Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.032930
Article

Tech Science Press

check for updates

Computing Connected Resolvability of Graphs Using Binary Enhanced Harris Hawks Optimization

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
*Corresponding Author: Basma Mohamed. Email: bosbos25jan@yahoo.com
Received: 02 June 2022; Accepted: 22 September 2022

Abstract: In this paper, we consider the NP-hard problem of finding the minimum connected resolving set of graphs. A vertex set B of a connected graph G 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. The cardinality of the minimal resolving set is the metric dimension of G and the cardinality of minimum connected resolving set is the connected metric dimension of G. The problem is solved heuristically by a binary version of an enhanced Harris Hawk Optimization (BEHHO) algorithm. This is the first attempt to determine the connected resolving set heuristically. BEHHO combines classical HHO with opposition-based learning, chaotic local search and is equipped with an *S*-shaped transfer function to convert the continuous variable into a binary one. The hawks of BEHHO are binary encoded and are used to represent which one of the vertices of a graph belongs to the connected resolving set. The feasibility is enforced by repairing hawks such that an additional node selected from $V\backslash B$ is added to B up to obtain the connected resolving set. The proposed BEHHO algorithm is compared to binary Harris Hawk Optimization (BHHO), binary opposition-based learning Harris Hawk Optimization (BOHHO), binary chaotic local search Harris Hawk Optimization (BCHHO) algorithms. Computational results confirm the superiority of the BEHHO for determining connected metric dimension.

Keywords: Connected resolving set; binary optimization; harris hawks algorithm

1 Introduction

Recently, a connected resolving set of graphs has been introduced in [1]. Let $G = (V, E)$ be a connected graph and $d(u, v)$ be the shortest path between two vertices $u, v \in V(G)$. An ordered vertex set $B = \{x_1, x_2, \ldots, x_k\} \subseteq V(G)$ is a resolving set of G if the representation

$$r(v|B) = (\mathrm{d}(v, x_1), \mathrm{d}(v, x_2), \ldots, \mathrm{d}(v, x_k))$$

is unique for every $v \in V(G)$. A resolving set B of G is connected if the subgraph $\bar{B}$ induced by B is a nontrivial connected subgraph of G. Let $|B|$ be the cardinality of B, the metric dimension and the connected metric dimension of G, denoted dim(G) and cdim(G), respectively, are defined as

dim(G) = min$\{|B_i|$: $B_i \subseteq 2^v$, B_i is a resolving set of $G\}$, cdim(G) = min $\{|B_i|$: $B_i \subseteq 2^v$, B_i is a connected resolving set of $G\}$.

Since every connected resolving set is a resolving set, dim(G) ≤ cdim (G) for all connected graphs G. To illustrate this notion, consider the graph G in Fig. 1.

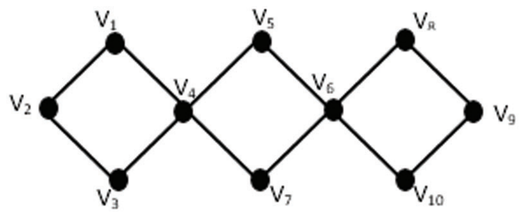

Figure 1: A graph G with dim (G) = 3 and cdim (G) = 5

Example 1.1

The set $B = \{v_1, v_5, v_8\}$ is a minimum resolving set for G since the representations

r($v_1|B$) = (0, 2, 4), r($v_2|B$) = (1, 3, 5), r($v_3|B$) = (2, 2, 4), r($v_4|B$) = (1, 1, 3), r($v_5|B$) = (2, 0, 2), r($v_6|B$) = (3, 1, 1), r($v_7|B$) = (2, 2, 2), r($v_8|B$) = (4, 2, 0), r($v_9|B$) = (5, 3, 1), r($v_{10}|B$) = (4, 2, 2)

for the vertices of G are distinct. Thus, dim(G) = 3. The subgraph induced by B, $\overline{B} = (B,E)$, $E = \varnothing$ is disconnected. Thus, B is not a connected resolving set for G. Namely, no 3-element subset is a connected resolving set of G. On the other hand, the set $B = \{v_1, v_4, v_5, v_6, v_8\}$ is a connected resolving set since the representations

r($v_1|B$) = (0, 1, 2, 3, 4), r($v_2|B$) = (1, 2, 3, 4, 5), r($v_3|B$) = (2, 1, 2, 3, 4), r($v_4|B$) = (1, 0, 1, 2, 3), r($v_5|B$) = (2, 1, 0, 1, 2), r($v_6|B$) = (3, 2, 1, 0, 1), r($v_7|B$) = (2, 1, 2, 1, 2), r($v_8|B$) = (4, 3, 2, 1, 0), r($v_9|B$) = (5, 4, 3, 2, 1), r($v_{10}|B$) = (4, 3, 2, 1, 2). The set $B = \{v_1, v_4, v_5, v_6, v_8\}$

are distinct and the subgraph induced by B, $\bar{B} = (B, E) = (\{v_1, v_4, v_5, v_6, v_8\}, \{\{v_1, v_4\},\{\{v_4, v_5\},\{\{v_5, v_6\},\{v_6, v_8\})$ is a connected resolving set, hence cdim (G) = 5.

