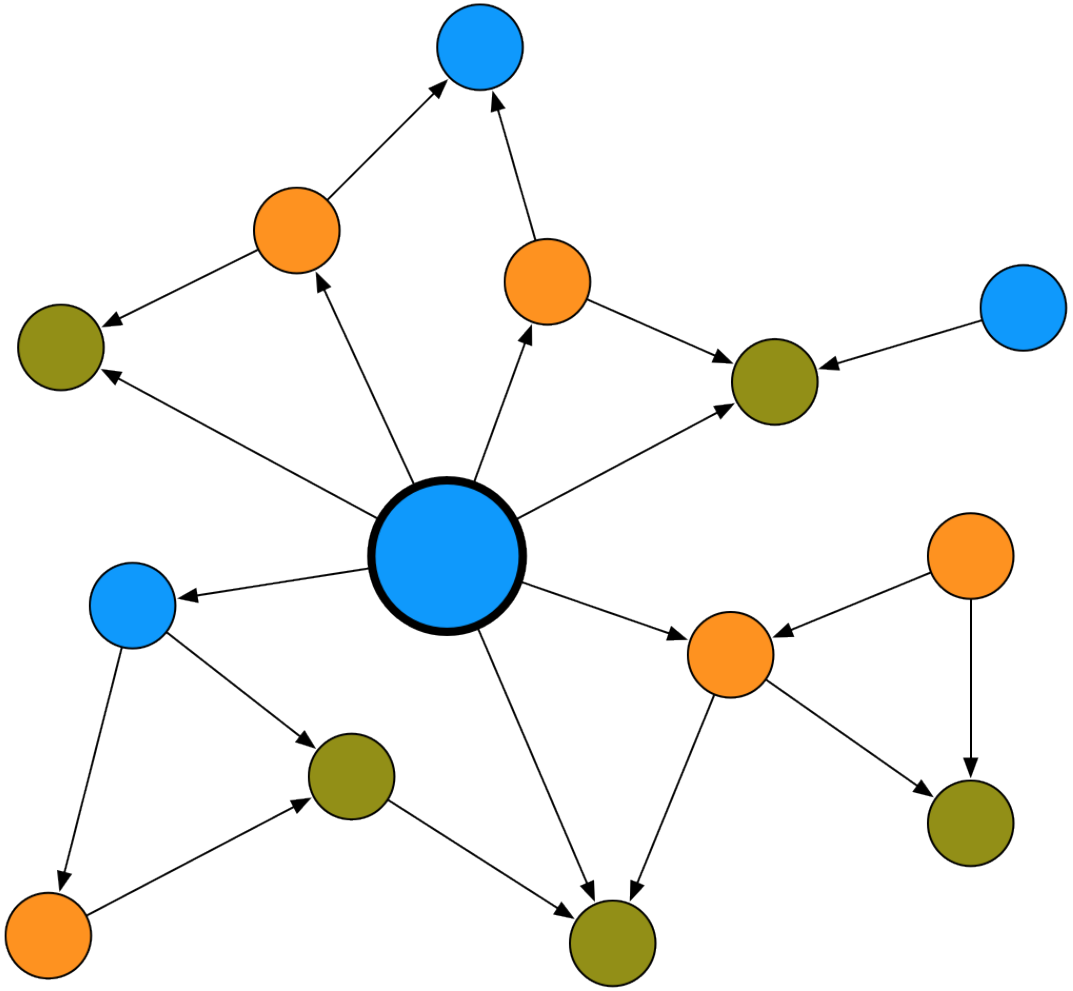
Pattern



# Start Node

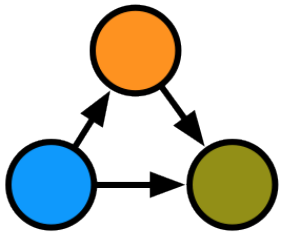


Pattern

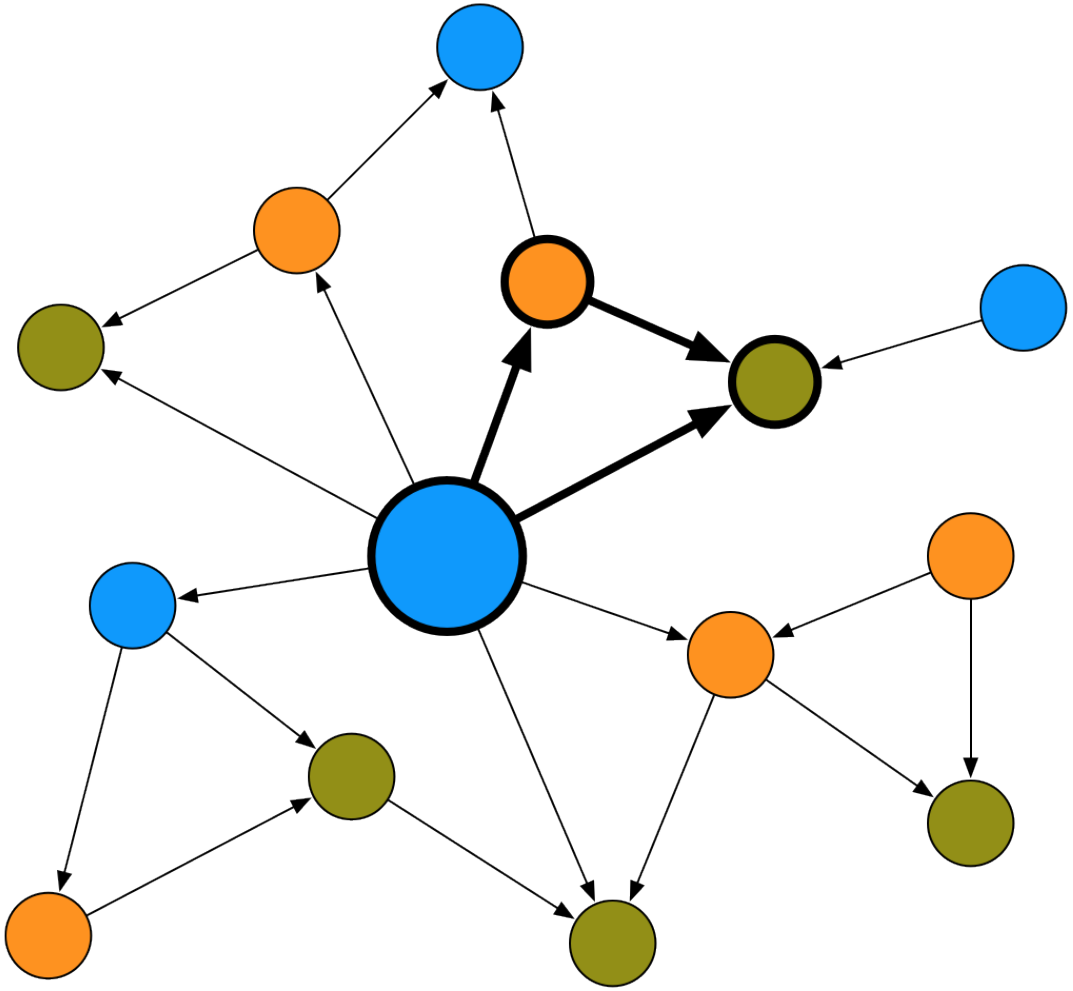


neotechnology
graphs are everywhere

# Match

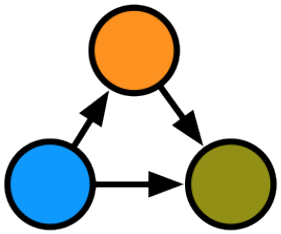


Pattern

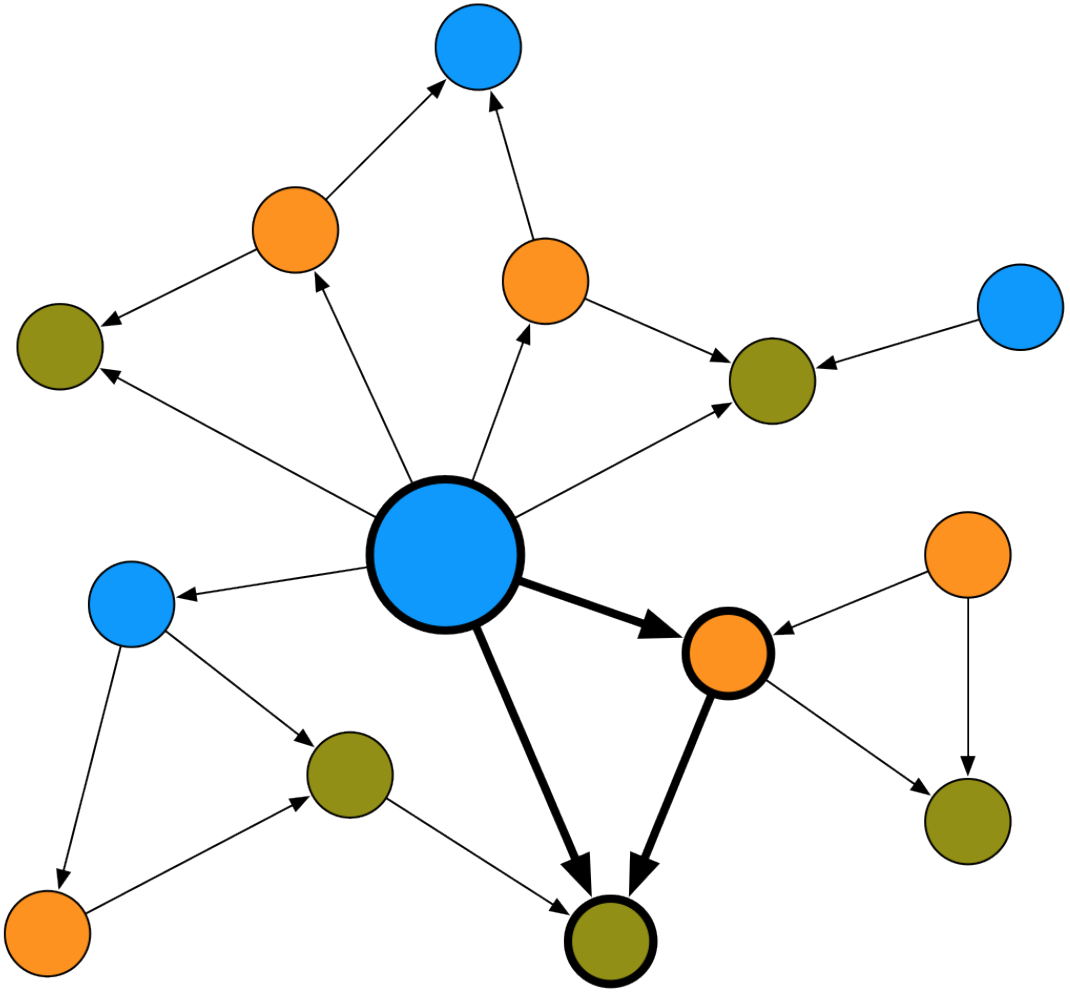


neotechnology
graphs are everywhere

# Match

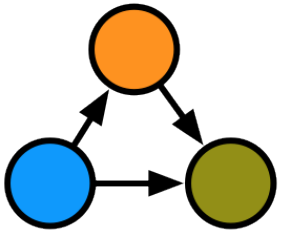


Pattern

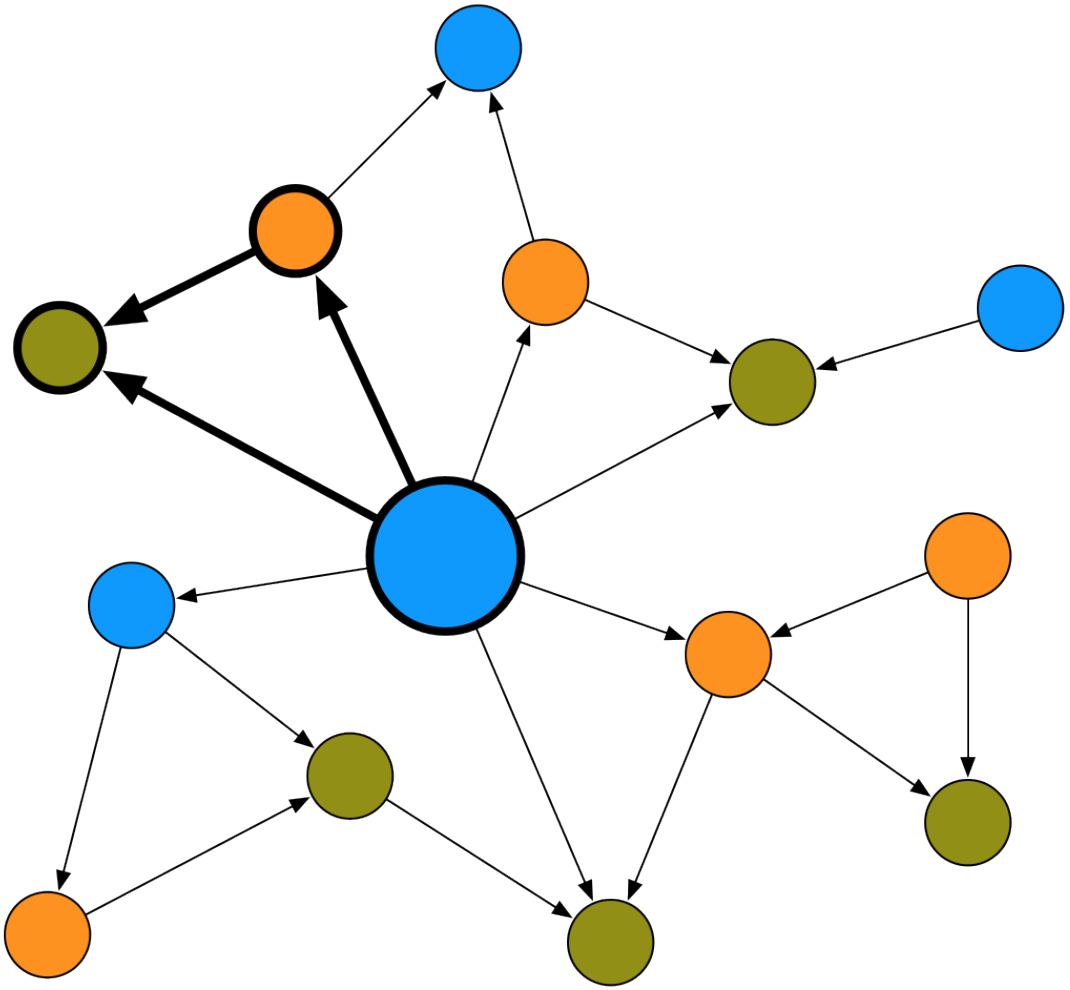


neotechnology
graphs are everywhere

# Match

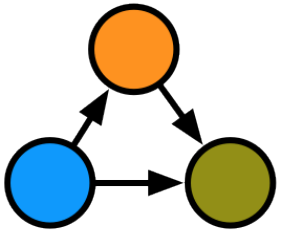


Pattern

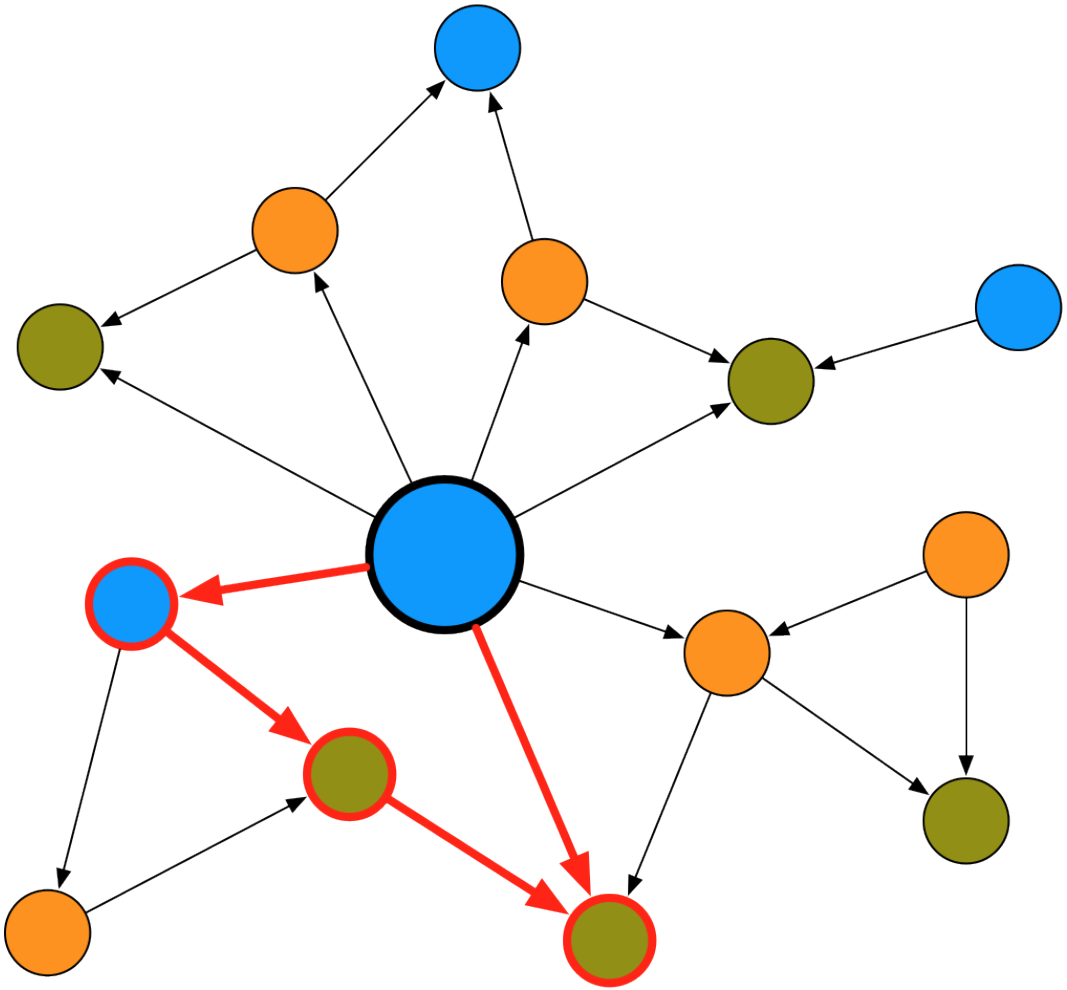


neotechnology
graphs are everywhere

# Non-Match

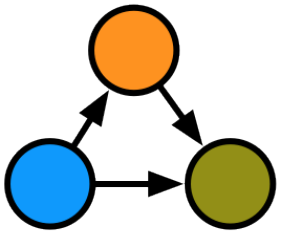


Pattern

