Hindawi
Scientific Programming
Volume 2022, Article ID 6076369, 15 pages
https://doi.org/10.1155/2022/6076369

Research Article

Binary Equilibrium Optimization Algorithm for Computing Connected Domination Metric Dimension Problem

Basma Mohamed,[1] **Linda Mohaisen**,[2] **and Mohamed Amin**[1]

[1]*Mathematics and Computer Science Department, Faculty of Science, Menoufia University, Shebin Elkom 32511, Egypt*
[2]*Faculty of Computer and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia*

Correspondence should be addressed to Basma Mohamed; bosbos25jan@yahoo.com

Received 5 May 2022; Revised 14 September 2022; Accepted 22 September 2022; Published 6 October 2022

Academic Editor: Guokai Zhang

We consider, in this paper, the NP-hard problem of finding the minimum connected domination metric dimension of graphs. A vertex set B of a connected graph $G = (V, E)$ resolves G if every vertex of G is uniquely identified by its vector of distances to the vertices in B. A resolving set B of G is connected if the subgraph $\overline{B}$ induced by B is a nontrivial connected subgraph of G. A resolving set is dominating if every vertex of G that does not belong to B is a neighbor to some vertices in B. The cardinality of the smallest resolving set of G, the cardinality of the minimal connected resolving set, and the cardinality of the minimal connected domination resolving set are the metric dimension of G, connected metric dimension of G, and connected domination metric dimension of G, respectively. We present the first attempt to compute heuristically the minimum connected dominant resolving set of graphs by a binary version of the equilibrium optimization algorithm (BEOA). The particles of BEOA are binary-encoded and used to represent which one of the vertices of the graph belongs to the connected domination resolving set. The feasibility is enforced by repairing particles such that an additional vertex generated from vertices of G is added to B, and this repairing process is iterated until B becomes the connected domination resolving set. The proposed BEOA is tested using graph results that are computed theoretically and compared to competitive algorithms. Computational results and their analysis show that BEOA outperforms the binary Grey Wolf Optimizer (BGWO), the binary Particle Swarm Optimizer (BPSO), the binary Whale Optimizer (BWO), the binary Slime Mould Optimizer (BSMO), the binary Grasshopper Optimizer (BGO), the binary Artificial Ecosystem Optimizer (BAEO), and the binary Elephant Herding Optimizer (BEHO).

1. Introduction

Recently, connected domination metric dimension of graphs is introduced in [1]. Metric dimension has several applications in robot navigation [2, 3], network discovery and verification [4], application to wireless sensor network localization [5], pattern recognition, image processing [6], combinatorial optimization [7], and applications to pharmaceutical chemistry [8]. Domination theory is applied in wireless communication networks [9], electrical networks [10], and chemical structures [11]. The connected domination set with smallest cardinality is a natural candidate to be used for the information exchange in any kind of network. The utilization of connected domination sets of graphs for defining virtual backbone architecture in ad hoc wireless networks has been explored in [12–15].

2. Problem Description

Let $G = (V, E)$ be a connected graph, and let $d(u,v)$ be the shortest path between two vertices $u, v \in V(G)$. An ordered vertex set $B = \{x_1, x_2, ..., x_k\} \subseteq V(G)$ is a resolving set of G if the representation

$$r(v|B) = (d(v, x1), d(v, x2), \ldots, d(v, xk)), \tag{1}$$

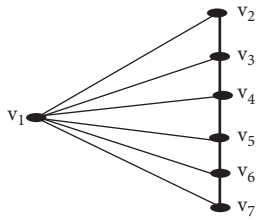

Figure 1: Fan graph F_7.

is unique for every $v \in V(G)$. A resolving set B is a dominating set of G if every vertex of $V\backslash B$ has at least one neighbor that belongs to B. A dominating resolving set B is connected if the subgraph $\overline{B}$ induced by B is a nontrivial connected subgraph of G. Let $Card\ (X)$ denote the cardinality of a set X. The metric dimension of G, denoted as $\dim(G)$, the domination metric dimension of G, denoted as $Ddim(G)$, and the connected domination metric dimension of G, denoted as $\gamma_{cr}\ (G)$, are defined as

$$\begin{aligned} \dim(G) &= \min\{\mathrm{Card}(B):\ B \text{ is a resolving set of } G\}, \\ \dim(G) &= \min\{\mathrm{Card}(B):\ B \text{ is a domination resolving set of } G\}, \\ Y_{cr}(G) &= \min\{\mathrm{Card}(B):\ B \text{ is a connected dominating resolving set of } G\}. \end{aligned} \tag{2}$$

Example 1. For the fan graph F_7 given in Figure 1, the set $B = \{v_1, v_3, v_5\}$ is a minimal resolving set, so $\dim(F_7) = 3$. B is also a minimal domination resolving set since every vertex of $V\backslash B$ has at least one neighbor that belongs to B, e.g., v_2 is adjacent to v_1 and v_3, $v_4$is adjacent to v_1, v_3 and v_5, $v_6$is adjacent to v_1, v_5 and v_7, and $v_7$is adjacent to v_1, v_6, so$D$ di m $(F_7) = 3$. However, B is not connected domination resolving set of F_7 since v_4 is not connected to vertices in B. On the other hand, the set $B = \{v_1, v_{3,} v_4, v_5\}$ is a minimal connected domination resolving set of F_7, so γ_{cr} $(F_7) = 4$.

The connected domination metric dimension problem combines three components: connectedness, domination, and metric dimension of graphs. The problem of finding the metric dimension of a graph G is described in terms of an integer programming problem [8]. Let $D = [d_{ij}]$ be the distance matrix of G, $d_{ij} = d(v_i, v_j)$ for $1 \le i, j \le n$. For $x_i \in \{0, 1\}$, $1 \le i \le n$, the function F is defined by

$$F(x_1, x_{2,} \ldots\ldots, x_n) = x_1 + x_2 + \cdots + x_n. \tag{3}$$

Minimizing F subject to the $(n/2)$ constraints

$$\begin{aligned} &\left|d_{i1} - d_{j1}\right| x_1 + \left|d_{i2} - d_{j2}\right| x_2 + \ldots\ldots\ldots\ldots \\ &+ \left|d_{\mathrm{in}} - d_{\mathrm{jn}}\right| x_n > 0 \text{for} 1 \le i < j \le n, \end{aligned} \tag{4}$$

is equivalent to finding a basis in the sense that if $x_1', x_2', \ldots, x_n'$ is a set of values for which F attains its minimum, then $B = \{v_i, x_i' = 1\}$is a basis for G and conversely, if $B = \{v_{i1}, v_{i2}, \ldots, v_{in}\}$is a basis for G and if we define

$$x_s' = \begin{cases} 1, & \text{if} s = i_j \text{forsome} j\ (1 \le j \le k) \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

then $F(\ x_1', x_2', \ldots, x_n')$ is a minimum subject to the given constraints.

The dominating set problem is NP-complete [16], and the metric dimension problem is NP-complete [17]. Consequently, the connected domination metric dimension γ_{cr} (G) is a typical NP-complete problem that involves determining if $\gamma_{cr}\ (G) \le K$ for a given graph G and input K.

The rest of the paper is organized as follows: Section 3 gives a literature review. Section 4 introduces the Equilibrium Optimization Algorithm (EOA). Section 5 gives the BEOA for computing the connected domination metric dimension. Section 7 reports computational results. Finally, Section 7 provides the conclusion and future work.

3. Literature Review

Metric dimension, domination metric dimension, and connected domination metric dimension of graphs are determined theoretically for several graphs in the literature. A short overview of the main recently determined theoretically metric dimension results [17–23] is given hereafter. The metric dimensions of the total graph of path powers three and four are determined theoretically in [18], symmetrical planar pyramid graph and reflection symmetrical planar pyramid graph in [19], m-level gear graph and generalized gear graph in [17], kayak paddles graph and cycles with chord in [20], circulant graphs in [21], generalized families of Toeplitz graphs in [22], and windmill graphs in [23].

