Big Data Praktikum

Abteilung Datenbanken

Wintersemester 2015/16

Orga

- **Ziel:** Entwurf und Realisierung einer Anwendung / eines Algorithmus unter Verwendung existierender Big Data Frameworks
- **Ablauf**
  - Anwesenheitspflicht der Gruppe zu allen Testaten
  - Bis Ende Oktober
    - Erstes Treffen mit Betreuer (Terminanfrage per Mail)
  - Ende November
    - Testat 1: System kennenlernen / Datenimport / Lösungsskizze
  - Anfang Februar
    - Testat 2: Implementierung und Ergebnisse vorstellen
  - Anfang März
    - Testat 3: Präsentation
    - 15 Minuten pro Gruppe
    - Anwesenheitspflicht aller Praktikumsteilnehmer

Technische Details

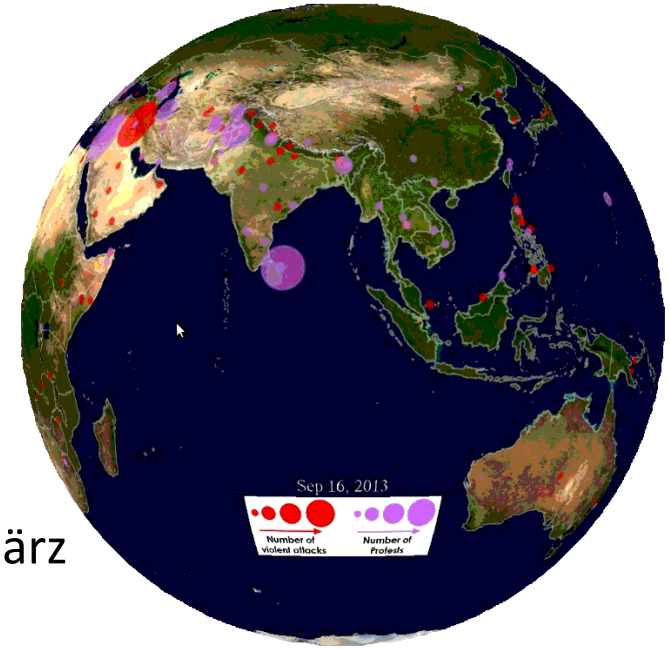
- Quellcode: GitHub Repository
  - Gruppe => Collaborators
  - Werden nach Praktikum zu *dbs-leipzig* geforked
- Java/Scala: Apache Maven 3 für Projekt Management
- Test Driven Development erwünscht
  - Siehe Dokumentation zu Unit Tests in jeweiligen Frameworks
- Quellcode Dokumentation zwingend erforderlich!
- Stabile Versionen verwenden (ggf. Rücksprache)
  - z.B. Flink 0.9.1 statt 0.10-SNAPSHOT
- Lokal lauffähige Lösungen können auf dediziertem Cluster ausgeführt werden
  - Terminabsprache **Ende Dezember** mit junghanns@informatik.uni-leipzig.de
- Datensätze https://github.com/caesar0301/awesome-public-datasets

Themenübersicht

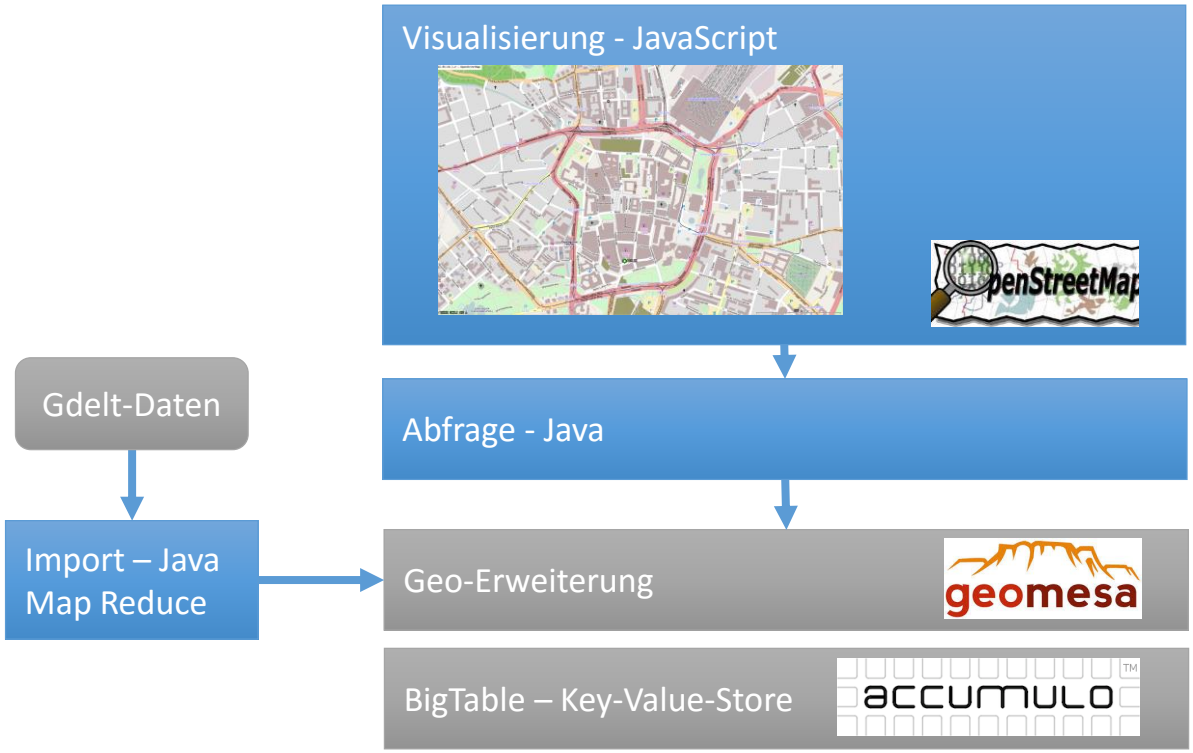
Thema	FW	#Studenten	Betreuer
Visualisierung von OSM-Daten	Geowave, Geomesa	2	Peukert
Tweet Analyse von News	Mahout	2	Christen
Holistic Ontology Matching	Gradoop, Flink	2	Christen
Analyse von Wetterdaten	Spark, Spark-R	2	Groß
Big OLAP: Datawarehouse	Kylin, Flink	3	Groß
Graph Metrics and Measures	Giraph, Flink, Spark	2	Junghanns
Graph Centrality Measures	Giraph, Flink, Spark	2	Junghanns
Random Walk With Restart	Giraph, Flink	2	Junghanns
Graph Summarization	Flink, Spark	2	Junghanns
Diffusion-based Graph Partitioning	Giraph, Flink	3	Junghanns
Frequent Subgraph Mining	Spark, Twill	3	Petermann

Speicherung und Visualisierung Geotemporaler Daten mit OpenStreetMap geomesa

- Geo-temporale Daten
  - Beispiele
    - Events(z.B. GDELT Project)
    - Fahrzeugbewegungen
    - Objektbewegungen (Vögel, Müll etc.)
  - Große Datenmengen
  - Ziel: effiziente Abfrage und Verarbeitung
    - z.B. Zeige alle Events von Januar bis März im
      Umkreis von Berlin



Ziel


Visualisierung - JavaScript
Gdelt-Daten
Abfrage - Java
Import – Java
Map Reduce
Geo-Erweiterung
geomesa
BigTable – Key-Value-Store
accumulo

Geo Mesa + Accumulo

- Apache Accumulo
  - Sortierter, verteilter key/value store (see Google BigTable)

Row ID	Family	Qualifier	Visibility	Timestamp	Value
derek	…	…	…	…	…
don	contact	email	admin | private	11905014	dminer@gopivotal.com
don	contact	email	admin | private	12412412	dminer@clearedgeit.com
don	contact	email	public	12412412	dm…@cl….com
don	contact	twitter	public	12423523	@donaldpminer
don	info	height	public	12314514	5' 9"
don	info	SSN	private	12314514	123-45-6789
erica	…	…	…	…	…

- GeoMesa
  - Effizienter Geo-Temporaler Index basierend auf Accumulo

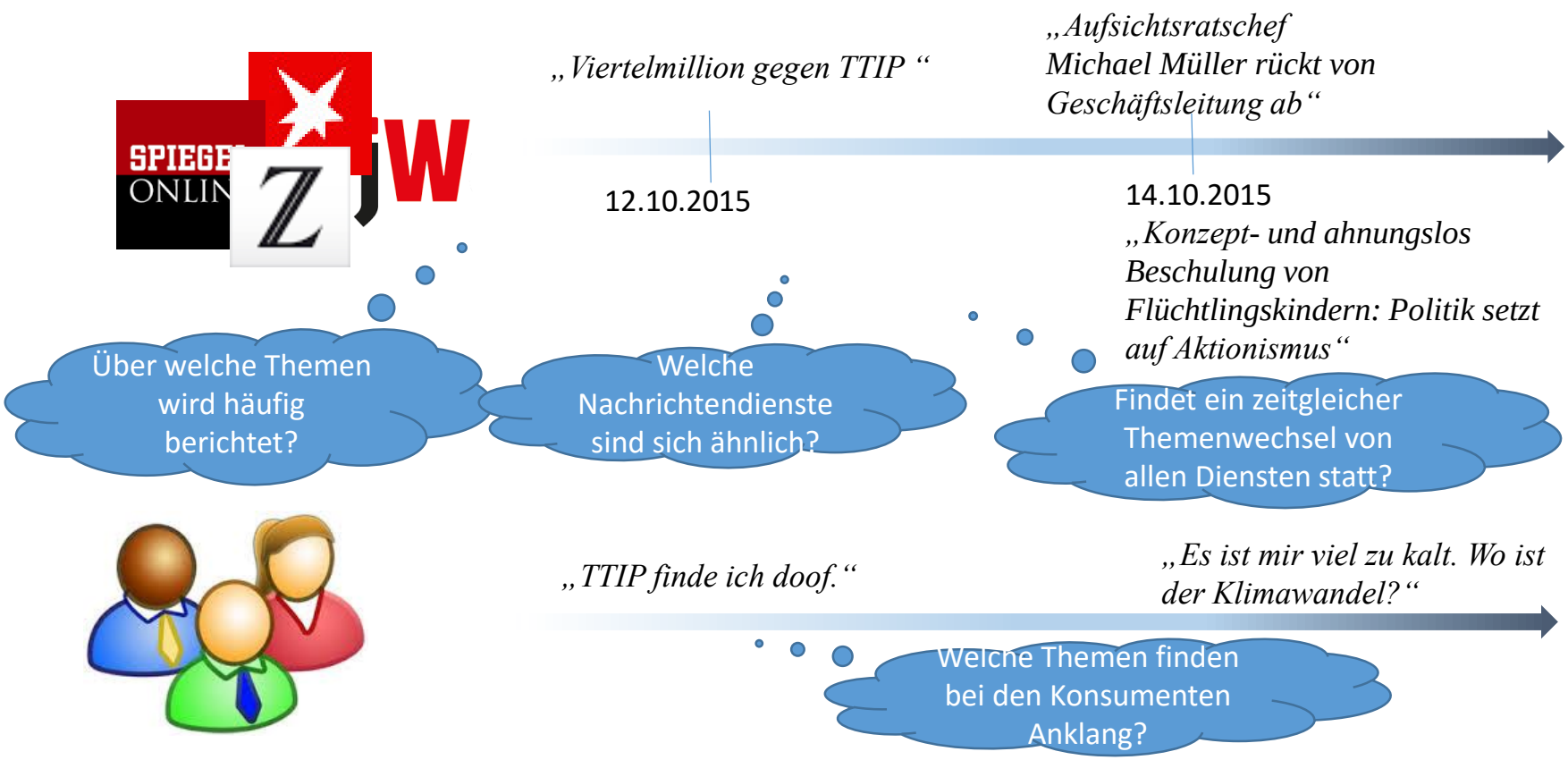
http://image.slidesharecdn.com/oreillyaccumulo-140805190637-phpapp01/95/an-introduction-to-accumulo-29-638.jpg?cb=1407265694

Open Streetmap/ OpenLayers

- JavaScript API for Open Streetmap


```
<!DOCTYPE HTML>
<title>OpenLayers Simplest Example</title>
<div id="demoMap" style="height:250px"></div>
<script src="OpenLayers.js"></script>
<script>
    map = new OpenLayers.Map("demoMap");
    map.addLayer(new OpenLayers.Layer.OSM());
    map.zoomToMaxExtent();
</script>
```

Tweet Analyse von News



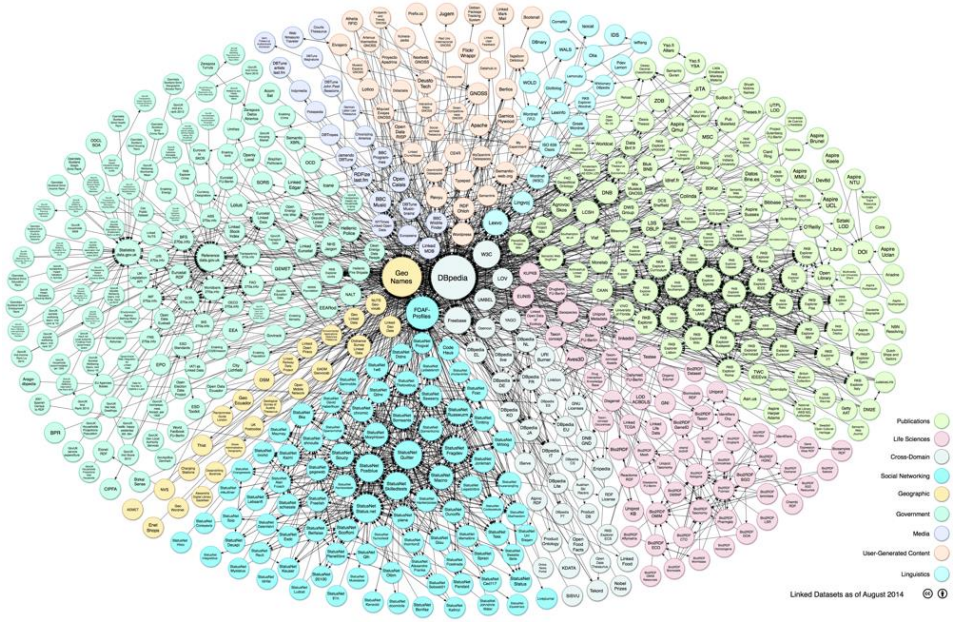
Workflow

1. Installation von Hadoop und Ausprobieren der mahout Beispiele
2. Speicherung der Tweets von Nachrichtendiensten sowie von Nutzern für Deutschland
3. Identifikation der Topics für definierte Zeitabschnitte
    1. Analysegeeignete Speicherung der Topic-Vektoren
4. Evaluierung der Hypothesen mittels Clustering und Präsentation
    1. Identifikation der Hot- Topics
    2. Identifikation ähnlicher Dienste → Dienste berichten über die selben Themen
    3. Identifikation der Themen, die die Community interessieren →Ähnlichkeit zwischen Topic-Vektoren der Community und der Dienste

