Received: May 11, 2021. Revised: June 6, 2021.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Designing a Tool to Estimate Software Projects Based on The Swarm Intelligence

Shahbaa I. Khaleel[1]*

[1] *Software Department, College of Computer Science and Mathematics, Mosul University, Iraq*
* Corresponding author's Email: shahbaaibrkh@uomosul.edu.iq

Abstract: Estimating models in software engineering are used to estimate some important and future characteristics of the software project, such as estimating the developed project effort, and that failure in the program is mainly due to wrong project management practices, so estimating the software effort is a very important step in the software management process for large projects. In this research, a software estimation tool was built to find an efficient and accurate method for estimating the effort. The average COnstructive COst MOdel COCOMO was used, which is classified as one of the best traditional methods among arithmetic estimation models. Four methods of swarm intelligence were used, the first of which is the Glowing worm Swarm Optimization GSO method and the second is the Bird Swarm Algorithm BSA and the third is the first proposed method hybrid BSA-GSO Method1 BSA-GSOM1, where the GSO and BSA algorithms were hybridized, and the performance of the third method was improved to form the fourth method, represented by the second hybrid new method, which called hybrid BSA-GSO Method2 BSA-GSOM2. The new tool was implemented with all its methods on the NASA data set and satisfactory results were obtained by the first and second swarms intelligence, and excellent results were obtained in the first proposed method, but the results of the second proposed method were better and more accurate than the previous ones. many measurements of performance were used for all the methods, the second proposed method yielded the best results from everyone.

Keywords: Swarm intelligent, Software estimation, Glowworm swarm optimization, Bird swarm algorithm, COCOMO model.

1. Introduction

Software engineering science is one of the most important scientific topics today, and it employs a wide range of programming techniques with the goal of producing high-quality software to satisfy customers. The science of software engineering is primarily concerned with the early stages of program development [1]. Where the software has become an important feature and has been relied upon in most industries and laboratories because of the accuracy and speed with which it completes work, and the use of these programs is profitable for businesses because it reduces the time and costs incurred by teams of workers and experts [2].

Software engineering requires programmatic estimation, which is a crucial part of the planning and management of a software project. Software project estimating is not a fixed science, but rather an organized approach that may demonstrate the accuracy of historical data estimation [3]. One of the most difficult and critical software development activities is active programmatic estimating [4]. The project manager must explicitly describe and identify the software development cost estimate so that it can assess the project's progress in relation to the expected budget, anticipated scheduling and resource utilization. It was found that effort is the most important cost factor, which is then converted into cost. The main element affecting the estimation of the effort is the thousand line of code KLOC, which contains all the program instructions, data and official statements and is considered a unit of measurement for the size of the program or project [5]. Effort estimation is often used as a synonym for cost estimation. In software development, the predominant and controlling cost is the cost of hand

DOI: 10.22266/ijies2021.0831.46

labor. This is why estimating the software development effort as precisely as possible is critical. The software cost models provide an estimate of effort, cost, and schedule [6].

Swarm intelligence is a description for calculations and behaviors in solving distributed problems, inspired by biological examples shown by social insects such as ants, termites, and bees, and by swarm such as swarms of birds and fish [7]. In other words, swarm intelligence is based on the basic principles of behavior of natural systems that consist of several components that use local communication and distributed control. Thus, swarm intelligence is an efficient and practical model that simplifies distributed solutions to a wide variety of issues [8, 9]. Swarm intelligence is a self-organizing phenomenon in which a group of organisms and insects solve complex problems together. There is no individual in the group to lead or responsible for the outcome of the entire process [10]. Then the swarm collectively reach a solution. As for the continuous and harmonious communication and response among the group members, and compliance with the common laws, one of the fundamental issues of the success of swarm intelligence [11].