The connected metric dimension are determined theoretically in [24, 25]. Cycle graph, wheel graph, star graph, complete graph, and path graph are determined in [24] and Petersen graph in [25].

The dominant metric dimension are determined theoretically in [26, 27]. Path graph, cycle graph, star graph, complete graph, and complete bipartite graph are determined in [26], the corona product graph of G and H is investigated whenever H is a path graph, and the cycle graph, complete bipartite graph, complete graph, and star graph S_n are determined in [27].

Initialize the particle's populations $i = 1,\ldots,n$
Assign equilibrium candidates' fitness a large number
Assign free parameters $a_1 = 2$; $a_2 = 1$; $GP = 0.5$;
While Iter < Max_iter
  For $i = 1$: number of particles (n)
  Calculate fitness of i^{th} particle
    If $(C_i) < (C_{\text{eq0}})$
      Replace $C_{\text{eq0}}$with C_i and fit (C_{eq0}) with (C_i)
  Convert each $C_i$into binary using S-shaped transfer function in Cbinary $_{ij}$
    Calculate the fitness of each Cbinary $_{ij}$
    Update new position of the candidate using (16)
Else if $(C_i) > (C_{\text{eq0}}$) & $(C_i)) < (C_{\text{eq1}})$
      Replace C_{eq1} with C_i and fit (C_{eq1}) with (C_i)
      Convert each $C_i$into binary using S-shaped transfer function in Cbinary $_{ij}$
      Calculate the fitness of each Cbinary $_{ij}$
      Update new position of the candidate using (16)
Else if $(C_i) > ((C_{\text{eq0}}$) & $(C_i) > (C_{\text{eq1}})$ & $(C_i) < (C_{\text{eq2}})$
      Replace C_{eq2} with C_i and fit (C_{eq2}) with (C_i)
      Convert each $C_i$into binary using S-shaped transfer function in $Cbinary_{ij}$
      Calculate the fitness of each $Cbinary_{ij}$
      Update new position of the candidate using (16)
Else if $(C_i) > (C_{\text{eq0}})$ & $(C_i) > (C_{\text{eq1}})$ & $(C_i) > (C_{\text{eq2}})$ & $(C_i)) < (C_{\text{eq3}})$
      Replace C_{eq3} with C_i and fit $(C_{\text{eq3}}$) with (C_i)
      Convert each $C_i$into binary using S-shaped transfer function in Cbinary $_{ij}$
      Calculate the fitness of each Cbinary $_{ij}$
      Update new position of the candidate using (16)
  End (If)
End (For)
$C_{\text{ave}} = C_{\text{eqs}(0)} + C_{\text{eqs}(1)} + C_{\text{eqs}(2)} + C_{\text{eqs}(3)}/4$
Construct the equilibrium pool $C_{\text{eqs,pool}} = \{C_{\text{eqs}(0)}, C_{\text{eqs}(2)}, C_{\text{eqs}(3)}, C_{\text{ave}}\}$
Accomplish memory saving (if Iter >1)
Assign $t = (1 - \text{iter/maxIter})^{(a_2 \text{iter/maxIter})}$using (12)
    For $i = 1$: number of particles (n)
Randomly choose one candidate from the equilibrium pool (vector)
Generate random vectors of $\vec{\lambda}$, $\vec{\text{r}}$ using (11)
Construct$F = \text{a}_1 \text{sign}(r_2 - 0.5)(e^{-\lambda t} - 1)$ using (11)
Construct $GCP = \begin{cases} 0.5 & \text{r}_2 \geq GP \\ 0 & \text{r}_2 < GP \end{cases}$ using (15)
Construct $G_o = GCP \cdot (C_{\text{eq}} - \lambda C)$using (14)
Construct $G = G_o F$ using (13)
Update concentrations $C = C_{\text{eq}} + (C - C_{eq}).F + G/\lambda V$ (1- F) using (16)
    End (For)
  Iter = Iter + 1
End while
Return candidate with best fitness value

Algorithm 1: Pseudo-BEOA.

The connected domination metric dimensions of complete graph, path graph, and cycle graph are determined theoretically in [1].

On the other hand, only a few algorithms have been proposed to compute heuristically the metric dimension problem [28–30]. In [28], a genetic algorithm has been developed for computing the metric dimension of many classes of graph instances including pseudo-Boolean, crew scheduling and graph coloring. The binary representation, frozen genes mutation, a limited number of different individuals with the same objective value, and the caching technique were all applied. Infeasible individuals are changed by the addition of the required nodes in order to become feasible. In [29], PSO is adapted for determining the metric dimension where a real valued vector of vertices is converted to binary valued vector by a linear function and infeasible particles are repaired by adding some vertices until the particles become feasible. The PSO is tested by computing the metric dimension of hypercube graphs. A variable neighborhood search approach has been proposed for solving metric dimension and minimal doubly resolving set problems in order to improve the existing upper bounds in [30]. The variable neighborhood search approach is based on a decomposition of the metric dimension problem and the minimal doubly resolving set problem into a sequence of subproblems with an auxiliary objective function. In

addition, for both problems, the corresponding new integer linear programming formulations are proposed.

Here, the operations of the EOA are encoded and adapted to solve the connected domination metric dimension problem. The proposed BEOA is tested using graph results that are computed theoretically and is compared to competitive algorithms.

4. Equilibrium Optimization Algorithm (EOA)

The original EOA is a physics-based metaheuristic algorithm for handling continuous optimization problems in [31]. The EOA is a recently proposed metaheuristic algorithm that uses an equilibrium pool and candidates to update particles. The EOA is based on the analytic solution procedure for a well-mixed dynamic mass balance on a control volume. The EOA avoids falling into the local optimum in addition to having great exploitation and exploration capabilities. These benefits of EOA are made possible by the concept of generation rate.

The mass balance equation is solved analytically, yielding the following results:

$$C = C_{eq} + \left(C_0 + C_{eq}\right).F + \frac{G}{\lambda V}\left(1 - F\right), \tag{6}$$

where V is considered to be 1 as the volume unit.

The EOA generates an equilibrium pool with four candidates and another averaged one:

$$C_{\text{eqs,pool}} = \left\{C_{\text{eqs}(0)}, C_{\text{eqs}(1)}, C_{\text{eqs}(2)}, C_{\text{eqs}(3)}, C_{\text{ave}}\right\}, \tag{7}$$

$$C_{\text{ave}} = \frac{C_{\text{eqs}(0)} + C_{\text{eqs}(1)} + C_{\text{eqs}(2)} + C_{\text{eqs}(3)}}{4}. \tag{8}$$

The fitness values of the candidates in the equilibrium pool should satisfy the following rules for a given problem represented by f:

$$f\left(C_{\text{eqs}(0)}\right) \le f\left(C_{\text{eqs}(1)}\right) \le f\left(C_{\text{eqs}(2)}\right) \le f\left(C_{\text{eqs}(3)}\right). \tag{9}$$

EOA uses the same initializing and iterating techniques as other bioinspired algorithms when solving problems, whether they are stated as benchmarks or come from real engineering work. While iterating, the EOA continues its domain exploration and exploitation operations. To improve performance, there are numerous specific operations in constructing the EOA.

4.1. The Initialization Procedure. Assuming that the presented problems are constrained by a symmetric or asymmetric domain with [lb,ub], the candidates of the swarms are uniformly distributed across the domain. To achieve this, the pseudo-random random number r_1 is introduced:

$$C_i = lb + (ub - lb).r_1. \tag{10}$$

The position vector for the ith candidate is C_i.

4.2. The Random Chosen Candidate. The positions of the candidates for the following iteration would be significant to three phases during the iterations, as equated in equation (6). C_{eq} is a randomly selected candidate from the swimming pool constructing with equation (7) for the first phase.

