# Link and Graph Mining in the Big Data Era

**Ana Paula Appel and Luis G. Moyano**

**Abstract** Graphs are a convenient representation for large sets of data, being complex networks, social networks, publication networks, and so on. The growing volume of data modeled as complex networks, e.g. the World Wide Web, and social networks like Twitter, Facebook, has raised a new area of research focused in complex networks mining. In this new multidisciplinary area, it is possible to highlight some important tasks: extraction of statistical properties, community detection, link prediction, among several others. This new approach has been driven largely by the growing availability of computers and communication networks, which allow us to gather and analyze data on a scale far larger than previously possible. In this chapter we will give an overview of several graph mining approach to mine and handle large complex networks.

## 1 Introduction

Over the past years the amount of data collected has increased substantially, especially with the growing availability of the World Wide Web, expansion not only for text but also with images and video. For instance, Facebook estimates that video exhibition move from 1 billion in 2015 to 8 billion in 2016.

Social networks and social media are becoming a regular part of people lives, as a person spends, in average, almost 2 hours per day in a social network. These kind of data was studied in the past by social scientists, however in a much smaller scale. They usually worked with hundreds of nodes to answer questions such as which person is the most connected in the network, or which one, if removed, could break the connection among all the individuals. Today, social networks are composed by hundreds of million users and the analysis is different not only in the class of

A.P. Appel (✉)
IBM Research, São Paulo, Brazil
e-mail: apappel@br.ibm.com

L.G. Moyano
CONICET and Facultad de Ciencias Exactas y Naturales,
Universidad Nacional de Cuyo, Mendoza, Argentina

A.Y. Zomaya and S. Sakr (eds.), *Handbook of Big Data Technologies*,
DOI 10.1007/978-3-319-49340-4_17

techniques but also in which type of questions we want to answer. Asking which person will break the connection of the network does not make sense anymore, since there could be no one to cause this kind of damage in a network of such size. This crucial change in scale made these type of problems much more interesting and lead to emergence of a new research field: **graph mining**.

These data are naturally mapped in complex networks that are represented using graphs. Under this model, nodes are entities (e.g. people or groups of people), and edges represent some kind of interaction between entities (e.g. friendship). Another example is that of information networks, in which the nodes are information resources such as Web pages or documents, and edges represent logical connections such as hyperlinks, citations, or cross-references and so on.

These large volume of data makes it easy to study global phenomena that are not discernible in smaller networks, for example, how a community is born, how a network evolves over time, understand the importance of single node or an edge, among many examples that were almost impossible to tackle in small networks.

Graphs are very a powerful tool to express and model mathematically complex network structures, appearing in many domains, whenever it is useful to represent how things are either physically or logically linked to one another in a network structure.

A graph $\mathcal{G}$ is mathematically represented as $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, on which $|\mathcal{V}| = n$ represents the number of nodes (or vertex set), and $|\mathcal{E}| = m$ represents the number of edges (or links), and a relation that associates with each edge two vertexes [109]. We say that two nodes are neighbors if they are connected by an edge.

The graph-based representation used for data has substantial non-trivial topological features, with patterns of connections between their elements that are neither purely regular nor purely random. For this reason this representation is usually referred to as a Complex Network.

The study of complex networks brought to light important properties such as power-law degree distributions [47], the Small World phenomenon [104], among several others [109]. These patterns help us understand the interaction of people in social networks [91, 99] as well as the dissemination of information and diseases [36], and has other practical applications such as anomaly detection [7] and so on.

In this chapter we will navigate through several graph mining task, such as pattern discovery using statistical properties (Sect. 2), community detection (Sect. 5), link prediction (Sect. 4). We also will talk about how to represent networks with weight (Sect. 7), multiple edges (Sect. 8) and temporally (Sect. 3). We also we show that networks can be use to represent knowledge and map knowledge bases (Sect. 6.2). We also present several platforms to store and process large graphs (Sect. 6). We finish this chapter present what are the big challenges and open issues of this chapter (Sect. 9) and than we conclude (Sect. 10).

## 2 Definitions

A graph is a useful way to specify relationships among a collection of items. A graph consists of a set of objects called *nodes*, with certain pairs of these objects connected by links called *edges*. We say that two nodes are neighbors if they are connected by an edge. A complex network is modeled as a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, on which $\mathcal{V}$ represents the number of nodes or verticals $N = |\mathcal{V}|$, and $\mathcal{E}$ represents the number of edges or links $M = |\mathcal{V}|$. The traditional way of represent a graph $\mathcal{G}$ computationally is the adjacency matrix, which is a square matrix $\mathbf{A} = N \times N$ where $\mathbf{A}_{i,j} = 1$ is $(v_i, v_j) \in \mathcal{E}$ and 0 otherwise [109].

A graph is undirect if $(v_i, v_j) \in \mathcal{E} \Leftrightarrow (v_j, v_i) \in \mathcal{E}$, that is, the edges are unordered pairs.

However, in many cases, it is desirable to express asymmetric relationships and other attributes such as weights, time or multiple relations in the links. Thus, the graph can become directed where the presence of $v_j, v_i \in \mathcal{E}$ does not imply that $v_i, v_j \in \mathcal{E}$. For example, A points to B but not vice versa, which means that edges are ordered pairs. It is possible to also have weighted networks, $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$ where $w_i \in \mathcal{W}$, which means that each edge will have a weight $w_i$ associated to it (weights are discussed in more detail in Sect. 7).

The *node degree* $\tau(v_i)$, also called *neighborhood* of a node, can be defined by the amount of incident edges on node $v_i$ (Table 1).

Another important measure in real complex network is called the *clustering coefficient*, which is proportional to the total number of triangles that a node or a network has.

A *triangle* $\Delta$ of a graph $\mathcal{G}$ is a set of three completely connected nodes where $(u, v, w) \in \mathcal{V}$ and edges $(u, v), (v, w), (w, u) \in \mathcal{E}$.

In many networks it is found that if vertex A is connected to vertex B and vertex B to vertex C, then there is a high probability that vertex A will also be connected to vertex C. In the language of network theory, a friend's friend is likely also to be

**Table 1** Symbols used in this chapter

| Symbols | Description |
|---|---|
| $\mathbf{A}$ | Adjacency matrix |
| $\mathcal{G}$ | Graph |
| $\mathcal{E}$ | Set of edges |
| $\mathcal{V}$ | Set of nodes |
| $v_i$ | Node |
| $e_k$ | Edge |
| $\Delta$ | Triangle |
| $\tau(v_i)$ | Degree of node $v_i$ |
| $\mathrm{C}(v_i)$ | Cluster coefficient of node $v_i$ |
| $\mathrm{C}(\mathcal{G})$ | Cluster coefficient of $\mathcal{G}$ |

a friend. In terms of network topology, transitivity means the presence of a high number of triangles in the network [146].

Formally, the cluster coefficient can be expressed by the following equation:

$$C(v_i) = \frac{2 * \Delta(v_i)}{d(v_i) * (d(v_i) - 1)} \tag{1}$$

A node $v_i$ with degree $|\tau(v_i)|$ has at most $\tau(v_i) * (\tau(v_i) - 1)/2$ edges that could exist among them, being $\Delta(v_i)$ the fraction of edges that really exist (the number of triangles). Thus, the cluster coefficient in Eq. 1 $C(v_i)$ of a node $v_i$ is the proportion of edges among nodes that are at distance 1 if the neighborhood of $v_i$, divided by the total number of edges that could exist among them. Also, $C(v_i)$ is the fraction of triangles centered at node $v_i$ which $(d(v_i) * (d(v_i) - 1))/2$ triangles could exist.

The *global cluster coefficient* $C(\mathcal{G})$ is the average of the clustering of all nodes $C(v_i)$ from graph $\mathcal{G}$, divided by the total number of nodes $n$.

$$C(\mathcal{G}) = \frac{1}{N} * \sum_{i=1}^{N} C(v_i) \tag{2}$$

Nodes are also important in real graphs to analyze *degree probability distribution* [47]. We define *pk* to be the fraction of nodes in the network that have degree $k$. Equivalently, *pk* is the probability that a node chosen uniformly at random has degree $k$. A plot of *pk* for any given network can be formed by making a histogram of the degrees of nodes. This histogram is the degree distribution for the network. An example of degree distribution from DBLP[1] is presented in Fig. 1.

In a random graph of the type studied by Erdõs and Rényi [109], each edge is present or absent with some constant probability $p$. As a result, the degree distribution is, as mentioned earlier, binomial, or Poisson in the limit of large graph size.

Real-world networks are mostly found to be very unlike the random graph in their degree distributions [2]. Many of them asymptotically follow power-laws in their tails: $pk \approx k^{\alpha}$ for some constant exponent $\alpha$. The degrees of the nodes in most networks are strongly right-skewed, meaning that their distribution has a long right tail of values that are far above the mean. In the context of social networks, for example, power-law distribution means that most of people has few friends and few people has a lot of friends. There are several metrics that also follows a power-law in real networks, as nodes and edges during evolution [90], triangle distribution [142] and weights [100].

Another important characteristic observed in real networks is the small diameter. In graph theory, the *diameter* is defined as been the longest path among all the shortest paths. However, in this definition the diameter is susceptible to outliers if the graph has a long chain. Also, to calculate the diameter in real large graphs is computationally very expensive.

[1] http://dblp.uni-trier.de.

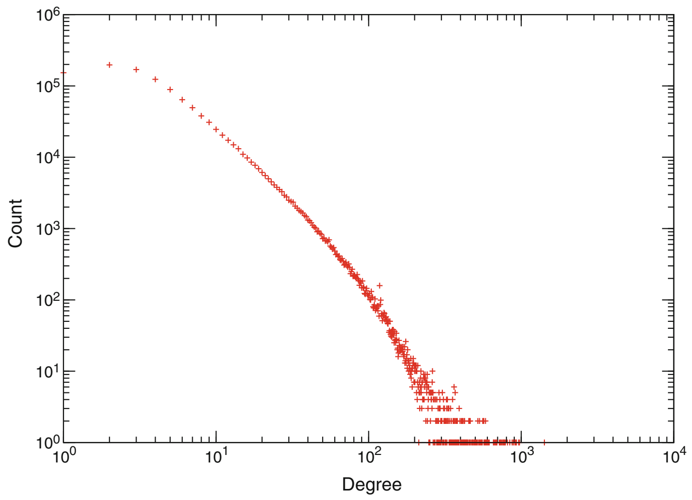

**Fig. 1** Degree distribution from co-authorship network extract from DBLP

One of the traditional metrics for diameter is the effective diameter [79, 117], defined as $f(\mathcal{G})$ of a graph $\mathcal{G}$. It is the minimum number of hops (steps or links) in which 90% of all connected pairs of nodes can reach each other. One way to calculate the effective diameter is build what we call a *Hop Plot*. The hop plot shows each distance, starting in one, until a distance where the number of nodes reach does not change over a very small threshold. An example of this plot is shown in Fig. 2. As we can see, after distance 7 the number of nodes reached (y axis) do not longer increase.