Technologien und Hinweise

- Map-Reduce
- Verwendung der Streaming API oder der Rest API von Twitter
  - Eingrenzung auf Deutschland
- Topic Detection mittels der Mahout Bibliothek http://mahout.apache.org/

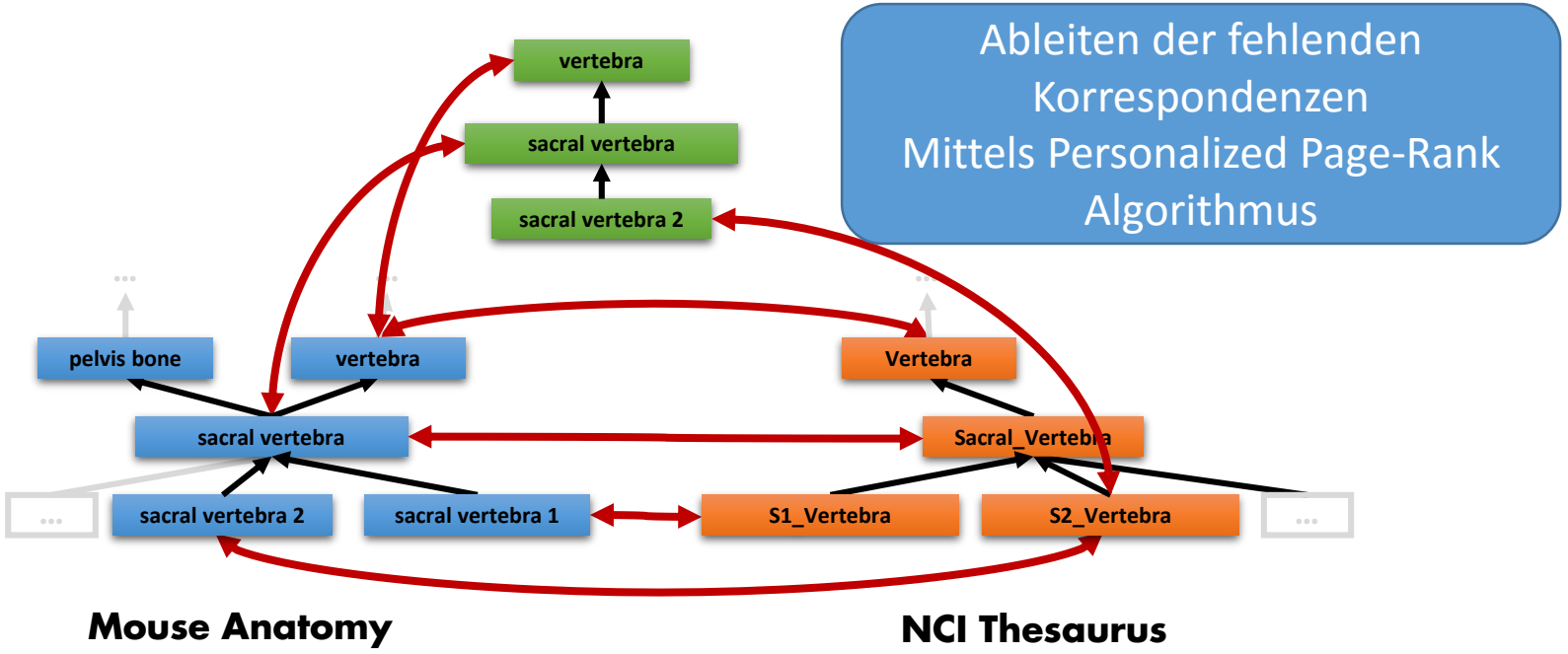
Holistic Ontology-Matching



- Beschreibung aller Entitäten im Semantic Web mittels Ontologien
- Integration von mehren Quellen erfordert Fusion der entsprechenden Ontologien
  - Komplexität der Berechnung eines Ontologie-Mappings zwischen n Ontologien n*(n-1)/2

→ ineffizient

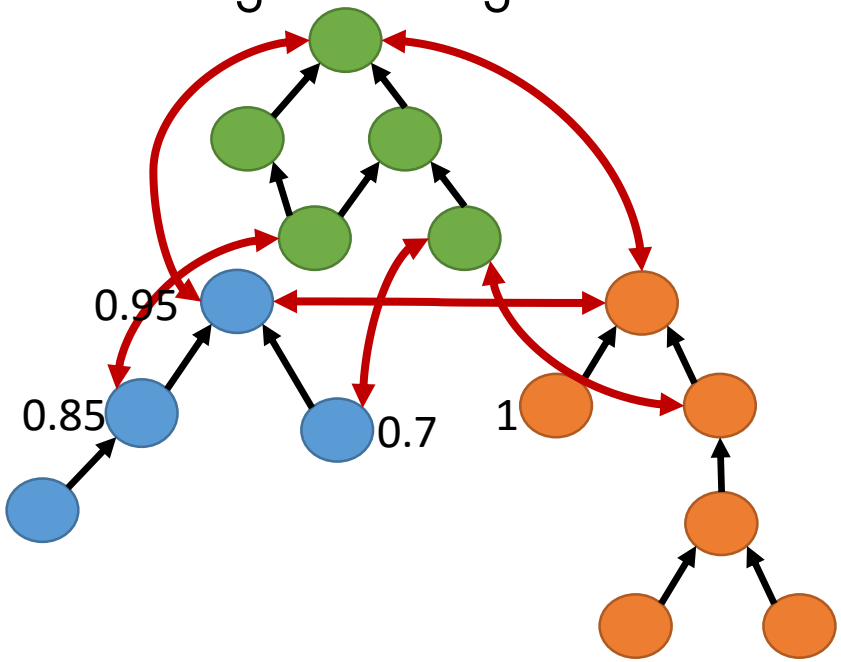
Holistic Ontology-Matching



Workflow

1. Installation von Hadoop/Flink
2. Existierende Ontologien und Mappings in eine Graphdatenstruktur überführen
3. Implementierung eines Verfahrens, das für ein Konzept fehlende Links zu den Konzepten der jeweils anderen Ontologien identifiziert

   z.B. Personalized Page-Rank Algorithmus



Technologien & Datensätze

Technologien

- Flink, Gelly
  - Graph-Processing

Datensätze

- 6 Ontologien im Bereich Lifescience (JSON-LD)
- Mappings zwischen den 6 Ontologien

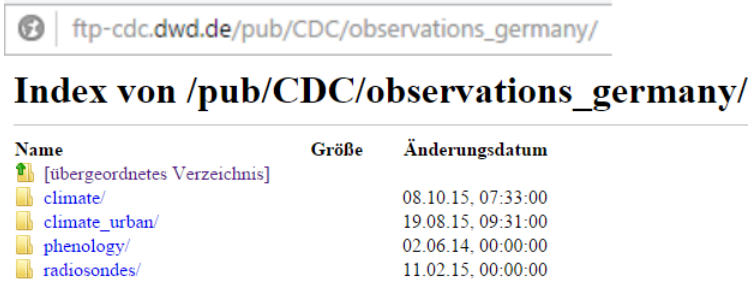
Analyse von Wetterdaten mit SparkR

- R: beliebte, freie Programmiersprache für statistisches Rechnen und Grafiken (Datenverarbeitung, maschinelles Lernen, …)
  - 1 Rechner, single-threaded
- SparkR: R package für Apache Spark
  - Usability von R + Skalierbarkeit von Spark
  - Nutzt Spark's *Distributed Computation Engine*
  - Alpha-Komponente in Apache Spark seit Release 1.4
- Zentrale Komponente: **SparkR DataFrame**
  - Verteiltes Data Frame implementiert auf Spark
  - Automatische Verteilung von Operationen auf SparkR DataFrames auf alle verfügbaren Cores und Maschinen im Cluster

http://www.dwd.de

Wetterstationsdaten

- Download via FTP Server des DWD
- Temperatur, Niederschlag, Luftdruck, Wind, …



Stations_id	von_datum	bis_datum	Hoehe	geoBreite	geoLaenge	Stationsname	Bundesland
2444	18240101	20151004	155	509.251	115.830	Jena (Sternwarte)	Thüringen
2932	19340101	20151004	131	514.348	122.396	Leipzig/Halle	Sachsen
1182	19470101	20010531	146	515.029	115.705	Eisleben-Helfta	Sachsen-Anhalt
4271	19470101	20151004	4	541.802	120.805	Rostock-Warnemünde	Mecklenburg-Vorpommern
433	19480101	20151004	48	524.675	134.021	Berlin-Tempelhof	Berlin
1270	19510101	20151004	316	509.829	109.608	Erfurt-Weimar	Thüringen
1957	19510101	20150322	93	515.138	119.499	Halle-Kröllwitz	Sachsen-Anhalt
1987							
3204							
…							

STATIONS_ID	MESS_DATUM	LUFTTEMPERATUR	SONNENSCHEINDAUER	REL_FEUCHTE	LUFTDRUCK_STATIONSHOEHE	WINDGESCHWINDIGKEIT
2932	2015-10-01	7.8	0	2.8	70.17	15.2
2932	2015-10-02	8.4	0	2.3	70.92	17.9
2932	2015-10-03	10.5	0	1.9	72.38	20.6
2932	2015-10-04	11.8	0	2.1	75.71	19.6
2932	2015-10-05	11.9	0	2.0	77.63	20.0

Bauern“regeln“?

- Ist der **Januar hell und weiß**, wird der **Sommer** sicher **heiß**.
- **Wie's Wetter** am **Siebenschläfertag**, so bleibt es **sieben Wochen danach**.
- **Hundstage hell und klar** deuten auf ein **gutes** Jahr, werden **Regen** sie bereiten, kommen **nicht** die **besten Zeiten**.
- Hat **Martini** einen **weißen Bart**, dann wird der **Winter lang** und **hart**.

Workflow: Umsetzung mit SparkR

- Import der DWD-Daten
- Analysen
  - Wettervergleich: Aggregation über mehrere Jahre/Orte ..
    - Temperatur, Niederschlag, …
  - Statistiken: kältester Winter, wärmster Ort, durchschnittlich kältestes Jahr, Vergleich versch. Regionen …
  - Testen der Bauernregeln (=Hypothesen) auf ihre Richtigkeit, z.B. durch
    - Erkennung von Mustern / Trends mit statistischen Tests
      - z.B. mit linearer Regression / Korrelation
- Präsentation der Ergebnisse (Plots etc.)

- Kenntnisse von R hilfreich
- Material
  - https://databricks.com/blog/2015/06/09/announcing-sparkr-r-on-spark.html
  - spark.apache.org/docs/latest/sparkr.html
  - Example: https://gist.github.com/shivaram/d0cd4aa5c4381edd6f85
  - Slideshare: http://de.slideshare.net/SparkSummit/07-venkataraman-sun
  - Slideshare: http://de.slideshare.net/SparkSummit/a-data-frame-abstchris-freeman

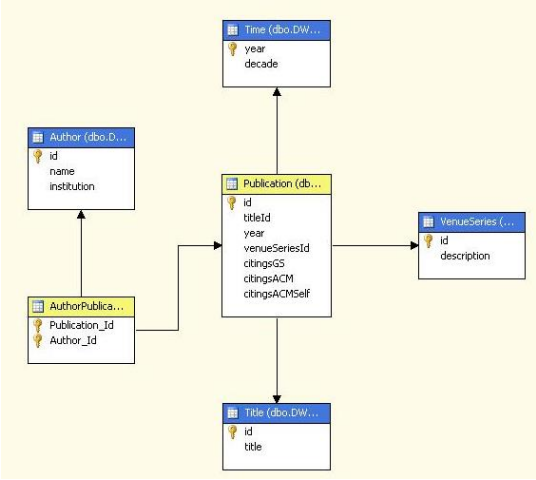
Big Data Praktikum

Big OLAP: Data Warehouse

Zitierungsanalyse

- Innerhalb wissenschaftlicher Arbeiten werden andere Arbeiten zitiert
- Anzahl der Zitierungen charakterisiert wissenschaftlichen Einfluss
  - Zeitlicher Verlauf der durchschnittlischen Zitierungszahlen einer Venue-Serie
  - Durchschnittliche Zitierungszahl einer Publikation pro Venue-Serie
  - Liste der zehn am häufigsten zitierten Institutionen
  - …

Zitierungsanalyse



Datensätze

- DBLP Bibliography
  - manuell gepflegte Website, die komplette Listen verschiedener Venues aus dem Informatik-Bereich enthält
- Relevante Teilmenge der Daten steht als CSV- und XML-Dateien zur Verfügung
- Kompletter Datensatz: http://dblp.uni-trier.de/xml/

Aufgaben

1. Installation Hadoop/Flink/Hive/HBase/Kylin
   - Linux VM einrichten oder Cloudera Quickstart VM 5.1
   - Analyse des DBLP Datensatzes
2. ETL (Flink, MapReduce, Hive)
   - Datennormalisierung: Normalisierung der Institutionsnamen
   - Import der XML-Daten, Transformation in Star Schema
   - Relationale Speicherung der Daten in Hive
3. Cube-Erstellung, OLAP (Kylin)
   - Anfragen analog DWH-Praktikum

