Unit 5: ADVANCE DATA ANALYSIS

- Graph
- The NOSQL Universe
- Overview of Graph Databases and Neo4j
- Link analysis
- High Dimensional Clustering

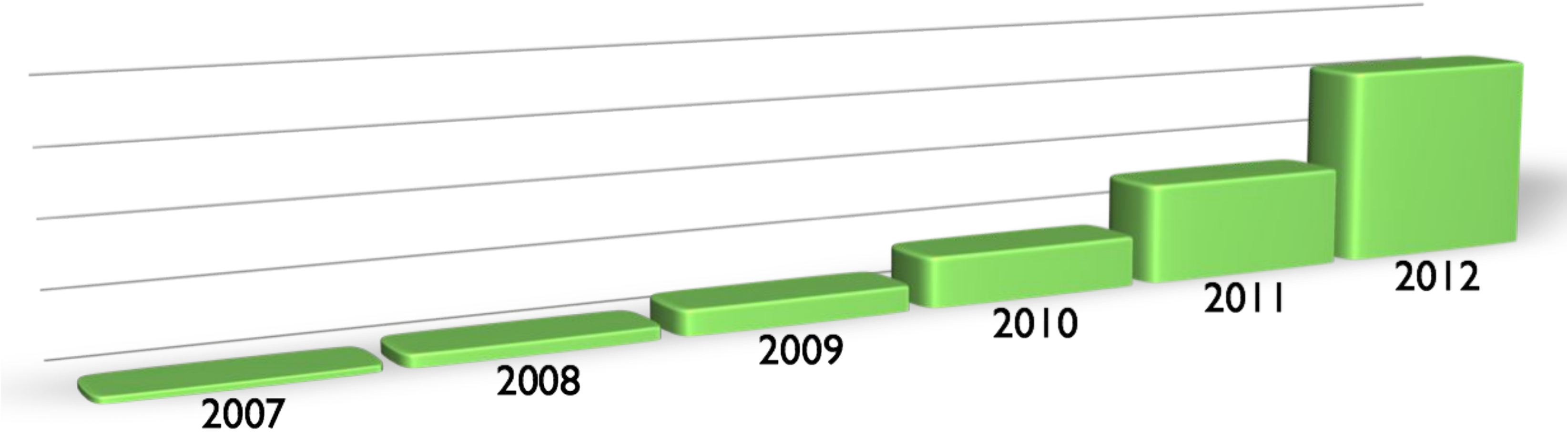
Data, information, knowledge

- **Data -** Facts, observations, or perceptions.
- **Information -** Subset of data, only including those data that possess context, relevance, and purpose.
- **Knowledge -** **A more simplistic view** considers knowledge as being at the highest level in a hierarchy with data (at the lowest level) and information (at the middle level).

- **Data** refers to bare facts void of context.
  - *–A telephone number.*
- **Information** is data in context.
  - *–A phone book.*
- **Knowledge** is information that facilitates action.
  - *–Recognizing that a phone number belongs to a good client, who needs to be called once per week to get his orders.*

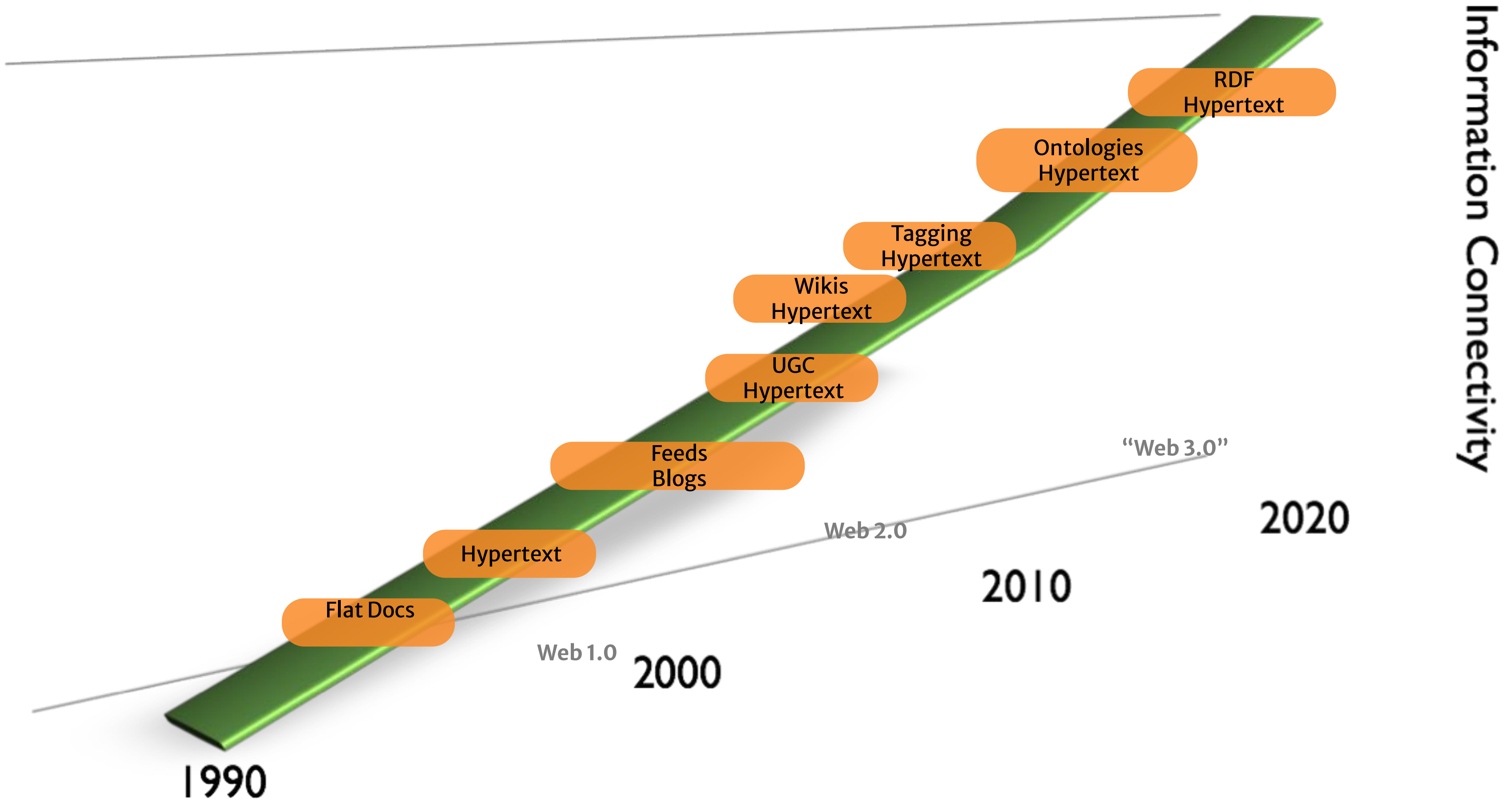
Why NOSQL?

