Community Detection in Social Networks

Doxakis Chovardas, Dimitrios Vasdekis, Theodoros Liapikos, Konstantinos Serderidis

***Abstract* — Revealing the community structure, a task of great importance in understanding the properties of a network, is a problem in network analysis affecting various fields in science and economy. There is a vast amount of literature about this topic approaching the problem from various aspects. For the sake of the second phase of this project, four papers out of the eight that were presented during the first phase were chosen. The algorithms proposed in these four papers were implemented by our team, applying improvements and modifications where necessary. After the implementation of the algorithms, their performance regarding the task of community detection were evaluated by applying them on various well-known datasets. Finally, comparisons between the results achieved by the algorithms were made, allowing us to draw useful conclusions and insight.**

I. INTRODUCTION

In this second phase of the project, we decided that it would be interesting to apply some of the novel approaches presented in the papers reviewed in the first phase, on well-known datasets and in doing so, achieving two goals. Firstly, to verify the results described by the authors in the corresponding papers and secondly, to draw useful conclusions by comparing the performance of the algorithms, with each other.

During our research in relevant literature during the first project phase we stumbled upon the fact that as regards novel algorithms addressing the task of community detection there was little or no work in comparing the various approaches on common datasets, stating the strengths and weaknesses of each algorithm, in order for the researchers to decide whether an algorithm is appropriate to address a particular problem in practice. The problem stated above can be justified by the fact that the algorithms chosen are relatively new and there is insufficient experimental data.

Our contribution with this implementation is a useful report for researchers in the future, highlighting through experimental processes the strengths and weaknesses of five novel community detection algorithms, with little or no literature reviewing their performance so far. Also, this report serves as a means to compare these algorithms with each other in order for the reader to decide which of them is appropriate, if any, to address their task at hand.

In the process of addressing the tasks described above, we first went through a thorough analysis of the papers presented in the first phase of the project. Due to time constraints it was necessary to make a selection of the algorithms that were to be implemented and tested. We concluded that in order to be able to be consistent in our experimental process we should choose five out of the eight algorithms corresponding to different approaches of four different papers to tackle the problem at hand . Each of us would undertake the task of implementing one algorithm and after that, comparisons and experiments would be made by binding our implementations together and applying them on the same datasets. The algorithms chosen based on their novelty, their estimated potential for future usage and the degree of challenge in implementing them.

During the implementation of the algorithms, various difficulties that could not be predicted, raised and were overcome. In some aspects of the algorithms presentation in their corresponding papers, serious omissions like lack of handling for outlier nodes or ambiguities in results, were identified and thus, improvements and modifications were made. Finally, we were able to complete the task at its full extent and draw useful insights for each algorithm based on our results.

In the following sections of this report, our approach, methodology, difficulties and results are described in detail. In section II, we present the related work briefly, since it was analyzed in full extent in the first phase of this project. Next, in section III our methodology is stated in detail, followed by section IV in which we present our results and findings. In section V, we state the final conclusions drawn along with future work opportunities. Finally, in section VI, we quote a link for a GitHub repository were the implementation in python for all the algorithms presented in this report can be found.

II. RELATED WORK

Based on the literature review, it was decided to perform a comparative analysis of selected algorithms for community detection on well-known datasets for social networks. The main focus of the work was to identify the performance of different algorithms, adopting various approaches on a common pool of datasets.More specifically, the following algorithms out of the eight proposed in the literature review were selected:

1. ASOCCA, a Community Detection Algorithm Based on Local Similarity of Clustering Coefficient in Social Networks [1]
2. SCAN, a Structural Clustering Algorithm for networks [2]

3. CC-GA (Clustering Coefficient Genetic Algorithm), a clustering coefficient based genetic algorithm for detecting communities in social networks [3]
4. AMKM/WAMKM, (Weighted) Adjacent Matrix for K-means Clustering [4]: an improved version of the classic K-means clustering in two variations unweighted/weighted: AMKM/WAMKM

A brief summary of the selected algorithms follows:

II.1 ASOCCA

In the paper describing ASOCCA [1] (Adjacent Node Similarity Optimization Combination Connectivity Algorithm) the authors base their work on well known metrics and measures used in social network analysis. A key part for ASOCCA implementation is the usage of clustering coefficient of adjacent nodes. The most important metric though, in ASOCCA's implementation is the similarity between adjacent nodes, this is the factor that is used to decide which nodes belong to the same communities. The authors of the paper experiment with various similarity indices like Salton index, Jaccard index, Sørensen index and Resource Allocation (RA) index. The best results occurred when a new definition of similarity, introduced in the paper, was used. This newly defined similarity index decides the degree of similarity between two nodes based on the sum of the local clustering coefficients of their common neighbors. Thus, the accumulated value of similarity of two nodes is determined by two aspects: the number of common neighbors and the clustering coefficient of each common neighbor.

