A Reliable Graph Clustering Method Using Genetic Algorithm

Goutam Chakraborty[†] and Naoki Sato[†]

†Faculty of Software and Information Science, Iwate Prefectural University
152-52 Sugo, Takizawa, Iwate 020-0693, Japan
Email: goutam@iwate-pu.ac.jp

Abstract—
In several domains, the available information can be represented as a graph where an unit of information is a node, link between two nodes is a relation. Two web-sites on the internet are connected, if there is a hyperlink from one to the other. Members of a social network are connected as a graph. Scientific papers can be connected through common key-words, citations, or authorship. Such networks form communities, where within communities node interconnections are dense, and between communities sparse. Finding those communities efficiently, and central nodes in them, are two major problems for several mining applications. To identify communities one needs to cluster the network. Existing algorithms, like k-means, need a pre-defined value of the number of clusters. In this work, we introduce a genetic algorithm based approach, where the whole graph is automatically divided into its intrinsic clusters, through a few generations of genetic search. By simulations on large networks (up to 1000 nodes) with known clusters, we have shown that the algorithm finds correct clusters in a few generations with very few mis-labeled nodes. With sufficient run time it always finds the correct solution.

1. Introduction

Units of information are often related and form a graph, where a link represents a defined relation. Its analysis reveal interesting, useful and important knowledge. As information is growing, interest in analyzing such network to facilitate useful applications, is getting strong attention. Knowing the community structure leads to higher level knowledge. One classical example of information network is Erdos collaboration graph, where two mathematicians are joined by a link, whenever they co-authored a paper together, and grown inductively. "During the height of Lewinsky scandal, the *New York Times* printed a diagram of the famous people within six degrees (links) of Monica" [1]. On the internet, one web site is connected to another if there is a hyperlink. This will lead to a directional graph. Clustering and ranking (page-ranking) of the web-sites is the key to web searching algorithms. Another example is Twitter, where users are connected by follow-up and follower relationship. Efficient clustering of the graph, where the number of clusters is not known, is the first important step to discover the pattern from a graph.

Social network and many other biological and naturally occurring structures form such relational graphs, which fall into a class of random graphs called scale-free or Small-world network (SWN) [1] [3] [2]. For SWN, the estimated distance (hop count) between two randomly selected nodes is short, and grows proportionally to the logarithm of the number of nodes in the network, while the clustering coefficient is not small. Another property is that, a few nodes have a high degree, and others low. The degree distribution follows a power law [1]. Around nodes with high degree, there are formation of clusters. WWW, electric power grid, network of brain neurons, voter networks, airline routes are only a few examples. Many other naturally occurring phenomenon form this so-called SWN [4]. Physically, Small World Network forms a set of communities, where within a community there is a strong relationship whereas a few links among communities. This work is to find these communities in an evolutionary way, without a-prior knowledge of how many communities are there.

Clustering is an age old problem, where we have data points in a high dimensional space and their mutual distances are defined, or needs to be defined in certain way depending on the application. By clustering, we cluster data points into groups where inter cluster distances are more than intra-cluster distances. Graph clustering problem is different in the sense that there is only the concept of connectivity between two nodes. A good survey of graph clustering is available at [5].

The main contribution of this work is that (1) the algorithm automatically evolves to find the correct number of clusters, and (2) it does not converge in local minimum. We synthetically created SWNs of different sizes with different numbers of clusters to evaluate our algorithm. Benchmark graphs were created using method described in [11], where the clustering results are known. With a large number of experiments, we have shown that with enough generations, the algorithm always converges with all nodes correctly assigned to their clusters.

The rest of the paper is as follows. Section 2 is about related works. In section 3, we explain the proposed algorithm. In section 4, we discuss about motivation of preliminary experiments and their results to plan the final experiments. The results of the final experiments and conclusion are in section 5.

2. Related Works

Graphs created using links between two information nodes could be directed, forming a digraph. In this work, we consider undirected graphs.

Graphs, created from information gathered from social network sites like twitter or facebook, brings to light users' choices and preferences, and promotes business opportunities. In recent years, this is one of the main research topics in social network information [6]. Depending on the target application, different relationsip is used for connecting two nodes. In many applications, not only clustering but also finding central nodes in individual clusters is important.

