Journal of Engineering and Applied Sciences 14 (Special Issue 6): 9289-9293, 2019
ISSN: 1816-949X

Ant Colony Optimization Method for Graph Clustering

Ondrej Klapka and Adrea Vokalova
Faculty of Informatics and Management, University of Hradec Kralove, Hradec Kralove, Czech Republic

Abstract: Many current practical problems in many areas (such as bioinformatics, social networks, science databases, etc.,) can be solved through using graph algorithm which is consisting of vertices (nodes) and edges interconnecting these vertices. Although, the graph is formally very well described, some problems over the graphs have not yet been formalized or their solution is a NP-hard problem. These problems also include the process of identification of clusters in a graph. The following text provides a way to use the method of ant colonies optimization algorithm for the identification of clusters in the graph.

Key words: Ant colony optimization, graph clustering, algorithm identification of clusters, social networks, vertices, solution

INTRODUCTION

Graph networks consist of vertices and edges and are currently the objective of intensive scientific research. Many natural and developed structures such as the internet, social media relationships, mutual citations of scientific publications and many more can be intuitively understood as a graph. Development of Computer Technology (ICT) in recent decades enhanced the interest in the area of graph networks because it is easily processable and machine-oriented structure which can represent complex relationships between data objects.

However, very often the created networks are quite extensive and it is not usual that the networks may consist of hundreds of thousands or millions of vertices and edges. Graph networks of this scale are too difficult to process using conventional methods even with a help of the contemporary and very powerful hardware.

Classical algorithms working with graphs provide quite exact solution of addressed problem but often at the cost of high computational power. On the other hand, heuristic methods provide just an approximate solution of the task but the computational complexity of these algorithms is considerably lower than in case of conventional algorithms. Also, some problems are NP hard to complete, so, the process of finding a solution using a conventional method is nearly impossible.

Furthermore, it should be mentioned that some problems have still not been formally defined and assumptions for finding a solution are then perceived as intuitive rules which must be adhered to. These problems include among other, cluster analysis which is aimed on finding all sections of graph whose vertices are mutually and densely linked to edges. The result of such an cluster analysis algorithm are called clusters and sometime are referred as communities (Girvan and Newman, 2002; Schaeffer, 2007). The following text is dealing with a design of clustering algorithm using the method of ant colonies.

State of art: The goal of the cluster (community) recognition process is to divide graph vertices into groups whose vertices have a greater mutual interconnection and, on the contrary, the interconnections between individual groups of vertices are lower. Generally, the cluster is not formally described and this is only a term of densely interconnected parts of the graph (Schaeffer, 2007). In the area of formalized problems in clustering is the similar problem named graph partitioning. The partitioning consists in distribution of graph's vertices into groups of predefined size, so that the amount of edges among these groups is minimized (Fortunato, 2010). However, in the case of clustering is not known the size and often the number of groups in advance and the algorithm must identify these two important values by itself.

Due to the absence of a formal description, there are no exact methods, how to find these clusters. On the other hand there can be found several heuristic methods which provide results equivalent to the concept/idea of what is a cluster (Schaeffer, 2007). Particular heuristic methods used for the cluster's recognition are based on very different ideas as well as their accuracy varies (Schaeffer, 2007; Fortunato, 2010).

Corresponding Author: Ondrej Klapka, Faculty of Informatics and Management, University of Hradec Kralove, Hradec Kralove, Czech Republic

There can be found many methods and metrics which declare how to compare the quality of the results in the case of particular clustering methods in the contemporary literature. Nevertheless, because of the absence of a formal cluster's description, these result quality measuring methods are not enough reliable. Small graphs are still best evaluated by a human person. The user can compare the graphical representation better and also can compare if his own idea is corresponding to a result of clustering algorithm. In case of large graphs, there is no accurate way to assess the quality of the found solution (Brandes *et al*., 2003).

MATERIALS AND METHODS

The girvan-newman algorithm: The method providing to be very accurate during the cluster finding process in graph is called the Girvan-Newman algorithm. This algorithm is based on a calculation of evaluation for each edge. The disadvantage of this algorithm is its high computational complexity. During the processing of this algorithm is necessary to find the shortest path between any two vertices in the graph. Girvan and Newman (2002) It is obvious that with the increasing number of vertices the computational complexity of the algorithm increases also steeply and this algorithm is unsuitable for very large graphs. The total computational complexity of one iteration in optimized algorithm for the unevaluated input graph is:

$$O(m^2n) \quad (1)$$

Where:
m = The number of vertices and
n = The number of edges in a graph G

The random walk algorithm: Another way to identify clusters is a random walk algorithm. This algorithm is based on the idea that during the random browse/walk on graph edges exists higher probability that we will remain in a single cluster. These algorithm then works in individual iterations with the probability of transition from the vertex v_i to the vertex v_j. Transition probabilities are derived accordingly to the degree of the vertex. It is necessary to establish the transition probability for each pair of nodes v_i and v_j in a graph G by creating a probability transition matrix. Repeated exponentiation of the probability transition matrix together with its other modifications gives us the probability transition matrix after n steps (Macropol, 2009). The increasing graph structure of course quadratically increases size of the transition matrix and therefore, the memory requirements of the algorithm. These matrices must be repeatedly multiplied with each other thus significantly increasing the computational complexity of the algorithm. Computational complexity of the random walk algorithm is for each iteration defined as:

$$O(m^3) \quad (2)$$

where, m is the number of vertices in a graph

Spectral clustering: Third frequently used method is detecting the communities by spectral clustering. Spectral clustering is based on the adjacency matrix from which is composed Laplacian matrix. From the Laplacian matrix is then calculated eigenvector from which is again formed a matrix that needs to be then normalized. The result matrix is clustered by the k-means algorithm (Tsironis *et al*., 2013). To achieve lower computational complexity or to make parallel algorithm run there exists many variations of this algorithm. It can be adapted for specific areas of application. The complexity of the algorithm presented in (Tsironis *et al*., 2013) is:

$$O(m^3) \quad (3)$$

where, m is the number of vertices in the graph. Langone *et al*. (2012) can be found a variant of an algorithm which complexity in some cases is shown as:

$$O(m^2) \quad (4)$$

where, m is the number of vertices in a graph.

Ant colony optimalization: Simulation of the large insect's group behaviour or any animal in general has in recent years proved to be an interesting alternative in optimization solving problems in comparison to the classical methods. Although, individual creatures forming groups show no complex behaviour (often merely a question of the ability of predetermined responses to a limited group of complaints) large groups of these creatures possess very advanced behaviour. The first ant colony algorithm emerged in the early 90's and it turned out that using these advanced capabilities called "swarm intelligence" can be used in a wide variety of practical problems (Dorigo *et al*., 2006).

Optimization algorithm using ant colonies essentially simulates the behaviour of individual ants. The assembled simulation models often utilizes discrete time when the passing time is divided into time intervals (steps) and one step corresponds to one iteration of the algorithm. Individual ants forms colony and are modelled as a

software function that based on input data, perform simple action. In each iteration every ant once evaluates an input data and performs one action.

Another important element in modelling ant colonies is called pheromone. The pheromone is a substance naturally produced by ants returning from the source of food to the anthill. Other ants while deciding about the path prefer just the directions in which the pheromone is most intense. Pheromone remains most intense in places where most ants passed by on their track from the source of food to anthill. Consequently, in many cases the ants find very quickly shortest path between a food source and the colony. On the other hand the paths not leading directly to the source of food very quickly cease to exist. (Deneubourg *et al*., 1990). The pheromone is a sort of environment memory that helps ants in decision making.

Clustering with the help of an ant colony: When the clustering is implemented by using an ant colony it can be formed on the idea of a random walk algorithm. In the assembled model the ants will randomly move on the graph edges. This application of graph clustering was also studied in emergent colonization and graph partitioning. In each iteration of the algorithm each ant of model will made just one step, i.e., the ant will move from the vertex v_i from the set of vertices V in the graph G to the vertex v_j from the set of vertices V while the vertices v_i and v_j must be connected by an edge. The ant passes exactly one edge e within the set of edges E on the graph G and passes one step of the model.

Figure 1 shows a simple graph where the two clusters can be fully recognized by intuition. The cluster C_1 consists of the vertices V_1 = {A, B, C, D} and the cluster C_2 consists of the vertices V_2 = {E, F, G}. The idea of the proposed algorithm is as follows:

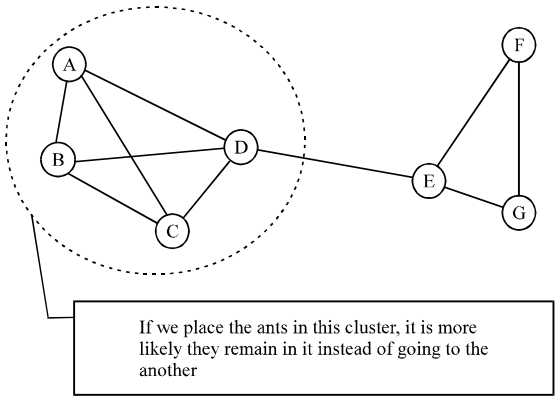

Fig. 1: Principle explanation of the introduced algorithm

All the ants will be placed to the selected vertex v_s. Ants will move step by step along edges between vertices and the algorithm stops after a predefined number of iterations. After the simulation stops there will be significantly more ants in the same cluster as the vertex v_s is located compared to the vertices that are not part of the cluster in which the vertex v_s is located.

