5-2018

Efficient Reduced-Bias Genetic Algorithm (ERBGA) for Generic Community Detection Objectives

Aditya Karnam Gururaj Rao
University of Missouri-St. Louis, agtk4@umsl.edu

Cezary Janikow
University of Missouri-St. Louis, janikowc@umsl.edu

Sanjiv Bhatia
University of Missouri-St. Louis, sanjiv@umsl.edu

Sharlee Climer
University of Missouri-St. Louis, climers@umsl.edu

Efficient Reduced-Bias Genetic Algorithm (ERBGA) for Generic Community Detection Objectives

Aditya Karnam Gururaj Rao
University of Missouri-St. Louis
agtk4@umsl.edu
Dr. Sanjiv Bhatia
University of Missouri-St. Louis
sanjiv@umsl.edu

Dr. Cezary Janikow
University of Missouri-St. Louis
janikowc@umsl.edu
Dr. Sharlee Climer
University of Missouri-St. Louis
climers@umsl.edu

ABSTRACT

Community structure identification has been an important research area for biology, physics, information systems, and social sciences for studying properties of networks representing complex relationships. Lately, Genetic Algorithms (GAs) are being utilized for community detection. GAs are machine-learning methods that mimic natural selection. However, previous approaches suffer from some deficiencies: redundant representation and linearity assumption, that we will try to address. in. The algorithm presented here is a novel framework that addresses both of these above issues. This algorithm is also flexible as it is easily adapted to any given mathematical objective. Additionally, our approach doesn't require prior information about the number of true communities in the network. Overall, our efficient approach holds potential for sifting out communities representing complex relationships in networks of interest across different domains.

Keywords

Genetic Algorithms, Clustering, Community Detection, Modularity, Network Structure, Machine Learning, Big Data Analytics

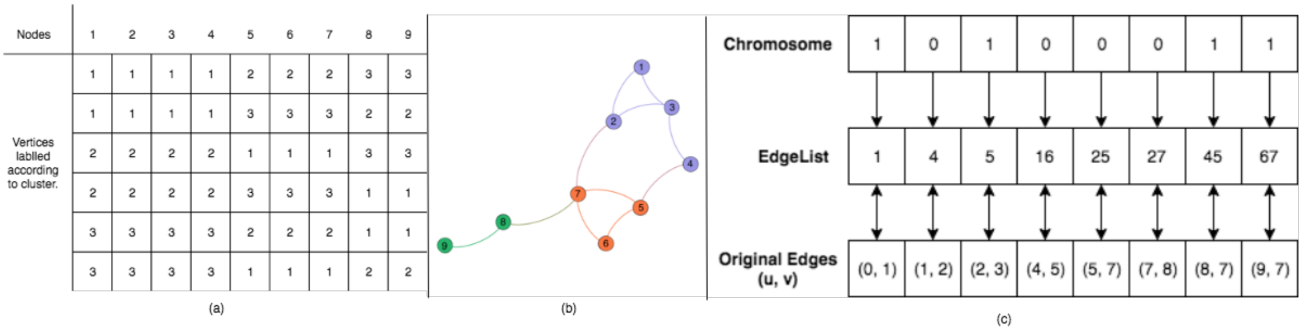
BACKGROUND

Networks are popular modeling tools for researchers in diverse fields because they can be used to represent many real world systems. For example, applications in the field of Information Systems include large social or preference-based customer association networks for e-commerce, where users with common behaviors could be identified easily to target different advertisements and provide personalized product recommendation tools (Chen et al. 2017). In the area of cybersecurity, network packets may be modeled into a graph-based model and clustering would potential help identify malicious packets; the cybersecurity strategies themselves can be analyzed using a network model (Kolini 2017).

Nodes in a network are generally connected to one another in a way that represents certain pairwise relationships of a given domain. One of the most important network properties to investigate is the *community structure*. Community structure captures collections of nodes with high densities of pairwise relationships, resulting in the formation of distinct communities, which are also referred to as *clusters*. These clusters can reveal information about the interactions of the central forces of the system being modeled and how those forces affect the physical objects represented by nodes.