Table 1: Parameter setting.

Algorithms	Parameter name	Value
BEOA	Particles numbers	50
	Max iteration	300
	a_1, a_2, GP, λ	[2, 1, 0.5, [0–1]]
	Number of runs	20
BGWO	Search agents	50
	Max iteration	300
	a_1, a_2	[2, 0]
	Number of runs	20
BPSO	Swarm size	50
	C_1	Increases linearly from 0.5 to 2.5
	C_2	Decreases linearly from 2.5 to 0.5
	Inertia weight (w)	0.8
	Max iteration	300
	Number of runs	20
BWOA	Population size	50
	a_1	[2, 0]
	a_2	−2, 1]
	b	1
	Max iteration	300
	Number of runs	20
BSMA	Population size	50
	z	0.03
	Max iteration	300
	Number of runs	20
BGOA	Population size	50
	G_{MaX}	1
	G_{Min}	0.004
	Max iteration	300
	Number of runs	20
BAEO	Population size	50
	r_1,r_2, and r	rand
	Max iteration	300
	Number of runs	20
BEHO	Elephants number	50
	Clans number	5
	Kept elephants number	2
	The scale factor α	0.5
	The scale factor β	0.1
	Max iteration	300
	Number of runs	20

4.3. The Exponential Parameter. There is an exponential parameter F for the second part of equation (6), which is determined as follows:

$$F = \mathrm{a}_1 \mathrm{sign}\left(r_2 - 0.5\right)\left(e^{-\lambda t} - 1\right), \tag{11}$$

where a_1 is a constant variable that controls exploring capabilities as well as a parameter splitting the exploring and exploiting procedure. The higher the value of a_1, the higher

Table 2: Results on path graph P_n.

Path graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
4	3	γ_{cr}	2	2	2	2	2	2	2	2
		t (sec)	**21.1**	32.3	29.8	43.2	27.9	58.1	46.3	53.2
		Iteration	1	1	1	2	1	3	2	2
5	4	γ_{cr}	3	3	3	3	3	3	3	3
		t (sec)	57.2	65.1	43.9	65.9	**33.1**	76.8	89.1	81.7
		Iteration	2	6	3	8	2	7	10	8
6	5	γ_{cr}	4	4	4	4	4	4	4	4
		t (sec)	**94.7**	151.8	127.4	189.6	114.9	235.7	218.6	256.1
		Iteration	4	9	24	15	10	39	28	41
7	6	γ_{cr}	5	5	5	5	5	5	5	5
		t(sec)	**135.9**	196.1	211.6	269.4	173.2	291.8	265.1	379.7
		Iteration	9	38	32	18	26	43	29	51
8	7	γ_{cr}	6	6	6	6	6	6	6	6
		t (sec)	**227.6**	212.4	284.2	376.6	218.1	325.7	337.5	412.9
		Iteration	12	15	45	39	18	52	35	49
9	8	γ_{cr}	7	7	7	7	7	7	7	7
		t(sec)	**159.1**	281.2	324.1	428.2	292.1	417.7	421.6	563.2
		Iteration	8	52	23	47	28	63	40	31
10	9	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	**194.6**	338.3	369.5	491.7	350.6	501.9	547.4	612.9
		Iteration	23	76	16	59	42	0	29	63
11	10	γ_{cr}	9	9	9	9	9	9	9	9
		t(sec)	238.2	391.7	412.1	570.3	**211.4**	588.1	682.9	659.1
		Iteration	16	31	49	35	56	73	81	24
12	11	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	**304.9**	473.7	554.03	635.9	398.5	657.4	739.5	745.8
		Iteration	40	67	72	84	34	62	83	56
13	12	γ_{cr}	11	11	11	11	11	11	11	11
		t(sec)	**279.5**	592.6	386.8	719.1	490.3	745.8	823.6	869.9
		Iteration	24	28	40	92	77	54	69	128
14	13	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	**196.4**	689.8	716.5	835.1	583.6	809.4	904.7	972.6
		Iteration	13	39	84	71	55	104	90	62
15	14	γ_{cr}	13	13	13	13	13	13	13	13
		t(sec)	**232.8**	794.4	819.2	903.6	668.19	916.3	1071.5	1057.2
		Iteration	37	56	92	80	48	74	141	97
16	15	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**327.5**	862.07	1045.99	1134.2	805.37	1089.5	1167.2	1196.1
		Iteration	28	75	56	68	39	106	129	83
17	16	γ_{cr}	15	15	15	15	15	15	15	15
		t(sec)	**253.9**	949.2	1112.7	1285.3	928.7	1175.3	1342.6	1308.9
		Iteration	11	63	80	68	39	106	129	83

the probability for candidates to carry on exploration, and the smaller the probability for candidates to carry on exploitation. For experience and convenience, $a_1 = 2.r_2$ is another random number with the interval of [0, 1], and t is a parameter formulated to be related to the iteration times.

$$t = \left(1 - \frac{\text{iter}}{\text{maxIter}}\right)^{\left(a_2 \text{iter}/\text{maxIter}\right)}, \tag{12}$$

where iter denotes the current iteration number, and maxIter denotes the maximum iteration number restricted at the start.

4.4. The Generation Rate Parameter. The word G stands for the generation rate parameter, which improves the exploiting capability of candidates. The rate of generation is proportional to the exponential parameter, which is given as follows:

Table 3: Results on cycle graph.

Cycle graph N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
4	4	γ_{cr}	2	2	2	2	2	2	2	2
		t(sec)	**32.1**	48.8	39.8	56.3	35.2	51.7	63.9	54.6
		Iteration	1	1	1	2	1	2	1	3
5	5	γ_{cr}	3	3	3	3	3	3	3	3
		t(sec)	91.1	85.2	72.1	103.9	**52.8**	125.7	149.5	111.3
		Iteration	1	4	24	21	8	17	13	29
6	6	γ_{cr}	4	4	4	4	4	4	4	4
		t(sec)	**127.7**	156.5	191.4	215.9	144.3	203.6	227.1	219.7
		Iteration	6	16	32	35	11	37	53	48
7	7	γ_{cr}	5	5	5	5	5	5	5	5
		t(sec)	**191.6**	125.03	234.2	302.4	203.5	319.3	337.8	326.2
		Iteration	19	54	45	82	29	64	56	70
8	8	γ_{cr}	6	6	6	6	6	6	6	6
		t(sec)	**115.9**	284.2	301.1	396.8	281.6	381.4	413.2	397.9
		Iteration	3	20	23	38	16	31	47	63
9	9	γ_{cr}	7	7	7	7	7	7	7	7
		t(sec)	**183.2**	317.8	369.5	443.7	325.1	465.7	499.4	482.8
		Iteration	17	39	16	49	32	61	47	63
10	10	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	248.6	386.9	412.1	506.03	**212.4**	536.8	574.9	601.3
		Iteration	29	53	59	78	42	92	85	109
11	11	γ_{cr}	9	9	9	9	9	9	9	9
		t(sec)	**347.8**	439.3	490.03	589.7	434.9	592.5	687.1	694.7
		Iteration	52	64	72	126	69	87	123	95
12	12	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	**381.2**	472.6	596.8	654.7	509.4	671.4	756.9	741.2
		Iteration	11	142	50	94	42	63	38	116
13	13	γ_{cr}	11	11	11	11	11	11	11	11
		t(sec)	**226.9**	585.9	503.5	728.4	564.1	745.3	819.2	804.9
		Iteration	9	38	184	146	19	88	72	61
14	14	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	**295.7**	703.5	756.99	801.7	638.4	815.1	910.3	895.6
		Iteration	43	95	132	161	57	115	96	158
15	15	γ_{cr}	13	13	13	13	13	13	13	13
		t(sec)	**312.6**	872.8	845.1	879.6	717.3	897.4	1067.5	980.3
		Iteration	16	27	71	109	26	147	65	106
16	16	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**266.2**	924.4	908.7	955.3	802.6	980.2	1206.7	1096.1
		Iteration	12	32	39	67	50	135	46	83
17	17	γ_{cr}	15	15	15	15	15	15	15	15
		t(sec)	**231.9**	1165.3	1013.1	1168.5	854.3	1091.8	1315.7	1289.6
		Iteration	5	11	57	73	42	90	81	65

$$G = G_o F, \tag{13}$$

$$G_o = GCP \cdot \left(C_{\text{eq}} - \lambda C\right), \tag{14}$$

$$\text{GCP} = \begin{cases} 0.5, & r_2 \geq \text{GP}, \\ 0, & r_2 \geq \text{GP}. \end{cases} \tag{15}$$

GP stands for generation probability. *GP* is set to be 0.5 to achieve better results in balancing the probability between exploration and exploitation. Updating rule of EO will be as follows:

$$C = C_{\text{eq}} + \left(C - C_{\text{eq}}\right).F + \frac{G}{\lambda V}\left(1 - \text{F}\right), \tag{16}$$

where F is defined in equation (11) and V is considered as unit.

