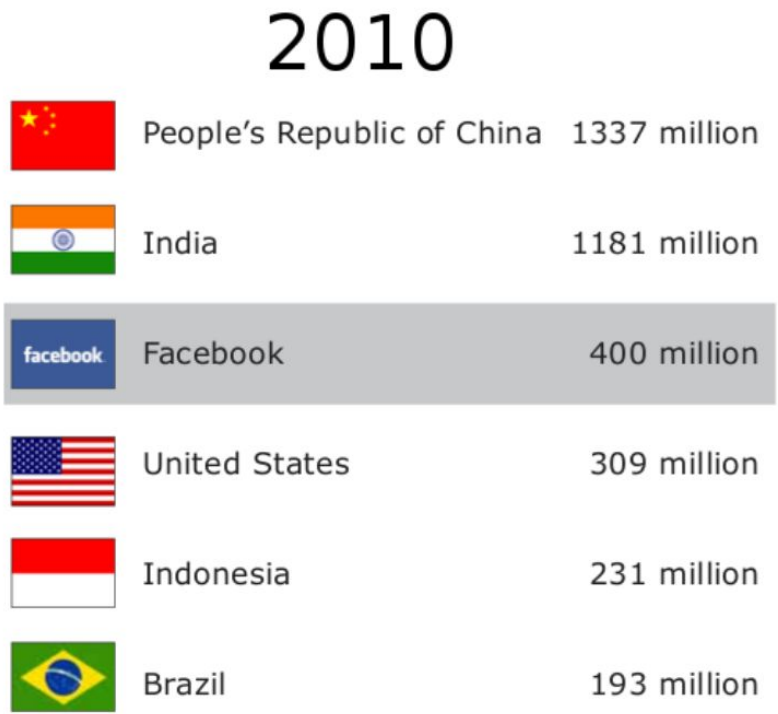
Mining Massive Databases

Graphs and Social Networks

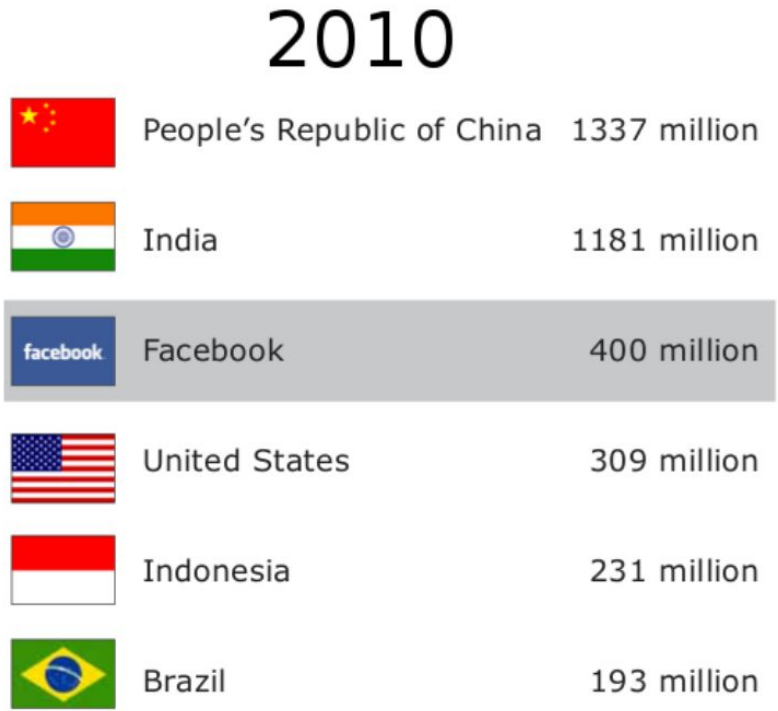
Diego Saez-Trumper

Data Science - UCU

Social media everywhere



Social media everywhere



2014

Facebook: 1350 Millions of active users (1120 Millionst through mobiles).

Twitter: 284 million active users.