Driving Trends - Data Size

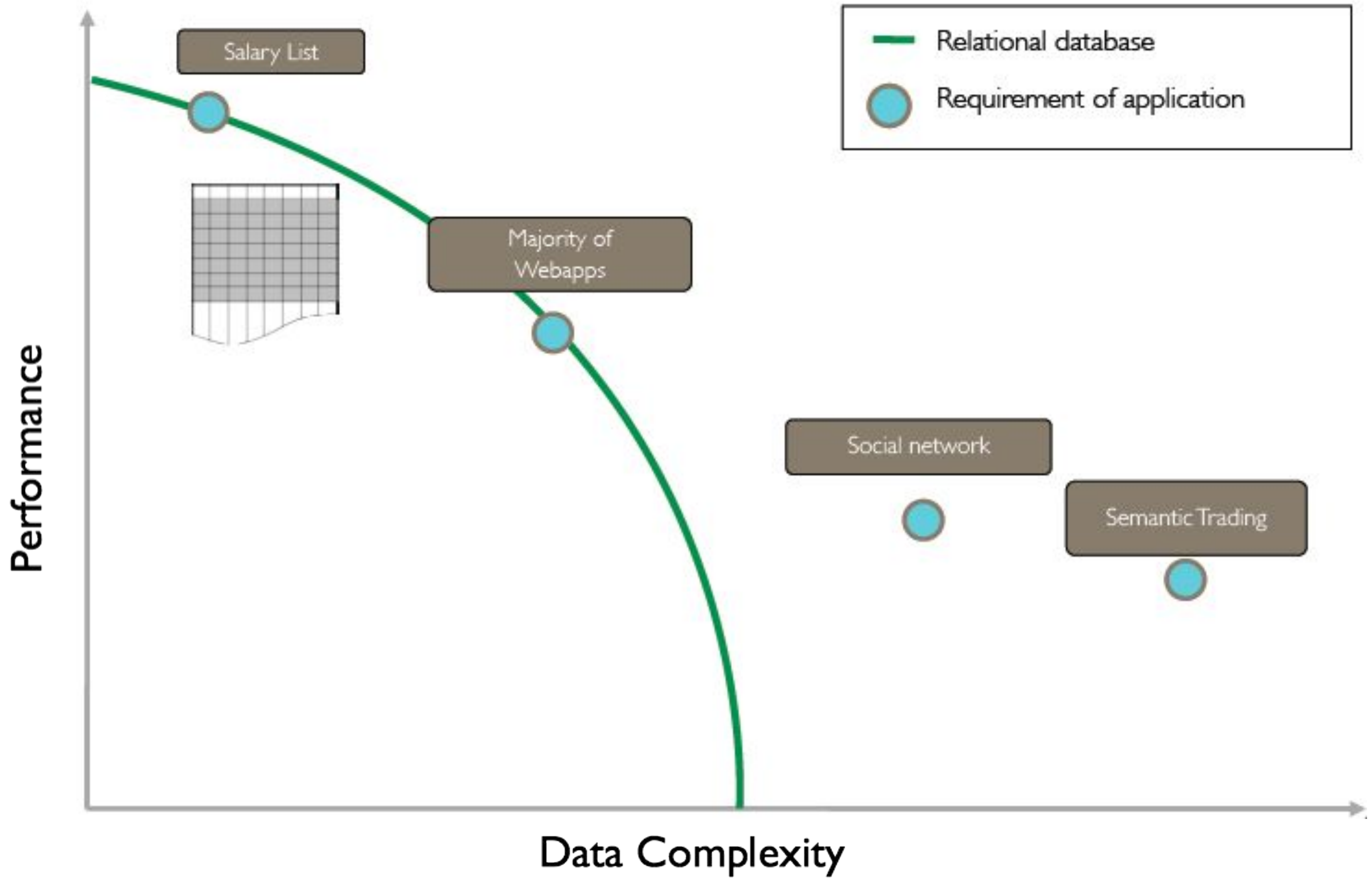


- Data size is increasing exponentially year after year

Driving Trends - Connectivity of Data



RDBMS Performance Curve



NOSQL Database Types

Key-Value

riak redis

Column Family

Cassandra H·BASE

Document

mongoDB CouchDB relax

Graph

Neo4j the graph database

sones

Graph Databases

- Data Model
  - Nodes with properties
  - Named relationships with properties
- Examples
  - Neo4j, Sones GraphDB, OrientDB, InfiniteGraph, AllegroGraph

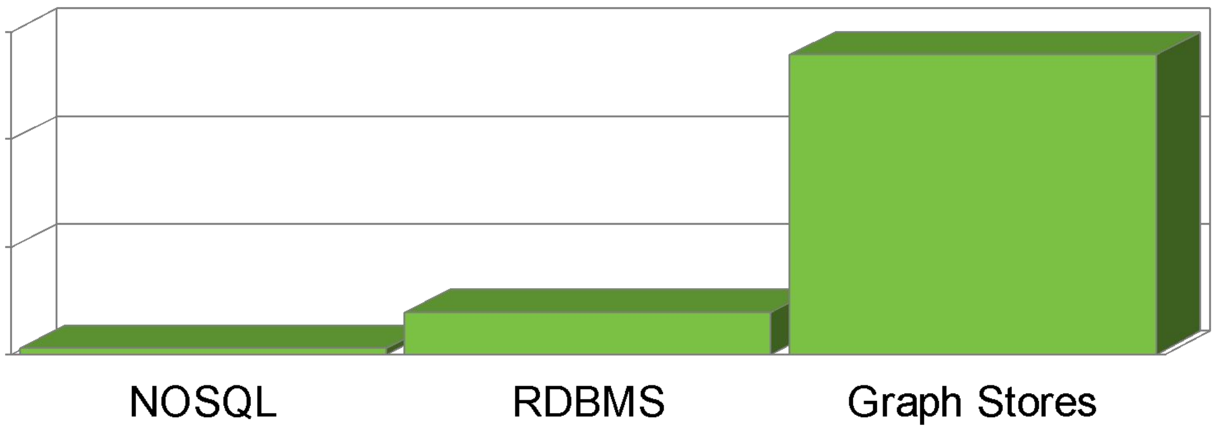
Graph Databases: Strengths and Weaknesses

- Strengths
  - Extremely powerful data model
  - Performant when querying interconnected data
  - Easily to query
- Weaknesses
  - Sharding
  - Rewiring your brain

Typical Use Cases for Graph Databases

- Recommendations
- Business Intelligence
- Social Computing
- Master Data Management
- Geospatial
- Genealogy
- Time Series Data
- Web Analytics
- Bioinformatics
- Indexing RDBMS