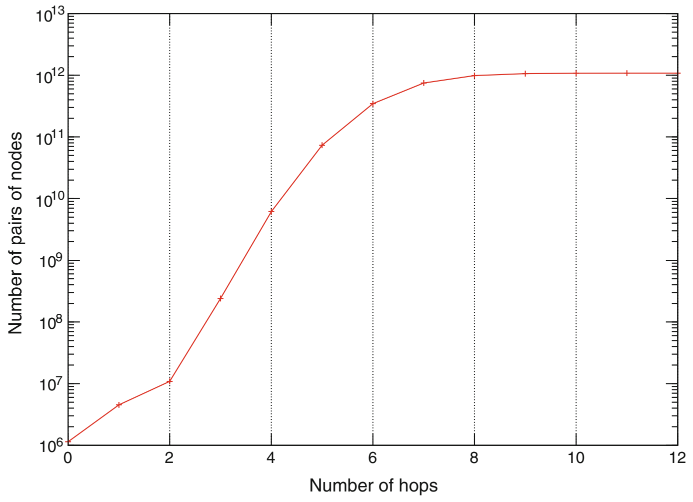

**Fig. 2** Hop Plot from co-authorship network extract from DBLP

## 3 Temporal Evolution

One of many interesting characteristics of complex networks are the ones connected with temporal evolution. Network evolution has attracted a lot of interest in the last years, not only because the complex network field has reached a stable knowledge point about the simplest case, that is the undirected, unweighted, simple network. Thus, the next step is to start to understand special cases of networks and evolution is one of these cases. A complex network evolves over time with the creation and deletion of nodes and edges, for example, when people join or leave a social network, they create or break friendship ties.

Today, there are mainly three models of temporal complex networks.

The most traditional such network is a condensate in snapshots, each one representing a period of time. The snapshots are represented by a series of graphs $\mathcal{G}_1, ..., \mathcal{G}_T$, so that $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ represents the graph at time $t$. Since $\mathcal{G}_1, ..., \mathcal{G}_T$ represent different snapshots of the same graph, we have $\mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{E}_t \subseteq \mathcal{E}$. For simplicity most of presentation assumes that, as the graph evolves, nodes and edges are only added and never deleted, that is, $\mathcal{V}_1 \subseteq \mathcal{V}_2 \subseteq ...\mathcal{V}_T$ and $\mathcal{E}_1 \subseteq \mathcal{E}_2 \subseteq ...\mathcal{E}_T$.

The second one follows the data stream model, where a large number of edges representing interactions are continuously received over time and are superposed over a much larger network. An example of such a scenario would be a Twitter post stream, in which several posts are continuously received over time [3].

Another way to represent temporal networks is through Time-Varing Graphs models (TGVs) also known in the literature as temporal or time-dependent networks [29].

TGVs graphs can be easily converted in snapshots by creating a graph with an edge between nodes, if and only if there is connection between nodes during a time interval. A example is shown in Fig. 3 where we illustrate a network in time $t_0$, $t_1$ and $t_2$. Networks (a), (b) and (c) represents the network in each timestamp using TGV model while networks (a'), (b') and (c') represents the network using snapshots model, network (c') being the whole network.

One of the biggest differences between the TVG and the snapshot models is how transitivity is addressed. As we saw in previous sections, transitivity is important in some network phenomena, for instance, in link prediction. However, in the TVG model, the edges are not carried on from one timestamp to another, thus transitivity can be used within a particular timestamp [81]. In the snapshot model, the edges from one timestamp are carried to the next one, so transitivity can use the whole past network to predict the future network. This can be true in several scenarios, as organizations, or social networks, where an acquaintance is carried for life even if the contact is lost. However, there are situations where the TVG model is more suitable, as is the case of the air-transport network for example where to build a route is necessary to take time into account [71].

In the TVG model, all the measures such as diameter, paths, triangles, etc., need to be rethought in the sense of being "time-respectful" [72]. Another issue in this model is how to represent the network, since one will have a different network in each timestamp and sometimes one wants to represent an edge that repeats from one

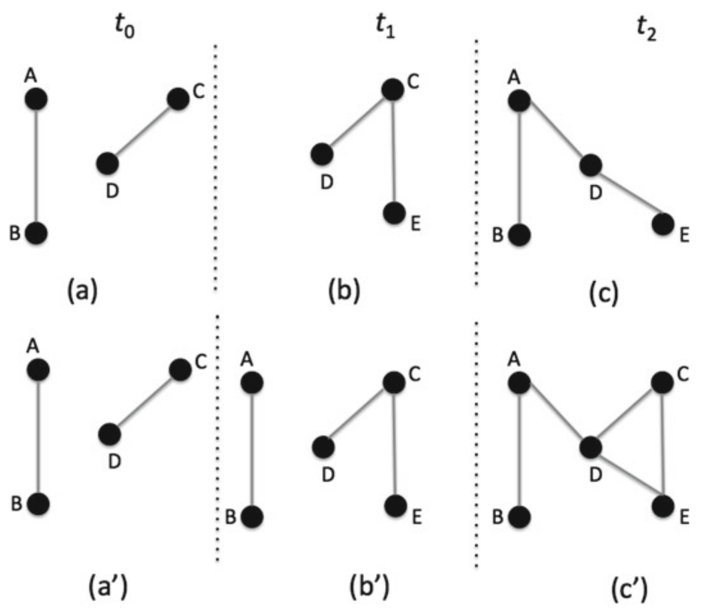

**Fig. 3** Networks in three different timestamps, using model TVG and Snapshots to represent them in each timestamp. Network (c') represents the whole network using snapshots model

timestamp to another. In [147], the authors propose a unifying model for representing TVG graphs. In this model a square adjacency matrix of $n * t$ (where $n$ is the number of nodes and $t$ is the number of timestamps) is created and follows the same principles of traditional adjacency matrix.

The third model considers the network as a graph data stream. These networks are based on transient interactions, such as email or telecommunication networks. To process a graph data stream is usually necessary real-time analytical methods [3] as the one presented in [4].

Depending on what it is the purpose of the task, the choice of the model could have a big impact. Also, graph data streams are far more challenging than graph snapshots, since the first one is in general not possible to store in memory (even in disk) for analysis.

Many interesting properties about network evolution have been studied over the years using the snapshot model. In one case, the network presents the characteristic of a shrinking diameter, i.e., in most cases the effective diameter of a network decreases over time while the network grows. Another one is the network densification which means the network becomes denser and the average degree increases as it evolves [90].

$$|\mathcal{E}_t| \propto |\mathcal{V}_t|^{(\alpha)} \tag{3}$$

As shown in Eq. 3 the number of nodes grows as function of number of edges. For $\alpha = 1$ there is a complex network that grows with a constant average degree over time, while with $\alpha = 2$ there is a much denser network. These helps not only to understand real network evolution but also to discriminate them from random graph models.

One of the most traditional ways to model evolution in network is *Tensor Factorization*. In the most simple case, a tensor is a three dimensional matrix, where in our contect usually one dimension is time [46]. However, tensor factorization present scaling issues for large graphs.

Network evolution is extremely useful for complex network analysis, especially to understand dynamic systems. How a network evolves is far from being answered simply. Link prediction is one of few techniques that incorporates the time notion in its definition, however time is still not being used as information that may improve the method's accuracy.

Other techniques, as community detection, are not directly applied in temporal networks, and some surveys, as the one presented in [49, 129] show that despite there being few methods that work in temporal networks, there's still is a large gap in techniques to solve this problem in a scalable way.

## 4 Link Prediction

Social networks are one of the most clear and well-known examples of complex networks. These applications typically need to recommend connections among users, such as the *"People you may know"* feature. Thus, the problem of how people get connected is relevant not only for social network but also for a large number of use cases such as organizations [42].

There are many reasons, sometime exogenous to the social network, why two individuals will become friends: they may happen to be geographically close if one moves to a city near the other's neighborhood, or they may attend to same party, or go to the same school, and so on. Such type of interaction can be hard to predict.

Commonly, two nodes are more likely to be connected if they are more similar. Similarity may be based only on network structure. Thus, a large number of new interactions are hinted at by the topology of the network: two individuals who are close in the network will have friends in common, and this suggests that they are more likely to become a friend in the near future.

Understanding the mechanisms by which people get connected and how networks evolve have been addressed by what is known as *the link prediction task*. Link prediction methods are based on graph snapshots as a model to support evolution. Thus, in a formal way, link prediction can be defined as: Given a snapshot of a graph $\mathcal{G}$ at time $t$, predict accurately which edges will appear in the network in time $t + 1$ [93]. Translated to the context of social networks, friends of your friends are likely also to be your friends.

In terms of network topology, transitivity means the presence of a high number of triangles in the network [146].

Most of traditional linking prediction methods are based on graph structural properties as assigning a connection value, called $score(u, w)$, to pairs of unconnected nodes $< u, w >$ based on a desired graph $\mathcal{G}$. The scores are ranked in a list in decreasing order of $score(u, w)$ and afterwards predictions are made according to this list.

Let $\tau(u) = (u \in \mathcal{V} : \exists(v, u) \in \mathcal{E})$ of node $u$ be defined as the set of nodes in $\mathcal{V}$ that are adjacent to $u$. For a node $u$, let $\tau(u)$ denote the set of neighbors of $u$ in $\mathcal{G}$. A number of link prediction approaches are based on the idea that two nodes $u$ and $w$ are more likely to form a link, in the future, if their sets of neighbors $\tau(u)$ and $\tau(w)$ have large overlap. The most direct implementation of this idea for the link-prediction problem is the common-neighbors predictor, which may be defined as follows:

$$score(u, w) = |\tau(u) \cap \tau(w)|$$

The common-neighbors predictor captures the notion that a friend may introduce two strangers who have a common friend. This introduction has the effect of "closing a triangle" in the graph and feels like a common mechanism in real life. In this sense some other measures, also neighborhood based, were propose to rank nodes that are likely to have a link in a near future.

The *Jaccard coefficient* is a similarity metric that is commonly used in information retrieval. It is used to measure the probability that both $u$ and $w$ have a feature $f$, for a randomly selected feature $f$ that either $u$ or $w$ has. In the case of networks, the feature $f$ can be a list of friends that a node has. Formally, the Jaccard predictor uses the following measure:

$$score(u, w) = |\tau(u) \cap \tau(w)|/|\tau(u) \cup \tau(w)|$$

The *Adamic/Adar predictor* [1] evaluates the degrees of the common neighbors and emphasizes the nodes that share neighbors with small degree. This is because a high degree node has a higher chance to be in the common neighborhood of other nodes. This method computes features of the nodes, and defines the similarity between two nodes to be the following:

$$score(u, w) = \sum_{(V \in \tau(u) \cap \tau(w))} \frac{1}{\log \tau(v)}$$

While the neighborhood-based measures provide a robust estimation of the likelihood of a link forming between a pair of nodes, they are not quite as effective when the number of shared neighbors between a pair of nodes is small. A particular walk-based measure that is used commonly to measure the link-prediction strength is the Katz [80] measure, which is arguably one of the best link predictors available because it has been shown to outperform many other methods as showed in [93].

$$score(u, w) = \sum_{l=1}^{\infty} \beta^l |path^l_{(u,w)}|$$

Another path measured in link prediction task is Random Walking which is a path that consisted of succession of steps chosen randomly. As showed in [94] one difficulty with all Random-walk to link prediction is their sensitive dependence to parts of the network far away from target nodes. For example, in a random walk from x to y, the walker has a certain probability to go too far away from both x and y although they may be close to each other. This may lead to a low prediction accuracy since in most real networks nodes tend to connect with the ones nearby rather than far away. Another algorithm for link prediction in social network based on Random Walks is presented in [17]. As we will present in Sect. 6.2, PRA is a random walk-based algorithm to predict relations in a Knowledge base mapped as a network.