Leicht et.al. [9] develop ideas of concept equivalence. The link structure around similar nodes is considered as similarity indexes of nodes and classify nodes on basis of this similarity.

Modified k-means clustering can find clusters in a graph [6]. But, one needs to assign the value of k a-priori, a correct guess of which is impossible with many applications.

In [7], a genetic algorithm approach is proposed. Chromosome structure used in this work is similar. Simulation results show that their GA would converge to local minimum around 20% of the time. The reason is the fitness function which defines the search space. We used a different fitness function as explained next.

Algorithm proposed in [8] is to find the part of the network where nodes are densely linked. It gives a mathematical measure of how much modular [2] a network is. We used the same measure to find fitness of a chromosome which is explained here. Suppose, A_{ij} is the adjacency matrix. Nodes i and j belongs to two clusters, C_i and C_j. We define function δ, similar to Kronecker delta function. $\delta(C_i, C_j)$ is 1, when nodes i and j belong to the same cluster. Thus, the fraction of edges that fall within communities, is

$$\frac{\sum_{ij} A_{ij}\delta(C_i, C_j)}{\sum_{ij} A_{ij}} = \frac{1}{2\,E}\sum_{ij} A_{ij}$$

where, E is the total number of edges. Once we cluster the nodes, the more the above value is, the better is the modularity of partitions. As such, it could not be used as the fitness function, as then the best result will be when we have just one cluster. Once we subtract from it, the quantity expected for a random network,

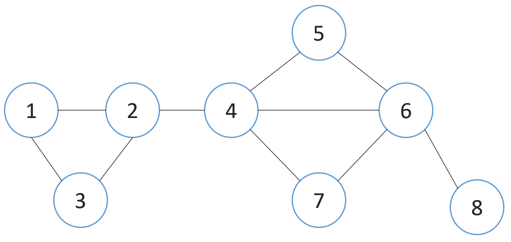
$$Q = \frac{1}{2E}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2E}\right) \times \delta(C_i, C_j) \qquad (1)$$

we get Q which could be a measure of how successfully the network is divided into clusters. Here, k_i and k_j are degrees of node i and j, respectively. If Q is zero, connections within clusters are no better than a random network. A value of 0.3 or more, with clusters properly defined, indicates reasonable good community structure.

3. Proposed Method

We used conventional genetic algorithm [10]. The pseudocode of the algorithm is as follows. The best chromosome is used for clustering.

```
Input: Graph(V,E)
Output: Set of clusters $C_1, C_2, \ldots$
1. Generation of the Initial population
2. Perform Cross-over on selected chromosomes
3. Perform mutation on the selected genes
4. Fitness evaluation and tournament selection,
     formation of next generation
5. if (convergence == true)
     end
7. Else
```



Node	1	2	3	4	5	6	7	8
Chromo some	2	3	1	5	4	5	6	6

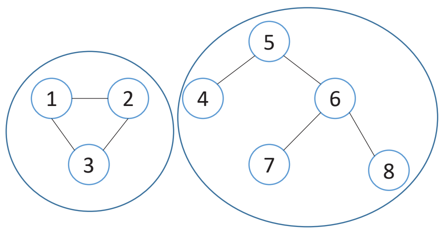


Figure 1: How a chromosomes creates community?

To explain how initial chromosomes are generated, in Fig. 1, we give an example of a 8-node network, a sample chromosome and how that chromosome is interpreted as two clusters.

The length of the chromosome is the same as the number of nodes. Chromosome is the lower row, where genes denote node numbers. The entry of slot #1 is filled by any of the nodes connected to node #1. Since nodes #2 and #3 are connected to node #1, any one of the two is entered. Similarly, below #2 any of the nodes that are connected to node #2 are put. We continue till the node count, here 8 is reached. Genes could be repet, like 5 and 6 appear two times. If average node degree is k and N the number of nodes, k^N is the size of the search space.