In this research, a new tool for software estimation was built that used two swarm intelligence algorithms were used to make the estimating process for the programmatic effort of the projects using a NASA data set, and the dataset consisted of three groups: NASA60, NASA63 and NASA93, where the glow worm algorithm was used to make the estimating process and gave good results, and then it was done. Using the optimal bird swarm algorithm to calculate the programmatic effort of the projects, and its results were better than the glowworm method. After that, the two previous swarm intelligence algorithms were hybridized to obtain a new hybrid method, BSA-GSOM1, which achieved better results than the previous two algorithms, and the fitness function was the Euclidean distance between estimated effort and actual effort. To improve performance of this proposed algorithm, two steps were added to it to produce a new method, BSA-GSOM2, with high performance and excellent accuracy. The two steps consisted of determining the values of a and b by relying on the value of the Effort Adjustment Factor EAF to improve the solution, as well as using a second fitness function which is all the measures adopted to find out the extent to which the estimated effort matches the values of the actual effort, i.e. this method has two fitness functions to improve its work performance and obtain the best results and solution optimum.

Section 2 includes the previous work, and then a detailed explanation of the COCOMO model, while the fourth section includes the measures adopted to evaluate the performance of the methods used, and the fifth section includes an explanation of the topic of swarm intelligence that includes the algorithms used in research in addition to the methods proposed here, followed by Section 6 results and discussion, and finally conclusions.

2. Previous work

Deepak Nandal and Om Prakash Sangwan hybridized the improved bat algorithm and gravitation search algorithm (GSA) to produce a new hybrid BATGSA algorithm for use in the estimate process for projects where NASA data were taken to implement the algorithm. The hybrid method gave better results than the previous methods. Although the hybrid method was improved the work, the mean absolute error values were also high, meaning that there was a difference between estimated effort and actual effort [12]. Researcher Bhaskar Marapelli introduced a working method for software estimating, proposing linear regression techniques for machine learning and closest K neighbors to predict program effort estimation using the COCOMO81, COCOMONasa and COCOMONasa2 data sets. According to the standards used, it became clear that the linear regression method was much better than the k-nearest neighbors method [13]. In reference [13], the best mean absolute errors MAE value was: 247.047 for the dataset NASA60, the MAE value: 874.477 for the dataset NASA63, and the MAE value: 430.727 for the dataset NASA93. In our research, the value of the MAE measure for the dataset NASA60 and for all research methods used was as follows: In the first method, GSO, the value of MAE was: 118.196, and in the second method, BSA, it was MAE: 87.594, and in the third method, BSA-GSOM1 MAE: 40.618, and in the fourth method, BSA-GSOM2 the value of MAE: 6.709. It was noted that it is the last method, and it is the second suggested method for our work, BSA-GSOM2. Its results were the best among all the methods used. Likewise, for the NASA63 dataset and the NASA93 dataset, the results of our methods were much better than the reference method [13], as shown in the tables in the section of Results and Discussion. Researcher Abdel Karim Baareh used two models of artificial neural networks to perform the estimating process of the software, namely the backpropagation algorithm and the radial base function algorithm. The goal is to reduce project risks that related to time, and increase the likelihood that all project requirements will be received on time. The two algorithms were implemented on a NASA dataset, and the backpropagation algorithm performed the best according to the metrics used [14]. In this paper, when implementing the two models from networks on NASA data with 60 projects, the best values obtained for the metrics were MAE: 48.757, root mean squared error RMSE value: 69.732, relative absolute error RAE value: 27.044, and root relative squared error RRSE: 25.752. In our research, much better values for these measures were obtained when implementing our methods on the same data set NASA60, where in the second proposed method, BSA-GSOM2, in our research the following values were obtained: MAE: 6.709, RMSE value: 13.753, RAE scale value: 1.594 and RRSE: 2.111 Where our research results are the best. The researcher Bilal Khan and others presented a proposed method for estimating the cost of the software, which is the flower pollination algorithm, in which a measure of the mean magnitude of relative error was used to evaluate the performance of the proposed model, and a dataset was adopted from the NASA software projects. The performance of the proposed method was much better than that of the COCOMO method [15]. In this paper, the MMRE value for the first data set (NASA60) 20.1620 and the second data set (NASA63) 16.0480 and the third data set (NASA93) 53.7338 were obtained, and this indicates that there are differences and not exactly match the values of the estimated effort with the actual effort. While in our research, the values of the mean magnitude of relative error MMRE measure were obtained as follows by using the first method GSO were 27.246, 39.861 and 53.763, respectively. When using the second method BSA the MMRE values were as follows 20.172, 27.645 and 34.191, and when using the first proposed method BSA-GSOM1 in our research, the values of MMRE were obtained as follows 10.161, 13.929 and 15.540. When using the second proposed method BSA-GSOM2 the measure values were as follows: 2.494, 2.178 and 2.436. It was noted that the second proposed method in our research was the best among the methods, and an estimated effort was obtained very near to actual effort of NASA projects.