Link prediction can be applied in other scenarios than social networks, for example predict interactions and collaborations among people in organizations can help manage companies in a productive way. The task of recommending unknowns but similar people is quite different from possible friend recommendation tasks, which focus on recommending individuals who have friends in common [67].

In [145], the author finds that the similarity between individuals' movements, their social connectedness and the strength of interactions between them are strongly correlated with each other. Thus, the authors also reports that human mobility could serve as a good predictor for the formation of new links, yielding comparable predictive power to traditional network-based measures.

Another challenge which remains largely open in link prediction methods is how to effectively combine the information from the network structure with rich node and edge attribute data. Social ties could improve link prediction metrics as the authors show in [17] where a supervised random walk that naturally combines the information from the network structure with node and edge level attributes.

There are many more metrics used to produce the score between two unconnected nodes as presented in [69, 97].

The link-prediction problem can also be related to the task of inferring missing links in complex networks: in many domains, one constructs a network and then tries to infer additional links that, while not directly visible, are likely to exist, as *Prophet* [13] presented in Sect. 6.2 [44].

This line of work differs from link prediction problem formulation in that it works with a static snapshot of a network, rather than considering network evolution. It also tends to take into account specific attributes of the nodes in the network, rather than evaluating the power of prediction methods based purely on the graph structure.

The metrics presented above for link prediction can actually be called in a more specific way as Link Existence Prediction. The link existence problem is defined as the problem of predict whether a link will or not exist in undirected networks, which is the same of link prediction, since most of studies emphasize only unweighted undirected networks.

However the link prediction problem could be extended for other problems related to discover a link, such as direction, multiplicity and weight. Follow we define each one of these problems are.

The link direction problem can be viewed as a link-prediction extension on directed networks, where the link and direction will be predicted. An example of the importance would be in a phone call network someone may want predict who-calls-whom. However, most of the work developed predicts the direction of an existent link [10].

Another possible extension is to apply the link prediction task in multiplex networks, where not only links between unconnected nodes could be predicted but also new link between node already connected (more in Sect. 8). This is a big challenge because links could have different meanings. Thus, most of work in this line needs extra information other than topology. Multiplex network commonly are seen in academic, companies and social networks where relationships among individuals can have different roles, as friend, family, co-work and so on [24].

In weighted networks, the link prediction problem can be viewed as the prediction of both the link and the weight associated with it. The most common use of weights in link prediction is to help predict the existence of links by combining them with the observed links [95]. In this case, most works adjust the metrics presented (common neighbor, Jaccard, Adamic/Adar) from unweighted to weighted networks. Yet, how weights improve the accuracy of a link prediction task and how to predict the weights together (or not) with the links has not been well studied. Example of problems that could benefit from weighted link prediction is urban or air transportation [138]. One of the few studies, and a very interesting one, on the link prediction problem in weighted networks is [96], where the authors find that weak links may play a more important role than strong links.

The works presented until now are based on homogeneous networks, meaning links are all from the same type and there are only static snapshots. When we say the network is dynamic, we are implying that new links are constantly being added to the network. Such new links may also arrive in the context of new nodes being added to the network, or they may correspond to edges between already existing nodes.

Recent works focus on heterogeneous networks where link prediction are extended for it, as co-author network [45], Location-based social networks [150], information network [83, 133, 134].

Link prediction problem becomes extremely challenging when it is addressed to dynamic massive heterogeneous network because of the challenges associated with the dynamic nature of the network, and the different types of nodes and attributes in it.

In [4] the authors present a method called “DYNALINK”, an algorithm for dynamic link inference in temporal and heterogeneous networks. The algorithm is able to construct link inference models for online and heterogeneous networks which are continuously evolving over time.

Time can have a big influence in link prediction, since old links are less important than the recent ones. For example, in a co-authorship network, new co-authors are more important than the oldest ones in terms of indicate new co-authors. The authors in [46] show that Katz metric can be improved adding a weight in the link reflecting how recent or how old the link is.

Another information that could be used to improve link prediction techniques is community structure information, as in traditional data mining, where cluster detection can be used as a pre-processing technique. In [127] the authors use the community structure to help in link prediction. The same technique is used in [4].

An interesting type of research area that may extend research for weighted link prediction is in signed networks. A signed network consists of a network composed by positive and negative links, which could mean friends and foes [84], trusted and distrusted peers [99]. A method to predict the signs of links (positive or negative) is proposed in [89, 126], however the prediction of both the existence of a link and its sign simultaneously has not been addressed yet.

## 5 Community Detection

A very important and rich research area in network theory is that of community detection. The basic idea behind community detection is the possibility to group nodes into larger groups with some criterion of similarity. The goal is to have a way to capture mesoscopic structures and in some way decrease the complexity of the original graph. This fertile research area has produced many community finding methods and algorithms [50]. Many of these methods rely in the optimization of a special function of the edges of the graph, usually called *modularity*, as we will se next.

### *5.1 Modularity Maximization*

Many real networks have some type of inner structure beyond local edges, but which at the same time is different from and contained within the complete graph. For instance, a social graph may be though locally by studying ego-networks (i.e. just the immediate connections in a node), or may be analyzed globally, may be expressing scale-free structure which is evidenced by the whole set of nodes and edges. But in a social network it is also common to find friendship groups, or work groups which are larger than ego networks and smaller than the whole graph. The aim of community algorithms is finding these kind of mesoscopic structures, usually refered to as communities or modules [27, 50, 110].

Girvan and Newman [54, 112] propose an elegant way of finding these structures. They reasoned that by analyzing edges and assuming that a set of nodes with larger number of edges between them (compared to what would be expected if the edges were randomly placed) could be thought to form a community. They defined an objective function, the *modularity* $Q$, which represents the fraction of edges inside communities minus the fraction of edges in groups *if they were randomly assigned*.

In a network with $n$ nodes, one can propose a given partition of nodes between just two communities, so as to assign $s_i = 1$ if node $i$ belongs to one community and $s_i = -1$ if belongs to the other community. One can express the modularity of such setting as follows:

$$Q = \frac{1}{4m} \sum_{ij} \left( A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j, \tag{4}$$

where $A_{ij}$ is the adjacency matrix of the network, $k_i$ is the degree of node $i$, and $m = \frac{1}{2} \sum_i k_i$ is the number of edges in the network. The second term in the parenthesis represents the expected fraction of nodes if the edges where randomly assigned. Note that this assumes a certain *null-model*, i.e. an idea of which structure would the network have is it was randomly generated. This is an important point that may influence the outcome of the final set of communities, and that needs discussion in any kind of generalization of the concept, as will be further discussed in Sect. 8.2. The above framework may be repeated iteratively to the found subgraphs to subsequently find smaller communities, taking care of modifying Eq. 4 to correctly account for all edges [112]. Whenever a proposed split gives a contribution non-positive to the total modularity, then the algorithm is carried no further. Thus, the definition of community in the Girvan-Newman method is a subgraph that is not further divisible for maximizing its modularity $Q$. The modularity method has been widely successful and even though it has been extended in many ways it continues to be the basis of the most robust methods for community finding [110].

## *5.2 The Louvain Method for Community Detection*

Another successful method for finding communities in very large networks is known as the Louvain community detection method [26], a very efficient method that has proved extremely useful in a number of big data graphs, taking a few minutes in regular hardware to compute communities for graphs of hundredths of millions of nodes and a few billion edges [16]. It was originally proposed by Lefebvre,[2] and later further developed by a group of researchers led by Blondel, all which at some point had worked at the Universit Catholique de Louvain, hence its name. The Louvain method is a modularity optimization algorithm based in the same principle that of the Girvan-Newman algorithm developed in [110, 112] (see previous Sect. 5.1) It was originally defined for weighted networks by allowing the adjacency matrix $A_{ij}$ to contain weights $w_{ij}$ (as is also the case of Eq. 4).

The algorithm is a heuristic greedy optimization method performed in two steps. First the method acts locally finding small communities. Initially every node is assigned to its own community. Next, for each node $i$, the node is assigned to the community $C$ of each of its neighbors and the corresponding change in modularity $\Delta Q$ is computed. The expression for $\Delta Q$ may be expressed as follows:

[2]https://perso.uclouvain.be/vincent.blondel/research/louvain.html.

$$\Delta Q = \left[ \frac{\sum_{in} + k_{i,in}}{2m} - \left( \frac{\sum_{tot} + k_i}{2m} \right)^2 \right] - \left[ \frac{\sum_{in}}{2m} - \left( \frac{\sum_{tot}}{2m} \right)^2 - \left( \frac{k_i}{2m} \right)^2 \right], \quad (5)$$

where $\sum_{in}$ is the sum of weights of edges in $C$, $\sum_{tot}$ is the sum of weights of edges incident to nodes in $C$, $k_i$ is the sum of weights of edges incident to node $i$, $k_{i,in}$ is the sum of weights of edges from $i$ to nodes in $C$ and finally $m$ is the total sum of weights of edges in the network. Node $i$ is moved to the community that contributes the most to the total modularity (with a breaking rule in case of ties) or stays in its original community in case no positive gain in $Q$ is possible. This procedure is repeated until no improvement in $Q$ is possible.

In the second part of the algorithm a new network is built by now grouping all nodes belonging to the communities found in the first part as new nodes in a new network. The edges between two nodes in this new network have weights equal to the sum of weights in edges between the original communities from the first stage. Self-loops appear whose weights are equal to the sum of weights of all edges within the community in the first stage of the algorithm corresponding to the new node. One this new network is created, another pass (stages one and two) is applied to the new network.

The Louvain community finding method is by no means the only community finding method for large networks [39, 92], but it has proved a nice example of an efficient and successful algorithm for networks by-product of rapidly growing— and increasingly common—big datasets.

## 6 Graphs in Big Data

Graphs and networks are a fundamental concept in the context of big data, as big data business is being driven by the possibility of quantifying relationship data. Indeed, much of the value provided by the availability of vast quantities of data resides in the ability to spot and quantify these relationships. Users that express an interest, friends that stay in touch, clients that spend in a given item, all these have in common that represent a relationship between two entities. And as we have seen in the previous sections, relationships are susceptible to be efficiently described by networks or graphs.