The cluster from the chromosome is shown in the lower half of Fig. 1. Here, we connect node 1 to 2, 2 to 3, 3 to 1, etc. till the last node 8. This chromosome divides the network into 2 clusters. Finally, we connect the rest of the links, and evaluate the fitness

3.2. Fitness function

The fitness Q is calculated as described in Eq. 1 [8]. We calculate the value of $Q = 0.56$ for this partition. For a bad chromosome, which would create two clusters as $C_1 = \{1, 2, 3, 4, 5, 7\}$ and $C_2 = \{6, 8\}$, the Q value will be 0.25. Chromosome is first expressed as set of clusters, and then Q value for that clustering is calculated as the chromosome's fitness.

3.3. Crossover, Mutation, Selection

Uniform crossover is used. A mask pattern of length equal to that of the chromosome, consists of a random sequence of 0 and 1, forms the mask. Two chromosomes are first selected for crossover. Where the mask pattern is 1, the entries in two chromosomes are swapped. If the mask entry is 0, leave the corresponding gene as it is. The crossover probability is set to 90%. The population size was 100. At every generation 90 chromosomes were randomly selected for crossover. Mutation probability was 0.1%. During mutation, network connectivity information is used, so that mutation do not generate invalid chromosome.

Tournament selection with tournament size 2 is used. Though convergence is slow because of small tournament size, the exploration of search space was good. The best chromosome is saved at every generation. If the best chromosome of the next generation is better than the one saved, it is replaced.

3.4. Convergence rule

For an unknown network, it is impossible to know what would be the Q value, even when all nodes are correctly assigned to their respective clusters. The convention is to run for a large number of generations and expect that the algorithm will converge to global optimum. The other possibility is to check whether the fitness is improving or not, for a few generations in a row. We performed a set of preliminary experiments, with networks of known clusters, to see how it is converging, and from there to find a cue to set the stopping criterion.

4. Preliminary Experiments

We created a set of benchmark networks [11] for which the clusters and their member nodes are known, and from which Q values for correct clustering are calculated. An example 100-nodes network is shown in Fig. 2. Please note that for visual clarity links within clusters are drawn very short.

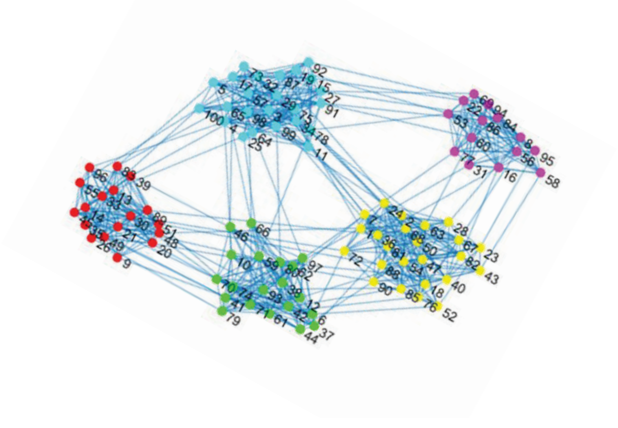


Figure 2: Clusters of 100-nodes network

We created a 500-nodes network and calculated its Q value when all nodes are correctly classified. We run genetic algorithm five times. The algorithm is stopped when the fitness reached the maximum Q value of 0.76. We use the term that 100% Q value is achieved. Depending on the problem and randomly generated initial population, the number of generations needed are different. The results with running time are shown in the top 5 rows of Table. 1. The average run time was about 74 mins. In the next experiment the execution of the genetic search was stopped when fitness reached 95% of the Q value. Results, this time, are shown in the lower 5 rows of Tab. 1. The average run time now is only 11 mins. i.e., a reduction of 7 times.