3. The COCOMO model

Many mathematical models have been proposed that can be used to evaluate the effort, scheduling, and costs of the new project. All models are similar in principle to each other, but they use different values for their parameters [12]. The COCOMO is an experimental model which has been developed by gathering massive data for a large number of projects. Where these data are analyzed to discover the equations that most appropriate to notes reached by researchers. These equations relate to size of system, project characteristics, product, development team, and the effort required to development system. This is why COCOMO is used extensively by project managers [16]. The COCOMO is one of the best known traditional models of programmatic estimate theories and it was proposed by the scientist Boehm in 1981 [17] and it is also called COCOMO1 or COCOMO81 and it is a group of three typical levels: basic, Intermediate, and detailed. They all have a relationship between system size measured in KLOC and effort developed in person per month [18]. There are three patterns of project development [19]: Organic pattern where the project is developed in a well-known and stable environment similar to previous developed projects. Semi-detached pattern which is an intermediate state between organic and embedded. An embedded pattern here the project requires more creativity, has narrow themes, and inflexible interface requirements.

3.1 The intermediate COCOMO

The intermediate COCOMO modifies the basic equation used by the base COCOMO by using cost factors. These adjustments must be calculated for the characteristics of a particular project to deviate from the productivity of the nominal projects. It is based on 15 cost factors, and each factor has a specific effect, and this effect will take a certain numerical value called the effort multiplier, which in turn either increases or decreases the project effort. The equation for the intermediate COCOMO model is [20]:

$$Effort = a \times (KLOC)^b \times \prod_{i=1}^{15} EFM_i \qquad (1)$$

Where EFM_i is numerical value of cost factor i, $\prod_{i=1}^{15} EFM_i$ is the value of Effort Adjustment Factor which refer as *EAF.*

Table 1. Coefficients for intermediate COCOMO

Project mode	A	B
organic	3.2	1.05
Semi- detached	3.0	1.12
embedded	2.8	1.20

Table 2. Factors of effort multipliers

Number	factors	Description
1	rely	reliability of software
2	data	size of database
3	cplx	complexity of product
4	time	execution time
5	stor	storage constraint
6	virt	Virtual machine volatility
7	turn	turnaround time
8	acap	analyst capability
9	aexp	application experience
10	pcap	programmer capability
11	vexp	virtual machine experience
12	lexp	language experience
13	modp	modern programming practices
14	tool	software tool
15	sced	development schedule

The estimation of the normal effort in the intermediate COCOMO is obtained using the power function with three groups of {a, b} depending on the type of project as shown in Table 1 [21].

The 15 cost factors used in the COCOMO model to describe the software product under development are described in Table 2 [20].

These factors were classified into four categories: product factors, computer, personal factors, and project factors, as shown in Table 3. Each of the attributes of the fifteen cost factors in COCOMO has a rating scale and a set of effort multipliers that indicate what the numerical rated effort value corresponds to the specific rating (the fuzzy value) for each factor and by project. As the numerical values for each of these factors have been shown in the table below [15, 20].