Google and Weibo funded their businesses linking users to topics and interest by search and advertisement. Facebook linked people and of course interests, LinkedIn connects people and professional opportunities. They all gather an enormous amount of information from their data, so the ability to extract useful insights, in the form of distilled bits of data, is crucial.

Many data wealthy businesses, different from social networks, are also starting to see the fact that they could benefit greatly by the possibilities offered by graph analytics and graph methodologies in general. For instance, there is increasing interest in graph methods applied to healthcare [118, 135].

In this section we will lay out some important examples and use cases, and we will mention the most important methods, applications and tools for the kind of networks typically found in Big Data business.

## 6.1 Graphs in the Big Data Era

**Google's PageRank** One of the most interesting examples of graph methods applied to big data was the PageRank algorithm [116], a cornerstone of Google's early success. PageRank is the ranking algorithm originally used by Google Search to present an ordered set of web pages to the user, and even though today has been considerably extended, today continues to play a relevant part in Google search results.[3] PageRank represented a success factor for Googles search engine, as it performed quite accurately for search results ranking. The ranking algorithm computes the "importance" of webpages with simple notion: based on the structure of the web page graph, use links from other pages as a proxy for the importance of the page. In essence, PageRank computes the probability that a random walk will end in a given node of the network. The algorithm is iterative, and can compute the rank of all nodes in a graph of arbitrary size. At every iteration $s$, the algorithm computes the (unnormalized) probability $PR$ for every node $i$ in the network:

$$PR^{t+1}(i) = r + (1-r)\sum_{j} \frac{PR^{t}(j)}{k_j^o}, \tag{6}$$

where $r$ is the probability of a step for the random walk and $k_j^o$ is the our-degree of node $j$. Even though there has been many variations to the algorithm since its introduction, it is undeniable that PageRank still remains important to Google's business.

**Facebook's Graph Search** Beginning 2013, Facebook introduced its Graph Search product,[4] as "a new way to navigate (the graph's) connections and make them more useful", i.e. with the aim of improving the efficient exploration of the wealth of data produced by their social network. Facebook Graph Search is designed for users to be able to make *semantic* searches for entities and their relationships. The tool uses a battery of techniques to deliver search results, such as named entity queries as well as structured queries, but it also relies heavily in graph-related quantities such as graph distance (which is fundamental to the result) in tight combination with attributes of nodes and edges such as friendship relationships, age, gender, number of friends, celebrity status, among others [130].

[3] https://www.google.com/insidesearch/howsearchworks/algorithms.html.

[4] http://newsroom.fb.com/news/2013/01/introducing-graph-search-beta/.

## *6.2 Knowledge Graphs*

Tom Gruber defines an ontology as follows: *"An ontology is a description (like a formal specification of a program) of the concepts and relationships that can formally exist for an agent or a community of agents. This definition is consistent with the usage of ontology as set of concept definitions, but more general. And it is a different sense of the word than its use in philosophy"* [64].

The RDF (https://www.w3.org/RDF/) data model, also known as RDF triples composed by subject-predicate-object, is a standard model for data interchange on the Web. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object. For example, one way to represent the notion "Messi plays soccer" in RDF is as the triple: a subject denoting "Messi", a predicate denoting "plays", and an object denoting "soccer". RDF data model is naturally suited to knowledge representation and collection of RDF statements can be represents a labeled, directed multi-graph. Mapping a Knowledge Base as graph, categories become nodes and relations are edges.

Over the last few years, many research projects focused on building large scale ontological knowledge bases (OKB) have been developed, such as Google Knowledge Graph based on Freebase [28], YAGO [131], DBpedia [15], Elementary/DeepDive [114], Walmart [43], Microsoft Satori and a continuously learning program called NELL (Never Ending Language Learner) [34]. These projects store their knowledge using what we call Knowledge Bases (KBs) with millions of facts about the world, such as information about people, places and things referred as entities.

Traditionally, an knowledge base (KB) organizes and stores knowledge in two different parts, namely: (i) an ontological model, where categories (*city, company, person*, etc.) and relations (**worksFor**(*person*, *company*), **headQuarteredIn**(*company*, *city*))) are defined, and (ii) a set of facts which are instances of categories (**city** (*New York*), **company**(*Disney*), **person**(*Walt Disney*) and relations.

Despite their size, KB are far from complete. As showed in [44] 71% of people in Freebase have no known place of birth, and 75% have no known nationality. Furthermore, coverage for less common relations can be even lower.

Therefore, a new approach is necessary to further scale up knowledge base construction. Such an approach should automatically extract facts from the whole Web, to augment the knowledge we collect from human input and structured data sources. Unfortunately, standard methods for this task often produce very noisy, unreliable facts. To alleviate the amount of noise in the automatically extracted data, the new approach should automatically leverage already-cataloged knowledge to build prior models of fact correctness.

One of the biggest problems in knowledge bases is extending it by inferring new relations. The ability to infer new knowledge may be straightforward for humans, but is tipically very hard be done automatically by a machine, as learning programs populates the KB from corpora or the Web, which may be a difficult task.

Mapping a KB as a network allows us to apply graph mining techniques to infer new relations. Thus, one of task that are mainly used is link prediction, which is applied to find implicit information so as to populate the KB. There are several projects that use graph mining, such as *Prophet* [13] in NELL, or Knowledge Vault [44] from Google. NELL also uses Random Walks to infer relations [86] and PageRank for search [144].

*Prophet* [13] was created to be one of NELL's components, to apply link-prediction techniques into NELL's KB maped to a graph. It executes a link-prediction task using a metric called extra-neighbors to extend NELL's ontology by finding new possible relations and also it finds new instances of these relations and some possible misplaced facts present on the KB (Fig. 4).

Knowledge Vault used PRA [86] to extend their KB that is based on Freebase. Similar to distant supervision, PRA begins with an instance of a relation such as (*Basketball*, *MichaelRedd*), i.e. a pair of entities, then it performs a random walk on the graph, starting at all the subject (source) nodes. Paths that reach the object (target) nodes are considered successful. For example [44], PRA learns that pairs $(X, Y)$ which are connected by a *marriedTo* edge often also have a path of the form $X \xrightarrow{childOf} Z \xleftarrow{childOf} Y$, since if two people share a common child, they are likely to be married. The paths that PRA learns can be interpreted as rules.

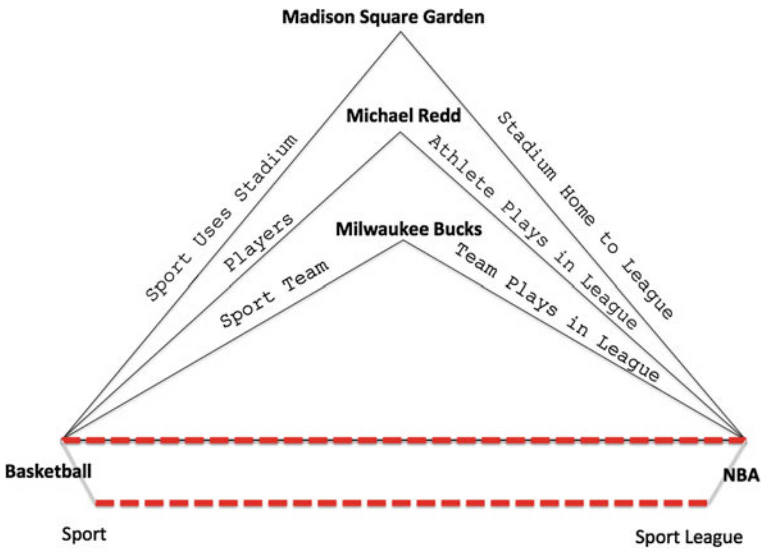

**Fig. 4** An example of rule (*Sport*, *SportLeague*) and instance (*Basketball*, *NBA*) infer by *Prophet* based on three independent paths (Madison Square Garden, Michael Redd, Milwaukee Bucks)

### *6.3 Graph Sampling*

One way to work with massive amounts of data is sampling. Data sampling has been applied over years in large data sets to extract representative small portions from data allowing us to apply data mining techniques as clustering and classification algorithms, usually computational expensive in the complete dataset. However, in the context of complex networks, the question of how to extract a representative small network from the original dataset is nontrivial. Also, one issue with graph sampling is if one should focus on nodes, edges or both. Additionally, in non-linked data, random sampling usually performs reasonably, however in complex network random sampling, be it in nodes or edges, the process usually produces a disconnected graph [88].

It can be quite difficult to find a method to reduce the size of network and keep all the important measures, useful for graph mining techniques as cluster and link prediction [143].

Another area where graph sampling techniques are widely applied is in crawling. Collecting data for analysis is a very important task and how these data are collected is extremely important to the post analyze that will be done. The crawling process of a graph starts from selecting one (or multiple) node(s) called seed(s). After a node has been visited, the edges incident in this node are known and the next node can be chosen. The policy of choosing the next node depends on the design of the crawling. Among the possible policies are: Breadth-First, Edge based [5], Random Walking [87, 87], Weighted edges [85], degree, triangles [119], among many other [9, 153]. All the nodes and edges measures can be used for crawling and, of course, for sampling, since we can extend the idea of selecting data from the original graph (already collected) instead of collecting data [73].

If we add temporal evolution in the network, sampling becomes an even harder problem, since now we have a timestamp as an extra information associated with the edges. The same happens when we are dealing with multiplex networks that have more than one link between nodes and this link could have both a weight and a timestamp. Sampling is an important question in big data networks, but is still a very open question without a closed solution.

### *6.4 Graph Analytics Tools*

In this section we will briefly mention some graph tools and systems for modeling and analytics especially suited for very large networks, i.e. with tens to hundredths of millions of nodes and up to hundredths of billions of links. These systems are fundamental to efficiently support Big Data applications, such as Natural Language Processing tasks or targeted advertising.

Apache Giraph [12] is an open source distributed system for large scale graph processing, based on Google's proprietary Pregel [98]. It is an iterative graph system designed for high scalability (based on Apache Hadoop's MapReduce implementation), and it extends Pregel with a number of features such as edge-oriented output, master computation, sharded aggregators among others. Giraph is used by Facebook to analyze its vast social graph, and is able to process trillions of edges [37], and new, faster extensions are being developed based on it [137, 139] as well as dedicated machine learning libraries [59].

Another graph analytics example is PowerGraph [58], a distributed graph placement and representation that exploits a know feature of social networks: their power-law degree probability distribution. PowerGraph was shown to process PageRank and other tasks such as LDA in data from the Twitter social network, containing 41 million nodes and 1.5 billion edges. GraphLab, a CMU initiative, and afterwards GraphLab Create [62], an open source framework for distributed, high-performance computation over graphs, stemmed originally from PowerGraph.

Project Pegasus [120] is another CMU-based open-source, big graph-mining system designed for high scalability. In [78], the authors make an interesting comparison between Pegasus, Pregel, GraphLab and Microsoft's contribution, names Trinity at the time (now GraphEngine) [123, 124]. They compare system performance in a number of graph-oriented tasks over two big datasets, a snapshot of the World Wide Web (2002), crawled by Yahoo! with 1.4 billion nodes (web pages) and 6.6 billion links, and a Twitter who-follows-whom graph (2009), containing 63 million nodes (users) and 1.8 billion links.