Table 4: Results on triangular snake graph.

Triangular snake graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
3	3	γ_{cr}	2	2	2	2	2	2	2	2
		t(sec)	**29.1**	64.1	45.1	83.5	39.8	86.04	95.8	57.3
		Iteration	1	1	1	2	1	3	2	3
5	6	γ_{cr}	3	3	3	3	3	3	3	3
		t(sec)	**53.3**	99.4	76.4	109.5	68.1	123.8	162.3	117.4
		Iteration	1	3	2	4	2	3	8	6
7	9	γ_{cr}	4	4	4	4	4	4	4	4
		t(sec)	83.8	121.2	97.2	135.7	**77.03**	165.8	191.5	178.3
		Iteration	2	2	4	7	3	9	21	13
9	12	γ_{cr}	5	5	5	5	5	5	5	5
		t(sec)	**92.2**	143.9	171.3	198.2	134.9	209.8	227.1	213.5
		Iteration	5	8	11	26	18	52	39	21
11	15	γ_{cr}	6	6	6	6	6	6	6	6
		t(sec)	131.4	221.6	286.9	311.8	**109.2**	278.1	301.4	298.3
		Iteration	9	13	25	41	17	67	82	70
13	18	γ_{cr}	7	7	7	7	7	7	7	7
		t(sec)	**187.2**	273.5	314.1	393.4	205.7	345.9	383.2	374.1
		Iteration	21	20	16	73	31	55	67	92
15	21	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	**233.4**	301.3	423.9	458.2	290.3	401.7	469.1	418.3
		Iteration	14	28	41	65	24	71	52	78
17	24	γ_{cr}	9	9	9	9	9	9	9	9
		t(sec)	**315.1**	479.1	567.2	509.4	368.9	486.5	553.8	546.7
		Iteration	30	25	32	50	38	66	87	74
19	27	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	**401.3**	592.9	645.3	587.5	419.6	533.1	607.8	603.5
		Iteration	19	46	66	93	60	84	95	101
21	30	γ_{cr}	11	11	11	11	11	11	11	11
		t(sec)	**519.2**	708.2	742.2	665.9	536.7	598.5	691.3	679.4
		Iteration	28	53	40	82	51	45	68	94
23	33	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	612.4	873.1	807.1	748.3	**583.5**	681.5	776.4	780.6
		Iteration	53	95	54	69	43	108	85	72
25	36	γ_{cr}	13	13	13	13	13	13	13	13
		t(sec)	**527.2**	928.9	892.4	811.3	671.2	764.9	859.4	906.3
		Iteration	25	73	92	77	58	84	98	117
27	39	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**631.1**	815.7	985.8	994.3	728.2	841.4	904.7	1021.8
		Iteration	14	57	89	63	35	76	73	93
29	42	γ_{cr}	15	15	15	15	15	15	15	15
		t(sec)	598.4	1073.2	1044.2	1175.1	804.1	917.5	1015.3	1139.5
		Iteration	9	48	75	113	51	106	67	83
31	45	γ_{cr}	16	16	16	16	16	16	16	16
		t(sec)	702.6	1291.8	1459.1	1286.5	892.6	1093.7	1276.3	1197.9
		Iteration	17	72	91	57	38	63	81	70

Table 5: Results on double triangular snake graph.

Double triangular snake graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
4	5	γ_{cr}	2	2	2	2	2	2	2	2
		t(sec)	**32.1**	56.2	79.3	89.4	45.2	92.02	81.3	94.1
		Iteration	1	1	1	3	1	2	3	2
7	10	γ_{cr}	4	4	4	4	4	4	4	4
		t(sec)	**82.5**	91.5	97.1	105.3	94.1	130.5	109.6	107.4
		Iteration	1	5	4	2	3	4	2	9
10	15	γ_{cr}	6	6	6	6	6	6	6	6
		t(sec)	116.3	157.4	199.4	181.9	103.1	191.4	**75.3**	184.9
		Iteration	5	7	11	13	6	17	14	32
13	20	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	**194.4**	248.1	273.2	265.4	227.6	263.2	230.8	271.4
		Iteration	14	4	6	29	19	17	14	40
16	25	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	**231.7**	376.2	327.9	341.5	284.9	380.4	312.2	359.3
		Iteration	9	9	10	28	15	25	14	31
19	30	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	361.8	295.8	415.2	497.3	**342.1**	449.2	428.6	418.9
		Iteration	25	22	38	46	31	40	28	34
22	35	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**436.8**	578.4	568.1	560.7	471.8	509.1	569.3	556.2
		Iteration	33	36	44	71	50	63	47	68
25	40	γ_{cr}	16	16	16	16	16	16	16	16
		t(sec)	**621.4**	759.2	889.8	749.1	702.6	657.5	673.8	651.6
		Iteration	27	65	87	132	43	80	65	51
28	45	γ_{cr}	18	18	18	18	18	18	18	18
		t(sec)	**745.6**	956.3	1034.7	913.5	827.9	795.6	768.4	780.5
		Iteration	15	49	26	35	19	62	73	59
31	50	γ_{cr}	20	20	20	20	20	20	20	20
		t(sec)	**867.5**	1132.6	1282.6	1158.8	935.2	891.4	911.7	857.4
		Iteration	54	113	72	66	76	83	109	94
34	55	γ_{cr}	22	22	22	22	22	22	22	22
		t(sec)	991.2	1439.4	1545.2	1282.3	1091.6	1018.2	**966.7**	1026.3
		Iteration	41	85	119	73	62	67	81	101
37	60	γ_{cr}	24	24	24	24	24	24	24	24
		t(sec)	**1023.9**	1674.2	1617.5	1376.5	1180.7	1227.8	1149.1	1304.8
		Iteration	12	38	38	56	68	79	95	83
40	65	γ_{cr}	26	26	26	26	26	26	26	26
		t(sec)	**984.1**	1813.8	1849.9	1455.3	1273.2	1315.9	1292.7	1419.3
		Iteration	35	57	69	119	45	104	62	70
43	70	γ_{cr}	28	28	28	28	28	28	28	28
		t(sec)	**1179.2**	1734.7	2011.2	1629.8	1390.5	1509.2	1487.3	1593.8
		Iteration	16	49	74	67	58	96	69	54
46	75	γ_{cr}	30	30	30	30	30	30	30	30
		t(sec)	**1045.9**	2006.1	2273.4	1937.5	1511.1	1784.9	1673.5	1760.3
		Iteration	9	36	52	90	34	83	48	39

5. Binary EOA for the Connected Dominated Metric Dimension Problem

[31]. The high performance of EOA is mainly derived from its extensive design in balancing exploration and exploitation abilities. This advantage makes a binary version of the algorithm that uses binary encoding, which can be adapted to solve the connected domination metric dimension problem. In the continuous version of EOA, particles can move around the search space using position vectors within the continuous real domain. We convert EOA to binary values by applying an S-shaped transfer function to transform the continuous variable into a binary one. In discrete binary search space, position updates require switching between 0 and 1.