The problem of accurately detecting these communities is a pressing issue to extract useful information from big data, with numerous different community detection approaches proposed. Some available software, such as DBSCAN (Ester et al. 1996), are algorithmic, with no precise objective function defined, while others aim to optimize specific objectives. Several objective functions are proposed for community detection in the literature, such as K-Means clustering (Jain 2010) and Newman-Girvan's Modularity (Newman 2006). A basic assumption of K-Means is that clusters have a relatively spherical shape, resulting in incomplete exploration of structural properties of the network and partitioning of elongated clusters. Modularity is a quantitative definition used for assessing the partitioning of a network into clusters that does not assume sphericity. Note that identification of optimal solutions for either Modularity or K-Means is NP-hard and consequently approximation algorithms are utilized.

Some of the widely used approximation algorithms include Lloyd's algorithm (Lloyd 1982) for K-Means and Genetic Algorithms (GAs) for Modularity (Newman 2006; Tasgin et al. 2007). GAs are randomized search and optimization techniques guided by the principles of evolution and genetics. The solution space is expressed in the form of *chromosomes* (strings of *genes*). A collection of chromosomes forms a *population*. Initially, a random set of solutions is generated, represented by a population of chromosomes. These chromosomes are then evaluated using a *fitness* function, which is the objective function, to assess the quality of the solutions. Then, the breeding cycle to evolve a new generation of the population is performed by using crossover and mutation operators. Traditional crossover operators use a single crossover point and all the genes after that selection point are exchanged between chromosomes. This exchange of genes results in introducing bias towards maintaining linearity of structures, which is not a property of clusters. Another problematic issue with existing algorithms is the representation of cluster information by labelling the vertices, leading to redundancy. **Figure 1** shows an example network with three clusters represented by six distinct encodings. This representation clearly is not efficient since it expands the search space by an order of *k!*



Nodes	1	2	3	4	5	6	7	8	9
Vertices lablled according to cluster.	1	1	1	1	2	2	2	3	3
	1	1	1	1	3	3	3	2	2
	2	2	2	2	1	1	1	3	3
	2	2	2	2	3	3	3	1	1
	3	3	3	3	2	2	2	1	1
	3	3	3	3	1	1	1	2	2

(a)

Chromosome	1	0	1	0	0	0	1	1
EdgeList	1	4	5	16	25	27	45	67
Original Edges (u, v)	(0, 1)	(1, 2)	(2, 3)	(4, 5)	(5, 7)	(7, 8)	(8, 7)	(9, 7)

(c)

Figure 1. Methods of representing chromosomes. (a) The table shows six distinct ways that the clustering indicated by colors in the network **(b)** could be encoded when labelling nodes with cluster numbers. In general, if there are *k* clusters, then there would be *k!* possible chromosomes representing identical clustering of the nodes. **(c)** ERBGA's mapping of an example chromosome to edges.

OVERVIEW

In this paper, we introduce an efficient and flexible GA which addresses the linearity and ambiguous labelling issues. Our approach provides four important contributions. First, our novel two-level method of representing chromosomes drastically reduces redundancies in the solution space generated by other GA approaches. Second, it is optimized for reduced memory consumption with increasing complexity of the networks, resulting in an efficient execution environment. Third, our approach is flexible and can be readily adapted for any arbitrary objective function. We demonstrate this Efficient Reduced-Bias GA (ERBGA) using Modularity. We compare our outcomes with previously published results for benchmark instances (Li & Liu 2016).

METHODS

Our algorithm is based on the optimization of a given community detection objective function using a set of distinct islands of populations that evolve over a predefined number of generations (iterations). An initial population of chromosomes is randomly created, and subsequent populations are produced, using selection across evolved chromosomes. ERBGA is generational, where we maintain two populations, one corresponding to the i^{th} and the other to the $i+1^{th}$ generation.