Another Big Graph system is Twitter's Cassovary [35], a processing library for the Java Virtual Machine, which is now open source. Cassovary, written in Scala, is designed to handle large graphs such as Twitter's and also to be space-efficient. In [66], the authors describe some variants of recommender systems implemented in Cassovary, and a very interesting take on the architecture design, as the entire graph is put in a single server for optimization purposes, contrary to the mainstream tendency of distributed architectures. There are several other initiatives for big graph analytics and processing systems, ranging from industrial tools such as IBM System G [75], DataStax/Aurelius Faunus [48] or Teradata's SQL-GR Graph Analytics engine [14], to less production-oriented such as Microsoft's GraphEngine [61] and even more academic research-oriented systems such as the Stanford Network Analysis Project (SNAP) [125], Galois [51], from University of Texas, GUESS [65] as well as iGraph [76] (in its three flavors, R, C/C++ and Python), Gephi [21, 53] and Python-based NetworkX [70], among several others.

It is also worth mentioning a different but important class of systems, graph databases. Graph databases are generally not relational databases and exploit graph structure to optimize searches and semantic queries, most commonly by keeping track of relationships among nodes, among other things. Even though their purpose may be diverse (some are operational, while others are for analytics or development, and so on), we will focus on these differences and instead we will just present some of the most known solutions as of 2016. Some examples include Titan [141] (which Amazon integrates through their NoSQL database Amazon DynamoDB), Neo4j [108], OrientDB [115], Sparksee [128], IBM Graph [74], and GraphX [63] (Apache Spark's API for graphs and graph-parallel computation), among several others. Some of these have in common that they use the Apache TinkerPop graph computing framework [140], in particular, they are able to process instructions from the Gremlin traversal language [122], a cross-platform virtual machine and language that supports imperative and declarative querying for graph databases and graph analytical engines.

## 7 Weighted Networks

Single networks represent their connections as binary entities, i.e. an edge is present or not. Usually, edges do not provide more information than if they are present or not. However, links between nodes may have some describing attribute, reflecting their intensity, capacity, duration, intimacy or exchange of services [19, 60], which may be encoded in some variable usually known referred to as *weight* of the link.

The study of weighted networks has not been thorough in the last decade. Some initial work was done in [18, 149]. Many methods developed for single networks are not trivial to extend for weighted networks. A weighted network can be treated as a multiplex network where the weight becomes the number of edges between two verticals, as shown in Fig. 5.

The weights in the edges of a network help model and define week and strong ties. One way to measure this is the so-called *strength* of a node $V_i$ (Eq. 7) defined as the sum of all weights of neighbors of node $v_i$. The strength of a node integrates the information both with its connectivity and the importance of the weights of its links, and can be considered as the natural generalization of the connectivity [102].

$$S_{v_i} = \sum_{v_j \in V(V_i)} w_{V_i,V_j} \tag{7}$$

In [6] the authors present *OddBall*, a fast, unsupervised method to detect abnormal nodes in weighted graphs. They also show that the total weight $W_{v_i}$ and the number of edges $\mathcal{E}_{v_i}$ of $\mathcal{G}$ follow a power-law probability distribution. In another interesting work [8], the authors study the weights associate to the reciprocity in mobile phone calls and describe several patterns found. In the particular case of link prediction, models using

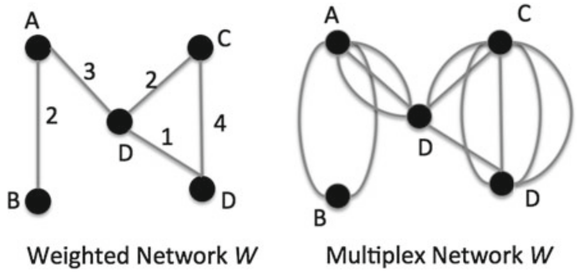

**Fig. 5** Weighted network W and multiplex network W

weighted networks are usually simpler than models that use the multiplex network framework, since it has been shown that weights can also help in the algorithm accuracy to predict new edges [95].

## 8 Extending Graph Models: Multilayer Networks

There have been constant research efforts to extend network models to describe more adequately complex real life networks. As we will see, many of this efforts involve generalizing the definition of the basic blocks that constitute a graph. In this section we will briefly describe one of the most interesting attempts, which is to consider graphs with several types of edges. As we will see this allows for very rich networks models.

### *8.1 The Layered Point of View: Multilayer Networks*

As we have mentioned Sect. 1, graphs are an extraordinarily useful representation of real networks, i.e. any collection of entities with a given relationship between each other. In some practical settings, it is interesting to model a given type of relationship, and the definition of edge is clear. This is the case, for instance, of a co-authorship network, where we are interested in which authors published together, so the edge definition is simply if there exists any publication with both authors' names.

On the other hand, other settings are much more complex, given that the two entities may have more than one type of relevant relationship. For instance, if we are interested in modeling a social network, it could be useful to distinguish among family and work relationships. One way of dealing with this situation is to count everything as a link and try to keep the nature of the edge somehow, for instance as weights in the link. In this way, we can use these weights to restrict or filter some calculations.

Another equivalent approach is to think about these multiple types of relationships as defining different networks or subnetwork, and then connect these accordingly, as they are defined by the same set of nodes. By this point of view, each network is characterized as a layer, so a network may be represented by a number of interconnected layers [82].

This way of framing the structure of interactions among nodes has gained a lot of track in particular in the community of complex networks, which has termed the concept generally as *multiplex networks* [56, 57, 107] or *networks of networks* [40, 52].

One example of this concept in very practical scenario is the case of overlay networks [11]. In the field of engineering and computer science, and specifically in the context of network virtualization, an overlay network is a virtual network created on top of an existing "substrate" network, where only some of its nodes and links are used for the virtual network. An overlay network can be used to share infrastructure and simplify topology, defining a network with different properties than the underlying network, in terms of routing, security, caching, or other network functionalities. Thus, an overlay network can be though of a multiplex network where certain nodes have special types of *virtual* links [38]. The Internet itself started out as an overlay network over the telephone network, and currently many services such as VoIP applications are also defined over the Internet, and so are also work on top of overlays.

Indeed, over the last 40 years many fields of research have in some way or another turned their attention to the same problem under different names: multiplex networks, networks of networks, multidimensional networks, multislice networks, etc. In [82], the authors make a thorough review of different approaches and a provide a complete historic perspective, as well as the different technical aspects of the state of the art. In the following sections we will address some points directly related to the Big Data scenario.

## *8.2 Models, Methodologies and Other Tools*

To get some intuition, we will consider that a multilayer network is a set of nodes belonging to a set of layers, and where any node in any layer may be connected by an edge [82]. (There are even more general and elaborate models of multilayer networks [41], but for our purposes we will set with this description.)

In Fig. 6 we show a basic representation of such a construction, with only two layers and a few nodes. Not every layer must have the same nodes, and edges may connect any pair of nodes between layers. Layers may have any meaning, for instance they could define a type od interaction, so edges have a particular meaning according to the layer, or they may signify a particular time frame, in fact describing the dynamics of the network. These and many other choices fit in this very general definition. Additional constraints may also be put in place, for instance, considering that the set

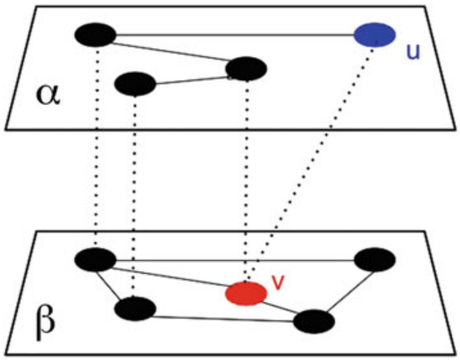

**Fig. 6** Multilayer network representation, showing two layers and two single-layer networks, with some edges connecting nodes between layers

of nodes is fixed for all layers (node alignment), or that edges across layers can only connect the same node in each layer.

**Generalization of single-layer algorithms** Naturally, research efforts in multilayer networks have focused on extending tools and methods successfully used to study and diagnose single-layer networks. New definitions have been developed for generalizing basic concepts such as node degree, neighborhoods, clustering coefficients, etc. as well as methods and models such as community detection or diffusion dynamics. We explore a few of these ideas next.

In the same way that single-layer networks may be represented by and adjacency matrix, a natural generalization for multi-layer networks is by describing them through tensor representations [41]. Thus, one way to represent a multi-layer network is by providing the tensor $\mathcal{A}_{uv\alpha\beta}$, whose elements take value 1 if node $u$ of layer $\alpha$ is connected with node $v$ of layer $\beta$, and 0 otherwise. This generalization corresponds to multi-layer models with node alignment, but it is possible to go beyond to the mode general case where this constraint is relaxed.

One of the most basic concepts in networks is that of degree. As we have seen, this is simply the number of edges incident in a given node. Is it possible to generalize the degree by having into account the weight of the node (weight degree or strength) as well as discriminating by edge direction (*in* or *out*), for directed networks.

Generally speaking, a general procedure to transform a multi-layer network into a single-layer network, so as to apply known (single-layer) techniques is by aggregating layers in some way. This *network aggregation* may be done in several ways according to the study being made.

In the case of the degree, one way to extend this concept is precisely by aggregating through all the layers, i.e. summing all (or some) type of edges to get a value for the degree, e.g. $k_u = \sum_{v\alpha\beta} \mathcal{A}_{uv\alpha\beta}$. There are of course other approaches, for instance, considering thresholds to the quantity of edges that contribute to the degree, and also different normalizing methods [41]. The authors in [30–32] describe various definitions of degree as well as diverse methods to compute degree centrality, which we will comment in detail later, and other related concepts such as neighborhood.

As we have seen in Sect. 2 other central concept in network theory is that of clustering coefficients, and, generally speaking, the notion of transitivity [111]. The extension of these ideas to multi-layer networks has been quite challenging as there are inherent ambiguities in the different possible definitions of these quantity. One well known interpretation of the clustering coefficient for single-layer networks is that is equal to the ratio of closed triples to connected triples. But the definition of triple in multi-layer networks requires some care, as a triple may be defined in multiple ways in this case, depending on the set of constraints that may be imposed regarding the layers (i.e., the type of edges) allowed for consideration in the definition. Great effort has been made in defining a suitable generalization of the clustering coefficient (see [30, 32, 82] among many others). One important conclusion that one may reach from these efforts is that an adequate definition of the clustering coefficient is dependant on the domain on is trying to describe, as the notion of neighborhood or path may differ for different type of networks.

A related and also central concept in single-layer networks is that of communities or modules [111], i.e. the fact that a set of nodes within the network may be more related between them than with other nodes not in the set (or rather, compared to what is expected in a randomly connected network). Despite the numerous research efforts involving some variant of multi-layer networks, few community detection algorithms have been put forward compared to the huge number of such algorithms for single-layer networks. One again, the additional degrees of freedom introduced by these models makes previous definitions ambiguous and non-trivial to generalize.