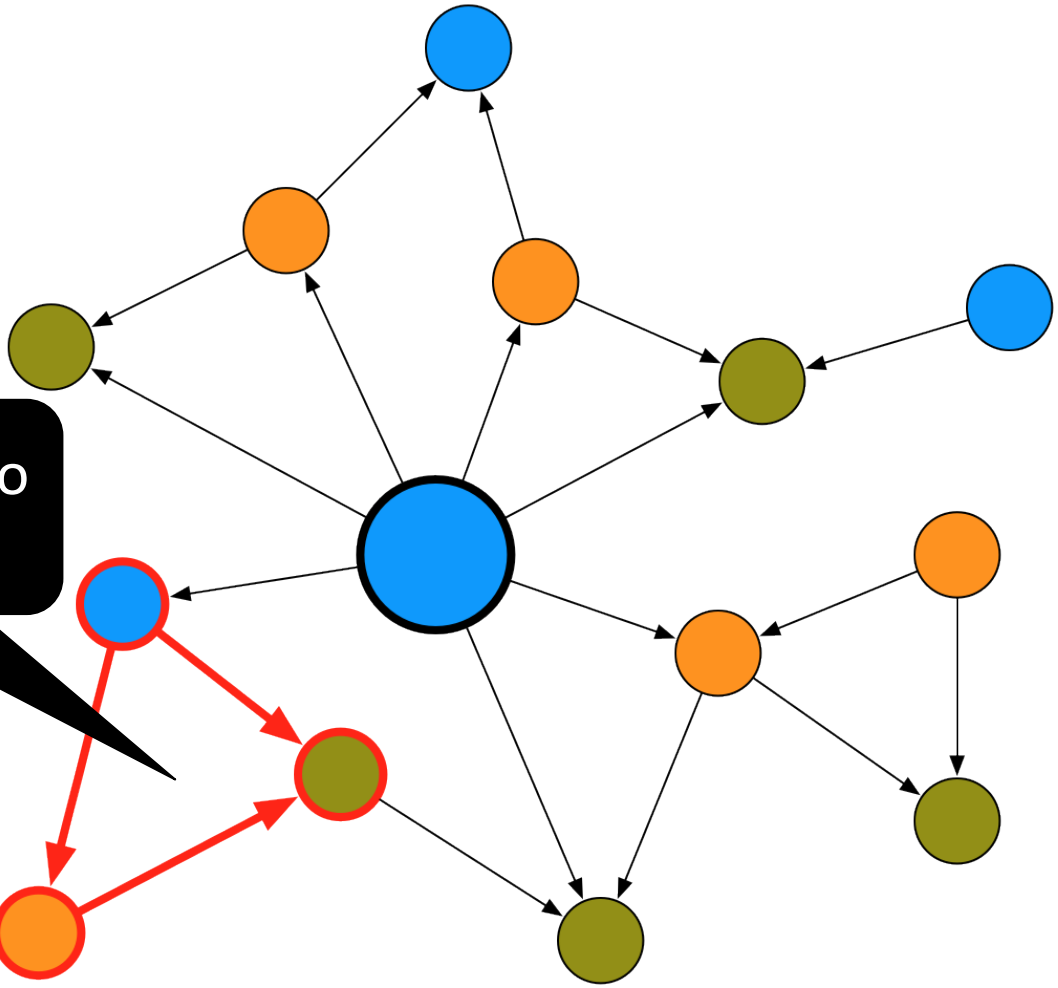


neotechnology
graphs are everywhere

# Non-Match

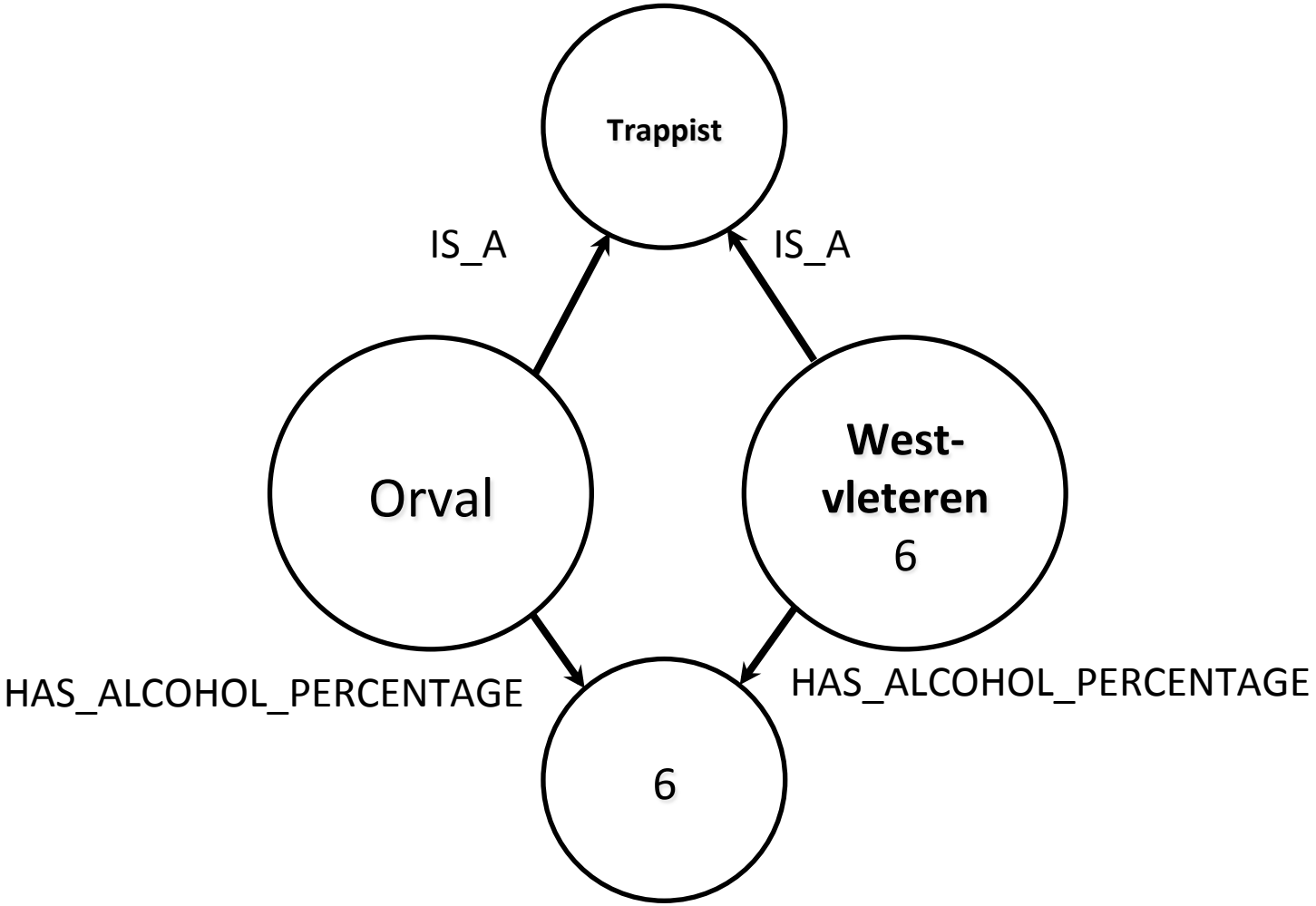


Pattern



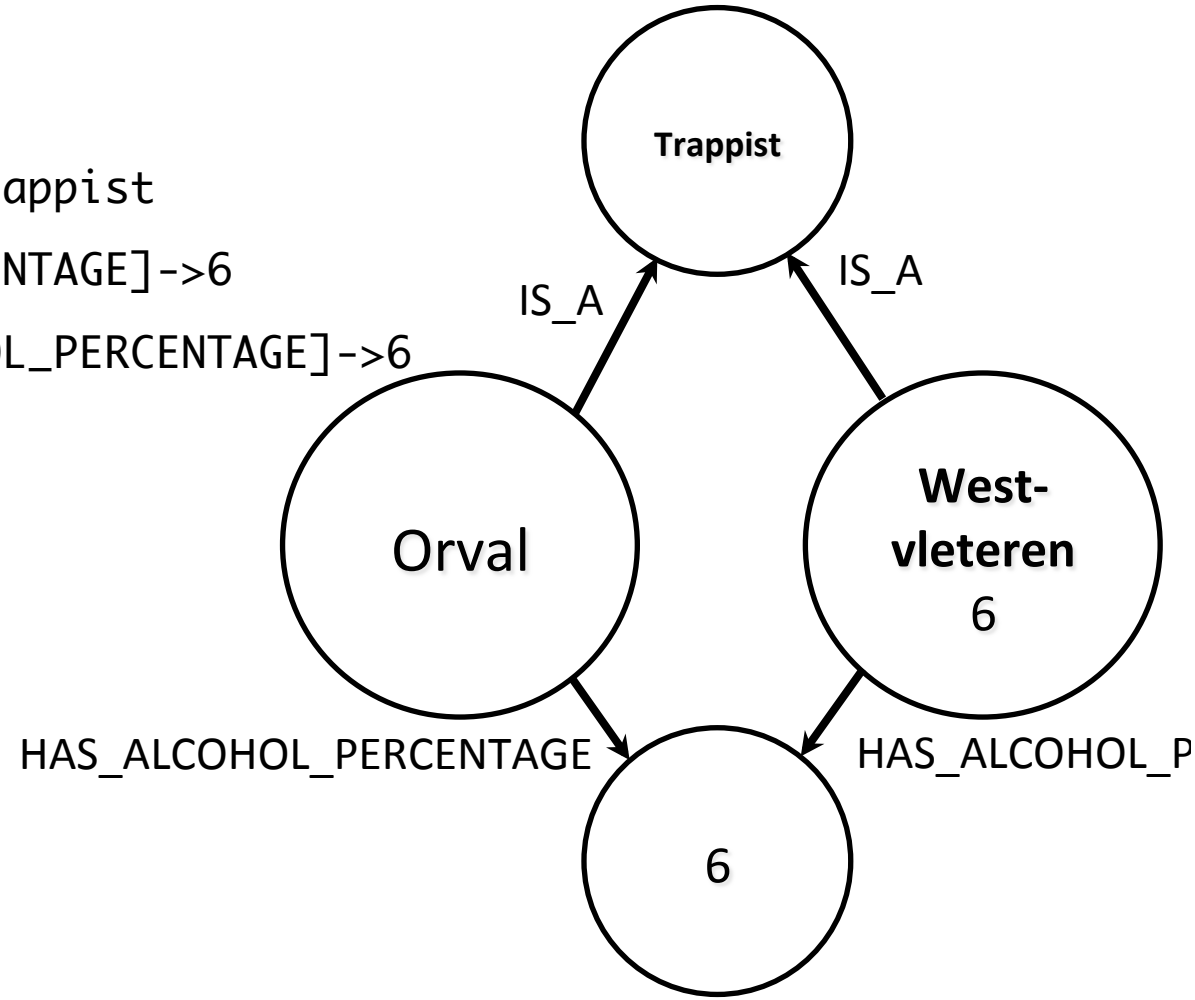

neotechnology
graphs are everywhere

# Beers of a similar type



# Cypher Pattern

```
Orval-[:IS_A]->Trappist
Westvleteren6-[:IS_A]->Trappist
Orval-[:HAS_ALCOHOL_PERCENTAGE]->6
Westvleteren6[:HAS_ALCOHOL_PERCENTAGE]->6
```



# Beers of same brewery

```
start
duvel=node:node_auto_index(name="Duvel")
match
   Duvel<-[:BREWS]-brewery,
   otherbeer-[:BREWS]-brewery,
   otherbeer-[:IS_A]->beertype
Return
otherbeer AS name,
collect(beertype.name) AS beertype;
```

# Beers of same brewery

```
