Graph Aggregation : Application to Social Networks

Amine Louati* Marie-Aude Aufaure**
Yves Lechevallier***

*University Paris-Dauphine - Lamsade,
Place du Maréchal de Lattre de Tassigny, F-75775 Paris 16 cedex, France
Amine.louati@dauphine.lamsade.fr
**Laboratoire MAS Ecole Centrale de Paris Grande Voie des Vignes
Chatenay-Malabry, France
Marie-Aude.Aufaure@ecp.fr,
***INRIA-Rocquencourt,Domaine de Voluceau 78150 Rocquencourt
Yves.Lechevallier@inria.fr

Abstract. In the enterprise context, people need to exploit and mainly visualize different types of interactions between heterogeneous objects. Graph model seems to be the most appropriate way to represent those interactions. However, the extracted graphs have in general a huge size which makes it difficult to analyze and visualize. An aggregation step is needed to have more understandable graphs in order to allow users discovering underlying information and hidden relationships between entities. In this work, we propose new measures to evaluate the quality of summaries based on an existing algorithm named k-SNAP that produces a summarized graph according to user-selected node attributes and relationships.

1 Introduction

Data manipulated in an enterprise context are structured data as well as unstructured content such as e-mails, documents, etc. Graphs are a natural way of representing and modeling such data in a unified manner (structured semi-structured and unstructured ones). The main advantage of such structure resides in its dynamic aspect and its capability to represent relations, even multiple ones, between objects. People need to visualize different types of interactions between heterogeneous objects (e.g. product and site, customers and products, people interaction like social networks, etc.). In order to analyze these interactions and facilitate their visualization, it is relevant to modulate such interaction by using a graph structure.

However, graphs extracted are often large, with thousands or even millions of nodes and edges. As a result, it is almost impossible to understand the information encoded in these graphs by mere visual inspection. In order, to facilitate the visualization and data interpretation, it seems interesting to perform an operation of summarization. The objective of graph summarization is to produce small and understandable summaries and can highlight communities in the network, which greatly facilitates the interpretation. Today, summarization has attracted a lot of interest in the database community therefore, it should use not only the relations between nodes but also the characteristics of each one, and very few algorithms are adapted to such complex graphs. In this context, we propose a general tool for graphs summarization which takes into account the heterogeneity of our data. This tool is based on an existing technique called k-SNAP (Tian et al. (2008)) that integrates an interactive querying scheme by allowing users to customize the summaries based on user-selected node attributes and relationships.

The remainder of this paper is organized as follows: After a brief review of the existing graph aggregation methods in section 2, we focus our critique on k-SNAP method to identify the missing components that limit the use of this technique in practice. Section 4 describes the global architecture of our summarization tool. The formalism of the new proposed measures that evaluate the quality of summaries is presented in Section 5. Experimental results are interpreted in Section 6, and Section 7 contains our concluding remarks.

2 Graph Aggregation Algorithms

When graphs of extracted social networks are large, effective graph aggregation and visualization methods are helpful for the user to understand the underlying information and structure. Graph Aggregation Algorithms produce small and understandable summaries and can highlight communities in the network, which greatly facilitates its interpretation.

The automatic detection of communities in a social network can provide this kind of graph aggregation. The community detection is a clustering task, where a *community* is a cluster of nodes in a graph (Girvan and Newman (2002), Newman and Girvan (2004)), such as the nodes of the cluster must be more connected with inside nodes, than with nodes outside of the cluster (see found Schaeffer (2007) and Santo (2010) for extended reviews).

The first class of clustering algorithms are the *partitional algorithms*, which try to find a partition of a set of data, with a given number of clusters, using jointly, most of the time, similarity or a dissimilarity **measures** and a quality criterion of the obtained partition. The most popular partitional algorithm (with several variants), the *k-means clustering* (MacQueen (1967)), tries to find a partition of the set of objects which minimizes the *sum-of-square* criterion which adds the dissimilarities from each object to the center of its own cluster. Several (di)similarity measures can be defined in the social network context, like those based on the *Jaccard index*, which measures similarity between the sets of *neighbors* of the two nodes, but other measures can be defined (Schaeffer (2007) and Santo (2010)).

Hierarchical clustering algorithms try to organize data into a hierarchical structure, and are divided into *agglomerative* and *divisive* algorithms, depending on whether the partition is coarsened, or refined, at each iteration. The basic idea beyond *agglomerative algorithms* is simple: at the starting point, the objects to cluster are their own classes, and then at each stage we merge the two most similar clusters. Of course a dissimilarity measure between two clusters is mandatory, and for a given dissimilarity measure d between objects, several cluster-dissimilarities exist. The result of the clustering process is a *dendrogram*, which can be cut to give one single partition. *Divisive clustering algorithms,* split the dataset iteratively or recursively into smaller and smaller clusters, with respect to a quality criterion. The most popular method for divisive hierarchical clustering of social networks uses the notion of edge betweenness (Freeman (1977)), because finding the connecting edges between communities is also finding these communities. The algorithm given in Girvan and Newman (2002) splits the network into clusters by removing, step by step, the edge with the higher betweenness value. The use of a stopping criterion which measures the improvement at each step should permit to stop when no improvement is gained with an iteration. In most cases the *modularity* defined by Newman (2004) is used. SuperGraph (Rodrigues et al. (2006)) employs hierarchical graph partitioning to visualize large graphs.

Specially designed for graphs, *spectral algorithms* (Von Luxburg (2006)) are based on the notion of connected components. These algorithms work with a Laplacian matrix based on the adjacency (or weight) matrix (Shi and Malik (2000), Ng et al. (2001)). If the graph of the social network contains k completely disjoints communities (*i.e.* without any link between them), called *connected components*, then the k *eigenvectors* which their eigenvalue are equal 0 are the indicator vectors of the k connected components. If the clusters of the social network do not contain "clean" connected components (*i.e.* if there are links between existing communities), then a simple clustering on the k *eigenvectors* associated to the k least eigenvalues, can retrieve the k communities.

Some other algorithms use statistical methods to study graph characteristics, such as degree distributions (Newman (2003)), hop-plots (Chakrabarti et al. (2007)) and clustering coefficients (Watts and Strogatz (1998)). The results are often useful but difficult to control and especially to exploit. Methods for mining frequent graph patterns (Yan and Han (2002)) are also used to understand the characteristics of large graphs. Washio and Motoda (Newman (2003)) provide an elegant review on this topic.