Anforderungen

- Existierende Lösung für MS SQL Server (DWH Praktikum)
- Ziele:
  - Umsetzung unter Verwendung von Flink / Kylin
  - Beurteilung / Vergleich mit SQL Lösung
- Anforderungen
  - Erfolgreiche Teilnahme am DWH-Praktikum (mind. 1 Gruppenmitglied)
  - Erfolgreiche Teilnahme am Hadoop Testat (mind. 1 Gruppenmitglied)
  - SQL, Java und Linux Kenntnisse
- Material
  - http://www.heise.de/developer/artikel/Apache-Kylin-OLAP-im-Big-Data-Massstab-2824878.html
  - Cloudera Quickstart VM 5.1
- 2-3 Studenten

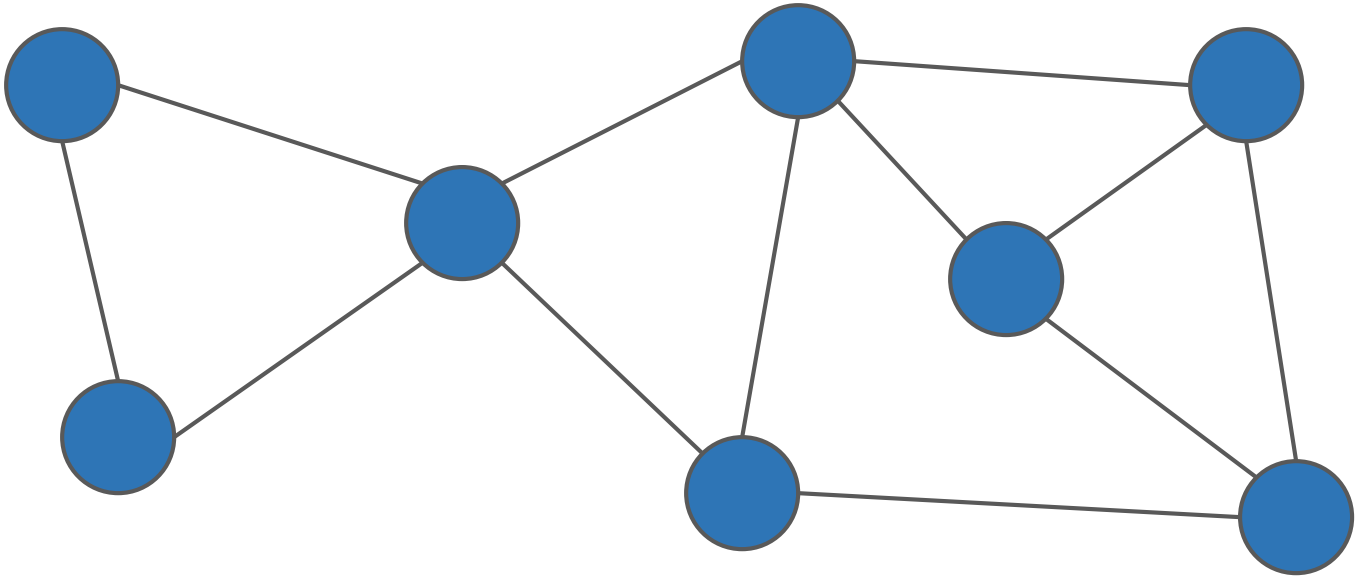
Big Data Praktikum

Graph Topics

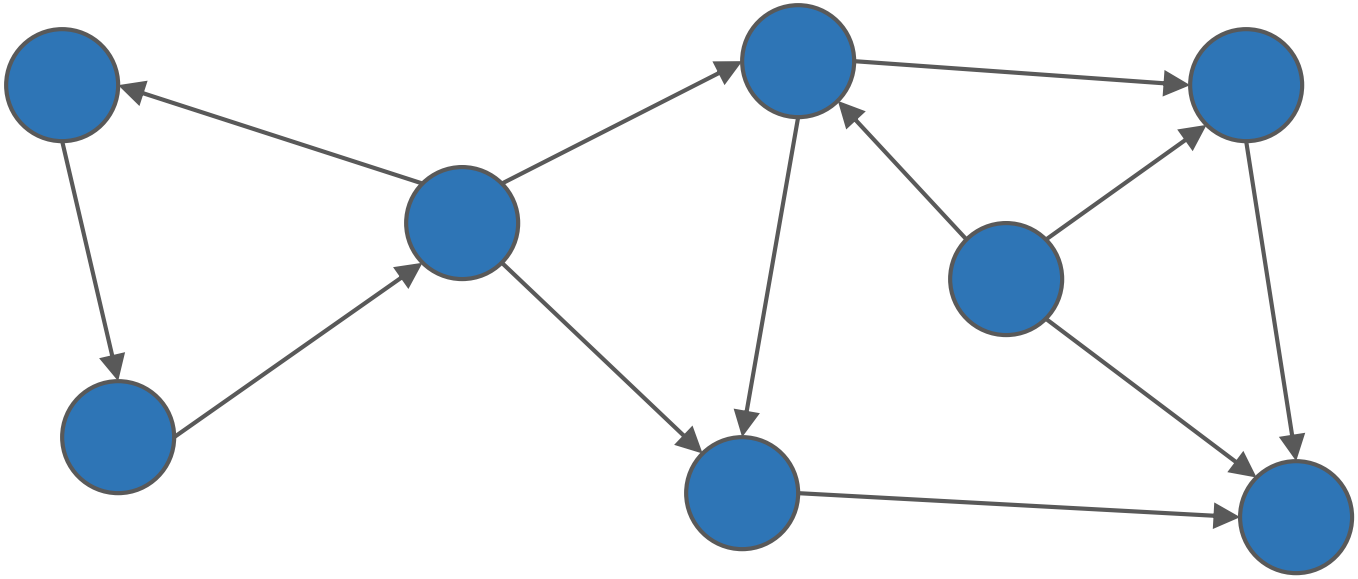
Graph Topics – General Info

- Requirements
  - Java / Scala / Git Skills
  - Interest in Algorithms and Datastructures
- Systems
  - Apache Giraph
  - Apache Flink / Gelly
  - Apache Spark / GraphX
  - Apache Twill
- Goals
  - Implementation (+ Testing / Validation)
  - Benchmarking on a Cluster (Linux!)
  - Visualization of results (gnuplot / Gephi / GraphViz / igraph / R / …)
- Datasets available