start orval=node:node_auto_index(name="Orval")
match
    orval<-[:BREWS]-brewery,
    otherbeer-[:BREWS]-brewery,
    otherbeer-[:IS_A]->beertype
return
otherbeer.name AS name,
collect(beertype.name) AS beertype;
```

neotechnology
graphs are everywhere

# Find Start Node

```
start orval=node:node_auto_index(name="Orval")
match
    orval-[:BREWS]->brewery,
    otherbeer-[:BREWS]-brewery,
    otherbeer-[:IS_A]->beertype
return
otherbeer.name AS name,
collect(beertype.name) AS beertype;
```

# Describe Pattern

```
start orval=node:node_auto_index(name="Orval")
match
    orval<-[:BREWS]-brewery,
    otherbeer-[:BREWS]-brewery,
    otherbeer-[:IS_A]->beertype
return
otherbeer.name AS name,
collect(beertype.name) AS beertype;
```

# Construct Results

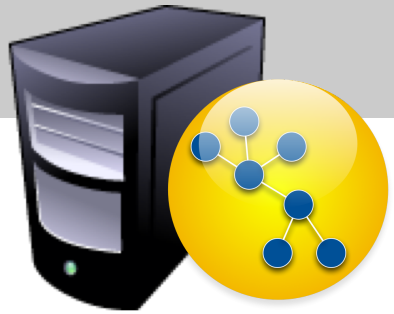
```
start orval=node:node_auto_index(name="Orval")
match
    orval<-[:BREWS]-brewery,
    otherbeer-[:BREWS]-brewery,
    otherbeer-[:IS_A]->beertype
return
otherbeer.name AS name,
collect(beertype.name) AS beertype;
```

# More examples

- All Belgian Trappists
- Same Alcohol Percentage as Duvel
- Same Alcohol Percentage and BeerType as Duvel, Orval
- All paths between two beers

- Using an in-graph alcoholpercentage-index

# Graph Visualisations

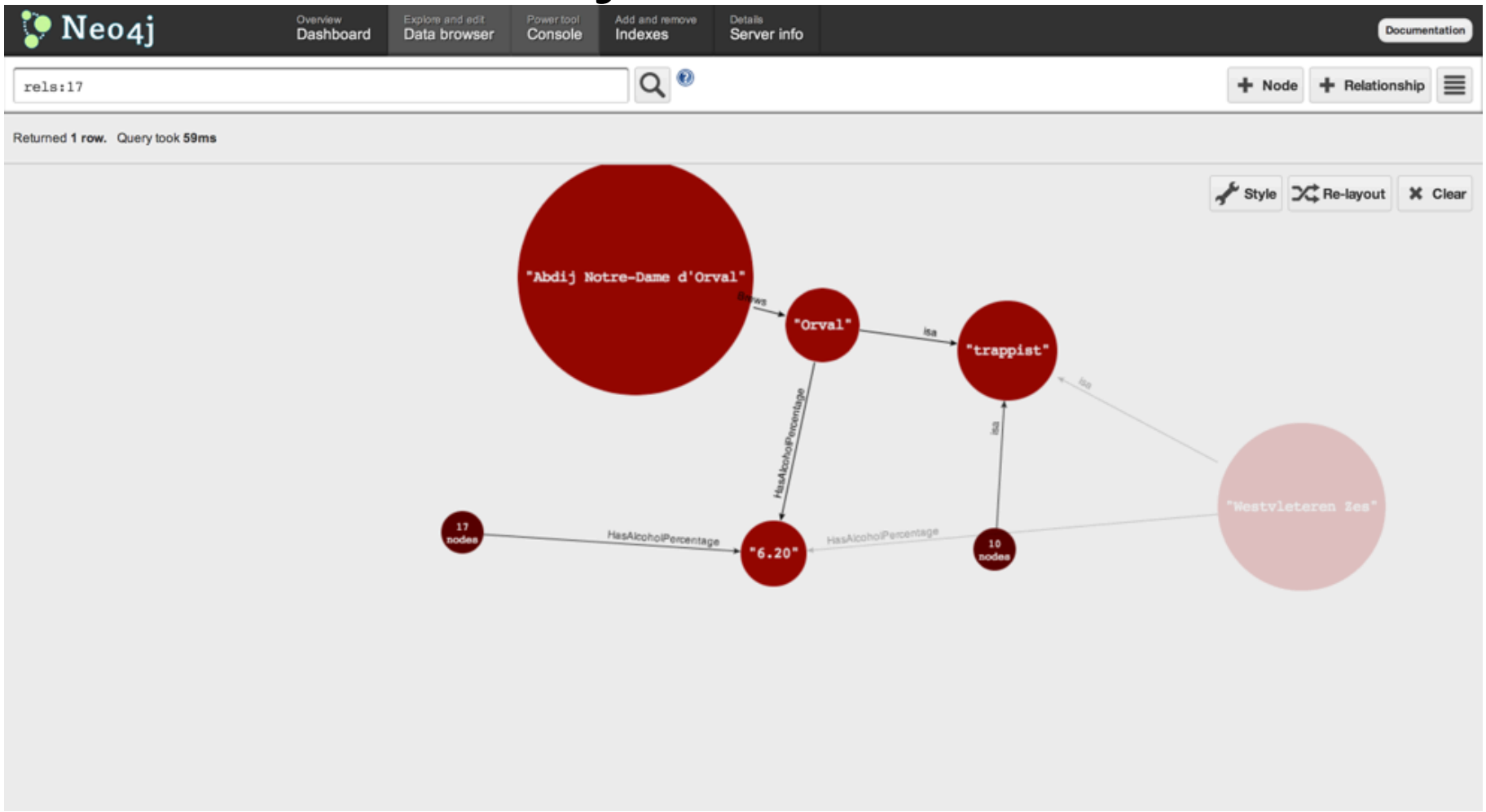


- Web Admin
- Neography / Neovigator
- Neoclipse
- Gephi, Prefuse, d3.js, etc….