Table 1: Convergence with 100% Q - 500 node network

Expt #	**Generations**	**time(sec.)**
1	255	2957.7
2	178	2135.6
3	263	3197.5
4	834	10217.4
5	305	3588.2
average	367	4419.2
1	76	643.3
2	75	8226.0
3	109	962.3
4	63	600.2
5	49	468.2
average	74.4	700.0

But, what is the quality of result? The number of clusters in both the cases were same, i.e., we could terminate with correct number of clusters. But, we end up with some nodes mis-classified - some peripheral nodes are assigned to neighbor cluster. For them links towards its assigned cluster is less compared to outgoing links to the cluster to which it actually belongs. They are a few, around 2 to 3, which could be quickly corrected manually. We also did similar experiments with 1000 nodes network with similar results. They are not included due to space constraint. In summery, terminating with 95% of Q we could get order of time saving, still with correct number of clusters. Only a few peripheral nodes are mis-classified, which could be easily corrected manually.

For an unknown problem, as we do not know 100% Q value, this can not be used as a convergence criterion. Number of generations is the simplest stopping criterion. To find the required number of generations, with respect to number of nodes, we run the experiment 100 times with 500 nodes network. The maximum, minimum, and average of Q value, at the end of every 10 generations were plotted in Fig. 3. The average approaches the maximum, as we near 200 generations. In fact, the minimum at 200 is from only one run. From this result, we can claim with high confidence that for a 500 node network, 200 search generations are sufficient. At the end of 200 generations, 77 out of 100 runs the fitness value exceeded 98% of $Q = 0.7805$. For different randomly generated 500 nodes networks, % of Q achieved after 200 generations is shown in Table. 2

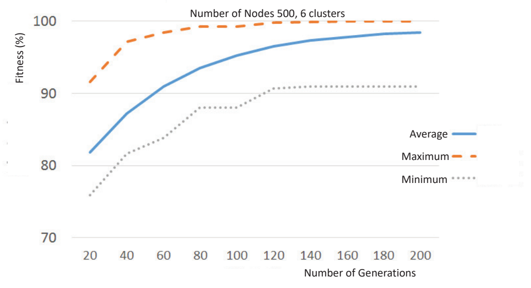


Figure 3: Average of fitness as it improves with generations

5. Conclusion and Future Plan

For a modular network, with high Q value, our algorithm gives better results. Finally, only 2 or 3 fringe nodes are wrongly classified.

Though, the proposed algorithm could always find global optimum, the fitness function computation is complex and prohibitive when the number of nodes is 10,000 or more. We are working on a more efficient fitness function to be able to process a million nodes net.

Table 2: Results: 500-nodes networks, 200 generations

Expt No.	Q value	Q 200 gens	% of Q	# error nodes
1	0.7473	0.7410	99.1	2.4
2	0.7323	0.7151	97.7	3.1
3	0.7753	0.7695	99.3	2.1
4	0.7519	0.7445	99.0	2.8
5	0.7705	0.7638	99.1	1.8

References

[1] Steven H. Strogatz, "Exploring Complex Networks," Nature, vol 410, pp. 268–276, March 2001. 2015.

[2] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in Networks," Physical Review, E, vol. 69, 026113, 2004.

[3] M. Girvan and M. E. J. Newman, "Community Structure in social and biological networks," Proc. of National Academy of Science, USA, vol. 99, pp. 7821–7826, 2002.

[4] Qawi K. Telesford, et. al., "The Ubiquity of Small-World Networks," Brain Connect, Vol1(5), pp. 367–375, December, 2011.

[5] Schaeffer, Satu Elisa, "Graph clustering," Computer science review, Elsevier, 1.1. pp. 27–64, 2007.

[6] Lemaire, Vincent, Fabrice Clrot,and Nicolas Creff, "K-means clustering on a classifier induced representation space: application to customer contact personalization," Real World Data Mining Applications. Springer, pp. 139–153, 2015.

[7] Clara Pizzuti, "GA-Net: A genetic algorithm for community detection in social ," LNCS 5199, pp. 1081–1090, 2008.

[8] Aaron Clauset, M.E.J.Newman, and Cristopher Moore, "Finding community structure in very large networks," PHYSICAL REVIEW E 70, 066111, 2004.

[9] E. A. Leicht, Petter Holme, and M. E. J. Newman,

[10] M Mitchell, "An Introduction to Genetic Algorithms," MIT press, (ISBN 0262631857), 1998.

[11] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi, "Benchmark graphs for testing community detection algorithms," PHYSICAL REVIEW E 78, 046110. 2008.