Review

Graph Theory: A Comprehensive Survey about Graph Theory Applications in Computer Science and Social Networks

Abdul Majeed [1],* and **Ibtisam Rauf** [2]

[1] School of Information and Electronics Engineering, Korea Aerospace University, Deogyang-gu, Goyang-si, Gyeonggi-do 412-791, Korea

[2] Department of Computer Science, Virtual university of Pakistan, Islamabad 1239, Pakistan; Ms130400028@vu.edu.pk

* Correspondence: abdulmajid09398@kau.kr or abdulmajid09398@gmail.com; Tel.: +82-10-9503-9597

check for updates

Abstract: Graph theory (GT) concepts are potentially applicable in the field of computer science (CS) for many purposes. The unique applications of GT in the CS field such as clustering of web documents, cryptography, and analyzing an algorithm's execution, among others, are promising applications. Furthermore, GT concepts can be employed to electronic circuit simplifications and analysis. Recently, graphs have been extensively used in social networks (SNs) for many purposes related to modelling and analysis of the SN structures, SN operation modelling, SN user analysis, and many other related aspects. Considering the widespread applications of GT in SNs, this article comprehensively summarizes GT use in the SNs. The goal of this survey paper is twofold. First, we briefly discuss the potential applications of GT in the CS field along with practical examples. Second, we explain the GT uses in the SNs with sufficient concepts and examples to demonstrate the significance of graphs in SN modeling and analysis.

Keywords: graph theory; clustering; social networks; social network analysis; cryptography

1. Introduction

1.1. Historical Background of the Graph Theory

The concept/origin of graph theory (GT) started with the Koinsber Bridge's problem in 1735 for the first time in history [1]. With the help of the Koinsber Bridge's problem, the concept of the Eulerian graph was derived. After that, Euler studied the concept of the Koinsber Bridge's problem, and devised a new structure which was named the Eulerian Graph. Later, A.F Mobius gave the concept of two graphs i.e., complete graph and bipartite graph, in 1840 [2]. Kuratowski used both these graphs in some recreational problems and proved that both these graphs are planar in the recreational problems domain [3]. Gustav Kirchhoff gave the idea of tree structure without cycles in 1845 [4]. In addition, authors explained the GT concepts usage in calculating current in electrical circuits. In 1852, Thomas Gutherie introduced the four color problems which are extremely useful in today's computer applications [5]. After that, in 1856, the Hamiltonian graph was invented by Thomas. P. Kirkman and Hamiltonian, and this type of graph has been extensively used in the literature [6–9]. The puzzle problem was introduced in 1913 by H. Dudeney using the prior concepts of the GT [10]. In 1878, James Joseph Sylvester introduced the word "Graphs" [11]. He used two concepts of algebra i.e., co-variants and quantic invariants, and drew an analogy between them. Ramsey devised the concept of the external graph theory in 1941 by working on colorations [12]. Until then, GT concepts were adopted by a majority of scientists in their works. Recently, graphs are extensively used in many

disciplines including social networks (SNs) modelling, big data analysis, natural language processing (NLP), complex network analysis, and patter recognition applications.

1.2. Overview of the Existing Surveys and Applications of Graph Theory in Computer Science and Social Networks

A number of studies have reported the graph theory applications in unique aspects of computer science and social networks. The unique aspects related to both these fields employing GT are cryptography [13], quantum cryptography [14], coding theory [15], construction of asymptotically shortest k-radius sequences [16], privacy-preserving graph publication for informative analysis [17], studying the human brain anatomical network via graphs [18], landscape connectivity and conservation planning [19], image processing [20], information retrieval [21], social network analysis [22], signal processing via graphs [23], knowledge discovery in social networks [24], robotics [25], [26], network analysis of the world's subway systems [27], semidefinite programming [28,29], image segmentation [30], clustering [31–35], data science [36–38], and pattern recognition [39]. These, among others, are the well-known graph theory unique applications. The seven most relevant surveys related to our work are: (1) Riaz et al. [40], (2) Shirinivas et al. [41], (3) DurgaPrasad et al. [42], (4) Liu et al. [43], (5) Qingbao Yu et al. [44], (6) Sporns et al. [45], and (7) Goyal et al. [46]. These studies have explained the GT applications in the field of computer science (CS). Meanwhile, the practical feasibility of the GT applications has not been comprehensively reported by these studies. In addition, there is no single survey that covers all practical applications of GT in both CS and SN domains, respectively.

Graph theory techniques [47] are widely used in different disciplines including chemistry, biology, physics, and computer science [48–50]. GT is basically a mathematics concept with widespread applications in every discipline for modelling and analysis. In this paper, we briefly discuss the uses of GT in the field of CS and SNs [51–55]. We present different applications of graph theory in both these domains. Recently, many efforts have been devoted to developing complex software/environments/packages that can render the complex graph containing excessive amounts of data in a single window or an interface [56–58]. A comprehensive overview of the GT applications in the CS field is presented in Figure 1.

Figure 1. Graph theory applications in the computer science field.

A graph is a set of nodes and edges. The nodes are referred to as vertices/vertex, and the edges are lines or arcs that connect any two/multiple nodes in a graph. A graph can be mathematically represented with **G**. For example, in SNs a user's graph is *G(U, V)* where *U* is the list of users, and *V* represents the set of edges showing the relationship between the users or items. The graphs are of two types: directed and undirected. In undirected graphs, the edges have no direction. An example of the undirected graph is shown in Figure 2a. For example, in Figure 2a depicted below, there is a relationship between *L* and *M*, and this is the same thing as saying there is a relationship between

M and *L*. We can refer to the line between *M* and *L* as (*L*→*M*) or (*M*←*L*) as it makes no difference in interpretation/understanding. The potential examples of the undirected graph are people and friendship in a SN or scientist and co-authored papers in a collaboration network. In contrast, in the directed graphs, the edges do have a particular direction (i.e., they can be regarded as incoming or outgoing). In these cases, the graphs are drawn with edges having arrowheads. The directed graphs are commonly known as digraphs. An example of directed graph is presented in Figure 2b. This might record the social relation "who likes whom" in a SN. Persons *A*, *B*, and *D* all say they like person *C*. Note that person *C* does not say that he likes *A, B,* or *D*. *B* and *A* like each other. Nobody says they like *G*. The example of directed graph is shown in Figure 2b. The potential applications of the directed graph are web pages and hyperlinks connections, Twitter follower graphs, interactions between users in a SN, and influence of one use on another user in SNs.
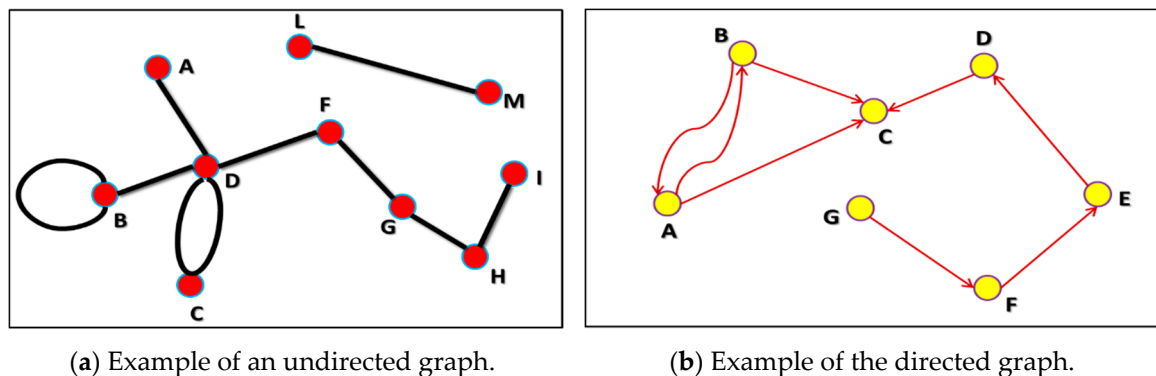

(**a**) Example of an undirected graph.   (**b**) Example of the directed graph.

Figure 2. Pictorial overview of the directed and undirected graphs.

Furthermore, graphs can be classified into two additional categories: valued and non-valued. A valued graph has values (i.e., in the form of numbers) attached to the lines that represent the intensity or strength or quantity of the tie or frequency between the two nodes. The latter just represent the connection/relationship between nodes without any number. In SNs, the lines can represent the relationship (e.g., relative, friend, lover) between two users or the number of times a user *X* interacted with user *Y*. An example of a valued graph is shown in Figure 3, where the edge weight represents the amount of trade, in trillions of dollars, between five different countries. The possible example of a non-valued graph can be the close border of some countries with each other without specifying the length of the borders.
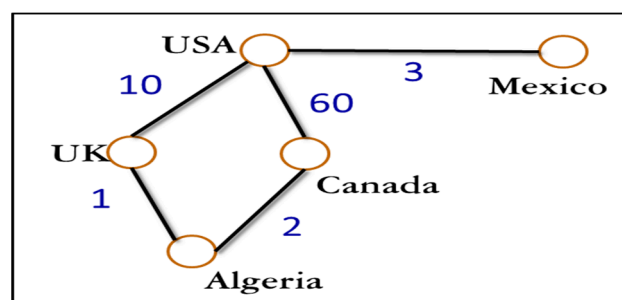

Figure 3. Example of a valued graph (i.e., edges have values).

There exist multiple types of the graphs that can be classified into several categories. Gartner [59] suggests the five broad categories of graphs based on their use, which are presented below.

▪ **Social graph:** This type of graph is about the connections between people/users. Examples of the social graph include Facebook, Twitter, and the idea of six degrees of separation.

- **Intent graph:** This type of graph deals with reasoning and motivation. For example, on Twitter, a user's tweets can be analyzed, and intents are mined from the Twitter posts [60]. Later, the intent can be classified into six categories, namely food and drink, travel, career and education, goods and services, event and activities, and trifle.

- **Consumption graph:** The consumption graph is also known as the "payment graph". It is mainly used in the retail industry. Many e-commerce companies such as eBay, Amazon, and Walmart use this type of graphs to track the consumption of their customers/subscribers. Examples of the consumption graph include credit risk analysis and chargebacks.

- **Interest graph:** Interest graphs models a user's interests. It has been used for the organizing the web by interest rather than indexing only webpages.

- **Mobile graph:** This type of graph is built from mobile data. The mobile data include browsing data from the web, smart applications, web applications, digital wallets, global positioning systems (GPS), and Internet of Things (IoT) devices.

1.3. Our Contribution

The contributions of this research in the field of graph theory (GT) can be summarized as follows: (i) it explains the relevant details about the graph theory concept, different types of the graphs, historical background of the GT, its uses, and practical applications in the computer science (CS) field; (ii) it summarizes the uses of the graphs in social networks (SNs) with sufficient details and concepts; (iii) it covers many realistic examples from both CS and SN in which the graphs were increasingly used in the recent past, and where the same trends may go in the future; (iv) it explains some open technical challenges related to the graph's handling which needs further research and development from the research community; (v) it provides the visual examples with sufficient details that can help researchers to get a concrete overview of the GT use from basic to advanced level in CS and SN domains; and (vi) to the best of our knowledge, this is the first survey that covers GT use in both SNs and CS, respectively.

1.4. Paper Organization

The rest of the paper is structured as follows: Section 2 explains the GT practical applications in the field of CS with examples. Section 3 explains the novel applications of the GT in SNs. Section 4 discusses the challenges of the GT and summarizes the GT use in both fields (i.e., CS and SN). Finally, conclusions and future directions are presented in a Section 5.

2. Graph Theory Practical Applications in the Computer Science Field

This section explains the GT applications in the field of computer science, with examples.

2.1. Uses of Graph Theory in Algorithms

In the field of CS, algorithms have very important roles for developing and upgrading applications. Generally, software developers make the complete design of their applications prior to development, and then they follow this design to develop applications, so that applications are flawless. GT plays a very important role in designing algorithms. There are many algorithms developed with the help of graphs to solve different real-world problems. Some of them are listed as: (1) Depth First Search (DFS) and Breath First Search (BFS) algorithms used in the data structure for searching a node in a directed or undirected graph, (2) MST (Minimum Spanning Tree) algorithm, (3) algorithms for finding the shortest path in a network, (4) Graph Planarity algorithm, and (5) depicting the data transfer in complex applications. There are numerous computer programming languages which are used in manipulating graph theory concepts. Some of the graph programming languages are explained below:

2.1.1. GIRL (Graph Information Retrieval Language)

Graph information retrieval language (GIRL) is a programming language that permits the IRD operations (i.e., insertion, retrieval, and deletion) of information mapped onto the graph structures. It has important significance in the three prospects: (1) syntactic and semantic network generation, (2) syntactic and semantic network recognition, and (3) dynamic operations (i.e., IRD operations) on the graph structure A practical example to retrieve relevant information using GIRL about animal's source information is shown in Figure 4.
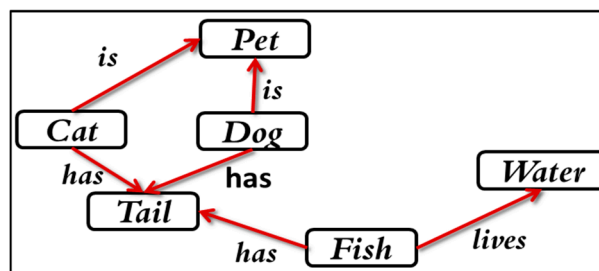

Figure 4. Semantic network retrieving animals source information.

2.1.2. GASP (Graph Algorithm Software package)

GASP is the most widely used discrete event simulation language over graphs. It is flexible, easy to code, and important for continuous/discrete event modeling and simulation. It has different blocks for each specified task. The blocks supported by the GASP language in combination with the other modules are listed in Table 1, and all these blocks are explained in the study [61].

Table 1. The list of blocks supported by the GASP language (Ref. [61]).

ADVANCE	GENERATE	PRIORITY	SEIZE
ALTER	JOIN	QUEUE	SPLIT
ASSIGN	LEAVE	RELEASE	TABULATE
BUFFER	LINK	REMOVE	TERMINATE
DEPART	MARK	RETURN	TEST
ENTER	MSA VEVALUE	SAVE VALUE	TANSFER
EXAMINE	PREEMPT	SCAN	UNLINK

The graph given in Figure 5 is an example of the graph algorithm software packages used with a GASP language for representing the weights.
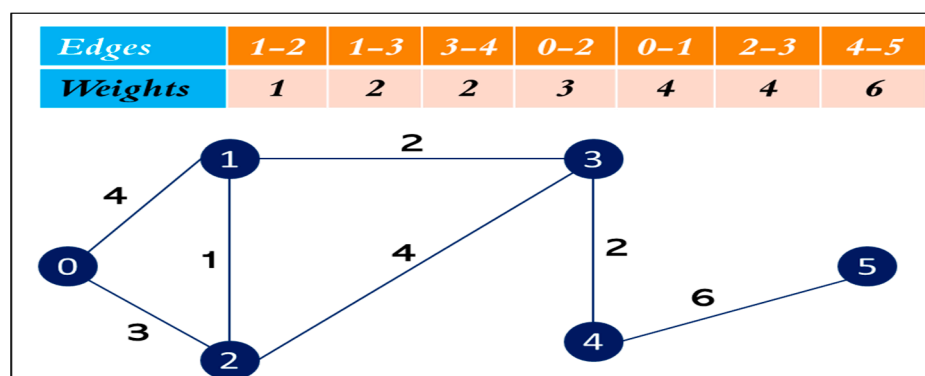

Figure 5. Example of the list to graph formation in software using GASP (graph algorithm software package) language.

2.1.3. GTPL (Graph Theoretic Programming Language)

GTPL is an extension of the FORTRAN language for handling graphs. It is regarded as a dialect of the FORTRAN programming language. GTPL and FORTRAN have lots of similarities in their syntax. Meanwhile, GTPL is able to handle the collections of graphs. The comprehensive details about the language can be found in the studies [62,63]. The example of the GTPL is shown in Figure 6. The choice of the language depends upon the given problem and objectives of the analysis.
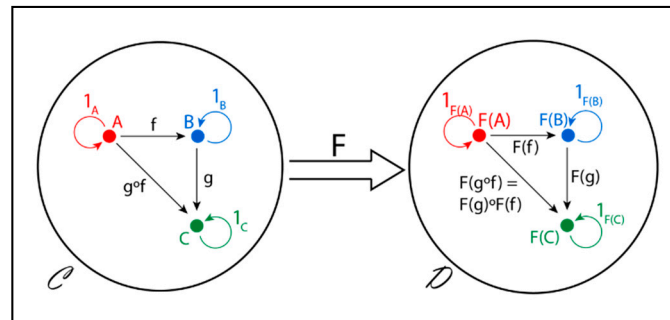

Figure 6. Object to object mapping example in a GTPL (graph theoretic programming language).

