Acta Cybernetica **25** (2021) 151–169.

Symbolic Regression for Approximating Graph Geodetic Number*

Ahmad T. Anaqreh[ab], Boglárka G.-Tóth[ac], and Tamás Vinkó[ad]

Abstract

In this work, symbolic regression with an evolutionary algorithm called Cartesian Genetic Programming, has been used to derive formulas capable to approximate the graph geodetic number, which measures the minimal-cardinality set of nodes, such that all shortest paths between its elements cover every node of the graph. Finding the exact value of the geodetic number is known to be NP-hard for general graphs. The obtained formulas are tested on random and real-world graphs. It is demonstrated how various graph properties as training data can lead to diverse formulas with different accuracy. It is also investigated which training data are really related to each property.

Keywords: symbolic regression, cartesian genetic programming, geodetic number

1 Introduction

Geodetic number is the minimal-cardinality set of nodes, such that all shortest paths between its elements cover every node of the graph [16]. Calculating the geodetic number proved to be an NP-hard problem for general graphs [5]. The integer linear programming (ILP) formulation of geodetic number problem was given in [16], containing also the first computational experiments on a set of random graphs.

The trivial upper bound for the geodetic number is $g(G) \leq n$. Chartrand *et al.* [10] proved that $g(G) \leq n - d + 1$, where d is the diameter of G. Other

*The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002), by grant NKFIH-1279-2/2020 of the Ministry for Innovation and Technology, Hungary and by the grant SNN-135643 of the National Research, Development and Innovation Office, Hungary.

[a]Department of Computational Optimization, Institute of Informatics, University of Szeged, Hungary

[b]E-mail: ahmad@inf.u-szeged.hu, ORCID: 0000-0002-3971-2684

[c]E-mail: boglarka@inf.u-szeged.hu, ORCID: 0000-0002-0927-111X

[d]E-mail: tvinko@inf.u-szeged.hu, ORCID: 0000-0002-3724-4725

DOI: 10.14232/actacyb.289041

upper bounds are also given in [6, 30, 31], but these are concerning specific graph structures.

Chakraborty *et al.* [9] proposed an algorithm to approximate the geodetic number on edge color multigraph. A polynomial algorithm to compute the geodetic number of interval graphs has been proposed in [12]. Greedy-type algorithms are developed in [3] to find upper bound of the geodetic number on general graphs based on shortest paths information.

There are varied applications of geodetic sets and geodetic number. Clearly, they can be applied in computational sociology as it is hinted in [7, 31]. The definition of convexity of set of nodes in a graph [18] is a somewhat converse property to geodetic set. Related notions are the graph hull number [14] and the domination number [15]. All these concepts have practical applications, e.g., in public transportation design [9], in achievement and avoidance games [8], in location problems [25], in maximizing the switchboard numbers on telephone tree graphs [23], in mobile ad hoc networks [26], and in design of efficient typologies for parallel computing [24].

Graph properties are certain attributes that could make the structure of the graph understandable. Occasionally, standard methods to calculate exact values of graph properties cannot work properly due to their huge computational complexity, especially for real-world graphs. In contrast, heuristics and metaheuristics are alternatives which have proved their ability to provide sufficient solutions in a reasonable time. However, in some cases even heuristics fail to succeed, particularly when they need some less easily obtainable global information of the graph. The problem thus should be dealt with in a completely different way by trying to find features that are related to the property, and based on these data build a formula which can approximate the graph property.

Topological representation is the simplest way to represent graphs, where the graph is a set of nodes and edges. However, the spectral representation (e.g., adjacency matrix, Laplacian matrix) can significantly help to describe the structural and functional behavior of the graph. Adjacency matrix is a square matrix in which a non-zero element indicates that the corresponding nodes are adjacent. Implementations of well known algorithms like Dijkstra's or Floyd–Warshall algorithm usually use the adjacency matrix to calculate the shortest paths for a given graph. The diameter of a graph is the length of its longest shortest path. It is known that the diameter of a given graph is small if the absolute value of the second eigenvalue of its adjacency matrix is small [11]. Laplacian matrix is a square matrix which can be used to calculate, e.g., the number of spanning trees for a given graph. The eigenvalues of the Laplacian matrix are non-negative, less than or equal to the number of nodes, and less than or equal to twice the maximum node degree [4]. Considering these important relations between the graph properties, eigenvalues of spectral matrices and more parameters (to be discussed in the forthcoming sections), which can be calculated easily even for complex graphs, symbolic regression is one of the good choices to verify the connection between graph parameters and properties, and use such parameters for approximating hard to compute network

properties.

Symbolic regression (SR) is a mathematical model which attempts to find a simple formula such that it fits a given output in term of accuracy based on a set of inputs. In conventional regression techniques, a pre-specified model is proposed, while symbolic regression avoids a particular model as a starting point to give a formula. Instead, in SR, initial formulas are formed randomly by combining the inputs: parameters, operators, and constants. Then, new formulas are assembled by recombining previous formulas by using one of the evolutionary algorithms, which is the genetic programming in our work. Symbolic regression practically has infinite search space, hence infinite formulas to assemble. Nevertheless, this can be considered as an advantage when symbolic regression uses an evolutionary algorithm called genetic programming, which requires diversity to efficiently explore the search space, ensuring a highly accurate formula.