The following equation is used in the initialization step.

$$C\text{binary}_{ij} = \begin{cases} 1, & \text{ran d} \geq 0.5, \\ 0, & \text{else.} \end{cases} \tag{17}$$

Table 6: Results on $(2,k)c_4$-snake graph.

$(2,k)c_4$-snake graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
6	8	γ_{cr}	4	4	4	4	4	4	4	4
		t(sec)	**35.1**	81.5	74.2	65.4	42.3	56.9	48.5	50.7
		Iteration	1	1	1	4	1	3	2	2
11	16	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	**93.3**	134.3	117.3	109.1	98.5	103.2	97.6	101.8
		Iteration	1	3	6	8	2	5	4	3
16	24	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	162.5	190.2	136.9	186.5	190.2	186.4	201.3	**128.3**
		Iteration	3	9	15	19	7	13	16	5
21	32	γ_{cr}	16	16	16	16	16	16	16	16
		t(sec)	281.2	245.1	341.5	319.4	**230.8**	291.5	308.3	275.8
		Iteration	15	28	27	25	18	34	41	37
26	40	γ_{cr}	20	20	20	20	20	20	20	20
		t(sec)	356.4	307.9	474.8	381.6	**301.2**	387.2	393.5	369.1
		Iteration	19	51	39	54	47	78	57	61
31	48	γ_{cr}	24	24	24	24	24	24	24	24
		t(sec)	**408.7**	591.4	627.6	495.3	452.6	507.2	489.4	496.5
		Iteration	12	80	45	76	58	65	49	68
36	56	γ_{cr}	28	28	28	28	28	28	28	28
		t(sec)	**561.8**	831.2	791.4	713.8	609.1	628.4	652.6	638.4
		Iteration	28	59	26	64	33	83	56	45
41	64	γ_{cr}	32	32	32	32	32	32	32	32
		t(sec)	**637.2**	955.1	1011.9	970.6	736.8	721.3	781.5	806.1
		Iteration	11	36	74	53	45	91	73	51
46	72	γ_{cr}	36	36	36	36	36	36	36	36
		t(sec)	**801.9**	1146.8	1273.8	1085.3	876.9	842.6	905.1	957.4
		Iteration	18	47	62	61	70	59	45	39
51	80	γ_{cr}	40	40	40	40	40	40	40	40
		t(sec)	**903.6**	1586.2	1415.2	1272.6	990.4	1077.6	1058.5	1131.9
		Iteration	37	54	129	78	46	80	59	53
56	88	γ_{cr}	44	44	44	44	44	44	44	44
		t(sec)	**1095.5**	1710.4	1649.6	1384.3	1248.5	1361.8	1390.7	1459.4
		Iteration	10	121	74	95	34	53	40	68
61	96	γ_{cr}	48	48	48	48	48	48	48	48
		t(sec)	**1207.8**	1824.5	1922.6	1572.9	1391.7	1538.9	1571.2	1598.3
		Iteration	29	109	69	60	51	65	70	49
66	104	γ_{cr}	52	52	52	52	52	52	52	52
		t(sec)	**1176.9**	2017.6	2196.7	1738.1	1473.4	1714.5	1749.8	1824.5
		Iteration	25	83	115	72	36	40	67	31
71	112	γ_{cr}	56	56	56	56	56	56	56	56
		t(sec)	**1311.7**	2186.5	2338.8	1899.4	1612.9	1873.8	1971.1	1912.4
		Iteration	17	76	61	55	42	61	38	51
76	120	γ_{cr}	60	60	60	60	60	60	60	60
		t(sec)	**1455.8**	2252.3	2513.1	2047.3	1739.5	2096.4	2134.7	2086.3
		Iteration	14	48	75	81	25	34	47	42

The rand() has uniform distribution, with a value of [0.0, 1.0], and $Cbinary_{ij}$ is binary valued position vector. To convert continuous values to binary ones, a transfer function is used. The sigmoid function (S) is used in this study as follows:

$$S = \frac{1}{1 + e^{-10x^d}}, \tag{18}$$

where x^d indicates the continuous-valued position at dimension d and S is the function output. To generate a binary value, apply the following equation:

$$Cbinary_{ij} = \begin{cases} 1, & \text{ran d} < S, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

The proposed algorithm deals with the connected domination resolving set problem as an optimization problem where it searches for the best solution, so each

Table 7: Results on linear kc_4-snake graph.

Linear kc_4-snake graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
4	4	γ_{cr}	2	2	2	2	2	2	2	2
		t(sec)	**26.1**	48.2	59.3	80.7	36.2	73.1	62.3	57.1
		Iteration	1	1	1	3	1	2	3	1
7	8	γ_{cr}	4	4	4	4	4	4	4	4
		t(sec)	**71.3**	115.3	87.4	109.1	67.8	90.5	75.8	76.3
		Iteration	1	3	2	5	2	3	4	2
10	12	γ_{cr}	6	6	6	6	6	6	6	6
		t(sec)	165.4	198.7	131.2	148.9	94.6	121.	110.1	**91.6**
		Iteration	5	8	2	10	4	8	13	9
13	16	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	**207.5**	286.3	245.6	269.3	233.5	250.2	262.8	243.9
		Iteration	9	19	14	23	16	13	19	12
16	20	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	335.8	369.1	**312.8**	352.8	340.9	359.3	381.4	361.8
		Iteration	15	22	25	40	28	17	36	27
19	24	γ_{cr}	12	12	12	12	12	12	12	12\
		t(sec)	**399.3**	528.4	496.7	461.3	408.1	420.7	473.5	451.9
		Iteration	13	39	44	68	36	25	49	32
22	28	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**482.2**	690.8	702.4	583.1	571.9	539.4	552.9	502.5
		Iteration	28	52	59	57	44	36	63	40
25	32	γ_{cr}	16	16	16	16	16	16	16	16
		t(sec)	596.4	**531.9**	853.1	675.8	610.3	621.9	643.4	619.7
		Iteration	15	18	58	65	30	52	22	37
28	36	γ_{cr}	18	18	18	18	18	18	18	18
		t(sec)	**711.9**	825.2	949.5	784.5	739.1	760.3	726.1	704.9
		Iteration	19	26	109	82	54	70	41	58
31	40	γ_{cr}	20	20	20	20	20	20	20	20
		t(sec)	**842.1**	1094.5	1015.4	917.3	901.9	938.8	952.7	986.1
		Iteration	42	98	84	50	40	61	38	45
34	44	γ_{cr}	22	22	22	22	22	22	22	22
		t(sec)	**957.5**	1238.1	1284.7	1149.8	1054.6	1127.3	1014.9	1096.5
		Iteration	34	74	71	64	51	24	32	39
37	48	γ_{cr}	24	24	24	24	24	24	24	24
		t(sec)	**1095.1**	1452.4	1457.4	1323.6	1179.2	1315.8	1205.2	1273.9
		Iteration	12	92	98	56	38	29	59	43
40	52	γ_{cr}	26	26	26	26	26	26	26	26
		t(sec)	**1264.2**	1698.9	1736.2	1494.8	1321.6	1429.5	1339.4	1407.2
		Iteration	37	63	59	31	47	36	44	51
43	56	γ_{cr}	28	28	28	28	28	28	28	28
		t(sec)	**981.4**	1759.2	1977.2	1607.3	1458.8	1537.9	1503.4	1591.6
		Iteration	23	48	42	37	29	34	52	47
46	60	γ_{cr}	32	32	32	32	32	32	32	32
		t(sec)	**1149.3**	1904.4	2193.8	1725.9	1631.6	1729.2	1681.5	1708.1
		Iteration	15	39	67	52	38	45	31	35

particle can be represented as a one-dimensional vector $C\text{binary}ij = \{C_{i1}, C_{i2}, C_{i3}, \ldots, C_{ij}\}$, and $C\text{binary}ij$ is a binary valued position vector if j-th element of the vector has a value of 1; it means that vertex j belongs to B. If every $v \in V(G)$ has a unique representation $r(v|B)$, then B is a connected domination resolving set. The value of a binary valued position vector is produced by computing the value of the S-shaped transfer function. In the BEO algorithm, when a particle is not feasible as a connected domination resolving set, that particle is repaired by adding a vertex from $V\backslash B$. This repair is applied until that particle becomes connected domination resolving set.