Table 3. The numeric values of effort multipliers (vl: very low, l: low, n: nominal, h: high, vh: very high, exh: extra high)

Cost driver	**Rating**					
	vl	**l**	**N**	**h**	**vh**	**exh**
Product factors						
rely	0.75	0.88	1.00	1.15	1.40	
data		0.94	1.00	1.08	1.16	
cplx	0.70	0.85	1.00	1.15	1.30	1.65
Computer factors						
time			1.00	1.11	1.30	1.66
stor			1.00	1.06	1.21	1.56
virt		0.87	1.00	1.15	1.30	
turn		0.87	1.00	1.07	1.15	
Personal factors						
acap	1.46	1.19	1.00	0.86	0.71	
aexp	1.29	1.13	1.00	0.91	0.82	
pcap	1.42	1.17	1.00	0.86	0.70	
vexp	1.21	1.10	1.00	0.90		
lexp	1.14	1.07	1.00	0.95		
Project factors						
modp	1.24	1.10	1.00	0.91	0.82	
tool	1.24	1.10	1.00	0.91	0.83	
sced	1.23	1.08	1.00	1.04	1.10	

4. Evaluation criteria

For the purpose of making a comparison between the methods used in assessing the effort in this research, as well as in order to use these measures as a function of fitness for research methods that used here, to improve their performance and reach the best solution. The following metrics were used:

Mean squared error (MSE): This scale calculates the average square error between the actual value of the project effort and the resulting effort value, i.e. the estimated effort value from the research methods, as in the following equation [14].

$$MSE = \frac{1}{NP} \sum_{p=1}^{NP} \left(act\ effort_p - \ est\ effort_p \right)^2 \quad (2)$$

Where NP represents number of projects that used, $act\ effort_p$ is actual effort, and $est\ effort_p$ is estimate effort.

Relative absolute error (RAE): [14]

$$RAE = \frac{\sum_{p=1}^{NP} |est\ effort_p - act\ effort_p|}{\sum_{p}^{NP} |act\ effort_p - act\ mean|} \quad (3)$$

$act\ mean$ is mean of all actual effort.

Mean of absolute errors (MAE): [13]

$$MAE = \frac{1}{NP} \sum_{p=1}^{NP} \left| act\ effort_p - \ est\ effort_p \right| \quad (4)$$

and the measure of complementary accuracy is:[22]

$$PRED = \frac{1}{NP} \sum_{p=1}^{NP} \begin{cases} 1 \,, \ if\ MRE \leq m \\ 0 \,, otherwise \end{cases} \quad (5)$$

Where MRE is magnitude of relative error as follow, and m equal 0.25: [22]

$$MRE = \left| \frac{act\ effort_p - est\ effort_p}{act\ effort_p} \right| \quad (6)$$

Root mean squared error (RMSE): [23]

Received: May 11, 2021. Revised: June 6, 2021.

$$RMSE = \sqrt{\frac{1}{NP}\sum_{p=1}^{NP}\left(act\ effort_p - est\ effort_p\right)^2} \quad (7)$$

Mean magnitude of relative error (MMRE): [24]

$$MMRE = \frac{1}{NP}\sum_{p=1}^{NP}\frac{act\ effort_p - est\ effort_p}{act\ effort_p} \times 100 \quad (8)$$

Root relative squared error (RRSE): [13]

$$RRSE = \sqrt{\frac{\sum_{p=1}^{NP}\left(est\ effort_p - act\ effort_p\right)^2}{\sum_{p=1}^{NP}\left(act\ effort_p - act\ mean\right)^2}} \quad (9)$$

5. Swarm intelligence

More than fifty years ago, biologists have proven the existence of multiple types of intelligence emanating from the community of insects, birds, fish or mammals [25]. Swarm intelligence is an artificial intelligence technology that involves studying group behaviors in decentralized systems. It represents the modeling and application of a set of interactions discovered in the insect population [26]. Swarm intelligence describes the ability of groups of animals and insects to exhibit highly regular behaviors to solve complex problems that allow the group as a whole to accomplish tasks that exceed the ability of the single individual who composes the group [27]. This natural phenomenon is the inspiration for swarm intelligence systems, a class of algorithms that use and take advantage of naturally occurring swarm models and templates to solve computational problems [28]. Computationally, it emulates the naturally emerging behavior of a community of insects or swarms to simplify the design of distributed solutions to complex problems. Emergent behavior refers to the way complex systems or models arise from the multiplication of relatively simple interactions [29].