However, all the previous algorithms use only on links between nodes of the graph, and do not take into account the internal values contained in each node, while classical clustering algorithms applied on tables of values, work only on these values ignoring completely the possible links between individuals. An algorithm which can take into account both kinds of information would be very valuable. Designed for graphical graph aggregation the k-SNAP algorithm Tian et al. (2008), in its divisive version, begins with a grouping based on attributes of the nodes, and then tries to divide the existing groups according to their neighbors groups, to minimize a loss information measure and find the summary of size k with the best quality. We discuss k-SNAP algorithm in the next section.

3 Discussions

Most existing work such as algorithms already mentioned, are rather methods of structural partitioning that completely ignore the attributes associated with nodes and also the multi-relation aspect of social network, which makes interpretation very difficult. The summarization operation should use not only structural information related to graph organization but it should also take into consideration semantic information including among other nodes description in form of attributes (for instance, student node may be characterized by the attributes: sex, department, class . . .) and interaction types between them (different kind of relationship). These additional knowledge may guide the process of network aggregation and can also provide more relevant analysis according to different perspectives and point of view.

Although k-SNAP summarization method provides useful features that can help users to extract and to understand the underlying information encoded in large graphs, two key components are missing and this limits the practical application of this technique in many cases.

First, the k-SNAP approach only deals with categorical node attributes. But in the real world especially in a business context, many node attributes are numerical, such as the age of the employees or the number of e-mails exchanged between users in an enterprise network. Simply running the graph summarization method on the numerical attributes will result in summaries with large sizes (at least as large as the number of distinct numerical values). On the other hand, k-SNAP is not practical with the presence of a large number of attributes mainly with multiple modalities. In this case, it produces scattered and non informative summaries formed by a large number of groups with small size (at least as large as the cardinal of Cartesian product of all modalities).

The second missing component that affects the usability of k-SNAP, is the restriction of aggregation on homogeneous graphs where nodes should be characterized by the same description *i.e*, the graph is composed by a one type of object so, we dispose a single set of attributes defined for all nodes.

Nevertheless, new requirements related to the enterprise context appear ; actually, people need to analyze different types of interactions between heterogeneous objects to exploit them for commercial purposes: sending product recommendations to a targeted customer and guide preferences. Also for organizational purposes: learn about the roles of individuals (eg who hold the information, who is the responsible, who is the expert) and their interactions (eg, who works with whom). Thus, the graphs collected are no longer homogeneous but rather heterogeneous as they are extracted from relational databases Soussi et al. (2010); such graph contains several kinds of relations and objects. Each object owns a set of characteristics which can be different from object to another. In this work, we not limit ourselves to homogeneous graphs but we consider also other kind of graphs among other heterogeneous ones. Our aggregation process is established in two steps like k-SNAP: the first step is based only on attributes (*A-compatible grouping*) and the second step is based on relationships (*(A, R)-compatible grouping*).

4 Architecture overview

Figure 1 shows the global architecture of our graph aggregation tool along with its key components. This architecture highlights the multi strategies and scenarios aspects of this system according to the input graph. The objective is to summarize any type of graph; heterogeneous or homogeneous and even without knowledge. First, we start by importing a graph to aggregate. A pre-treatment is conducted to detect the nature of this graph. Once identified, the aggregation operation is launched. We emphasize the notion of interactivity by giving to the user the possibility to intervene by selecting a scenario of analysis according to her point of view.

Homogeneous Graph

This is the most common case, wherein the graph represents a social network formed by a single community of entities (e.g., student community, readers community . . .). These entities are characterized by the same set of attributes. However, in some cases, retrieved graphs may be without knowledge (nodes are not attributed) which makes k-SNAP operation as it is, powerless to analyze these kind of graphs. Therefore, a phase of clustering seems to be necessary to generate an initial partition. The obtained partition is considered similar to that one obtained

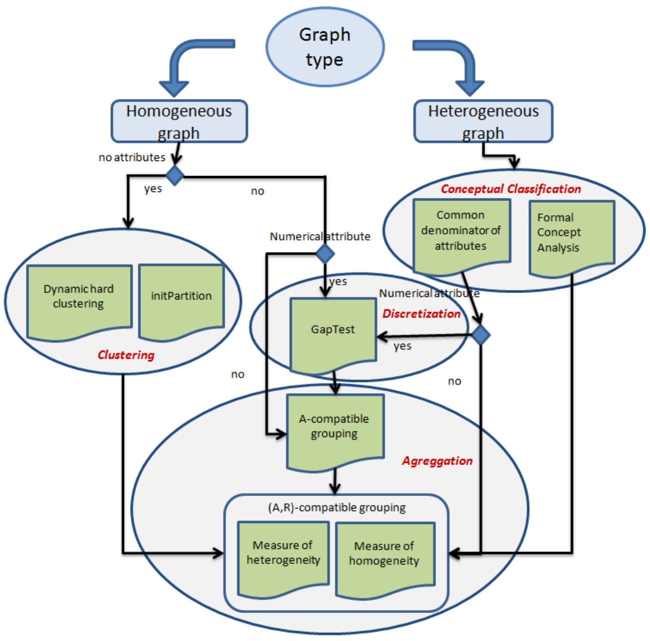


FIG. 1 – *Global architecture.*

from the *A-compatible grouping* step and it will be used later as an input for the aggregation step. In this first step, the user can choose one method of clustering among two: *dynamic hard clustering* (De Carvalho et al. (2012)) or *initPartition* (a priori partition). *Dynamic hard clustering* is a clustering algorithm appropriate to graphs issued from multiple relationships. It is able to partition nodes by taking into account their relational description given by multiple dissimilarity matrices. Concerning *initPartition*, it is also a clustering algorithm applied to classic graphs which nodes are not described by a set of attributes. Once the partition is generated, we execute the second step of the algorithm ensured by the function of *(A, R)-compatible grouping* in which user can also choose an evaluation measure among two. On the other side, if nodes are described by a set of attributes, two cases arise : (1) All attributes are categorical: this is the standard case of data sets, one joins the conventional operation of aggregation k-SNAP. (2) Presence of numerical attribute(s): we perform a discretization of numerical attributes in order to transform them into categorical attributes with few modalities. For this, we use the *Gap Test* algorithm.

Heterogeneous Graph

By Heterogeneous Graphs we mean, a kind of graph which is formed by different types of entities *i.e*, they do not represent a single community thus, they are not characterized by the same list of attributes. Typically, this type of graph describes the interactions between multiple corporate entities. A relational database can be a rich source for building such as graph (Soussi et al. (2011)). Knowing that the user chooses the set of attributes from a list, if this list is not generic especially since each attribute describes a subset of nodes and not all, the *A-compatible grouping* step may not be applicable. A trivial solution is to consider only the set of common attributes by searching the common denominator of all node types. Another possibility is to make conceptual classification using a formal concept analysis. Once the Galois lattice is built, the idea is to select a level wherein there is no redundancy between concepts in term of attributes. The graph partition will be made by putting the set of individuals related to each concept in a separate group.