The metric dimension of several graphs is computed theoretically in the literature [2–19]. A few algorithms are proposed in the literature to compute the metric dimension of graphs heuristically. These are genetic algorithm [20], particle swarm optimization [21] and variable neighborhood search [22].

The connected metric dimension of some graphs is computed theoretically in [1,23]. In [1] it is proved that the connected metric dimension of the wheel graph W_n, $n \geq 7$ is $\left\lfloor \frac{2n+2}{5} \right\rfloor + 1$, the star graph $K_{1,n-1}$, $n \geq 4$ is $n - 1$. In [23], it is proved that the connected metric dimension of path graph P_n, $n \geq 2$ is 2 and the complete graph K_n, $n \geq 3$ is $n - 1$.

This study is the first attempt to compute the minimum connected resolving set of graphs heuristically. For this purpose, we use a binary version of an enhanced Harris Hawk Optimization (BEHHO) algorithm. The results of graphs that are computed theoretically are used for testing the proposed BEHHO algorithm. The proposed BEHHO algorithm is compared to competitive algorithms on the family of triangular snake graphs.

This paper is organized as follows: Section 2 presents the Harris Hawks optimizer. Section 3 presents the proposed BEHHO for determining the connected resolving set of a given graph. Computational results are presented in Section 4. Finally, the conclusion is stated in Section 5.

2 Harris Hawks Optimizer

The Harris hawk optimizer (HHO) is a swarm-based algorithm that simulates the cooperative manner and chasing behavior of harris [24,25]. The hawks try to hunt the prey by using several approaches (tracing, encircling and finally approaching and attacking). Hawks' skillful tactic to hunt the escaping prey is known as "surprise pounce". The mathematical model comprises three phases: exploration, the transition between exploration and exploitation; and exploitation (seeking and detecting prey). Based on the powerful characteristics of hawks in the exploration and exploitation processes of the search space, HHO is used to solve complex optimization tasks in a reasonable time [26]. The hawk's position can be determined from Eq. (1) as follows:

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1|X_{rand}(t) - 2r_2X(t)| & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)) & q < 0.5 \end{cases} \tag{1}$$

t is the current iteration, X_{rabbit} is the prey position, X_{rand} is the hawk which has been chosen randomly, q, r_1, r_2, r_3 and r_4 are random numbers in the range [0, 1], LB and UB refer to the lower bound and upper bound of variables respectively and X_m is the average position of hawks and can be calculated from Eq. (2) as follows:

$$X_m = \frac{1}{T} \sum\nolimits_{i=1}^{T} X_i(t) \tag{2}$$

where T is the maximum number of iterations and X_i (t) refers to each hawk position. The transition from exploration to exploitation is the second stage. Hawks can use many exploitative behaviors depending on the prey's escaping energy. In order to simulate the prey energy, the following energy is given:

$$E = 2E_0\left(1 - \frac{t}{T}\right) \tag{3}$$

$$E_0 = 2r - 1 \tag{4}$$

where E_o and E stand for the initial energy state and the prey's escaping energy respectively. E_o was randomly chosen in [−1, 1]. Exploration occurs when the $|E| \geq 1$ and exploitation occurs when $|E| < 1$.

The third stage is the exploitation phase in which hawks perform their surprise pounce. There are four situations to describe chasing tactics and astonishing attacks. Let r be a random number ∈ [0, 1].

- **Soft Besiege** When both r and $|E| \geq 0.5$, this behavior can be defined by the next Eq. (5)

$$X(t+1) = \Delta X(t) - E|JXrabbit(t) - X(t)| \tag{5}$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \tag{6}$$

where $\Delta X(t)$ is the distance between the current location and the prey position.

$$J = 2(1 - r_5), \text{where } r_5 \in [0,\ 1] \tag{7}$$

- **Hard Besiege** When $r \geq 0.5$ and $|E| < 0.5$. This behavior can be stated as follows:

$$X(t+1) = X_{rabbit}(t) - E|\Delta X(t)| \tag{8}$$

- **Soft Besiege with Rapid Dive** When $r < 0.5$ and $|E| \geq 0.5$. Then the prey has enough energy to escape from the attack. In order to describe this escaping pattern mathematically, Lévy fight (*LF*) is used as seen in Eq. (9).

$$LF(x) = 0.01 + \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}},\ \sigma = \left(\frac{\Gamma(1+\beta) \times \ sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \ \beta \ \times \ 2\ \left(\frac{\beta-1}{2}\right)} \right)^{\frac{1}{\beta}} \tag{9}$$

As a result, the hawk's position can be determined from Eq. (10) as follows:

$$X(t+1) = \begin{cases} X_{rabbit(t) - E|JX_{rabbit} - X(t)|} & F(Y) < \ F(X(t)) \\ Z = Y + S \times LF(D) & F(Z) < F(X(t)) \end{cases} \tag{10}$$