2.2. Group Special Mobile Networks and Maps Coloring using Graphs

Group special mobile (GSM) is a geographical area network used for mobile phones. Geographical area is divided into hexagonal areas or cell of varying sizes. A communication tower is connected to a specific cell. Mobile phones are connected within a specific cell using that communication tower and all mobile phones are connected to the GSM network by finding cells in a nearby area. GSM networks have four different frequency ranges. Hence, a graph with four colors can be used to color the cellular areas. A vertex coloring algorithm can be used to allocate four different frequencies for any GSM network. This not only makes the network easier, but it improves the frequency adjustment as per the user's requirements. An example of the GSM architecture is given in Figure 7. GSM and their related components can be modeled with graphs.
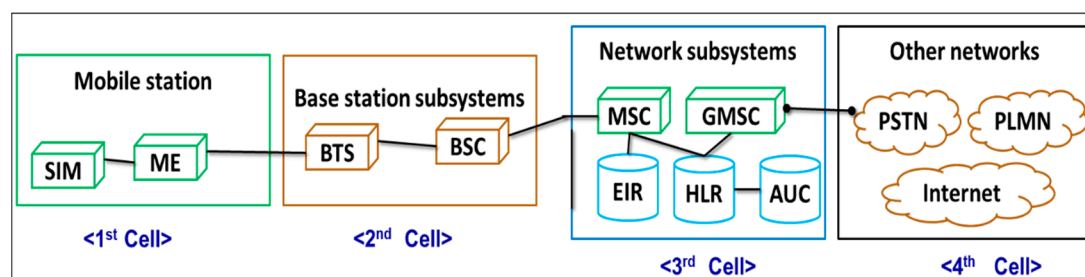

Figure 7. A detailed architecture of the GSM (group special mobile) network (example of the graphs based modelling of GSM).

2.3. Uses of Graph Algorithms/Concepts in Network Security Monitoring

We can use GT concepts in network security to represent the network attacks in real time/offline. For example, a graph G with n number of vertices in which we search for a vertex of at most size k. Here, our aim is to find a minimum vertex cover in the graph whose vertices can be used as routing servers and whose edges will be used to connect these routing servers with each other [64,65]. Then, a worm circulation solution should be found, and we can try to define a defense strategy for networks against worm circulation only on the relevant vertex. A graph G which has a set of edges E is said to cover G if each vertex of G occurs on at least one edge in e. An example of the G and vertex that can cover the graph fully is given in Figure 8.
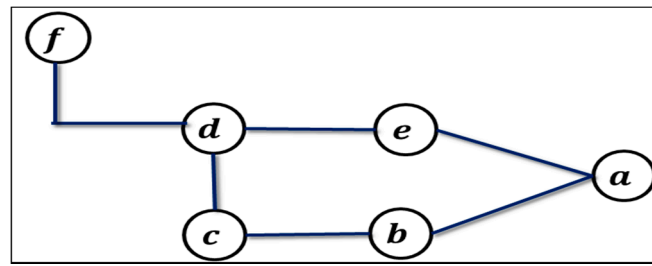
Figure 8. Vertex set V = {*b*, *d*, *e*} covers all nodes in graph *G*.

In addition, we can use graphs to perform real-time threat analysis. This can enable security analysts to deploy defense mechanism accordingly [66]. We can also present the failure attempts on a particular system via graphs. An example of this is shown in Figure 9.
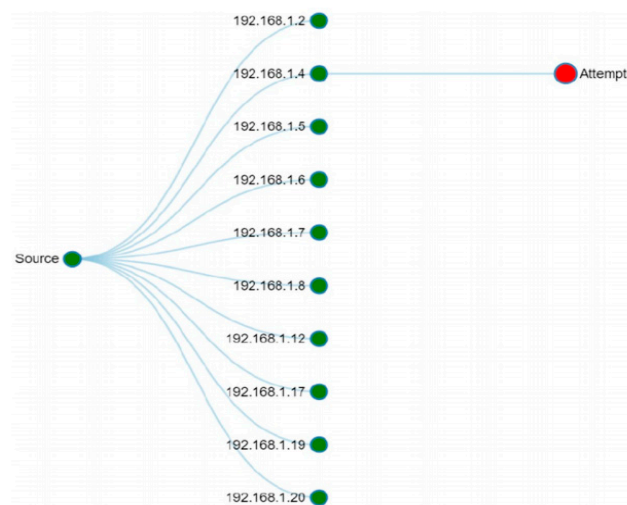

Figure 9. Login failures attempt analysis on a specific system using graphs.

The comprehensive details given in Figure 9 enable the security analyst to ensure best protection for the victim machine. Accordingly, the classification of the insider and outsider threats can be performed by informative analysis offered by interactive graphs.

2.4. Services Connectivity Analysis Using Graphs

An algorithm can be presented with the help of graph where different services are presented as nodes of the graph and edges are used to link these services with each other. These services are used to execute the algorithm. If there is a computer system *S* which is used to execute any specific algorithm A. If A is an isomorphic algorithm/service to a sub graph of computer system *S*. There is one-to-one mapping from nodes of algorithm/service *A* to nodes of computer system *S*. All the other services are connected between these services held by computer system *S* which are required by algorithm/service *S*. So, we can embed A in *S*. If G1 represents a computer system and G2 represents an algorithm/service, then we can execute G2 (see Figure 10) with the help of G1. The accurate relationship modelling between systems and algorithms is shown in Equation (1).

$$(Y1, Y2) \rightarrow (X1, X2), (Y1, Y2) \rightarrow (X2, X3) \qquad (1)$$
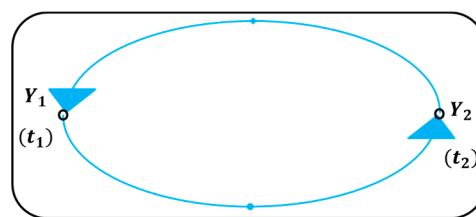
Figure 10. Representation of G2.

A well-known design pattern, model, view, and controller (MVC) [67] interactions, can also be modeled with the help of graphs. The graphical representation of the MVC pattern makes it simple for the designers/developers to follow during actual development. An abstract view of the MVC design pattern is depicted in Figure 11a, where the nodes are the module of the MVC, and edges represent the interaction and data/control flow in a web application. In addition, the service-use flow can be modelled via graphs with sufficient details (as shown in Figure 11b). For example, the user request can be gathered at the controllers which makes the application error-free.
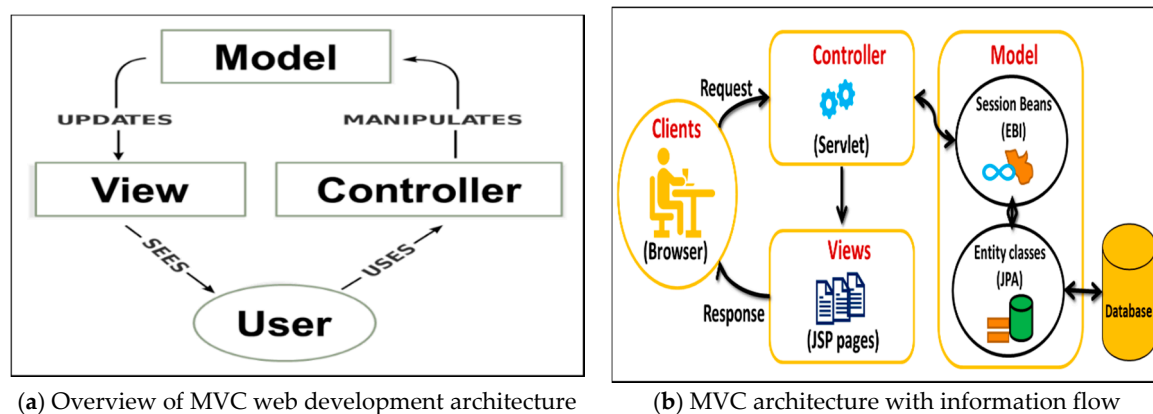

(**a**) Overview of MVC web development architecture   (**b**) MVC architecture with information flow

Figure 11. Model, view, and controller (MVC) development architecture: overview and information flow modelling via graphs.

With graphs, the applications testing can become easy and robust. Furthermore, the information flow can be monitored easily. In some cases, it can be used to organize the complete files via graphs for each distinct language used in an application. A generic overview of graphs used to represent the main modules and related files of each scripting language are shown in Figure 12.

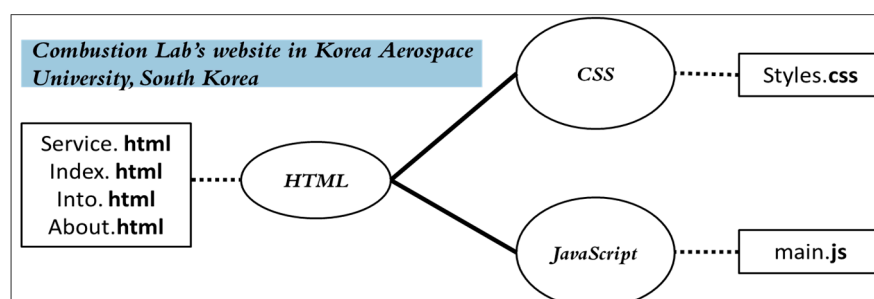**Figure 12.** The use of graphs in organizing distinct category files used in a web application.

In Figure 12, the language names are the vertex and files are attributes (files) of each language. With the help of the above concepts, one can organize all related files used in a web application for simplicity. Graphs can also be used in analyzing the website use trends and user visits frequency.

2.5. Web Documents Clustering Using Graph Theory Concepts

Web documents can be presented as a cluster in search engines. Web documents clustering is a process of organizing similar types of web documents together in a same class [68,69] or server. To optimize the query and appropriate information retrieval, it is desirable to keep similar types of web documents in one cluster [70]. Clustering is widely used in information retrieval processes [71–73] and web document collections [74]. The concept of web document clustering is shown in Figure 13.
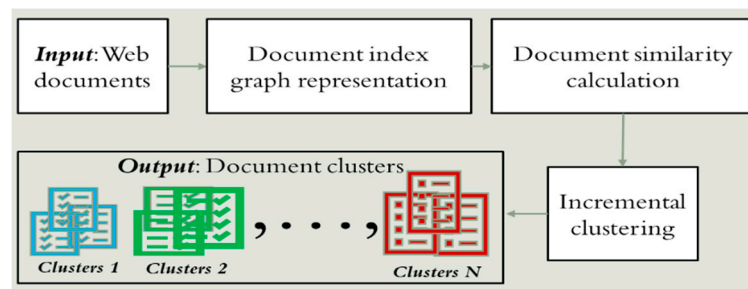

Figure 13. Conceptual overview of web documents clustering.

k-means clustering algorithm is an exclusive clustering algorithm in which each data item is related to only one cluster. In the *k*-mean clustering algorithm, each cluster is represented by the center of the cluster called the mean point. *k*-mean algorithm can be implemented in four steps: (1) make the partition of the different documents into *k* non-empty subsets, (2) compute the mean point of each cluster, also known as seed point or centroid, (3) associate each document with its nearest cluster seed point, and (4) go back to step 2; stop when there are no more documents remaining separate from a cluster. An example of *k*-mean clustering working is given in Figure 14.
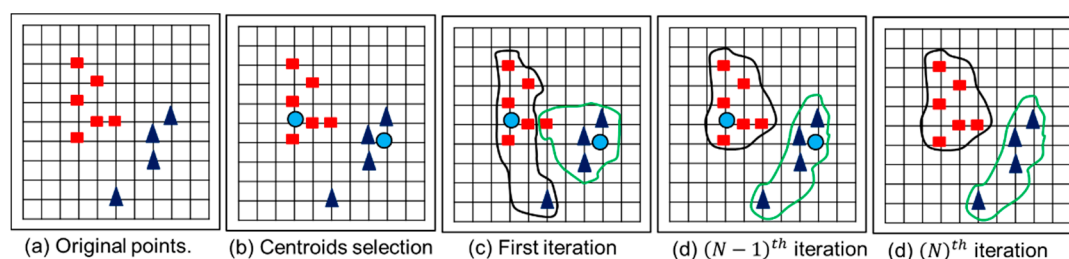

(a) Original points.   (b) Centroids selection   (c) First iteration   (d) $(N-1)^{th}$ iteration   (d) $(N)^{th}$ iteration

Figure 14. The formulation of different coloring options used in *k*-mean clustering algorithm.

k-mean clustering has variety of applications in each sector. The potential five applications of *k*-mean clustering are given as: (1) modelling customer characteristics for target marketing and campaigns, (2) customer need and preferences segmentation, (3) image compression for reducing the space complexity in computer vision applications, (4) network security monitoring, and (5) SNA and mining, etc. Another clustering method is a median graph which is an undirected graph used for clustering in which there are any three vertices let's say *x*, *y*, and *z*. These three vertices have a unique median. In this undirected graph, the median is a vertex (*x*, *y*, *z*) that is related to the shortest path between any two vertices of *x*, *y*, and *z*. A median graph of three vertices is shown in Figure 15.
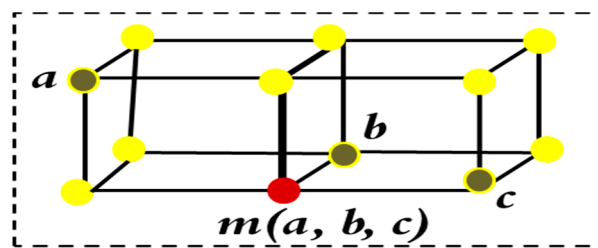
Figure 15. The median of three vertices in a median graph.

Presenting web documents in the form of graphs has two benefits: (1) it uses the inherit structure of original web documents instead of using vectors that include its frequencies, and (2) it is not necessary to develop a new clustering algorithm from scratch for everything. An example of graphical representation of web documents is shown in Figure 16. The entities are the vertices of the graphs and linking between them is shown on the edges.
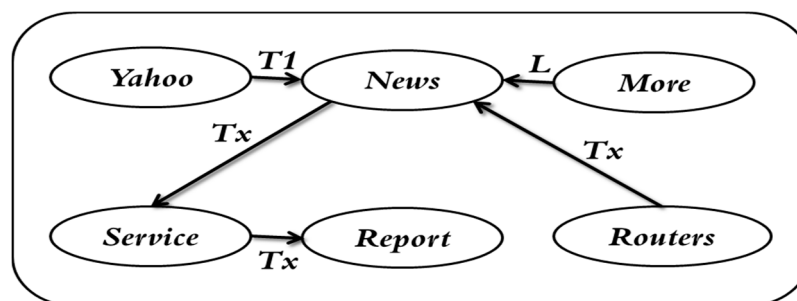

Figure 16. Graphical representation of web document clustering.

2.6. Representing/Modelling Wireless Sensor Network as a Graph

There are many applications of a wireless sensor network (WSN) such as defense applications and tracking of different mobile objects. A special type of graph is called the Voronoi graph, which decomposes a metric space determined by distance to identify the distinct set of objects within the space. A Voronoi graph is drawn in a plane with the help of polygons in which different nodes are presented as sensors and the boundaries of the polygon are considered as the range of every sensor. A pictorial overview of WSN as a Voronoi graph is shown in Figure 17. In Figure 17, alphabets *a-p* represent the sensor's deployment location, and regions that are modelled with cells represent the coverage range of the sensors. With the help of graphs, sensor deployment and their coverage in an area of interest (AOI) can be modelled effectively.
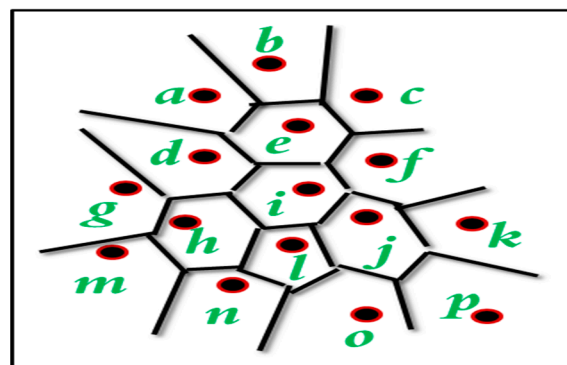

Figure 17. Pictorial overview of a wireless sensor network (WSN) in a graph form.

Figure 17 contains the regions for the sensor deployment. The related measures can be collected from each region and are processed at a base station. WSNs are useful for modelling and optimizing

the transmission schemes [75]. A pictorial overview of a WSN as a graph is shown in Figure 18a. The nodes are the sensors and edges are the links between nodes. Each sensor is used for monitoring and recording the physical measures of the dedicated environment and retaining the collected data from a specific location *S* shown with a square in Figure 18a. There is one source which collects data from all sensor nodes. Some nodes from the network behave as relays for data forwarding to the sink. In addition, GT can be used to model the load balancing to maximize the WSN lifetime and real-time data collection from nodes [76]. The WSN topology and information flow is given in Figure 18b.
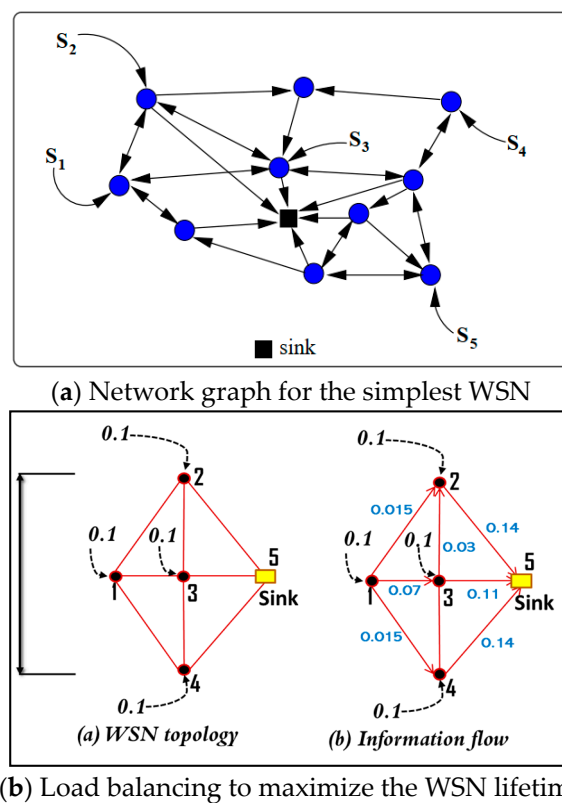

(**a**) Network graph for the simplest WSN

(**b**) Load balancing to maximize the WSN lifetime

Figure 18. Overview of the WSN modeling as a graph and load balancing to maximize WSN's lifetime.