As for the evaluation of ASOCCA's performance, the modularity measure was used. The final partition of a graph given as input to the algorithm, is decided through an iterative process, which as a final result returns the partition with the highest modularity value. It is worth to be noticed that the ground truth partition (when there is one) is not always the one with the highest modularity and that is something that the authors also comment in their work. Six well known datasets were used in the process of evaluating the algorithm's performance. Three of them with ground truth communities (karate, dolphins and pol books) and three with no ground truth (facebook, internet and amazon).

The structure of the paper, using examples, charts and matrixes made the implementation and reproduction of the results stated in the paper relatively easy. Despite that, the pseudocode was not very specific and some serious ambiguities in handling certain issues (like isolated nodes) were found during the process of implementing the algorithm. Thus, improvements and modifications of the original work described in the paper were necessary. The process of implementing the algorithm, the improvements and the final outcome are described in detail in section III.

II.2 SCAN

SCAN (Structural Clustering Algorithm for Networks) is a method for finding communities in networks. According to the authors, the algorithm can detect clusters, bridges and outliers, based on the network's structure and the connectivity between nodes. Also, it is considered to be a fast clustering method with time complexity O(m) in a network with n nodes and m edges. This happens because the algorithm visits the nodes of a network only once.

The algorithm's main concept is that densely connected nodes should be placed in the same cluster. The density of the network is measured by a metric called structural similarity. The structural similarity for a pair of neighboring nodes represents the number of shared neighbors of these nodes. Based on structural similarity, two neighboring nodes can be strongly or weakly connected. The number of strong connections that a node has with its neighbors determine if the node will be considered a core node or not. Core nodes and their strongly connected neighbors form a cluster. If a core node is strongly connected with another core node, they form one cluster along with their strong connected neighbors. Additionally, SCAN can identify nodes that are not members of any cluster, because they are not strongly connected with their neighbors. These nodes are classified as bridges or outliers based on the number of clusters that are weakly connected with. Bridges and outliers give additional information about the network, that can be used in many fields like viral marketing and epidemiology.

The authors tested and evaluated the algorithm with 4 datasets with known communities. The evaluation score was adjusted rand index [5] and the results were compared with the results of another clustering algorithm, FastModularity [6].

The success of this method is easily understood by the fact that a considerable number of variations have been proposed in order to surpass its limitations and improve its performance. One of these variations is the DSCAN algorithm [7], which was presented in the phase I of this project and it is a distributed implementation of SCAN.

II.3 CC-GA

Genetic Algorithms (GA) is a class of algorithms that is based on the principles of biological evolution and find great application in analyzing and solving real-world search or optimization problems. Therefore, GA have been used by many researchers to develop new methods for detecting network or graph communities. These algorithms use modularity as a fitness function, in order to evaluate the quality of discovered communities.

Existing GA-based methods suffer from certain limitations, especially with regard to the creation of the initial population (IP). Generation of IP is usually a random process, resulting in low fitness-valued initial population, e.g. by connecting distant nodes. So the algorithm often sticks on local optimum and take much longer to converge.

The purpose of this work is to implement a new GA-based algorithm that produces a high-quality IP

using the Clustering Coefficient (CC) metric (CC-GA algorithm). An IP node will only connect to a neighbor with the highest CC value, which leads the densely connected parts of a network to be divided into separate communities, since algorithm splits the initial network based on local bridges. Such an improved IP may lead the algorithm to reach a near-global optimum much faster, requiring fewer iterations. Furthermore a new, extended mutation method is introduced, that exploits the discovered community structure to create larger communities.

II.4 AMKM/WAMKM

Adjacent **M**atrix-based **K**-**Me**ans clustering method (**AMKM**) and **W**eighted **A**djacent **M**atrix-based **K**-**M**eans clustering (**WAMKM**) are proposed as two novel clustering methods to improve the renowned k-means clustering algorithm and commonly used algorithms such as k-means++ and spectral clustering. The basic goal is to simplify and provide a less computational demanding method for partitioning of data points into clusters-communities.

The algorithm relies on calculating the similarity between data points by building two new representations of the original input datasets via an Adjacent matrix and a Weighted Adjacent matrix for the methods respectively. Following that, k-means clustering is applied on the new representations to output the clustering results in both cases.

Since the algorithm follows a machine learning approach in the paper based on predetermined identified communities, we decided to develop also a variant version to apply it for community detection for this study.