Maturity of Data Models

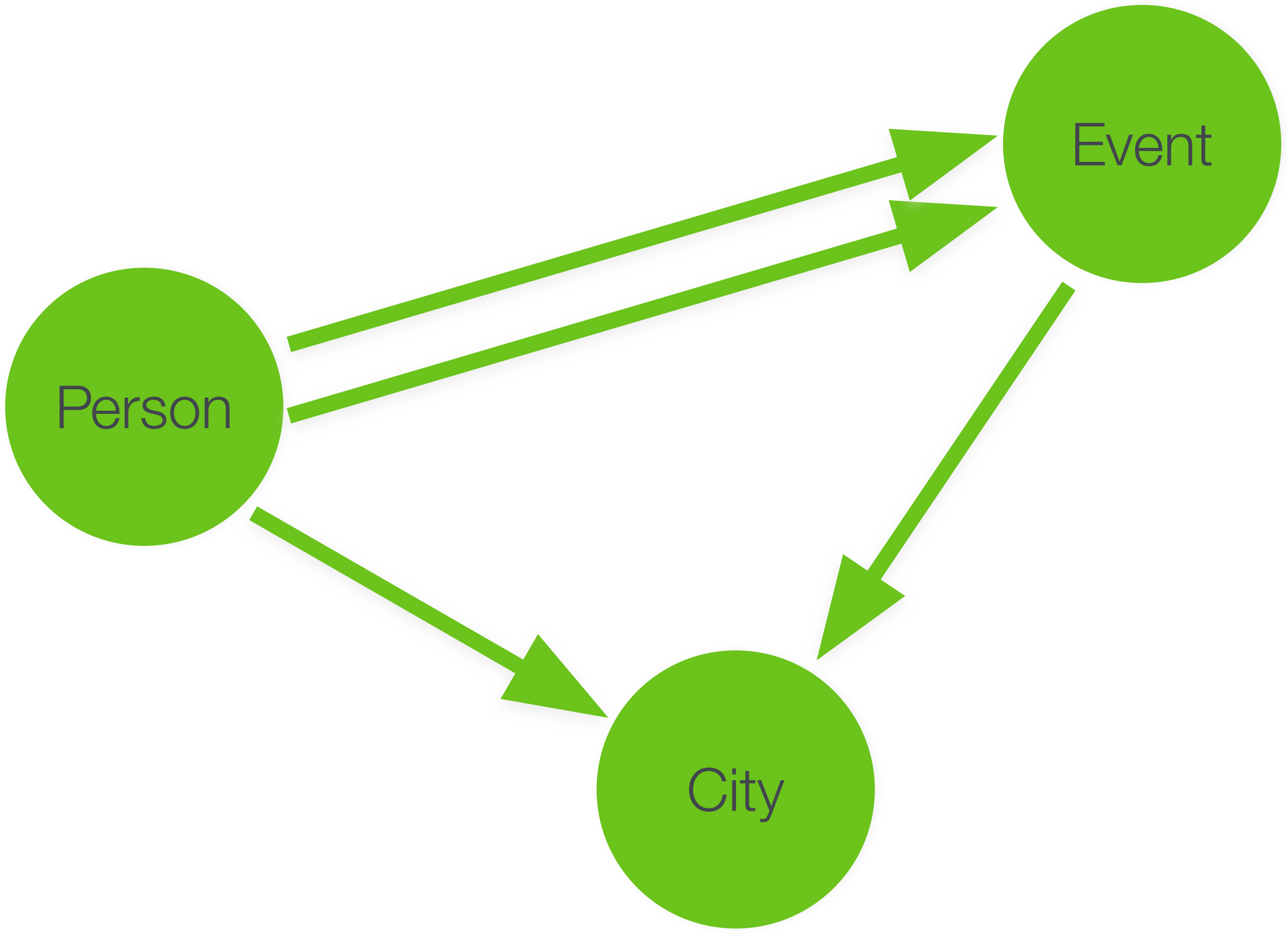


- Most NOSQL: ~6 years
- Relational: 42 years
- Graph Theory: 276 years

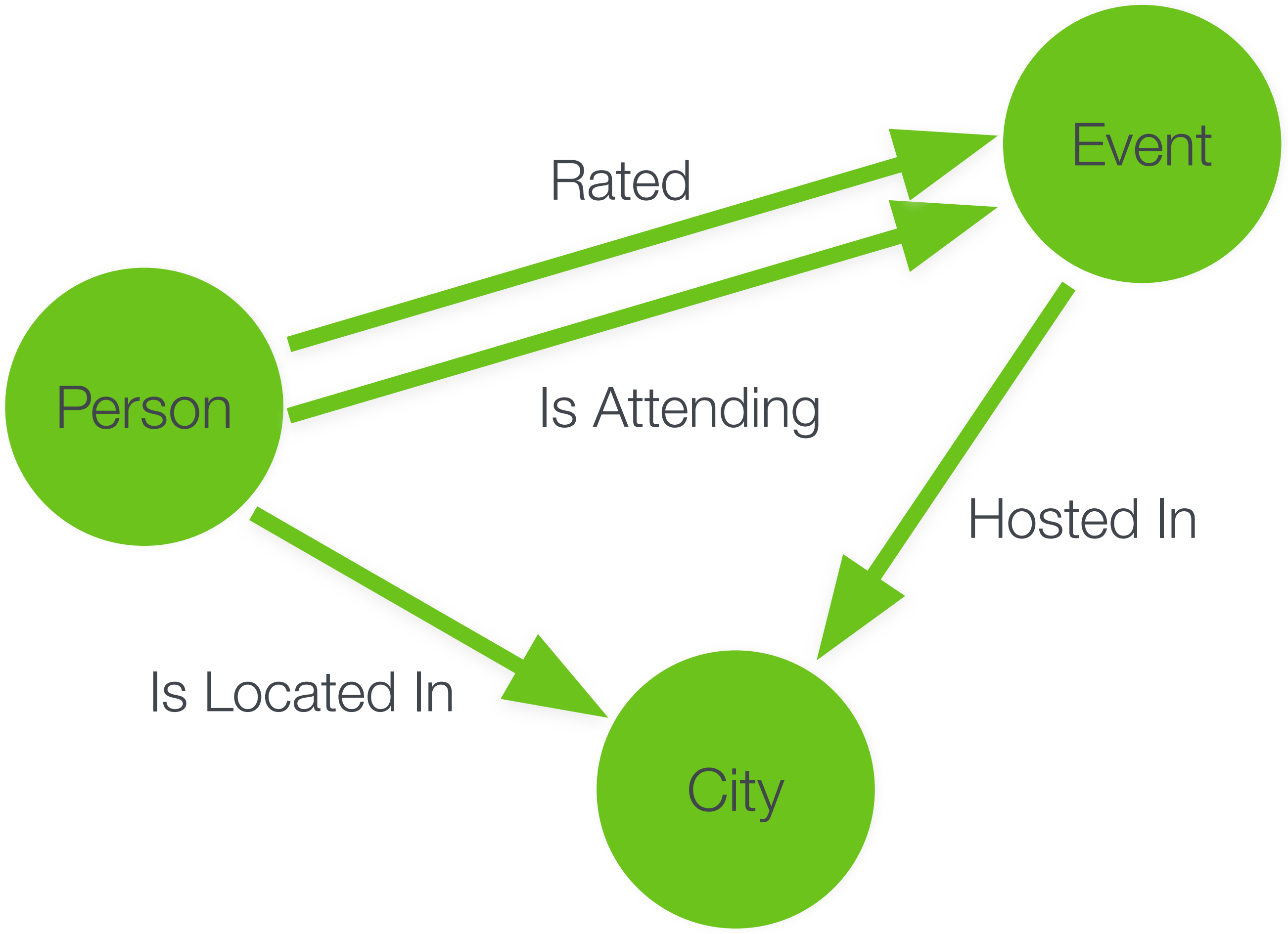
Leonhard Euler

- Inventor of Graph Theory (1736)
- Swiss mathematician

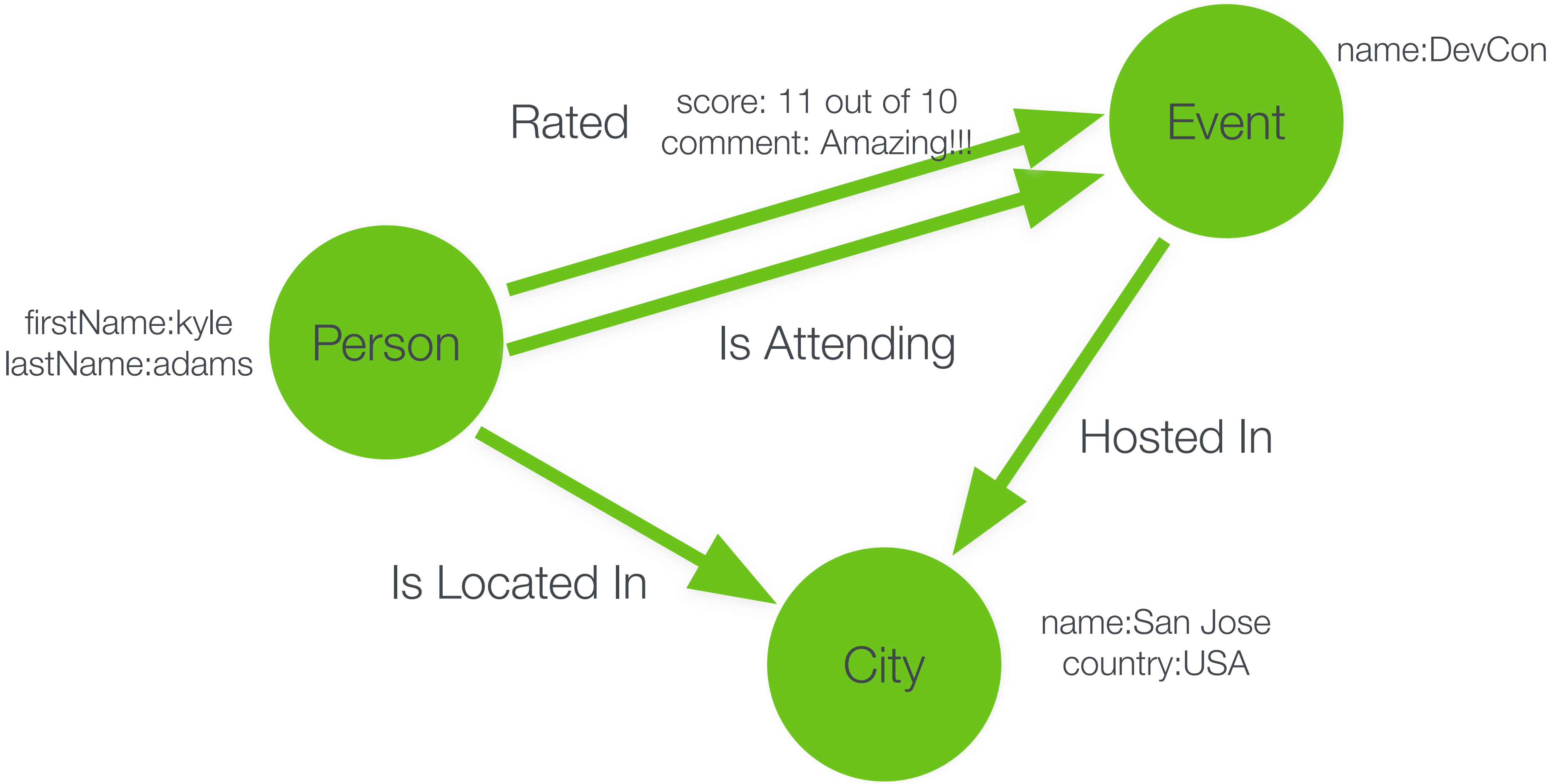
Graph Data Model



Graph Data Model



Graph Data Model



Neo4j
(...finally)

What is Neo4j?