Social media everywhere

2010

2017

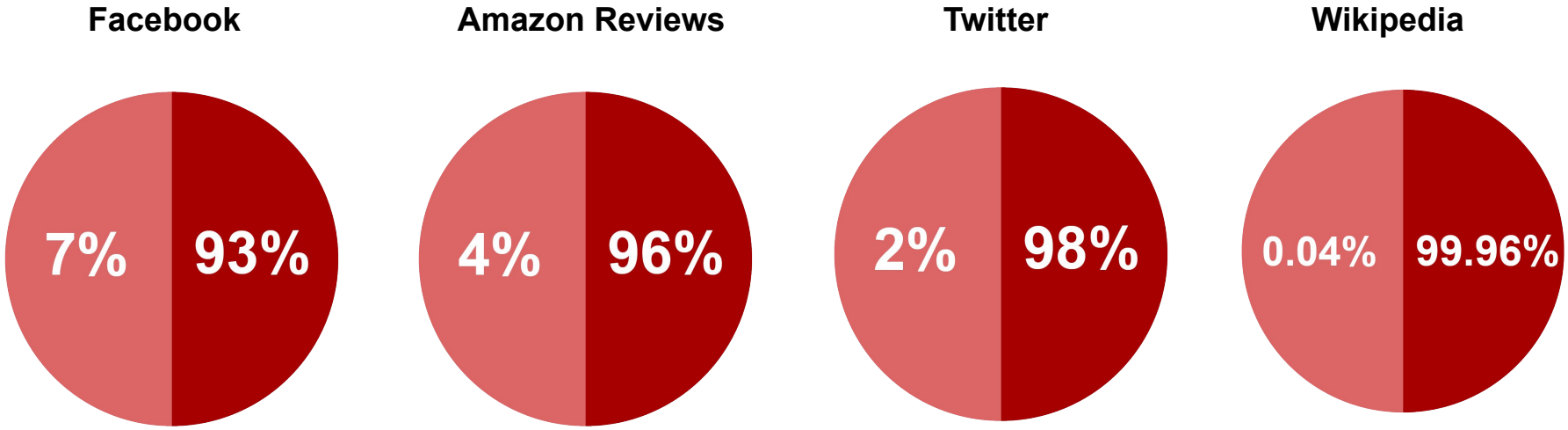
1. Facebook (1.9 billion users)
2. WhatsApp (1.2 billion users as of February 2017)
3. Messenger (1.2 billion users as of April 2017)
4. YouTube (1 billion users)
5. WeChat/Weixin (889 million users)
6. QQ (869 million users)
7. Instagram (700 million users)
8. Qzone (638 million users)
9. Twitter (328 million users)
10. Weibo (313 million users)

2024

Ranking	Social Media Network	Monthly Active Users (MAUs)	Monthly Organic Traffic
1	Facebook	3.06 billion	13,100,000,000
2	YouTube	2.70 billion	73,000,000,000
3	WhatsApp	2.40 billion	3,900,000,000
4	Instagram	2.35 billion	6,700,000,000
5	TikTok	1.67 billion	2,700,000,000
6	WeChat	1.31 billion	6,100,000
7	Messenger	1.10 billion	253,700,000
8	Telegram	900 million	615,100,000
9	Viber	820 million	17,300,000
10	Snapchat	800 million	189,300,000

source:https://explodingtopics.com/blog/top-social-media-platforms

How many users produce most of the content?



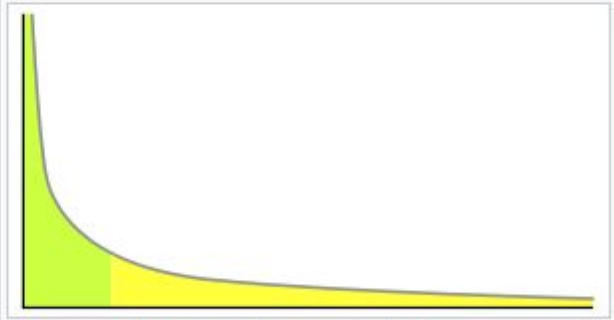
Baeza-Yates, R., & Saez-Trumper, D. (2015, August). Wisdom of the crowd or wisdom of a few? An analysis of users' content generation. In *Proceedings of the 26th ACM conference on hypertext & social media* (pp. 69-74).

Power law

From Wikipedia, the free encyclopedia

Not to be confused with Force (law).
For other uses, see Power.