Regardless of the type of the input graph, homogeneous, heterogeneous or even without knowledge, all these scenarios converge to (A, R)*-compatible grouping* which is the last step in the aggregation process. Given that (A, R)*-compatible grouping* function is based on relations between nodes like k-SNAP, the user can choose according to her point of view one evaluation measure among two as shown in Figure (Fig 1). The first measure is a measure of homogeneity, and the second is a measure of heterogeneity. Both are different from the measure proposed by k-SNAP method. In the next section, we formally describe the principle of these measures and present our algorithm.

5 Quality measures proposed

Prior to presenting our quality measures, let us look at how the evaluation measure k-SNAP is formulated. Recall that this measure denoted $\triangle$ is based on the notion of participation ratio of each pair of group (C_i, C_j) according to a relationship of type R_t (Tian et al. (2008)). The aggregation operation chooses at each step, the group that makes the most contribution to the $\triangle$ value with one of its neighbor groups according to the relationship R_t to be cutted into only two subgroups until the size of the grouping is equal to *k*. The evaluation measure of k-SNAP is defined as follows : but first, note that all calculations will be made from the incidence matrix $A_t = (\overset{t}{a}_{i,j})_{1<i,j<n}$ associated with the relation R_i. We define N_{R_t} the participation matrix of rank $|P|(|P|$ is the cardinal of the partition P) corresponding to the relation R_t by:

$$(\overset{t}{n}_{i,j})_{1<i,j<|P|} = \sum_{k=0}^{|C_i|}(1 - \prod_{l=0}^{|C_j|}(1 - a^t_{kl})) \tag{1}$$

Then, we define M the matrix of rank $|P|$ which contains the ratios of participation of different groups with respect to the relation R_t:

$$(\overset{t}{m}_{i,j})_{1<i,j<|P|} = \frac{n^t_{ij} + n^t_{ji}}{|C_i| + |C_j|} \tag{2}$$

For a given graph G, a set of attributes A and a set of relations R, the evaluation measure Δ of a the partition P is defined as follows:

$$\Delta(P) = \sum_{1\leq i,j\leq|P|} \sum_{R_t\in R} \delta^t_{ij} \tag{3}$$

with

$$\delta^t_{ij} = \begin{cases} n^t_{ij} & if \;\; m^t_{ij} \leq 0.5 \\ |C_i| - n^t_{ij} & otherwise \end{cases} \tag{4}$$

This measure is based on determining the difference in participation of each pair of groups with respect to the relationship R_t ,*i.e*, Δ-measure counts the minimum number of differences in participations of group relationships between the given *A-compatible grouping* and a hypothetical (A, R)- *compatible grouping* of the same size. According to equation 4 we have two possible cases:

- If this group, the relationship is weak $(m^t_{i,j} \leq 0.5)$, then it counts the participation differences between this weak relationship and a non-relationship $(m^t_{i,j} = 0)$.
- On the other hand, if the group relationship is strong $(m^t_{i,j} > 0.5)$, it counts the differences between this strong relationship and a 100% participation-ratio group relationship $(m^t_{i,j} = 1)$.

$W_{R_t} = (\delta^t_{ij})_{1 \leq i,j \leq |P|}$ from equation 4, that evaluates the part of the Δ value contributed by a group C_i with one of its neighbors C_j in a group relationship R_t.
Given k the desired number of groups, the k-SNAP operation produces an (A, R)*-compatible grouping* with the minimum Δ value, starting from a *A-compatible grouping* and Δ initialized to zero, the procedure is to look for each iteration the group to split. For this, we introduce an heuristic that chooses the group making the biggest contribution to Δ with one of its neighbor groups. More formally, for each group C_i, we denote $CT(C_i)$ as follows: $CT(C_i) = max\left\{\delta^t_{ij}\right\}$.
Then, at each iterative step, we always choose the group with the maximum CT value to split, based on whether nodes in this group C_i which have relationships with nodes in its neighbor group C_e, where: $C_e = \arg\max_{C_j}\{\delta^t_{ij}\}$ and then split it into two sub-groups according to the following strategy: one of these groups contains all nodes participating in the relationship with the group C_e and the other contains the rest, *i.e.* the nodes that have no relation with the group C_e.

In this section, we propose two new evaluation measures based on the principle of common neighbors. Two nodes are assigned to a cluster according to how they share neighbors. This makes sense when you consider social communities. People who share many friends create a community, and the more friends they have in common, the more intimate is the community. Thus, our objective is to establish an evaluation measure which is more refined comparing to that of k-SNAP. In other words, we no longer based on the notion of neighbors groups but rather on the notion of common neighbors. We also propose another mechanism for cutting based on the notion of the *Central Node* of a group. Similar to the k-SNAP algorithm, we also start from the *A-compatible grouping* based only on attributes, and then iteratively split existing groups until the number of groups reaches k. At each iterative step, we have to make the following decisions: (1) which group to split and (2) how to split it. The first measure is a measure of homogeneity (similarity) while the second is a measure of separability (distance).
First, we present some useful concepts and definitions for the formalization of these measures.

5.1 Concepts and definitions

In a graph, objects are represented by nodes, and relationships between objects are modeled as edges. In this paper, we support a general graph model, where objects (nodes) have associated attributes not necessarily identical and different types of relationships (edges).

Graph model
Formally, from a set of nodes V and a set of relationships types $R = \{R_1,\ R_2,\ ...,\ R_r\}$ defined

on V we denote a graph G as (V, E) where $E = \{E_1, E_2, ..., E_r\}$ is the set of edges such that $(u, v) \in E_t$, if, uR_tv. Each element v of V is characterized by a set of $|p|$ attributes, which is denoted as $\Lambda(v) = \{a_1, a_2, ..., a_p\}$. These attributes are used to describe the features of the objects that the nodes represent.

Partition Structure

$P = \{C_1, C_2, ..., C_k\}$ is a partition of V, into disjoint and exhaustive groups that satisfies the following conditions:

- NonEmpty: $\forall i \in [\![1, k]\!], C_i \neq \emptyset$
- Disjoint: $\forall (i, j) \in [\![1, k]\!]^2, i \neq j \Rightarrow C_i \cap C_j \neq \emptyset$
- Structure Covering: $\cup_{i=1}^{k} C_i = V$

In other words, in P, each node belongs to exactly one group, and some nodes may be alone in their groups.