- Leading Open Source graph database
- Embeddable and Server
- ACID compliant
- White board friendly
- Stable
    - Has been in 24/7 operation since 2003

More Reasons Why Neo4j is Great

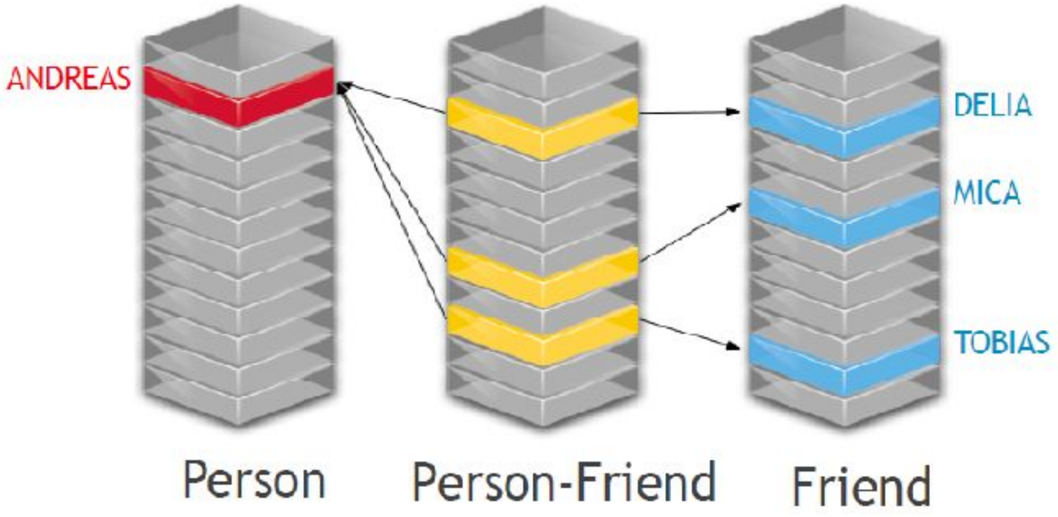
- High performance graph operations
  - Traverse 1,000,000+ relationships/sec on commodity hardware
- 32 billion nodes & relationships per Neo4j instance
- 64 billion properties per Neo4j instance
- Small footprint
  - Standalone server is ~65mb

If NOSQL stands for Not Only SQL,

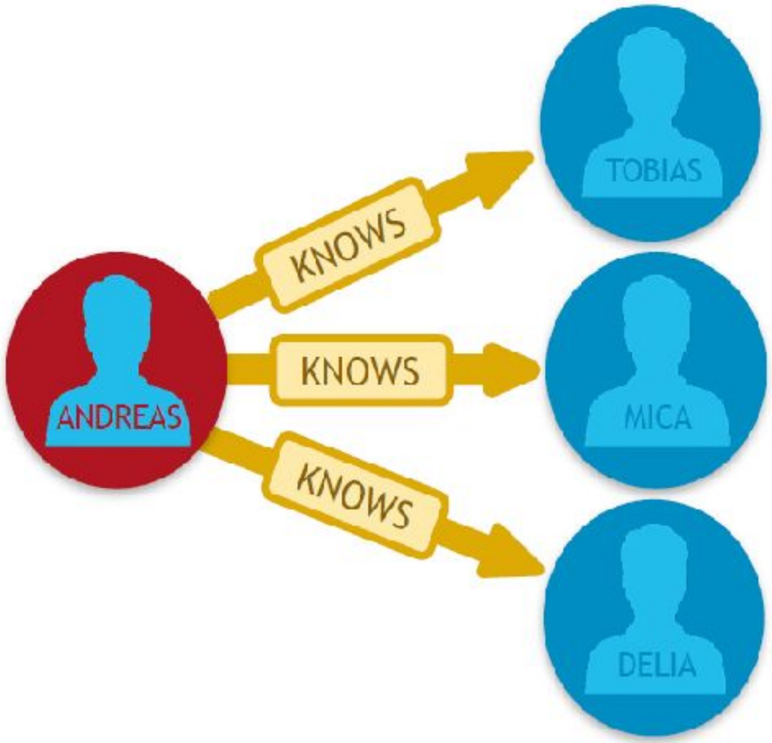
....then how do we execute queries?!