2.7. Uses of the Graph Theory in Operational Research Problems

Apart from the GT applications discussed earlier, the graph theoretical concepts are very much useful in operational research (OR). Many OR problems can be solved with the help of graphs, either directed or undirected. GT helps in simplifying and modelling of complex OR problems.

2.7.1. Use of Graph Coloring

In many real-time applications of computer science, graph coloring technique is very important. There are many coloring methods available depending upon the requirement of the real-time applications. In graphs, proper coloring is used for the vertices and edges where no two vertices/ edges should have the same color. This type of graph is called a colored graph, and the minimum number of colors used in a graph is known as the chromatic number. The coloring is done in a way that two adjacent vertices/edges get distinct colors for the clarity. The example of edges and vertex coloring is shown in Figure 19a,b, respectively.
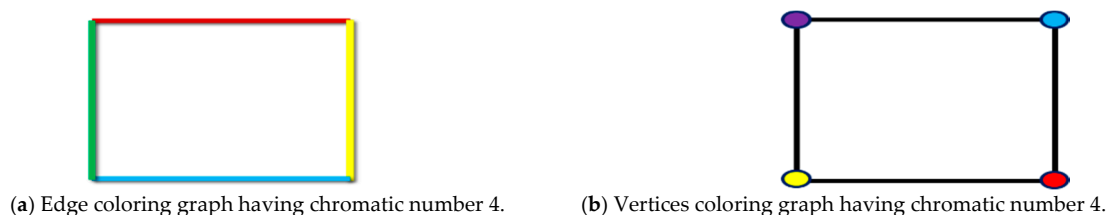
(**a**) Edge coloring graph having chromatic number 4.   (**b**) Vertices coloring graph having chromatic number 4.

Figure 19. Overview of the colored graphs (vertex and edges).

2.7.2. Applications of Graph Theory in Scheduling-Related Problems

Graphs can be used in job-scheduling and flight-scheduling problems. Another example of a scheduling problem is bi-processor tasks, for example when there are many processors and multiple tasks to be executed. Each task should be executed by any two processors simultaneously, and we should pre-allocate these two processors to the tasks. A processor will work only on one task at a time, and it cannot execute multiple tasks simultaneously. Another example is a professor's meeting with multiple students in a lab that can be modelled using a graph. For example, the students who have a meeting at the same time can be clustered in a group. The remaining students can be represented as a graph. Transfer of files among the processors can be applicable in this type of problem. Applying this scenario on the graph's processors will be presented as vertices of graph and if there are any two processors which will work on the same job, they will be connected using edges. Later, we can use the coloring techniques in such a way that each color must appear only on one vertex. We have shown a graph which represents the above scenario in Figure 20.
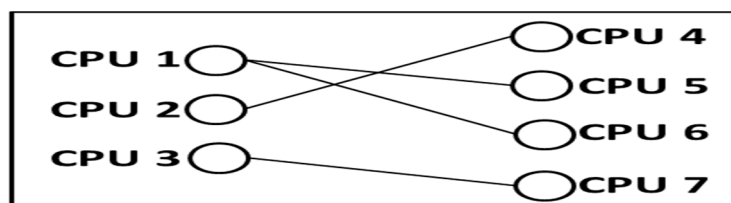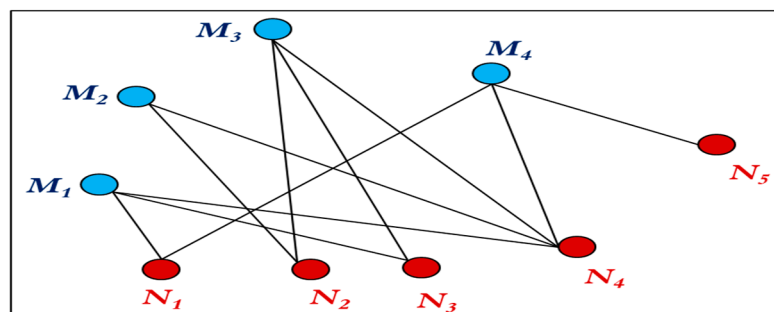

Figure 20. Processors having different tasks in a simultaneous fashion.

The above graph is showing that task1 is allocated to the processors (*P1, P5*), task2 is allocated to processors (*P1, P6*), task3 is allocated to the processors (*P2, P4*) and task4 is allocated to the processors (*P3, P7*). We can use the precoloring technique where allocation of jobs is predefined. Here, some of the vertices of the graph will be pre-colored. List coloring is another technique which is used in graph coloring where there is a list of colors available and we have to find a suitable color from the available list of colors for each vertex. Timetable-scheduling problems can also be solved with the help of bipartite graphs. A bipartite graph is a type of graph where vertices can be divided into two disjoint sets of *V* and *U*. Each edge connects a vertex in *U* to one in *V*.

Suppose there are *m* number of professors with *n* number of subjects and *p* number of periods should be arranged. We should present *m* number of professors and *n* number of subjects as the vertices of the graph and these vertices will be connected with p number of edges where P represents periods. We predefined that each processor can teach one subject at any one period. A maximum of one processor should be taught one subject. The solution of this timetabling problem can be obtained by dividing the edges of the graph *G* by the minimum number of its matches. Also, we can color the edges of a graph. Requirements of teaching with four professors (i.e., $M_1 \sim M_4$) and five subjects ($N_1 \sim N_5$) are summarized in Table 2, the corresponding bipartite graph is given in Figure 21.

Table 2. Teaching with four professors and five subjects.

Professor\Subjects	N_1	N_2	N_3	N_4	N_5
M_1	1	0	1	1	0
M_2	0	1	0	1	0
M_3	0	1	1	1	0
M_4	1	0	0	1	1

Figure 21. Bipartite graph of four professors and five subjects.

Finally, we can color the graph with the help of four colors using a vertex color algorithm. Four colors can be recognized as four subjects. The selected subjects for the one professor are presented in Table 3. Furthermore, graphs can be used for any scheduling problem involving multiple entities/ users/deadlines.

Table 3. Selected subjects for Professor M_4.

Professor\Subjects	1	2	3
M_4	N_1	N_4	N_5

2.8. Applications of Graph Theory in the Internet of Things (IoT)

Due to the advancement in information and communication technologies (ICT), the internet of things (IoT) has emerged as a new area of research. The well-known applications of the GT in the IoT area are: (1) graph-based clustering for two-tier architecture [77], (2) risk assessment in internet of things environment [78], (3) data functional networks and a domain functional networks [79], (4) information diffusion in narrowband internet of things [80], (5) smart transportation using internet-of-things-generated big data [81], (6) vulnerability assessment method in an industrial internet of things (IIoTs) based on the attack graph and maximum flow [82], and information fusion of multiple sources to defend intentional attacks [83].

2.9. Uses of Graphs in a Blockchin and Related Technologies

With the evolution of the 4th industrial revolution, blockchain technology has become popular in recent years. Almost every user can now directly connect to any other user in the world, allowing for point-to-point (p2p) information sharing. There exist plenty of GT applications in the blockchain domain such as graph-based computing resource allocation, smart contract based agreement modelling, transaction analysis, users' interactions, and parallel healthcare systems (PHS). These, among others, are well-known applications of GT in a blockchain domain [84–88].

2.10. Uses of Graph Theory in a Computer Vision Domain/Applications

Due to the recent advancement in ICT, computer vision applications are expanding day by day. GT can be extensively used in computer vision applications by abstracting the underlying process.

Furthermore, it can be used in modelling and analysis of the complex applications. Some examples of GT in the computer vision domain are listed as: (1) graph-coloring-based surveillance video synopsis [89], (2) visual tracking using sparse representation combined with context information [90], two graph classes characterization by small forbidden induced structures using the weighted coloring problem [91], NP-complete problems solutions [92], users' mobility graph, and the computer vision applications such as representation and matching of categorical shape, and human activity recognition. These, and many others, are promising GT-based applications [93].

3. Applications of the Graph Theory in Social Networks (SN)

With the significant development in information and communication technology (ICT), the adoption of the social network (SN) is increasing exponentially [94]. In recent years, SN has become a popular means of information sharing and communicating [95,96] among adolescents. In addition, business/service providers are using user information from the SN for multiple purposes such as latest brands recommendation and campaigns. [97]. Analytics firms are heavily relying the SN user data for analyzing social trends, user attitudes towards latest brands, intent mining, sentiment analysis, and personality analysis [98]. The related concepts, applications, uses, and technological advancements of SN are comprehensively explained in studies [99–103]. SNs are complex networks and can be represented by a graph *G(U, V)*, where *U* is the list of SN users or entities, and *V* represents the set of edges showing the relationship between the users or items. The relationship might be the friendship/lover/family member/sibling on a Facebook. A typical overview of the SN users in the form of a graph is shown in Figure 22. The vertices are users, and edges represent the relationship between the users. The edges can be directed or undirected and are used to represent the similarity, trust, communication cost, interactions frequency, and influence between users in a SN.
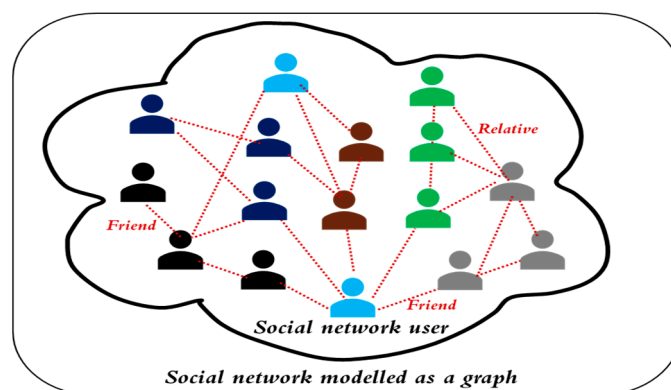

Figure 22. Social network (SN) users modelled as a graph *G*.

The SN users' graph can be either vertex labeled, or edge labelled. The vertices are mainly users (i.e., containing users names or user id) and edges are the relationship (relation, trust, similarity, and/or association) between them. The links can be directed or undirected depending on the problem. The example of a labeled-vertex graph is shown in Figure 23a. The node labels are the users' names and edges are the relationship (i.e., friendship) between users. The example of a labeled-edges graph is shown in Figure 23b. The nodes are the entities and edges are the relationship between them. In this graph, users have five various kinds of relationships with each other.
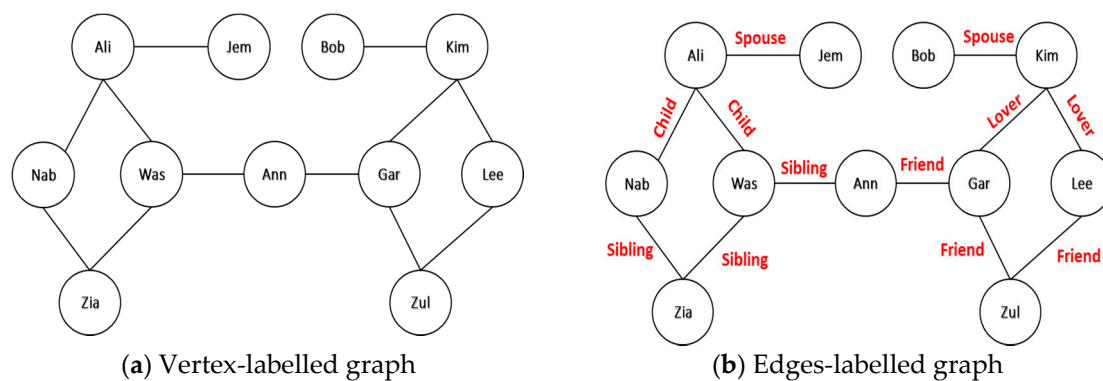
(**a**) Vertex-labelled graph   (**b**) Edges-labelled graph

Figure 23. Example of the labelled-edges and -vertices graph of SN users.

GT applications in a SN include SN evolution, information diffusion, community clustering and detection, SN trust modelling, influence modelling, information contagion, and privacy-preserving data publishing. All these applications employ the GT concept to represent the users as a graph, and show the related statistics (i.e., trust value, similarity, influence score) on the edges. Recently, graphs data publishing about SN users has also become a hot research area [104–108]. The GT applications in the field of SN along with sufficient description and examples are explained below.

3.1. Use of Graphs for Community Clustering in a SN

A community is a group of people/users with some common properties/attributes/ interests/ hobbies/locality/preferences, and/or backgrounds [109]. Community detection in a SN has a range of advantages including target advertising, collaborative filtering, group recommendation, interest-based marketing, information diffusion, personalized advertisement, opinion leader selection, information contagion, and accelerating the information spread. An example of a community with 100 users is given in Figure 24a,b. The users are grouped into two and five communities, respectively.

(**a**) Two communities in a graph   (**b**) Five communities in a graph
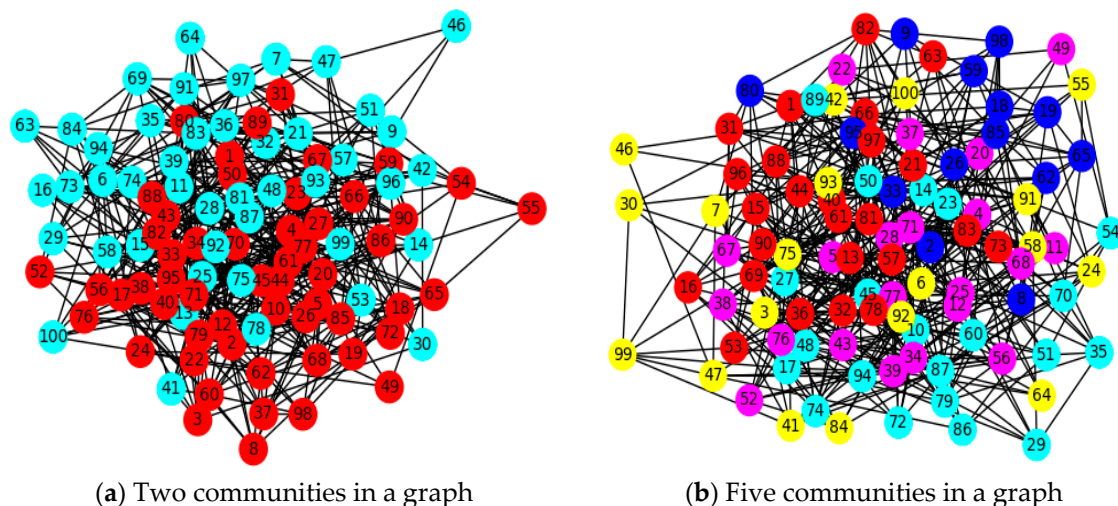
Figure 24. Examples of the detected communities from a graph of 100 nodes in a SN.

A relatively complex example of clustering taken from Tabrizi et al. [110] is shown in Figure 25. In Figure 25, vertices with the identical color correspond to the clusters.
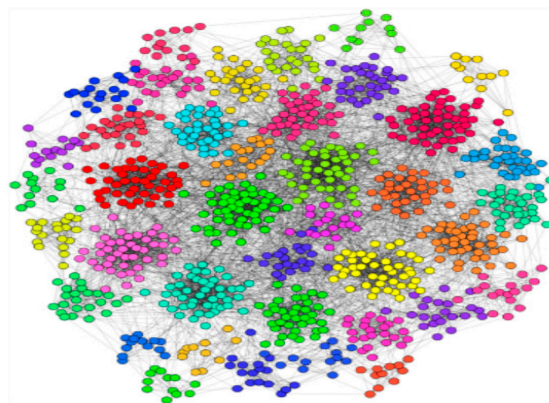
Figure 25. Visualization of the clusters found by the clustering algorithm (Ref. [110]).

There exist plenty of methods to detect communities in SNs. For example, a node can be assigned to a community based on a similarity value with other users in attribute values. In addition, users in a SN can be placed in different communities based on their interests/attributes/behaviors/preferences. Users can be regarded as one community having similar behaviors or attitudes toward some given topic or event. In addition, user communities can be formed based on the activities in an online SN (OSNs) such as commenting on the similar topic and posting similar contents on the SNs sites.

3.2. Use of Graphs for Information Diffusion in Social Networks

SNs allow hundreds of millions of Internet users around the globe to produce and consume digital content [111]. The contents produced by SN users are text, images, videos, audios, memes, and their combinations. The diffusion process is generally to share the relevant information with a large number of audiences in a short period of time. If the contents reach a substantial number of audiences in a short time, then the diffusion process is appropriate [112]. The diffusion process of a particular information *I* is depicted in Figure 26. The red nodes took part in the diffusion process.