III. METHODOLOGY

The aim of the work is to assess the algorithms' applicability in social networks and to identify possible issues that make them more or less appropriate for certain use cases. The work involved the implementation on the basis of the reviewed papers' descriptions and references, whereas a small variation of the AMKM/WAMKM was developed to adapt it for use with the common datasets for the purpose of community detection

More specifically, we followed a stepwise implementation in the manner each individual reference paper approaches each proposed solution. The comparative analysis is performed on the following common datasets:

- Zachary Karate Club
- American Football
- Dolphins
- Polbooks
- Emails EU

The selection was made in order that we can address an adequate pool of already tested datasets with different characteristics and variable number of communities against our algorithms.

The datasets were retrieved and incorporated in our algorithms in various formats (as edge lists in .txt format, in gml format or directly loaded via already implemented graphs in Networkx, e.g for the Karate Club) depending on the requirements of each algorithm. The common assessment metric used is Modularity, whereas where appropriate other indicative metrics or graphs to enhance the evaluation per algorithm are presented as well.

The detailed methodology followed for the implementation per algorithm is presented in the sections below.

III.1 ASOCCA

In the process of implementing ASOCCA, the main target was to be able to reproduce the exact same results regarding the datasets used in the paper [1]. The code was written in python utilizing also, the built in functionality of the networkx library.

A very helpful feature in ASOCCA's paper regarding our attempt to implement the algorithm, was the usage of a small graph of 11 nodes as an example from the authors, through which the implementation of the algorithm amongst with the results to be expected in the end of each phase were presented. The above example was used in order to assess our progress in the process of creating our ASOCCA's implementation. We were able to implement each of the following steps in ASOCCA's workflow as presented in the figure below.

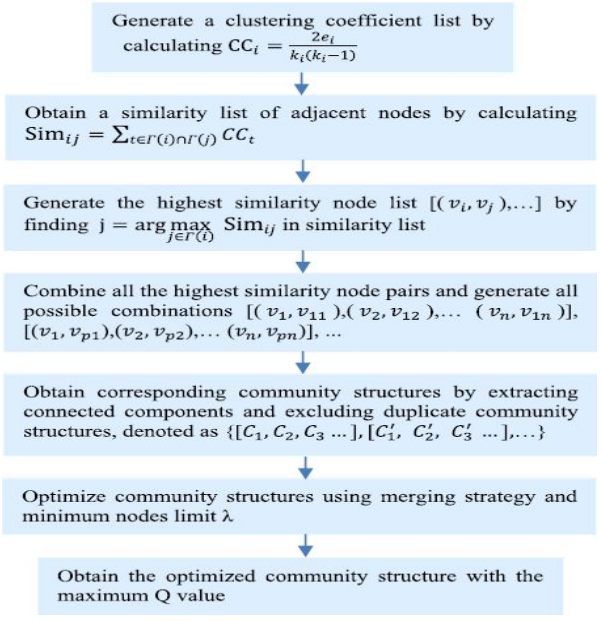

Figure 1. ASOCCA's workflow

While experimenting with our implementation, two major issues occurred. Despite us being able to reproduce the exact same results on all of the datasets used by the authors of the papers up to 12 decimal places, when the algorithm was applied on different

datasets than the ones in the paper (specifically the football and e-mails datasets), the program crashed. By analyzing the code it was clear that our implementation was correct compared to the one proposed in the paper and behave exactly as it was expected to behave regarding the paper's datasets. It was the paper's implementation though that had two issues. The first one, was that there was no provision for handling cases were an edge was connecting a node with itself. To tackle this issue we considered that this edge beard no information regarding similarity of two nodes and cases like this were not taken into account in the calculations of the various algorithm's stages (calculation of clustering coefficient, similarity index, most similar nodes etc.). The second and maybe the most serious omission was that there was no provision regarding isolated nodes for the creation of the final partition of the graph. The paper's implementation did not take such nodes into account, giving invalid partitions in the final results. In order to tackle this issue, we considered that since the whole algorithm is based on similarity of adjacent nodes, it is justified to assume that an isolated node with no edges or neighbors, is not similar to any other node. Thus, for the final partition, each isolated node formed its own community. The two approaches described above, not only had no impact on the results of the paper's datasets but also, gave the algorithm the necessary functionality to process datasets with isolated nodes and nodes with edges to themselves. Based on the above we did not only manage to reproduce the algorithms results and behaviour based solely on the paper's description, but also managed to improve ASOCCA's functionality.

III.2 SCAN

For the implementation of the SCAN algorithm, python and networkx package were used. The use of NetworkX's built-in functions such as graph.nodes(), graph.edges() and graph.neighbors() helped at several occasions and reduced the difficulty of the task.