The inputs are predefined parameters and constants. SR combines these parameters and constants by a set of given arithmetic operators (such as $+, -, \times, \div$, etc.) to assemble a formula. In the papers by Schmidt and Lipson, symbolic regression was used to find physical laws based on experimental data [28], and then they used it to find analytical solutions to iterated functions of an arbitrary form [29]. Even though there are some algorithms in the literature that use symbolic regression apart from genetic programming [21], essentially genetic programming is considered as one of the most popular algorithms applied by symbolic regression [19].

The rest of the paper is structured as follows. Section 2 discusses the specific genetic programming approach we used together with the list of graph properties. Section 3 discusses the methodology used to approximate the graph geodetic number. Section 5 reports the numerical results to show the efficiency of the formulas we obtained. The conclusion of our work is presented in Section 6. In the Appendix, we report all the formulas we obtained during this work.

2 Preliminaries

2.1 Cartesian Genetic Programming

One of the most famous genetic programming tools is called Cartesian Genetic Programming (CGP) developed by Miller [22]. CGP is an iteration-based evolutionary algorithm and works as it follows. CGP begins by creating a set of initial solutions, from which the best solution is chosen by evaluating these solutions based on the fitness function. Then these solutions will be used to create the next generation in the algorithm. The next generation's solutions will be a mixture of chosen solutions from the previous generation's, where the new generation's solutions should not be identical to the previous ones', which can be done by mutation. Mutation is used to change small parts of the new solutions and it usually occurs probabilistically for CGP. The mutation rate is the probability of applying the mutation on a specific new solution. Eventually, the algorithm must terminate. There are two cases in

which this occurs: the algorithm has reached the maximum number of generations, or the algorithm has reached the target fitness. At this point, a final solution is selected and returned.

Cartesian Genetic Programming has several parameters to set up, which certainly have effects on its performance. The specific parameters used in this paper are detailed later in Section 4.3.

2.2 Geodetic Number

A simple connected graph is denoted by $G = (V, E)$, where V is the set of nodes and E is the set of edges.We have $N = |V|$ and $M = |E|$. Geodetic number is the minimal-cardinality set of nodes, such that all shortest paths between its elements cover every node of the graph [16]. The formal description is as follows. Given $i, j \in V$, the set $I[i, j]$ contains all $k \in V$ which lies on any shortest paths between i and j. The union of all $I[i, j]$ for all $i, j \in S \subseteq V$ is denoted by $I[S]$, which is called as *geodetic closure* of $S \subseteq V$. Formally,

$$I[S] := \{x \in V : \exists i, j \in S, x \in I[i, j]\}.$$

The *geodetic set* is a set S for which $V = I[S]$. The *geodetic number* of G is

$$g(G) := \min\{|S| : S \subseteq V \text{ and } I[S] = V\}.$$

2.3 Graph Properties

Adjacency matrix. The adjacency matrix is a square $|N| \times |N|$ matrix A such that its element A_{ij} is equal to one when there is an edge from node i to node j, and zero when there is no edge.

Shortest path. The series of nodes $u = u_0, u_1, \ldots, u_k = v$, where u_i is adjacent to u_{i+1}, is called a *walk* between the nodes u and v. If $u_i \neq u_j$ $(\forall i, j)$, then it is called a *path*. The path's length is k. Given all paths between nodes u and v, a *shortest path* is a path with the fewest edges. Shortest paths are usually not unique between two nodes.

Diameter. Graph diameter is the length of the longest path among all the shortest paths in the graph.

Degree, degree-one node. The *degree* of a node is the number of edges linking the node to other nodes in the graph, denoted by $\deg(v)$. If $\deg(v) = 1$, which means there is only one edge connecting the node, this node is called a *degree-one node*. It is known from the literature that degree-one nodes are always part of the geodetic set, see [17]. The number of degree-one nodes in the graph is denoted by δ_1.

Laplacian matrix. The Laplacian matrix is a square $|N| \times |N|$ matrix L such that $L = D - A$, where A is the adjacency matrix and D is the degree matrix,

i.e., the elements in its main diagonal are defined as $D_{ii} = \deg(v_i)$, where $v_i \in V$ $(i = 1, \ldots, N)$.

Simplicial node. node v is called a *simplicial node* if its neighbors form a clique (complete graph), namely, every two neighbors are adjacent. If G is a non-trivial connected graph and v is a simplicial node of G, then v belongs to every geodetic set of G, see [1]. The number of simplicial nodes in the graph is denoted by σ.

Betweenness centrality. Betweenness centrality (BC) for a specific node v is the proportion of all the shortest paths pass through this node. It is shown in [17] that if G is a star graph with n nodes then $g(G) = n - 1$, where the central node with the highest BC, that all the shortest paths passing through, will never be in the geodetic set. Moreover, in the tree graph G with k leaves $g(G) = k$, that means the leaves with low BC are geodetic nodes while the root and the parents with higher BC are not part of the geodetic set.

3 Methodology

Although there are not many papers proposing the idea of using symbolic regression for approximating graph properties, the work by Martens *et al.* [20] was a good starting point for us. They used the eigenvalues of the Laplacian matrix and of the adjacency matrix as inputs for CGP, the experiments made on real-world networks to optimize the diameter and isoperimetric number. In our case, we aim at obtaining results for the geodetic number on random and real-world graphs. Thus, we investigated graph properties that are strongly related to the geodetic number, which have been discussed in Section 2.3.

We have used CGP-Library which is a cross-platform Cartesian Genetic Programming implementation developed by Andrew Turner[1]. The library is written in C and it is compatible with Linux, Windows and MacOS.