neotechnology
graphs are everywhere

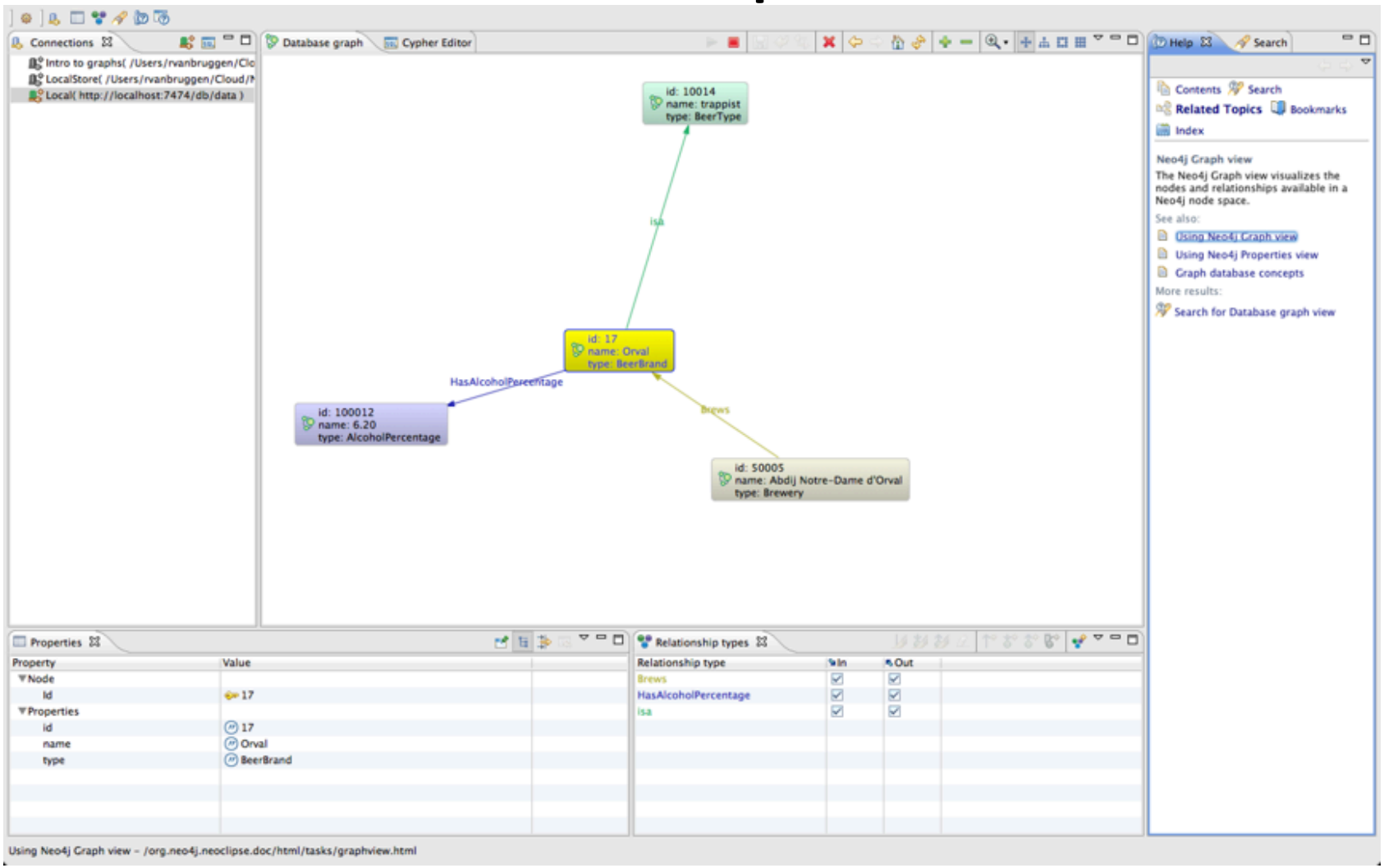
goto; conference

# Neo4j Web Admin



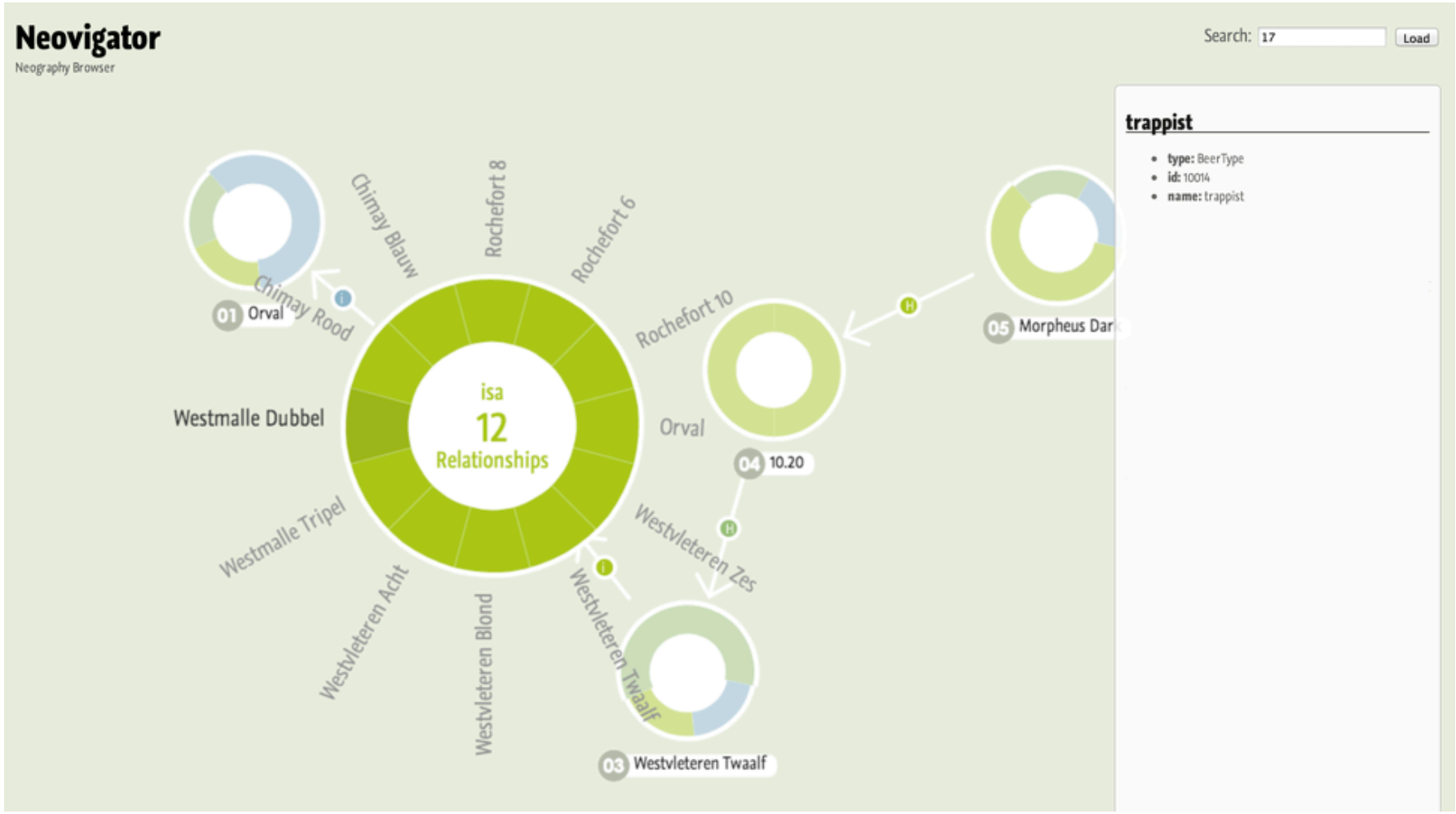

neotechnology
graphs are everywhere

# Neoclipse




neotechnology
graphs are everywhere

# Neovigator

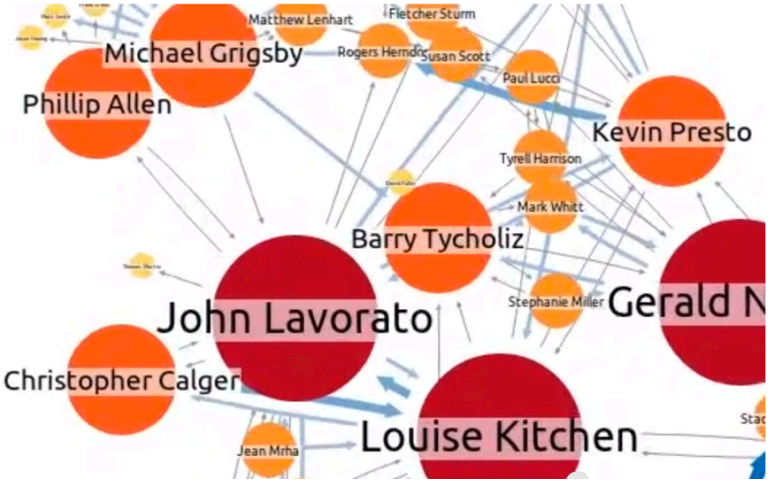



neotechnology
graphs are everywhere

# Other Visualisation options:

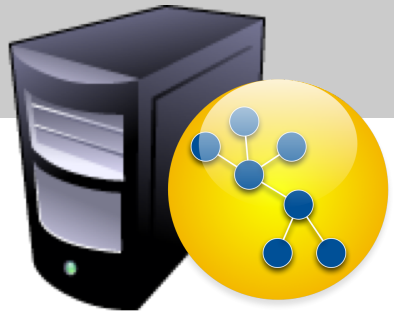
- Linkurio.us


- KeyLines




neotechnology
graphs are everywhere

# Case Studies



neo technology
graphs are everywhere

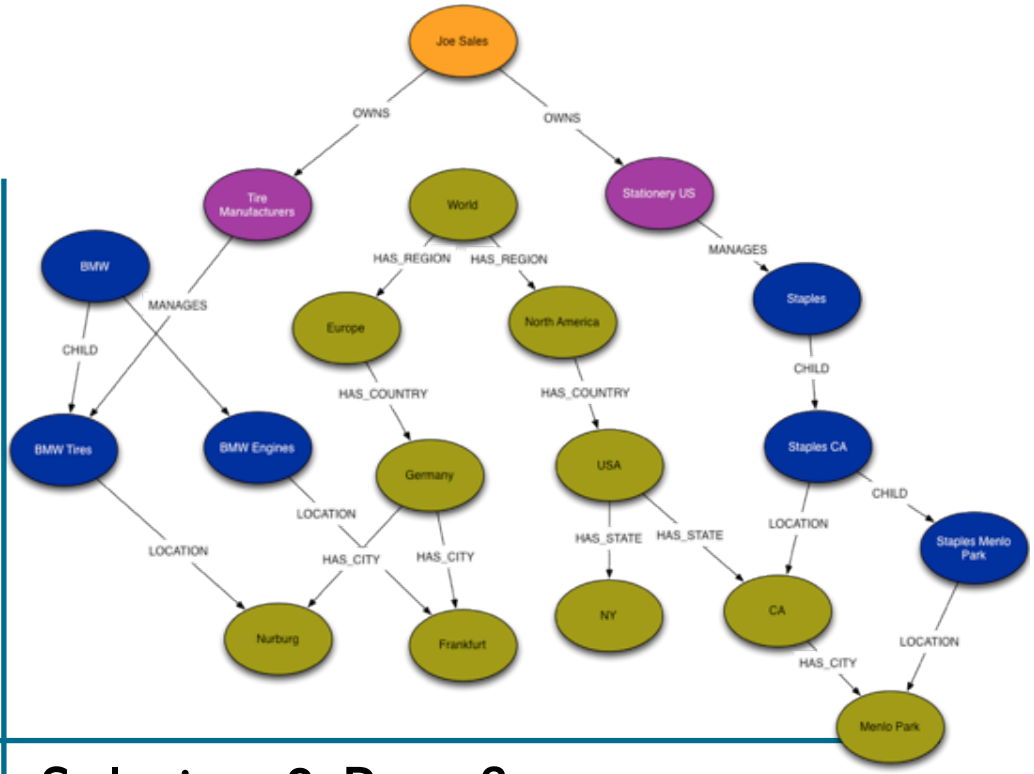
goto; conference

*Industry: CommunicationsUse case: Master Data Management*
**San Jose, CA**

# Cisco HMP

## Background

- One of the world's largest communications equipment manufacturers
- #91 Global 2000. $44B in annual sales.
- Needed a system that could accommodate its master data hierarchies in a performant way
- HMP is a Master Data Management system at whose heart is Neo4j. Data access services available 24x7 to applications companywide




## Business problem

- Sales compensation system had become unable to meet Cisco's needs
- Existing Oracle RAC system had reached its limits:
- *Insufficient flexibility for handling complex organizational hierarchies and mappings*
- *"Real-time" queries were taking > 1 minute!*
- Business-critical "P1" system needs to be continually available, with zero downtime

## Solution & Benefits

- Cisco created a new system: the Hierarchy Management Platform (HMP)
- Allows Cisco to manage master data centrally, and centralize data access and business rulesNeo4j provided "Minutes to Milliseconds" performance over Oracle RAC, serving master data in real time
- The graph database model provided exactly the flexibility needed to support Cisco's business rules
- HMP so successful that it has expanded to include product hierarchy

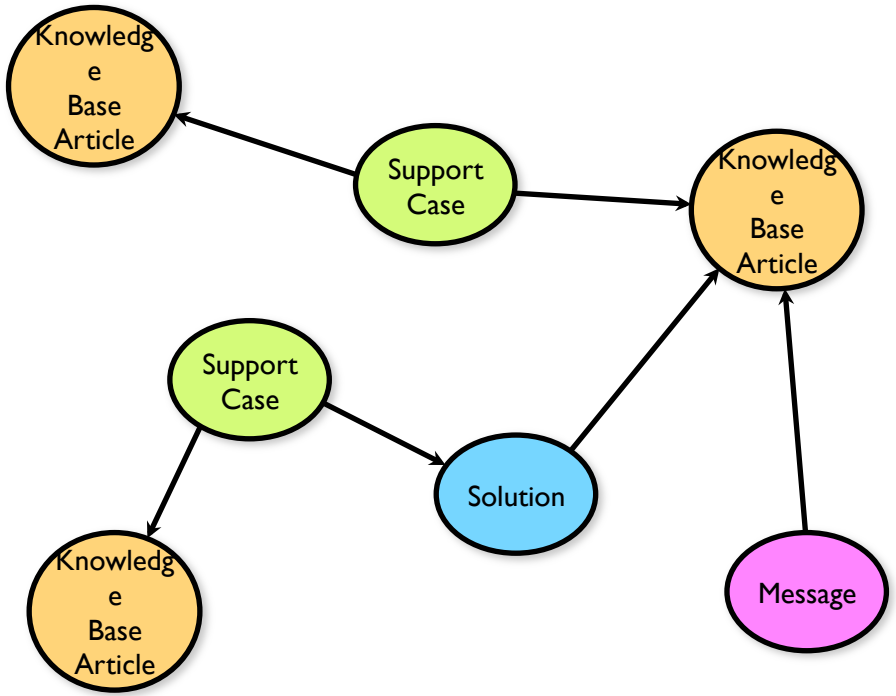
neo technology
graphs are everywhere

*Industry: Communications*
*Use case: Recommendations*