The first task was the creation of the algorithm exactly as it was described in the papers [2] and [5]. The execution of the algorithm took place in the following steps:

1. Calculating structural similarity for network edges.
2. Finding the core nodes of the network.
3. Defining the clusters.
4. Classifying the remaining nodes as bridges or outliers.
5. Evaluating the results.

The first test of the algorithm was conducted with a small test graph of 11 nodes in paper [5] because it was the only example with values of structural similarity which is a crucial metric for the success of the algorithm. For further testing, the two customer segmentation datasets that are cited in the paper of the original algorithm were used. The algorithm was able to produce the same clusters as in the papers.

The second task of this project was the comparison between the algorithms. The datasets that were used had known communities but most of the real world algorithms have their communities unknown. One of the most used metrics for evaluation in datasets with unknown communities is modularity. For this reason, the next task was the configuration of the algorithm in order to score the best possible modularity scores.

Finally, because of the vagueness in the selection of the best values for the hyperparameters m and e which are used in the definition of the core nodes, it was decided to try several different combinations of these parameters and choose the best of them based on modularity score. Although the choice of trying to find the best possible modularity score increased the execution time, the algorithm is still extremely fast, therefore is suitable for the computation of large graphs.

III.3 CC-GA

The implementation of the CC-GA algorithm used the Python programming language as well as the appropriate functionality of the Python's NetworkX module. Next, the main points of the algorithm will be mentioned.

IP Representation

CC-GA uses locus-based adjacency IP representation, in which each member (chromosome) is represented as a separate graph (Fig.2). The size of each chromosome is equal to the number of original network nodes. Each node is linked to only one of its neighboring nodes. An important issue was to develop appropriate code for the reverse process, that is, to detect the community structure from the adjacency table, as shown in Fig.2.

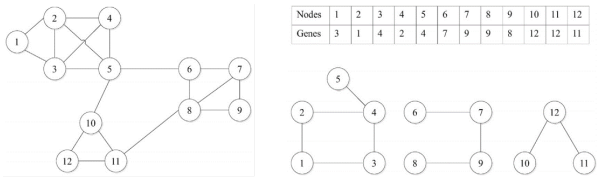

Figure 2. On the left side is the network under investigation. A chromosome is represented as an adjacency matrix (right top) which corresponds to a particular community structure (right below).

Population Evaluation - Fitness Function

Population evaluation is based on the well-known metric Modularity, which is widely used in this kind of algorithms, and represents a measure of the quality of network community structure. The higher the value of modularity (in the range of [-1,1]) the better the quality. The proper functionality of the python's NetworkX module is used to measure the modularity.

Genetic Operators

a. Uniform Crossover

Crossover operator combines the genes of two different chromosomes to create a new chromosome. The uniform crossover approach is used, since it maintains the connectivity of the genes (nodes) in new chromosomes. A random binary vector is created, with a length equal to the number of nodes in the network. Binary vector values determine which parent will donate the gene to the offspring, at each particular position of the chromosome. A probability value (Crossover rate) is specified, which defines whether or not to crossover 2 chromosomes.

b. Mutation

Based on a probability value (Mutation rate), a random gene (node) in each chromosome is chosen to be replaced with another random node. Reference paper does not specify whether the new node will have to be one of the existing neighbors or any other random node. In the implementation code the first approach was chosen.

c. Extended Mutation

Based on a probability value (Extended Mutation rate), in each chromosome a random gene (node), which must have neighbors belonging to different communities, in the original network, is chosen to exchange its existing neighbor with another random one, belonging to different community. This ensures that smaller communities are merged into larger ones, a procedure that increases the modularity value.

Population Update

After each Genetic Operator application, the performance of the resulting population is measured and compared to that of the already existing population. Only if the new population shows an increased performance will the old population be replaced. At this point, the reference paper doesn't clarify whether it takes into account the best chromosome performance (single best modularity score) or the whole population average performance. In the implementation code the first approach was chosen.

III.4 AMKM/WAMKM

The implementation deployed as well Python programming language and the appropriate functionality of NetworkX module.

In contrast to the first three algorithms, aiming at directly identifying the communities, the approach of AMKM/WAMKM in the reference paper is that they work on a *pre-set* number of clusters-communities. Their performance is then assessed versus the actual communities provided by the ground truth available. To overcome this and work on the chosen common approach with the rest of the algorithms, a *variant* version was developed to test their capacity to identify the suitable number of communities too. The basic paper methodology for the implementation is described below.