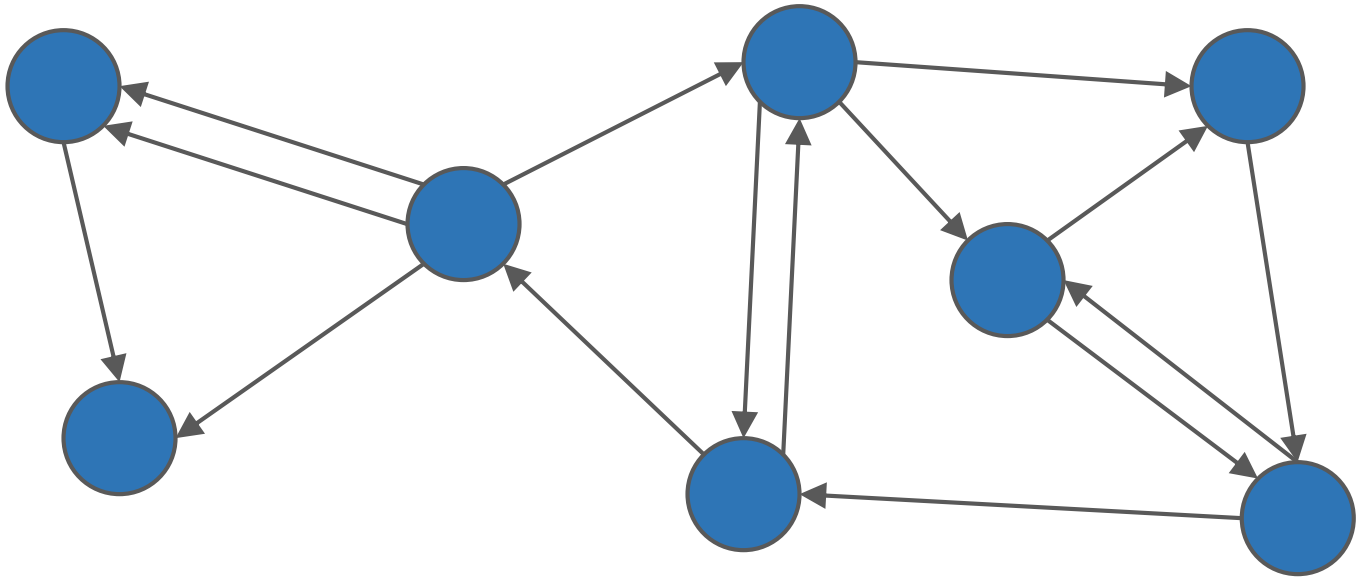
Undirected Graph



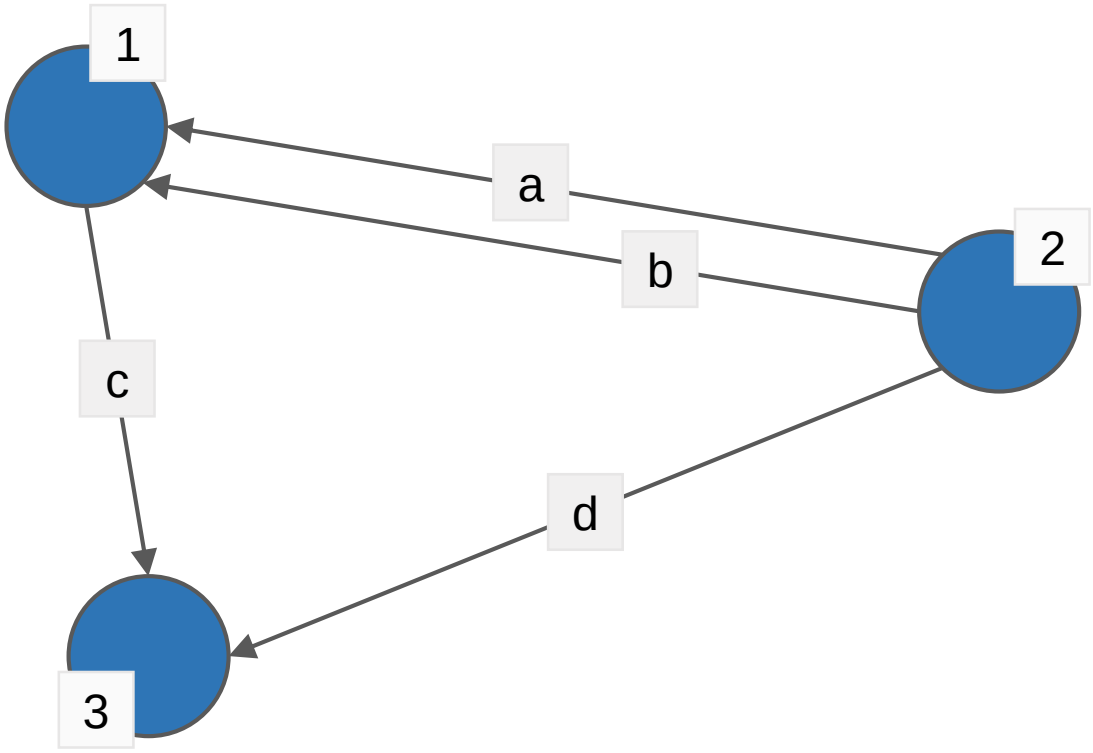
Directed Graph



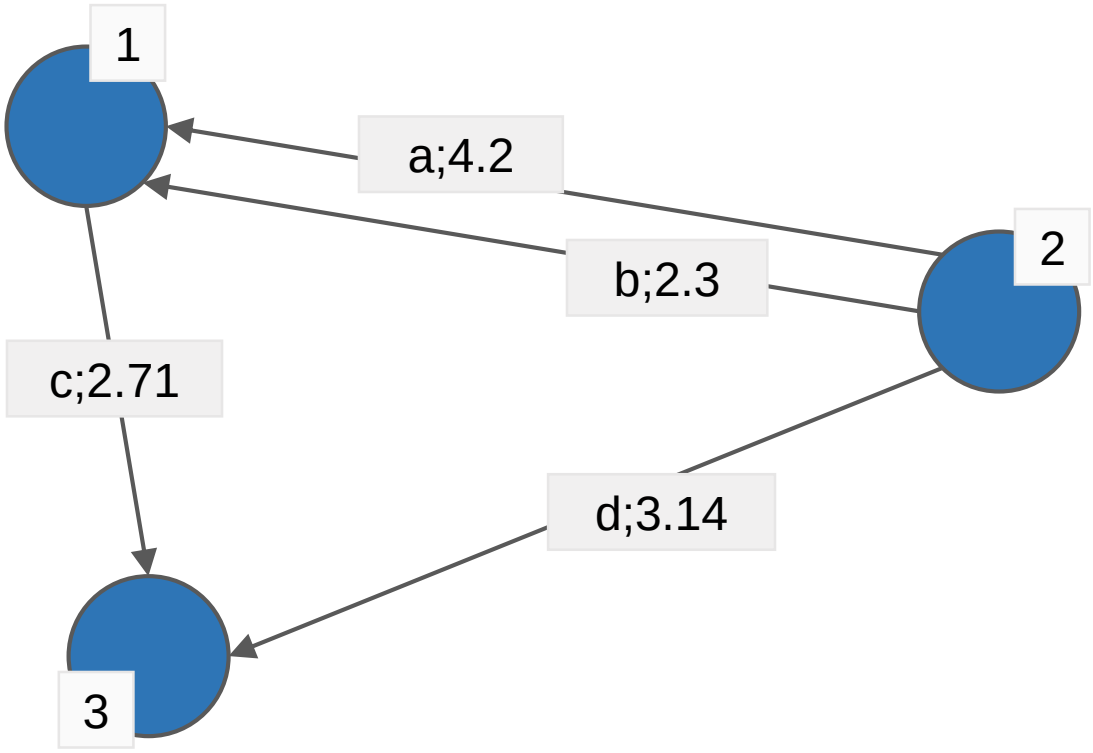
Directed Multigraph



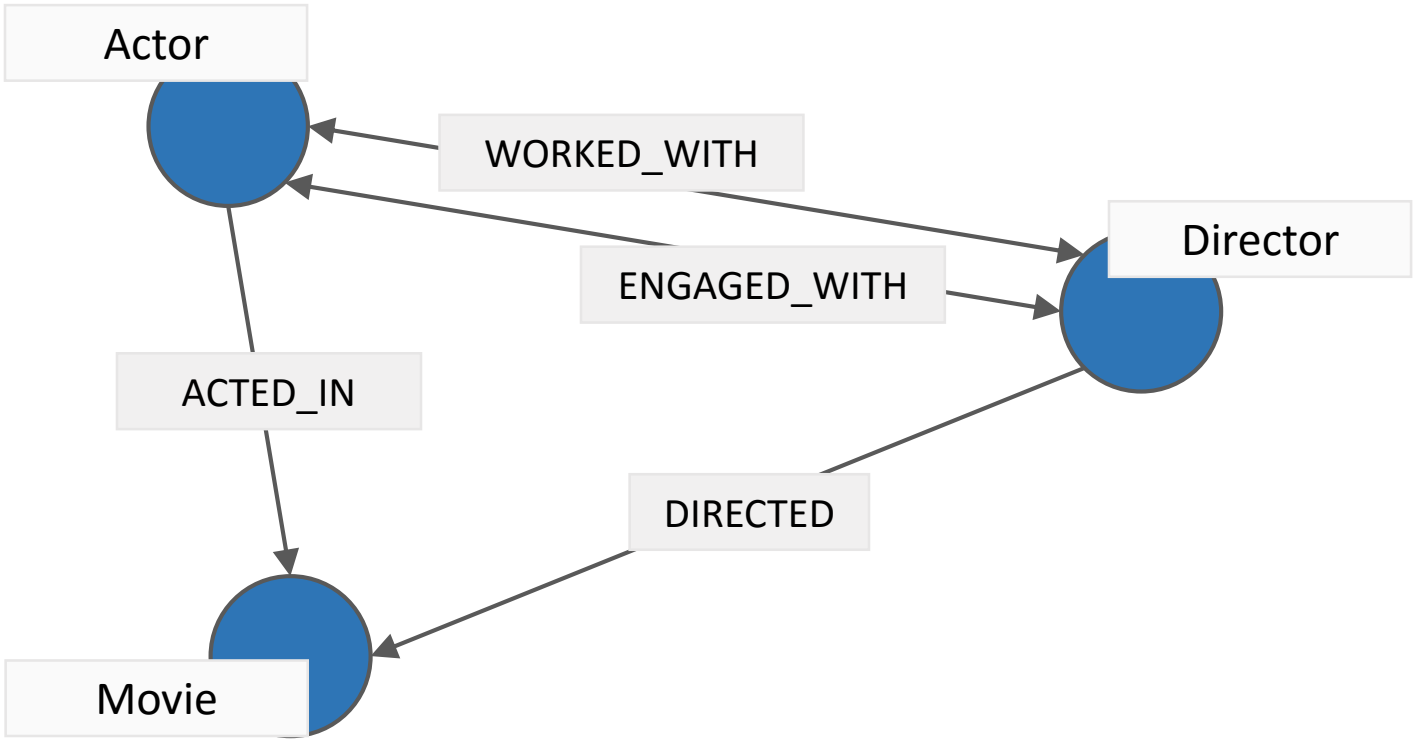
Directed Labeled Multigraph



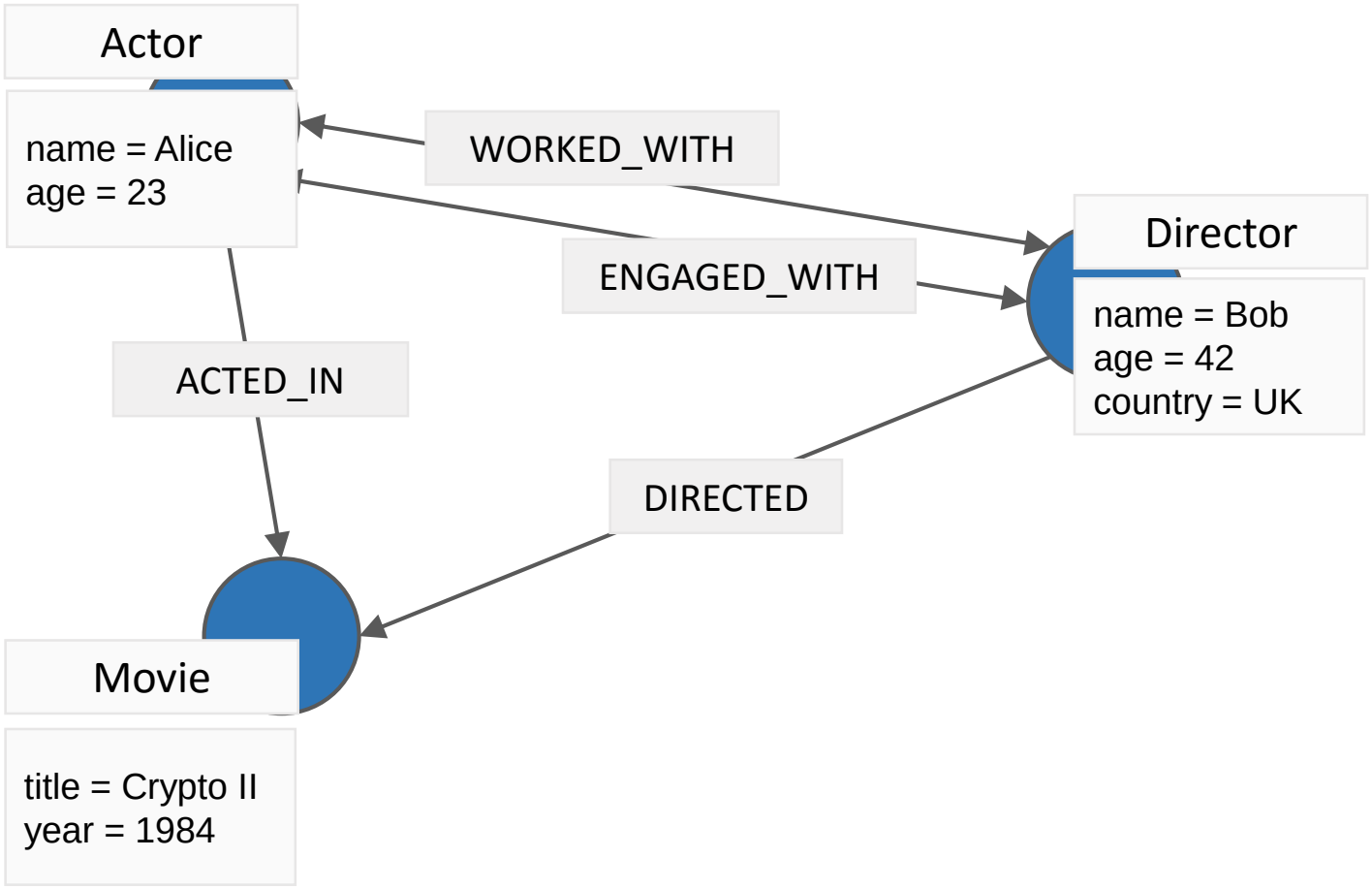
Directed Labeled Weighted Multigraph



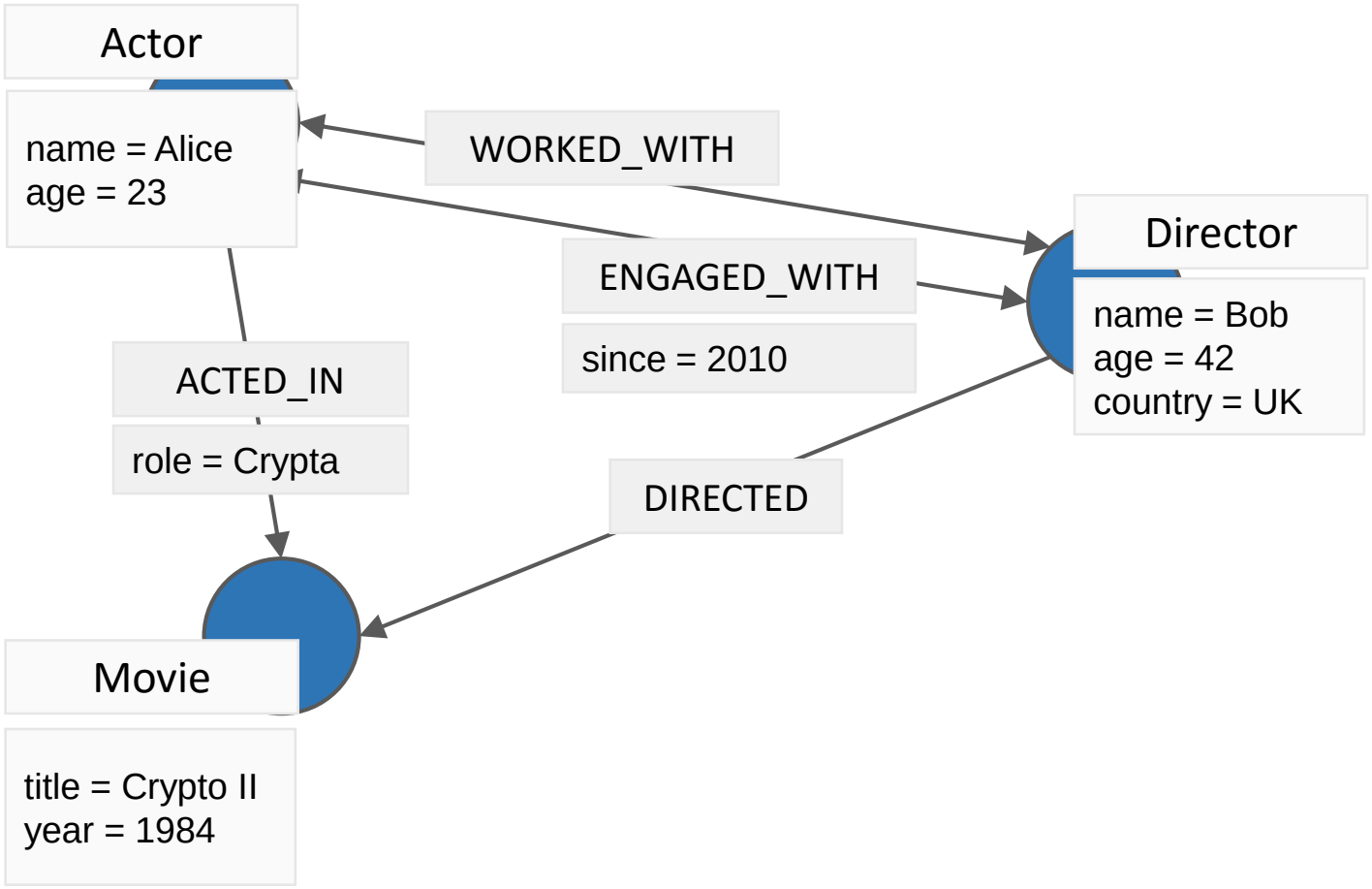
Directed Labeled Multigraph



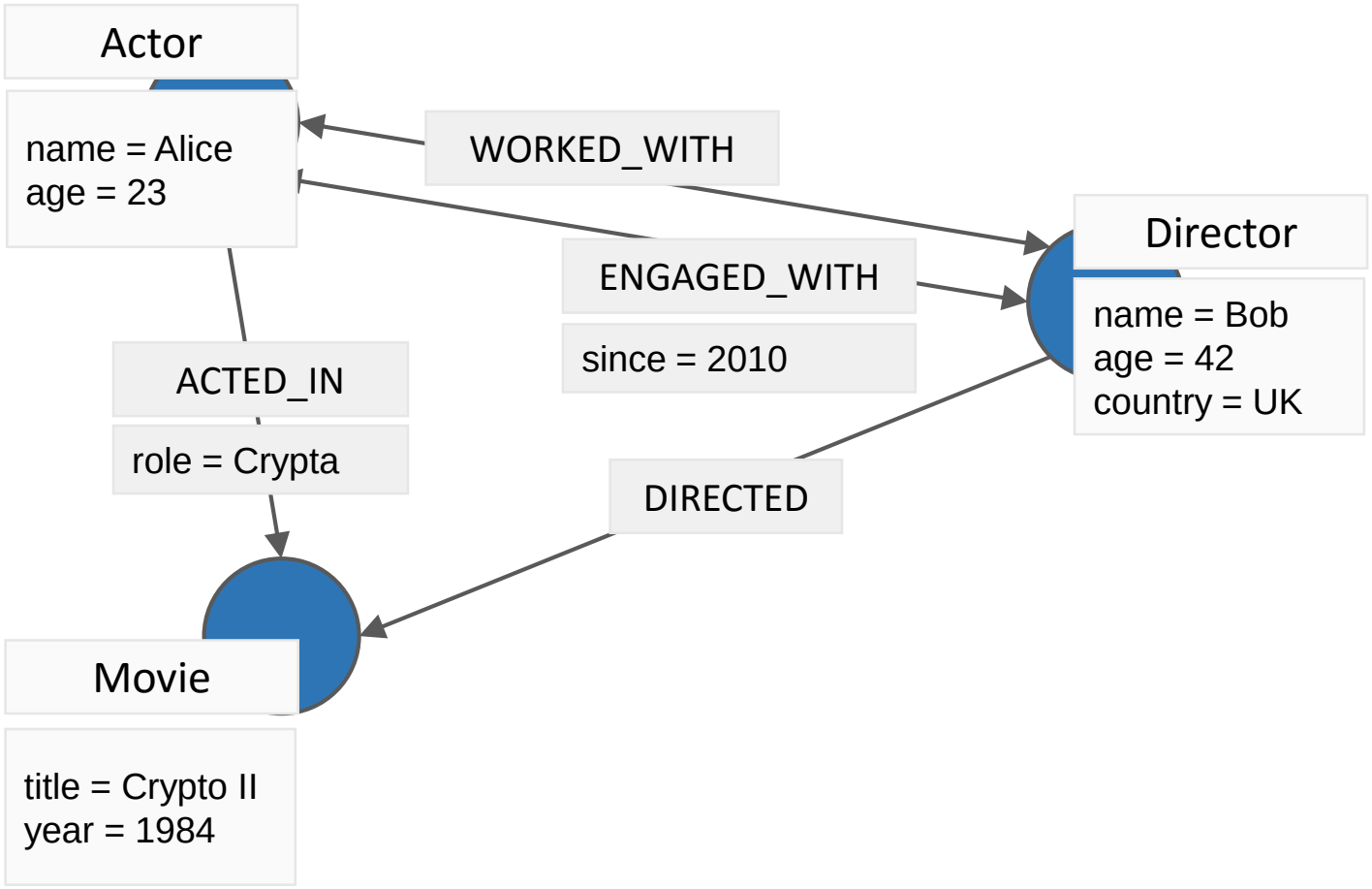
Directed Labeled Attributed Multigraph



Directed Labeled Attributed Multigraph



Property Graph Model



Task 1: Graph Measures and Metrics

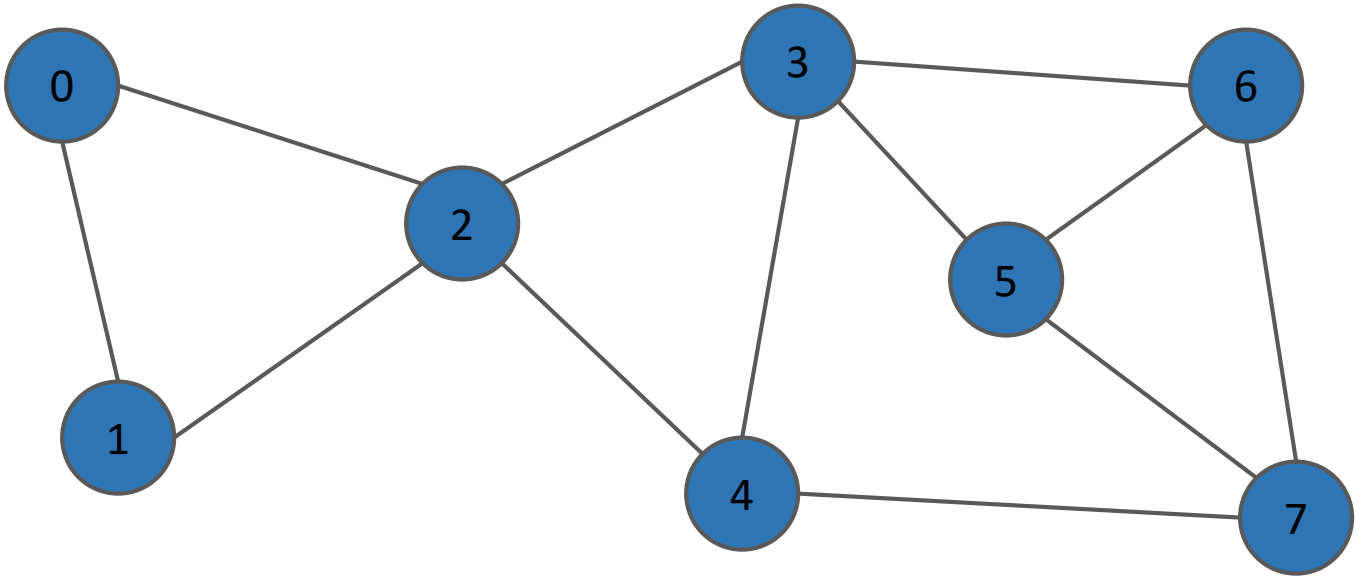
- Basic metrics
  - Vertex / Edge count
  - (In- / Out-) Degree (Distribution)
- Basic metrics in Labeled / Attributed Graphs
  - Label and Property Distribution
- Structural metrics
  - Distribution of Clustering Coefficient
  - Distribution of sizes of Weak / Strong connected components
  - Page Rank Distribution
  - Diameter (Longest shortest Path)
  - …

Task 1: Graph Measures and Metrics

- Goals
  - Algorithm Design
  - Implementation in Apache Flink, Giraph or Spark
  - Testing / Validation
  - Visualization (if necessary)
- Requirements
  - Knowledge in Java or Scala
  - Gnuplot / R
- 2 Students

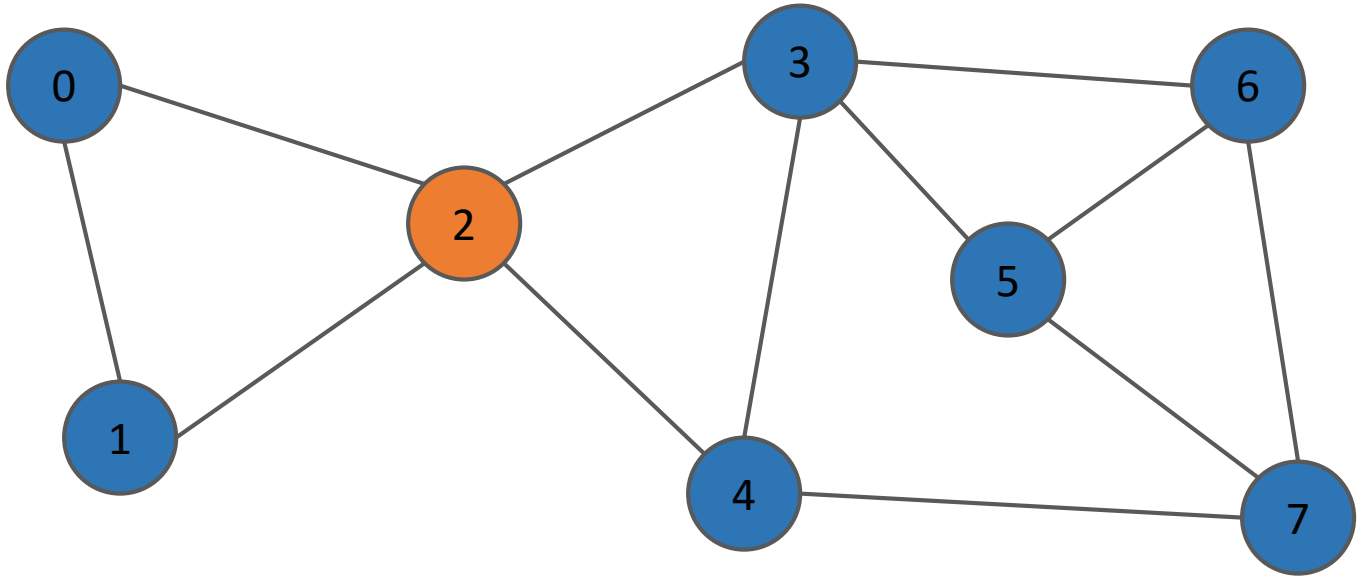
Task 2: Vertex Centrality Measures

- Betweenness centrality
  - How many times does a vertex occur in a shortest path between two other vertices?