5.1 Glowworm swarm optimization (GSO)

The Glowworm swarm optimization (GSO) algorithm is one of the algorithms for swarm intelligence, discovered by researchers K. Krishnanand and D. Ghose. This algorithm simulates the flashing behavior of glow worms. Each glow worm has an amount of luminescence and is called luciferin [8]. At the beginning of the algorithm and within the general search area, the population of the glow worm is assigned and then a group of glow worms is randomly selected from within the entire population, in addition to determining the amount of luciferin for all glow worms. Hence, in each cycle of the algorithm there are three stages: luciferin refresh, locomotion, and juxtaposition as shown in the Fig. 1 [10].

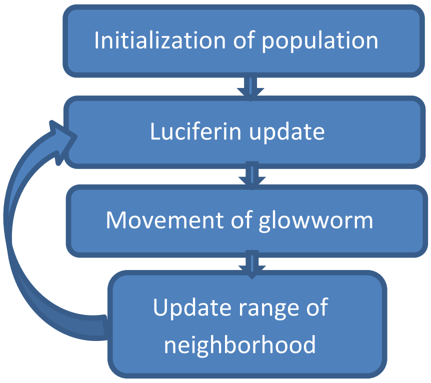

Figure. 1 Stages of GSO algorithm

Glowworm swarm algorithm: an algorithm whose principle is based on the behavior of luminous worms; Whereas, the light worm that produces a lot of light (high luciferin) is closer to an actual location and has a high objective value, and the worms move to the neighboring worm that has the highest light. Where the value of luciferin is modified depending on the location of the glowing worm, since its location is an important function value for adjusting the amount of luciferin for each glowing worm, and the glowing worm adds an amount of luciferin to its previous level of luciferin in a value proportional to the fitness of its current location within the target function space. Likewise, the value of luciferin is simulated over time by subtracting a small value from it [30]. The level of luciferin is adjusted using the following equation:

$$lu_j^t = (1 - ld)lu_j^{t-1} + \gamma fit(pos_j^t) \quad (10)$$

Where lu_j^{t-1} represents previous level of luciferin for jth glowworm; ld is positive value that represent luciferin decay; γ is enhancement fraction of luciferin, t is current iteration $fit(pos_j^t)$ is objective function of jth glowworm at current position pos_j.

After that, the glowworm explores the surrounding area to extract the neighboring worms that have the highest level of luciferin, according to the following equation [31]:

$$z \in NS_j^t \quad if\ dis_{jz} < rl_j^t\ and\ lu_z^t > lu_j^t \quad (11)$$

Received: May 11, 2021. Revised: June 6, 2021.

Where z represents the one of closer glowworms to jth glowworm, dis is the Euclidean distance between jth glowworm and z glowworm, NS_j^t is neighborhood set, rl_j^t is local decision range of jth glowworm.

Then the best neighbor is determined from among this neighborhood group, by calculating the probabilities of all neighbors and this is done using the following equation [32].

$$prob_{jz} = \frac{lu_z^t - lu_j^t}{\sum_{k \in NS_j^t} lu_k^t - lu_j^t} \qquad (12)$$

Where z represents one of neighborhood set NS_j^t of jth glowworm. After that, each glowworm determines the direction of movement by using the roulette wheel, as the glowworm with the highest probability has the greatest opportunity to choose from the neighborhood group. The location (i.e position) of the glowworm is then modified depending on the location of the glowworm selected from the group. This is done using the following equation [33]:

$$pos_j^t = pos_j^{t-1} + s\frac{pos_z^t - pos_j^t}{dis_{jz}} \qquad (13)$$

Finally, rl_j^t is updated by using the following equation:

$$rl_j^t = min\{rds, max[0, rl_j^{t-1} + \beta(cf - |NS_j^{t-1}|)]\} \qquad (14)$$

Where rds is a constant of radial of sensor range, cf is constant factor that used to restrict the size of neighborhood set [8].