- **Hard Besiege with Rapid Dive** IF both r and $|E| < 0.5$. Then, the prey has enough energy to escape.

$$X(t+1) = \begin{cases} X_{rabbit(t) - \ E|JX_{rabbit}(t) - X_m(t)|} & F(Y) < \ F(X(t)) \\ Z = Y + S \times LF(D) & F(Z) < \ F(X(t)) \end{cases} \tag{11}$$

3 Proposed Binary Enhanced Harris Hawk Optimization (BEHHO)

Several recent developments to improve the performance of the Harris hawk optimizer are proposed [27–29]. Here, a binary enhanced hawk optimizer (BEHHO) that combines classical HHO, chaotic local search (CLS) and opposition-based learning techniques (OBL) is used to determine the connected resolving set. In the OBL strategy, each individual fitness is calculated, compared to its corresponding opposite number and the better one is then used for the next iteration [30]. The opposite number $\bar{x}$ is determined by the following Eq. (12) if x is a real number and $x \in [lb, ub]$.

$$x = ub + lb - x \tag{12}$$

where lb and ub stand for lower bound and upper bound.

Opposite vector If $x = (x_1,\ x_2, \ldots, x_D)$ and $x \in [lb,\ ub]$, then the following formula can be used to compute x_i:

$$x_i = lb_i + ub_i - x_i \tag{13}$$

if $f(\bar{x}) < f(x)$, then x_i will insert the current solution.

Chaos is known as a random-like phenomenon that occurs in both deterministic and non-linear systems. It is quite susceptible to its starting state [31]. Chaos completes the search more quickly than ergodic search [32]. A large variety of sequences can be generated by only modifying their initial values. This work employs a logistic map to create the following chaotic sequence:

$$o^{s+1} = Co^s(1 - o^s) \tag{14}$$

$C = 4$, $o^s = $ rand (0, 1) and $C_1 \neq 0.25$, 0.5 and 0.75 are the initial parameters.

Chaos optimization allows for reasonable execution times on a small scale, but as the search space grows, these times become unacceptable [33]. The properties of a chaotic system can be used to create a search operator, which can subsequently be incorporated into metaheuristic algorithms. The solutions produced by *CLS* can be acquired by

$$C_s = (1 - \mu) \times T + \mu \grave{C}_l, \ i = 1, 2, 3, \ldots, n \tag{15}$$

where C_s denotes the candidate solution and T denotes the target position μ is determined by the following equation.

$$\mu = \frac{MaxIter - currIter + 1}{Max_iter} \tag{16}$$

where the terms MaxIter and currIter stand for the maximum and current iterations, respectively. In order to maps $\grave{C}_i$ into the domain

$$\grave{C}_i = LB + C_i + (UB - LB) \tag{17}$$

where the initial solution upper and lower bounds are denoted by UB and LB.

Several approaches have been proposed to convert continuous algorithms to binary ones [34]. These binarization methods can be divided into two categories: The first is known as the continuous-binary operator transformation, in which the original real operators of metaheuristic equations are redefined into binary operators [35]. While in the second group, which is known as two step binarization, the real operators are used without modifications, while the produced continuous solutions are converted into binary by using two extra steps. The first step uses a transfer function (*TF*) that aims to transform the continuous solution R^n into an intermediate probability vector $[0, 1]^n$, where each element in this vector represents the probability of transforming the corresponding element in R^n to 1 or 0. In the second step, the intermediate solution is transformed into binary by applying various binarization methods [34].

$$x_{i,j} = \begin{cases} 0 & rand \geq 0.5 \\ 1 & rand < 0 \end{cases} \tag{18}$$

The binary value of *i*-th agent in the *j*-th dimension is denoted by $x_{i,j}$. A transfer function is used to be able to map continuous values to binary ones. In this study, the sigmoid function (*S*) is used as follows:

$$S = \frac{1}{1 + e^{-10x^d}} \tag{19}$$

where x^d indicates the continuous-valued position at dimension d and S is the function output. The following equation is used to generate a binary value.

$$x_{i,j} = \begin{cases} 1 & rand \geq S \\ 0 & rand < S \end{cases} \tag{20}$$

4 BEHHO for Connected Resolving Set Problem

When designing any optimizer, two fundamental components of the optimization problem should be considered; the solution representation and the evaluation function. The HHO algorithm was designed to solve continuous optimization problems, which is not appropriate for the connected resolving set problem. In the continuous version of HHO, hawks can move around the search space using position vectors within the continuous real domain. We convert the variables of EHHO to binary version 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 proposed algorithm deals with the connected resolving set problem as an optimization problem where it searches for the best solution, so each hawk can be represented as a one-dimensional vector $xbinary_{ij} = \{x_{i1}, x_{i2}, x_{i3}, \ldots, x_{ij}\}$, $xbinary_{ij}$ is a binary valued position vector if j-th element of the vector has a value 1, it means that vertex j belongs to B. If every $v \in V(G)$ has a distinct representation $r(v|B)$, then B is a connected resolving set. The value of a binary valued position vector is produced by computing the value of S-shaped transfer function. In BEHHO algorithm, when a hawk is not feasible as a connected resolving set, that hawk is repaired by adding a vertex from V\B. This repairing is applied until that hawk becomes a connected resolving set.