Table 8: Results on $nc_4 \circ 2p_n$ graph.

$nc_4 \circ 2p_n$ graph										
N	M		BEOA	BGWO	BPSO	BWO	BSMO	BGO	BAEO	BEHO
4	4	γ_{cr}	2	2	2	2	2	2	2	2
		t(sec)	**19.2**	31.4	27.1	49.8	43.9	62.5	54.8	47.5
		Iteration	1	1	1	4	2	2	5	2
8	10	γ_{cr}	5	5	5	5	5	5	5	5
		t(sec)	51.4	95.8	**46.5**	84.3	66.4	93.8	99.1	80.6
		Iteration	2	3	2	7	5	4	9	6
12	16	γ_{cr}	6	6	6	6	6	6	6	6
		t(sec)	**99.3**	148.5	103.4	112.6	101.7	139.3	128.4	119.2
		Iteration	8	5	9	14	10	8	7	13
16	22	γ_{cr}	8	8	8	8	8	8	8	8
		t(sec)	145.4	259.2	**131.2**	181.9	196.5	204.7	171.3	165.7
		Iteration	13	18	15	23	19	32	14	25
20	28	γ_{cr}	10	10	10	10	10	10	10	10
		t(sec)	**234.1**	374.1	285.8	299.1	271.8	262.3	256.9	240.4
		Iteration	27	23	32	58	36	41	29	44
24	34	γ_{cr}	12	12	12	12	12	12	12	12
		t(sec)	**315.4**	452.3	432.2	467.4	349.3	398.5	362.1	335.7
		Iteration	34	69	48	71	55	46	37	50
28	40	γ_{cr}	14	14	14	14	14	14	14	14
		t(sec)	**452.2**	561.8	590.3	594.3	512.8	529.2	478.9	496.4
		Iteration	19	15	87	33	28	57	45	38
32	46	γ_{cr}	16	16	16	16	16	16	16	16
		t(sec)	607.5	705.1	745.1	718.9	**571.6**	639.5	595.1	624.9
		Iteration	25	43	103	85	37	64	59	41
36	52	γ_{cr}	18	18	18	18	18	18	18	18
		t(sec)	**748.4**	890.3	913.6	794.3	789.2	802.6	757.4	763.8
		Iteration	17	76	71	62	30	41	38	56
40	58	γ_{cr}	20	20	20	20	20	20	20	20
		t(sec)	**873.1**	1087.8	1169.4	1015.6	918.3	1048.9	961.4	1091.1
		Iteration	31	121	52	40	25	46	62	48
44	64	γ_{cr}	22	22	22	22	22	22	22	22
		t(sec)	**994.5**	1264.2	1325.7	1293.8	1076.1	1181.4	1074.2	1238.4
		Iteration	39	87	98	63	71	80	57	51
48	70	γ_{cr}	24	24	24	24	24	24	24	24
		t(sec)	**1127.2**	1416.5	1694.8	1470.3	1213.9	1309.1	1256.9	1395.8
		Iteration	22	49	63	134	36	51	68	77
52	76	γ_{cr}	26	26	26	26	26	26	26	26
		t(sec)	**1209.4**	1634.2	1383.2	1596.4	1408.5	1484.6	1393.4	1504.1
		Iteration	15	32	58	86	29	62	54	39
56	82	γ_{cr}	28	28	28	28	28	28	28	28
		t(sec)	**1028.4**	1913.5	1894.1	1852.3	1543.8	1631.9	1519.2	1665.5
		Iteration	19	76	71	79	37	55	46	43
60	88	γ_{cr}	30	30	30	30	30	30	30	30
		t(sec)	**1125.3**	2026.9	2257.5	2067.4	1712.9	1822.6	1705.1	1839.3
		Iteration	10	51	94	61	26	42	50	38

The algorithm represents each solution (individual) in the population as a string of binary in which 1 means that the connected domination resolving set will be chosen, and then the corresponding value will be "1," and if the connected domination resolving set is not selected, then the corresponding value will be "0."

The proposed BEO algorithm is displayed in Algorithm 1.

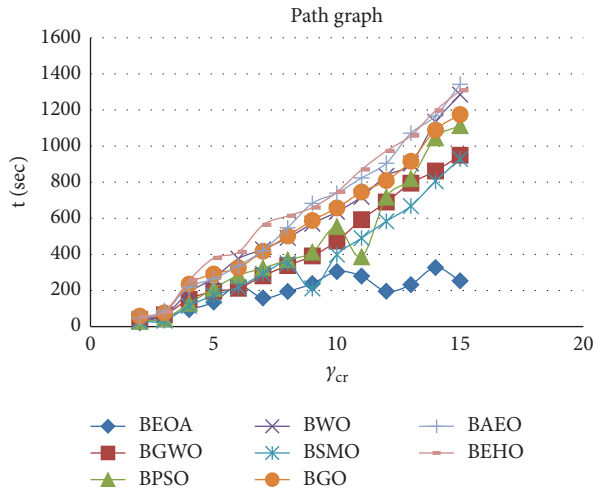

Figure 2: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for path graph.

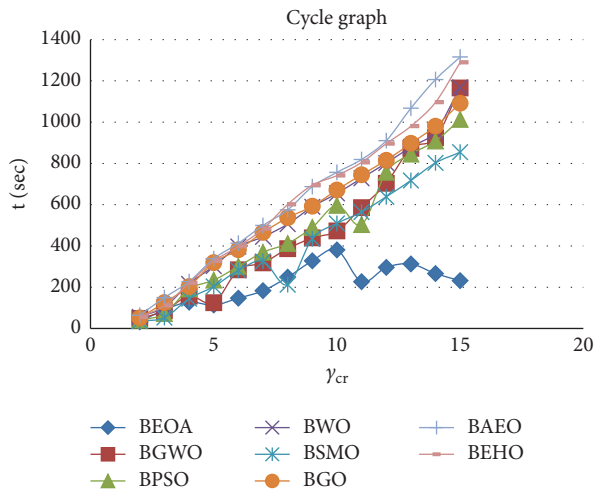

Figure 3: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for cycle graph.

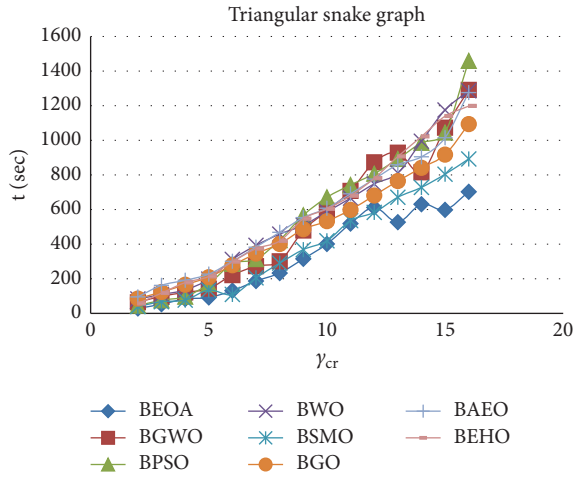

Figure 4: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for triangular snake graph.