Node structure

Let $v \in V$, $R_t \in R$, the structure of a node v is defined by its neighborhood, denoted by: $N_{R_t}(v) = \{w \in V \mid (v, w) \in E_t\} \cup \{v\}$.

At each iteration, the algorithm consists in choosing the group that minimizes (respectively maximizes) the evaluation criterion to be divided into two groups. The mechanism of division is based on the notion of the *Central Node* of a group. Before characterizing the structure of a *Central Node*, we introduce a function denoted Φ_p to control the index assignment of a member to a group. This assignment function is defined as follow:

Definition : Assignment function

Let $v \in V$, the assignment function Φ_p that assigns a node v to a group C_i of a partition P is a function that associates an index $i = 1, ..., |P|$ for each node indicating the group to which it belongs.

We also introduce other definitions.

Definition : Local degree of a node

Let $v \in V$, $R_t \in R$, the local degree of a node v associated with the relationship R_t and the partition P is defined by :

$$Deg_{R_t,P}(v) = |N_{R_t}(v) \cap C_{\Phi_p(v)}|.$$

In reality, it is the cardinal of links issued from v but only within the class $C_{\Phi_p(v)}$ for the relationship R_t. From this definition, another one may be inferred, the complementary local degree which includes the rest of the links issued from v. It will be noted as follows :

$$\overline{Deg}_{R_t,P}(v) = |N_{R_t}(v) \cap \overline{C}_{\Phi_p(v)}|.$$

Definition : Characterization of a Central Node *of a group*

The *Central Node*, noted v_d, of a group C_i belonging to the partition P and associated with a relationship R_t is given by :

$$d = argmax_{v \in C_i} Deg_{R_t,P}(v).$$

The advantage of introducing the concept of *Central Node*, is to be able to summarize a group by a single representing node (prototype), which greatly facilitates thereafter the visualization.

It should be noted that during the execution of the algorithm [see Algo 1], some situations require the introduction of another concept for the centrality to avoid redundancy, named *"the centrality of the second order"*.
Definition : Centrality of second order
A *Central Node* of second order, noted v_s, of a group C_i that belongs to the partition P and associated with a relationship R_t is defined by :

$$s = argmax_{v \in C_i \setminus \{v_d\}} Deg_{R_t,P}(v).$$

Definition : Betweenness centrality
The betweenness centrality can be defined as the ability of an actor to pose as intermediary in the relations between other members. This actor is quite similar to the definition of a hub in a network or organization. Now, we can describe the formalism of our two evaluation measures proposed. First, we introduce a measure of homogeneity that evaluates locally every group and tends to find the least homogeneous to be divided. Then, we present the second measure which is a measure of heterogeneity. At each iteration, to decide which group to split, these two measures assess the groups based on the notion of common neighbors. On how to divide this group, both use the concept of *Central Node*.

5.2 The measures of homogeneity of the partition P

5.2.1 Measure of homogeneity using the local degree

This measure is a combination of two evaluation criteria. The first assesses the homogeneity of the group; it is a intra-group criterion while the second criterion, evaluates the interconnectivity of a group with the other groups; it is a inter-group criterion. The overall measure is the ratio of these two criteria associated with a relationship R_t. We define the intra-group criterion of C_i as follows :

$$IA^t(C_i) = \frac{1}{|C_i|} \sum_{v \in C_i} Deg_{R_t,P}(v)$$

Practically, it is the sum of local degrees of all nodes belonging to the group C_i according to the relationship R_t and divided by the cardinal of this group. Then we define the inter-group criterion of the group C_i and the relationship R_t as :

$$IE^t(C_i) = \frac{1}{|E \cap C_i|} \sum_{v \in E \cap C_i} \overline{Deg}_{R_t,P}(v)$$

This criterion examines each group C_i and sum the external links of R_t in the partition P and divided by the cardinal of all edges associated to this relationship R_t. Finally, our evaluation measure is defined as the ratio of these two criteria. More formally, we define $\triangle$ as follows :

$$\triangle = \sum_{i=1}^{|P|} \sum_{R_t \in R} \delta_i^t = \sum_{i=1}^{|P|} \sum_{R_t \in R} \frac{IA^t(C_i)}{IE^t(C_i)}$$

The aim of this measure is to look at each iteration the group that is not only little connected but also containing a large number of nodes interacting with outsides to be cut. In other words,

this measure privileges groups that are dense and have few links with the outside for a given relationship. Unlike the k-SNAP evaluation, we do not take into account the locality of the connected node. The essential is that this connexion is an external relationship, regardless of the group membership.
Formally, we choose the group and the relationship that make the least local contribution to the value of $\triangle$:

$$(i^*, t^*) = argmin_{1 \leq i \leq |P|, 1 \leq t \leq |R|} \delta_i^t$$

Once the group C_{i^*} is selected to be cutted , the mechanism of division consists in determining of a node called *Central Node* noted $v_{C_{i^*}}$. In fact, we have two strategies to determinate this node, the first one is to look for the node which has the highest value of centrality degree and the second one, is to search the node with the higher value of betweenness. Then, we cut this group into two subgroups according to the following strategy : one contains the *Central Node* with its neighbors, the other the rest of the group.

5.2.2 Measure of homogeneity using a distance

This measure evaluates locally by a distance the density of each group according to the relationship and tends to split the group with the minimum density. We use the contingency table (or association table) defined for each pair (m,n) of nodes for a given relationship R_t as follows (Tab 1) :

	object m	
object n	a	b
	c	d

TAB. 1 – *Contingency Table.*

where : $a = |N_{R_t}(m) \cap N_{R_t}(n)|$, $b = |N_{R_t}(m)| - a$, $c = |N_{R_t}(n)| - a$.
We propose to use the Jaccard distance to calculate the measure of evaluation. This measure is defined as:

$$\triangle(P) = \sum_{R_t \in R} \triangle_t(P) = \sum_{R_t \in R} \sum_{1 \leq i \leq |P|} \delta_i^t$$

with

$$\delta_i^t = \sum_{m \in C_i} \sum_{n \in C_i} d^t(m, n)$$

where $d^t(m, n) = (b + c)/(a + b + c)$ is the Jaccard distance according to the relationship R_t. To better understand the principle of this distance based on the notion of common neighbors, we presented an example illustrated through the following figure (Fig 2). For the pair (A_1, A_2), the value of the parameter *a* which represents the number of their common neighbors is equal to 1. Concerning the parameter *b* which is the number of neighbors of the node A_1 deprived of the common neighbors with the node A_2, its value is equal to 3. Symmetrically, the parameter *c* is equal to 1.Thus, the Jaccard distance between these two nodes is equal to $4/5$. Similarly for the pair (A_1, A_2), the parameters a and b for the pair (A_1, A_3) are respectively equal to the values 1 and 3. However, c is equal to 0 as the node A_1 has one neighbor which is a common neighbor with A_3. Finally, for the last pair (A_2, A_3), the value of the parameter *a* is zero since