In order to use CGP a set of training data is needed. Each training data will contain instances and each instance contains two parts: (i) parameters of graph properties and chosen constants as inputs, (ii) exact value of the graph property as output. Thus, CGP will attempt to join the parameters and constants by using arithmetic operators to achieve the output. The set of arithmetic operators we have used in all the cases is $\{+, -, \times, \div, \sqrt{x}, x^2, x^3\}$. For the graph properties we have used the ones discussed in Section 2.3: eigenvalues of the adjacency matrix and Laplacian matrix, number of degree-one nodes, number of simplicial nodes, etc. It will be shown that these parameters strongly related to the graph geodetic number so they can be beneficial inputs for CGP. The classification of these inputs into categories will be shown in the following section which reports the results of our numerical experiments.

[1] `http://www.cgplibrary.co.uk/`

4 Parameters of the Numerical Experiments

The main goal of our experiments was to investigate the graph geodetic number for random graphs and real-world graphs. Since the most related paper to our work of Märtens *et al.* [20] contains results for the graph diameter (which is, similarly to the geodetic number, also based on shortest paths) we report our results obtained for the diameter and compare these values. The metrics used to measure the goodness of a formula are mean absolute error and mean relative error.

In the following subsection we describe the graphs used for the training as well as for the validation.

4.1 Random Graphs

Set of 120 random graphs created by using the three well-know generative models: Erdős-Rényi [13], Watts-Strogatz [32], and Barabási-Albert [2]. Regarding the number of nodes and edges the following approach were used:

- the number of nodes were $n = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100$, and
- for the number of edges we followed the scheme as in [16]:
  - for each case one can have maximum $n \cdot (n-1)/2$ edges,
  - and we took 20%, 40%, 60% and 80% of this maximum number of edges.

4.2 Real-World Graphs

As a set of real-world graphs we used 10 graphs from the Network Repository[2] [27]. For the training part, 120 connected sub-graphs of these networks with different sizes ($14 \leq N \leq 140$) were created from this set by using the following simple procedure. For a given real-world graph $G(V, E)$, first, a random set $W \subset V$ of nodes were selected. Then, the induced sub-graph of G with node set W is taken. This sub-graph $\hat{G}$ might not be connected, so, as a final step, the largest connected component of $\hat{G}$ is selected.

4.3 CGP Parameters

CGP needs predefined parameters to work properly. Table 1 summarizes the values of the parameters we have used in the experiments. The details of the parameters used are the following.

Evolutionary Strategy The evolutionary strategy uses selection and mutation as search operators. The usual version used by CGP is the one which we also apply in this paper, which is called $(1+4)$-ES. Here, the procedure selects the fittest individual as the parent for the next generation, from the combination of the current parent and the four children.

[2] `http://networkrepository.com/`

Table 1: Parameters of CGP

Parameter	**Value**
Evolutionary Strategy	$(1+4)$-ES
Node Arity	2
Mutation Type	Probabilistic
Mutation Rate	0.05
Fitness Function	Supervised Learning
Target Fitness	0.1
Selection Scheme	Select Fittest
Reproduction scheme	Mutation Random Parent
Number of generations	$200,000$
Update frequency	100
Threads	1
Function Set	`add sub mul div sqrt sq cube`

Node Arity Each node is assumed to take as many inputs as the maximum node arity value, namely, the maximum number of inputs connected to a specific node.

Mutation Type The mutation, as basic search operator of the evolutionary strategy, is performed by adding a random vector to the current solution. In our paper this is done probabilistically.

Mutation Rate The probability of applying mutation on a specific solution.

Fitness Function The supervised learning fitness function applies to each solution and assigns a fitness value to how closely the solution output match the desired output. Based on that, the solutions with better fitness value will be chosen for next generations.

Target Fitness The fitness function used in this work is the absolute differences (absolute error) between the generated and predefined outputs, where the best solution is the one with absolute difference less than or equal to the given value.

Selection Scheme The applied fittest selection schemes select the best solutions based on the closest fitness obtained by the solution.

Reproduction scheme There are two ways in which new children can be created from their parents. In the first method the child is simply a mutated copy of the parent. In the second method the child is a combination from both parents with or without mutation. This latter method is referred to recombination. Usually, CGP-Library uses the random parent reproduction scheme which simply creates each child as a mutated version of its parents.

Number of generations How many iterations CGP will apply before termination, unless one of the solutions obtained the target fitness.

Update frequency The frequency at which the user is updated on progress, where the progress details shown on the terminal.

Threads The number of threads the CGP library will use internally.

Function Set the arithmetic operators used by CGP to combine the inputs.

4.4 Training data parameters

The list of parameters used as input in the training data, separated into different sets as follows.
For random graphs:

1) $N, M, \lambda_N, \lambda_i$ $(i = 1, 2, 3)$
2) N, M, μ_{N-1}, μ_i $(i = 1, 2, 3)$
3) $N, M, \lambda_i, \lambda_{N-i-1}$ $(i = 1, \ldots, 5)$
4) N, M, μ_i, μ_{N-i-1} $(i = 1, \ldots, 5)$
5) $N, M, \lambda_i, \lambda_{N-i-1}$ $(i = 1, \ldots, 5)$ and constants $1, 2, 3, 4, 5$
6) N, M, μ_i, μ_{N-i-1} $(i = 1, \ldots, 5)$ and constants $1, 2, 3, 4, 5$

where N is the number of nodes, M is number of edges, λ_i is the i-th eigenvalue of adjacency matrix, μ_i is the i-th eigenvalue of Laplacian matrix.