Network model. The network is defined by an undirected graph $G = (V, E)$ where V is a set of vertices and E is a set of edges connecting those vertices. A list of nodes adjacent to u is known as an adjacency list and is denoted by *Adj(u) = (list of end points of edges incident to vertex u) and vice-versa*.

We generate unique edge identification e_{id} using a function $\varphi(u, v) = V * u + v$, where u and v are the endpoints. Furthermore, we define a sorted list, *EdgeList* of unique edge IDs generated using φ. This list is used to map chromosomes back to the network structure. To decode e_{id} back to edge representation $E\ (u, v)$ we use an inverse function φ. $\varphi(e_{id}) = (e_{id} / V, e_{id} \% V)$.

Chromosome representation. In our approach, clustering is defined by a set of removed edges $RE = \{e_1, e_2, \ldots, e_p\}$ when removed from the network breaks the network to separate it into clusters. These separated components indicate the current clustering of the network. We define a dual layer representation of the chromosomes for representing the solution space. In contrast to the traditional approach of using cluster assignment numbers (**Figure 1a**), this representation serves us in identifying 'physically' unique individuals after breeding.

Each chromosome is a bit string with a sequence of 0's and 1's. The length of each chromosome is equal to the number of edges in the network. Let the chromosome c: $\{b_1 b_2 b_3 \ldots b_E\}$, where b_i of 0 denotes that the edge is present in the clustering scheme and 1 denotes that the corresponding edge is removed. That is to say that the removed edges are used to physically separate the clusters in the current clustering scheme. The edges are mapped from chromosomes to edges using *EdgeList*, where *φ(EdgeList$_i$)* for each bit at index i in the chromosome. **Figure 1(b)** shows an example chromosome mapped to the edges using the *EdgeList*.

Initialization. GA is initialized by randomly generating Population Size (P_{size}) bit strings of size equal to the number of edges in the network. The Random Population Rate (P_{rate}) ensures the minimum percentage of 1's in the chromosome.

Elitism. Elitism refers to moving the best E_{rate} * P_{size} individuals from previous generation to the next unaltered. This strategy guarantees that the solution quality doesn't decrease from one generation to the next.

Selection. Selection is used to select chromosomes that participate in the crossover and mutation breeding phases. We use a Tournament-based selection operator, where a predefined number of chromosomes are arbitrarily selected and put into the Tournament Pool (T_{Pool}). The two chromosomes with highest fitness move to Crossover phase.

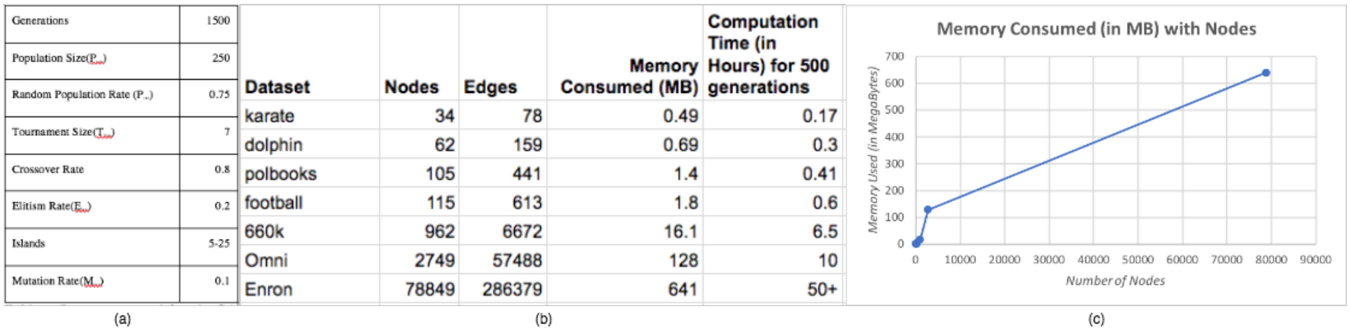
Non-Linear Crossover. Because nodes in clusters are not linearly ordered, for each pair of chromosomes derived via Tournament Selection, we arbitrarily generate a list of crossover points and single genes are exchanged only at these points. This approach breaks the linearity of traditional crossover and facilitates exploration of remote regions of the search space.