Figure 26. The diffusion process of a particular information *I* in a SN (Ref. [112]).

The activation sequence can be extracted based on the time when the messages were exchanged, $[u_4, u_2, u_3, u_5]$, with $t_1 < t_2 < t_3 < t_4$. In some cases, the user's communities are formed based on the trust/interest/similarities, and information diffusion can be done with the relevant communities. The three communities formed based on the interest are shown in Figure 27. With the help of these communities, the diffusion process can become more robust and information can reach the users.

Figure 27. Information diffusion in a SN via communities.

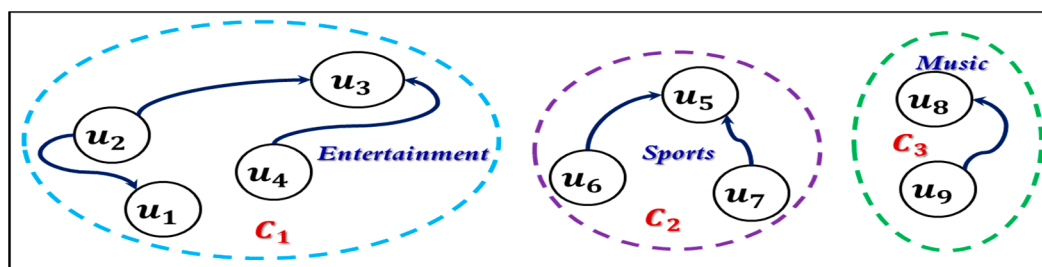Identifying the influential spreaders (IS) in a SN plays a crucial role in various areas, such as disease outbreak, virus propagation, public opinion controlling/mining, and political campaigns. Information diffusion can reach a substantial number of audiences via an IS in a short time. There exist plenty of methods in research to identify ISs for information spread and control in SN [113–115].

3.3. Users' Influence/Trust Score Representation in a Social Networks Via Graphs

Apart from the community clustering and information diffusion, GT concepts can be applied to represent the users' influence/trust on each other in a SN [116]. An example of experts' influence network is shown in Figure 28. In Figure 28, the nodes represent the experts, and edges represent the influence of each expert on each other. The edges can also be used to show the trust score. With the help of a graph, such complex modelling involving more than a million users can be represented effectively. Furthermore, graphs can be used to model the association between different companies for the group decision-making [117]. In addition, graphs can be used for trust modelling and analysis.
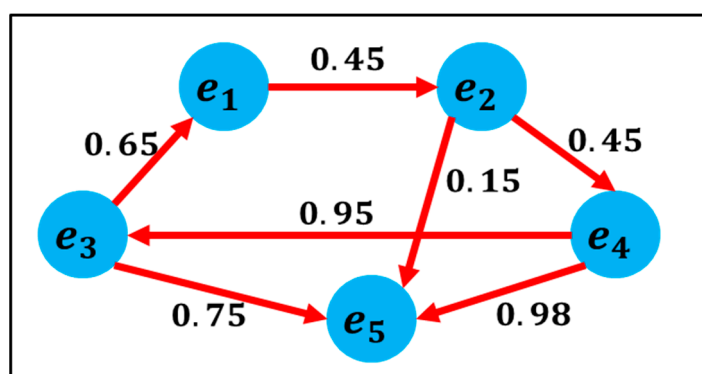

Figure 28. An example of the experts' influence network (Ref. [117]).

3.4. Similarity Modeling/Representing between Social Network Users via Graphs

Finding similar users in a SN for marketing, recommendation, information control, latest brands advertisement, promotion, sales announcements, and/or information diffusion is extremely important. Similar users can be found in the SN by analyzing the similarities, SNs patterns of use, activities, hobbies, buying choices, preferences, opinions, and interest. GT is a handy solution to represent the complex networks of the users in a compact way. The users can be clustered based on different similarities such as demographics, location, and behaviors. GT can be used to show the similarity/dissimilarity between users. The dissimilarity graph among five SN users is shown in Figure 29. Similarly, the similarity concept can be used in many other disciplines and users can be grouped based on the similarity/dissimilarity measures [118].
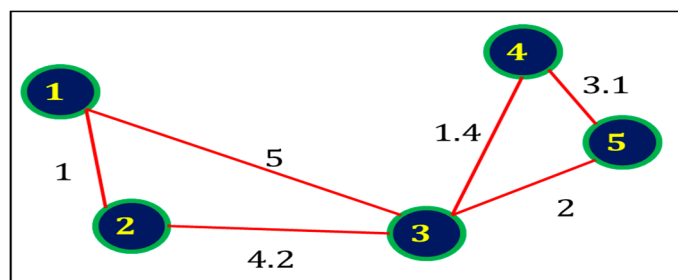
Figure 29. The dissimilarity graph among five SN users.

3.5. Social Network Analysis via Graphs

Social network analysis (SNA) is a current research area, and many studies have been proposed using GT to represent status of entities in a SN. SNA is equipped with a set of methodologies and mathematical formulas for calculating a variety of criteria for the informative analysis. These criteria map and measure the links among things (i.e., people–people, people–item, item–item, and so on). SNA is a process of analyzing the social structures through the joint use of networks and GT [119–121]. SNA characterizes networked structures represented as graph, where nodes (individual actors, users, or things within a network) and the edges/links (relationships or interactions) that connect these nodes [122]. Using social network analysis, one can get answers to five questions such as: (1) How highly connected is an entity (SN user) within a network? (2) What is an entity's overall importance in a network? (3) How does information flow within a network? (4) How central is an entity within a network? (5) Can a particular entity be an influential spreader in a graph? In this work, we define four important metrics related to SNA, and show them graphically. A comprehensive description of the four centrality measures is given in Figure 30.

Figure 30. Description of the four centralities used in social network analysis (SNA).

3.5.1. Closeness Centrality

Closeness centrality (CC) generally measures how quickly a user or entity can access a large number of entities in a network. An entity with a high CC generally has four characteristics: (1) it has quick access to other entities in a network, (2) it has a short path to reach other entities, (3) it is close to other entities in a network, and (4) it has high visibility about what is happening in the network. An example of CC using graph is shown in Figure 31a. If the network contains any entities (i.e., users) that are un-linked (i.e., not linked to any other entities in a given network), the CC value for all entities in the network is 0. This is due to formulas and algorithms established in the SNA. The entities or nodes with high CC value can be used for the information spread control or diffusion purposes.

(**a**) Pictorial overview of the closeness centrality (CC) in a SN.

(**b**) Pictorial overview of the degree centrality (DC) in a SN.

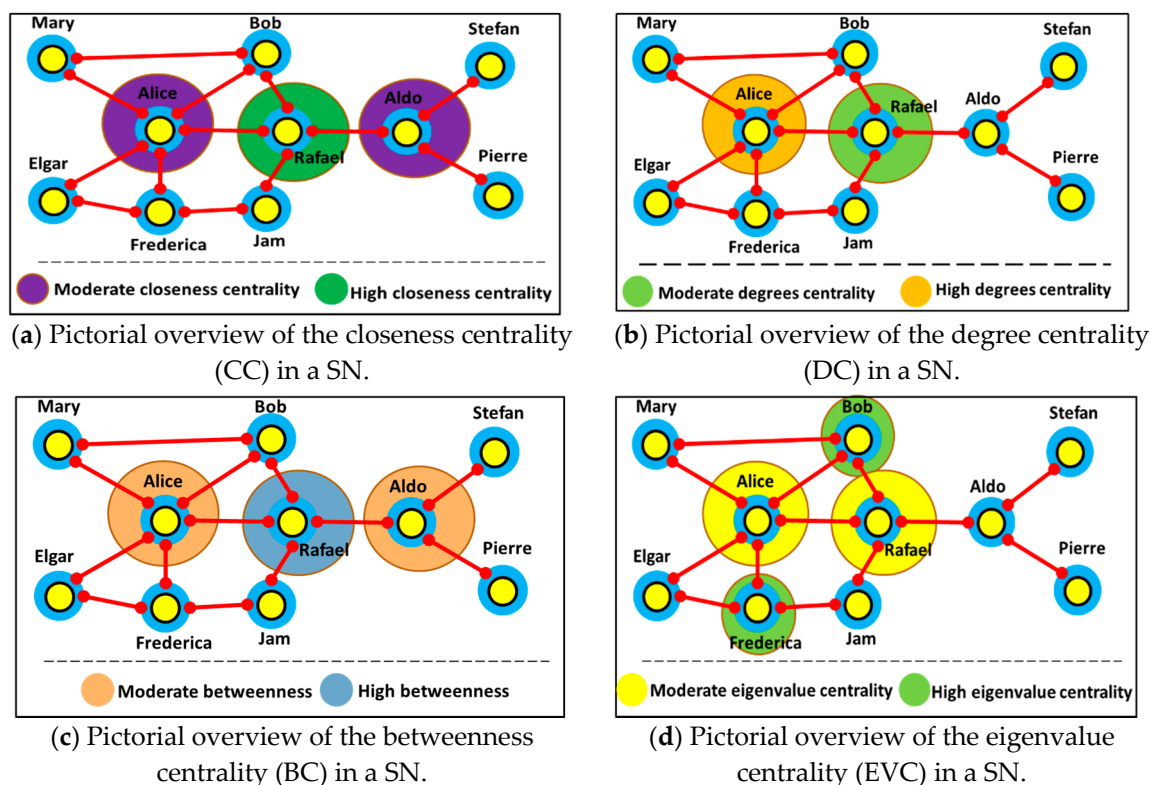(**c**) Pictorial overview of the betweenness centrality (BC) in a SN.

(**d**) Pictorial overview of the eigenvalue centrality (EVC) in a SN.

Figure 31. Overview of different types of centralities in a social network modelled as graph.

3.5.2. Degree Centrality

Degree centrality (DC) represents the number of direct relationships of an entity in a network. A node with high degree centrality has six properties: (1) it is regarded as an active user in a network, (2) it often performs the role of a connector or hub in a network, (3) it is not generally a most connected entity in a network (an entity may have a substantial relationship, the majority of which refer to low-level entities), (4) it might be in a privileged position in a network, (5) it may have alternative paths to satisfy organizational requirements, and consequently may have less dependability on other individuals in a network, and (6) it can often be regarded as third parties or deal makers. An example of DC using graph is shown in Figure 31b. In our example network diagram, Alice has the highest DC, which means that she is more active in a network compared to the all other users. However, she is not necessarily the most influential person because she is directly connected within one degree to the people in her group—she has to go through Rafael to get to another user's group.

3.5.3. Betweenness Centrality

Betweenness centrality (BC) mainly identifies a user's placement within a graph in terms of its capacity to make connections to other users or users' groups in a graph/network. An entity with a high BC generally has three characteristics: (1) it holds a favored or powerful position in a network, (2) it is prone to a single point of failure, i.e., take the single betweenness spanner out of a network and you sever ties between cliques, and (3) it has a significant amount of influence over what happens in a network. An example of BC using graph is shown in Figure 31c. In the presented example, Rafael has the highest BC because he is between Alice and Aldo, who are between other users. Alice and Aldo have a slightly lower betweenness because they are essentially only between their own groups. Therefore, although Alice has a higher DC, Rafael has more importance in a network in certain aspects considering the SNA.

3.5.4. Eigenvalue Centrality

Eigenvalue centrality (EVC) measures how close a user is to other highly close entities within a network. A high EVC has two main properties: (1) it represents an actor that is more central to the main pattern of distances among all entities in a network, and (2) it is an appropriate measure of one aspect of centrality in terms of positional advantages. An example of EVC using graph is shown in Figure 31d. In Figure 31d, it can be observed that Rafael and Alice are closer to other highly close entities in a network. Frederica and Bob are also highly close, but to a lesser value. In addition, GT concepts can be applied to show the minimum number of connections that can be made by different numbers of users in a SN [123]. The edges represent the connection with different users, and vertices represent the actual users.

3.5.5. Jordan Centrality

Jordan centrality (JC) is considered as a node having the smallest maximum distance to other contaminated and recovered nodes [124]. Accordingly, the number of Jordan centers is equivalent to the radius of a graph [125]. Figure 32 shows an example of the JC, where nodes Jam, Alice, and Frederica have a JC value of 3. The nodes in the example having a JC of value three are highlighted with yellow color.
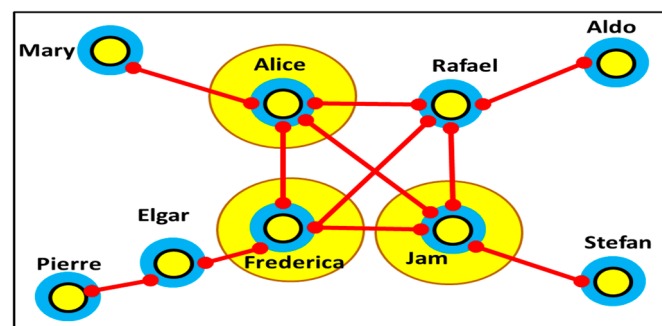

Figure 32. Illustration of the Jordan centrality measure in a SN.

3.6. Analyzing the Modularity in a Social Network Users' Graph

In many cases, we can cluster the network into a number of communities (also called subgraphs). We can use the modularity score/measure to assess the 'quality' of this clustering of a complete SN graph. A higher score represents that the network/graph splitting into small subgraphs/communities is 'highly accurate', whereas a low score shows our clusters are more random than insightful. The latter case can lead to the wrong recommendation/analysis in most cases. Generally, the partition with the strong correlation of the objects in the subgraph is preferred. In simple words, modularity measures the quality of partitioning. The example of the modularity including both cases (high and low) is shown in Figure 33a,b, respectively. With the significant increase in the use of SNs in recent years, the modularity concept has been extensively studied in the research [126–128].
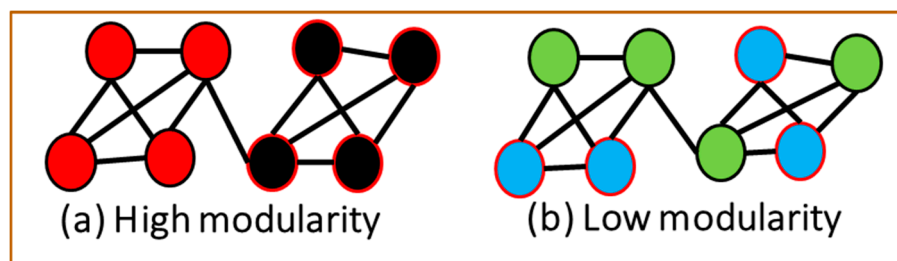

Figure 33. Overview of the high and low modularity in a graph (Ref. [128]).

3.7. Topic of Interest Modelling Using Graphs in Social Networks

Graphs can be used to model the topic of interests. With the wide adoption of SNs, topic modelling has been significantly applied to extract multifaceted information about users. Topic modelling can be defined as a method/procedure to find a group of words (i.e., topic) from a collection of documents/paragraphs (for example, a user's posts, opinions, reviews, and comments on SN platforms) that best represent the information in the collection. It can also be thought of as a form of text mining/processing, or a way to obtain frequently occurring patterns of words in a textual material. Many studies have been proposed for the topic modelling including information-based, content-based, activity-based, and/or aspect-based topic modelling [129–131].

In a micro-blog scenario/application, the noun entities play an important role in language understanding, interpretation, and content generation, which efficiently reflects personal interest knowledge. An example of five topics extracted from micro-blogs is shown in Figure 34a. In Figure 34a, the vertices are the topics with names, and edges model the relationship between them. These topics can be further classified according to the information they contain. The sample of information contained in all five topic categories (listed in Figure 34a) is presented in Figure 34b. Through the topic modelling, information retrieval and processing can become robust. In addition, topic modelling can be used to analyze SN a user's behaviors in a SN. Furthermore, it can be used for user's profile modelling for target advertisements. Recently, topic modelling has been extensively used in the sentiment analysis for cyberbullying detection in SNs. Furthermore, topic modelling has been significantly used in the natural language processing (NLP) for text classification and analysis. With the advancements in machine learning and deep learning techniques, topic modelling has been extensively used in sentiment analysis regarding topic of interest, feedback about shows, and product reviews.
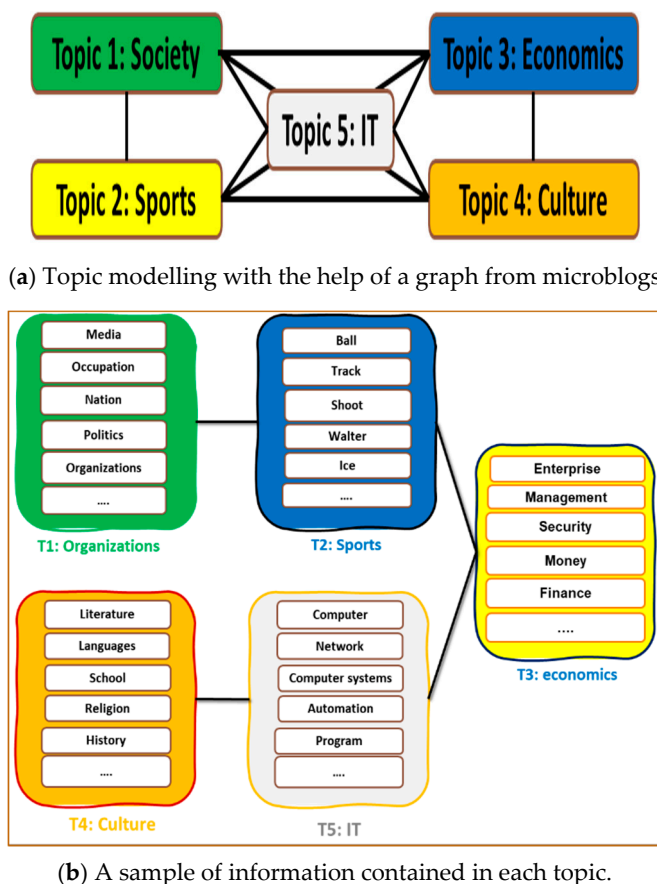

(**a**) Topic modelling with the help of a graph from microblogs.