The algorithm represents each solution (individual) in the population as a string of binary in which 1 means that the connected resolving set will be chosen, then the corresponding value will be "1" and if the connected resolving set is not selected, then the corresponding value will be "0". Thus, the flowchart of the proposed BEHHO algorithm is displayed in Fig. 2 and the pseudo-code in Algorithm 1, respectively.

Algorithm 1: Pseudo-code of BEHHO

```
Set up the population parameters (Popsize (N), UB, LB, MaxIter(T) and Dimension of optimization problem)
Suppose i = 1
Begin evaluating the fitness function fitness[N] for each hawk x_i
Using Eq. (15), compute the opposition X → X̄ and the fitness function
From X∪X̄, find the best N solution
X_rabbit = currently ranked as the best solution
Let iter = 0
While (iter ≤ MaxIter) do
  For each hawk x_i, calculate the fitness function.
   The minimum connected resolving set is X_rabbit
          For each hawk (x_i) do
              Update the starting Energy E_0, jump force J and then update E using Eq. (3)
              If (|E| ≥ 1) then
                    Using Eq. (1), update the hawk position
               end if
   If (r ≥ 0.5 and |E| ≥ 0.5), then
    Update the hawk position by Eq. (5)
   else if (r ≥ 0.5 and |E| < 0.5) then
    Update hawk position by Eq. (8)
   else if (r < 0.5 and |E| ≥ 0.5) then
    Update the hawk position by Eq. (10)
```

(Continued)

Algorithm 1 (continued)

```
  else
    Use Eq. (11) to update the hawk position
  end if
end for
If (rand < OP) then
    Determine X̄_{i+1} and its fitness
      x_{i+1} = X̄_{i+1} if f(X̄_{i+1}) < f(x_{i+1})
end if
end while
Update X_rabbit
Using Eqs. (17)–(20), execute Chaotic Local Search
Return minimum connected resolving set
end
```

5 Experimental Results

The proposed BEHHO algorithm is compared to binary Harris Hawk Optimization (BHHO), binary opposition-based learning Harris Hawk Optimization (BOHHO), binary chaotic local search Harris Hawk Optimization (BCHHO) algorithms. The algorithms are applied to the star graph, wheel graph, the snake graphs instances: a triangular snake graph, a double triangular snake graph, a linear kC_4-snake graph and a (2, n) C_4-snake graph. The algorithms were run on the Windows 10 Ultimate 64-bit operating system; the processor was an Intel Core i7, 16 GB of RAM and the code was implemented in MATLAB 2021b. The parameter setting values are presented in Table 1.

Table 1: Experimental parameter setting

No.	Parameter name	Value
1	Pop size	50
2	Max iteration	100
3	Number of dimension of optimization problem	15
4	Chaotic initial parameter	4

All algorithms have been run 20 times for each instance and the results are summarized in Tables 2–7. The tabs are organized as follows:

- The first three columns contain the name of the test instance and the number of nodes and edges, respectively.
- The average execution time (t) used to reach the final algorithms for the first time is given.
- The iteration columns contain the average number of iterations for finishing algorithms.

In [1] Saenpholphat et al. computed the connected resolving set for a star graph and a complete graph only theoretically.