In statistics, a **power law** is a functional relationship between two quantities, where a relative change in one quantity results in a proportional relative change in the other quantity, independent of the initial size of those quantities: one quantity varies as a power of another. For instance, considering the area of a square in terms of the length of its side, if the length is doubled, the area is multiplied by a factor of four.[1]



An example power-law graph that demonstrates ranking of popularity. To the right is the long tail, and to the left are the few that dominate (also known as the 80–20 rule).

https://en.wikipedia.org/wiki/Power_law

Social networks by type / purpose

Type / Example(s)

- **Friends** / Facebook, VK
- **Information** / Twitter, Reddit
- **Geosocial** / Foursquare
- **Content sharing** / Youtube, Snapchat
- **Communication** / Whatsapp, FBsharing
- **Professional Contacts** / Linkedin
- **Etc.**

Social Networks or Social Media?

Social Networks as Graphs

Graph type / Example

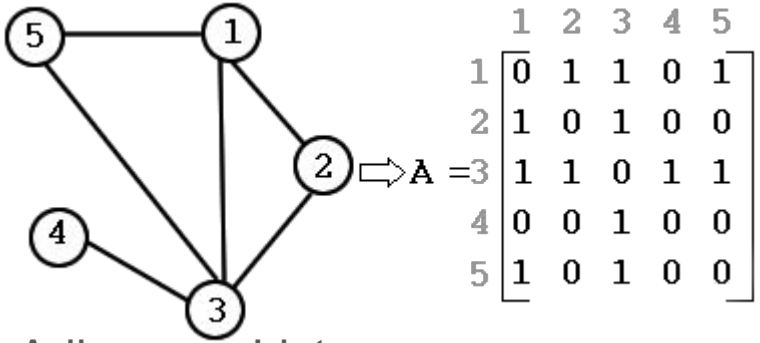
- **Undirected Graph /** Friends network in Facebook
- **Directed Graph /** Follower network in Twitter, Instagram, TikTok
- **Bipartite Graph /** Users to places in Foursquare, Users Comments to Videos in Youtube, Guest to Hotels in Booking

Acyclic?

Weighted?

Graph representation

- Adjacency Matrix



$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

- Adjacency List:
  - (1,2),(1,3),(1,5),(2,3),(3,4),(3,5)

Graphs - Basic Metrics

- **Global:**
  - Number of nodes and edges
  - Diameter
  - Number or percentage of triangles
  - Density
  - Avg. Indegree
- **Local:**
  - Clustering Coefficient
  - Indegree
  - Outdegree
  - Centrality metrics:
    - Betweenness centrality
    - Pagerank
    - HITS

Measuring User Influence in Twitter: The Million Follower Fallacy

Meeyoung Cha[*] **Hamed Haddadi**[†] **Fabrício Benevenuto**[‡] **Krishna P. Gummadi**[*]

[*]Max Planck Institute for Software Systems (MPI-SWS), Germany
[†]Royal Veterinary College, University of London, United Kingdom
[‡]CS Dept., Federal University of Minas Gerais (UFMG), Brazil

Abstract

Directed links in social media could represent anything from intimate friendships to common interests, or even a passion for breaking news or celebrity gossip. Such directed links determine the flow of information and hence indicate a user's *influence* on others—a concept that is crucial in sociology and viral marketing. In this paper, using a large amount of data collected from Twitter, we present an in-depth comparison of three measures of influence: indegree, retweets, and mentions. Based on these measures, we investigate the dynamics of user influence across topics and time. We make several interesting observations. First, popular users who have high indegree are not necessarily influential in terms of spawning retweets or mentions. Second, most influential users can hold significant influence over a variety of topics. Third, influence is not gained spontaneously or accidentally, but through concerted effort such as limiting tweets to a single topic. We believe that these findings provide new insights for viral marketing and suggest that topological measures such as indegree alone reveals very little about the influence of a user.