(**b**) A sample of information contained in each topic.

Figure 34. Overview of topic modelling and information contained in each topic by leveraging graphs.

3.8. User Identification across Social Networks by Analyzing Structural Properties of the Graphs

Users in a SN are modelled/represented as graphs. By analyzing the structural properties of the graphs, users can be identified across the SNs. In addition, they can be used by attackers to re-identify the user's behaviors/activities in various SNs. An example of behavioral habit-based user identification across social networks is given by authors [132]. The survey about the across-SNs user identification in a comprehensive way is explained by a related study [133]. The across-social-network user identification (ASNUI) can help perfect user information and user activity analysis, offer personalized service recommendations, and be used for data mining, as well as provide support for scientific research collaboration. The across-SN user mapping is given in Figure 35a. The user's re-identification example by exploiting the structure of the two different SNs user graph is depicted in Figure 35b.

(**a**) Cross-SNs user mapping (Ref. [133])

(**b**) Across-social-networks user identification (Ref. [132])

Figure 35. Overview of the users mapping and identification across SNs (i.e., multiple SNs).

3.9. Social-Attribute Network (SAN) Modelling and Analysis via Graphs

Given a directed social network in the form of a graph G, where nodes are users and edges represent relationships (i.e., friends) between users, and M distinct binary attributes, which could be static attributes (e.g., name of school, name of employer, major) or dynamic attributes (e.g., interest groups, preferences), a social attribute network (SAN) is an enlarged network with M additional nodes where each such node corresponds to a specific binary attribute, either static or dynamic. For each user

u in *G* with attribute *a*, we can create an undirected link between *u* and *a* (*user -> attribute*) in the SAN. An example of a SAN is shown in Figure 36, and it contains six users and four attributes.
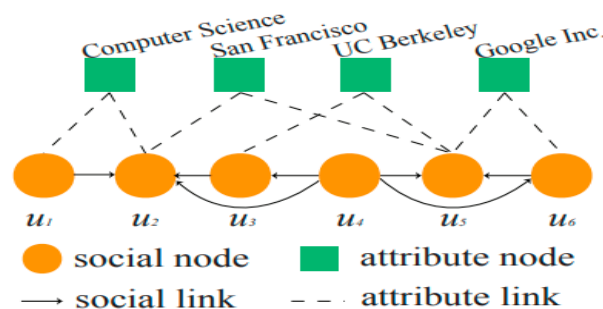

Figure 36. Social attribute network (SAN) with six users and four attributes.

The above figure contains the social links (direct edges) and attributes links (undirected edges). Graphs can be potentially applicable to represent a SAN for large networks. SANs can be used in analyzing the sparse distribution of the attributes and missing values. In recent years, graphs modelling for SAN analysis is attracting significant attention from other related domains [134].

3.10. Graph Theory Applications in Recommendation Systems

The amount of digital content that is being produced by social media sites, movies, television shows, and browsing history is increasing tremendously, and it is very difficult for a user or an individual to choose relevant content according to his/her interest from such a huge pool of the available content. There are endless choices for each item such as movies, songs, purchasing items, friends in a SN, and books. Hence, we need a system to filter out most of this content and give suggestions to a user about the relevant content only [135]. Recommendation systems are designed to solve this problem, and to give consumers/users a best suggestion based on existing data, historical data, and user preferences. Recommendation systems are widely used in e-commerce sites, SN service delivery, healthcare, government sectors, and academia. For example, Netflix suggests movies, Facebook suggests friends based on similarities (i.e., demographic, spatial, temporal, interest, and relationships) and interests, Amazon suggests products, Google Scholar recommends relevant articles based on the area of expertise of researchers, and music applications such as iTunes, YouTube, and Spotify suggest next songs that the user may like to hear based on the current song a user is listening to. It can even be applied to other domains such as social networking micro-blogs, information spread, rumor control, cyber aggression, and hate speech. Facebook uses it for suggesting friends, posts, and events. GT can play a vital role to model the association between users and items of their interest. For example, to model the association between users and movies, the nodes can be the customer/movie and edges will represent whether a particular user has watched a movie or not. We present an example of the bipartite graph to show the usage of graphs in a recommendation system. The information about three customers and four movies is depicted in Table 4.

Table 4. Relevant information about the customers and movies.

Customer/Movie	Movie 1	Movie 2	Movie 3	Movie 4
Customer 1	0	1	0	1
Customer 2	0	1	1	1
Customer 3	1	0	1	0

In Table 4, the *zeros* represent whether a customer has watched a particular movie or not. The *non-zeros* represent that a customer has watched the particular movie and the numeric value represents the rating he/she has given for that movie. The equivalent bipartite graph obtained from the data given

in Table 4 is shown in Figure 37. The actual dataset of the users has a total of *9123* movies and *671* users. Therefore, the bipartite graph matrix has a size of (*9123 × 671*). The association between the movies and users can be easily modelled using a bipartite graph. The recommendation about relevant movies/songs can be done by the content analysis and collaborative filtering [136].
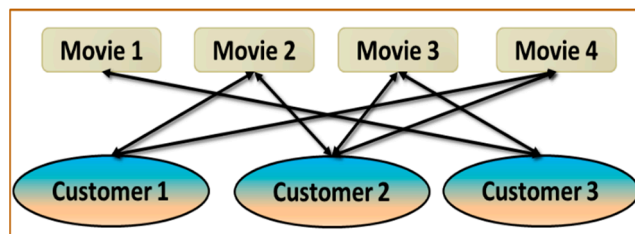

Figure 37. Example of the bipartite graph between customers and movies.

3.11. Social Interactions Modelling between Users in Social Networks via Graphs

SNs provide the means of interaction between users and it is well-documented that individuals care about how others around them are doing or behaving [137]. There exist studies that report a production economy in which consumers provide a labor-type supply to a representative firm to earn income for consumption, and their utility depends on their own leisure time, their own consumption level, as well as their relative's consumption levels. Four network structures (empty network, ring network, star network, and core-periphery network) with different production technologies are analyzed. The proposed study applied the general equilibrium effect of the social preferences and network schemas [138]. An example of four network structures modelled with graphs is shown in Figure 38.
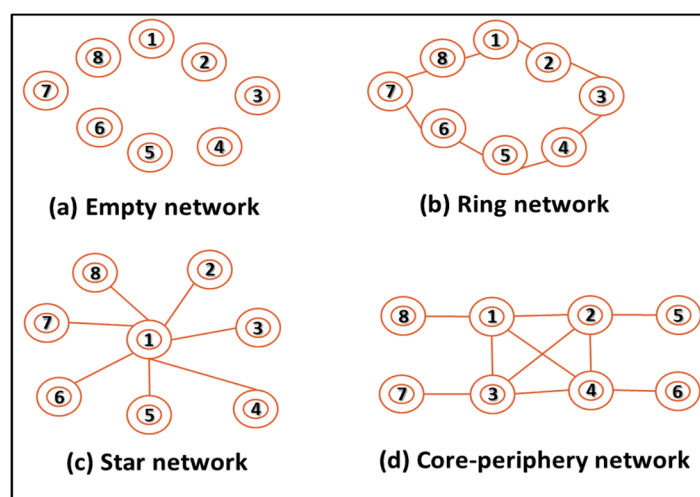

Figure 38. Equilibrium pattern-based four network structures used in production technologies (Ref. [138]).

3.12. Privacy-Preserving Social Network Data Publishing With Researchers/Analysts For Analysis

Tremendous amounts of data are being generated/collected by organizations like hospitals, banks, e-commerce, and retail and supply chains from their users/consumers/subscribers. Just like traditional organizations, SNs such as Facebook, Twitter, and YouTube also collect an excessive amount of data about their users. On the one hand, releasing stored data is beneficial for data mining firms. On the other hand, releasing data may violate the users' privacy. In this work, we discuss the privacy-preserving SN data publishing in a structural (i.e., graph) form. Related studies which publish the SN user's data by perturbing graph structure can be found in the literature [139–143]. The SN data

is in a structural form and is perturbed before publishing with the analytics firm for analysis. In the literature, there exist plenty of techniques which modify the graph structure by either nodes/vertex [144], edges/connections [145], and/or both [146]. For decreasing the attacker's knowledge to re-identify the individuals, the degree concept is employed on graph structure. An example of the 4-degree and 2-degree (i.e., *k = 2, 4*) anonymous graph is shown in Figure 39. The purpose of imposing the degree constraint is to hide the user in the crowd of *k* other users to protect his/her private information privacy.
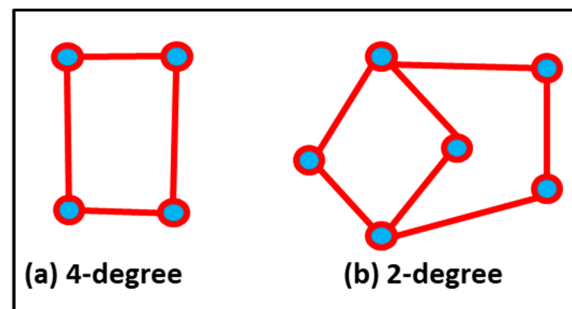

Figure 39. An example of 2- and 4-degree anonymous graph (Ref. [147]).

Figure 39 illustrates 2-degree anonymous graphs. In Figure 39a all four nodes have the same degree, which is 2, so the graph is 4-degree anonymous. In Figure 39b two nodes have degree 3 and four nodes have degree 2, so the graph is 2-degree anonymous. In every *k*-degree anonymous graph, for every node *v* belonging *V* there exist at least *k-1* other nodes that have the same degree as of *v* [147]. A general example of the naïve anonymization of the SN data is shown in Figure 40. Here the user's direct identities (i.e., names, SSN, cell phone numbers, and emails) are removed from the anonymous graph.
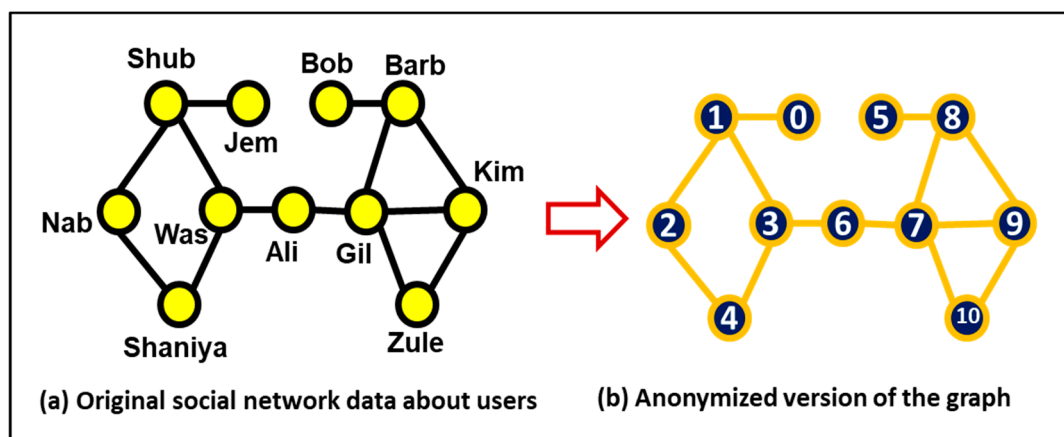

Figure 40. Example of the social network anonymization given in the form of a graph.

Recently, due to the widespread adoption of SNs around the globe, many efforts have been devoted to anonymous SN user data publishing by perturbing the graph structures [148,149]. In the following two examples, we present the anonymization of the graph by adding a new vertex and edges, respectively. The purpose to perturb the graph structure is to increase uncertainty for the intruders. An example of the *3*-degree anonymous graph by adding edges is shown in Figure 41a. In Figure 41a, the edges with red color are the newly added edges to increase uncertainty. By adding the new edges, the utility of the original graphs decreases but the privacy guarantee increases due to more uncertainty. The decision about the vertex addition or edges addition is made on the basis of the published data on consumers, target application, and nature of the data. An example of the anonymization by adding the

vertices/nodes is shown in Figure 41b. The marked vertices are the newly added vertices. By adding the new edges and vertices, the information preservation/revelation degree can be adjusted accordingly.
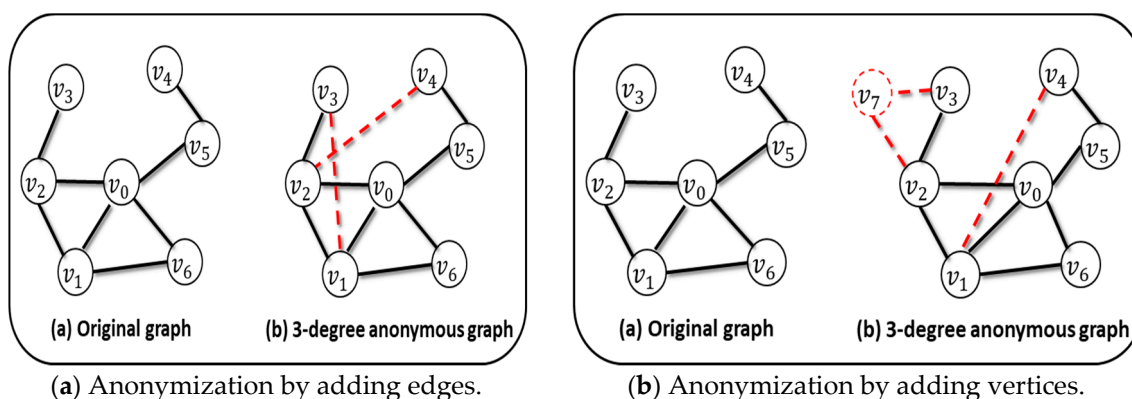

(**a**) Anonymization by adding edges.   (**b**) Anonymization by adding vertices.

Figure 41. An example of the SN users graph anonymization by adding edges and vertices.

In some cases, the relational anonymization techniques are applied to the graph structure to preserve user privacy in publishing SN data [150,151]. An example of the *l*-diverse and *k*-anonymous graph is shown in Figure 42. Such concepts were used for the tabular data anonymization but due to the widespread uses of the SN data, such concepts have been increasingly employed on the SN data. However, these concepts are combined with some structural modifications to produce the anonymous graph from the original graph. Meanwhile, due to the conceptual simplicity of the *l*-diversity and *k*-anonymity privacy models, both these models have been extensively used in relational and graph data anonymization. These models significantly preserve the user's privacy in data publishing with analysts/researchers without significantly degrading data usefulness.
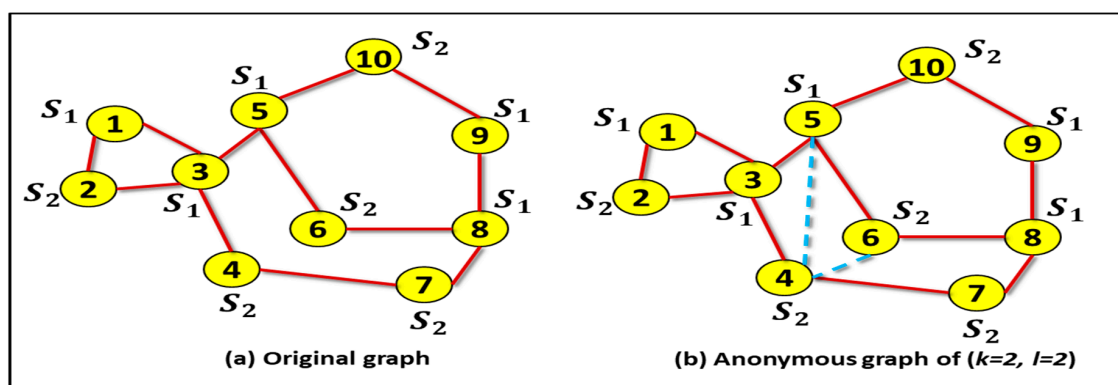

Figure 42. Example of the *l*-diverse and *k*-anonymous graph (where both *k* = 2 and *l* = 2).

The *l*-diversity model/concept states that for every equivalence group based on *k*-value, there should be at least *l* distinct sensitive attribute labels. In Figure 42, we illustrate how an original graph can be modified to satisfy 2-degree anonymity and 2-diversity property i.e., for every node (i.e., user in real-world), there exists at least *l* other nodes having the same degree and for very equivalence group, at least *2* different sensitive labels are present to decrease the sensitive attribute disclosure. In Figure 42, S_1, S_2, etc., show the sensitive label of a different type (for example if the sensitive attribute is salary, then S_1 value can be >50K and S_2 value can be ≤ 50K dollars etc.).