If we place ants in the vertex A in the graph shown in Fig. 1, accordingly to the shown algorithm it is more likely that the ants remain in this cluster before they pass into the second cluster.

Model draft: The whole model is constructed so there is no need to model each ant as a separate software entity and implement the movements of individual ants. It works only with the numbers of ants relocated at every step which reduces the computational complexity of such an algorithm.

The vertices to which the ant moves in the iteration of the model are selected on the available options (edges exiting from the vertices where ant is currently located) and are based on the pheromone marks of each edge allowing move to the next vertex. The number of ants (C_j) moving from the vertex v_i to the vertex v_j is defined by the equation:

$$C_j = C_i\left(p.\frac{1}{\deg(v_i)}+f.w\right) \quad (5)$$

Where:

C_i = The number of ants currently located at the vertex v_i
p = The probability constant
deg (v_i) = The grade of the vertex v_i
f = The pheromone constant and w is the weight of pheromone

The values of p and f listed in a previous equation are determined by the ratio between the ant's random decisions and following the pheromone marks. The values are based on stated equation:

$$p = 1\text{-}f \quad (6)$$

With the increasing value of p the ants behave more randomly while more ants follow the existing pheromone marks when increasing the value f. Weight of the pheromone (w) is then determined by the relation:

$$w = \frac{e_{i,j}}{\sum_{n=1}^{\deg(y_t)} e_{i,n}} \quad (7)$$

Where:

$e_{i,j}$ = The current pheromone mark on the edge e connecting the vertices

v_i, v_j and $e_{i,n}$ = The actual pheromone mark on the edge e-Connecting the vertex v_i with one of the neighbouring vertices

Each ant passing over the edge of the graph leaves a pheromone mark by adding a constant defined as one of the model parameters to the existing pheromone mark. After each iteration of the algorithm the pheromone mark is lowered by a fix constant defined as an additional model parameter.

When the algorithm is set all the ants are placed into one of the vertices. Then the algorithm is started and each iteration relocates ants between vertices of the graph. The relocation is based on the relations stated above. After sufficient number of iterations (the derivation of iterations number will be discussed further) the model stops and the vertices are divided into two groups according to the number of ants currently located in the vertex.

The vertices are divided into two sets v_l and v_h accordingly to the value of x. In the first set V_l are counted vertices with number of ants smaller than x and the second set V_h contains vertices having number of ants greater than x. The value of the parameter x will be set by the user and allows to correct the model outcome according to current needs. If among any pair of vertices in one group does not exist any edge, then the vertices belongs to a different cluster.

It is possible to repeatedly apply the algorithm to the individual clusters to obtain a larger number of clusters. The input of the algorithm is made by individual graphs that are created by vertices and edges located in the cluster.

Setting the model parameters: The model contains parameters which should be determined experimentally. The parameter value is determined by one of three means:

- Fixed parameter the value depends on the size or characteristics of the input graph
- Graph parameter the value is based on the characteristics of the input graph (number of vertices in the graph)
- User value a value determined by the user as needed

Each parameter was determined on small test graphs (10-15 vertices). On such a graph can be observed the progression of the number of ants in each vertex and the model can be corrected accordingly. Designated parameter values are shown in the table below including how the parameters are derived. While determining the parameters of the model it is advisable to work with a larger number of ants relative to the size of the input graph (see derivation in the table). The number of ants in the individual vertex should be integers. Small number of ants generates "rounding" error in the model. Conversely, a higher number of ants mean greater accuracy of the model. As the model is constructed, the number of ants has no influence on the complexity of the algorithm simulating the proposed model.

Asymptotic complexity: The introduced algorithm works in iterations and the number of iterations is derived from the magnitude of the input graph. In each iteration there is need to determine the number of ants for every edge. Consequently is necessary to weaken the pheromone marks for each edge. Therefore the asymptotic complexity of the algorithm is:

$$O = (m.n) \quad (8)$$

where, m is the number of graph vertices and n is the number of edges.

RESULTS AND DISCUSSION

Model testing: The proposed model was tested on network created from the digital library database content. Network were compiled on the basis of the citing relationships between scientific publications in the Citeseer database available at Bergstrom, using visualization and analytical application introduced in (Klapka, 2013).

Figure 2 is shown the result of cluster analysis carried out by the algorithm over the graph based on the citation network of the selected publications from the CiteSeer database. There are 78 graph nodes and 151 edges. When looking at a visualized graph it is clear that the proposed algorithm could correctly separate densely interconnected peaks from the less coherent.

Figure 3 shown cluster analysis performed over a further series of test graphs which contained a total of 97 nodes and 172 edges. Also in this case we can see satisfactory results, however some vertices were apparently incorrectly classified.