Relational Versus Graph Models

Relational Model



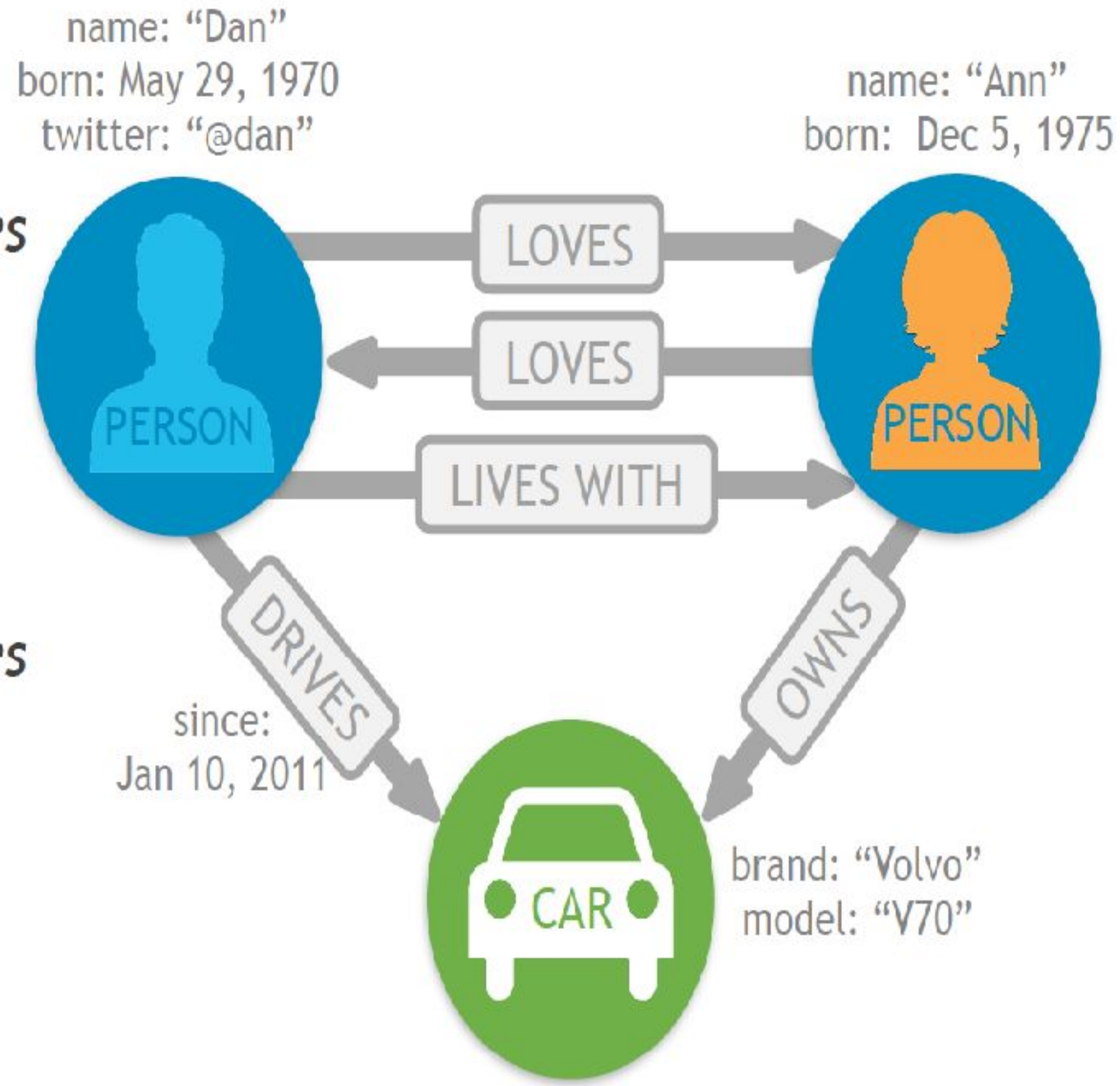
Graph Model



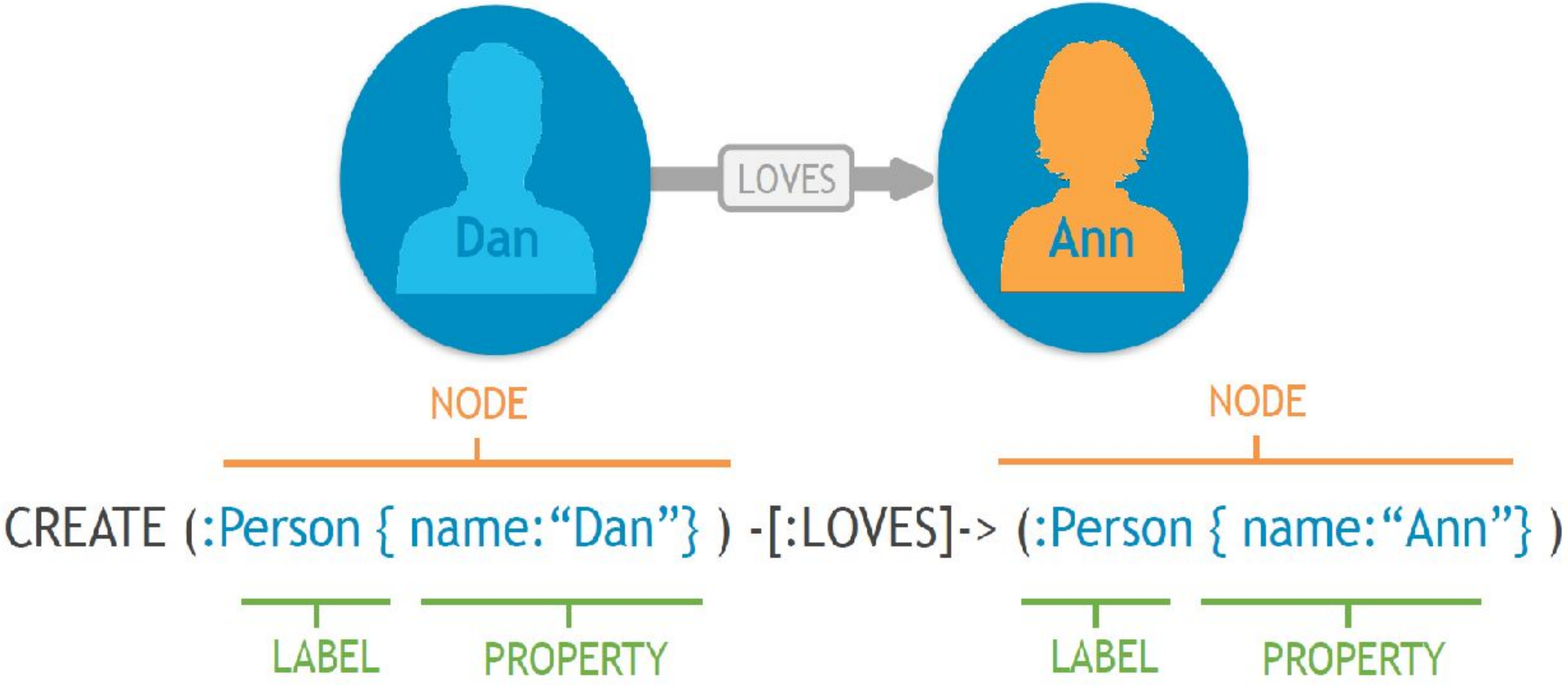
Property Graph Model Components

Nodes
- The objects in the graph
- Can have name-value *properties*
- Can be *labeled*

Relationships
- Relate nodes by type and direction
- Can have name-value *properties*



Cypher: Graph Query Language



Traversals!

Social Network Performance

MySQL vs Neo4j

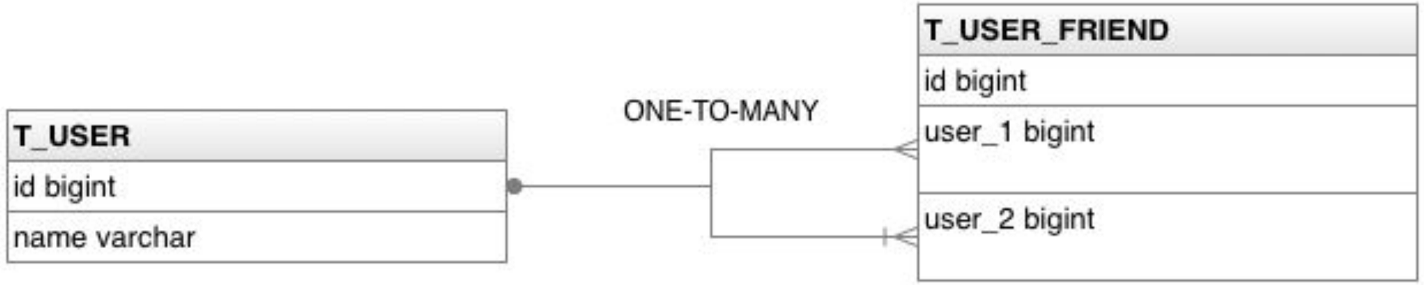
Social Network Performance

The Experiment: Round 1

- First rule of fight club:
  - Run a friends of friends query
- Second rule of fight club:
  - 1,000 Users
- Third rule of fight club:
  - Average of 50 friends per user
- Fourth rule of fight club:
  - Limit the depth of 5
- Fifth rule of fight club:
  - Intel i7 commodity laptop w/8GB RAM

Social Network Performance

RDBMS Schema



T_USER

id	name
1	John S
2	Kate H
3	Aleksa V
4	Jack T
5	Jonas P
5	Anne P

T_USER_FRIEND

id	user_1	user_2
1000	1	2
1001	3	5
1002	4	1
1003	6	2
1004	4	5
1005	1	4

Social Network Performance

SQL: Friends of friends at depth 3

```
select distinct uf3.*    from t_user_friend uf1
inner join t_user_friend uf2 on
uf1.user_1 = uf2.user_2
inner join t_user_friend uf3 on
uf2.user_1 = uf3.user_2
where uf1.user_1 = ?
```

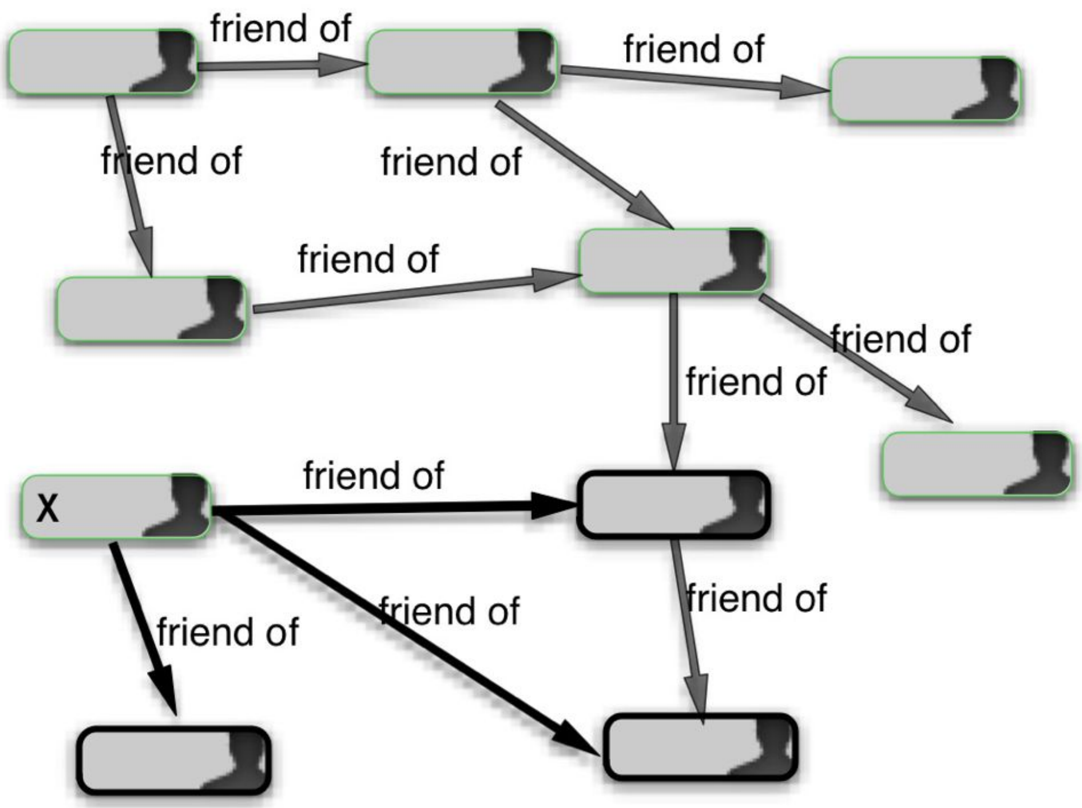
Social Network Performance

MySQL Results: Round 1- 1,000 Users

Depth	Execution Time (sec)	Records Returned
2	0.028	~900
3	0.213	~999
4	10.273	~999
5	92,613.150	~999

