1 Abstract

Our work presents a genetic algorithm (GA) for finding central vertices in a graph that uses a different approach to the method presentation of the solution and describes a new look at the crossover process. The resulting algorithm was compared with the already existing exact and other genetic algorithms on various random graph models. From the obtained results we can conclude that this approach can be used in applications and compete with existing algorithms.

2 Introduction

In graph theory, finding the radius of a graph and central vertices is one of the important problems solved in theoretical and practical problems. Consider an unweighted undirected graph $G = (V, E)$, where $|V| = n$ — the number of vertices, $|E| = m$ — the number of edges. Since the graph is unweighted, the path length between the vertices of u and v is equal to the number of edges. To find the center of the graph, use the concept vertex eccentricity — the distance from the vertex to the most distant vertex. Based on this, the radius of the graph can be described as the minimum of the eccentricities of all the vertices. At the same time, the nodes on which this minimum is reached are usually called central. This question most often arises in optimization problems in computer networks and in transport routing problems.

This task is well studied from the point of view of exact algorithms. Often, it is considered together with the problem of calculating all-pairs shortest path in graphfor unweighted graphs, we can solve this problem in a trivial way. For this you need to run breadth-first search (BFS) from each vertex,and this algorithm has a time estimate $O(nm)$ that for large m equals $O(n^3)$. All existing algorithms that give the exact solution are created, in an attempt improve this asymptotic estimate. In general, there are two approaches to solving this problem. The first one is proposed in [1] and is based on matrix multiplication. Today, there are fast matrix multiplication algorithms that give a theoretical estimate $O(n^{2.376})$ or more practiced $O(n^{2.81})$.

Another approach was developed in [2]. In this case, the idea of dividing vertices into a set of vertices with a high degree and low degree is used. Using this approach, the time estimate can be improved and the asymptotics $O(m\sqrt{n})$ can be obtained. This work gave impetus to further research using a similar method and there are a number of algorithms [3], [4], [5] that improve this estimate. These algorithms use special data structures or assume low density graphs.

These methods can be widely applied on small graphs, however, real-life graphs are of most interest for research, but these graphs contain tens of thousands of nodes.As a result, exact algorithms cannot be applied to solve similar tasks due to unacceptable time costs. The most suitable in this situation may be heuristic algorithms that give better temporal results, but allow for the existence of a percentage of error.

In our work, we present a genetic algorithm that allows to solve the problem of finding the central vertices and the radius of a graph. At the same time, in comparison with some algorithms, ours gives better temporal results.

3 Algorithm Description

Genetic algorithms are a well-known approach for solving optimization problems. The basic idea of the GA was introduced by Holland [6]. The algorithm uses genetics processes that ensure the evolutionary development of living organisms. According to the proposed approach, the solution is represented by a certain set of genes and the main stages of the algorithm are the mutation process crossing-over and natural selection. In our algorithm, all these processes are implemented taking into account the problem in question.

The basic idea of constructing an algorithm can be described as follows. We can create an abstract graph image. In this case, the central vertices should be placed in the center of the image and the remaining vertices will form discrete "spheres", located at a distance of 1, 2, etc. from the center. Using this representation of the graph, it is easy to notice that to find the center of the graph, we can create a set of vertices, which will represent the "sphere" within which the center will lie. Then each iteration of the algorithm should reduce the distance between elements of the created "sphere" and to tie it to the center. Our algorithm does this with the help of a crossover operator.

3.1 Problem Representation

The population for the genetic algorithm is described by one set of vertices, where each vertex represents a potential solution. In the language of GA, this means that each individual in the population is a vertex of the graph. A set of random vertices is generated as the initial population.

3.2 Fitness Function

The most natural way to assess the quality of the solution obtained in the framework of the problem in question is the value of eccentricity vertices that is found using a BFS. In this case, natural selection gives priority to vertices with a lower eccentricity.

3.3 Mutation

As a mutation process, we chose an approach in which to change an existing population, the vertex is replaced by a random one from the set of its neighbors with a probability of mutation.

3.4 Crossover

As a crossover operator, we used the following heuristics. Consider the current population, as mentioned earlier she can be interpreted as a "sphere" inside which lies the center. In this regard, you can consider a couple of vertices and find between them the shortest path using the BFS. After that, as a descendant from two individuals, a random vertex is selected from the found ways. This method allows for each iteration to approach a certain vertex with an optimal eccentricity.