For real-world graphs:

1) $N, M, \delta_1, \sigma,$ and constants $1, 2, 3, 4, 5$
2) $N, M, \delta_1, \sigma, \lambda_i, \lambda_{N-i-1}$ $(i = 1, \ldots, 5)$
3) $N, M, \delta_1, \sigma, \mu_i, \mu_{N-i-1}$ $(i = 1, \ldots, 5)$
4) $N, M, \delta_1, \sigma, \lambda_i, \lambda_{N-i-1}$ $(i = 1, \ldots, 5)$ and constants $1, 2, 3, 4, 5$
5) $N, M, \delta_1, \sigma, \mu_i, \mu_{N-i-1}$ $(i = 1, \ldots, 5)$ and constants $1, 2, 3, 4, 5$

where δ_1 is the number of nodes with degree one in the graph, σ is the number of simplicial nodes in the graph.

Note that in Section 2.3 the betweenness centrality was also discussed as shortest path based graph centrality measure, which has relation to the geodetic number. In the conducted experiments we were trying to involve the betweenness values of the nodes by putting them into categories. However, none of the best approximating formulas we have obtained by the symbolic regression included this information.

5 Results

To obtain the formulas for either random graphs or real-world graphs, we have run the CGP dozen times for each different category (see Section 4.4 for the list of these categories). Amongst all the formulas we choose the best ones according to its output's absolute error and relative error compared to the exact value. Hence, the best formulas gave the smallest error. The full list of the chosen formulas are given in the Appendix. In the following we report and discuss the top formulas for each case.

Both the diameter and the geodetic number are of course integers. However, the obtained formulas by the symbolic regression usually result in non-integer number. Hence, in the tables which report the results, first we rounded the values given by the formulas and then the errors were calculated.

Consequently, the results are reported in two types of tables. For the random graphs, only the summary of the approximation errors are shown. Regarding the real-world graphs, the full details are given, i.e., the calculated values of the diameter as well as the geodetic number using the best formulas are presented.

5.1 Diameter

5.1.1 Random graphs

Table 2 summarizes the results obtained for the different random graphs, where (6) gives the best approximation:

$$\frac{N + \lambda_{N-2} + 4}{\sqrt{M}}.$$

For the investigated set of random graphs, λ_{N-2} is in the range $[-7, -1]$, which is, on average,cancelled out by the constant factor $+4$ in formula (6). Moreover, for these graphs we have $M = O(N)$, which means that formula (6) is roughly $O(\sqrt{N})$. The square root function, at least in the range where our experiments were done, is close to the logarithm function. It is well know that the diameter of (random) graphs can be estimated by $\log(N)$. The symbolic regression did not use the log

Table 2: Diameter validations on random graphs

	formula	(2)	(3)	(4)	(5)	(6)	(7)
ER	mean absolute error	1	1.5	0.6	6.05	**0.4**	0.9
	mean relative error	0.4	0.53	0.19	2.46	**0.1**	0.33
BA	mean absolute error	1.3	0.8	0.35	4.2	**0.1**	0.55
	mean relative error	0.52	0.28	0.14	1.86	**0.03**	0.2
WS	mean absolute error	1.7	1.7	0.5	6.15	**0.35**	1.15
	mean relative error	0.57	0.57	0.18	2.48	**0.1**	0.4

function, as it was not in its function set, see Table 1. Nevertheless, it is interesting to see that it found formula (6), which is close to the logarithm of the number of nodes in the graphs.

5.1.2 Real-world graphs

For the diameter of real-world graphs, as it is shown in Table 3, the formula (15) was the best by giving very close values to the exact diameter:

$$\frac{2M}{\lambda_1\lambda_2^2} + \frac{\lambda_5^2 + 2(\lambda_N - \lambda_3) + 50}{\lambda_1} + \frac{2}{\lambda_1\lambda_2}$$

Closer inspection reveals that the last term in the formula usually has very small values, below 0.1. The other parts of (15) contribute by roughly equal quantity to the final result. The formula includes the first three, the fifth and the last eigenvalue of the adjacency matrix, as well as the number of edges. Thus, it is a nice demonstration of the surprising power of symbolic regression that it can find non-trivial combination of graph features which can well approximate a graph measure such as diameter. On the other hand, the computational cost is $O(N^3)$ due to the need of calculating the eigenvalues. This means that it has the same cost as directly applying an exact algorithm such as Floyd-Warshall to obtain the diameter.