3.13. Community-Based Event Detection in Temporal Networks via Graph Analysis

The modelled graph of the communities can be used for detecting events in temporal/social networks. The suggested method used for event detection is motivated by the fact that viral information

spreading has distinct diffusion/spread patterns with respect to the community structures. Specifically, Moriano et al. [152] suggest that global events trigger viral information cascades and can thus be detected by monitoring intra- and inter-community communications that occur across the community boundaries. By comparing the expanse of communication patterns among intra- and inter-communities, authors show that it is possible to detect several types of events, even when they do not trigger a significantly larger communication among users. A schematic representation of the proposed event detection method taken from the Moriano et al. [152] study is shown in Figure 43.

From Figure 43, it can be observed that for each network, nodes are belonging to the same communities but different patterns of communication within and across communities appear. In Figure 43a, when there is no event, most of the communication takes place within communities only. In Figure 43b, when a global event occurs, more communication takes place among the communities because of the global relevance and the virality of the specific event. Through this way the events can be accurately detected in any social network modelled as a graph *G*. In addition, it is a robust way of event detection in SN.
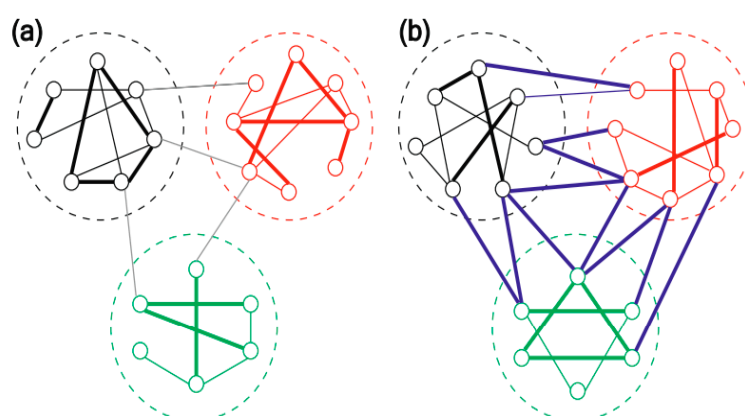

Figure 43. Schematic overview of the community-based event detection method (Ref. [152]).

3.14. Affiliation Network Modelling Using Graphs in Social Network

In a SN, users form different types of networks such as friendship networks, community networks, interest networks, and affiliation networks (AN). An AN [153] is represented as a bipartite graph with two types of nodes *V* and *H,* and the affiliation links between them E_h. Figure 44 shows an example of this AN graph. In this AN graph, the left-hand side represents different users, and on the right-hand side there are different movies that the users have rated. The affiliation links have a weight corresponding to the movie ratings provided by each user, on a scale that ranges from 1~5. These types of the network are common in a recommender system and user–user/item modelling.

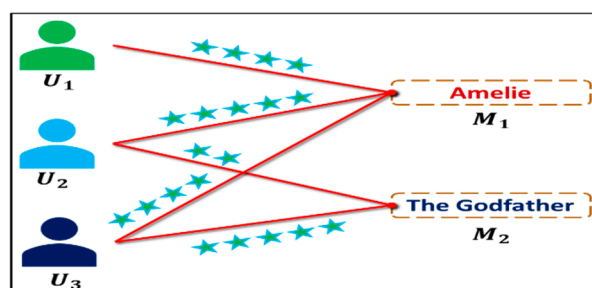**Figure 44.** An affiliation network as a bipartite graph between three users (U_1, U_2, U_3) and two movies (M_1, M_2). The affiliation links show the ratings that users gave to the movies (Ref. [153]).

With the help of affiliation networks, the URL access patterns can be modeled with ease. For example, there is a bipartite graph of (query, URL) just like (users, movies) pairs. Here, the links

have a weight corresponding to the number of users who posed a particular query and clicked on the particular URL while using the web. GT concepts are extensively used for the appropriate modelling of such problems. GT concepts are helpful in representing the large data volumes concisely.

3.15. Modelling of the Sybil Attack in Social Networks Using Graphs

SNs such as Facebook, YouTube, and BitTorrent have become vulnerable to Sybil attacks [154]. Many existing studies reported that Sybil users affect the correct functioning of any SN by contributing malicious content. Hence, illegitimately increasing influence and power with legitimate users. Malicious activities from the Sybil users are posing serious threats to the SN users, who trust the service and depend on it for online interactions just like normal users. There exist two categories of solutions that are available to defend Sybils attacks in a SN: (1) sybil detection/identification, and (2) sybil resistance/admittance control. Sybil detection schemes leverage the SN structure to identify whether a given user is sybil or non-sybil. We present an example of the graph that can be used to model the sybils in a SN (see Figure 45).

Figure 45. The system model for Sybil attack detection using graphs in SNs (Ref. [154]).

Apart from the sybil attack, SNs confront with the friend in the middle (FITM) attack [155]. The graphical representation of the FITM attack is shown in Figure 46. In this attack, the adversary hijacks the session to infer the personal information of the users. Such attacks target the user's sensitive information shared/exchanged with other users.

Figure 46. Outline of a large-scale spam campaign via the friend-in-the-middle attack. A social networking session is hijacked to fetch personal information from a victim's profile. The extracted information is then used for spam and phishing emails targeted at the victim's friends (Ref. [155]).

3.16. Estimating and Inferring the Strengths of Social Relations among Different People in SN Using Graphs

GT can be applied to estimate the strength of social relationships among different users in a SN users' graph. In social relationship graphs, the nodes represent the users and edges represent the interactions between users. Through the analysis of the interactions, the user's opinions can b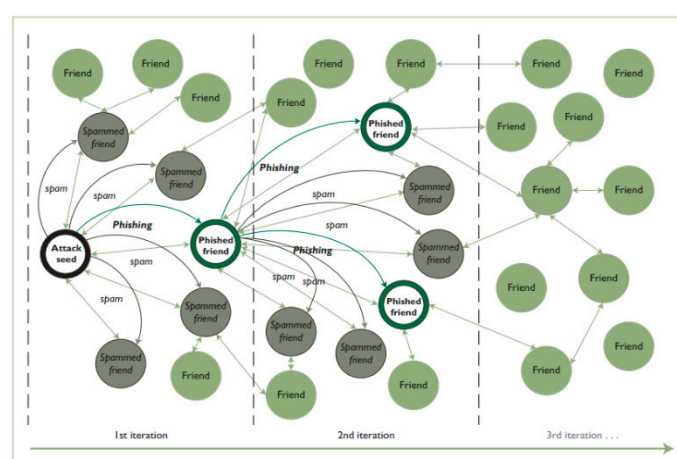e modelled. There exist direct relationships between the opinion and interactions [156]. An example to estimate the strengths of social relations among users via SN interactions is depicted in Figure 47.

Figure 47. Social network mining to estimate the strengths of social relations among users (Ref. [156]).

In addition to the above sixteen potential applications of the graphs in SNs, we present the SN problem comprehensive taxonomy in Figure 48. Graphs can be utilized in all of the given problems/phenomena of SNs for the modelling and analysis [157]. With the help of graphs, the modelling and analysis of SNs become much easier. Furthermore, the user's representation and modelling via graphs has number of advantages in term of summarizing a large dataset in a visual form, easy comparison between different datasets, better clarification of trends in the data than other representation such as tables, and at a glance estimation of the key values. In addition, the graphs hold more pieces of information about the entities such as node properties, node's degree, edge/node labels, and graph metrics (closeness, centralities, path lengths, cliques, subgraphs, reachability). Such valuable information plays a vital role in achieving many scientific and business objectives by analyzing the SN graphs.

Figure 48. Taxonomy of the online social networks (OSNs) analysis problems (Ref. [157]).

4. Summary and Discussion

GT concepts are potentially applicable to model/analyze the various problems of CS and SNs. In this paper, we summarize the uses o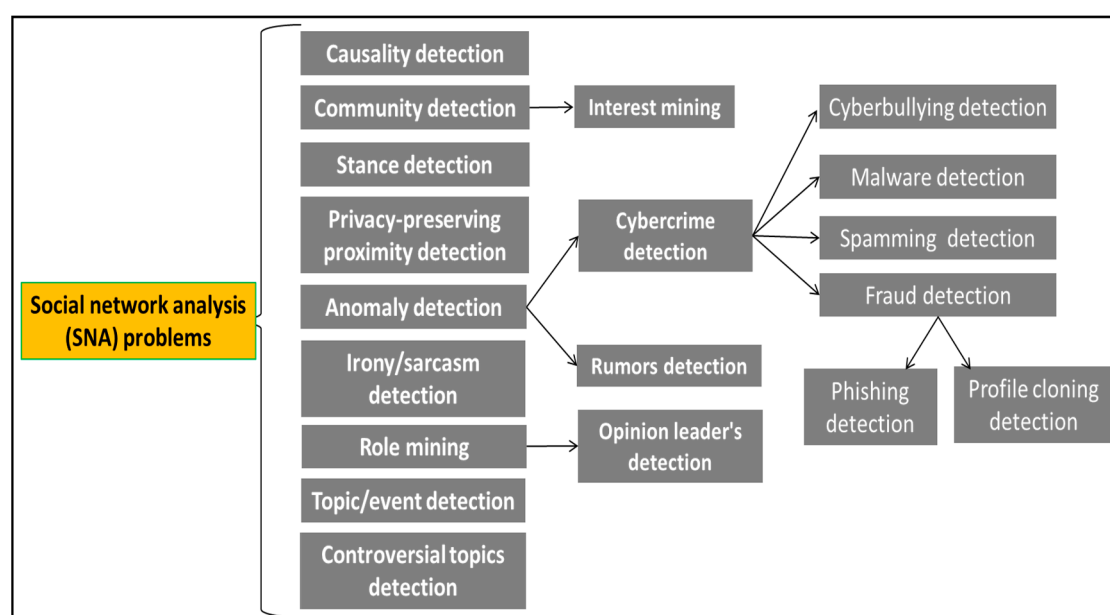f the GT in ten general applications related to CS, and sixteen applications of GT in SNs. We found that GT concepts are fundamental in SN analysis and modelling. Furthermore, we present examples with real-world cases. For example, in algorithm execution, the graphs can be used to model the statement and their connections with each other, the execution order of the algorithm's statements, the control flow, the data flow, and the algorithm testing. In service connectivity analysis, graphs can be used to analyze the links between different services, communication patterns, order of the call-backs, service request and response order, and execution order. In web documents clustering, graphs can be used to cluster relevant documents for queries to make the searching and information retrieval robust. In WSN, graphs can be used to represent the topology of node deployment, order of the values/data collection, faulty and non-faulty components representation, and sensing co-ordination and controls. In IoT domain, graphs can be used for information diffusion, data collection, flow analysis, monitoring of attacks, risk assessment, information retrieval, and collaboration among devices. In the blockchain domain, graphs can be used to represent the p2p network users, links between peers, peer communication analysis, peer interaction patterns, and peer status analysis. In computer vision, the graphs are used to perform the image analysis, corner detections, histogram representations, and linkages between moving objects.

SNA applications include marketing as well as risk and fraud detection. Marketing applications are customer buying patterns prediction and announcing the marketing campaigns. Since user group characteristics can impact buying rates/behaviors, firms may be able to guess and estimate sales using SNA to better understand group behavior and thereby individual behavior. Similarly, by identifying influencers (e.g., influential spreaders, opinion leaders) within groups, firms can launch marketing campaigns. The influence of a group member makes other members more likely to purchase the offering or change their opinion. On the risk and fraud detection front, SNA can be employed to detect money misuse and fraud involving credit cards (using people-buyer patterns or activities over websites). Applying SNA to online customer data created through online activities among students and faculty may allow us to understand which types of online activities are most beneficial for e-learning. GT can be used in SN security and privacy analysis. In addition, GT can be used in all SNA-related problems (see Figure 48).

Aside from the numerous applications of GT in CS and SN, the users/developer of GT face several challenges while processing complex graphs. We summarize a few unique technical challenges discussed by Sahu et al. [158]. The technical challenges related to graphs are presented in Figure 49 in brief.
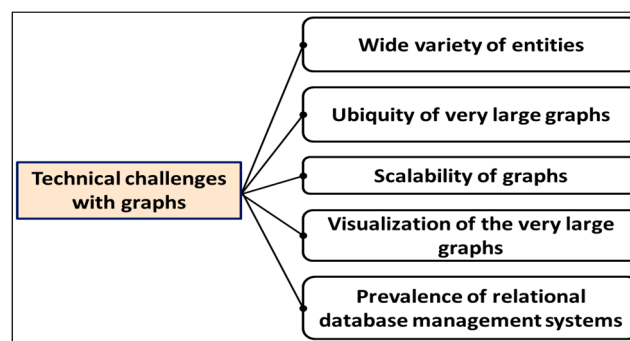

Figure 49. Technical challenges with the graph's handling/processing.

A brief discussion of each technical challenge listed in Figure 49 is explained below.

- Variety of entities: Generally, graphs represent a very wide variety of entities. However, many of them are not naturally distinguished to be used as vertices or edges. Interestingly, traditional companies

collected data comprised of products, quotes, orders, and transactions, which are stored easily in the relational systems. Meanwhile, the graph rendering, and representation is not straightforward. Hence, it demands a flexible storage and retrieval system in graphical form.

- Ubiquity: In many real-world problems, the size of graphs can be very large, often containing over a billion/trillion number of edges and nodes. These large graphs represent the entities and relationships between them. Therefore, their handling and processing is a big challenge. Organizations such as Google, Facebook, and Twitter are facing such problems.
- Scalability: The capability to process very large graphs containing many users' data efficiently is very challenging. Rendering of the large graphs needs extensive computing power and storage.
- Data visualization: Visualization is an important task in participants' graph processing. After scalability, the graph developers/users need appropriate visualization as their second most pressing challenge. The visualization of the large dataset containing many entities is extremely complex and selecting appropriate visualization is very challenging.
- Prevalence of relational database management systems (RDBMs): In the presence of RDBMS, sometimes it becomes difficult to choose the task modelling between relational and structural.

A number of technical challenges reported by Nadya et al. [159] have also been extensively studied in the recent past. According to the authors, the technical challenges related to the graph are:

- Few trusted datasets of relevant scales exist that allow for rigorous evaluation of detection techniques on the graph data. The description of the backgrounds (normal patterns and noise) and foregrounds (target and anomalous patterns) are still being formalized.
- Relationships of interest to many applications are highly dynamic. Dealing with large-scale dynamic graphs is an emerging research area, and significant efforts are needed to deal with such large-scale graphs.
- In practice, many graphs can be made from a given dataset. However, determining what the graph's representation will be the most effective for a particular task is a highly complex task.
- In some real-world problems, it is necessary to quickly generate multiple different graphs from the same dataset or from multiple datasets. Meanwhile, storage, representation, and data-access techniques are often hard-coded, making this a difficult, error-prone, and time-consuming task.
- Many graph algorithms have high computational complexity. Also, the complexity increases exponentially with the increase in number of entities. Many factors contribute to this inefficiency, for example, sparsity of data and poor data locality of operations.
- The detection theory concept for graphs is a new area of research, and for the most settings, the performance bounds have not been explored well. Furthermore, for most datasets related to networks, there is no fine-grained manner to specify what type of patterns will be explored (i.e., be detectable), and what type of patterns will be subsumed in the noise.

The technical challenges summarized above need significant attention from developers and researchers. In addition, there is an emerging need of the databases which can store the graphs. There exist some databases like Neo4j, but the scalability of such tools is not suited to large-scale applications involving multiple entities data such as Facebook user data. In addition, the graph rendering software does not scale well with the size of the graph. Thus, development of the tools, software, and libraries that can process large-scale graphs efficiently has become imperative.

5. Conclusion and Future Works

In this paper, we have presented a review of the works carried out in the field of computer science (CS) and social networks (SN) which employs the concepts of graph theory (GT). The contribution can assist many researchers from various perspectives by analyzing graph properties and choosing the right combination of the graphs according to their problem. The well-summarized potential applications of GT can assist beginner researchers to grasp its concepts and existing use in many diverse applications of CS and SN domains. In addition, we provided practical examples and explanations of

the potential use of GT to emphasize the importance of graphs in modern research. Apart from the novel applications of the graphs in CS and SN such as modelling of the network security breaches, modelling user interactions with e-commerce sites, network topology modelling, scheduling-related problems, network routing analysis, operating environment modelling in robotics for pathfinding, network traffic flow analysis, effective representation of wireless sensor networks, SN user's interaction/relationship modelling, topic of interest modelling, user's community clustering/detection, information diffusion, opinion leader detection, user behavior analysis, privacy-preserving user data publishing, cybercrime detection, and influence/trust modeling, graphs can be used in modelling vehicular networks and provision of digital signage (i.e., digital contents) on the roads. Graphs can also be used for a vehicle's mobility analysis, audio/video content dissemination, vehicle-to-vehicle routing, capturing information about traffic flow, and/or for safety indications on the roads. In addition, they can be applied to assist in executing traffic management and routing plans. In the future, it will be imperative to devise new graphical tools and applications for the vehicular network management due to the emergence of smart cities.

In the future, we plan to extend the results in describing the uses of the GT in specific sectors such as healthcare, green energy environment, smart home environment, advanced SN user analysis, federated learning, and other related areas. In addition, we intend to investigate the GT potential applications in other disciplines of science including chemistry, biology, and physics. The intent analysis [160] by fusing multiple graphs data and analysis, is a promising area to be explored further leveraging the graphs. Finally, we intend to explore the usage of GT in data science, which has become a very popular area of research in recent years.