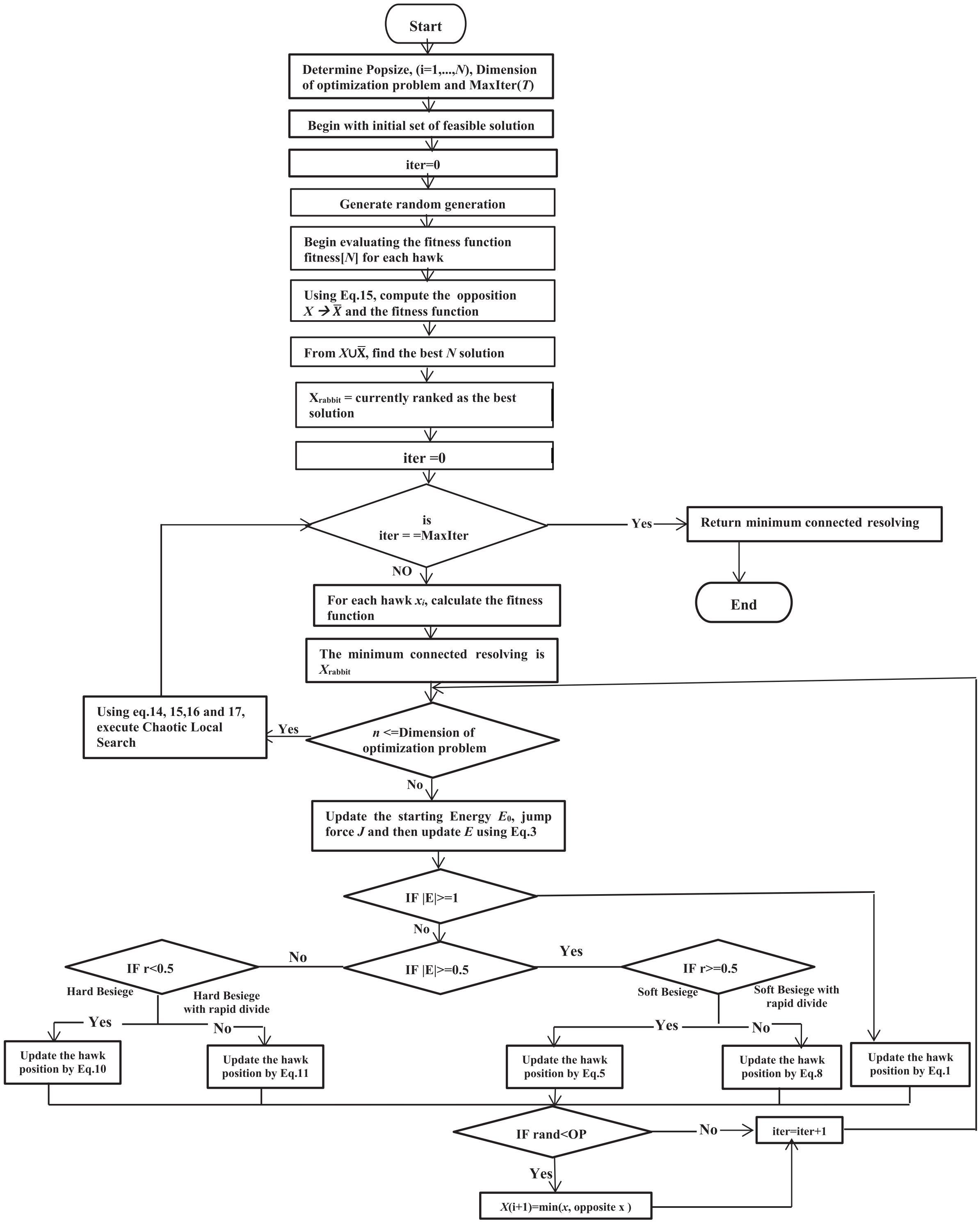

Figure 2: The flowchart of BEHHO

Table 2: Results on star graph

Instance	n	m	BHHO	t (s)	Iteration (generation)	BOHHO	t	Iteration	BCHHO	t	Iteration	BEHHO	t	Iteration
S_1	3	2	2	2.24	1	2	1.81	1	2	1.43	1	2	**1.06**	1
S_2	4	3	3	4.89	8	3	3.63	3	3	2.97	2	3	**1.59**	1
S_3	5	4	4	9.14	2	4	8.27	5	4	10.64	9	4	**3.22**	1
S_4	6	5	5	12.84	12	5	13.95	3	5	**12.28**	6	5	16.37	2
S_5	7	6	6	19.32	25	6	17.54	18	6	21.16	13	6	**11.85**	4
S_6	8	7	7	26.93	36	7	**14.09**	10	7	23.42	8	7	21.51	5
S_7	9	8	8	39.18	22	8	41.72	19	8	33.45	11	8	**19.04**	3
S_8	10	9	9	48.11	6	9	52.87	14	9	45.02	6	9	**26.74**	2
S_9	11	10	10	65.02	4	10	59.05	2	10	54.89	3	10	**48.16**	1
S_{10}	12	11	11	84.15	7	11	73.43	5	11	66.08	4	11	**31.27**	1

Table 3: Results on wheel graph

Instance	n	m	BHHO	t (s)	Iteration (generation)	BOHHO	t	Iteration	BCHHO	t	Iteration	BEHHO	t	Iteration
W_1	4	6	3	6.45	1	3	4.93	1	3	3.05	1	3	**1.84**	1
W_2	5	8	2	14.32	2	2	9.61	4	2	10.45	3	2	**5.03**	1
W_3	6	10	2	29.16	12	2	21.59	7	2	35.84	8	2	**13.97**	6
W_4	7	12	3	85.74	9	3	37.99	5	3	28.07	19	3	**24.53**	15
W_5	8	14	4	73.11	41	4	54.82	29	4	**36.14**	12	4	43.37	8
W_6	9	16	5	159.25	58	5	**31.07**	36	5	88.12	25	5	64.59	19
W_7	10	18	6	192.09	37	6	109.25	22	6	98.03	34	6	**57.75**	7
W_8	11	20	7	267.68	23	7	143.05	18	7	112.56	10	7	**81.13**	3
W_9	12	22	7	345.43	12	7	127.32	9	7	98.17	7	7	**65.94**	1
W_{10}	13	24	8	438.76	7	8	275.19	5	8	187.44	4	8	**94.21**	2