One important concept in the are of community detection is the choice of a null-model to which compare the network in order to precisely quantify the meaning of "expected in a randomly connected network". The authors in [20, 107] study this problem in the case of multislice modularity, which has been used particularly for the case of temporal networks [106] (i.e. layers representing time).

Centrality measures seek to inform about the relative importance of a given element in the network [111]. There are several versions of centrality measures for different contexts. Central of network science, many of these measures have already been generalized for the case of multi-layer centrality. For instance, PageRank centrality (introduced in Sect. 6.1) has been extended for the more general case through random walker able to also traverse across layers, with suitable definitions of rank also for the layers [113], or differentiating probabilities inside layers from those across layers [152], or with the introduction of biased walkers [68].

## 8.3 *Theoretical Models, Empirical Applications and Other Examples*

The possibilities open by the multi-layer formalism are indeed enormous [82]. Many networks from different fields of study fit very well with this formalism.

For instance, Morris et al. [105] develop a two layer theoretical model for transportation in spacial networks and show that different transport regimes may be found. In turn, Cardillo et al. [33] apply the multiplex theoretical framework to the European Air Transportation Network, and they claim that the topology of each layer affects the emergence of structural properties in the aggregate network.

In [106, 107], the authors show how a multislice framework may help understand the communities developed as a function of time in a dataset of the U.S. Senate roll call voting, from 1798 to 2008.

On a more theoretical note, the authors in [56] explore the diffusion properties in the context of (node-aligned) multiplex networks, and explain the dynamics of the diffusion process through the mathematical properties of the system, more specifically, the spectrum of eigenvalues of a matrix built with the Laplacians of each layer.

In [25], Bianconi proposed a statistical mechanics framework to study mutliplex networks, on the premise that a given link between nodes in a layer may be highly correlated to another link between in another layer. The author develops entropy expressions for the multiplex system that may be useful tools for inference problems.

Another interesting application of multiplex networks is in the field of evolutionary game-theory, in particular the study of cooperation in the context of interacting agents. The authors in [57] explore the Prisoner's Dilemma (PD) game in a multiplex setting with random (Erdös-Rényi) networks in each layer, where layers are coupled via the payoff parameter of the PD game, which is the sum of the payoffs in all layers. They show that the resilience of the fraction of agents which stay in the cooperation state is boosted by the introduction of interaction between layers in the system, an important result indicating that the multiplex character of, for instance, social networks, could influence favorably for the emergence of stable of cooperation.

## 9 Open Challenges

In this chapter, we have shown the relevance and ubiquity of graphs and networks in big data systems and applications. Graph analytics and mining provide value to these big data systems, and the field is really starting to emerge, driven by ever new technologies and applications [55, 148]. Naturally, this remarkable growth pushes the state-of-the-art of current systems until limitations are reached. Next we mention some of the future challenges of the area and some efforts to push the capabilities of today's systems to meet tomorrow's needs.

**Streaming Analytics** The data stream model may be useful in many situations where data is constantly produced. Many non-trivial challenges arise when dealing with graph data streams, such as the trade-off between data size and accuracy in the computation of graph measured destined to summarize the data. In this way, streaming methologies have become central in many applications where the real-time nature of information flow is relevant or needed [55, 151], as well as in other issues of a more technological origin such as how to compute graph quantities in a distributed or parallel setting [101]. Usually, graph streaming is done typically by providing a stream of edge information to add or subtract, and make some computation in the resulting graph [77]. The challenge is to maintain a precise or at least approximate picture of the network or associated summary variables. A research field strongly connected with this type of issues is graph visualization. For example, in streaming visualization of power-grid networks, the accurate and quick description of the network may be crucial at times of failure where responsibles have minutes or even seconds to respond [148]. In general, there is currently a research effort in this field is focusing in algorithms for directed edges, which may be more general as most practical examples are directed [101].

**Representation learning for networks** Graphs are intrinsically a non-linear combination of data, not always readily summarizable in every aspect by a few parameters. One interesting research area in network theory is how to decrease the dimensionality of a given graph, while at the same time preserving useful characteristics or informative traits we are in the first place interested in. This dimension reduction techniques play an important part in prediction tasks, such as link prediction (see Sect. 4) or prediction of node attributes such as user interests or functional labels in biological networks.

On the other hand, a manifest characteristic in a graph dataset is its sparseness. Depending on the task or application one is trying to accomplish, this sparseness may be of help or, on the contrary, become a computational burden, leading to inefficient algorithms. For instance, statistical learning in a graph may be hindered by the inherent sparsity of some graphs, especially as they turn into very large graphs, typical of big data domains. Traditional dimensionality reduction methods, such as Principal Component Analysis or Multi-Dimensional Scaling, have been studied widely in the literature [22]. But these techniques usually involve finding eigenvalues of the adjacency matrix (or another equivalent matrix), which normally does not scale well for large graphs.

A way to go around this problem is to find *latent representations* of the network, which encode what is interesting or useful from the graph but are defined in a space with much lower dimensionality [23]. Versions of such latent representations based in deep neural networks have been quite successful in the context of Natural Language Processing [103] and have lead the way for applications in many research areas, including graph analytics and social networks.

In [121], the authors apply this ideas to social networks with the aim of encode social relationships in a continuous vector space, which are then easily exploited by statistical models. The authors propose a random walk algorithm which is used to capture neighborhood similarity, in the sense that nodes with similar neighborhoods will finally present similar representation in vector space. In [136], the authors propose finding latent representations as an optimization problem, by carefully devicing an objective function to capture both local and global characteristics of network structure.

With these type of methods, the learned representations may be used to perform classification tasks or link prediction tasks, which proves to be much more efficient due to the decreased dimensionality. However there is still much research needed as, to date, representation learning have been mainly proposed as heuristic methods aiming to automatically capture useful features from networks, with not much study as to the general validity of the results, both in types of networks and in types of features learned.

***All-pairs*** **computation** On a related note, graphs have the inevitable characteristic of scaling as $O(n^2)$ when taking into account all pairs of nodes. Due to this fact, some graph analytics methods in big data recurrently find limitations whenever computations involve computing over all pairs of nodes, e.g. all-pair shortest paths, or any other type of similar quantity. This is the case, for instance, discussed in Sect. 4 for the Katz measure, which is based in a sequence of matrix-matrix computations [80, 132]. The usual way of bypassing this scaling limitation is to provide estimates or approximations, which depending on the problem and the size of the data may no longer be a satisfactory solution.

## 10 Conclusions

Many data sources in big data scenarios represent *relationship data*. Being social data, Internet-of-things data, or even semi-structured data such as Twitter posts or document corpora, relationships among entities are usually present, thus making it viable to represent the data as graphs. The graph representation will not always be necessary, but most of the times will be convenient and useful.

In this chapter we have covered many aspects of graph algorithms and network analysis which are important, especially in the case of very large graphs. We have shown that there is an increasing confluence of efforts towards the practical use of graph analytics in the context of big data. One the one hand, the research community continually provides powerful graph algorithms and methods, some of outstanding business success such as PageRank, or the link prediction algorithms in Facebook to grow their social base. On the other hand, a relentless developer community yields ever more powerful software, libraries, as well as commercial and open-source tools, focusing in the implementation of improved graph algorithms and in providing more efficient ways to capture, handle and process large quantities of data as graph information.

Graphs and networks are at the core of the big data era. The advent of big data tools and systems has changed radically the access to real data, and both businesses and the research community have benefited from this by leveraging graph analytics and methods to produce new sources of wealth and information.

## References

1. L.A. Adamic, E. Adar, Friends and neighbors on the web. Soc. Network. **25**(3), 211–230 (2003)
2. L.A. Adamic, B.A. Huberman A. Barabási, R. Albert, H. Jeong, G. Bianconi, Power-law distribution of the world wide web. Science **287**(5461):2115a+ (2000)
3. C. Aggarwal, K. Subbian, Evolutionary network analysis: a survey. ACM Comput. Surv. **47**(1), 10:1–10:36 (2014)
4. C. Aggarwal, Y. Xie, P.S. Yu, *On Dynamic Link Inference in Heterogeneous Networks, chap. 35*, pp. 415–426
5. N. Ahmed, J. Neville, R.R. Kompella, Network sampling via edge-based node selection with graph induction (2011)
6. L. Akoglu, M. McGlohon, C. Faloutsos, Oddball: spotting anomalies in weighted graphs, in *Advances in Knowledge Discovery and Data Mining*, ed. by M.J. Zaki, J.X. Yu, B. Ravindran, V. Pudi (Springer, Heidelberg, 2010), pp. 410–421

7. L. Akoglu, H. Tong, D. Koutra, Graph based anomaly detection and description: a survey. Data Min. Knowl. Discov. **29**(3), 626–688 (2015). May
8. L. Akoglu, P.O.S. Vaz de Melo, C. Faloutsos, Quantifying reciprocity in large weighted communication networks, in *Proceedings of the 16th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II, PAKDD'12* (Springer, Heidelberg, 2012), pp. 85–96
9. M. Al Hasan, M.J. Zaki, Output space sampling for graph patterns. Proc. VLDB Endow. **2**(1), 730–741 (2009)
10. U. Alon, Network motifs: theory and experimental approaches. Nat. Rev. Genet. **8**(6), 450–461 (2007)
11. D. Andersen, H. Balakrishnan, F. Kaashoek, R. Morris, Resilient overlay networks (ACM, 2001)
12. Apache Giraph, an iterative graph processing system. http://giraph.apache.org/. Accessed 10 March 2016
13. A.P. Appel, E.R.H. Junior, Prophet – a link-predictor to learn new rules on nell, in *2011 IEEE 11th International Conference on Data Mining Workshops (ICDMW)*, Dec 2011, pp. 917–924
14. Aster SQL-GR Big Data Parallel Graph Analytics. http://www.teradata.com/SQL-GR-Engine/. Accessed 10 March 2016
15. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, DBpedia: a nucleus for a web of open data, in *The Semantic Web: 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, 11–15 November 2007. Proceedings* (Springer, Heidelberg, 2007), pp. 722–735
16. T. Aynaud, V.D. Blondel, J.-L. Guillaume, R.Lambiotte, Multilevel local optimization of modularity, in *Graph Partitioning* (2013), pp. 315–345
17. L. Backstrom, J. Leskovec, Supervised random walks: predicting and recommending links in social networks, in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM'11* (ACM, New York, 2011), pp. 635–644
18. A. Barrat, M. Barthélemy, R. Pastor-Satorras, A. Vespignani, The architecture of complex weighted networks. Proc. National Acad. Sci. **101**, 3747–3752 (2004)
19. M. Barthélemy, A. Barrat, R. Pastor-Satorras, A. Vespignani, Characterization and modeling of weighted networks. Physica A **346**, 34–43 (2005)
20. D.S. Bassett, M.A. Porter, N.F. Wymbs, S.T. Grafton, J.M. Carlson, P.J. Mucha, Robust detection of dynamic community structure in networks. J. Nonlinear Sci. **23**(1), 013142 (2013)
21. M. Bastian, S. Heymann, M. Jacomy et al., Gephi: an open source software for exploring and manipulating networks. ICWSM **8**, 361–362 (2009)
22. M. Belkin, P. Niyogi, Laplacian eigenmaps and spectral techniques for embedding and clustering. NIPS **14**, 585–591 (2001)
23. Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
24. M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, D. Pedreschi, Multidimensional networks: foundations of structural analysis. World Wide Web **16**(5), 567–593 (2012)
25. G. Bianconi, Statistical mechanics of multiplex networks: entropy and overlap. Phys. Rev. E **87**(6), 062806 (2013)
26. V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks. J. Stat. Mech. Theory Experiment **2008**(10), P10008 (2008)
27. S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, D.-U. Hwang, Complex networks: structure and dynamics. Phys. Rep. **424**(4), 175–308 (2006)
28. K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in *Proceedings of SIGMOD* (2008)
29. D. Braha, Y. Bar-Yam, Time-dependent complex networks: dynamic centrality, dynamic motifs, and cycles of social interactions, in *Adaptive Networks: Theory, Models and Applications* (Springer, Heidelberg, 2009), pp. 39–50
30. P. Bródka, K. Musial, P. Kazienko, A method for group extraction in complex social networks, in *Knowledge Management, Information Systems, E-Learning, and Sustainability Research*,

ed. by M.D. Lytras, P. Ordonez De Pablos, A. Ziderman, A. Roulstone, H. Maurer, J.B. Imber (Springer, Heidelberg, 2010), pp. 238–247