4 Experimental results

All algorithms were performed on a computer with AMD A8-7410 2.20 GHz CPU and 6 GB RAM. All algorithms were implemented in the programming language C ++.

To determine the accuracy of the proposed algorithm, the exact algorithm [2] was implemented, which allows to draw conclusions about the correctness of the found solution. Also, to compare the time costs, another genetic algorithm was used, described in [7], which has similar approaches to solving the considered problem, but differs by the mutation, crossover operator and population representation. This algorithm uses the mutation operator, which was named by the authors N4N, therefore we will denote it in the following way.

Algorithms were compared on two random graph models. The first of these is the Barabasi-Albert model [8], second is random geometric graph [9]. For the graph BA parameter $m = 2$, and in a geometric random graph $r = 0.1$. For both algorithms, time measurements and accuracy of the parameters found were made. For a clearer picture, both algorithms were run 100 times on each test, which allowed us to obtain the average time and percentage of errors. Since our algorithm uses the idea of abstract spheres we will call it "spherical". The results of the experiments are given in table 1 and 2.

Table 1: Time of operation and percentage of algorithm errors on BA graph

	Graph size		Time, sec.		Eroor, %	
	N	M	Spherical alg.	N4N alg.	Spherical alg.	N4N alg.
1	500	996	0.07	0.29	16.0	0.0
2	1000	1996	0.18	0.68	12.0	0.0
3	1500	2996	0.37	1.24	4.0	0.0
4	2000	3996	0.55	1.67	1.0	0.0
5	2500	4996	0.69	2.18	0.0	0.0
6	5000	9996	1.84	8.28	0.0	0.0
7	10000	19996	3.90	15.8	0.0	0.0

From the obtained results it can be seen that the proposed algorithm works many times faster. In this case, the algorithm gives a significant percentage of

Table 2: Time of operation and percentage of algorithm errors on random geometrix columns

	Graph size		Time, sec.		Eroor, %	
	N	M	Spherical alg.	N4N alg.	Spherical alg.	N4N alg.
1	500	3572	0.11	0.39	38.0	40.0
2	1000	14202	0.31	0.77	21.0	50.0
3	1500	31861	0.71	1.44	13.0	62.0
4	2000	57438	1.20	1.92	8.0	48.0
5	2500	90268	1.76	2.68	4.0	30.0
6	5000	358553	4.48	8.61	0.0	0.0
7	10000	1439255	13.54	26.0	0.0	0.0

error only on graphs of relatively small size, where it is not advisable to use this approach, since the time costs of exact algorithms are insignificant. However, with increasing graph dimension, the percentage of incorrect answers tends to minimize.

5 Conclusion

We have created and implemented a genetic algorithm for solving the problem of finding the central vertex and radius of the graph. We tested the created algorithm on two random graph models. Having obtained empirical results, we can say that the proposed algorithm can be used to solve practical problems on graphs on large-dimension graphs.

References

[1] Seidel R 1995 *J. Comput. Syst. Sci.* **51** 400–403

[2] Aingworth D, Chekuri C, Indyk P and Motwani R 1999 *SIAM J. Comput.* **28** 1167–1181

[3] Berman P and Kasiviswanathan S P 2007 Faster approximation of distances in graphs *Algorithms and Data Structures* ed Dehne F, Sack J R and Zeh N (Berlin, Heidelberg: Springer Berlin Heidelberg) ISBN 978-3-540-73951-7

[4] Chan T M 2012 *ACM Trans. Algorithms* **8** 34:1–34:17 ISSN 1549-6325 URL `http://doi.acm.org/10.1145/2344422.2344424`

[5] Roditty L and Vassilevska Williams V 2013 Fast approximation algorithms for the diameter and radius of sparse graphs *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing* STOC '13 (New York, NY, USA: ACM) pp 515–524 ISBN 978-1-4503-2029-0 URL `http://doi.acm.org/10.1145/2488608.2488673`

[6] Holland J 1975 *Adaption in Natural and Artificial Systems Adaption in Natural and Artificial Systems* (University of Michigan Press)

[7] Alkhalifah Y and Wainwright R L 2004 A genetic algorithm applied to graph problems involving subsets of vertices *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)* vol 1 pp 303–308

[8] Albert R and Barabasi A L 2002 *Reviews of Modern Physics* **74** 47–97

[9] Gilbert E 1961 *Journal of the Society for Industrial and Applied Mathematics* **9** 533–543