Table 4: Results on triangular snake graph

Instance	n	m	BHHO	t (s)	Iteration	BOHHO	t	Iteration	BCHHO	t	Iteration	BEHHO	t	Iteration
Δ1-*snake*	3	3	2	2.05	1	2	1.26	1	2	0.73	1	2	**0.08**	1
Δ2-*snake*	5	6	3	5.78	1	3	3.72	1	3	3.19	1	3	**1.95**	1
Δ3-*snake*	7	9	4	10.72	1	4	8.43	2	4	7.81	3	4	**4.29**	6
Δ4-*snake*	9	12	5	16.24	25	5	11.21	13	5	9. 43	17	5	**8.17**	10
Δ5-*snake*	11	15	6	23.64	54	6	**18.05**	38	6	21.65	9	6	19.99	12
Δ6-*snake*	13	18	7	28.25	42	7	24.53	68	7	36.03	24	7	**15.01**	19
Δ7-*snake*	15	21	8	37.95	1	8	32.17	9	8	**25.08**	7	8	28.43	3
Δ8-*snake*	17	24	9	56.62	40	9	48.98	23	9	52.49	18	9	**12.11**	11
Δ9-*snake*	19	27	10	64.99	1	10	39.46	3	10	37.11	9	10	**32.03**	5
Δ10-*snake*	21	30	11	82.46	5	11	66.75	11	11	56.45	10	11	**41.18**	8
Δ11-*snake*	23	33	12	116.63	19	12	**51.74**	27	12	97.08	15	12	55.92	3
Δ12-*snake*	25	36	13	94.64	1	13	85.63	1	13	83.29	2	13	**71.09**	9
Δ13-*snake*	27	39	14	111.91	1	14	99.01	5	14	79.52	3	14	**48.16**	2
Δ14-*snake*	29	42	15	87.62	1	15	115.08	3	15	94.17	4	15	**65.34**	1
Δ15-*snake*	31	45	16	233.88	4	16	148.16	2	16	107.32	2	16	**51.12**	1

Table 5: Results on double triangular snake graph

Instance	*n*	*m*	BHHO	*t* (s)	Iteration	BOHHO	*t*	Iteration	BCHHO	*t*	Iteration	BEHHO	*t*	Iteration
2Δ1-snake	4	5	2	5.55	1	2	3.17	1	2	2.46	1	2	**1.94**	1
2Δ2-snake	7	10	4	15.38	1	4	10.82	3	4	11.15	2	4	**6.43**	1
2Δ3-snake	10	15	5	24.24	29	5	19.21	3	5	17.26	9	5	**13.04**	3
2Δ4-*snake*	13	20	7	46.74	50	7	**12.15**	15	7	25.04	24	7	32.98	18
2Δ5-snake	16	25	9	77.04	46	9	51.98	37	9	47.63	13	9	**39.87**	29
2Δ6-snake	19	30	11	76.85	1	11	59.01	12	11	33.96	7	11	**18.25**	4
2Δ7-snake	22	35	13	127.07	1	13	96.13	9	13	85.17	15	13	**73.24**	7
2Δ8-snake	25	40	15	109.34	1	15	138.45	4	15	122.19	2	15	**94.58**	1
2Δ9-snake	28	45	17	175.32	1	17	146.99	8	17	135.03	5	17	**63.14**	3
2Δ10-snake	31	50	19	407.50	21	19	233.14	1	19	189.76	13	19	**116.41**	5
2Δ11-snake	34	55	21	423.13	7	21	185.92	5	21	225.42	3	21	**159.07**	2
2Δ12-snake	37	60	23	226.54	1	23	217.38	1	23	**158.03**	4	23	194.13	3
2Δ13-snake	40	65	25	406.53	1	25	295.26	4	25	205.74	1	25	**148.72**	1
2Δ14-snake	43	70	27	515.55	1	27	312.87	1	27	224.15	2	27	**197.94**	1
2Δ15-snake	46	75	29	536.67	1	29	243.38	1	29	189.37	1	29	**115.83**	1

Table 6: Results on linear kC_4-snake graph

Instance	*n*	*m*	BHHO	*t* (s)	Iteration	BOHHO	*t*	Iteration	BCHHO	*t*	Iteration	BCHHO	*t*	Iteration
*C*4-snake	4	4	2	3.24	1	2	2.7	1	2	2.09	1	2	**1.55**	1
2*C*4 -snake	7	8	4	9.15	1	4	6.04	2	4	4.92	3	4	**3.87**	2
3*C*4-snake	10	12	5	16.09	21	5	10.32	18	5	12.54	11	5	**7.16**	7
4*C*4- *snake*	13	16	7	36.16	7	7	25.84	5	7	19.12	8	7	**12.35**	2
5*C*4-snake	16	20	9	54.26	1	9	46.39	2	9	34.05	3	9	**27.82**	1
6*C*4-snake	19	24	11	77.19	1	11	21.46	14	11	48.11	5	11	**19.03**	1
7*C*4-snake	22	28	13	62.31	1	13	59.92	1	13	42.07	1	13	**34.61**	1
8*C*4-snake	25	32	15	96.52	1	15	83.87	1	15	91.15	1	15	**78.53**	1
9*C*4-snake	28	36	17	111.81	1	17	102.26	5	17	**68.04**	1	17	**92.07**	1
10*C*4-snake	31	40	19	152.82	1	19	133.91	1	19	117.54	3	19	**105.45**	1
11*C*4-snake	34	44	21	200.42	1	21	159.03	1	21	129.11	2	21	**87.94**	1
12*C*4- snake	37	48	23	217.87	1	23	178.52	7	23	145.08	1	23	**132.71**	1
13*C*4 -snake	40	52	25	317.15	1	25	259.07	1	25	213.44	1	25	**185.02**	1
14*C*4-snake	43	56	27	283.87	1	27	237.65	1	27	184.22	1	27	**145.23**	1
15 *C*4-snake	46	60	29	349.51	1	29	208.02	1	29	167.49	1	29	**106.98**	1