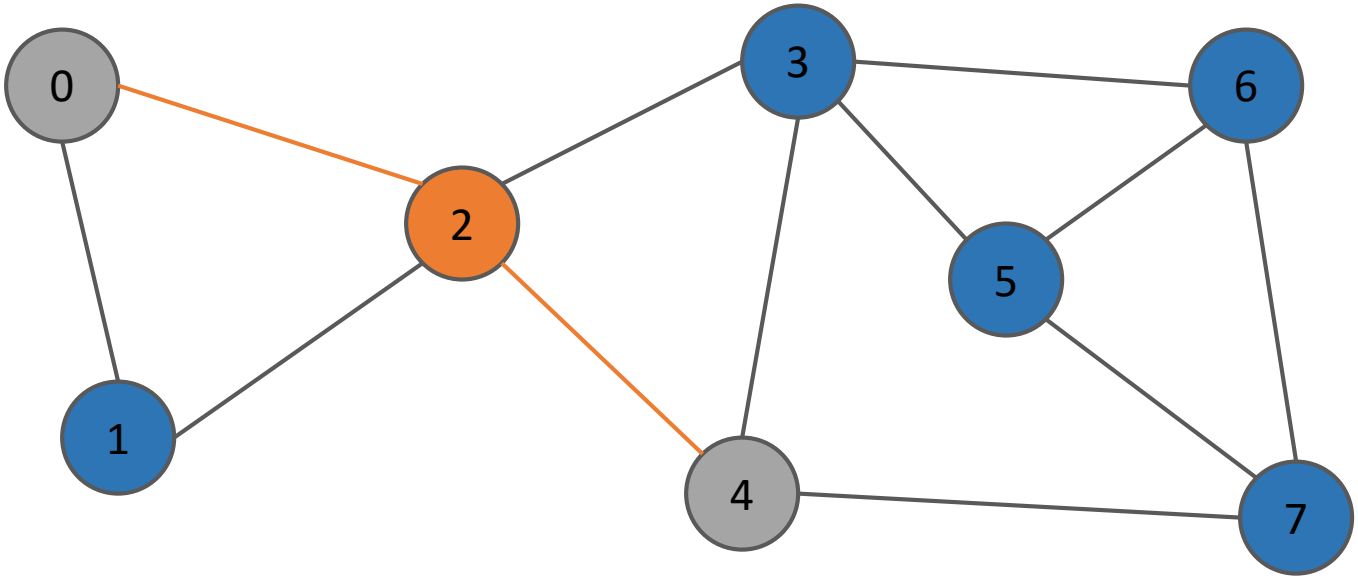
Task 2: Vertex Centrality Measures

- Betweenness centrality
  - How many times does a vertex occur in a shortest path between two other vertices?



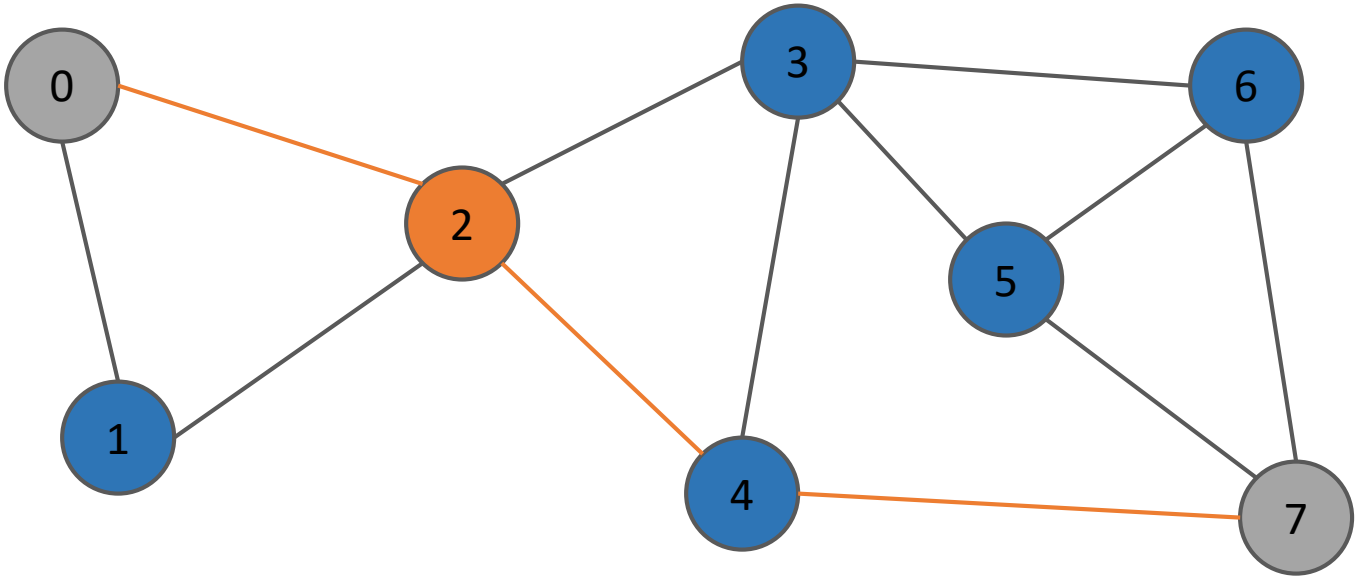
Task 2: Vertex Centrality Measures

- Betweenness centrality
  - How many times does a vertex occur in a shortest path between two other vertices?



Task 2: Vertex Centrality Measures

- Betweenness centrality
  - How many times does a vertex occur in a shortest path between two other vertices?



Task 2: Vertex Centrality Measures

- Farness / Closeness centrality
  - Given a distance matrix, how far / close is a vertex to all other vertices?

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	1	2	2	3	3	3
1	1	0	1	2	2	3	3	3
2	1	1	0	1	1	2	2	2
3	2	2	1	0	1	1	1	2
4	2	2	1	1	0	2	2	1
5	3	3	2	1	2	0	1	1
6	3	3	2	1	2	1	0	1
7	3	3	2	2	1	1	1	0

Task 2: Vertex Centrality Measures

- Farness / Closeness centrality
  - Given a distance matrix, how far / close is a vertex to all other vertices?

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	1	2	2	3	3	3
1	1	0	1	2	2	3	3	3
2	1	1	0	1	1	2	2	2
3	2	2	1	0	1	1	1	2
4	2	2	1	1	0	2	2	1
5	3	3	2	1	2	0	1	1
6	3	3	2	1	2	1	0	1
7	3	3	2	2	1	1	1	0
∑	**15**	**15**	**10**	**10**	**11**	**13**	**13**	**13**

Task 2: Vertex Centrality Measures

- Farness / Closeness centrality
  - Given a distance matrix, how far / close is a vertex to all other vertices?