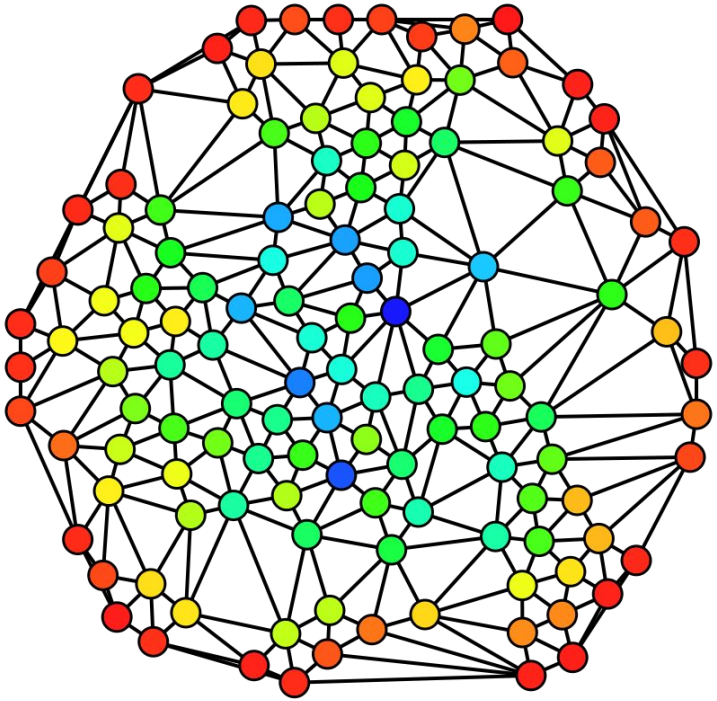
a minority of users, called *influentials*, excel in persuading others (Rogers 1962). This theory predicts that by targeting these influentials in the network, one may achieve a large-scale chain-reaction of influence driven by word-of-mouth, with a very small marketing cost (Katz and Lazarsfeld 1955). A more modern view, in contrast, de-emphasizes the role of influentials. Instead, it posits that the key factors determining influence are (*i*) the interpersonal relationship among ordinary users and (*ii*) the readiness of a society to adopt an innovation (Watts and Dodds 2007; Domingos and Richardson 2001). This modern view of influence leads to marketing strategies such as collaborative filtering. These theories, however, are still just theories, because there has been a lack of empirical data that could be used to validate either of them. The recent advent of social networking sites and the data within such sites now allow researchers to empirically validate these theories.

Moving from theory into practice, we find that there are many other unanswered questions about how influence diffuses through a population and whether it varies across topics and time. People have different levels of expertise on

Betweenness centrality

- It is related to the number of shortest path that goes through each node.
- In the figure, betweenness centrality is represented by node's color, being red the less central and blue the most central.

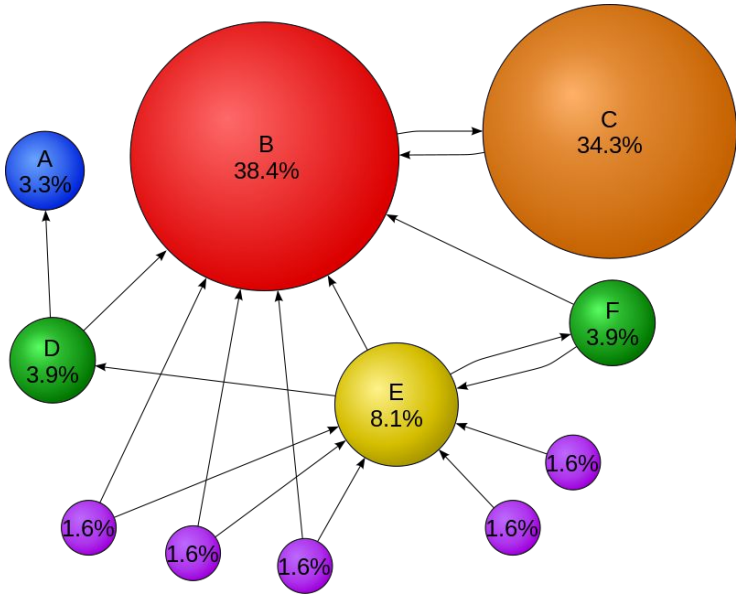
[source: wikicommos]



Pagerank

The general idea works as follows:

- Each node *n* starts with a budget *b*
- That budget is equally distributed across all the nodes in their outgoing edges. So if
- In the next iteration *n* (could) receive new budget from the nodes in its incomings edges, and then distribute them to nodes in the outgoing edges.
- This is repeated until budget in each node is stable (converge).