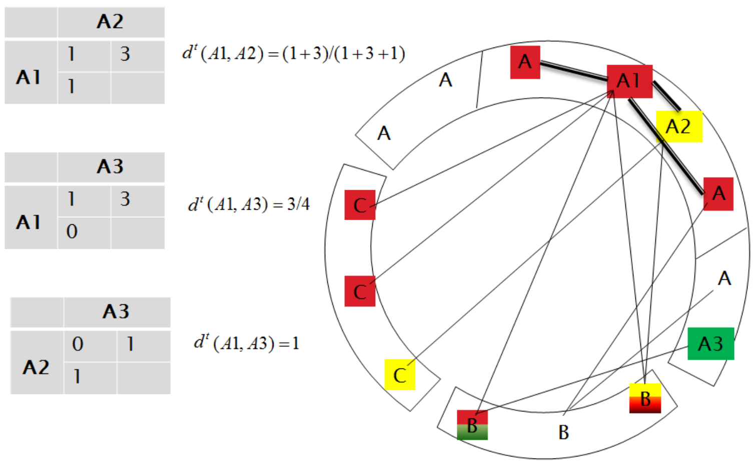


FIG. 2 – *Example of aggregation using our algorithm.*

they have no common neighbors. In this case, and regardless of the values of other parameters (b and c both equal to 1) the distance between the two nodes equals 1 thus, the distance is maximum. Using the same procedure, we calculate the remaining distances of the other pairs of nodes, we then evaluate the current partition in order to determine the group to split. In our case, the group A is the most heterogeneous because it maximizes the evaluation measure. So given that the node A_1 is the central node of the group, the division is made as shown in (Fig 2), the first subgroup contains the *Central Node* with its neighbors, the other the rest of the basic group.
For the same data set, we apply the k-SNAP operation to aggregate the corresponding graph and also to visually demonstrate the difference between the two techniques. As the evaluation measure of k-SNAP is based on the notion of neighbor groups, after two iterations we obtain a graph formed by 5 groups as shown in the following figure (Fig 3).
Indeed, only the group A underwent two successive subdivisions. All nodes belonging to A_1 interact only with the group B *i.e*, each node of the group A_1 has a relationship with at least one node in the group B. About the group A_2, these nodes have the same list of neighboring groups which are B and C. Finally the nodes belonging to the group A_3 are isolated nodes, they have no connections with the outside.
To summarize, starting from a *A-compatible grouping*, our procedure is to look at each iteration, the relationship and the group to divide that maximizes the evaluation measure : this is the step of selection until the cardinal of the partition is equal to k. Like the other measure of homogeneity, the step of division consists in determining of the *Central Node* of the group to divide, and splitting this group into two subgroups according to the same strategy. Comparing to k-SNAP method, this measure assesses the dissimilarity between two nodes in a more fine way because it is based on the concept of common neighbors regardless of group membership. The algorithm is summarized below.

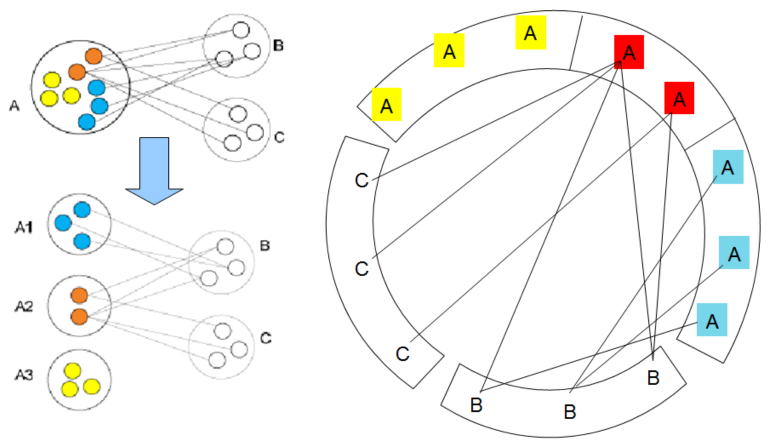


FIG. 3 – *Example of aggregation using k-SNAP.*

5.3 The algorithm

In this subsection, we describe our algorithm which produces a summary graph through an operation of aggregation of the input graph's nodes based on user-selected node attributes and relationships. The pseudo code of the algorithm is presented in below (Algo 1).

This operation of aggregation tries to find the best grouping for a graph, a set of nodes attributes, and the specified relationship type. Similar to the k-SNAP algorithm, the evaluation algorithm starts from the maximum *A-compatible grouping* (STEP 1 in Algo 1) based only on attributes, and iteratively splits groups in the current grouping, until the grouping is also compatible with the relationships and its cardinal is equal to k . In the following steps, we develop our approach for deciding which group to split and how to split it at each iterative step. As defined in Section 4, we specify that our aggregation tool is a tool of multiple strategies and scenarios; allowing the user to have multiple points of views of the solution. According to the chosen evaluation measure (separability or homogeneity), our objective is to minimize (respectively maximize) the Δ measure. Therefore, the principle is to choose the group C_{i^*} and the relationship R_{j^*} that make the most (respectively less) contribution to its value (STEP 9 in Algo 1) this is the *selection step*. Once the class to be cutted is selected, the *division step* of the class C_{i^*} (STEP 10 in Algo 1) consists in determining of a node called *Central Node* noted $v_{C_{i^*}}$. In this step, the user can choose also one among two modes of centrality, the first mode is to determine the node that maximizes the local degree and the second is to detect the node with the higher value of betweenness. However, if this node has been selected in a previous iteration, naturally, it is connected to all other nodes belonging to the group to divide. In this case and to ensure the division of this group, we proceed to find the *Central Node of the second order*. It's the node that has the second highest value of local centrality. Whatever the method of centrality chosen, the division is performed as follows : the group C_{i^*} keep all nodes $v \in N_{R_{j^*}}(v) \cup \{v_{C_{i^*}}\}$ (STEP 11 in Algo 1) and a new group is generated in STEP 11 which will contains the rest of nodes. In STEP 13 and 14, we update the current partition P and the value of $\triangle$ and we check if the number of groups is already equal to k.