Overall, the algorithm proved to be reliable but in some cases it should be altered appropriately to adjust the individual parameters according to the visual inspection result. The alteration could be based on the values indicated in Table 1. The issue is possible to solve by the dynamic changes in input parameters made by user and by its immediate presentation.

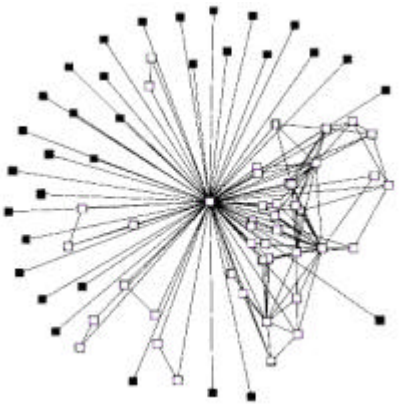

Fig. 2: The results of cluster analysis vertices belonging to the same cluster are marked white

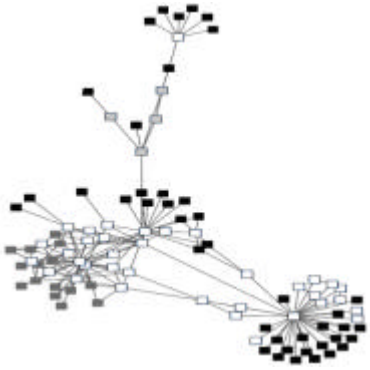

Fig. 3: Color-coded vertices are divided into clusters by the algorithm above the citation network

Table 1: Designated parameter values

Parameter	Type	Values
Ant's decisions ratio (f)	Fixed	0.1
Actual pheromone mark	Fixed	0.3
Actual pheromone mark lowering after each iteration	Fixed	0.03
Number of ants	Graph	150 V
Number of iterations	Graph	3 V
Graph division based on number of ants in vertices (x)	User	-
Input vertex (v_s)	User	-

CONCLUSION

This text has introduced the issue of searching a densely interconnected parts of graphs (clustering) and has also briefly described the selected algorithms of cluster analysis. In the second part of the text there was presented own design of clustering methods, using an artificial ant colony which can in many cases, find at least an approximate solution for NP-hard problems. The introduced method has been implemented and tested using a previously designed application for visualization the content of scientific publications digital library in the form of network diagram. The ability to successfully identify graph clusters while maintaining an acceptable asymptotic complexity was demonstrated when verifying the model. On the other hand, there was shown that the model is very sensitive to the input parameters values. Their inappropriate settings may result in incorrect classification of graph vertices into clusters.

ACKNOWLEDGMENT

This study is supported by the SPEV project No. 2105, Faculty of Informatics and Management, University of Hradec Kralove.

REFERENCES

Brandes, U., M. Gaertler and D. Wagner, 2003. Experiments on Graph Clustering Algorithms. In: European Symposium on Algorithms, Battista, G.D. and U. Zwick (Eds.). Springer, Berlin, Germany, ISBN:978-3-540-20064-2, pp: 568-579.

Deneubourg, J.L., S. Aron, S. Goss and J.M. Pasteels, 1990. The self-organizing exploratory pattern of the argentine ant. Dans J. Insect Behav., 3: 159-168.

Dorigo, M., M. Birattari and T. Stutzle, 2006. Ant colony optimization. IEEE Comput. Intell. Mag., 1: 28-39.

Fortunato, S., 2010. Community detection in graphs. Phys. Rep., 486: 75-174.

Girvan, M. and M.E.J. Newman, 2002. Community structure in social and biological networks. Proc. Natl. Acad. Sci. USA., 99: 7821-7826.

Klapka, O., 2013. Vizualni analyza informaci. Master Thesis, University of Hradec Kralove, Hradec Kralove, Czech Republic.

Langone, R., C. Alzate and J.A. Suykens, 2012. Kernel spectral clustering for community detection in complex networks. Proceeding of the 2012 International Joint Conference on Neural Networks (IJCNN), June10-15, 2012, IEEE, Brussels, Belgium, ISBN: 978-1-4673-1490-9, pp: 1-8.

Macropol, K., 2009. Clustering on graphs: The Markov Cluster Algorithm (MCL). University of Utrecht, Utrecht, Netherlands. http://www1.se.cuhk.edu.hk/~seem5010/slides/yxzhang_MCL1.pdf.

Schaeffer, E.S., 2007. Graph clustering. Comput. Sci. Rev., 1: 27-64.

Tsironis, S., M. Sozio, M. Vazirgiannis and L.E. Poltechnique, 2013. Accurate spectral clustering for community detection in MapReduce. Athens University of Economics and Business, Athens, Greece. http: // citeseerx.ist psu.edu/viewdoc/download?doi = 10.1.1.404.1188&rep = rep1&type = pdf.