In this research, the glow worm algorithm was used to accurately perform the estimation process of the software, due to the ability of the swarm algorithm to model complex groups of relationships between the dependency variables represented in effort and non-dependency variables represented by cost factors, which made it a potential tool for guessing. The Euclidean distance law between estimated effort and actual effort of the software was used in the calculation of the fitness function for this algorithm as in the following equation, which in turn depended on the value of the EAF that was derived from the 15 cost factors of each project.

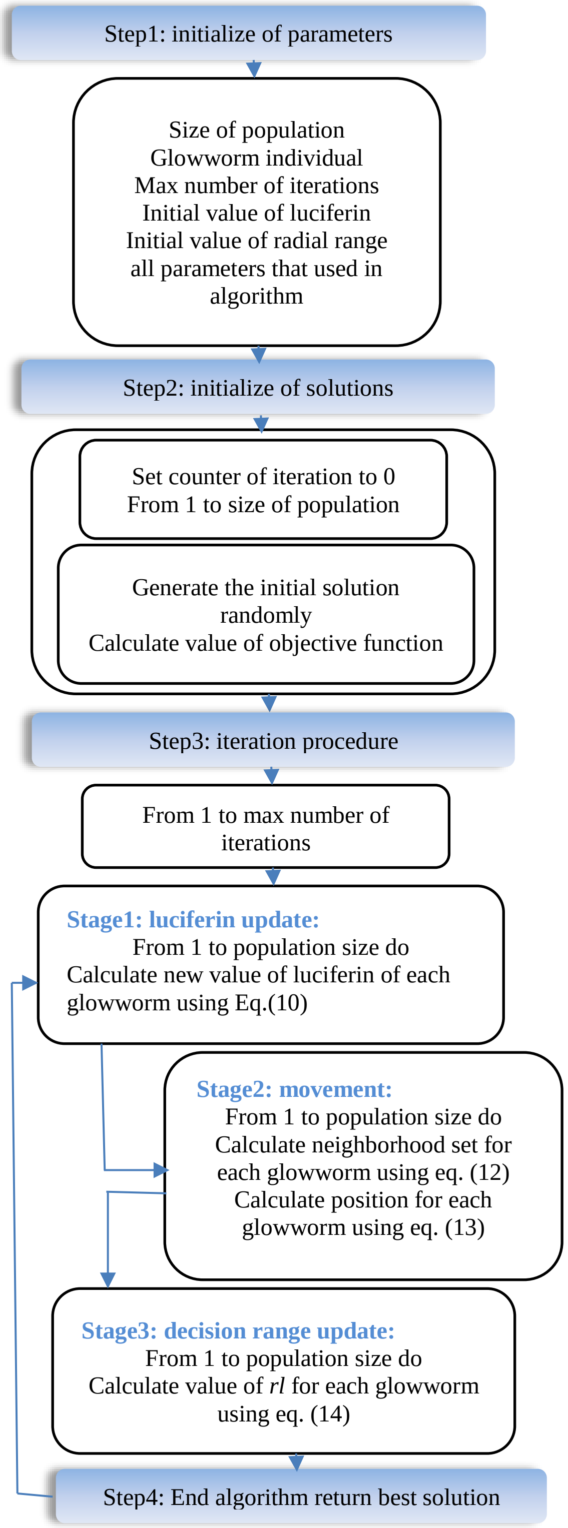
Step1: initialize of parameters
Size of population
Glowworm individual
Max number of iterations
Initial value of luciferin
Initial value of radial range
all parameters that used in algorithm
Step2: initialize of solutions
Set counter of iteration to 0
From 1 to size of population
Generate the initial solution randomly
Calculate value of objective function
Step3: iteration procedure
From 1 to max number of iterations
Stage1: luciferin update:
From 1 to population size do
Calculate new value of luciferin of each glowworm using Eq.(10)
Stage2: movement:
From 1 to population size do
Calculate neighborhood set for each glowworm using eq. (12)
Calculate position for each glowworm using eq. (13)
Stage3: decision range update:
From 1 to population size do
Calculate value of rl for each glowworm using eq. (14)
Step4: End algorithm return best solution