Algorithm 1 Search for groups based on the notion of common neighbors

Input : G a graph; k the number of groups; $A \subseteq \Lambda(G)$ a set of attributes; $R = \{R_1, R_2, ..., R_r\}$ a set of relationship.
Output : aggregated graph with K groups.

```
1: $P = \{C_1, C_2, ..., C_k\}$ is the $A - compatible$ partition based on attribute values of $A$.
2: $\Delta = 0$.
3: WHILE $|P| < K$
4:   FOR $t = 0$ to $r$
5:     FOR $i = 0$ to $|P|$
6:       Compute $\delta_i^t$ the evaluation measure of the class $C_i$ for the relationship $R_t$.
7:     ENDFOR
8:   ENDFOR
9:   Look for $(i^*, t^*) = argmax(min)_{1 \leq i \leq |P|, 1 \leq j \leq |R|} \delta_i^j$ and select the group $C_{i^*}$ to be divided according to the relationship $R_{t^*}$.
10:  Look for the central node $v_{C_{i^*}}$ such that $Deg_{R_{t^*},P}(v_{C_{i^*}}) = max_{v \in C_{i^*}} Deg_{R_{t^*},P}(v)(max_{v \in C_{i^*}} BetweennesCentrality)$ or eventually the central node of the second order.
11:  Keep all nodes $v \in N_{R_{t^*}}(v) \cup \{v_{C_{i^*}}\}$ within the group $C_{i^*}$.
12:  Put the rest into a new group $C_{|P|+1}$.
13:  update $P$.
14:  update $\Delta$.
15: ENDWHILE.
```

6 Experimental results

In this section, we interpret the experimental results and we evaluate the effectiveness and efficiency of our method on two real datasets, namely the Books about US politics and the political blogs datasets. All algorithms are implemented in Java and the graph data is stored in a GraphML format (See GML (2011)). First of all, we describe the datasets used in our empirical evaluation. Then, we try to interpret the results obtained through the k-SNAP and our algorithms and make a comparison. Finally, to evaluate the efficiency and the effectiveness of our graph summarization methods, we compare in first the execution times for the two approaches on the political blogs network (1490 nodes and 19090 edges) and we evaluate the quality of the results of our tool in the case of an unsupervised classification.

Books about US politics
This network, compiled by V. Krebs (unpublished), represents a recent study about books on US politics. Links connecting pairs of books that are often purchased by the same customers on the on-line bookseller Amazon.com. The nodes are described by the values "liberal", "neutral" or "conservative" related on a single attribute that reflects the political ideology, it indicates if the books are, respectively, liberal, neutral or conservative. These annotations were assigned separately by Mark Newman based on reading reviews and descriptions of books published on Amazon. This network is illustrated by the figure (Fig 4), and the corresponding graph contains 105 nodes and 441 edges.

Political Blogs Network
To obtain a representative view of the community of liberal and conservative blogs during the

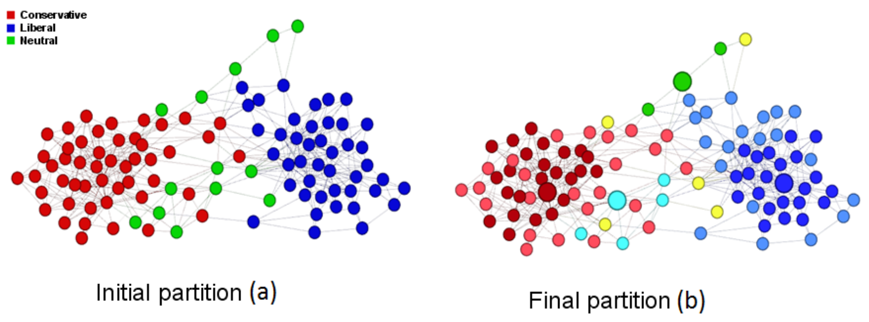


FIG. 4 – *Aggregation using our algorithm.*

election period, Lada Adamic and Natalie Glance studied the binding modes and discussion topics of political bloggers (Adamic and Glance. (2005)). Their objective is to measure the degree of interaction between liberal and conservative blogs, and to discover all the differences in the structure of both communities. Specifically, they analyze the positions of 40 "A-list" blogs over the period of two months preceding the presidential election. In fact, they have managed to gather a broad range of political blog URL by uploading lists of several blog directories on-line, including eTalkingHead, BlogCatalog, CampaignLine and Blogarama. There are 1494 blogs in total described by one attribute, with 759 liberal blogs and 735 conservative blogs and 19090 edges.

Selected scenario

We first recall the principle of our algorithm ; it consists in selecting the group to be cutted that minimizes (or maximizes) the criterion according to the measure chosen, then look for the *Central Node* and divide the group into two subgroups according to the following strategy: one contains the neighborhood of *Central Node*and the other the rest of the original group. For this dataset, we have chosen as a selection criterion: the degree of homogeneity, and as a criterion of division: the degree centrality.

6.1 Interpretation of results

In practice and for ease of visualization, users are more likely to choose small k values to generate summaries to make a meaningful interpretation. We choose to fix the size of summary graph to 7.

Our algorithm

Applying our method on the network of books about US politics, the function *A-compatible grouping* generates first a summary formed by three groups in accordance with the modalities liberal, neutral and conservative of the political tendency attribute. The network is divided as shown in the figure (Fig 4.a), with the colors of the nodes representing the values of the attribute. The two groups of books according to left-wing and right-wing points of view are respectively colored blue and red. However, the green group is formed by books neutral. After four iterations (Fig 4.b), the graph is formed by seven groups. The group color azure is formed by books that are basically neutral but has a conservative aspect. Indeed each of these books have at least one connection with a book belonging to the red group. Similarly, the green group is formed by books that are neutral but tinged with a liberal character because each of these books has at least one connection with a book belonging to the blue group. Finally, the yellow group consists of the remaining neutral books that belong to initial partition. The dark blue and dark red nodes are those that, by our calculation, belong most strongly to the two groups and are thus, perhaps, the "most left-wing" and "most right-wing" of the books under consideration. Those familiar with current US politics will be unsurprised to learn that the most left-wing book in this sense was the polemical Bushwacked by Molly Ivins and Lou Dubose. Perhaps more surprising is the most right-wing book: A National Party No More by Zell Miller. These books are in really the *Central Nodes* of groups and are represented in the figure by a node of larger size. The interest of introducing the concept of the *Central Node* in the process of aggregation is to summarize each class by a single representing node called also prototype, which greatly facilitates the visualization. Thanks to this new representation established, using the