Temporal Graphs vs Static Graphs

- Graph might change along time
  - Think in a communication graph where weights represent the number of messages exchanged by pair of nodes.
  - Messages are received as stream, changing the weights constantly.
- Temporal graphs can be represented in different ways:
  - As a set of snapshots of graphs within a given window.
  - Encoding temporal information on the edges

Popular tools for analysing and draw graphs

Some popular packages for graph processing:

- Python:
  - networkx
  - igraph
  - Graph-tool
  - Snap
- Spark
  - GraphX
  - Graphframes
- Other tools for drawing graphs
  - Gephi
  - Graphviz
  - d3

Graph manipulation

Some of usual manipulations, and operations that we do with graphs are:

- Obtain the number of connected components
- Traversals from a given node (BFS, DFS)
- Rank nodes by local metrics
- Get shortest path between pair of nodes

Other tasks on Graphs

- Graph partitioning
- Clustering
- Finding (overlapping communities) communities
- Counting Triangles
- Finding relevant nodes
- Plotting

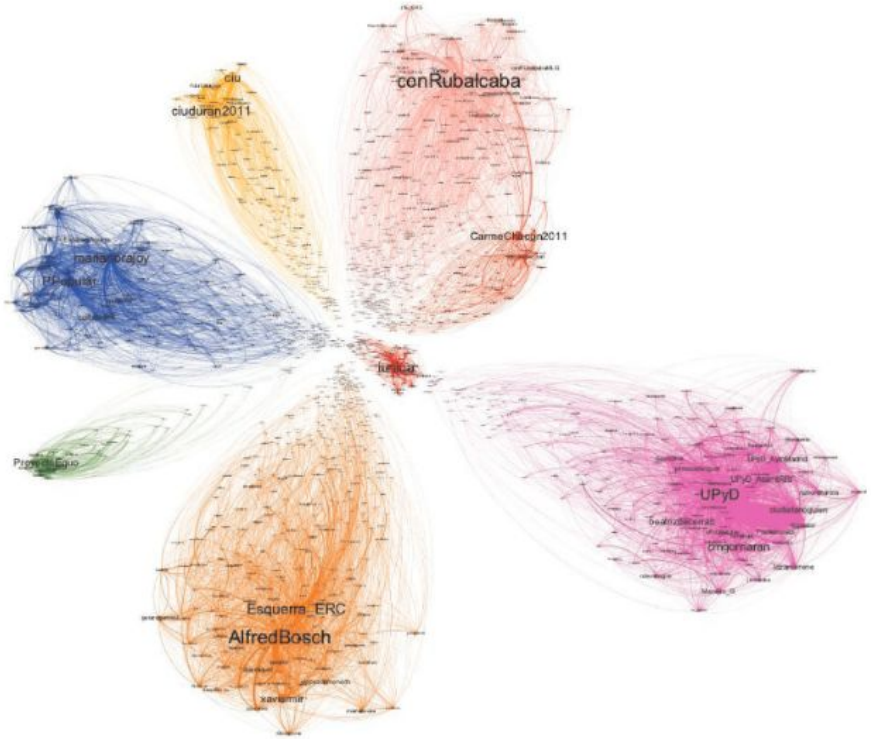


Figure 5. Retweet Graph.
Note: The size of each node represents its in-degree, while the color corresponds to the cluster it belongs to.

Aragón, Pablo, et al. "Communication dynamics in Twitter during political campaigns: The case of the 2011 Spanish national election." *Policy & internet* 5.2 (2013): 183-206.

Graph embeddings

Graph embedding techniques, applications, and performance: A survey

Palash Goyal*, Emilio Ferrara

University of Southern California, Information Sciences Institute 4676 Admiralty Way, Suite 1001 Marina del Rey, CA 90292, USA

ARTICLE INFO

Article history:
Received 10 December 2017
Revised 13 March 2018
Accepted 15 March 2018
Available online 16 March 2018

Keywords:
Graph embedding techniques
Graph embedding applications
Python graph embedding methods GEM library

ABSTRACT

Graphs, such as social networks, word co-occurrence networks, and communication networks, occur naturally in various real-world applications. Analyzing them yields insight into the structure of society, language, and different patterns of communication. Many approaches have been proposed to perform the analysis. Recently, methods which use the representation of graph nodes in vector space have gained traction from the research community. In this survey, we provide a comprehensive and structured analysis of various graph embedding techniques proposed in the literature. We first introduce the embedding task and its challenges such as scalability, choice of dimensionality, and features to be preserved, and their possible solutions. We then present three categories of approaches based on factorization methods, random walks, and deep learning, with examples of representative algorithms in each category and analysis of their performance on various tasks. We evaluate these state-of-the-art methods on a few common datasets and compare their performance against one another. Our analysis concludes by suggesting some potential applications and future directions. We finally present the open-source Python library we developed, named GEM (*Graph Embedding Methods*, available at https://github.com/palash1992/GEM), which provides all presented algorithms within a unified interface to foster and facilitate research on the topic.