Given an undirected graph G and a set of input data points, the algorithm is based on the construction of the Similarity matrix W, which in turn produces the Adjacent matrix A by determining the distance between two data points via a similarity function, Gaussian in this case. An actual variation of the algorithm was developed to work directly with the Adjacent Matrix since the selected common datasets provide no features. The difference in the weighted version WAMKM is that it builds a new Adjacent matrix Z instead via a weighted vector H applied on A. In both methods, the k-means clustering is applied to the Adjacent matrix produced in the former step.The implementation basic steps are briefly referred below:

Provide the Input Dataset, Set the number of clusters k

1. Produce the Similarity Matrix W (*if features exist*)
2. Calculate the Adjacent Matrix A
3. Calculate the weight factor H
4. Calculate the weighted Adjacent Matrix Z
5. Run k-means clustering on Z

In the variant version the most suitable No of clusters-communities is determined by iterating the algorithm for different values of k and identifying the best on the basis of common metrics.

IV. RESULTS AND FINDINGS

IV.1 ASOCCA

Regarding the results of ASOCA, as stated in the methodology section, our implementation was able to reproduce the results presented in the paper [1] up to 12 decimal places for the karate dataset and the example graph of the paper. Due to the fact that ASOCCA is a deterministic algorithm we were able to achieve such accuracy. There were deviations in modularity' s values in some cases though, due to the fact that during a specific phase of the algorithm, from all the possible community structures only 100 are chosen in order to reduce algorithms complexity. There is no reason to prefer a certain possible community over another and specifying which 100 possible community structures will be chosen for further preprocessing is not considered important by the authors. Thus, different implementations of the algorithm may choose the combinations to be processed in a different manner, since the authors do not specify how those combinations should be chosen. Due to the fact stated above, in datasets from which more than 100 different possible communities can be extracted, slightly different values in modularity may occur. Despite that, in order to be sure that our implementation was correct, we tried choosing different sets of 100 possible communities and we were able to reproduce identical results as the ones presented in the paper. The modularity achieved for each paper's dataset is as presented below.

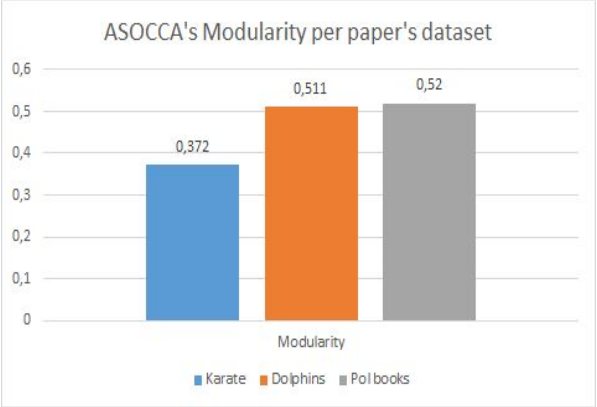

Figure 3. ASOCCA's modularity on paper's datasets

The performance in facebook, internet and amazon datasets was not tested due to hardware limitations and to the fact, that since the success in implementing the algorithm was established with the reproduction of identical results to those presented in the paper, we considered it to be more useful to apply the algorithm to datasets that were not in the original paper. An approach that as stated in the methodology section, revealed omissions and ambiguities in the original algorithm' s implementation and led in creating an improved version of ASOCCA.

IV.2 SCAN

As stated in section III.2 the implementation of SCAN produced the same clusters and allocated all the nodes correctly. The created clusters of datasets CG1 and CG2 are shown in the next picture.

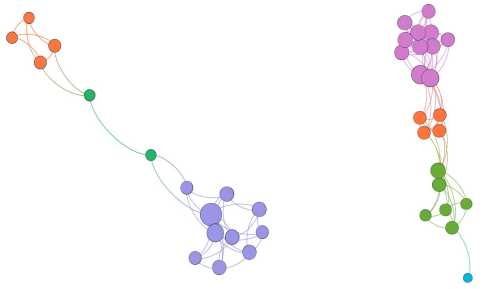

Figure 4. SCAN results for CG1 and CG2 datasets

In CG1, SCAN identified two communities at the edges of the graph and two outliers in its center. In CG2, there were 3 communities and an outlier in the bottom edge of the graph.

The two remaining datasets that are described in the paper were not tested due to the fact that the exact datasets or their ground truth communities were not found in the Internet. Specifically, a newer version of American Football dataset is used for the comparison of the algorithms in section IV.5

IV.3 CC-GA

To evaluate the performance of the implemented algorithm, as compared to the algorithm described in the reference paper, 4 different real-world networks were chosen (Zachary Karate Club, American Football, Dolphins and Polbooks), that were also used to evaluate the original algorithm. In each case the same values of the algorithm parameters were used as proposed in the reference paper. Obtained results show that implemented CC-GA performs equally well, showing an average performance difference of just 0.8%, as compared to published results (Table 1). These differences may be due to the points of implementation mentioned above, that the reference paper leave unclear.