Mutation. Following crossover, we arbitrarily mutate some bits in the chromosomes, resulting in adding/removing the corresponding edge in the network. Mutation rate M_{rate} varies in the range from 0 to 1.

GA Islands. ERBGA uses islands of populations evolving independently (Whitley et al. 1999), which helps in exploring more regions of the search space as each population may follow unique trajectories into the search space. We use the best of all islands to benchmark the results.

Efficiency. We have implemented our algorithm using C++ and, as shown in the Results section, our implementation is computationally efficient. It also scales well for complex networks as we use a 3-dimensional bit array to represent chromosomes. Effective size of a $G\ (V,\ E)$ dataset with population size P_{size} is: $P_{size} \times (E/8 + \neg(E\ \%\ 8))$, where $\neg$ is the logical negation operator.

Fitness Function. ERBGA is flexible for acceptance of any community detection-based objective for evaluating the fitness of chromosomes. Also, the algorithm doesn't require prior information about the number of clusters in the network. If a situation arises when a particular number of clusters is desired, this can typically be accomplished by building the value into the objective function.



Generations	1500
Population Size(P_{size})	250
Random Population Rate (P_{rate})	0.75
Tournament Size(T_{size})	7
Crossover Rate	0.8
Elitism Rate(E_{rate})	0.2
Islands	5-25
Mutation Rate(M_{rate})	0.1

(a)

Dataset	Nodes	Edges	Memory Consumed (MB)	Computation Time (in Hours) for 500 generations
karate	34	78	0.49	0.17
dolphin	62	159	0.69	0.3
polbooks	105	441	1.4	0.41
football	115	613	1.8	0.6
660k	962	6672	16.1	6.5
Omni	2749	57488	128	10
Enron	78849	286379	641	50+

(b)

Figure 2. Parameters and results. (a) Parameters used to run the experiments using ERBGA. **(b)** Memory consumption and computation time trends as the nodes and edges scale**. (c)** Memory consumed by the program with respect to the scaling of nodes.

RESULTS

The experiments were run on an i7 2.1GHz machine running Linux with 8GB of RAM. Using the parameters shown in **Figure 2(a)**, we tested four standard benchmarking datasets, namely Zachary's Karate club (Zachary 1977), Dolphin Social Network (Lusseau et al. 2003), American College Football (Girvan & Newman 2002), and US Politics Books (Krebs 2004). We also tested two networks, 660k and Omni, that have arisen in our research of genetic markers associated with Alzheimer's Disease. Finally, in order to test the scalability of our approach, we tested a network comprised of email correspondence, namely Enron (Agarwal et al. 2012).

Accuracy. We compare our results with the fitness reported for the four benchmark instances in the paper by Zhangtao Li (Li & Liu 2016) and CC-GA (Said et al. 2018). ERBGA results in achieving 0.420 for Karate Club and 0.465 for Dolphin Social

Network in contrast to the reported best values for Karate 0.419 and Dolphin 0.526, respectively. Currently, our approach does not perform well with US Politics Books and Football, and we are in the process of analyzing the results for those datasets.

Efficiency. ERBGA computation time and memory usage are shown in **Figure 2(b)**. To visualize the amount of memory consumed by the runs we plot trend in **Figure 2(c).** Memory is efficiently allocated when nodes scale. Our implementation can run huge datasets like Enron email network, which consists of 78,849 nodes and 286,379 edges, with less than 1GB of memory.