Social Network Performance

Social Graph



Social Network Performance

Neo4j Traversal API

```
TraversalDescription traversalDescription =
        Traversal.description()
          .relationships("IS_FRIEND_OF",Direction.OUTGOING)
          .evaluator(Evaluators.atDepth(2))
          .uniqueness(Uniqueness.NODE_GLOBAL);
    Iterable<Node> nodes = traversalDescription.traverse(nodeById).nodes();
```

Social Network Performance

Neo4j Results: Round 1- 1,000 Users

Depth	Execution Time (sec)	Records Returned
2	0.04	~900
3	0.06	~999
4	0.07	~999
5	0.07	~999

Social Network Performance

The Experiment: Round 2

- First rule of fight club:
  - Run a friends of friends query
- Second rule of fight club:
  - 1,000,000 Users
- Third rule of fight club:
  - Average of 50 friends per user
- Fourth rule of fight club:
  - Limit the depth of 5
- Fifth rule of fight club:
  - Intel i7 commodity laptop w/8GB RAM

Social Network Performance

MySQL Results: Round 1- 1,000,000 Users

Depth	Execution Time (sec)	Records Returned
2	0.016	~2,500
3	30.267	~125,000
4	1,543.505	~600,00
5	Did not finish after an hour	N/A

Social Network Performance

Neo4j Results: Round 1- 1,00,000 Users

Depth	Execution Time (sec)	Records Returned
2	0.010	~2,500
3	0.168	~110,000
4	1.359	~600,000
5	2.132	~800,000

Social Network Performance

Why is RDBMS performance horrible?

- To find all friends on depth 5,MySQL will create Cartesian product on t_user_friend table 5 times
  - Resulting in 50,000^5 records return
  - All except 1,000 are discarded
- Neo4j will simply traverse through the nodes in the database until there are no more nodes in which to traverse

Social Network Performance

The power of traversals

- Graphs data structures are localized
  - Count all of the people around you
  - Adding more people in the room may only slightly impact your performance to count your neighbors

Industry Leaders, Startup

- Google Knowledge Graph
  - Google Knowledge Vault
- Amazon Product Graph
- Facebook Graph API
- IBM Watson
- Microsoft Satori
  - Project Hanover/Literome
- LinkedIn Knowledge Graph
- Yandex Object Answer
- Diffbot, GraphIQ, Maana, ParseHub, Reactor Labs, SpazioDati

Questions?!

Influential Nodes

Categories

- Popular Nodes [7],
- Opinion Leaders [8],
- Topical Experts [9],
- Authoritative Actors [10, 11],
- Influence Spreader Or Disseminators [12].

Social Networks

- Social Network Data is a hybrid formation of user demographics, link information and the content generated by various users [1].

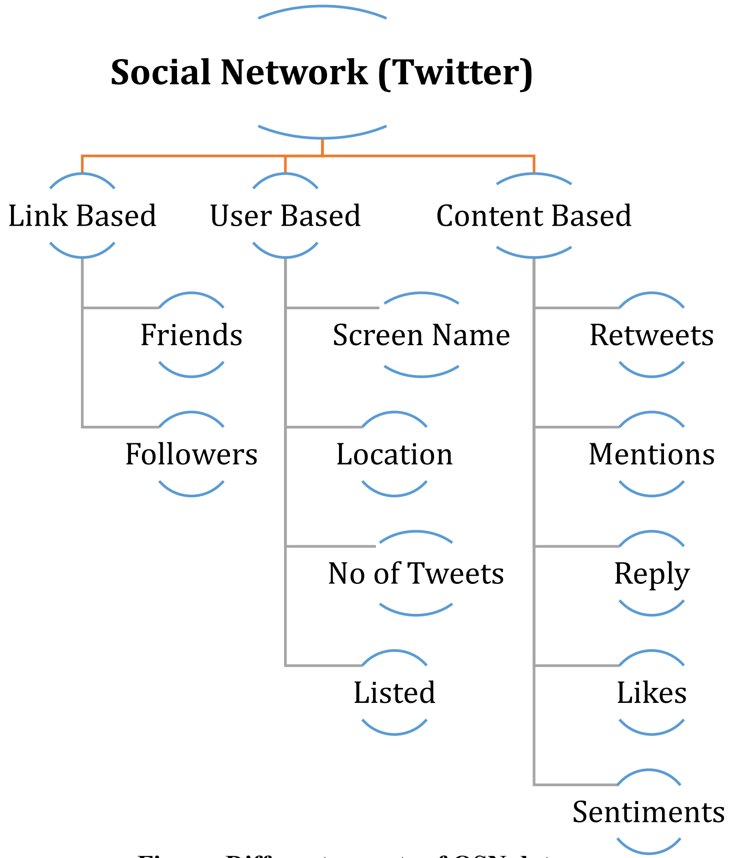


Figure. Different aspects of OSN data

121. Tidke, B., G Mehta, R., & Dhanani, J. (2018). Stakeholder-Centric Influence Analysis Approach Using Social Media for Smart City. *International Journal of Computational Intelligence & IoT*, *1*(2).

Challenges

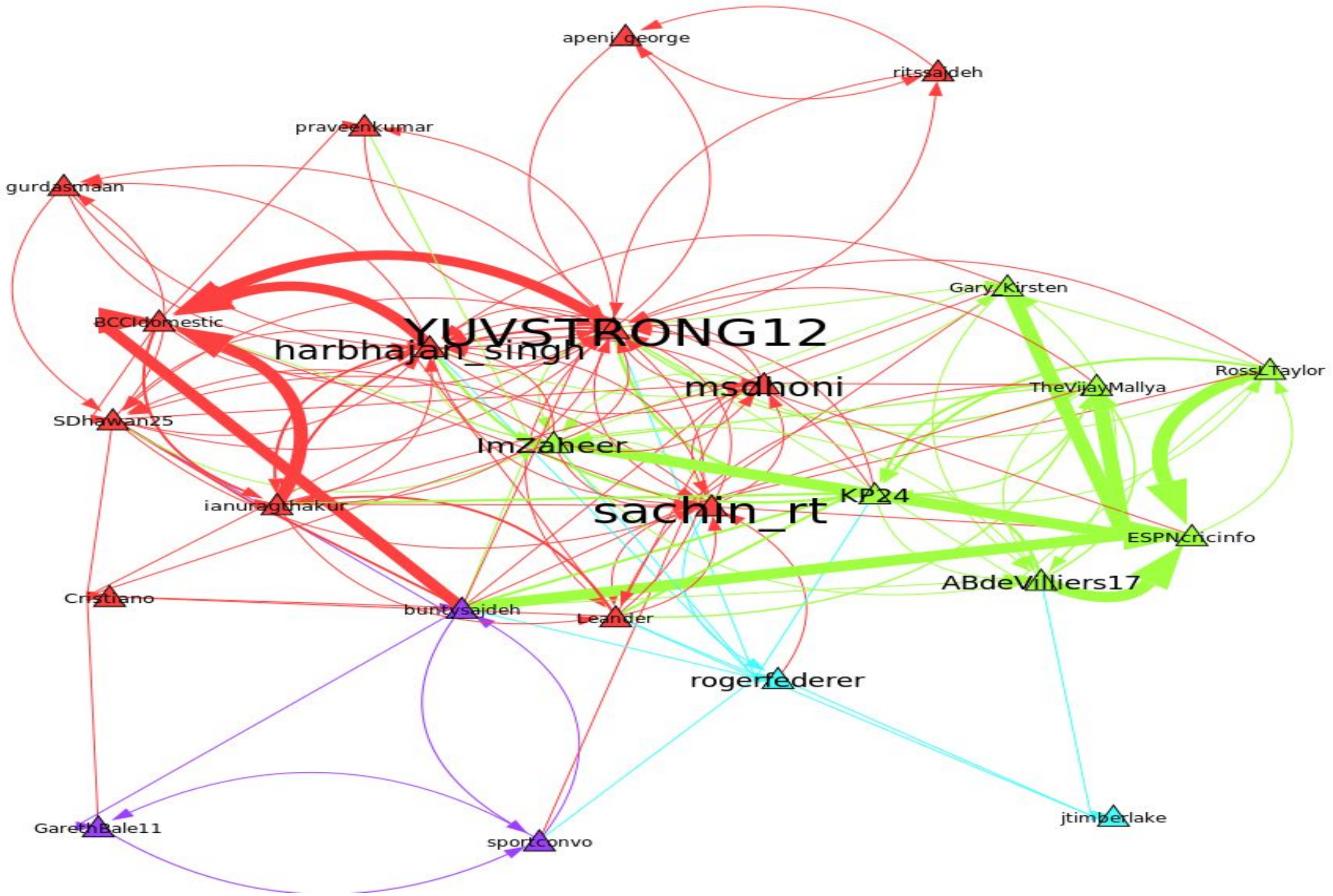
- Transformation of Raw Online Social Network (OSN) Data as Graph
- Computational Complexity
- Evolving and Heterogeneous OSN Data
- Topic Awareness for Influence Analysis
- Evaluation Metrics