Table 3: Diameter validations on real-world graphs

network	N	M	D	[20]	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)
`ca-netscience`	379	914	17	21	13	9	14	19	4	17	12	10
`bio-celegans`	453	2025	7	7	5	4	8	12	3	8	6	4
`rt-twitter-copen`	761	1029	14	16	13	14	14	19	12	17	20	11
`soc-wiki-vote`	889	2914	13	10	11	8	11	15	7	11	12	6
`ia-email-univ`	1133	5451	8	6	9	9	7	12	9	8	13	10
`ia-fb-messages`	1266	6451	9	7	10	8	8	12	7	9	11	6
`bio-yeast`	1458	1948	19	19	14	28	15	20	18	18	39	18
`socfb-nips-ego`	2888	2981	9	52	14	14	16	23	3	20	21	7
`web-edu`	3031	6474	11	36	14	11	15	22	13	19	16	8
`inf-power`	4941	6594	46	98	14	38	17	24	71	20	53	48
mean absolute error:				13.3	5.6	4	5.2	6.9	6.2	5.2	6.4	**3.3**
mean relative error:				0.92	0.28	0.27	0.27	0.53	0.37	0.31	0.48	**0.27**

As we can see, formulas (9) and (10) resulted the same mean relative error than (15), however, they were worse by the mean absolute error. Formula (10) involves some of the eigenvalues of the Laplacian matrix, and some constants. Formula (9) uses some of the eigenvalues of the adjacency matrix, number of nodes and it also uses the number of simplicial nodes. Thus, these formulas, although not giving as precise approximations as (15), are built up by some other graph parameters compared to (15).

Note that in the 5th column of Table 3 we included the results reported in [20] for the same set of graphs. Clearly, all the formulas we found gave smaller errors than the best solution from [20].

5.2 Geodetic number

In order to compare the approximations given by the formulas found by symbolic regression, the computation of the exact geodetic number of the input graphs were needed. For that, we used the integer linear programming formulation proposed in [16].

5.2.1 Random graphs

The results for the geodetic number of random graphs can be seen in Table 4. Formula (16) gave the best approximations for the ER and WS graphs:

$$\sqrt{\frac{N^{3/2}}{\lambda_1}} - \frac{\lambda_{N-4}N^{3/2}}{\lambda_1^2 + N^{3/2}}.$$

In case of BA graphs formula (17) resulted in the lowest error:

$$\frac{\mu_4^2}{\mu_2\mu_{N-3}} + \sqrt{N - \mu_3}$$

Practically, both formulas need the computation of all eigenvalues, thus their computational cost is $O(N^3)$. The exact computation of the geodetic number is NP-hard, whereas formula (16) and (17) can be evaluated in polynomial time.

Note that overall, formula (16) gives the best approximation for all three types of random graphs. Investigating the values one obtains by evaluating formula (16) on random graphs, it turns out that the second part is roughly half of the first part. Thus, a simpler formula would be

$$\frac{3}{2}\sqrt{\frac{N^{3/2}}{\lambda_1}}.$$

Table 4: Geodetic number validations on random graphs

	formula	(16)	(17)	(18)	(19)
ER	mean absolute error	**0.92**	1.31	1	1.07
	mean relative error	**0.1**	0.16	0.16	0.13
BA	mean absolute error	2.15	**1**	1.775	2.92
	mean relative error	0.18	**0.08**	0.17	0.26
WS	mean absolute error	**0.54**	1.38	0.92	0.69
	mean relative error	**0.04**	0.19	0.12	0.08

On average, this gives a bit more pessimistic approximation (namely, mean average error = 1.89, and mean relative error = 0.1). However, it needs the computation of the first dominant eigenvalue only, which costs $O(N^2)$.

5.2.2 Real-world graphs

Table 5 shows the results for the real-world graphs. It is important to emphasize here that since the real-world graphs in Table 5 have hundreds of nodes and thousands of edges, the calculation of the exact geodetic number, using the integer linear programming formulation proposed in [16], requires enormous computational time. For the three largest graphs (`socfb-nips-ego`, `web-edu` and `inf-power`) we were unable to compute the exact geodetic numbers due to time constraints, so they are left out from the comparison.

In this case the best approximation was obtained by the surprisingly compact formula (27):

$$\delta_1 + \sigma + \sqrt{M} - 2.$$

The number of degree-one nodes and the number of simplicial nodes appear in formula (27) because these nodes must be part of the geodetic set, as it was already mentioned in Section 2.3. In fact, these two factors appear in all the best formulas we have found, see Appendix. In the `ca-netscience` collaboration network and in the `bio-celegans` there are lots of simplicial nodes and not many degree-one nodes. For the other graphs it is just the other way around, i.e., the number of simplicial nodes is not more than 10. The remaining part of the geodetic number is approximated by $\sqrt{M} - 2$, which contributes to the approximation on these graphs 1/3 at most. The computational cost of formula (27) is $O(NM)$.

Table 5: Geodetic number validation on real-world graphs.

network	N	M	$g(G)$	(20)	(21)	(22)	(23)	(24)	(25)	(26)	(27)
`ca-netscience`	379	914	253	208	151	190	198	194	206	195	200
`bio-celegans`	453	2025	172	213	115	119	195	188	225	203	146
`rt-twitter-copen`	761	1029	459	436	437	438	439	428	446	442	444
`soc-wiki-vote`	889	2914	275	247	212	220	222	231	247	259	245
`ia-email-univ`	1133	5451	244	225	182	194	181	192	208	196	233
`ia-fb-messages`	1266	6451	318	266	254	264	276	280	296	313	311
`bio-yeast`	1458	1948	784	763	761	766	761	751	775	762	773
	mean absolute error:			32.7	56.1	44.9	39.9	39.0	29.7	28.1	**21.9**
	mean relative error:			0.12	0.21	0.17	0.14	0.13	0.12	0.11	**0.08**

5.2.3 Improvement

We have listed the best formulas and we verified them with specific random and real-world graphs. Our aim is to derive a general formula for the geodetic number that can give good approximation for any real-world graph. For that we wanted to

try and make formula (27) even sharper. One of the possible ways is to use linear regression.