31. P. Bródka, K. Skibicki, P. Kazienko, K. Musiał, A degree centrality in multi-layered social network, in *2011 International Conference on Computational Aspects of Social Networks (CASoN)* (IEEE, 2011), pp. 237–242
32. P. Bródka, P. Kazienko, K. Musiał, K. Skibicki, Analysis of neighbourhoods in multi-layered dynamic social networks. Int. J. Comput. Intell. Syst. **5**(3), 582–596 (2012)
33. A. Cardillo, J.Gómez-Gardeñes, M. Zanin, M. Romance, D. Papo, F. del Pozo, S. Boccaletti, Emergence of network features from multiplexity. Sci. Rep. **3** (2013)
34. A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., T.M. Mitchell, Toward an architecture for never-ending language learning, in *Proceedings of AAAI* (2010)
35. Cassovary. https://github.com/twitter/cassovary. Accessed 10 March 2016
36. D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, C. Faloutsos, Epidemic thresholds in real networks. ACM Trans. Inf. Syst. Secur. **10**(4), 1–26 (2008)
37. A. Ching, S. Edunov, M. Kabiljo, D. Logothetis, S. Muthukrishnan, One trillion edges: graph processing at facebook-scale. Proc. VLDB Endow. **8**(12), 1804–1815 (2015)
38. N.M.K. Chowdhury, R. Boutaba, A survey of network virtualization. Comput. Network. **54**(5), 862–876 (2010)
39. A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks. Phys. Rev. E **70**(6), 066111 (2004)
40. G. D'Agostino, A. Scala, *Networks of Networks: The Last Frontier of Complexity*, vol. 340 (Springer, Heidelberg, 2014)
41. M. De Domenico, A. Solé-Ribalta, E. Cozzo, M. Kivelä, Y. Moreno, M.A. Porter, S. Gómez, A. Arenas, Mathematical formulation of multilayer networks. Phys. Rev. X **3**(4), 041022 (2013)
42. R.A. de Paula, A.P. Appel, C.S. Pinhanez, V.F. Cavalcante, C.S. Andrade, Using social analytics for studying work-networks: a novel, initial approach, in *2012 Brazilian Symposium on Collaborative Systems (SBSC)*, Oct 2012, pp. 146–153
43. O. Deshpande, D.S. Lamba, M. Tourn, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, A. Doan, Building, maintaining, and using knowledge bases: a report from the trenches, in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, SIGMOD'13* (ACM, New York, 2013), pp. 1209–1220
44. X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, W. Zhang, Knowledge vault: a web-scale approach to probabilistic knowledge fusion, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'14* (ACM, New York, 2014), pp. 601–610
45. Y. Dong, J. Tang, S. Wu, J. Tian, N.V. Chawla, J. Rao, H. Cao, Link prediction and recommendation across heterogeneous social networks, in *Proceedings of the 2012 IEEE 12th International Conference on Data Mining, ICDM'12* (IEEE Computer Society, Washington, DC, 2012), pp. 181–190
46. D.M. Dunlavy, T.G. Kolda, E. Acar, Temporal link prediction using matrix and tensor factorizations. ACM Trans. Knowl. Discov. Data **5**(2), 10:1–10:27 (2011)
47. M. Faloutsos, P. Faloutsos, C. Faloutsos, On power-law relationships of the internet topology, in *ACM SIGCOMM Computer Communication Review*, vol. 29 (ACM, 1999), pp. 251–262
48. Faunus: Graph Analytics Engine. http://thinkaurelius.github.io/faunus/. Accessed 10 March 2016
49. S. Fortunato, Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
50. S. Fortunato, C. Castellano, Community structure in graphs, in *Computational Complexity*, ed. by R.A. Meyers (Springer, Heidelberg, 2012), pp. 490–512
51. Galois: The University of Texas at Austin. http://iss.ices.utexas.edu/?p=projects/galois. Accessed 10 March 2016
52. J. Gao, S.V. Buldyrev, S. Havlin, H.E. Stanley, Robustness of a network of networks. Phys. Rev. Lett. **107**(19), 195701 (2011)
53. Gephi: The Open Graph Viz Platform. https://gephi.org/. Accessed 10 March 2016

54. M. Girvan, M.E. Newman, Community structure in social and biological networks. Proc. National Acad. Sci. **99**(12), 7821–7826 (2002)
55. D.F. Gleich, M.W. Mahoney, Mining large graphs, in *Handbook of Big Data* (2016), p. 191
56. S. Gomez, A. Diaz-Guilera, J. Gomez-Gardeñes, C.J. Perez-Vicente, Y. Moreno, A. Arenas, Diffusion dynamics on multiplex networks. Phys. Rev. Lett. **110**(2), 028701 (2013)
57. J. Gómez-Gardeñes, I. Reinares, A. Arenas, L.M. Floría, Evolution of cooperation in multiplex networks. Sci. Rep. **2** (2012)
58. J.E. Gonzalez, Y. Low, H. Gu, D. Bickson, C. Guestrin, Powergraph: distributed graph-parallel computation on natural graphs, in *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)* (2012), pp. 17–30
59. Grafos.ML - Empowering Giraph. http://grafos.ml/index.html. Accessed 10 March 2016
60. M. Granovetter, The strength of weak ties. Am. J. Sociol. **78**(6), 1360–1380 (1973)
61. GraphEngine: serving big graphs in real-time. http://www.graphengine.io/. Accessed 10 March 2016
62. GraphLab Create - an extensible machine learning framework. https://dato.com/products/create/. Accessed 10 March 2016
63. GraphX: Apache Spark's API for graphs and graph-parallel computation. http://spark.apache.org/graphx/. Accessed 10 March 2016
64. T. Gruber, What is an ontology (1993). WWW Site http://www-ksl.stanford.edu/kst/whatis-an-ontology.html. Accessed 07 Sep 2004
65. GUESS: The graph exploration system. http://graphexploration.cond.org. Accessed 10 March 2016
66. P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, R. Zadeh, Wtf: the who to follow service at twitter, in *Proceedings of the 22nd International Conference on World Wide Web Conferences Steering Committee* (2013), pp. 505–514
67. I. Guy, S. Ur, I. Ronen, A. Perer, M. Jacovi, Do you want to know?: recommending strangers in the enterprise, in *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work, CSCW'11* (ACM, New York, 2011), pp. 285–294
68. A. Halu, R.J. Mondragón, P. Panzarasa, G. Bianconi, Multiplex pagerank. PloS One **8**(10), e78293 (2013)
69. M.A. Hasan, M.J. Zaki, A survey of link prediction in social networks, in *Social Network Data Analytics*, ed. by C.C. Aggarwal (Springer, Boston, 2011), pp. 243–275
70. High-productivity software for complex networks. https://networkx.github.io/. Accessed 10 March 2016
71. P. Holme, C. Edling, F. Liljeros, Structure and time-evolution of an internet dating community. Soc. NetworK. **26**, 155 (2004)
72. P. Holme, J. Saramäki, Temporal networks. Phys. Rep. **519**(3), 97–125 (2012)
73. P. Hu, W.C. Lau, A survey and taxonomy of graph sampling. arXiv preprint arXiv:1308.5865 (2013)
74. IBM Graph: easy-to-use, fully-managed graph database service. https://new-console.ng.bluemix.net/catalog/services/ibm-graph/. Accessed 10 March 2016
75. IBM System G. http://systemg.research.ibm.com/. Accessed 10 March 2016
76. igraph: The network analysis package. http://igraph.org/. Accessed 10 March 2016
77. M. Jha, C. Seshadhri, A. Pinar, A space efficient streaming algorithm for triangle counting using the birthday paradox, in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2013), pp. 589–597
78. U. Kang, C. Faloutsos, Big graph mining: algorithms and discoveries. ACM SIGKDD Explor. Newslett. **14**(2), 29–36 (2013)
79. U. Kang, C.E. Tsourakakis, A.P. Appel, C. Faloutsos, J. Leskovec, Hadi: mining radii of large graphs. ACM Trans. Knowl. Discov. Data (TKDD) **5**(2), 8 (2011)
80. L. Katz, A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953). March
81. D. Kempe, J. Kleinberg, A. Kumar, Connectivity and inference problems for temporal networks, in *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing, STOC'00* (ACM, New York, 2000), pp. 504–513