Tables 2–7 display the results for various graphs, which show that BEHHO can achieve the best optimal solution (known connected metric dimension) in a reasonable amount of time for the star graph, wheel graph, triangular snake graph, double triangular snake graph, linear kC_4-snake graph and (2, *n*) C_4-snake graph. It proves the correctness and superiority of BEHHO.

Experiments in this paper are performed on a subset of star graph instances with $n \leq 12$ and $m \leq 11$ in Table 2, wheel graph instances with $n \leq 13$ and $m \leq 24$ in Table 3, triangular snake graph instances with $n \leq 31$ and $m \leq 45$ in Table 4, double triangular snake graph instances with $n \leq 46$ and $m \leq 75$ in Table 5, linear *kC4*-snake graph instances with $n \leq 46$ and $m \leq 60$ in Table 6 and (2, *n*) C_4-snake graph instances with $n \leq 76$ and $m \leq 120$ in Table 7.

Table 7: Results on (2, *k*) C_4-snake graph

Instance	n	m	BHHO	t (s)	Iteration	BOHHO	t	iteration	BCHHO	t	Iteration	BEHHO	t (s)	Iteration
(2, 1) *C*4-snake	6	8	4	6.87	1	4	3.73	1	4	4.59	1	4	**2.21**	1
(2, 2) *C*4-snake	11	16	8	23.99	1	8	9.48	3	8	7.54	2	8	**5.13**	1
(2, 3) *C*4-snake	16	24	12	44.70	1	12	28.13	6	12	23.37	3	12	**11.58**	1
(2, 4) *C*4-snake	21	32	16	61.07	25	16	49. 04	17	16	**29.08**	10	16	34.81	7
(2, 5) *C*4-snake	26	40	20	99.64	54	20	83.15	28	20	68.12	35	20	**49.03**	21
(2, 6) *C*4-snake	31	48	24	156.23	42	24	118.79	13	24	125.36	27	24	**86.44**	34
(2, 7) *C*4-snake	36	56	28	213.83	1	28	171.15	5	28	**113.19**	3	28	124.65	1
(2, 8) *C*4-snake	41	64	32	227.20	40	32	201.63	26	32	217.12	15	32	**98.36**	19
(2, 9) *C*4-snake	46	72	36	390.79	1	36	288.35	1	36	178.59	1	36	**159.82**	1
(2, 10) *C*4-snake	51	80	40	374.97	5	40	328.06	8	40	256.92	4	40	**211.29**	3
(2, 11) *C*4-snake	56	88	44	529.43	19	44	391. 17	7	44	308.35	13	44	**252.06**	10
(2, 12) *C*4-snake	61	96	48	536.52	1	48	437.85	3	48	374.62	2	48	**293.71**	2
(2, 13) *C*4-snake	66	104	52	540.04	1	52	355.28	1	52	413.08	1	52	**308.99**	1
(2, 14) *C*4-snake	71	112	56	764.44	1	56	489.12	1	56	342.37	1	56	**255.18**	1
(2, 15) *C*4-snake	76	120	60	797.18	4	60	434.49	3	60	298.05	2	60	**221.54**	1

6 Conclusion

In this paper, we presented a binary enhanced Harris Hawk Optimization BEHHO algorithm for solving the connected metric dimension problem. The proposed algorithm is compared to classical HHO, chaotic local search HHO, opposition-based learning HHO. Comparisons were performed on the graphs: star graph, wheel graph, triangular snake graph, double triangular snake graph, linear kC_4-snake graph, and (2, n) C_4-snake graph. Computational results confirm the superiority of the proposed BEHHO algorithm for solving connected metric dimension problem.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

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

[2] P. J. Slater, "Leaves of trees," *Congressus Numerantium*, vol. 14, pp. 549–559, 1975.

[3] F. Harary and R. A. Melter, "On the metric dimension of a graph," *Ars Combinatoria*, vol. 2, pp. 191–195, 1976.

[4] M. Imran, F. Bashir, A. Q. Baig, A. U. H. Bokhary, A. Riasat *et al.,* "On metric dimension of flower graphs $f_{n\times m}$ and convex polytopes," *Utilitas Mathematica*, vol. 92, pp. 389–409, 2013.