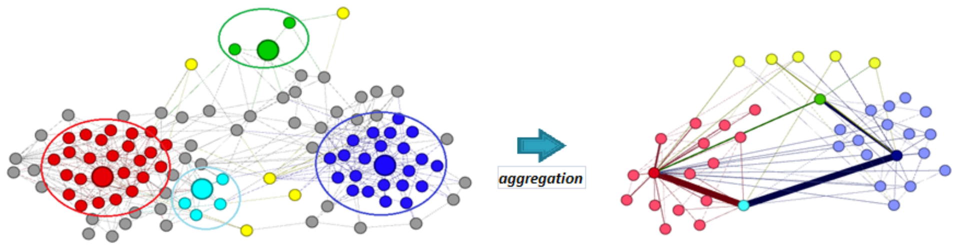


FIG. 5 – *Graph Summarization using central nodes.*

Central Node (Fig 5), other interpretations can be deduced: we can identify clearly through the figure that there are two crossing point or bridges that connect liberal books with conservative ones. On the other hand, all the books belonging to the yellow group are considered outliers. Indeed, this group will remain static and will no longer be splitted throughout the aggregation process since the intra-group density is zero and all nodes are seen as central in this case. We can say that these books have little or no influence, and may be isolated as noise in the data. The identification and isolation of bridges and outliers is essential for many applications. As an example, the identification of hubs in the WWW improves the search for relevant authoritative web pages. Furthermore, hubs are believed to play a crucial role in viral marketing and epidemiology.

k-SNAP algorithm

Now we make use of the k-SNAP operation to produce summaries from the same dataset. We fix also the value of the size of summary graph to 7. After four iterations, the graph is formed by seven groups. A visual interpretation of the figure (Fig 6.a) shows the existence of two particular groups, these are the groups C_1 and C_2. By the k-SNAP evaluation based on the notion of neighboring groups, the books belong to these groups are considered as the "most right-wing" and "most left-wing" of the books being studied. Indeed, each book belonging to C_1 (respectively C_2) connect only to books in its own community (conservative or liberal). Overall, conservative books show a slightly higher tendency to link with each other than liberal books, which is consistent with the conclusion from the analysis in Adamic and Glance. (2005). Comparing the summaries for the two communities for k = 7, we can see two major differences across the two communities : the first difference lies in the group of books neutrals (green color). In effect, this group remains intact throughout the process of aggregation. We can say that this group is the more homogeneous one with respect to the evaluation criterion of k-SNAP, which is based on the list of neighboring groups because the majority of books show a higher tendency to interact with other books both conservative and liberal thus, the summary graph has only one crossing point that connect liberal books with conservative ones. The second difference is that the liberal group underwent three successive divisions : they have very weak connections to other groups but are strongly connected among themselves. Thus, in term of k-SNAP's evaluation, this group maximizes the Δ-measure because the nodes are interacting with several groups and have not in majority the same list of neighboring groups. In some way , this summary does not provide a lot of information about customer's behavior. The result of k-SNAP algorithm is shown on the graph in Figure (Fig 6).

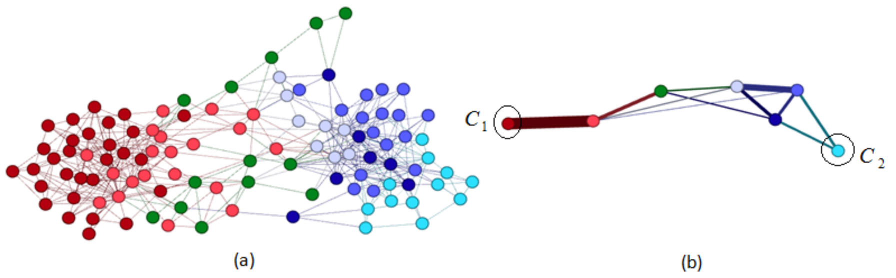


FIG. 6 – *Aggregation using k-SNAP algorithm.*

The criterion of k-SNAP is rather a separability criterion, in fact it separates the nodes according to the list of neighboring groups. In other words, every node in one group relates to some node(s) in the neighbor group thus, this criterion neglects the density of the group (intra-group) and favors the external relations (inter-group). However, our criterion evaluates locally the homogeneity of each group and tends to divide the least homogeneous according to the principle of the *Central Node*, the summary graph contains useful information about the network, indicating, as discussed above, the "strength" with which books belong to the communities in which they are placed.
Moreover, by means of effectiveness and efficiency, we present experiments comparing the performance of our algorithm with existing algorithm Tian et al. (2008) and evaluate the quality of the obtained results when applying a clustering method. Effectiveness and efficiency of the graph aggregation algorithm are the key performance indicators. The effectiveness of the algorithm measures the quality of the obtained results in the context of a clustering to discover the underlying structure of the network and compare the generated partition to that given by

the expert (*a priori* partition). The efficiency of the algorithm measures its computation cost by varying the number of groups (k) of the generated partition.

6.2 Efficiency Evaluation

In this subsection, we compare our algorithm and the k-SNAP algorithm in term of efficiency. To realize this evaluation, we use the political blogs network (1490 nodes and 19090 edges) and we apply both approaches by varying the size of the generated partition. For each simulation we choose a different value for k and we compute the execution times. The execution times (sec) for the two approaches are plotted in Figure (Fig 7). Our algorithm outperform the k-SNAP approach, except when k has small values exactly between 4 and 32. Concerning

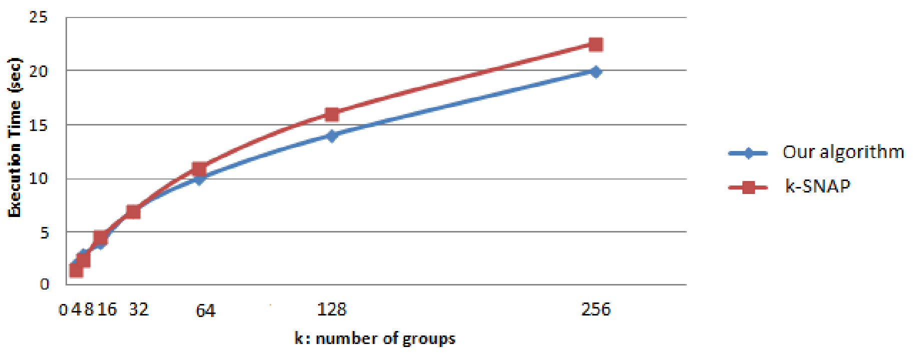
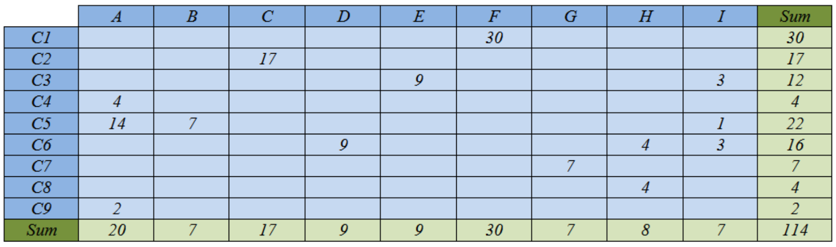


FIG. 7 – *Efficiency : k-SNAP vs. our algorithm.*

k-SNAP approach, the aggregation process does not take much time for small values of k. This situation becomes increasingly bad when the size of the partition rises. This situation is explained seeing that the principle of the algorithm is based on the comparison of each pair of groups. Therefore, the larger the partition size is, the more complex the computational cost is. However, our algorithm seems to be better for high values of k. This is quite logical as our quality measure evaluates locally each group. When the size of the partition becomes small, the evaluation takes less time and the computation cost is better. For this dataset, the performance behavior of both is close to logarithmic allure.