**San Jose, CA**

**Cisco.com**

## Background

- Cisco.com serves customer and business customers with Support Services
- Needed real-time recommendations, to encourage use of online knowledge base
- Cisco had been successfully using Neo4j for its internal master data management solution.
- *Identified a strong fit for online recommendations*




## Business problem

- Call center volumes needed to be lowered by improving the efficacy of online self service
- Leverage large amounts of knowledge stored in service cases, solutions, articles, forums, etc.
- Problem resolution times, as well as support costs, needed to be lowered

## Solution & Benefits

- Cases, solutions, articles, etc. continuously scraped for cross-reference links, and represented in Neo4j
- Real-time reading recommendations via Neo4j
- Neo4j Enterprise with HA cluster
- The result: customers obtain help faster, with decreased reliance on customer support

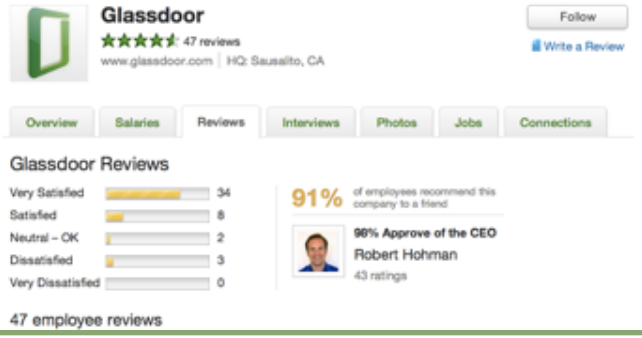
neotechnology
graphs are everywhere

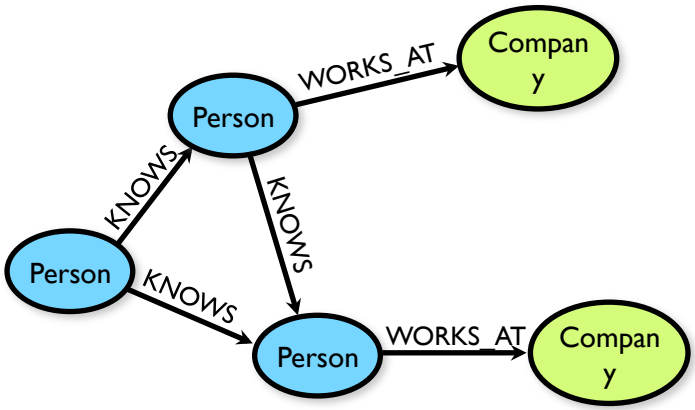
glassdoor.com

*Industry: Online Job Search*
*Use case: Social / Recommendations*
**Sausalito, CA**

# Background

- Online jobs and career community, providing anonymized inside information to job seekers




# Business problem

- Wanted to leverage known fact that most jobs are found through personal & professional connections
- Needed to rely on an existing source of social network data. Facebook was the ideal choice.
- End users needed to get instant gratification
- Aiming to have the best job search service, in a very competitive market

# Solution & Benefits

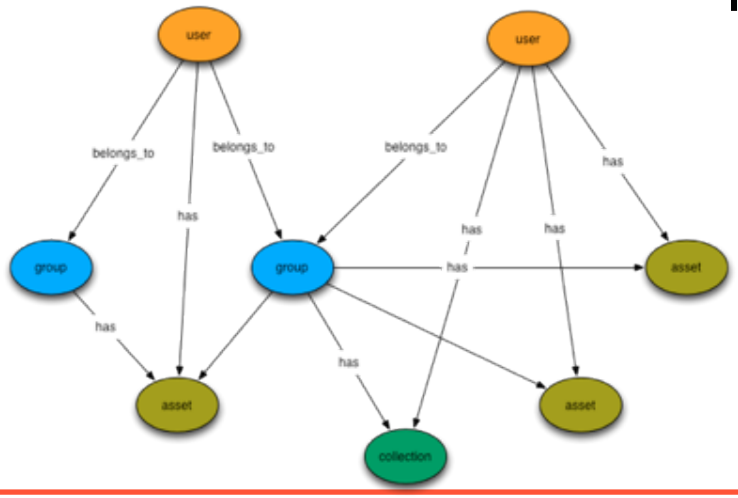
- First-to-market with a product that let users find jobs through their network of Facebook friends
- Job recommendations served real-time from Neo4j
- Individual Facebook graphs imported real-time into Neo4j
- Glassdoor now stores > 50% of the entire Facebook social graph
- Neo4j cluster has grown seamlessly, with new instances being brought online as graph size and load have increased