- Tidke, B., Mehta, R., & Dhanani, J., "*Evolutionary and Heterogeneous Social Network Data: A Survey on State-of-the-Art, Technologies, Challenges and Future Research Directions*", ***World Wide Web,*** **Springer. (SCI) (Under Review) (Impact: 1.770)**

Influential Nodes

- Influential and Popular users from the Twitter based on topological network *(Online Advertisements & Brand Promotions)*
- Active Influential users with specific relevant background and have Opinion on the subject of matter *(Scholarly Literature)*
- Influential users or Topical Authoritative *(Politics and Economy)*
- Time Aware Influential users or Topical Authoritative and Sentiment Analysis *(Smart City)*

Popular Nodes



apeni_george
ritssajdeh
praveenkumar
gurdasmaan
BCCIdomestic
YUVSTRONG12
harbhajan_singh
Gary_Kirsten
RossLTaylor
TheVijayMallya
msdhoni
SDhawan25
ImZaheer
KP24
ianuragthakur
sachin_rt
ESPNcricinfo
ABdeVilliers17
Cristiano
buntysajdeh
Leander
rogerfederer
jtimberlake
GarethBale11
sportconvo

What is Cluster Analysis?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Clustering: Rich Applications and Multidisciplinary Efforts

- Pattern Recognition
- Spatial Data Analysis
  - Create thematic maps in GIS by clustering feature spaces
  - Detect spatial clusters or for other spatial mining tasks
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Weblog data to discover groups of similar access patterns

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

Major Clustering Approaches (I)

- Partitioning approach:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS
- Hierarchical approach:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, ROCK, CAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue

Major Clustering Approaches (II)

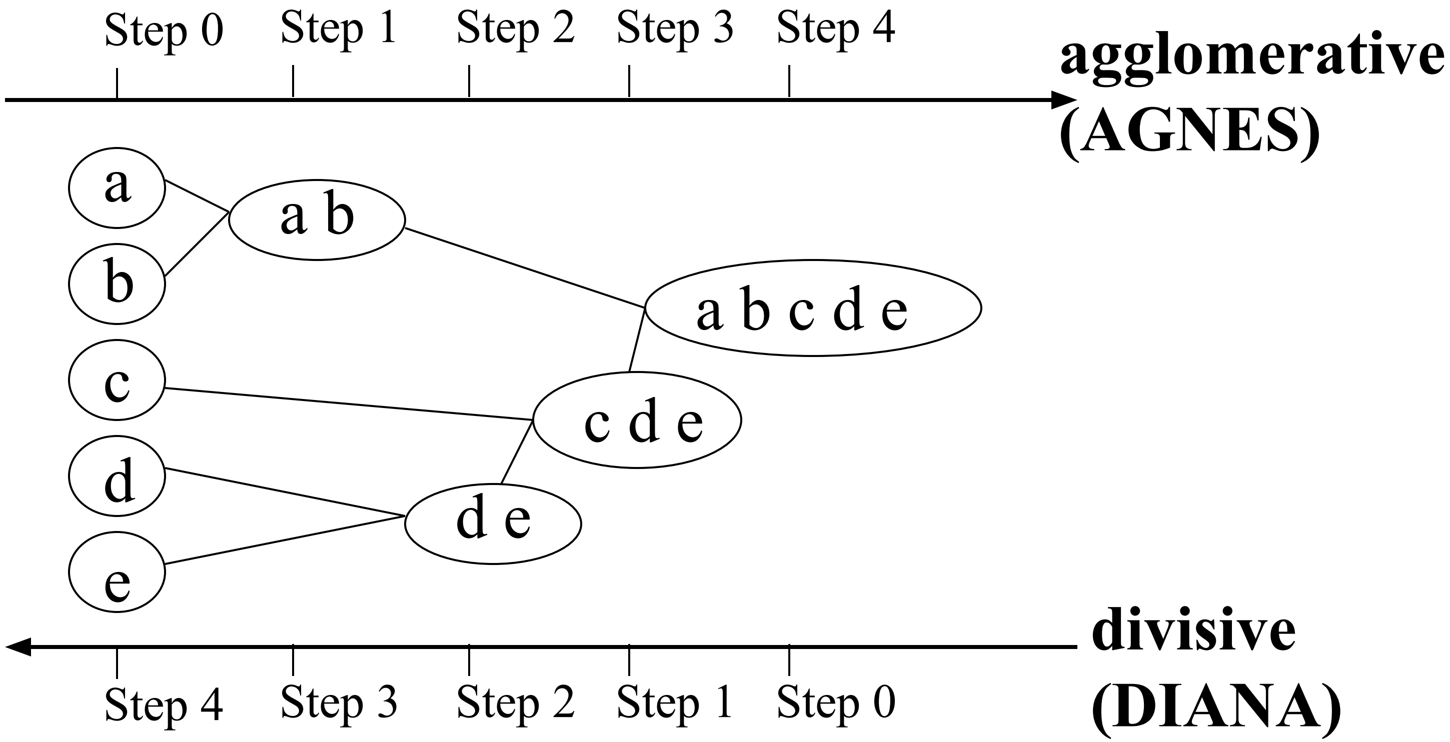
- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE
- Model-based:
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB
- Frequent pattern-based:
  - Based on the analysis of frequent patterns
  - Typical methods: pCluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering

Typical Alternatives to Calculate the Distance between Clusters

- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = avg(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e., $dis(K_i, K_j) = dis(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e., $dis(K_i, K_j) = dis(M_i, M_j)$
  - Medoid: one chosen, centrally located object in the cluster

Hierarchical Clustering

- Use distance matrix as clustering criteria. This method does not require the number of clusters ***k*** as an input, but needs a termination condition



Recent Hierarchical Clustering Methods

- Major weakness of agglomerative clustering methods
  - do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
  - can never undo what was done previously
- Integration of hierarchical with distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - ROCK (1999): clustering categorical data by neighbor and link analysis
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