[5] M. Munir, A. R. Nizami, Z. Iqbal and H. Saeed, "Metric dimension of the mobius ladder," *Ars Combinatoria*, vol. 135, pp. 249–256, 2017.

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

[7] 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.

[8] M. Imran, M. K. Siddiqui and R. Naeem, "On the metric dimension of generalized petersen multigraphs," *IEEE Access*, vol. 6, pp. 74328–74338, 2018.

[9] 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.

[10] M. Zayed, A. Ahmad, M. F. Nadeem and M. Azeem, "The comparative study of resolving parameters for a family of ladder networks," *AIMS Mathematics*, vol. 7, no. 9, pp. 16569–16589, 2022.

[11] A. Ahmad, M. Bača and S. Sultan, "Computing the metric dimension of kayak paddles graph and cycles with chord," *Proyecciones*, vol. 39, pp. 287–300, 2020.

[12] H. Alshehri, A. Ahmad, Y. Alqahtani and M. Azeem, "Vertex metric-based dimension of generalized perimantanes diamondoid structure," *IEEE Access*, vol. 10, pp. 43320–43326, 2022.

[13] B. Sooryanarayana, S. Kunikullaya and N. N. Swamy, "Metric dimension of generalized wheels," *Arab Journal of Mathematical Sciences*, vol. 25, no. 2, pp. 131–144, 2019.

[14] 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, 2022.

[15] A. N. Koam, A. Ahmad, M. S. Alatawi, M. F. Nadeem and M. Azeem, "Computation of metric-based resolvability of quartz without pendant nodes," *IEEE Access*, vol. 9, pp. 151834–151840, 2021.

[16] M. F. Nadeem, A. Shabbir and M. Azeem, "On metric dimension and fault tolerant metric dimension of some chemical structures," *Polycyclic Aromatic Compounds*, vol. 2, pp. 1–13, 2021.

[17] A. N. Koam, A. Ahmad, M. A. Asim and M. Azeem, "Computation of vertex and edge resolvability of benzenoid tripod structure," *Journal of King Saud University-Science*, vol. 34, no. 6, pp. 102–108, 2022.

[18] A. Al Khabyah, M. K. Jamil, A. N. Koam, A. Javed and M. Azeem, "Partition dimension of COVID antiviral drug structures," *Mathematical Biosciences and Engineering*, vol. 19, no. 10, pp. 10078–10095, 2022.

[19] M. Azeem, M. Imran and M. F. Nadeem, "Sharp bounds on partition dimension of hexagonal möbius ladder," *Journal of King Saud University-Science*, vol. 34, no. 2, pp. 101–107, 2022.

[20] 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.

[21] D. T. Murdiansyah and A. Adiwijaya, "Computing the metric dimension of hypercube graphs by particle swarm optimization algorithms," in *Proc. of ICSCDM*, Springer, Cham, pp. 171–178, 2016.

[22] 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.

[23] 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.

[24] J. C. Bednarz, "Cooperative hunting Harris' hawks," *Science*, vol. 239, no. 4847, pp. 1525–1527, 1988.

[25] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja *et al.,* "Harris hawks optimization: Algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[26] A. G. Hussien, L. Abualigah, R. Abu Zitar, F. A. Hashim, M. Amin *et al.,* "Recent advances in Harris hawks optimization: A comparative study and applications," *Electronics*, vol. 11, no. 12, pp. 19–68, 2022.

[27] A. G. Hussien and M. Amin, "A self-adaptive Harris hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection," *International Journal of Machine Learning and Cybernetics*, vol. 13, no. 2, pp. 309–336, 2022.

[28] H. Gezici and H. Livatyalı, "Chaotic Harris hawks optimization algorithm," *Journal of Computational Design and Engineering*, vol. 9, no. 1, pp. 216–245, 2022.

[29] Y. Zhang, R. Liu, X. Wang, H. Chen and C. Li, "Boosted binary Harris hawks optimizer and feature selection," *Engineering with Computers*, vol. 37, no. 4, pp. 3741–3770, 2021.

[30] H. R. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," in *Proc. of CIMCA-IAWTIC*, Vienna, Austria, pp. 695–701, 2005.

[31] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[32] L. dos Santos Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1905–1913, 2008.

[33] D. Jia, G. Zheng and M. K. Khan, "An effective memetic differential evolution algorithm based on chaotic local search," *Information Sciences*, vol. 181, no. 15, pp. 3175–3187, 2011.

[34] B. Crawford, R. Soto, G. Astorga, J. García, C. Castro *et al.,* "Putting continuous metaheuristics to work in binary search spaces," *Complexity*, vol. 2017, pp. 84–102, 2017.

[35] F. Afshinmanesh, A. Marandi and A. Rahimi-Kian, "A novel binary particle swarm optimization method using artificial immune system," in *Proc. of ICCT*, Eurocon, pp. 217–220, 2005.