	CC-GA algorithm	
Datasets	**Reference**	**Implemented**
Dolphins	0.526	0.527
Football	0.563	0.549
Karate	0.420	0.415
Polbooks	0.525	0.527

Table 1. Performance comparison of two versions of CC-GA algorithm on 4 common datasets. Values represent the mean modularity score of a number of independent experiments.

In order to compare the CC-GA algorithm against the rest algorithms presented here, an additional dataset (Emails EU) was evaluated. CC-GA showed competitive results over the other algorithms. The overall results are summarized in the next section of the paper.

IV.4 AMKM/WAMKM

The first step to validate our approach a cross check versus the reference paper was made. The Diabetes dataset used in the paper was selected as a reference. The results show a similar trend in determining the accuracy metric between the paper and our implementation. A demonstration of the results for WAMKM is presented in the figure below versus the changing values of hyperparameter σ.

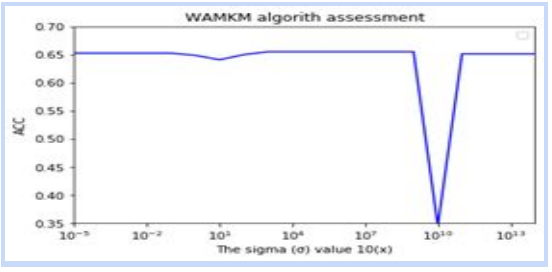

Figure 5. WAMKM results for Diabetes dataset

The development of the variant approach involved the determination of the most suitable number of clusters-communities. An assessment based on a combination of the Elbow Method and of common metrics for the well-known Karate Club (for a given σ defined as the mean of Similarity matrix proposed in the paper) allows to determine that the best No of communities is two (2), which coincides with the best performance in terms of Accuracy. To accommodate the fact that the labelling of the original data points (ground truth is not known) permutation is performed.

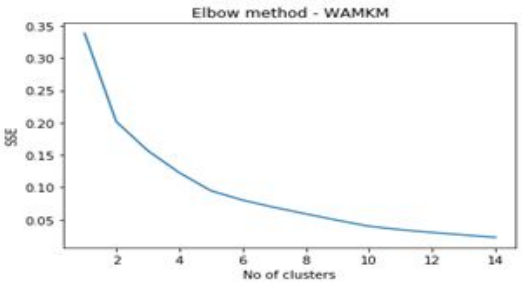

Figure 6. Elbow Method for Karate Club

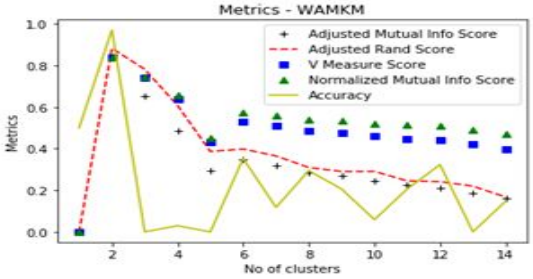

Figure 7. Different Metrics for Karate Club

However, in other cases it can become more difficult to determine clearly the suitable number of clusters. For instance, for the American Football or Emails dataset the actual number of communities of the ground truth is not easily discernible.For the purpose of cross reference to the paper, we assessed also common metrics (like Accuracy ACC or Normalized Mutual Information NMI, see a sample of the results in Table 2 below) for different values of the hyperparameter σ for the given datasets.

Dataset	AMKM	WAMKM	SPCL	AMKM	WAMKM	SPCL
	ACC			NMI		
Karate	0, 97	0,82	0,94	0,84	0,33	0,73
Email	0,04/ 0.46*	0,004/ 0.30*	0,001 0.48*	0,50	0,35	0,53
Polbook	0,10 0.69*	0,43/ 0.71*	0,13/ 0.83*	0, 34	0,38	0,54

Table 2. ACC/ NMI for various datasets *(*indicates permutation)*

We note that the performance of the algorithm is variable depending on the actual dataset used, whereas the performance in our tests remained stable, not affected by the changing values of σ or the number of iterations for the given datasets. As regards its applicability, no safe conclusion can be reached.

IV.5 Algorithms' Comparison

For the final phase of the project, after the implementation of all the algorithms, we experimented with various datasets in order to extract meaningful insight about their performance and their ability to partition a given graph efficiently. The measure used for the evaluation of the resulting partitions and the comparison of the algorithms is modularity.