CONCLUSION

A key issue for the use of GAs for community detection is a meaningful chromosomal representation that properly captures phenotypic characteristics in an efficient manner. Previous efforts have resulted in the search space being expanded by an order of *k!*, where *k* is the number of communities. Here we introduce a novel representation that uses the removal of edges to define each possible clustering configuration exactly once in the search space. One drawback of our current implementation is that the dense networks may have many edges removed yet remain connected, thus representing a single cluster. This behavior was observed for the US Politics Books, Football, 660k, Omni and Enron datasets. These results suggest development of a strategy to increase contextual removal of edges rather than removing them randomly. We are experimenting with methods to improve performance by considering the degree of vertices that are adjacent to a candidate edge. If the edge is incident to a vertex with high degree, the probability of selecting the edge for removal would be reduced. This strategy may help to break up large dense networks into distinct clusters. Also, computing more islands and possible migration of chromosomes between islands could be beneficial to increase accuracy. It should be noted that these trials can be run in parallel and it may be possible to compute large numbers of populations, given an adequate number of processors.

A broad issue for community detection is the selection of a meaningful objective function. The choice is dependent upon the characteristics of the particular network of interest. In some domains, sphericity is suitable, while in other domains, such as genetic associations with complex diseases, such a bias could be highly problematic. ERBGA flexibly allows any arbitrary objective, providing a convenient tool for comparing alternative functions.

Another issue for community detection using GAs is the enforcement of linearity for the chromosomes during crossover operations. ERBGA breaks up the linearity by randomly selecting genes. Also, our approach is efficient for both time and space complexity. Overall, ERBGA addresses key biases introduced by previous approaches and holds potential for future research as well as commercial applications.

REFERENCES

1. A. K. Jain (2010) Data clustering: 50 years beyond K-means, *Pattern Recognition Letters* 31.
2. A. Said, R. A. Abbasi, O. Maqbool, A. Daud, N. R. Aljohani (2018) CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks, *Applied Soft Computing*, 63, 59-70.
3. Agarwal, A., Omuya, A., Harnly, A., and Rambow, O. (2012) A Comprehensive Gold Standard for the Enron Organizational Hierarchy, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*, 2, 161–165.
4. Chen X., van der Lans R., Trusov M. (2017) Integrating Social Networks into Marketing Decision Models. In: Wierenga B., van der Lans R. (eds) Handbook of Marketing Decision Models. International Series in Operations Research & Management Science, vol 254. Springer, Cham
5. D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson. (2003) The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations, *Behav. Ecol. Sociobiol.* 54 (4) 396–405.
6. D. Whitley, S. Rana and R. B. Heckendorn. (1999) The Island Model Genetic Algorithm: On Separability, Population Size and Convergence*, Journal of Computing and Information Technology* 1, 33-47.
7. F. Kolini and L. Janczewski, "Clustering and Topic Modelling: A New Approach for Analysis of National Cyber security Strategies" (2017). *PACIS 2017 Proceedings*. 126.
8. Lloyd, S. P. (1982) Least squares quantization in PCM, *IEEE Transactions on Information Theory*, 28 (2): 129–137.
9. M. Ester, H.P. Kriegel, J. Sander, X. Xu. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining* KDD-96.

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

11. M. Tasgin, A. Herdagdelen, H. Bingol. (2007) Community detection in complex networks using genetic algorithms. *ArXiv preprint* arXiv:0711.0491.

12. Newman, M. & Girvan, M. (2004) Finding and evaluating community structure in networks, *Phys. Rev. E* 69**,** 26113.

13. Newman, M. E. J. (2006) Modularity and community structure in networks, *Proc. Natl. Acad. Sci. U. S. A.* 103**,** 8577–82.

14. V. Krebs, (2004) Books about US politics, http://www.orgnet.com.

15. W. W. Zachary. (1997) An information flow model for conflict and fission in small groups, *Journal of Anthropological Research* 33, 452-473.

16. Z. Li, J. Liu. (2016) A multi-agent genetic algorithm for community detection in complex networks, *Physica A* 449, 336–347.