82. M. Kivelä, A. Arenas, M. Barthelemy, J.P. Gleeson, Y. Moreno, M.A. Porter, Multilayer networks. J. Complex Network. **2**(3), 203–271 (2014)
83. X. Kong, J. Zhang, P.S. Yu, Inferring anchor links across multiple heterogeneous social networks, in *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management, CIKM'13* (ACM, New York, 2013), pp. 179–188
84. J. Kunegis, A. Lommatzsch, C. Bauckhage, The slashdot zoo: mining a social network with negative edges, in *Proceedings of the 18th International Conference on World Wide Web, WWW'09* (ACM, New York, 2009, pp. 741–750
85. M. Kurant, M. Gjoka, C.T. Butts, A. Markopoulou, Walking on a graph with a magnifying glass: stratified sampling via weighted random walks, in *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems* (ACM, 2011), pp. 281–292
86. N. Lao, T. Mitchell, W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Edinburgh, 2011), pp. 529–539
87. C.-H. Lee, X. Xu, D.Y. Eun, Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling, in *ACM SIGMETRICS Performance Evaluation Review*, vol. 40 (ACM, 2012), pp. 319–330
88. J. Leskovec, C. Faloutsos, Sampling from large graphs, in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining* (ACM, 2006), pp. 631–636
89. J. Leskovec, D. Huttenlocher, J. Kleinberg, Predicting positive and negative links in online social networks, in *Proceedings of the 19th International Conference on World Wide Web, WWW'10* (ACM, New York, 2010), pp. 641–650
90. J. Leskovec, J. Kleinberg, C. Faloutsos, Graph evolution: densification and shrinking diameters. ACM Trans. Knowl. Discov. Data **1**(1) (2007)
91. J. Leskovec, L. Backstrom, R. Kumar, A. Tomkins, Microscopic evolution of social networks, in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'08* (ACM, New York, 2008), pp. 462–470
92. J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2009)
93. D. Liben-Nowell, J. Kleinberg, The link prediction problem for social networks, in *Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM'03* (ACM, New York, 2003), pp. 556–559
94. W. Liu, L. Lü, Link prediction based on local random walk. EPL (Europhysics Letters) **89**(5), 58007 (2010)
95. L. Lü, T. Zhou, Role of weak ties in link prediction of complex networks, in *Proceedings of the 1st ACM International Workshop on Complex Networks Meet Information & Knowledge Management, CNIKM'09* (ACM, New York, 2009), pp. 55–58
96. L. Lü, T. Zhou, Link prediction in weighted networks: the role of weak ties. EPL (Europhysics Letters) **89**(1), 18001 (2010)
97. L. Lü, T. Zhou, Link prediction in complex networks: a survey. Physica A **390**(6), 1150–1170 (2011)
98. G. Malewicz, M.H. Austern, A.J. Bik, J.C. Dehnert, I. Horn, N. Leiser, G. Czajkowski, Pregel: a system for large-scale graph processing, in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data* (ACM, 2010), pp. 135–146
99. P. Massa, P. Avesani, Controversial users demand local trust metrics: an experimental study on epinions.com community, in *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1, AAAI'05* (AAAI Press, 2005), pp. 121–126
100. M. McGlohon, L. Akoglu, C. Faloutsos, Weighted graphs and disconnected components: patterns and a generator, in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'08* (ACM, New York, 2008), pp. 524–532

101. A. McGregor, Graph stream algorithms: a survey. ACM SIGMOD Rec. **43**(1), 9–20 (2014)
102. G. Menichetti, D. Remondini, P. Panzarasa, R.J. Mondragón, G. Bianconi, Weighted multiplex networks. CoRR, abs/1312.6720 (2013)
103. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in *Advances in Neural Information Processing Systems* (2013), pp. 3111–3119
104. S. Milgram, The small world problem. Psychol. Today **2**(1), 60–67 (1967)
105. R.G. Morris, M. Barthelemy, Transport on coupled spatial networks. Phys. Rev. Lett. **109**(12), 128703 (2012)
106. P.J. Mucha, M.A. Porter, Communities in multislice voting networks. Chaos **20**(4), 041108 (2010)
107. P.J. Mucha, T. Richardson, K. Macon, M.A. Porter, J.-P. Onnela, Community structure in time-dependent, multiscale, and multiplex networks. Science **328**(5980), 876–878 (2010)
108. Neo4j: The World's Leading Graph Database. http://neo4j.com/. Accessed 10 March 2016
109. M.E.J. Newman, The structure and function of complex networks. SIAM Rev. **45**(2), 167–256 (2003)
110. M.E. Newman, Modularity and community structure in networks. Proc. National Acad. Sci. **103**(23), 8577–8582 (2006)
111. M. Newman, *Networks: An Introduction* (Oxford University Press, Oxford, 2010)
112. M.E. Newman, M. Girvan, Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
113. M.K.-P. Ng, X. Li, Y. Ye, Multirank: co-ranking for objects and relations in multi-relational data, in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2011), pp. 1217–1225
114. F. Niu, C. Zhang, C. Ré, J. Shavlik, Elementary: large-scale knowledge-base construction via machine learning and statistical inference. Int. J. Semant. Web Inf. Syst. **8**(3), 42–73 (2012). July
115. OrientDB: Distributed Graph Database. http://orientdb.com/. Accessed 10 March 2016
116. L. Page, S. Brin, R. Motwani, T. Winograd, The pagerank citation ranking: bringing order to the web (1999)
117. C.R. Palmer, P.B. Gibbons, C. Faloutsos, Anf: a fast and scalable tool for data mining in massive graphs, in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2002), pp. 81–90
118. Y. Park, M. Shankar, B.-H. Park, J. Ghosh, Graph databases for large-scale healthcare systems: a framework for efficient data management and data services, in *2014 IEEE 30th International Conference on Data Engineering Workshops (ICDEW)* (IEEE, 2014), pp. 12–19
119. A. Pavan, K. Tangwongsan, S. Tirthapura, K.-L. Wu, Counting and sampling triangles from a graph stream. Proc. VLDB Endow. **6**(14), 1870–1881 (2013)
120. PEGASUS - Peta-scale graph mining system. http://www.cs.cmu.edu/~pegasus/. Accessed 10 March 2016
121. B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: online learning of social representations, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2014), pp. 701–710
122. M.A. Rodriguez, The gremlin graph traversal machine and language (invited talk), in *Proceedings of the 15th Symposium on Database Programming Languages* (ACM, 2015), pp. 1–10
123. B. Shao, H. Wang, Y. Li, The trinity graph engine. Microsoft Res., 54 (2012)
124. B. Shao, H. Wang, Y. Li, Trinity: a distributed graph engine on a memory cloud, in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data* (ACM, 2013), pp. 505–516
125. SNAP: Stanford Network Analysis Platform. http://snap.stanford.edu/. Accessed 10 March 2016
126. D. Song, D.A. Meyer, D. Tao, Efficient latent link recommendation in signed networks, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'15* (ACM, New York, 2015), pp. 1105–1114

127. S. Soundarajan, J. Hopcroft, Using community information to improve the precision of link prediction methods, in *Proceedings of the 21st International Conference on World Wide Web, WWW'12 Companion* (ACM, New York, 2012), pp. 607–608
128. Sparkse: Scalable high-performance graph database. http://www.sparsity-technologies.com/. Accessed 10 March 2016
129. M. Spiliopoulou, Evolution in social networks: a survey, in *Social Network Data Analytics*, ed. by C.C. Aggarwal (Springer, Heidelberg, 2011), pp. 149–175
130. N.V. Spirin, J. He, M. Develin, K.G. Karahalios, M. Boucher, People search within an online social network: large scale analysis of facebook graph search query logs, in *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management* (ACM, 2014), pp. 1009–1018
131. F.M. Suchanek, G. Kasneci, G. Weikum, Yago: a core of semantic knowledge, in *Proceedings of WWW* (2007)
132. X. Sui, T.-H. Lee, J.J. Whang, B. Savas, S. Jain, K. Pingali, I. Dhillon, Parallel clustered low-rank approximation of graphs and its application to link prediction, in *Languages and Compilers for Parallel Computing* (Springer, 2012), pp. 76–95
133. Y. Sun, R. Barber, M. Gupta, C.C. Aggarwal, J. Han, Co-author relationship prediction in heterogeneous bibliographic networks, in *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, ASONAM'11* (IEEE Computer Society, Washington, DC, 2011), pp. 121–128
134. Y. Sun, J. Han, C.C. Aggarwal, N.V. Chawla, When will it happen?: relationship prediction in heterogeneous information networks, in *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM'12* (ACM, New York, 2012), pp. 663–672
135. J. Sun, C.K. Reddy, Big data analytics for healthcare, in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2013), pp. 1525–1525
136. J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: large-scale information network embedding, in *Proceedings of the 24th International Conference on World Wide Web Conferences Steering Committee* (2015), pp. 1067–1077
137. S. Tasci, M. Demirbas, Giraphx: parallel yet serializable large-scale graph processing, in *Euro-Par 2013 Parallel Processing*, ed. by F. Wolf, B. Mohr, D. an Mey (Springer, Heidelberg, 2013), pp. 458–469
138. T.T. Tchrakian, B. Basu, M. O'Mahony, Real-time traffic flow forecasting using spectral analysis. IEEE Trans. Intell. Transp. Syst. **13**(2), 519–526 (2012)
139. Y. Tian, A. Balmin, S.A. Corsten, S. Tatikonda, J. McPherson, From think like a vertex to think like a graph. Proc. VLDB Endow. **7**(3), 193–204 (2013)
140. TinkerPop: an Apache2 licensed graph computing framework for both graph databases (OLTP) and graph analytic systems (OLAP). http://tinkerpop.apache.org/. Accessed 10 March 2016
141. Titan: Distributed Graph Database. http://thinkaurelius.github.io/titan/. Accessed 10 March 2016
142. C.E. Tsourakakis, Fast counting of triangles in large real networks without counting: algorithms and laws, in *ICDM'08* (IEEE Computer Society, Washington, DC, 2008), pp. 608–617
143. T. Wang, Y. Chen, Z. Zhang, T. Xu, L. Jin, P. Hui, B. Deng, X. Li, Understanding graph sampling algorithms for social network analysis, in *Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops, ICDCSW'11)* (IEEE Computer Society, Washington, DC, 2011), pp. 123–128
144. W.Y. Wang, K. Mazaitis, W.W. Cohen, Programming with personalized pagerank: a locally groundable first-order probabilistic logic, in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM 2013)* (2013, to appear)
145. D. Wang, D. Pedreschi, C. Song, F. Giannotti, A.-L. Barabasi, Human mobility, social ties, and link prediction, in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'11* (ACM, New York 2011), pp. 1100–1108
146. D.J. Watts, S.H. Strogatz, Collective dynamics of'small-world'networks. Nature **393**(6684), 409–10 (1998)

147. K. Wehmuth, A. Ziviani, E. Fleury, A unifying model for representing time-varying graphs. In *2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015, Campus des Cordeliers, Paris, France, 19–21 October 2015* (2015), pp. 1–10, 2015
148. P.C. Wong, C. Chen, C. Gorg, B. Shneiderman, J. Stasko, J. Thomas, Graph analyticslessons learned and challenges ahead. IEEE Comput. Graph. Appl. **5**, 18–29 (2011)
149. S.H. Yook, H. Jeong, A.L. Barabasi, Weighted evolving networks. Phys. Rev. Lett. **86**(25), 5835–5838 (2001)
150. J. Zhang, X. Kong, P.S. Yu, Transferring heterogeneous links across location-based social networks, in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM'14* (ACM, New York, 2014), pp. 303–312
151. Y. Zhao, Mining Large Graphs. Ph.D. thesis, University of Illinois at Chicago (2013)
152. D. Zhou, S.A. Orshanskiy, H. Zha, C.L. Giles, Co-ranking authors and documents in a heterogeneous network, in *Seventh IEEE International Conference on Data Mining, 2007. ICDM 2007* (IEEE, 2007), pp. 739–744
153. R. Zou, L.B. Holder, Frequent subgraph mining on a single large graph using sampling techniques, in *Proceedings of the Eighth Workshop on Mining and Learning with Graphs* (ACM, 2010), pp. 171–178