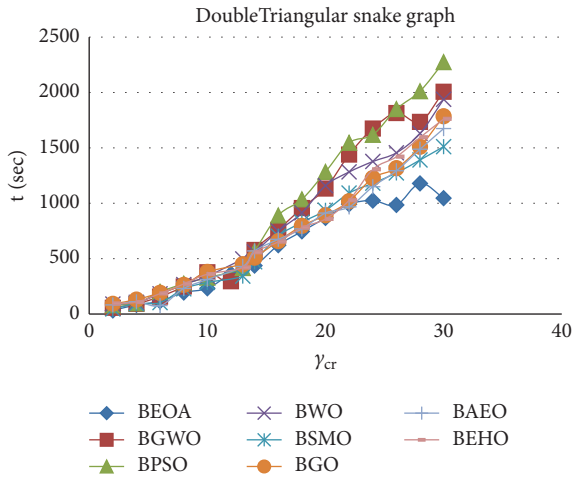

Figure 5: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for double triangular snake graph.

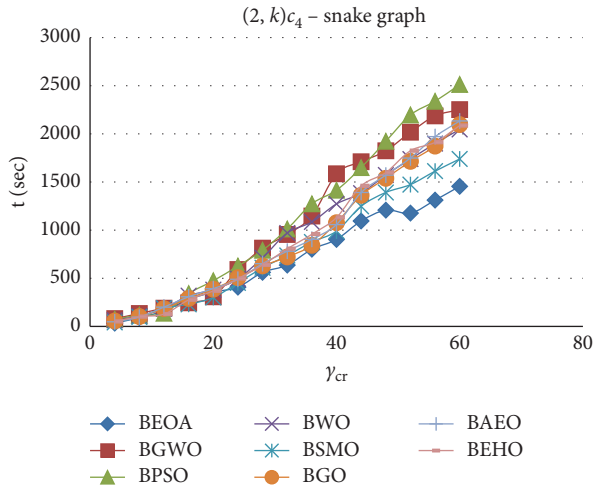

Figure 6: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for $(2,k)c_4$ – snake graph.

6. Results and Discussion

This section summarizes the results of BEOA applied to graph instances that are computed theoretically and the family of snake graphs.

6.1. Experimental Results. In this section, the proposed BEOA is compared to the BGWO, BPSO, BWOA, BSMA, BGOA, BAEO, and BEHO algorithms. The algorithms are applied to a path graph, a cycle graph, a triangular snake graph, a double triangular snake graph, $(2,k)c_4$– snake graph, a linear kc_4-snake graph, and a nc_4 ○ $2p_n$ graph. The algorithm tests and comparisons were performed on the windows 10 Ultimate 64 bit operating system; the processor was an Intel Core i7 running at 16 GB of RAM, the hard drive was a 1TBHDD+1TBSSD, and the code was implemented in MATLAB 2021b. The parameter setting values are presented in Table 1.

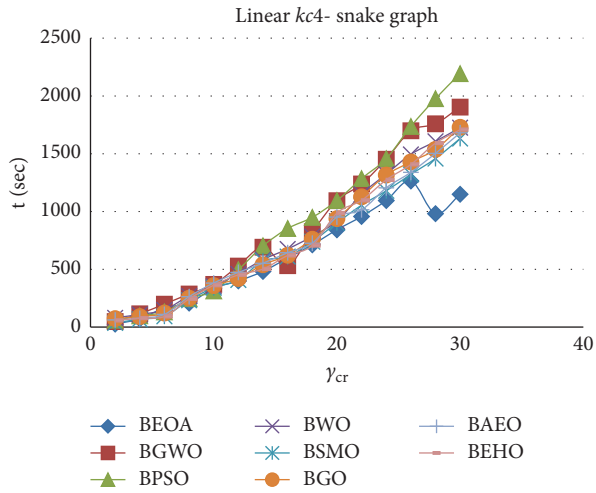

Figure 7: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for linear kc_4-snake graph.

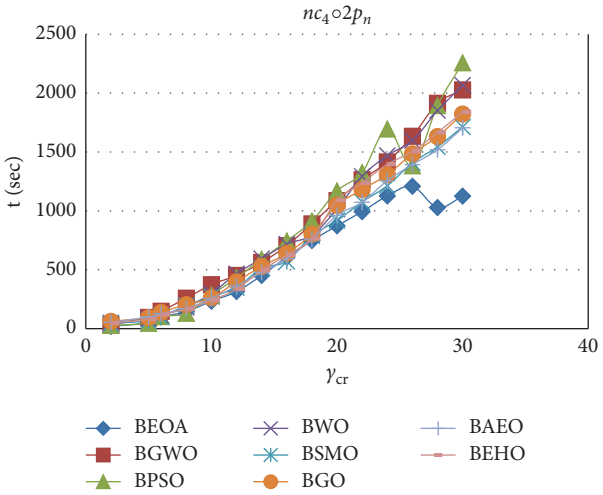

Figure 8: Comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for $nc_4 \circ 2p_n$ graph.

All algorithms have been run 20 times for each graph, and the results are summarized in Tables 2–8.

The tables are organized as follows: The first three columns contain the number of nodes N, the number of edges M, the connected domination resolving number γ_{cr}, and the CPU time (t) used to indicate the exactly connected domination resolving number and iteration: the average number of iterations for finishing the algorithms to achieve the best solution, respectively.

Table 2 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected domination resolving number for path graph P_n, $4 \le n \le 17$, and BEOA has reached an optimal solution.

Table 3 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for cycle graph C_n, $4 \le n \le 17$, and BEOA has reached an optimal solution.

Table 4 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for triangular snake graph T_n, $3 \le n \le 33$, and BEOA has reached an optimal solution.

Table 5 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for double triangular snake graph DT_n, $4 \le n \le 46$.

Table 6 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for (2,k) c_4– snake graph, $1 \le k \le 15$ and $6 \le n \le 76$.

Table 7 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for linear kc_4-snake graph, $1 \le k \le 15$ and $4 \le n \le 46$.

Table 8 shows a comparison between BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO for computing connected dominating resolving number for $nc_4 \circ 2P_n$ graph and $4 \le n \le 60$.

6.2. Comparison. To further demonstrate the excellence of proposed BEOA, we choose BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO algorithms to conduct experiments under the same conditions and compared the results.

The results on graphs are shown in Tables 2–8, which indicate that the proposed BEOA algorithm outperforms other algorithms on graphs, reaching 253.87 sec in BEOA, 949.24 sec in BGWO,1112.73 sec in BPSO,1285.3 sec in BWO, 928.7 sec in BSMO,1175.3 sec in BGO,1342.6 sec in BAEO, and 1308.9 sec in BEHO for path graph, 231.93 sec in BEOA, 1165.33 sec in BGWO,1013.1 sec in BPSO,1168.5 sec in BWO,854.3 sec in BSMO,1091.8 sec in BGO,1315.7 sec in BAEO, and 1289.6 sec in BEHO for cycle graph, 702.6 sec in BEOA,1291.8 sec in BGWO,1459.1sec in BPSO,1286.5 sec in BWO, 892.6 sec in BSMO, 1093.7 sec in BGO,1276.3 sec in BAEO, and 1197.9 sec in BEHO for triangular snake graph, and 1045.9 sec in BEOA, 2006.1sec in BGWO, 2273.4 sec in BPSO 1937.5 sec in BWO, 1511.1 sec in BSMO, 1784.9 sec in BGO,1673.5 sec in BAEO, and 1760.3 sec in BEHO for double triangular snake graph, 1455.8 sec in BEOA, 2252.3 sec in BGWO, 2513.1 sec in BPSO, 2047.3 sec in BWO, 1739.5 sec in BSMO, 2096.4 sec in BGO, 2134.7 sec in BAEO, and 2086.3 sec in BEHO for (2,k)c_4– snake graph, 1149.1 sec in BEOA, 1904.4 sec in BGWO, 2193.8 sec in BPSO, 1725.9 sec in BWO, 1631.6 sec in BSMO, 1729.2 sec in BGO, 1681.5 sec in BAEO, and 1708.1 sec in BEHO for linear kc_4-snake graph, and 1125.3 sec in BEOA, 2026.9 sec in BGWO, 2257.5 sec in BPSO, 2067.4 sec in BWO, 1712.9 sec in BSMO, 1822.6 sec in BGO, 1705.1 sec in BAEO, and 1839.3 sec in BEHO for $nc_4 \circ 2p_n$ graph. It proves the correctness and superiority of the proposed BEOA.