The results indicate a more or less uniform performance, with the exception of the AMKM/WAMKM for a single dataset (Dolphins) only, which behaves though equally badly in terms of modularity for any given number of clusters. This is possibly due to the peculiarity of the dataset itself and the nature of the algorithm, which is not oriented for community detection. In the following graph the modularity of the algorithms implemented and tested for each dataset is presented.

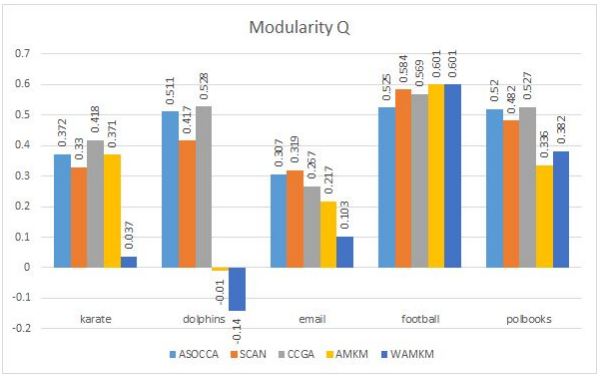

Figure 8. Algorithms' modularity for each dataset

For the sake of comparison and better assessment of the results, the datasets used were the ones with ground truth communities. In the chart below the number of communities estimated from each algorithm, amongst with the ground truth number of communities for each dataset is presented.

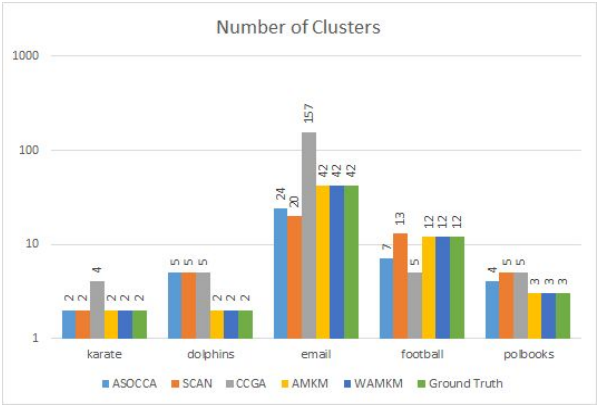

Figure 9. Number of communities for each dataset

Based on the data presented above, it is justified to assume that no algorithm has a significantly better or worse performance than the others. The five algorithms yield similar results regarding modularity in the particular datasets used, except AMKM/WAMKM for the dolphins dataset.

Thus, the five algorithms can be considered interchangeable regarding their performance on undirected, unweighted graphs of similar scale to the evaluation datasets used, with their major differences laying on other aspects, such as time complexity or the broad context of their implementation - i.e. the usage of machine learning methods or not. For more complex datasets with numerous nodes and edges, one should take into account that ASOCCA would require more running time than SCAN, AMKM and WAMKM. CC-GA is the most complex of the five algorithms and it is suggested to be used only in small and medium scale datasets. On the other hand, it seems that ASOCCA, CC-GA and SCAN are more consistent regarding their performance in this kind of datasets compared to AMKM/WAMKM.

In the following figures, the different partitions given by each algorithm implemented in this project regarding the american football dataset are presented as an example of their respective performances. It is worth noticing that the number of the ground truth communities for this particular dataset is 12.