For linear regression the generalized formula containing multipliers as variables has the form

$$a \cdot \delta_1 + b \cdot \sigma + c \cdot \sqrt{M} - d$$

The variables were initialized as $a = 1, b = 1, c = 1, d = 1$. The linear regression finds the values of the variables a, b, c and d minimizing the mean absolute error of the approximated value.

As a result, linear regression found that $a = 0.99, b = 0.79, c = 0.97, d = 0.99$, so the formula can be written as

$$0.99 \cdot \delta_1 + 0.79 \cdot \sigma + 0.97 \cdot \sqrt{M} - 0.99. \tag{1}$$

5.2.4 Validation of improved formula

For validating the quality of the formula (1), 120 sub-graphs (where $31 \leq N \leq 485$) from real-world graphs in Table 5 have been used. These graphs were created by the same procedure described in Section 4.2. Then the geodetic number was calculated twice: the exact value by using the ILP formulation [16], and then the approximation using the formula (1) obtained by linear regression. Figure 1 shows a comparison between the two values for the sub-graphs. It is clear that the approximations are close to the exact $g(G)$ values. For all the 120 graphs we obtained mean absolute error = 12.27 and mean relative error = 0.18 by using formula (1). This is just a slight improvement though, since formula (27) gives mean absolute error = 12.37 and the same relative error as (1).

There are two gaps in the figure indicating that for some graphs the approximation is much less than the exact value. For these graphs, the number of simplicial nodes was zero. Since formula (1) is the summation of the number of simplicial

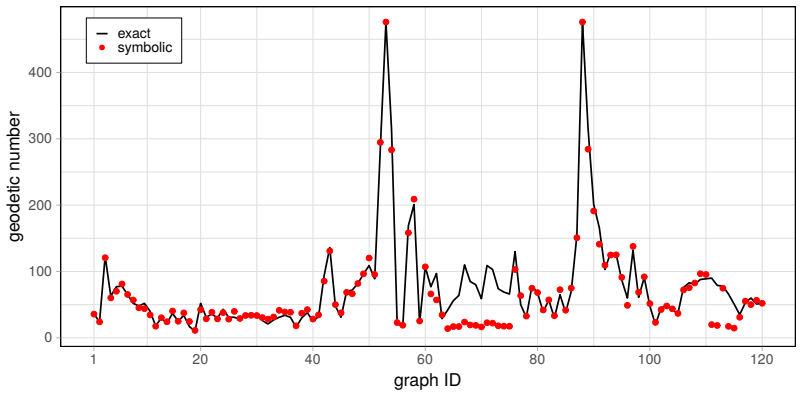

Figure 1: Exact $g(G)$ and values given by the optimized formula (1)

nodes, the number of degree-one nodes, and the number of edges, if one of these values is zero that will cause these gaps. For this type of graphs, where σ and δ_1 are close to zero, it might be more beneficial to use one of the formulas we found for the random graphs. For example, using formula (16) we get mean absolute error = 39.87 and mean relative error = 0.57 for these graphs, while formula (1) on the same graphs gives mean absolute error = 40.87 and mean relative error = 0.6.

6 Conclusion

Our work reports that symbolic regression is successfully applicable to derive optimized formulas for graph geodetic number $g(G)$. The best formula we found is very simple and it can estimate the value of $g(G)$ if the number of edges, the number of degree-one nodes and the number of simplicial nodes are known. Thus, the approximation of the geodetic number can be obtained in a reasonable computational time, even for graphs with thousands of nodes and edges, while obtaining the exact geodetic number is an NP-hard problem. We demonstrated how different training sets will lead to different formulas with different accuracy so that we can claim that finding good training data is essential. Hence, finding the best parameters of training graphs, where these parameters are highly related to the graph property are the most important part for symbolic regression to approximate in a better manner.

Acknowledgements

We thank our reviewers for their critical comments and valuable suggestions that highlighted important details and helped us in improving our paper.

References

[1] Ahangar, H A, Fujie-Okamoto, F, and Samodivkin, V. On the forcing connected geodetic number and the connected geodetic number of a graph. *Ars Combinatoria*, 126:323–335, 2016.

[2] Albert, R and Barabási, A-L. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002. DOI: 10.1103/RevModPhys.74.47.

[3] Anaqreh, A T, G.-Tóth, B, and Vinkó, T. Algorithmic upper bounds for graph geodetic number. *Central European Journal of Operations Research*, 2021. DOI: 10.1007/s10100-021-00760-7.

[4] Anderson, W and Morley, T. Eigenvalues of the Laplacian of a graph. *Linear and Multilinear Algebra*, 18(2):141–145, 1985. DOI: 10.1080/03081088508817681.

[5] Atici, M. Computational complexity of geodetic set. *International Journal of Computer Mathematics*, 79(5):587–591, 2002. DOI: 10.1080/00207160210954.

[6] Brešar, B, Klavžar, S, and Horvat, A T. on the geodetic number and related metric sets in cartesian product graphs. *Discrete Mathematics*, 308(23):5555–5561, 2008. DOI: 10.1016/j.disc.2007.10.007.

[7] Brešar, Boštjan, Kovše, Matjaž, and Tepeh, Aleksandra. Geodetic sets in graphs. In *Structural Analysis of Complex Networks*, pages 197–218. Springer, 2011. DOI: 10.1007/978-0-8176-4789-6_8.