Figures 2–8 show the superiority of the proposed BEOA on the BEOA, BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO according to the connected domination resolving number.

7. Conclusion

In this paper, a binary variant of the basic equilibrium optimization algorithm BEOA for determining the minimum connected domination resolving set of graphs and compared to BGWO, BPSO, BWO, BSMO, BGO, BAEO, and BEHO. Comparisons were performed on the graphs: a path graph, cycle graph, triangular snake graph, double triangular snake graph, $(2,k)c_4$– snake graph, linear kc_4-snake graph, and $nc_4 \circ 2p_n$ graph. Experimental results and their analysis confirmed the superiority of the proposed BEOA for solving the connected domination metric dimension problem.

For further work in the future, we plan to compute other variants of metric dimension by other metaheuristic algorithms and compare them with competitive algorithms.

Data Availability

https://invoicing.hindawi.com/payment-details/c4921783-ade7-470d-acf7-cd6f9ec421fb The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the present study.

References

[1] A. Naji and N. D. Soner, "Resolving connected domination in graphs," *Mathematical Combinatorics*, vol. 4, pp. 129–136, 2015.

[2] S. Khuller, B. Raghavachari, and A. Rosenfeld, "Landmarks in graphs," *Discrete Applied Mathematics*, vol. 70, no. 3, pp. 217–229, 1996.

[3] R. Manjusha and S. Kuriakose A, "Metric dimension and uncertainty of traversing robots in a network," *International Journal on Applications of Graph Theory In wireless Ad Hoc Networks And sensor Networks*, vol. 7, no. 2/3, pp. 1–9, 2015.

[4] Z. Beerliova, F. Eberhard, T. Erlebach et al., "Network discovery and verification," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2168–2181, 2006.

[5] M. Idrees, H. Ma, M. Wu, A. R. Nizami, M. Munir, and S. Ali, "Metric dimension of generalized Mobius ladder and its application to wsn localization," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 24, no. 1, pp. 3–11, 2020.

[6] R. A. Melter and I. Tomescu, "Metric bases in digital geometry," *Computer Vision, Graphics, and Image Processing*, vol. 25, no. 1, pp. 113–121, 1984.

[7] A. Sebő and E. Tannier, "On metric generators of graphs," *Mathematics of Operations Research*, vol. 29, no. 2, pp. 383–393, 2004.

[8] G. Chartrand, L. Eroh, M. A. Johnson, and O. R. Oellermann, "Resolvability in graphs and the metric dimension of a graph," *Discrete Applied Mathematics*, vol. 105, no. 1-3, pp. 99–113, 2000.

[9] J. L. Hurink and T. Nieberg, "Approximating minimum independent dominating sets in wireless networks," *Information Processing Letters*, vol. 109, no. 2, pp. 155–160, 2008.

[10] A. H. Karbasi and R. E. Atani, "Application of dominating sets in wireless sensor networks," *International Journal of Security and Its Applications*, vol. 7, no. 4, pp. 185–202, 2013.

[11] D. Vukičević and A. Klobučar, "K-Dominating sets on linear benzenoids and on the infinite hexagonal grid," *Croatica Chemica Acta*, vol. 80, no. 2, pp. 187–191, 2007.

[12] K. M. Alzoubi, P. J. Wan, and O. Frieder, "Distributed heuristics for connected dominating sets in wireless ad hoc networks," *Journal of Communications and Networks*, vol. 4, no. 1, pp. 22–29, 2002.

[13] S. Butenko, X. Cheng, D. Z. Du, and P. M Pardalos, "On the construction of virtual backbone for ad hoc wireless network," in *Cooperative Control: Models, Applications and Algorithms*, pp. 43–54, Springer, Boston, MA, 2003.

[14] R. Gandhi and S. Parthasarathy, "Distributed algorithms for connected domination in wireless networks," *Journal of Parallel and Distributed Computing*, vol. 67, no. 7, pp. 848–862, 2007.

[15] D. Z. Du and P. J. Wan, "Weighted CDS in unit disk graph," in *Connected Dominating Set: Theory and Applications*, pp. 77–104, Springer, New York, NY, 2013.

[16] M. R. Garey, "A guide to the theory of NP-completeness," *Computers and Intractability*, vol. 23, 1979.

[17] S. Imran, M. K. Siddiqui, M. Imran et al., "Computing the metric dimension of gear graphs," *Symmetry*, vol. 10, no. 6, p. 209, 2018.

[18] S. Nawaz, M. Ali, M. A. Khan, and S. Khan, "Computing metric dimension of power of total graph," *IEEE Access*, vol. 9, pp. 74550–74561, 2021.

[19] S. Nazeer, M. Hussain, F. A. Alrawajeh, and S. Almotairi, "Metric dimension on path-related graphs," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–12, Article ID 2085778, 2021.

[20] A. Ahmad, M. Bača, and S. Sultan, "Computing the metric dimension of kayak paddles graph and cycles with chord," *Proyecciones*, vol. 39, no. 2, pp. 287–300, 2020.

[21] A. Borchert and S. Gosselin, "The metric dimension of circulant graphs and Cayley hypergraphs," *Utilitas Mathematica*, vol. 106, pp. 125–147, 2018.

[22] M. F. Nadeem, S. Qu, A. Ahmad, and M. Azeem, "Metric dimension of some generalized families of toeplitz graphs," *Mathematical Problems in Engineering*, vol. 2022, pp. 1–10, Article ID 9155291, 2022.

[23] P. Singh, S. Sharma, S. K. Sharma, and V. K. Bhat, "Metric dimension and edge metric dimension of windmill graphs," *AIMS Mathematics*, vol. 6, no. 9, pp. 9138–9153, 2021.

[24] V. Saenpholphat and P. Zhang, "Connected resolvability of graphs," *Czechoslovak Mathematical Journal*, vol. 53, no. 4, pp. 827–840, 2003.

[25] L. Eroh, C. X. Kang, and E. Yi, "The connected metric dimension at a vertex of a graph," *Theoretical Computer Science*, vol. 806, pp. 53–69, 2020.

[26] L. Susilowati, I. Sa'adah, R. Z. Fauziyyah, A. Erfanian, and Slamin, "The dominant metric dimension of graphs," *Heliyon*, vol. 6, no. 3, Article ID e03633, 2020.

[27] R. P. Adirasari, H. Suprajitno, and L. Susilowati, "The dominant metric dimension of corona product graphs," *Baghdad Science Journal*, vol. 18, no. 2, 2021.

[28] J. Kratica, V. Kovačević-Vujčić, and M. Čangalović, "Computing the metric dimension of graphs by genetic algorithms," *Computational Optimization and Applications*, vol. 44, no. 2, pp. 343–361, 2009.

[29] D. T. Murdiansyah, "Computing the metric dimension of hypercube graphs by particle swarm optimization

algorithms," in *Proceedings of the International Conference on Soft Computing and Data Mining*, pp. 171–178, Springer, NY Cham, June 2016.

[30] N. Mladenović, J. Kratica, V. Kovačević-Vujčić, and M. Čangalović, "Variable neighborhood search for metric dimension and minimal doubly resolving set problems," *European Journal of Operational Research*, vol. 220, no. 2, pp. 328–337, 2012.

[31] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, "Equilibrium optimizer: a novel optimization algorithm," *Knowledge-Based Systems*, vol. 191, pp. 105190–190, 2020.