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	1	2	2	3	3	3
1	1	0	1	2	2	3	3	3
2	1	1	0	1	1	2	2	2
3	2	2	1	0	1	1	1	2
4	2	2	1	1	0	2	2	1
5	3	3	2	1	2	0	1	1
6	3	3	2	1	2	1	0	1
7	3	3	2	2	1	1	1	0
1/∑	**.07**	**.07**	**.1**	**.1**	**.09**	**.08**	**.08**	**.08**

Task 2: Vertex Centrality Measures

- Goals
  - Algorithmic Understanding / Design
  - Implementation in Apache Flink, Spark or Giraph
  - Testing / Validation
  - Visualization (if necessary)
- Requirements
  - Profound Algorithmic knowledge
  - Profound Knowledge in Java
  - Experience with Apache Giraph / Graph Processing
- Material
  - *A Faster Algorithm for Betweenness Centrality*
  - Vorlesung KIT: http://parco.iti.kit.edu/henningm/GALA/GALA-slides-04.pdf
- 2 Students

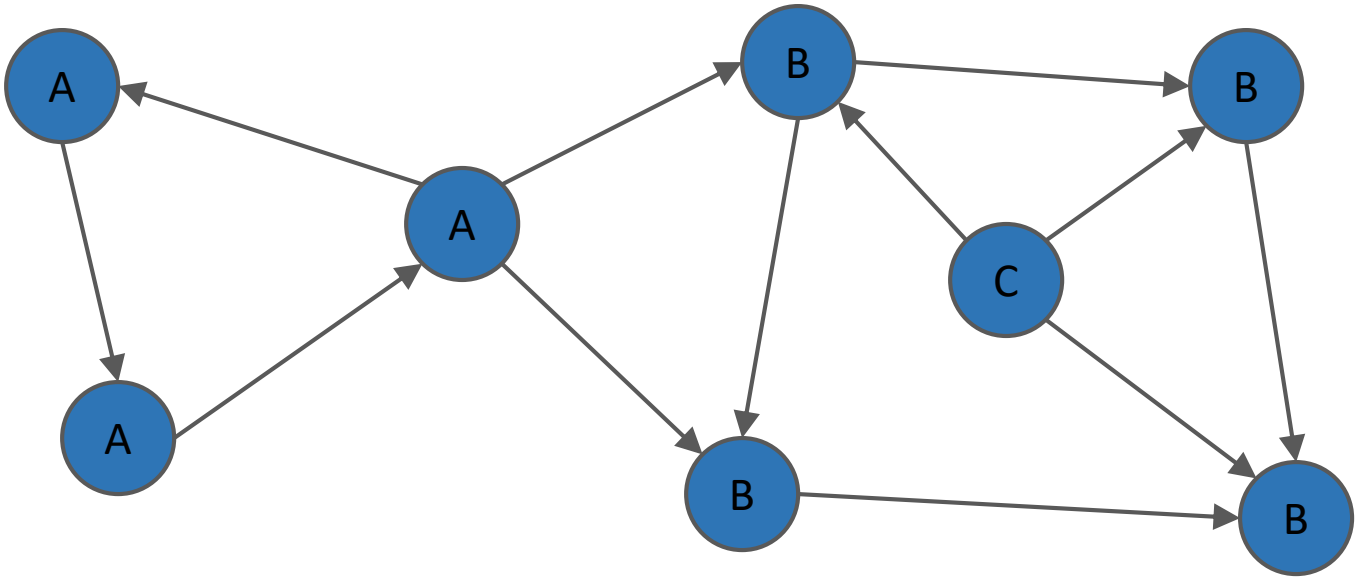
Task 3: Random Walk [with Restart]

- Traversal simulation:
  - Pick start vertex v0 at random
  - Pick next Vertex v1 from all Neighbours
  - Continue Walk from next Vertex v1
  - Restart: chance to go back to start vertex at any step

Task 3: Random Walk [with Restart]

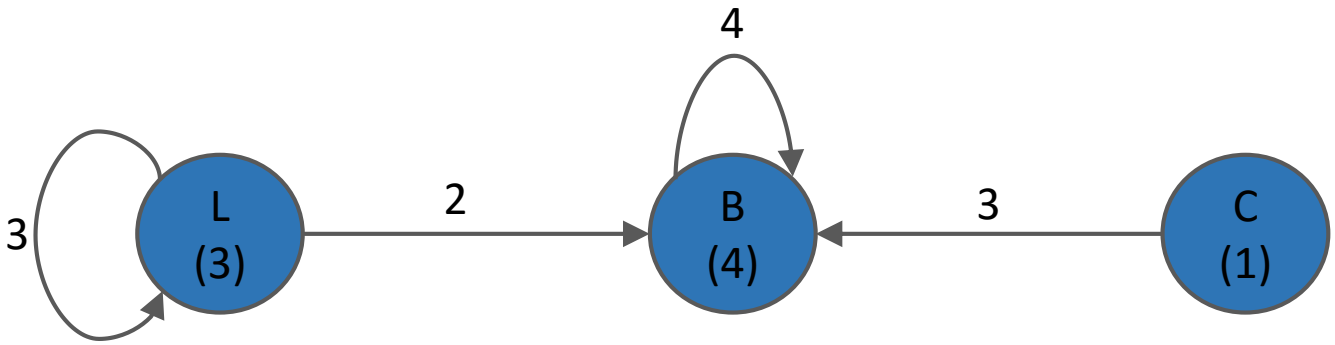
- Existing Implementations in Apache Giraph
- Goals:
  - Algorithmic Understanding / Design
  - Implementation in Apache Flink using Pregel API
  - Testing / Validation
  - Visualization (if necessary)
- Requirements
  - Profound Knowledge in Java
  - Experience with Apache Giraph / Graph Processing
- 2 Students

Task 4: Graph Summarization



- Problem
  - Input: Directed Labeled Graph
  - Output: Directed Labeled Graph
  - Summarize vertices and edges based on their label

Task 4: Graph Summarization

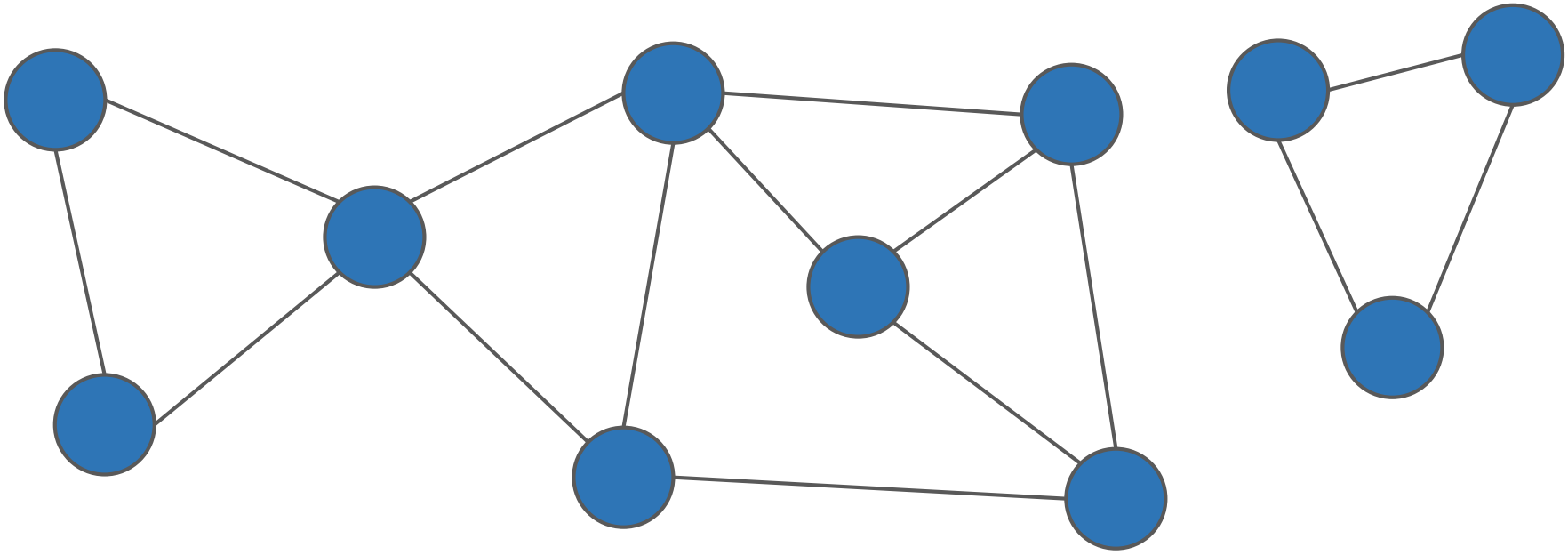


- Problem
  - Input: Directed Labeled Graph
  - Output: Directed Labeled Graph
  - Summarize vertices and edges based on their label

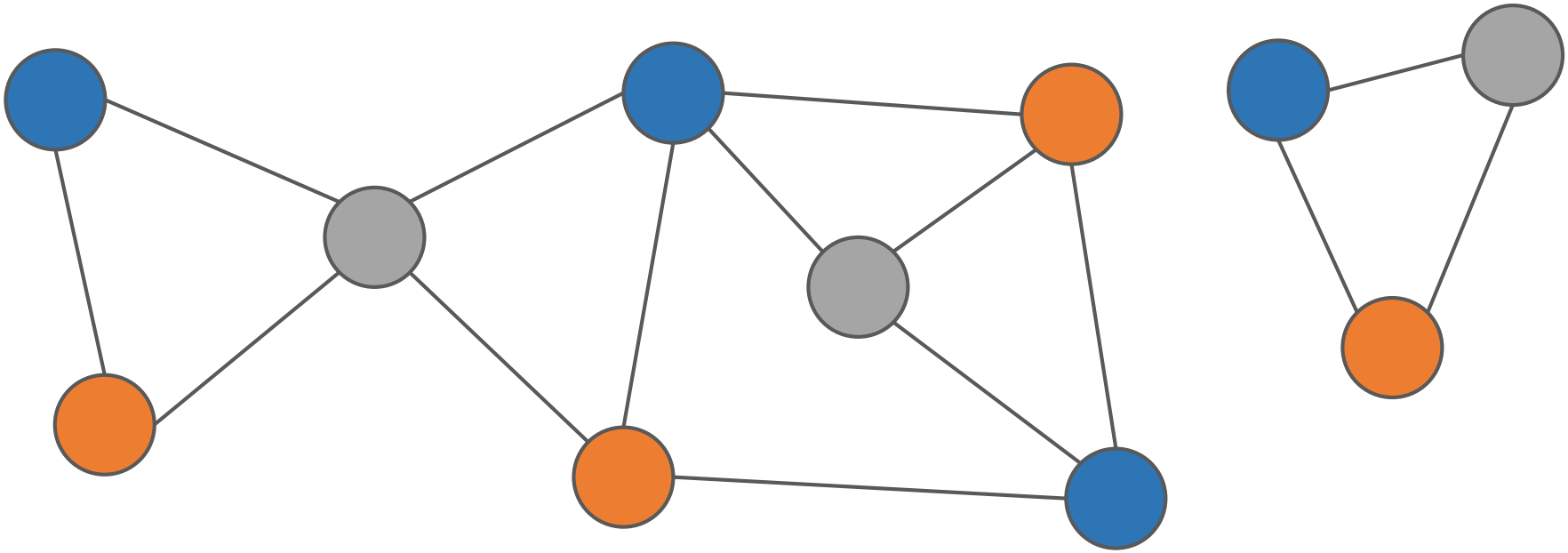
Task 4: Graph Summarization

- Existing Prototype implemented in Apache Flink
- Goal:
  - Algorithm Design
  - Implementation in Apache Spark / Apache Twill
  - Testing / Validation
  - Visualization (if necessary)
- Requirements
  - Knowledge in Java and/or Scala
  - Experience with Flink and/or Spark
- 2 Students

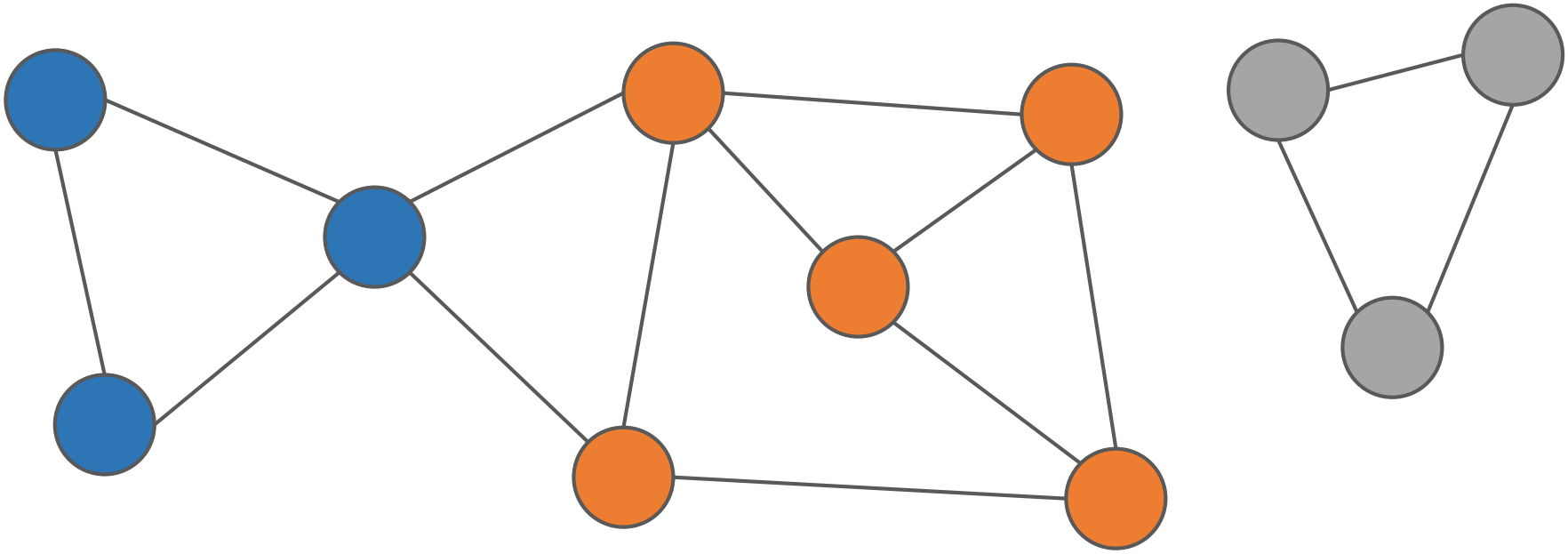
Task 5: Graph Partitioning using Diffusion