*Industry: Web/ISVUse case: Content Management, Social, Access Control*
**San Jose, CA**

## Background

- One of the ten largest software companies globally
- $4B+ in revenue. Over 11,000 employees.
- Launched Creative Cloud in 2012, allowing its Creative Suite users to collaborate via the Cloud


## User-Content-Access Graph




## Business problem

- Adobe needed a highly robust and available, 24x7 distributed global system, supporting collaboration for users of its highest revenue product line
- Storing creative artifacts in the cloud meant managing access rights for (eventually) millions of users, groups, collections, and pieces of content
- Complex access control rules controlling who was connected to whom, and who could see or edit what, proved a significant technical challenge

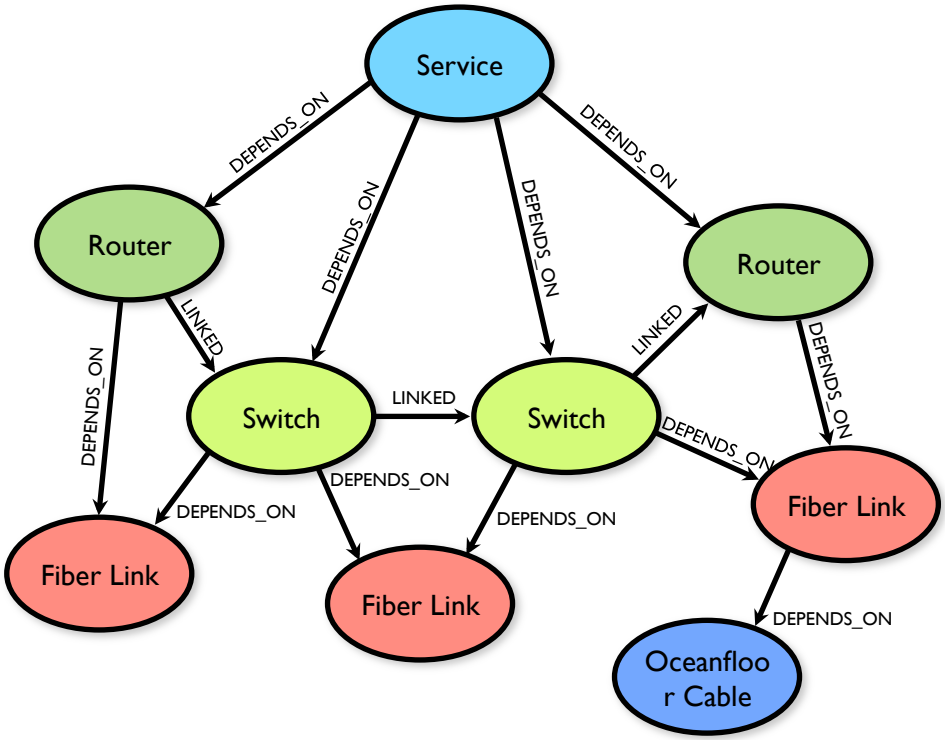
## Solution & Benefits

- Selected Neo4j to meet very aggressive project deadlines. The flexibility of the graph model, and performance, were the two major selection factors. Easily evolve the system to meet tomorrow's needs
- Extremely high availability and transactional performance requirements. 24x7 with no downtime.
- Neo4j allows consistently fast response times with complex queries, even as the system grows
- First (and possibly still only) database cluster to run across three Amazon EC2 regions: U.S., Europe, Asia

*Industry: Communications*
*Use case: Network Management*
**Paris, France**

# Background

- Second largest communications company in France
- Part of Vivendi Group, partnering with Vodafone




# Business problem

- Infrastructure maintenance took one full week to plan, because of the need to model network impacts
- Needed rapid, automated "what if" analysis to ensure resilience during unplanned network outages
- Identify weaknesses in the network to uncover the need for additional redundancy
- Network information spread across > 30 systems, with daily changes to network infrastructure
- Business needs sometimes changed very rapidly

# Solution & Benefits

- Flexible network inventory management system, to support modeling, aggregation & troubleshooting
- Single source of truth (Neo4j) representing the entire network
- Dynamic system loads data from 30+ systems, and allows new applications to access network data
- Modeling efforts greatly reduced because of the near 1:1 mapping between the real world and the graph
- Flexible schema highly adaptable to changing business requirements

Deutsche Telekom

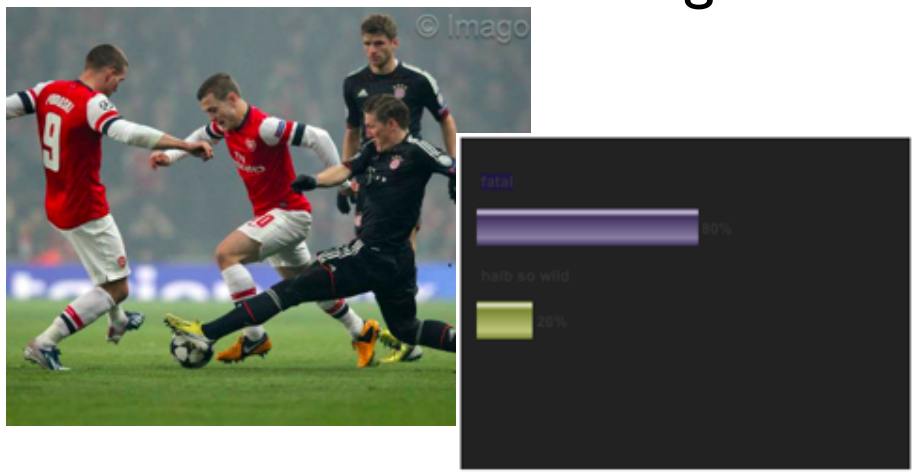
*Industry: Communications*
*Use case: Social gaming*
**Frankfurt, Germany**

## Background

- Europe's largest communications company
- Provider of mobile & land telephone lines to consumers and businesses, as well as internet services, television, and other services

| > 236,000 | 50 | > 58 bn. € |
|---|---|---|
| Employees worldwide in 2011 | Countries | Revenue in 2011 |

## Interactive Television Programming




## Business problem

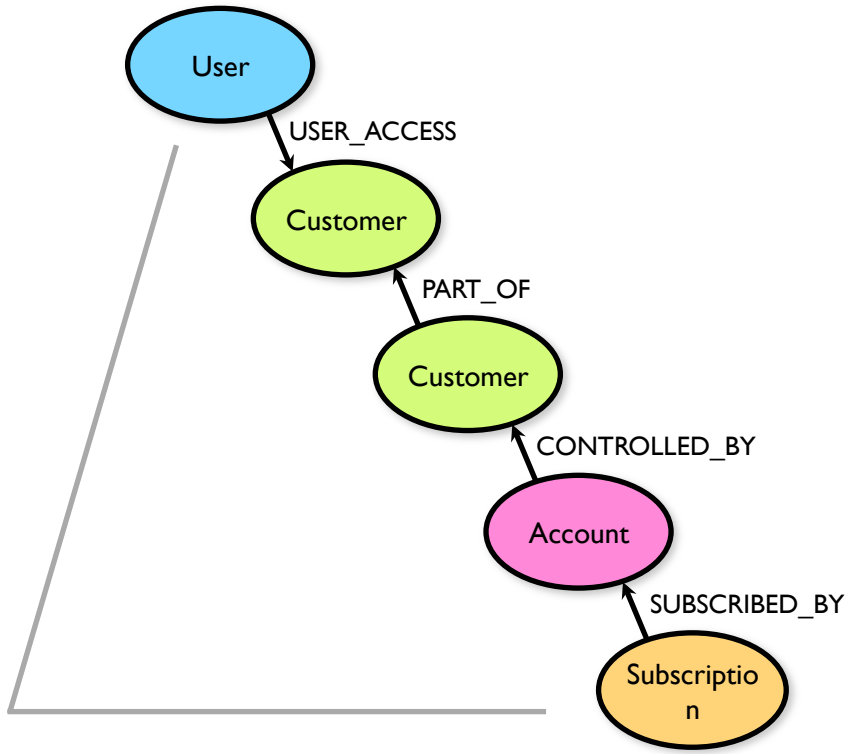
- The Fanorakel application allows fans to have an interactive experience while watching sports
- Fans can vote for referee decisions and interact with other fans watching the game
- Highly connected dataset with real-time updates
- Queries need to be served real-time on rapidly changing data
- One technical challenge is to handle the very high spikes of activity during popular games

## Solution & Benefits

- Interactive, social offering gives fans a way to experience the game more closely
- Increased customer stickiness for Deutsche Telekom
- A completely new channel for reaching customers with information, promotions, and ads
- Clear competitive advantage

telenor

*Industry: Communications*
*Use case: Resource Authorization & Access Control*
**Oslo, Norway**




## Background

- 10th largest Telco provider in the world, leading in the Nordics
- Online self-serve system where large business admins manage employee subscriptions and plans
- Mission-critical system whose availability and responsiveness is critical to customer satisfaction

## Business problem

- Degrading relational performance. User login taking minutes while system retrieved access rights
- Millions of plans, customers, admins, groups. Highly interconnected data set w/massive joins
- Nightly batch workaround solved the performance problem, but meant data was no longer current
- Primary system was Sybase. Batch pre-compute workaround projected to reach 9 hours by 2014: longer than the nightly batch window