[8] Buckley, F and Harary, F. Geodetic games for graphs. *Quaestiones Mathematicae*, 8(4):321–334, 1985. DOI: 10.1080/16073606.1985.9631921.

[9] Chakraborty, D, Foucaud, F, Gahlawat, H, Ghosh, S K, and Roy, B. Hardness and approximation for the geodetic set problem in some graph classes. *Conference on Algorithms and Discrete Applied Mathematics*, 12016:102–115, 2020. DOI: 10.1007/978-3-030-39219-2_9.

[10] Chartrand, G, Harary, F, and Zhang, P. On the geodetic number of a graph. *Networks: An International Journal*, 39(1):1–6, 2002. DOI: 10.1002/net.10007.

[11] Chung, F. Diameters and eigenvalues. *Journal of The American Mathematical Society*, 2(2):187–196, 1989. DOI: 10.1090/S0894-0347-1989-0965008-X.

[12] Ekim, T, Erey, A, Heggernes, P, van 't Hof, P, and Meister, D. Computing minimum geodetic sets of proper interval graphs. *LATIN 2012: Theoretical Informatics*, 7256:279–290, 2012. DOI: 10.1007/978-3-642-29344-3_24.

[13] Erdős, P and Rényi, A. On random graphs I. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[14] Everett, Martin G and Seidman, Stephen B. The hull number of a graph. *Discrete Mathematics*, 57(3):217–223, 1985. DOI: 10.1016/0012-365X(85)90174-8.

[15] Hansberg, A. and Volkmann, L. On the geodetic and geodetic domination numbers of a graph. *Discrete Mathematics*, 310(15):2140–2146, 2010. DOI: 10.1016/j.disc.2010.04.013.

[16] Hansen, P and van Omme, N. On pitfalls in computing the geodetic number of a graph. *Optimization Letters*, 1(3):299–307, 2007. DOI: 10.1007/s11590-006-0032-3.

[17] Harary, F, Loukakis, E, and Tsouros, C. The geodetic number of a graph. *Mathematical and Computer Modeling*, 17(11):89–95, 1993. DOI: 10.1016/0895-7177(93)90259-2.

[18] Harary, F and Nieminen, J. Convexity in graphs. *J. Differential Geom*, 16(2):185–190, 1981. DOI: 10.4310/jdg/1214436096.

[19] Koza, J R. Genetic programming: On the programming of computers by means of natural selection. *MIT Press, Cambridge, USA*, 1992.

[20] Märtens, M, Kuipers, F, and Van Mieghem, P. Symbolic regression on network properties. *Genetic Programming*, pages 131–146, 2017. DOI: 10.1007/978-3-319-55696-3_9.

[21] McConaghy, T. Ffx: Fast, scalable, deterministic symbolic regression technology. *Genetic Programming Theory and Practice IX*, pages 235–260, 2011. DOI: 10.1007/978-1-4614-1770-5_13.

[22] Miller, J. Cartesian genetic programming. *Springer*, 2011. DOI: 10.1007/978-3-642-17310-3.

[23] Mitchell, S L. Another characterization of the centroid of a tree. *Discrete Mathematics*, 24(3):277–280, 1978. DOI: 10.1016/0012-365X(78)90098-5.

[24] Peper, Ferdinand. *Efficient network topologies for extensible massively parallel computers*. PhD thesis, TU Delft, 1989. http://resolver.tudelft.nl/uuid:2e1e7c7e-b8b6-4883-bfcf-2b00d7aa5db8.

[25] Prakash, Veeraraghavan. An efficient g-centroid location algorithm for cographs. *International Journal of Mathematics and Mathematical Sciences*, 2005(9):1405–1413, 2005. DOI: 10.1155/IJMMS.2005.1405.

[26] Prakash, Veeraraghavan. Application of g-convexity in mobile ad hoc networks. In *6th International Conference on Information Technology in Asia*, pages 33–38, 2009.

[27] Rossi, R A and Ahmed, N K. An interactive data repository with visual analytics. *SIGKDD Explor*, 17(2):37–41, 2016. DOI: 10.1145/2897350.2897355.

[28] Schmidt, M and Lipson, H. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009. DOI: 10.1126/science.1165893.

[29] Schmidt, M and Lipson, H. Solving iterated functions using genetic programming. *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation*, pages 2149–2154, 2009. DOI: 10.1145/1570256.1570292.

[30] Soloff, J A, Márquez, R A, and Friedler, L M. Products of geodesic graphs and the geodetic number of product. *Discussiones Mathematicae Graph Theory*, 35(1):35–42, 2015. DOI: 10.7151/dmgt.1774.

[31] Wang, F H, Wang, Y L, and Chang, J M. The lower and upper forcing geodetic numbers of block–cactus graphs. *European Journal of Operational Research*, 175(1):238–245, 2006. DOI: 10.1016/j.ejor.2005.04.026.

[32] Watts, D J and Strogatz, S H. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998. DOI: 10.1038/30918.

Appendix

The best formulas, with respect to mean absolute deviation, found by CGP are listed for the different graph types and graph properties.

