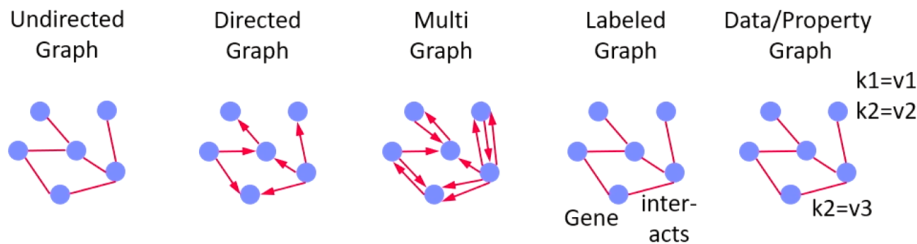


Terminology

- for further details see [[Graphs KR]]
- recap:

Terminology

- Graph $G = (V, E)$ of vertices V (set of nodes) and edges E (set of links between nodes)
- **Different types of graphs**



Terminology, cont.

- **Path**: Sequence of edges and vertices (**walk**: allows repeated edges/vertices)
- **Cycle**: Closed walk, i.e., a walk that starts and ends at the same vertex
- **Clique**: Subgraph of vertices where every two distinct vertices are adjacent

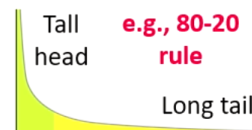
Metrics

- **Degree** (in/out-degree): number of incoming/outgoing edges of that vertex
- **Diameter**: Maximum distance of pairs of vertices (longest shortest-path)



Power Law Distribution

- Degree of most real graphs follows a power law distribution



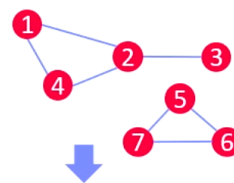
Vertex-Centric Processing

Programming Model

- Represent graph as collection of vertices w/ edge (adjacency) lists
- Implement algorithms via Vertex API
- Terminate if all vertices halted / no more msgs

```
public abstract class Vertex {
    public String getID();
    public long superstep();
    public VertexValue getValue();

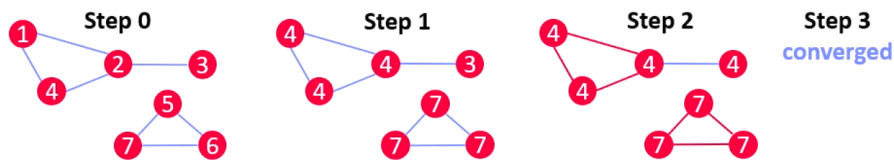
    public compute(Iterator<Message> msgs);
    public sendMsgTo(String v, Message msg);
    public void voteToHalt();
}
```



- 2 [1, 3, 4]
- 7 [5, 6]
- 4 [1, 2]
- 1 [1, 2, 4]
- 5 [6, 7]
- 3 [2]
- 6 [5, 7]

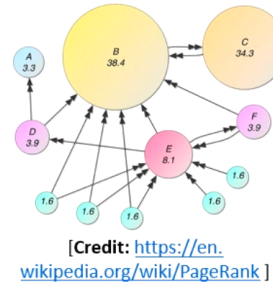
Example1: Connected Components

- Determine connected components of a graph (subgraphs of connected nodes)
- Propagate $\max(\text{current}, \text{msgs})$ if \neq current to neighbors, terminate if no msgs



Example 2: Page Rank

- Ranking of webpages by importance / impact
- #1: Initialize vertices to $1/\text{numVertices}()$
- #2: In each super step
 - Compute current vertex value:
 $\text{value} = 0.15/\text{numVertices}() + 0.85 \cdot \text{sum}(\text{msg})$
 - Send to all neighbors:
 $\text{value}/\text{numOutgoingEdges}()$



Graph-Centric Processing

Motivation

- Exploit graph structure for algorithm-specific optimizations (number of network messages, scheduling overhead for super steps)
- Large diameter / average vertex degree

Programming Model

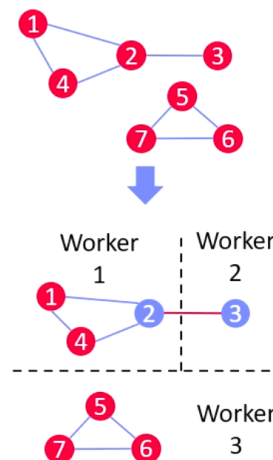
- Partition graph into subgraphs (block/graph)
- Implement algorithm directly against subgraphs (internal and boundary nodes)
- Exchange messages in super steps only between boundary nodes \rightarrow faster convergence



[Yuan Yuan Tian, Andrey Balmin, Severin Andreas Corsten, Shirish Tatikonda, John McPherson: From "Think Like a Vertex" to "Think Like a Graph". **PVLDB 2013**]

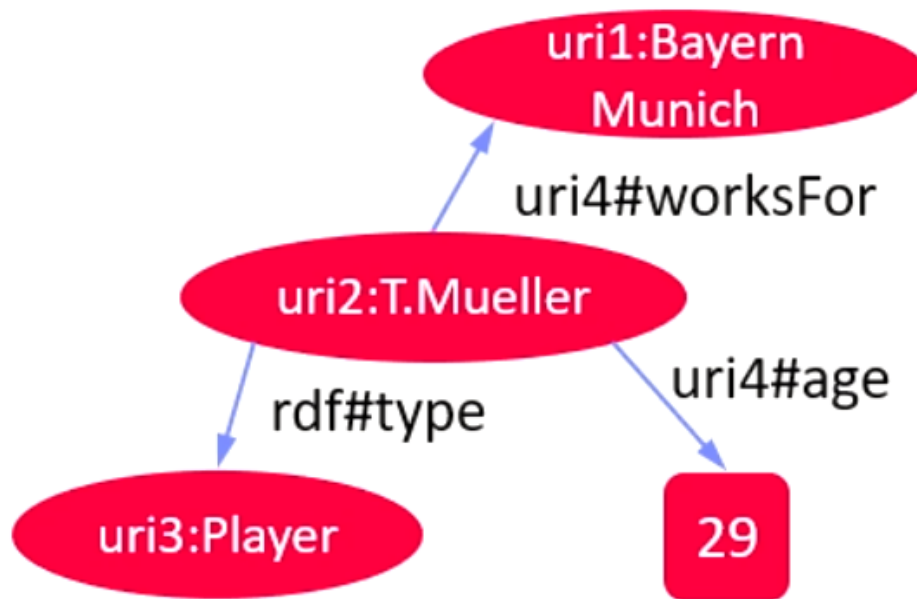


[Da Yan, James Cheng, Yi Lu, Wilfred Ng: Blogel: A Block-Centric Framework for Distributed Computation on Real-World Graphs. **PVLDB 2014**]



Ressource Description Framework

- RDF Data
 - data and meta data description as triples
 - (subject, predicate, object)
 - e.g. URIs or Literals



- RDF graphs are directed, labeled multigraph
- querying data

Querying RDF Data

- SPARQL (SPARQL Protocol And RDF Query Language)
- Subgraph matching

```
SELECT ?person
WHERE {
  ?person rdf:type uri3:Player ;
          uri4:worksFor uri1:"Bayern Munich" .
}
```

[[Data Models]]