Author Contributions: All authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sarma, S.V.M. Applications of Graph Theory in Human Life. *Int. J. Comput. Appl.* **2012**, *1*, 21–30.
2. Journal, I.; Core, O.; Ijcem, M. A study of Vertex—Edge Coloring Techniques with Application. *Int. J. Core Eng. Manag.* **2014**, *1*, 27–32.
3. Voloshin, V.I. *Introduction to Graph Theory*; Nova Science Publishers: New York, NY, USA, 2009; pp. 1–144.
4. Kocay, W.; Kreher, D.L. *Graphs, Algorithms, Optimization*; Chapman & Hall/CRC Press: Boca Raton, FL, USA, 2017; pp. 1–483.
5. Mondal, B.; De, K. Overview Applications of Graph Theory in Real Field. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.* **2017**, *2*, 751–759.
6. Robertson, N.; Seymour, P.; Thomas, R. Quickly excluding a planar graph. *J. Comb. Theory Ser. B* **1994**, *62*, 323–348. [CrossRef]
7. Kaundal, K. Applications of Graph Theory in Everyday Life and Technology. *Imp. J. Interdiscip. Res.* **2017**, *3*, 892–894.
8. Nagendram, N.V. A Note on Sufficient Cindition on Hamiltonian Path to Complete Graphs (SC-HPCG). *IJMA* **2011**, *2*, 1–6.
9. Gärtner, T.; Flach, P.; Wrobel, S. On graph kernels: Hardness results and efficient alternatives. *Lect. Notes Artif. Intell. (Subser. Lect. Notes Comput. Sci.* **2003**, *2777*, 129–143.
10. Bisen, S.K. Application of Graph Theory in Transportation Networks. *Int. J. Sci. Res. Manag.* **2017**, *5*, 10–12. [CrossRef]
11. Tyagi, S.S. *Statical Analysis of Social Network*; JUIT (Jaypee university of information technology): Himachal Pradesh, India, 2014; pp. 1–99.
12. Plummer, M.D. Some covering concepts in graphs. *J. Comb. Theory* **1970**, *8*, 91–98. [CrossRef]

13. Sciences, D.M. A Survey on some Applications of Graph Theory in Cryptography. *J. Discret. Math. Sci. Cryptogr.* **2015**, *18*, 209–217.

14. Ganzha, M.; Maciaszek, L. *Position Papers of the 2019 Federated Conference on Computer Science and Information Systems*; Springer: Leipzig, Germany, 2019; p. 19.

15. Polak, M.; Roma, U. On the applications of Extremal Graph Theory to Coding Theory and Cryptography. *Electron. Notes Discret. Math.* **2013**, *43*, 329–342. [CrossRef]

16. Jaromczyk, J.W.; Lonc, Z.; Truszczy, M. Constructions of asymptotically shortest k-radius sequences. *J. Comb. Theory Ser. A* **2012**, *119*, 731–746. [CrossRef]

17. Yuan, M.; Chen, L.; Yu, P.S.; Mei, H. Privacy preserving graph publication in a distributed environment. *World Wide Web* **2015**, *18*, 1481–1517. [CrossRef]

18. Iturria-medina, Y.; Sotero, R.C.; Canales-rodríguez, E.J.; Alemán-gómez, Y.; Melie-garcía, L. Studying the human brain anatomical network via diffusion-weighted MRI and Graph Theory. *Neuroimage* **2008**, *40*, 1064–1076. [CrossRef] [PubMed]

19. Minor, E.S.; Urban, D.L. A Graph-Theory Framework for Evaluating Landscape Connectivity and Conservation Planning. *Conserv. Biol.* **2008**, *22*, 297–307. [CrossRef] [PubMed]

20. Chrysochoos, A.; Louche, H. An infrared image processing to analyse the calorific effects accompanying strain localisation. *Int. J. Eng. Sci.* **2000**, *16*, 1759–1788.

21. Salembier, P.; Garrido, L. Binary Partition Tree as an Efficient Representation for Image Processing, Segmentation, and Information Retrieval. *IEEE Trans. Image Process.* **2000**, *9*, 561–576. [CrossRef]

22. Campbell, W.M.; Dagli, C.K.; Weinstein, C.J. Social network analysis with content and graphs. *Linc. Lab. J.* **2013**, *20*, 61–81.

23. Shuman, D.I.; Narang, S.K.; Frossard, P.; Ortega, A.; Vandergheynst, P. The Emerging Field of Signal Processing. *IEEE Signal Process. Mag.* **2013**, *30*, 83–98. [CrossRef]

24. Cordero, P.; Enciso, M.; Mora, A.; Ojeda-aciego, M.; Rossi, C. Knowledge-Based Systems Knowledge discovery in social networks by using a logic-based treatment of implications. *Knowl.-Based Syst.* **2015**, *87*, 16–25. [CrossRef]

25. Lee, J. Kinematic Analysis of Tendon-Driven Robotic Mechanisms Using Graph Theory. *ASME J. Mech. Trans. Automat. DXes.* **1989**, *111*, 59–65.

26. Demange, M.; Ekim, T.; de Werra, D. Discrete Optimization A tutorial on the use of graph coloring for some problems in robotics. *Eur. J. Oper. Res.* **2009**, *192*, 41–55. [CrossRef]

27. Derrible, S.; Kennedy, C. Network Analysis of World Subway Systems Using Updated Graph Theory. *Transp. Res. Rec.* **2009**, *2112*, 17–25. [CrossRef]

28. De Klerk, E. Exploiting special structure in semidefinite programming: A survey of theory and applications. *Eur. J. Oper. Res.* **2010**, *201*, 1–10. [CrossRef]

29. Man, A.; So, C.; Ye, Y. A Semidefinite Programming Approach to Tensegrity Theory and Realizability of Graphs. In Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, SMiami, FL, USA, 22–26 January 2006; Volume 6, pp. 1–17.

30. Saerens, M.; Fouss, F.; Yen, L.; Dupont, P. The Principal Components Analysis of a Graph, and Its Relationships to Spectral Clustering. In Proceedings of the European conference on machine learning, Pisa, Italy, 20–24 September 2004; pp. 371–383.

31. Qiantt, Y.; Suent, C.Y.; M, Q.H.G. Clustering Combination Method. In Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain, 3–7 September 2000; pp. 732–735.

32. Brás, H.; Brito, P.; Pinto, J. A partitional clustering algorithm validated by a clustering tendency index based on graph theory. *Pattern Recognit.* **2006**, *39*, 776–788. [CrossRef]

33. Brandes, U.; Gaertler, M.; Wagner, D. Experiments on Graph Clustering Algorithms. In Proceedings of the European Symposium on Algorithms, Copenhagen, Denmark, 7–9 September 2009; pp. 568–579.

34. Dodel, S.; Herrmann, J.M.; Geisel, T. Functional connectivity by cross-correlation clustering. *Neurocomputing* **2002**, *46*, 1065–1070. [CrossRef]

35. Pavan, M.; Pelillo, M.; Informatica, D.; Torino, V.; Mestre, V. A New Graph-Theoretic Approach to Clustering and Segmentation. In Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, WI, USA, 18–20 June 2003.

36. Durand, G.; Belacel, N.; Laplante, F. Graph theory based model for learning path recommendation. *Inf. Sci.* **2013**, *251*, 10–21. [CrossRef]

37.　Graves, M.; Bergeman, E.R.; Lawrence, C.B. A Graph-Theoretic Data Model for Genome Mapping Databases. In Proceedings of the Twenty-Eighth Annual Hawaii International Conference on System Sciences, Wailea, HI, USA, 3–6 January 1995.

38.　Kontokosta, C.E. Big Data + Big Cities: Graph Signals of Urban Air Pollution. *IEEE Signal Process. Mag.* **2014**, *31*, 130–136.

39.　Siqueira, S.; Eduardo, C.; Junior, B.; Comfort, W.E.; Rohde, L.A.; Sato, J.R. Abnormal Functional Resting-State Networks in ADHD: Graph Theory and Pattern Recognition Analysis of fMRI Data. *BioMed Res. Int.* **2014**, *2014*, 380531.

40.　Riaz, F.; Ali, K.M. Applications of Graph Theory in Computer Science. In Proceedings of the 2011 Third International Conference on Computational Intelligence, Communication Systems and Networks, Bali, Indonesia, 26–28 July 2011; pp. 142–145.

41.　Appel, K. Applications of Graph Theory in Computer Science an Overview. *Int. J. Eng. Sci. Technol.* **2010**, *2*, 4610–4621.

42.　Durgaprasad, D.; Snehadivya, M.; Kavitha, S. Applications of Computer Science Based on Graph theory. *Int. J. Eng. Sci.* **2017**, *6*, 1116–1122.

43.　Liu, Y.; Safavi, T.; Dighe, A.; Koutra, D. Graph Summarization Methods and Applications: A Survey. *ACM Comput. Surv.* **2018**, *51*, 1–34. [CrossRef]

44.　Yu, Q.; Du, Y.; Chen, J.; Sui, J.; Adalē, T.; Pearlson, G.D.; Calhoun, V.D. Application of Graph Theory to Assess Static and Dynamic Brain Connectivity: Approaches for Building Brain Graphs. *Proc. IEEE* **2018**, *106*, 886–906. [CrossRef] [PubMed]

45.　Sporns, O. Graph theory methods: Applications in brain networks. *Dialogues Clin. Neurosci.* **2018**, *20*, 111. [PubMed]

46.　Goyal, P.; Ferrara, E. Knowle dge-Base d Systems Graph emb e dding techniques, applications, and performance: A survey. *Knowledge-Based Syst.* **2018**, *151*, 78–94. [CrossRef]

47.　Bondy, J.A.; Murty, U.S.R. *Graph Theory with Applications*; Oxford: New York, NY, USA; Amsterdam, The Netherlands; Oxford, UK, 1982.

48.　Farahani, F.V.; Karwowski, W.; Lighthall, N.R. Application of Graph Theory for Identifying Connectivity Patterns in Human Brain Networks: A Systematic Review. *Front. Neurosci.* **2019**, *13*, 585. [CrossRef]

49.　Gupta, S.; Singh, M.; Madan, A.K. Application of graph theory: Relationship of eccentric connectivity index and Wiener's index with anti-inflammatory activity. *J. Math. Anal. Appl.* **2002**, *266*, 259–268. [CrossRef]

50.　Pavlopoulos, G.A.; Secrier, M.; Moschopoulos, C.N.; Soldatos, T.G.; Kossida, S.; Aerts, J.; Schneider, R.; Bagos, P.G. Using graph theory to analyze biological networks. *BioData Min.* **2011**, *4*, 10. [CrossRef]

51.　Hansen, P.; Mélot, H. Computers and discovery in algebraic graph theory. *Linear Algebra Appl.* **2002**, *356*, 211–230. [CrossRef]

52.　Cvetković, D.; Simić, S. Graph spectra in Computer Science. *Linear Algebra Appl.* **2011**, *434*, 1545–1562. [CrossRef]

53.　PalSingh, R.; Vandana, V. Application of Graph Theory in Computer Science and Engineering. *Int. J. Comput. Appl.* **2014**, *104*, 10–13. [CrossRef]

54.　Spielman, D.A.; Sachs, H.; Theory, A.G.; Godsil, C. Spectral Graph Theory and its Applications. In Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, Providence, RI, USA, 21–23 October 2007; pp. 29–38.

55.　Agarwal, S.; Mehta, S. Social Influence Maximization Using Genetic Algorithm with Dynamic Probabilities. In Proceedings of the 2018 Eleventh International Conference on Contemporary Computing (IC3), Noida, India, 2–4 August 2018; pp. 1–6.

56.　Science, C. Related: An R package for analysing pairwise relatedness from codominant molecular markers. *Mol. Ecol. Resour.* **2015**, *15*, 557–561.

57.　Hsiung, P.; Wang, F. A State Graph Manipulator Tool for Real-Time System Specification and Verification. In Proceedings of the Fifth International Conference on Real-Time Computing Systems and Applications, Hiroshima, Japan, 27–29 October 1998.

58.　Hurd, J. Composable Packages for Higher Order Logic Theories. In Proceedings of the Verification Workshop, Edinburgh, UK, 20–21 July 2010; Volume 3, pp. 79–93.

59. Valdes, R. *The Competitive Dynamics of the Consumer Web: Five Graphs Deliver a Sustainable Advantage*; Gartner: Stamford, CT, USA, 2012. Available online: https://www.gartner.com/doc/2081316/competitive-dynamics-consumer-web-graphs (accessed on 11 January 2019).

60. Wang, J.; Cong, G.; Zhao, W.X.; Li, X. Mining user intents in Twitter: A semi-supervised approach to inferring intent categories for tweets. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 318–324.

61. Reilly, K.D. The GPSS-GASP Combined (GGC) System. *Int. J. Comput. Inf. Sci.* **1983**, *12*, 111–136.

62. Chell, E.; Mercer, M.R. CADTOOLS: A CAD algorithm development system. In Proceedings of the 22nd ACM/IEEE Design Automation Conference, Las Vegas, NV, USA, 23 June 1985; pp. 658–666.

63. Rheinboldt, W.C.; Basilli, V.R.; Charles, K. Mesztenyi. On a programming language for graph algorithms. *BIT Numer. Math.* **1972**, *12*, 220–241. [CrossRef]

64. Mokhtari, H.; Dadgar, M. A Flexible Job Shop Scheduling Problem with Controllable Processing Times to Optimize Total Cost of Delay and Processing. *Int. J. Supply Oper. Manag.* **2015**, *2*, 871.

65. Dawood, H.A.; William, R. Graph T Theory and Cyber Security. In Proceedings of the 2014 3rd International Conference on Advanced Computer Science Applications and Technologies, Amman, Jordan, 29–30 December 2014; pp. 90–96.

66. Majeed, A.; Farooq, R.; Masoom, A.; Nadeem, A. Near—Miss situation based visual analysis of SIEM rules for real time network security monitoring. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 1509–1526. [CrossRef]

67. Majeed, A.; Rauf, I. MVC Architecture: A Detailed Insight to the Modern Web Applications Development. *Peer Rev. J. Solar Photoenergy Syst.* **2018**, *1*, 1–7.

68. Schenker, A.; Last, M.; Bunke, H.; Kandel, A. Chapter? Clustering of Web Documents Using a Graph Model. In *Web Document Analysis: Challenges and Opportunities*; World Scientific Publishing Company: Singapore, 2003; pp. 3–18.

69. Jain, B.J.; Obermayer, K. Graph quantization. *Comput. Vis. Image Underst.* **2011**, *115*, 946–961. [CrossRef]

70. Kalogeratos, A.; Likas, A. Data & Knowledge Engineering Document clustering using synthetic cluster prototypes. *Data Knowl. Eng.* **2011**, *70*, 284–306.

71. Jarvenpaa, S.L.; Todd, P.A. Consumer reactions to electronic shopping on the World Wide Web. *Int. J. Electron. Commer.* **1996**, *1*, 59–88. [CrossRef]

72. Zhao, R.; Grosky, W.I. Narrowing the Semantic Gap—Improved Text-Based Web Document Retrieval Using Visual Features. *IEEE Trans. Multimed.* **2002**, *4*, 189–200. [CrossRef]

73. Zeithaml, V.A.; Parasuraman, A.; Malhotra, A. Service quality delivery through web sites: a critical review of extant knowledge. *J. Acad. Mark. Sci.* **2002**, *30*, 362–375. [CrossRef]

74. Schenker, A.; Last, M.; Bunke, H.; Kandel, A. Graph Representations for Web Document Clustering. In *Iberian Conference on Pattern Recognition and Image Analysis*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 935–942.

75. Madan, R.; Cui, S.; Lall, S.; Goldsmith, A. Modeling and Optimization of Transmission Schemes in Energy Constrained Wireless Sensor Networks. *IEEE/ACM Trans. Netw.* **2007**, *15*, 1359–1372. [CrossRef]

76. Du, C.; Shao, S.; Qi, F.; Meng, L. Multi-requests satisfied based on energy optimization for the service composition in wireless sensor network. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719879049. [CrossRef]

77. Kumar, J.S.; Zaveri, M.A. Graph based clustering for two-tier architecture in Internet of things. In Proceedings of the 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Chengdu, China, 15–18 December 2016; pp. 229–233.

78. Shivraj, V.L.; Rajan, M.A.; Balamuralidhar, P. A Graph theory based Generic Risk Assessment framework for Internet of Things (IoT). In Proceedings of the 2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bhubaneswar, India, 17–20 December 2017; pp. 1–6.

79. Yao, B.; Liu, X.; Zhang, W.; Chen, X. Applying Graph Theory To The Internet of Things. In Proceedings of the 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, Zhangjiajie, China, 13–15 November 2013; pp. 2354–2361.

80. Ning, Z.; Wang, X.; Member, S. A Social-Aware Group Formation Framework for Information Diffusion in Narrowband Internet of Things. *IEEE Internet Things J.* **2018**, *5*, 1527–1538. [CrossRef]

81. Rathore, M.M.; Ahmad, A.; Paul, A. Efficient Graph-Oriented Smart Transportation using Internet of Things generated Big Data. In Proceedings of the 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Bangkok, Thailand, 23–27 November 2015; pp. 512–519.