Formulas for random graphs diameter. The results obtained with these formulas are shown in Table 2.

$$\sqrt{\frac{N}{\lambda_1}} + \sqrt{\frac{2N}{\lambda_1} - \lambda_3} + 1 \tag{2}$$

$$\frac{N(M + \mu_{N-2})^3(2N^3 + (M + \mu_{N-2})^3(N + \mu_1))}{(N^3 + \mu_1(M + \mu_{N-2})^3)^2} \tag{3}$$

$$\frac{N + 2\sqrt{\lambda_1} - 2\lambda_3}{\lambda_1} \tag{4}$$

$$\frac{N(M + N - \mu_{N-2})}{2(N + \mu_1\mu_4)} \tag{5}$$

$$\frac{N + \lambda_{N-2} + 4}{\sqrt{M}} \tag{6}$$

$$\frac{\mu_{N-2}(3N - \mu_1)}{N\mu_N + \mu_1\mu_{N-2}} \tag{7}$$

Formulas for real-world graphs diameter. The results obtained with these formulas are shown in Table 3.

$$\mu_{N-3} + \mu_{N-4} - 2 + 4(\mu_{N-4} - 2)^2 - \frac{\mu_{N-1}(\mu_{N-3} + \mu_{N-4} + \sigma)}{5} \tag{8}$$

$$\sqrt{\frac{2\lambda_4^3}{(\lambda_1 + \lambda_{N-4})^3} + \frac{(\lambda_1 + N + \sigma)}{(\lambda_1 + \lambda_{N-4})} + \lambda_N} \tag{9}$$

$$\left((\mu_{N-4} - 2)^2 + \frac{(\mu_{N-4}^2 + 1)}{\frac{\mu_2^2}{125} + \mu_5 - (\mu_{N-4} - 2)^2}\right)^2 \tag{10}$$

$$((\sqrt{(3-\sqrt{\mu_{N-4}})})^2((3-\sqrt{\mu_{N-4}})^2-2)) \tag{11}$$

$$\frac{N+2\mu_{N-3}\delta_1+\mu_2+\mu_5+\sigma-2(\mu_{N-4}-5)^3}{\mu_{N-3}\delta_1+\mu_2+\mu_3+\mu_{N-4}+\sigma} \tag{12}$$

$$\mu_{N-3}-2\mu_{N-4}^2-((\mu_{N-4}-1)^3-\sqrt{3})^3 \tag{13}$$

$$\sqrt{2}\sqrt{\lambda_N+\frac{\lambda_1^3}{(\lambda_1+\lambda_{N-4})^3}+\frac{\lambda_1+N+\sigma}{\lambda_1+\lambda_{N-4}}} \tag{14}$$

$$\frac{2M}{\lambda_1\lambda_2^2}+\frac{\lambda_5^2+2(\lambda_N-\lambda_3)+50}{\lambda_1}+\frac{2}{\lambda_1\lambda_2} \tag{15}$$

Formulas for random graphs geodetic number. The results obtained with these formulas are shown in Table 4.

$$\sqrt{\frac{N^{3/2}}{\lambda_1}}-\frac{\lambda_{N-4}N^{3/2}}{\lambda_1^2+N^{3/2}}. \tag{16}$$

$$\frac{\mu_4^2}{\mu_2\mu_{N-3}}+\sqrt{N-\mu_3} \tag{17}$$

$$\frac{N\lambda_4}{3\lambda_1}+\sqrt{5} \tag{18}$$

$$\frac{5(N\mu_4+5\mu_1-5\mu_{N-3})}{\mu_4^2+10\mu_4+25} \tag{19}$$

Formulas for real-world graphs geodetic number. The results obtained with these formulas are shown in Table 5.

$$\delta_1+\sigma+\sqrt{\delta_1}-\lambda_2-\lambda_3\lambda_{N-1}-\lambda_{N-2}\lambda_{N-4}+\frac{N}{\delta_1} \tag{20}$$

$$\frac{\delta_1^2(\delta_1+\sigma)+\delta_1 N}{(\delta_1^2+N)}+\sqrt{\delta_1+\sigma+\lambda_N+\frac{N}{\delta_1}}+\frac{N}{\delta_1}+\frac{N}{\delta_1^2}+1 \tag{21}$$

$$\delta_1 + \sigma + \lambda_{N-4} + \sqrt{N} + \frac{\lambda_N}{\delta_1} + \frac{N(\delta_1 + 1)}{\delta_1^2 + \delta_1 \sigma} \tag{22}$$

$$\delta_1 + \sigma + \sqrt{\delta_1} - \frac{\mu_2 \mu_{N-3}}{\mu_{N-4}} + \mu_2 + \frac{N}{\delta_1} - \sqrt{\frac{\delta_1^2}{\mu_2 N} + \frac{\delta_1 \sigma}{\mu_2 N}} \tag{23}$$

$$\delta_1 + \sigma - \frac{\delta_1}{\sqrt{N}} - \mu_2 \mu_{N-2} + \mu_2 \mu_{N-4} + 2\mu_4 - 2\mu_5 - \mu_{N-2} + \mu_{N-4} + \sqrt{N} \tag{24}$$

$$\delta_1 + \sigma - 2\mu_{N-2} + \sqrt{-2\mu_5 + N} - 8 + \frac{\mu_2}{\mu_{N-2} + 3} + \frac{N}{\delta_1} \tag{25}$$

$$\delta_1 + \sigma + \frac{\mu_1}{\delta_1 + \sigma} - \mu_2 \mu_{N-2} + \mu_4 + 9\mu_{N-2}^2 \mu_{N-3}^2 + \sqrt{\mu_{N-4}}(\mu_2 - \mu_5) \tag{26}$$

$$\delta_1 + \sigma + \sqrt{M} - 2 \tag{27}$$