## Solution & Benefits

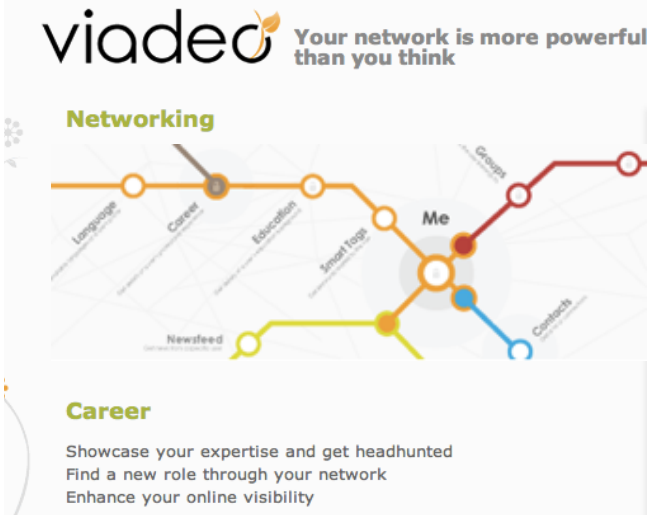
- Moved authorization functionality from Sybase to Neo4j
- Modeling the resource graph in Neo4j was straightforward, as the domain is inherently a graph
- Able to retire the batch process, and move to real-time responses: measured in milliseconds
- Users able to see fresh data, not yesterday's snapshot
- Customer retention risks fully mitigated

viadeo

*Industry: Professional Social NetworkUse case: Social, Recommendations*
**Silicon Valley & France**

# Background

- World's second-largest professional network (after LinkedIn)
- 50M members. 30K+ new members daily.
- Over 400 staff with offices in 12 countries




# Business problem

- Business imperative for real-time recommendations: to attract new users and retain existing ones
- Key differentiator: show members how they are connected to any other member
- Real-time traversals of social graph not feasible with MySQL cluster. Batch precompute meant stale data.
- Process taking longer & longer: > 1 week!

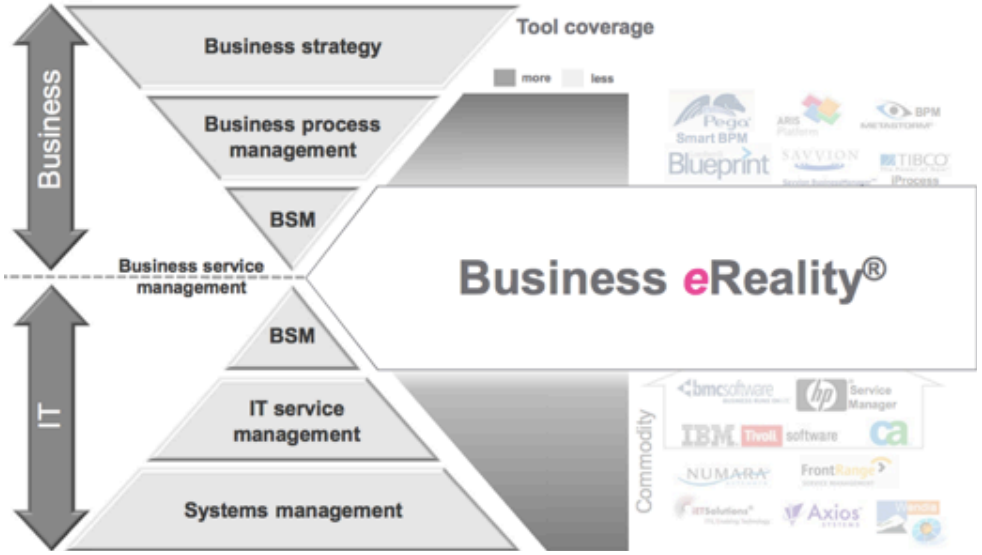
# Solution & Benefits

- Neo4j solution implemented in 8 weeks with 3 part-time programmers
- Able to move from batch to real-time: improved responsiveness with up-to-date data.
- Viadeo (at the time) had 8M members and 35M relationships.
- Neo4j cluster now sits at the heart of Viadeo's professional network, connecting 50M+ professionals

*Industry: Web/ISV, CommunicationsUse*
*case: Data Center Management*
**Zürich, Switzerland**

# Background

- Junisphere AG is a Zurich-based IT solutions provider
- Founded in 2001. Profitable. Self funded.
- Software & services.
- Novel approach to infrastructure monitoring: Starts with the end user, mapped to business processes and services, and dependent infrastructure




# Business problem

- "Business Service Management" requires mapping of complex graph, covering: business processes--> business services--> IT infrastructure
- Embed capability of storing and retrieving this information into OEM application
- Re-architecting outdated C++ application based on relational database, with Java

# Solution & Benefits

- Actively sought out a Java-based solution that could store data as a graph
- Domain model is reflected directly in the database:
- *"No time lost in translation"*
- *"Our business and enterprise consultants now speak the same language, and can model the domain with the database on a 1:1 ratio."*
- Spring Data Neo4j strong fit for Java architecture

teachscape

*Industry: EducationUse case: Resource Authorization & Access Control*

***San Francisco, CA***

# Background

•Teachscape, Inc. develops online learning tools for K-12 teachers, school principals, and other instructional leaders.

•Teachscape evaluated relational as an option, considering MySQL and Oracle.

•Neo4j was selected because the graph data model provides a more natural fit for managing organizational hierarchy and access to assets.


# Business problem

•Neo4j was selected to be at the heart of a new architecture.

•The user management system, centered around Neo4j, will be used to support single sign-on, user management, contract management, and end-user access to their subscription entitlements.

# Solution & Benefits

•Domain and technology fit

•simple domain model where the relationships are relatively complex. Secondary factors included support for transactions, strong Java support, and well-implemented Lucene indexing integration

•Speed and Flexibility

•The business depends on being able to do complex walks quickly and efficiently. This was a major factor in the decision to use Neo4j.

•Ease of Use

•accommodate efficient access for home-grown and commercial off-the-shelf applications, as well as ad-hoc use.

•Extreme availability & performance with Neo4j clustering

•Hugely simplified queries, vs. relational for complex routing

•Flexible data model can reflect real-world data variance much better than relational

•"Whiteboard friendly" model easy to understand

SevenBridges genomics

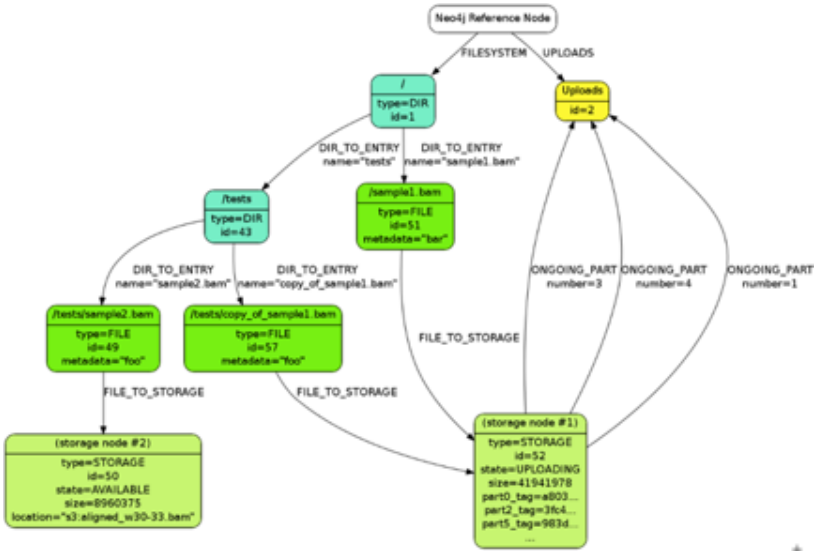
*Industry: Life SciencesUse case: Content Management*
**Cambridge, Massachusetts**

# Background

•Bioinformatics company offering gene sequencing "as a service" (over the web)

•Provider of genomic information services

•Needed a new platform to support storage & retrieval of sequenced genomes in the cloud




# Business problem

•Neo4j is used to store metadata about each sequenced genome (including a pointer to the sequenced genome itself, which is a binary file stored on Amazon S3), and to support search and other forms of information processing against the genomic data.

•graph database was chosen because "Our specific domain maps naturally onto graph paradigm".

# Solution & Benefits

•***Domain fit***

•Domain naturally lends itself to a graph representation.

•Graph model determined to be a perfect fit.

•***Agility & Performance***

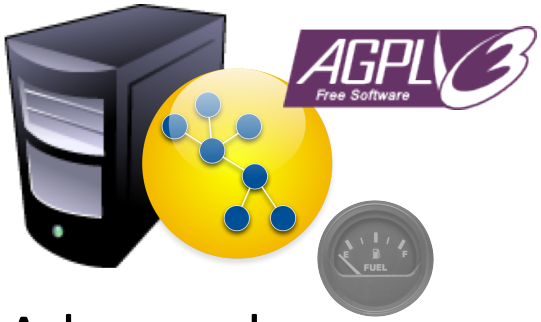
•Saved time with Neo4j as compared to the alternatives.

•Queries "practically write themselves."

•***Solution Completeness***

• "Neo4j is incomparably better than other graph databases."


Community




Advanced


Enterprise

http://neo4j.org/download/

neotechnology
graphs are everywhere

# Contact Us

- Neo Technology UK
  5-11 Lavington Street
  London
  SE1 0NZ
  ENGLAND
  +44 808 189 0493
- rik@neotechnology.com or
- +32 478 686800