Task 5: Graph Partitioning using Diffusion



Task 5: Graph Partitioning using Diffusion



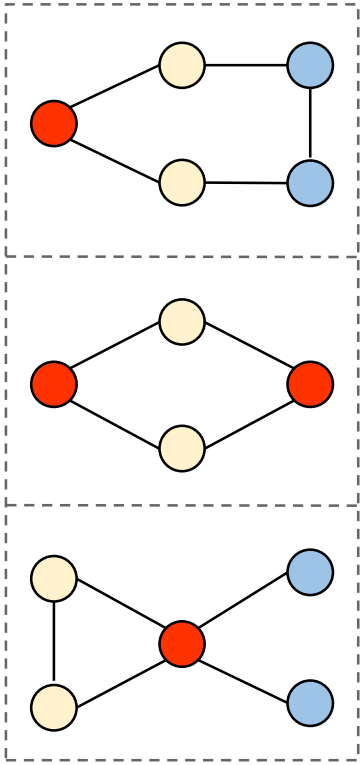
Task 5: Graph Partitioning using Diffusion

- DiDiC: Distributed Diffusive Clustering
- Heuristic Partitioning approach based on disturbed diffusion
- Initially used for load balancing in P2P systems
- Described in
  - *On Dynamic Graph Partitioning and Graph Clustering using Diffusion*
  - *A Distributed Diffusive Heuristic for Clustering a Virtual P2P Supercomputer*
  - *MA Thesis: Partitioning Graph Databases – A Quantitative Evaluation*

Task 5: Graph Partitioning using Diffusion

- Existing Prototype implemented in Neo4j
- Goal:
  - Algorithm Design
  - Implementation in Apache Flink or Apache Giraph
  - Testing / Validation
  - Visualization (if necessary)
- Requirements
  - Profound Knowledge in Java
  - Experience with Flink / Giraph
- 3 Students

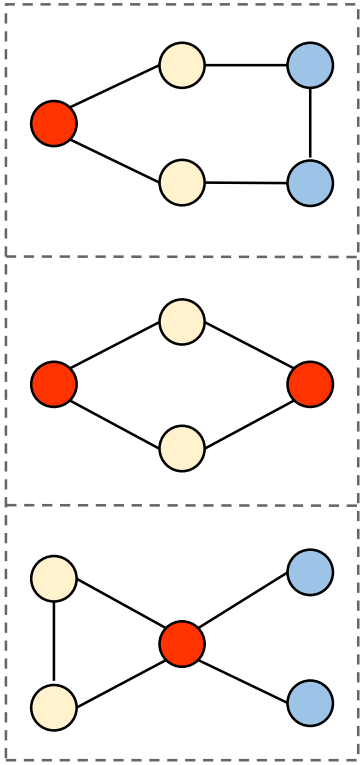
Task 6: Frequent Subgraph Mining



- Problem
  - Finding subgraphs occuring above a given threshold
- Graph Transaction Setting
  - Input is a collection of graphs
- Support Based Counting
  - A subgraph will be considered to be frequent, if a minimum number (threshold) of graphs contain it
- Pattern Growth Approach
  - Count support of n-edge subgraphs (start n=1), filter by threshold, grow them to n+1-edge subgraphs,
    repeat until all frequent ones are discovered