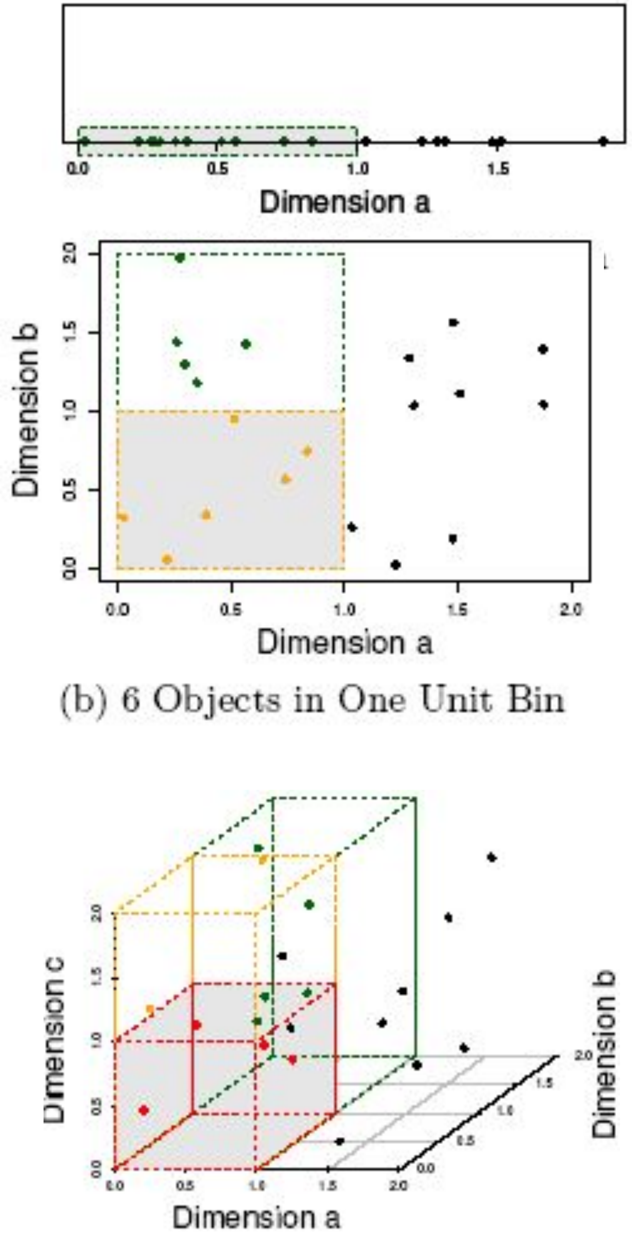
Clustering High-Dimensional Data

- Clustering high-dimensional data
  - Many applications: text documents, DNA micro-array data
  - Major challenges:
    - Many irrelevant dimensions may mask clusters
    - Distance measure becomes meaningless—due to equi-distance
    - Clusters may exist only in some subspaces
- Methods
  - Feature transformation: only effective if most dimensions are relevant
    - PCA & SVD useful only when features are highly correlated/redundant
  - Feature selection: wrapper or filter approaches
    - useful to find a subspace where the data have nice clusters
  - Subspace-clustering: find clusters in all the possible subspaces
    - CLIQUE, ProClus, and frequent pattern-based clustering

The Curse of Dimensionality

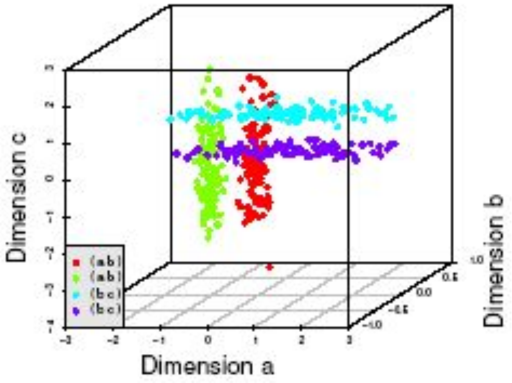
(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed
- Adding a dimension “stretch” the points across that dimension, making them further apart
- Adding more dimensions will make the points further apart—high dimensional data is extremely sparse
- Distance measure becomes meaningless—due to equi-distance



(b) 6 Objects in One Unit Bin

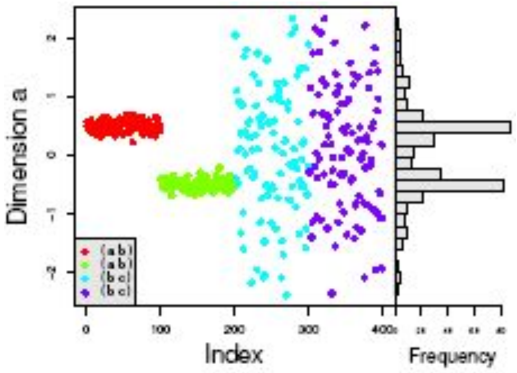
(c) 4 Objects in One Unit Bin



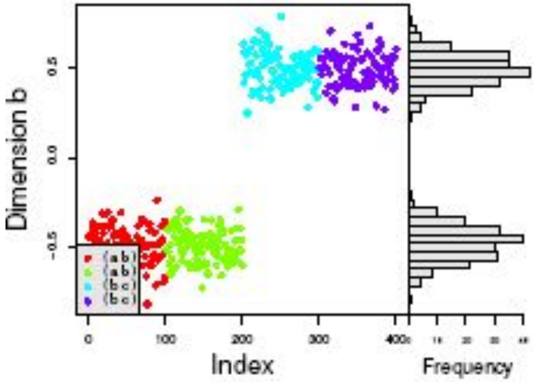
Why Subspace Clustering?

(adapted from Parsons et al. SIGKDD Explorations 2004)

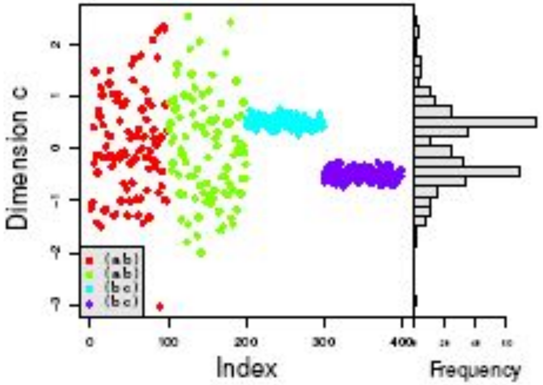
- Clusters may exist only in some subspaces
- Subspace-clustering: find clusters in all the subspaces



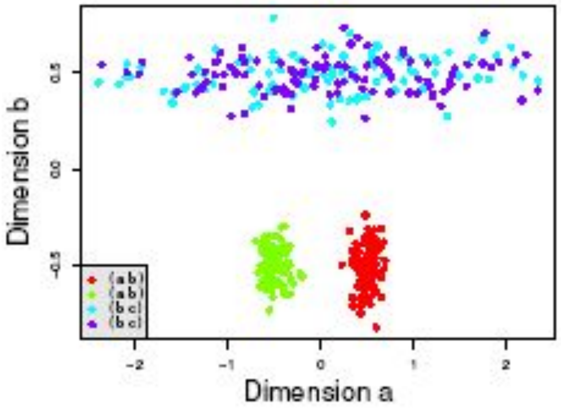
(a) Dimension a



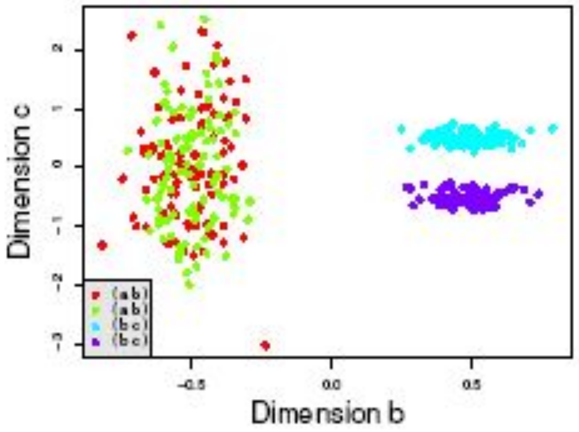
(b) Dimension b



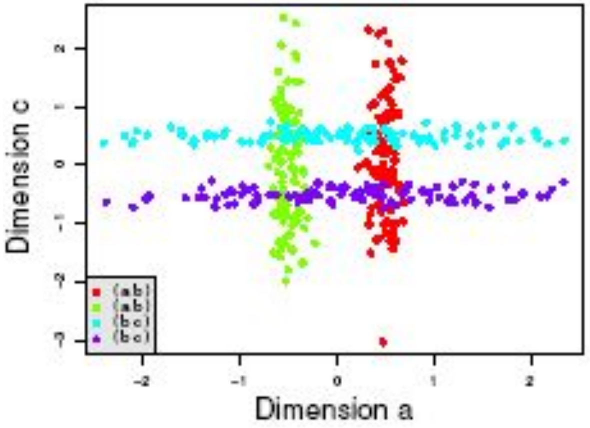
(c) Dimension c



(a) Dims a & b



(b) Dims b & c



(c) Dims a & c

Summary

- Cluster analysis groups objects based on their similarity and has wide applications
- Measure of similarity can be computed for various types of data
- Clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- Outlier detection and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches
- There are still lots of research issues on cluster analysis

Thank You