https://github.com/palash1992/GEM

Graph embeddings

Category	Year	Published	Method	Time Complexity	Properties preserved
Factorization	2000	Science[26]	LLE	$O(\|E\|d^2)$	1^{st} order proximity
	2001	NIPS[25]	Laplacian Eigenmaps	$O(\|E\|d^2)$	
	2013	WWW[21]	Graph Factorization	$O(\|E\|d)$	
	2015	CIKM[27]	GraRep	$O(\|V\|^3)$	$1 - k^{th}$ order proximities
	2016	KDD[24]	HOPE	$O(\|E\|d^2)$	
Random Walk	2014	KDD[28]	DeepWalk	$O(\|V\|d)$	
	2016	KDD[29]	*node2vec*	$O(\|V\|d)$	$1 - k^{th}$ order proximities, structural equivalence
Deep Learning	2016	KDD[23]	SDNE	$O(\|V\|\|E\|)$	1^{st} and 2^{nd} order proximities
	2016	AAAI[30]	DNGR	$O(\|V\|^2)$	$1 - k^{th}$ order proximities
	2017	ICLR[31]	GCN	$O(\|E\|d^2)$	$1 - k^{th}$ order proximities
Miscellaneous	2015	WWW[22]	LINE	$O(\|E\|d)$	1^{st} and 2^{nd} order proximities

Table 1: List of graph embedding approaches

https://github.com/palash1992/GEM

Graph Datasets

You can find large graph datasets in these sites:

- Stanford Network Analysis Project (SNAP)
- Koblenz Network Collection (KONECT)

Other resources:

- Lada Adamic: Social Network Analysis Course (Youtube, 2015).
- Easley & Kleinberg: Networks, Crowds, and Markets (Book, 2010)
- Fil Menczer et al.: A First Course in Network Science (Jupyter Notebooks)

Example of GraphFrames in Spark

```
# start with pyspark --packages graphframes:graphframes:0.5.0-spark2.1-s_2.11
# Create a Vertex DataFrame with unique ID column "id"
v = sqlContext.createDataFrame([
  ("a", "Alice", 34),
  ("b", "Bob", 36),
  ("c", "Charlie", 30),
], ["id", "name", "age"])
# Create an Edge DataFrame with "src" and "dst" columns
e = sqlContext.createDataFrame([
  ("a", "b", "friend"),
  ("b", "c", "follow"),
  ("c", "b", "follow"),
], ["src", "dst", "relationship"])
# Create a GraphFrame
from graphframes import *
g = GraphFrame(v, e)

# Query: Get in-degree of each vertex.
g.inDegrees.show()

# Query: Count the number of "follow" connections in the graph.
g.edges.filter("relationship = 'follow'").count()

# Run PageRank algorithm, and show results.
results = g.pageRank(resetProbability=0.01, maxIter=20)
results.vertices.select("id", "pagerank").show()
```

- To copy/paste this example, check here
- Tip: Graphframes works much faster when node IDs are integers
- You can use graphframes from collab.

Reading CSV: Defining a schema in PySpark

```
from pyspark.sql.types import *

schema = StructType([
    StructField("wiki_db", StringType(), True),
    StructField("page_id", IntegerType(), True),
    StructField("link",StringType() , True),
    StructField("creation_time",StringType() , True),
    StructField("wikidata",StringType() , True)
])


ref = spark.read.csv('references.csv',schema=schema)
```

Optional Homework

- Download the High-energy physics theory citation network
  - http://snap.stanford.edu/data/cit-HepTh.txt.gz
- From this graph report:
  - The number of connected components
  - **Two different** lists of top-10 most relevant papers in this graph (help: one can be the list of nodes by indegree)
- For this task you **must use Graphframes**:
  - Graphframes
    - Tips: pyspark --packages graphframes:graphframes:0.8.1-spark3.0-s_2.12
    - **OR**

```
from pyspark import SparkContext
from pyspark import SparkConf
from pyspark.sql.session import SparkSession
```

```
]: PACKAGES = ('graphframes:graphframes:0.5.0-spark2.1-s_2.11')

conf = SparkConf().setAll([('spark.master', 'local[*]'),
                           ('spark.app.name', 'Test app'),
                           ('spark.jars.packages', PACKAGES)])
sc = SparkContext.getOrCreate(conf)
sqlContext = SparkSession(sc)
```