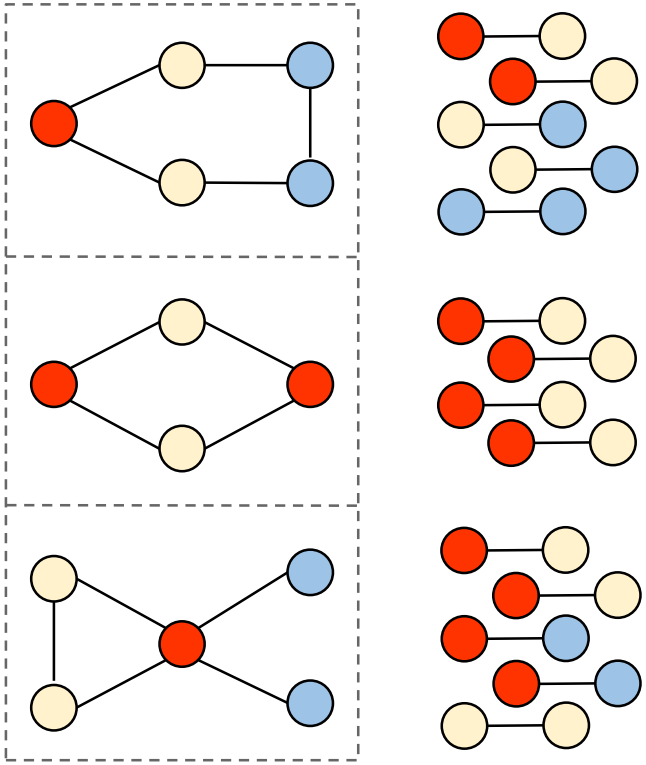
Task 6: Frequent Subgraph Mining

- Threshold 2/3



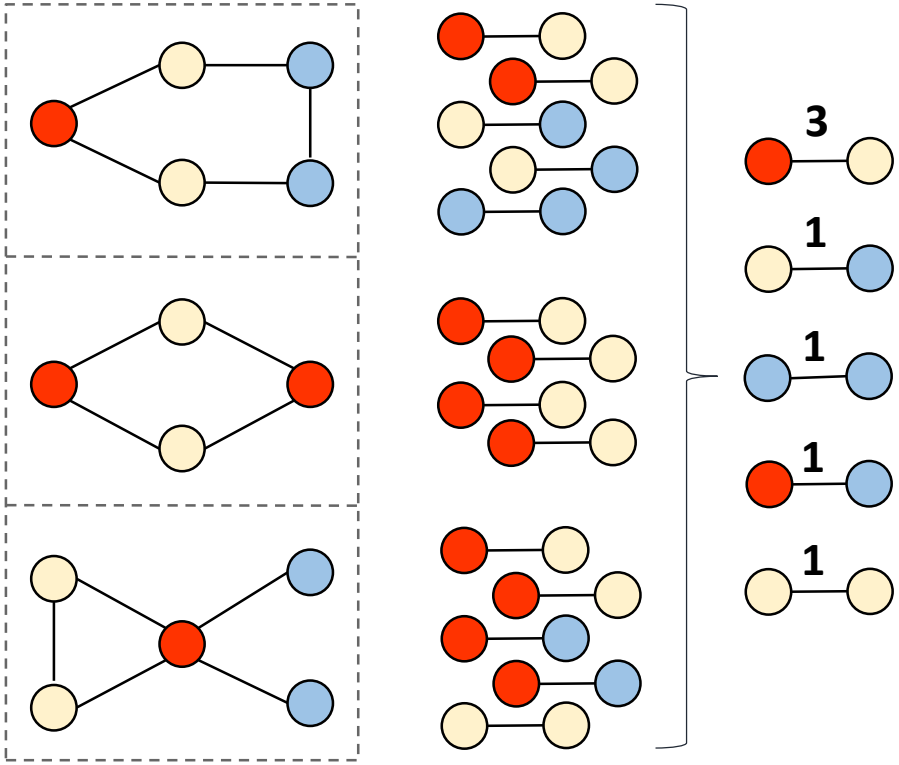
Task 6: Frequent Subgraph Mining

- Discover 1-edge subgraphs



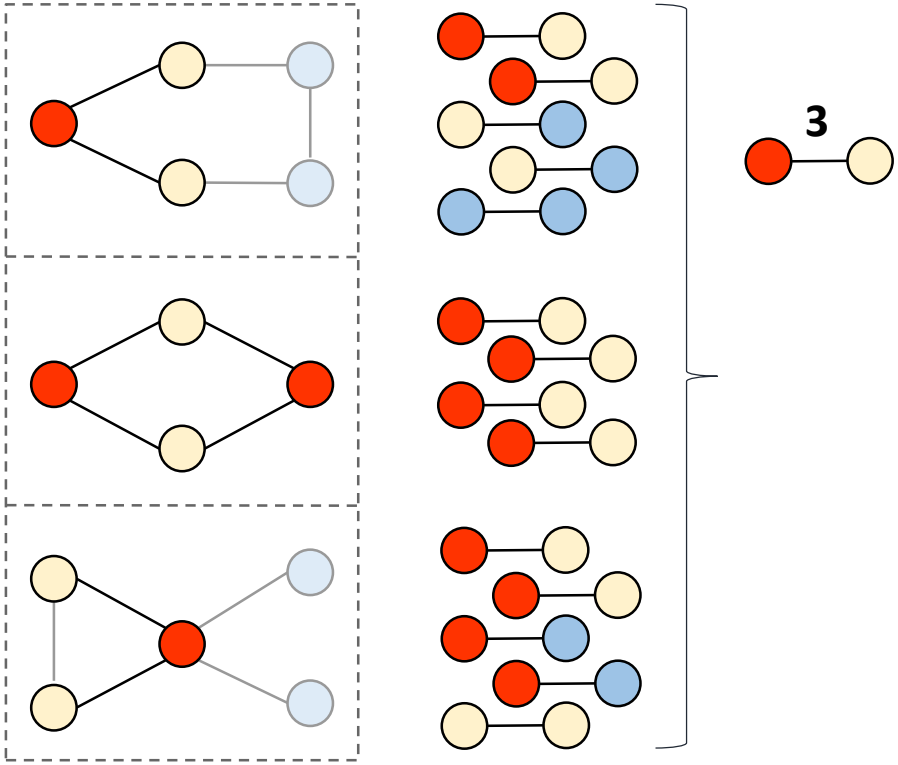
Task 6: Frequent Subgraph Mining

- Count support of 1-edge subgraphs



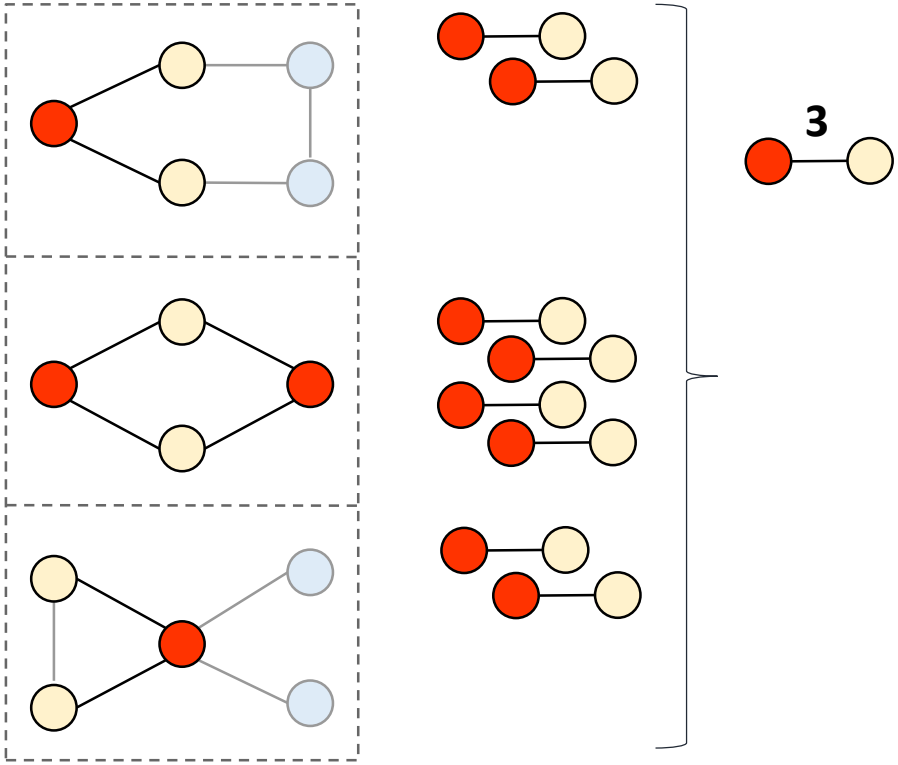
Task 6: Frequent Subgraph Mining

- Identify frequent 1-edge subgraphs



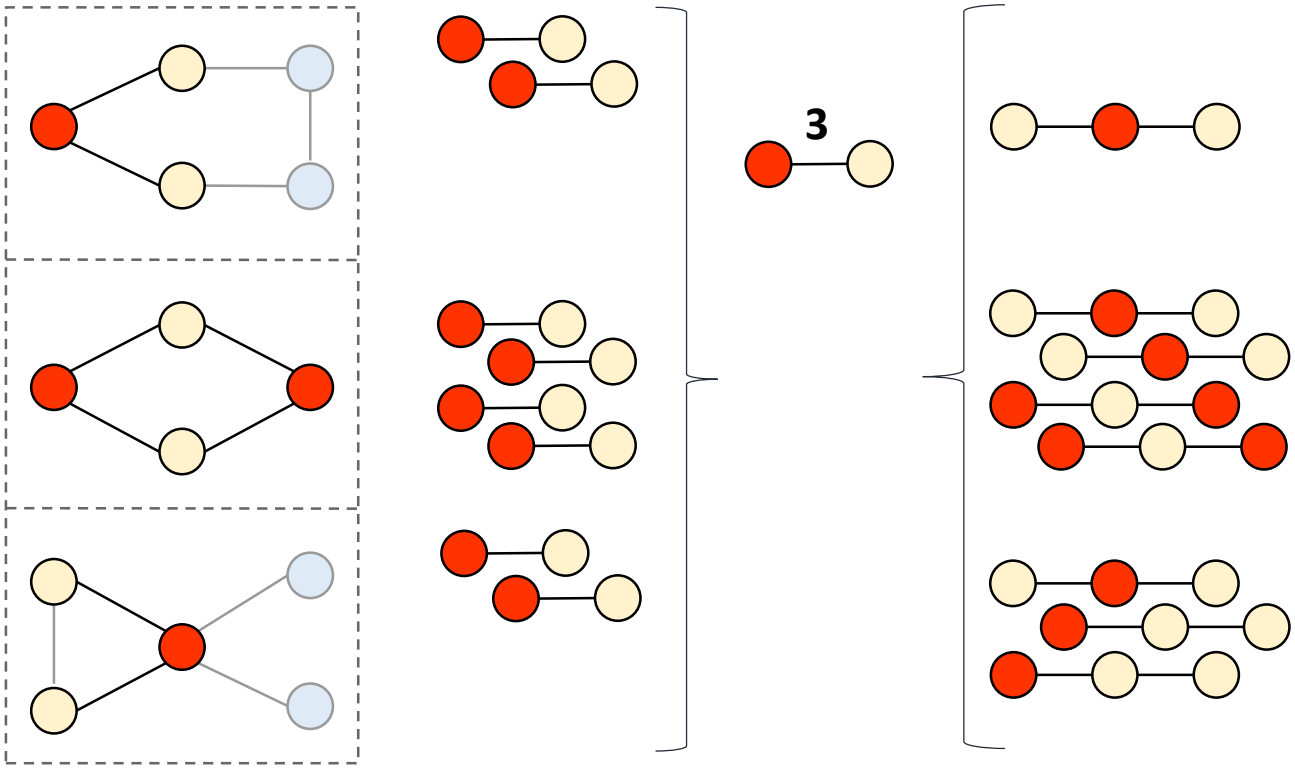
Task 6: Frequent Subgraph Mining

- Filter frequent embeddings of 1-edge subgraphs



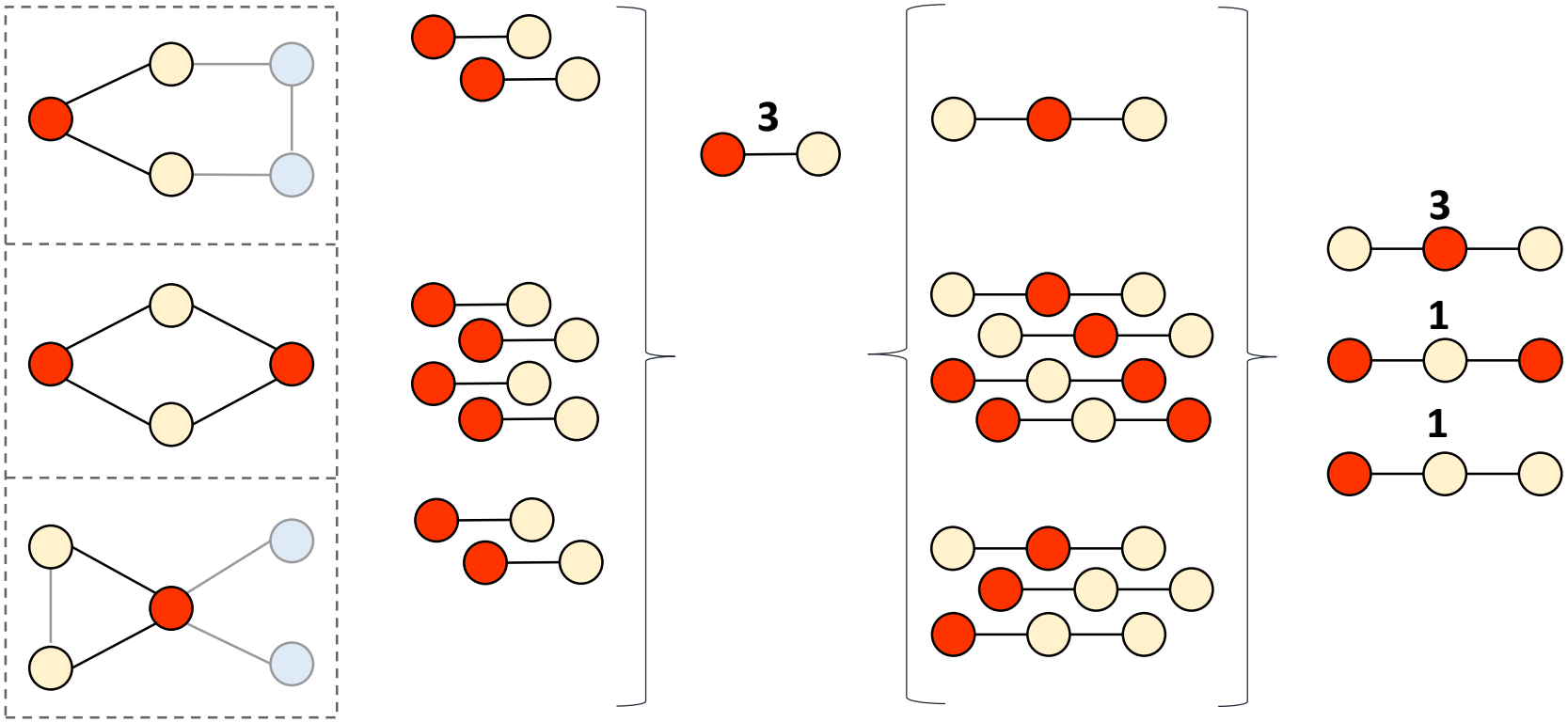
Task 6: Frequent Subgraph Mining

- Grow frequent subgraphs by 1 edge



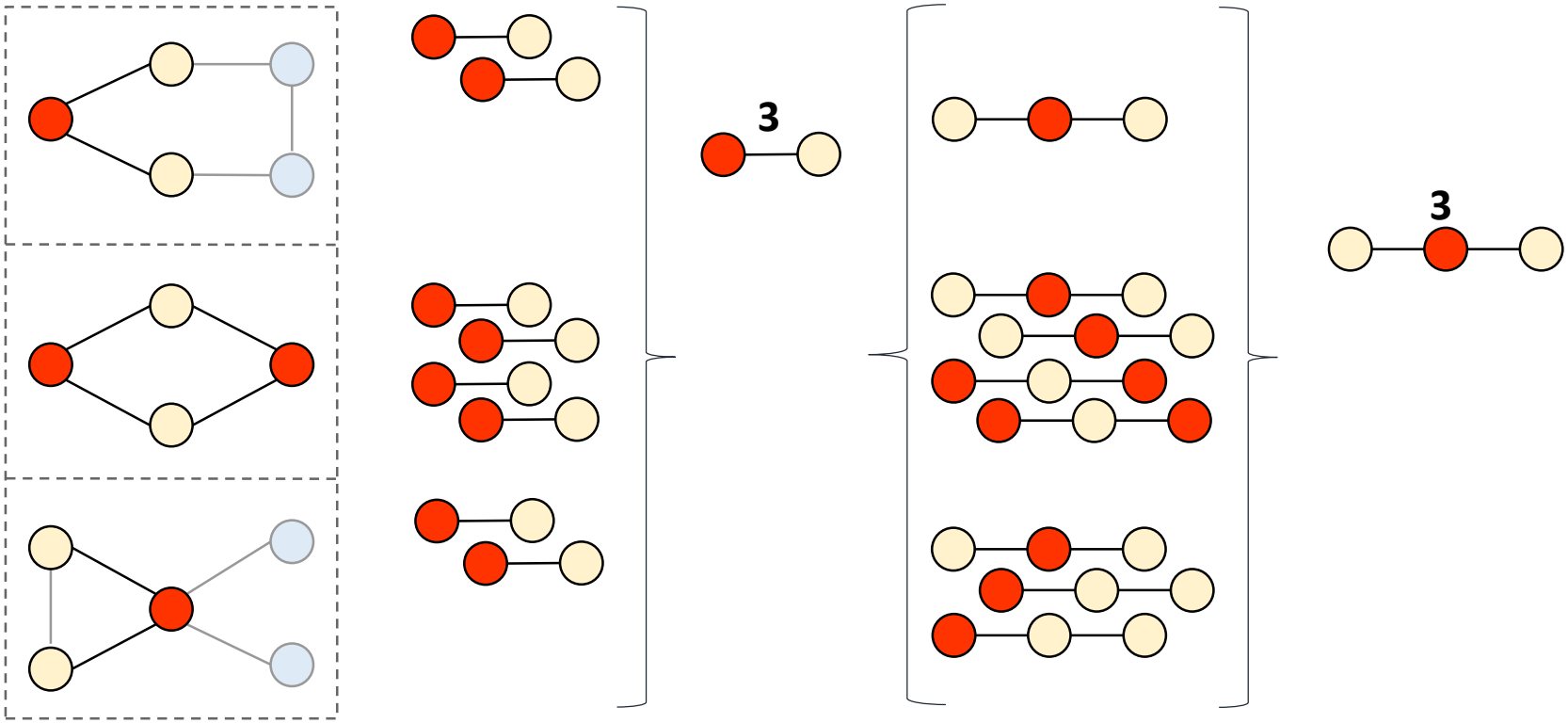
Task 6: Frequent Subgraph Mining

- Count support of 2-edge subgraphs



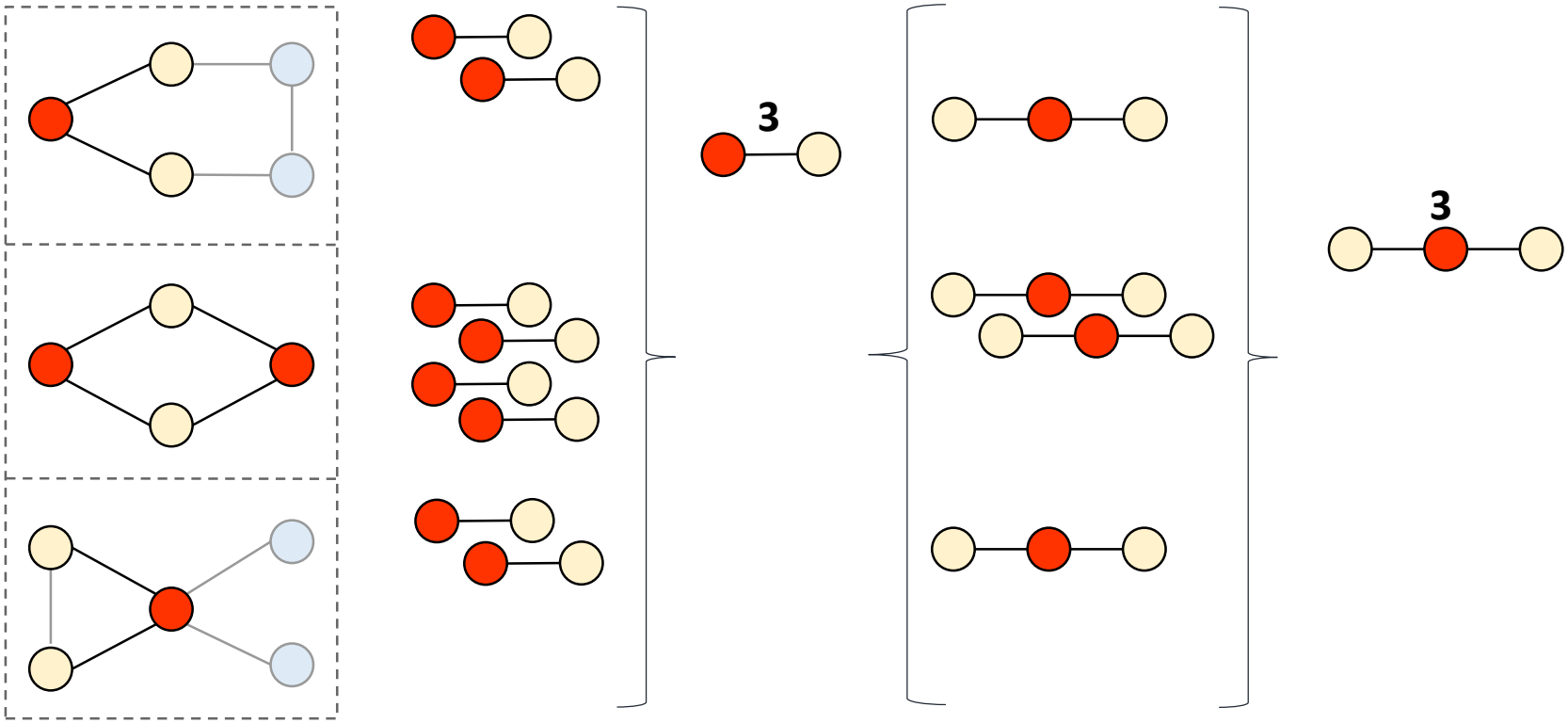
Task 6: Frequent Subgraph Mining

- Identify frequent 2-edge subgraphs

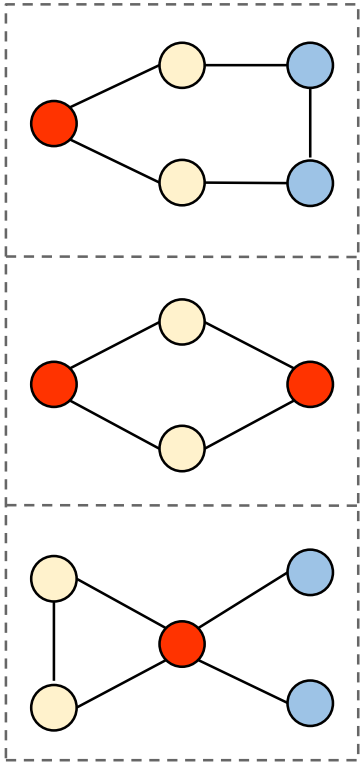


Task 6: Frequent Subgraph Mining

- Continue growing instances until all are infrequent



Task 6: Frequent Subgraph Mining



- Prototype implemented in Apache Flink
- Goal: Implementation in
  - Apache Spark
  - Apache Twill
  - Apache Storm
- Requirements
  - Profound knowledge in Java and/or Scala
  - First experience with Flink and target FW
- Material
  - *gSpan: Graph-based Substructure Pattern Mining*
- 2-3 Students

Themenübersicht

Thema	FW	#Studenten	Betreuer
Visualisierung von OSM-Daten	Geowave, Geomesa	2	Peukert
Tweet Analyse von News	Mahout	2	Christen
Holistic Ontology Matching	Gradoop, Flink	2	Christen
Analyse von Wetterdaten	Spark, Spark-R	2	Groß
Big OLAP: Datawarehouse	Kylin, Flink	3	Groß
Graph Metrics and Measures	Giraph, Flink, Spark	2	Junghanns
Graph Centrality Measures	Giraph, Flink, Spark	2	Junghanns
Random Walk With Restart	Giraph, Flink	2	Junghanns
Graph Summarization	Flink, Spark	2	Junghanns
Diffusion-based Graph Partitioning	Giraph, Flink	3	Junghanns
Frequent Subgraph Mining	Spark, Twill	3	Petermann