82. Wang, H.; Chen, Z.; Zhao, J.; Di, X.; Liu, D.A.N. A Vulnerability Assessment Method in Industrial Internet of Things Based on Attack Graph and Maximum Flow. *IEEE Access* **2018**, *6*, 8599–8609. [CrossRef]

83. Chen, P.; Member, S.; Cheng, S.; Chen, K. Information Fusion to Defend Intentional Attack in Internet of Things. *IEEE Internet Things J.* **2014**, *1*, 337–348. [CrossRef]

84. Abdellatif, K.; Abdelmouttalib, C. Graph-Based Computing Resource Allocation for Mobile Blockchain. In Proceedings of the 2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM), Marrakesh, Morocco, 16–19 October 2018; pp. 1–4.

85. Akcora, C.G.; Gel, Y.R.; Kantarcioglu, M. 1 Blockchain: A Graph Primer. *arXiv* **2017**, arXiv:1708.08749.

86. Salah, K.; Member, S.; Rehman, M.H.U.R. Blockchain for AI: Review and Open Research Challenges. *IEEE Access* **2019**, *7*, 10127–10149. [CrossRef]

87. Wang, S.; Wang, J.; Wang, X.; Qiu, T.; Yuan, Y.; Ouyang, L.; Guo, Y.; Wang, F.Y. Blockchain-Powered Parallel Healthcare Systems Based on the ACP Approach. *IEEE Trans. Comput. Soc. Syst.* **2018**, *5*, 942–950. [CrossRef]

88. Di, D.; Maesa, F.; Marino, A.; Ricci, L. Uncovering the Bitcoin blockchain: An analysis of the full users graph. In Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Montreal, QC, Canada, 17–19 October 2016; pp. 537–546.

89. He, Y.; Gao, C.; Sang, N.; Qu, Z.; Han, J. Neurocomputing Graph coloring based surveillance video synopsis. *Neurocomputing* **2017**, *225*, 64–79. [CrossRef]

90. Feng, P.; Xu, C.; Zhao, Z.; Liu, F.; Yuan, C.; Wang, T. Neurocomputing Sparse representation combined with context information for visual tracking. *Neurocomputing* **2017**, *225*, 92–102. [CrossRef]

91. Malyshev, D.S. The weighted coloring problem for two graph classes characterized by small forbidden induced structures. *Discret. Appl. Math.* **2018**, *247*, 423–432. [CrossRef]

92. Dabrowski, K.K.; Lozin, V.; Raman, R.; Ries, B. Colouring vertices of triangle-free graphs without forests. *Discret. Math.* **2012**, *312*, 1372–1385. [CrossRef]

93. Dickinson, S. Introduction to the Special Section on Graph Algorithms in Computer Vision. *IEEE Trans. Pattern Anal. Mach. Intell.* **2001**, *10*, 1049–1052. [CrossRef]

94. Durst, C.; Durst, C. Online Social Networks, Social Capital and Health- related Behaviors: A State-of-the-art Analysis. *Commun. Assoc. Inf. Syst.* **2013**, *32*, 5. [CrossRef]

95. Jin, R.; Zhang, H.; Zhang, Y. The social negative mood index for social networks. In Proceedings of the 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), Guangzhou, Chin, 18–21 June 2018; pp. 1–5.

96. Kolli, N.; Balakrishnan, N. Analysis of e-mail Communication Using a Social Network Framework for Crisis Detection in an Organization Science Direct. *Procedia—Soc. Behav. Sci.* **2013**, *100*, 57–67. [CrossRef]

97. He, C.; Li, H.; Fei, X.; Tang, Y.; Zhu, J. A Topic Community-based Method for Friend Recommendation in Online Social Networks via Joint Nonnegative Matrix Factorization. In Proceedings of the 2015 Third International Conference on Advanced Cloud and Big Data, Yangzhou, China, 30 October–1 November 2015; pp. 28–35.

98. Wieringa, J.; Kannan, P.K.; Ma, X.; Reutterer, T.; Risselada, H.; Skiera, B. Data analytics in a privacy-concerned world. *J. Bus. Res.* **2019**. [CrossRef]

99. Liu, F.; Joo, H. Expert Systems with Applications Use of social network information to enhance collaborative filtering performance. *Expert Syst. Appl.* **2010**, *37*, 4772–4778. [CrossRef]

100. Liu, D.; Wang, L.; Zheng, J.; Ning, K.; Zhang, L. Social Network. In Proceedings of the 2013 IEEE International Conference on Services Computing, Santa Clara, CA, USA, 28 June–3 July 2013; pp. 368–375.

101. Beach, A.; Gartrell, M.; Han, R. Social-K: Real-Time K-Anonymity Guarantees for Social Network Applications. In Proceedings of the 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Mannheim, Germany, 29 March–2 April 2010; pp. 600–606.

102. Zin, T.T.; Tin, P.; Hama, H.; Toriu, T. Knowledge based Social Network Applications to Disaster Event Analysis. In Proceedings of the International Multi Conference of Engineers and Computer Scientists, Hong Kong, China, 13–15 March 2013.

103. Li, Y.; Hsiao, H.; Lee, Y. Infor mation Sciences Recommending social network applications via social filtering mechanisms. *Inf. Sci.* **2013**, *239*, 18–30. [CrossRef]

104. Zhang, N. Preserving Relation Privacy in Online Social Network Data. *IEEE Internet Comput.* **2011**, *15*, 35–42.

105. Izuan, M.; Ninggal, H. Attack Vector Analysis and Privacy-Preserving Social Network Data Publishing. In Proceedings of the 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications, Changsha, China, 16–18 November 2011; pp. 847–852.

106. Chow, W.S.; Chan, L.S. Information & Management Social network, social trust and shared goals in organizational knowledge sharing. *Inf. Manag.* **2008**, *45*, 458–465.

107. Li, P.; Yu, J.; Liu, J.; Zhou, D.; Cao, B. Generating weighted social networks using multigraph. *Phys. A Stat. Mech. Its Appl.* **2020**, *539*, 122894. [CrossRef]

108. Zhou, B. A Brief Survey on Anonymization Techniques for Privacy Preserving Publishing of Social Network Data. *ACM Sigkdd Explor. Newsl.* **2008**, *10*, 12–22. [CrossRef]

109. Li, M.; Wang, X.; Gao, K.; Zhang, S. A Survey on Information Diffusion in Online Social Networks: Models and Methods. *Information* **2017**, *8*, 118.

110. Tabrizi, S.A.; Shakery, A.; Asadpour, M.; Abbasi, M.; Tavallaie, M.A. Personalized PageRank Clustering: A graph clustering algorithm based on random walks. *Phys. A Stat. Mech. Appl.* **2013**, *392*, 5772–5785. [CrossRef]

111. Rehman, A.U.; Jiang, A.; Rehman, A.; Paul, A.; Sadiq, M.T. Identification and role of opinion leaders in information diffusion for online discussion network. *J. Ambient. Intell. Humaniz. Comput.* **2020**, 1–13. [CrossRef]

112. Guille, A.; Hacid, H.; Zighed, D.A. Information Diffusion in Online Social Networks: A Survey. *ACM Sigmod Rec.* **2013**, *42*, 17–28. [CrossRef]

113. Bian, T.; Deng, Y. Identifying influential nodes in complex networks: A node information dimension approach. *Chaos: Interdiscip. J. Nonlinear Sci.* **2018**, *28*, 043109. [CrossRef] [PubMed]

114. Li, C.; Wang, L.; Sun, S.; Xia, C. Identification of influential spreaders based on classified neighbors in real-world complex networks. *Appl. Math. Comput.* **2018**, *320*, 512–523. [CrossRef]

115. Zeng, A.; Zhang, C. Ranking spreaders by decomposing complex networks. *Phys. Lett. A* **2013**, *377*, 1031–1035. [CrossRef]

116. Mao, C. Research Article A Comprehensive Algorithm for Evaluating Node Influences in Social Networks Based on Preference Analysis and Random Walk. *Complexity* **2018**, *2018*, 1528341. [CrossRef]

117. Zheng, Y.; Xu, J. A trust transitivity model for group decision making in social network with intuitionistic fuzzy information. *Filomat* **2018**, *32*, 1937–1945. [CrossRef]

118. Davies, R.; Ghosh-dastidar, U.; Knisley, J.; Samyono, W. *Function: Identifying Biologically Relevant Clusters with Graph Spectral Methods*; Elsevier Inc.: Geneva, Switzerland, 2019.

119. Cacheda, F.; Fernandez, D.; Novoa, F.J.; Carneiro, V. Early Detection of Depression: Social Network Analysis and Random Forest Techniques. *J. Med. Internet Res.* **2019**, *21*, e12554. [CrossRef] [PubMed]

120. Lee, S.; Cha, Y.; Han, S.; Hyun, C. Application of Association Rule Mining and Social Network Analysis for Understanding Causality of Construction Defects. *Sustainability* **2019**, *11*, 618. [CrossRef]

121. Atzmueller, M. Modeling and Mining Feature-Rich Networks. In Companion Proceedings of the 2019 World Wide Web Conference, San Francisco, CA, USA, May 2019; pp. 16–17.

122. Anufrieva, E.; Borodina, E. Analysis of the social well-being of urban citizens: Gender aspect in the conditions of digital transformation. In *Proceedings of the 1st International Scientific Practical Conference the Individual and Society in the Modern Geopolitical Environmentvol*; Atlantis Press: Prague, Czech Republic, 2019; pp. 34–39.

123. Mahmoudi, A.; Ridzwan, M.; Azuraliza, Y.; Bakar, A. A new method to discretize time to identify the milestones of online social networks. *Soc. Netw. Anal. Min.* **2018**, *8*, 34. [CrossRef]

124. Dekker, A. Centrality in social networks: Theoretical and simulation approaches. In Proceedings of the SimTect, Melbourne, Australia, 12–15 May 2008; pp. 33–38.

125. Shelke, S.; Attar, V. Source detection of rumor in social network—A review. *Online Soc. Netw. Media* **2019**, *9*, 30–42. [CrossRef]

126. Shiokawa, H.; Fujiwara, Y.; Onizuka, M. Fast Algorithm for Modularity-Based Graph Clustering. In Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, Bellevue, WA, USA, 14–18 July 2013; pp. 1170–1176.

127. Radley, S.; Sybi, C.J.; Premkumar, K. Multi Information Amount Movement Aware—Routing in FANET: Flying Ad-hoc Networks. In *Mobile Networks and Applications*; Springer: New York, USA, 2019.

128. Newman, M.E.J. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA* **2006**, *103*, 8577–8582. [CrossRef]

129. Chen, Z.; Liu, B. Mining Topics in Documents: Standing on the Shoulders of Big Data. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 1116–1125.

130. Poria, S.; Cambria, E.; Gelbukh, A. Knowle dge-Base d Systems Aspect extraction for opinion mining with a deep convolutional neural network. *Knowl.-Based Syst.* **2016**, *108*, 42–49. [CrossRef]

131. Chen, A.Z.; Mukherjee, M.; Hsu, M. Castellanos, Exploiting Domain Knowledge in Aspect Extraction. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1655–1667.

132. Xing, L.; Deng, K.; Wu, H.; Xie, P.; Gao, J. Behavioral Habits-Based User Identification across Social Networks. *Symmetry* **2019**, *11*, 1134. [CrossRef]

133. Xing, L.; Deng, K.; Wu, H.; Xie, P.; Zhao, H.V.; Gao, F. A Survey of Across Social Networks User Identification. *IEEE Access* **2019**, *7*, 137472–137488. [CrossRef]

134. Liao, L.; He, X.; Zhang, H.; Chua, T.S. Attributed social network embedding. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2257–2270. [CrossRef]

135. Ok, M.; Lee, J.S.; Kim, Y.B. Recommendation framework combining user interests with fashion trends in apparel online shopping. *Appl. Sci.* **2019**, *9*, 2634. [CrossRef]

136. Type, I.; Dissertation, E. *Graph-Based Analysis for E-In the Graduate College*; Academic Press: New York, NY, USA, 2019.

137. Feng, Z.; Lien, J.W.; Zheng, J. Keeping up with the Neighbors: Social Interaction in a Production Economy. *Mathematics* **2018**, *6*, 162. [CrossRef]

138. Shi, C. A Survey of Heterogeneous Information Network Analysis. *IEEE Trans. Knowl. Data Eng.* **2016**, *29*, 17–37. [CrossRef]

139. Yang, D.; Qu, B.; Cudre-mauroux, P. Privacy-Preserving Social Media Data Publishing for Personalized Ranking-Based Recommendation. *IEEE Trans. Knowl. Data Eng.* **2019**, *31*, 507–520. [CrossRef]

140. Abawajy, J.H.; Member, S.; Izuan, M.; Ninggal, H.; Herawan, T. Privacy Preserving Social Network Data Publication. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1974–1997. [CrossRef]

141. Xu, L.E.I.; Jiang, C.; Wang, J. Information Security in Big Data: Privacy and Data Mining. *IEEE Access* **2014**, *2*, 1149–1176.

142. Zhou, P.; Wang, K.; Guo, L. A Privacy-Preserving Distributed Contextual Federated Online Learning Framework with Big Data Support in Social Recommender Systems. *IEEE Trans. Knowl. Data Eng.* **2019**. [CrossRef]

143. Majeed, A. Attribute-centric anonymization scheme for improving user privacy and utility of publishing e-health data. *J. King Saud Univ.-Comput. Inf. Sci.* **2019**, *31*, 426–435. [CrossRef]

144. Wang, S.; Tsai, Z.; Hong, T.; Ting, I.; Engineering, I. A Nonymizing Shortest Paths on Social Network Graphs 1 Introduction. In Proceedings of the Asian Conference on Intelligent Information and Database Systems, Daegu, Korea, 20–22 April 2011.

145. Kiabod, M.; Dehkordi, M.N.; Barekatain, B. TSRAM: A time-saving k-degree anonymization method in social network. *Expert Syst. Appl.* **2019**, *125*, 378–396. [CrossRef]

146. Herrera-joancomartí, J.C.J. A survey of graph-modification techniques for privacy-preserving on networks. *Artif. Intell. Rev.* **2017**, *47*, 341–366.

147. Bhattacharya, M. Preserving Privacy in Social Network Graph with K-anonymize Degree Sequence Generation. In Proceedings of the 2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Kathmandu, Nepal, 15–17 December 2015.

148. Liu, P.; Li, X. An Improved Privacy Preserving Algorithm for Publishing Social Network Data. In Proceedings of the 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, Zhangjiajie, China, 13–15 November 2013; pp. 888–895.

149. Madan, S. A Privacy Preserving Scheme for Big data Publishing in the Cloud using k-Anonymization and Hybridized Optimization Algorithm. In Proceedings of the 2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET), Kottayam, India, 21–22 December 2018; pp. 1–7.

150. Chakraborty, S.; Ambooken, J.G.; Tripathy, B.K.; Purushotham, S. Analysis and performance enhancement to achieve recursive (c, l) diversity anonymization in social networks. *Trans. Data Priv.* **2015**, *8*, 173–215.

151. Casas-Roma, J. An evaluation of vertex and edge modification techniques for privacy-preserving on graphs. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *11*, 1–17. [CrossRef]

152. Moriano, P.; Finke, J.; Ahn, Y.Y. Community-Based Event Detection in Temporal Networks. *Sci. Rep.* **2019**, *9*, 1–9. [CrossRef]

153. Zheleva, E.; Getoor, L. Social Network Data Analytics. *Soc. Netw. Data Anal.* **2011**, 196–210. Available online: https://link.springer.com/chapter/10.1007/978-1-4419-8462-3_10 (accessed on 20 February 2020).

154. Kayes, I.; Iamnitchi, A. Privacy and security in online social networks: A survey. *Online Soc. Netw. Media* **2017**, *3*, 1–21. [CrossRef]

155. Huber, M.; Mulazzani, M.; Weippl, E.; Kitzler, G.; Goluch, S. Friend-in-the-middle attacks: Exploiting social networking sites for spam. *IEEE Internet Comput.* **2011**, *15*, 28–34. [CrossRef]

156. Yeung, A.C.M.A.; Iwata, T. Research on social network mining and its future development. *NTT Technol. Rev.* **2011**, *9*, 1–4.

157. Can, U.; Alatas, B. A new direction in social network analysis: Online social network analysis problems and applications. *Phys. A Stat. Mech. Appl.* **2019**, *535*, 122372. [CrossRef]

158. Sahu, S.; Mhedhbi, A.; Salihoglu, S.; Lin, J.; Özsu, M.T. The ubiquity of large graphs and surprising challenges of graph processing: Extended survey. *VLDB J.* **2019**, *29*, 1–24. [CrossRef]

159. Bliss, N.T.; Schmidt, M.C. Confronting the Challenges of Graphs and Networks. *Linc. Lab. J.* **2013**, *20*, 4–9.

160. Ren, X.; Wang, Y.; Yu, X.; Yan, J.; Chen, Z.; Han, J. Heterogeneous graph-based intent learning with queries, web pages and Wikipedia concepts. In Proceedings of the 7th ACM International Conference on Web Search and Data Mining, New York, NY, USA, 24–28 February 2014; pp. 23–32.