6.3 Effectiveness Evaluation

In this experiment, we evaluate the effectiveness of our graph aggregation method on a real dataset(114 nodes and 731 edges, Scapin et al. (2011)). The expert partition of this network is known and consists of 9 groups. Unlike k-SNAP method which is based mainly on the attributes contained in the nodes, our tool can handle also graphs without *a priori* knowledge by carrying out a clustering method (see Section 4). The challenge is to discover the underlying structure of this network and compare the obtained results to the partition given by the expert. The experimental part consists of the realization of an unsupervised classification of all documents into 9 groups and make a correspondence between the two partition by means of contingency table. This real dataset includes all types of document that a person can possess (finance, contact, calendar, health,...) which are represented by nodes. Links are connecting pairs of documents that are often classified by the same users in the same box. A link will be created if at least 30 users (threshold value) classify both documents in the same box. Note that the experience was applied to 43 subjects. We adopt the *Contingency Table* (Fig 8) as a means of representation in order to detect the difference between the resulting partition and the expert partition.



	A	B	C	D	E	F	G	H	I	Sum
C1						30				30
C2			17							17
C3					9				3	12
C4	4									4
C5	14	7							1	22
C6				9				4	3	16
C7							7			7
C8								4		4
C9	2									2
Sum	20	7	17	9	9	30	7	8	7	114

FIG. 8 – *Effectiveness : Contingency Table.*

By analyzing this table, we note that in general, the partition obtained using our algorithm confirms that given by the expert with the exception of two classes "H" and "I". Indeed, the documents belonging to these classes are a bit scattered over several classes. We can say that the definition made by the expert is ambiguous so that most subjects could not properly classified these documents. The interest of this study is to reconsider the classification made by the expert by proposing some recommendations that may be useful for a new re-definition.

7 Conclusion

This paper has introduced a general tool for graphs aggregation which takes into consideration the aspect of heterogeneity of these graphs. Our tool allow users to freely choose according to the kind of graph, a scenario of aggregation and produce summaries based on the selected features. Furthermore, like the k-SNAP algorithm, our method allow users to select node attributes and relationships that are of interest and to fix *a priori* the size of the graph. We have formally described our measures of evaluation based on the notions of common neighbors and *Central Node* and presented the algorithm. Through experiments on a real datasets, we interpreted the results obtained by applying our algorithm and show that this interpretation is more significant than that the one obtained from k-SNAP. Our experiments also demonstrate the effectiveness and efficiency of our method. As part of future work, we plan to change the step of settings based on user selection; the goal is not to impose from the beginning a list of attributes, but to dynamically choose the most effective attributes to split the set of nodes.

References

Adamic, L. A. and N. Glance. (2005). The political blogosphere and the 2004 us election: Divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, Japan. Workshop on the Webloging Ecosystem: ACM.

Chakrabarti, D., C. Faloutsos, and Y. Zhan (2007). Visualization of large networks with min-cut plots, a-plots and r-mat. *Int. J. Hum.-Comput. Stud. 65*(5), 434–445.

De Carvalho, F. A. T., Y. Lechevallier, and F. M. De Melo (2012). Partitioning hard clustering algorithms based on multiple dissimilarity matrices. *Pattern Recognition 45*(1), 447–464.

Freeman, L. C. (1977). A set of measures of centrality based upon betweenness. *Sociometry 40*(1), 35–41.

Girvan, M. and M. E. J. Newman (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the USA*, 8271–8276.

GML (2011). *http://graphml.graphdrawing.org/.*

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Symp. Math. Stat. Probab., Univ. Calif. 1965/66.*

Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM REVIEW*, 167–256.

Newman, M. E. J. (2004). Detecting community structure in networks. *The European Physical Journal B 38*, 321–330.

Newman, M. E. J. and M. Girvan (2004). Finding and evaluating community structure in networks. *Phys. Rev. E. 70.*

Ng, A. Y., M. I. Jordan, and Y. Weiss (2001). On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14*, Vancouver, Canada, pp. 849–856. NIPS 2001: MIT Press.

Rodrigues, J., F. Jose, A. J. M. Traina, C. Faloutsos, and T. J. Caetano (2006). Supergraph visualization. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, Washington, DC, USA, pp. 227–234. IEEE Computer Society.

Santo, F. (2010). Community detection in graphs. *Physics Reports 486*, 75–174.

Scapin, D. L., P. Marie-Dessoude, M. Winckler, and C. Detraux (2011). Personal information systems: User views and information categorization. In *Proc. Centric2011 - 4th. International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services*, Barcelona, Spain.

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, 27–64.

Shi, J. and J. Malik (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*(8), 888–905.

Soussi, R., M.-A. Aufaure, and H. Baazaoui (2010). Towards social network extraction using a graph database. *Advances in Databases, First International Conference on 0*, 28–34.

Soussi, R., E. Cuvelier, M.-A. Aufaure, A. Louati, and Y. Lechevallier (2011). Db2sna: an all-in-one tool for extraction and aggregation of underlying social networks from relational databases. In T. Ozyer (Ed.), *Social Network Analysis and Mining*, pp. 2–25. Springer

LNSN.

Tian, Y., R. A. Hankins, and J. M. Patel (2008). Efficient aggregation for graph summarization. In J. T.-L. Wang (Ed.), *SIGMOD Conference*, pp. 567–580. ACM.

Von Luxburg, U. (2006). A tutorial on spectral clustering. *Technical Report, Max Planck Institute for Biological Cybernetics 17*(4), 395–416.

Watts, D. J. and S. H. Strogatz (1998). Collective dynamics of 'small-world' networks. *Nature 393*(6684), 440–442.

Yan, X. and J. Han (2002). gspan: Graph-based substructure pattern mining. In *Proc. 2nd IEEE Int. Conf. on Data Mining ICDM'02*, pp. 721–724. IEEE Press.