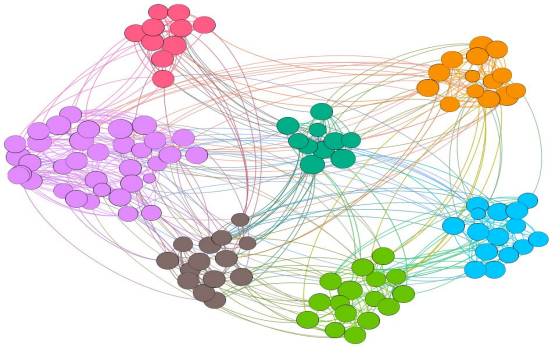

Figure 10. ASOCCA communities for American Football

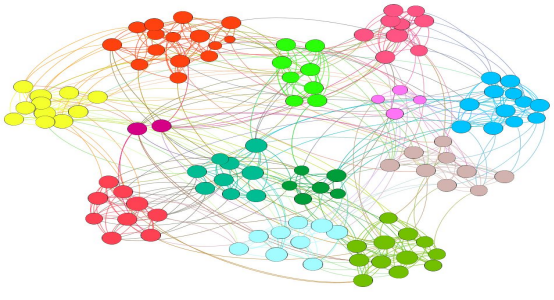

Figure 11. SCAN communities for American Football

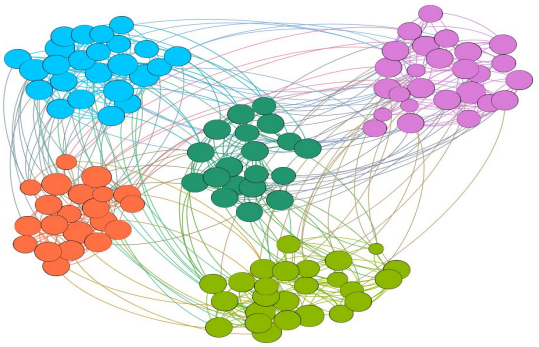

Figure 12. CC-GA communities for American Football

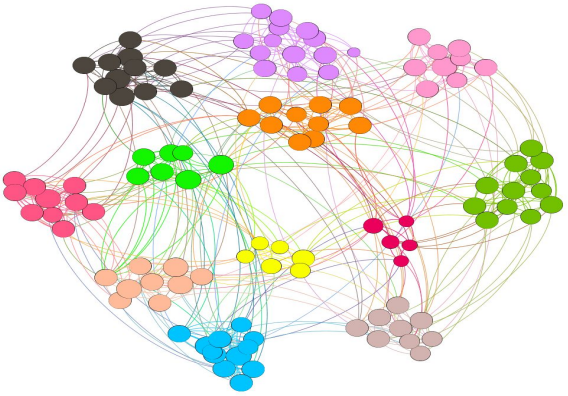

Figure 13. AMKM communities for American Football

V. CONCLUSIONS AND FUTURE WORK

This work has demonstrated the applicability of the implemented algorithms for community detection overall with a comparative performance despite their different approach. Even the ones (AMKM/WAMKM) not by nature oriented for this purpose appear to be able to identify communities, although in a not so straightforward way.

As concerns future work, we propose that the current work is extended to test datasets of variable size and number of communities beyond the ones tested in the frame of this project to simulate as much as possible social networks. Additionally, it would be useful to investigate the applicability of the algorithms in newly constructed real-world datasets as well as to introduce additional metrics used in the field that could effectively assess their performance to a greater extent.

In a further future step, the delivery of a new algorithm that could possibly integrate certain features shared amongst the chosen algorithms could be discussed.

VI. GITHUB REPO - WHO DID WHAT

For the implementation of this project the tasks were distributed as follows:

1. All experiments and implementation regarding ASOCCA: Dimitrios Vasdekis
2. All experiments and implementation regarding SCAN amongst with Gephi visualizations: Doxakis Chovardas
3. All experiments and implementation regarding CC-GA: Theodoros Liapikos
4. All experiments and implementation regarding AMKM and WAMKM: Konstantinos Serderidis

We would like to point out that our collaboration was a challenging task since we had no acquantainance with each other prior to this project and our daily schedules were somehow incompatible. However, these difficulties led us into adopting new methods for remote collaboration through the usage of online tools like video calls, shared documents and git and we would like to believe that we managed to overcome any obstacles and cooperate smoothly in order to present a decent project.

The source code of the project can be found on https://gitlab.com/VasdekD/community-detection?fbclid=IwAR1Vwr4yV292wUL2ZJ3_UNA7WbqAPuBZRY14gzgF-86g7eB8iLV31UQHvOU

REFERENCES

[1] Pan, X., Xu, G., Wang, B., & Zhang, T. (2019). A Novel Community Detection Algorithm Based on Local Similarity of Clustering Coefficient in Social Networks. *IEEE Access,* *7*, 121586-121598. doi:10.1109/access.2019.2937580

[2] Xu, X., Yuruk, N., Feng, Z., & Schweiger, T. A. (2007). Scan. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '07*. doi:10.1145/1281192.1281280

[3] Said, A., Abbasi, R. A., Maqbool, O., Daud, A., & Aljohani, N. R. (2018). CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks. *Applied Soft Computing*, *63*, 59–70. doi: 10.1016/j.asoc.2017.11.014

[4] Jukai Zhou, Tong Liu, Jingting Zhu, (2019). Weighted adjacent matrix for K-means clustering. # *Springer Science + Business Media, LLC, part of Springer Nature 2019.* https://doi.org/10.1007/s11042-019-08009-x

[5] Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification,* *2*(1), 193-218. doi:10.1007/bf01908075

[6] (n.d.). Retrieved from https://www.cs.unm.edu/~aaron/research/fastmodularity.htm

[7] Inoubli, W., Aridhi, S., Mezni, H., Mondher, M., Nguifo, E., (2019). A Distributed Algorithm for Large-Scale Graph Clustering. *hal-02190913, version 2*