Figure. 2 Flowchart of Glowworm swarm optimization

In this research, the glow worm algorithm was used to accurately perform the estimation process of the software, due to the ability of the swarm algorithm to model complex groups of relationships between the dependency variables represented in effort and non-dependency variables represented by cost factors, which made it a potential tool for guessing. The Euclidean distance law between estimated effort and actual effort of the software was

used in calculation of the fitness function for this algorithm as in the following equation, which in turn depended on the value of the EAF that was derived from the 15 cost factors of each project.

$$fit = |act\ effort - est\ effort| \quad (15)$$

Where estimate effort is calculated as follow:

$$est\ effort = KLOC\ \times EAF \quad (16)$$

The algorithm is executed for a certain number of iterations to obtain the best fitness value as the best solution for estimating the development effort for each project based on the effort adjustment factor.

5.2 Bird swarm algorithm (BSA)

The bird swarm algorithm belongs to the class of swarm intelligence techniques used in solving optimization problems. It is one of the very recent versions, the bird swarm model helps to find the best areas of complex search spaces through the interaction of individuals, and the idea of this algorithm was inspired by the social behavior of swarm of birds. Most birds have three types of behaviors: first foraging, second vigilance behavior, and third flight behavior [34].

Foraging behavior: All birds in swarm depend in searching for food on their previous experience, and the experience of the swarm which belonging to them, represented by the previous location in which the bird is as well as the best previous location of the swarm. This behavior is mathematically expressed as follows [35, 36]:

$$Loc_{ij}^{t+1} = Loc_{ij}^{t} + \left(P_{ij} - Loc_{ij}^{t}\right) \times Co \times R + (g_j - Loc_{ij}^{t}) \times Sa \times R \quad (17)$$

Where R represents random number the value of it between (0,1). Sa is social accelerated, Co is cognitive accelerate factors. The best previous position of ith bird is represented by P_{ij} , and g_j represent best previous position of swarm.

Vigilance behavior: Birds in the swarm are vigilant as to protect themselves from predation, each bird competes with rest of birds in swarm for a middle location of the swarm. This behavior can be represented mathematically as follows [36]:

$$Loc_{ij}^{t+1} = Loc_{ij}^{t} + F1\left(m_j - Loc_{ij}^{t}\right) \times R + F2(P_{kj} - Loc_{ij}^{t}) \times R1 \quad (18)$$

$$F1 = f1 \times exp\left(-\frac{Pfit_i}{sumfit+\varepsilon} \times N\right) \quad (19)$$

$$F2 = f2 \times exp\left(\left(\frac{Pfit_i - Pfit_k}{|Pfit_k - Pfit_i| + \varepsilon}\right)\frac{N \times Pfit_k}{sumfit+\varepsilon}\right) \quad (20)$$

Where k is positive number that selected randomly between 1 and N, and k not equal i, also $f1$ and $f2$ positive number [0 2]. $Pfit_i$ is the best fitness of i^{th} bird, $sumfit$ represent sum of best fitness in swarm, ε constant to avoid zero division, finally m_j is average position of jth bird in swarm.

Flight behavior: when searching for food, the birds travel from one position to another due to several reasons, including the threat of predation. Upon reaching another location, they search for food. Some of the birds search for food and are called producers, the rest of swarm birds, who are called scroungers, are feed from food spots that producers find. To mathematically represent the behavior of the producer and the scroungers as follows [37]:

$$Loc_{ij}^{t+1} = Loc_{ij}^{t} + Rn \times Loc_{ij}^{t} \quad (21)$$

$$Loc_{ij}^{t+1} = Loc_{ij}^{t} + \left(Loc_{kj}^{t} - Loc_{ij}^{t}\right) \times Fsp \times R \quad (22)$$

Where Rn is random value with 0,1, $Fsp \in [0\ 2]$ which means during searching food the scrounger follow the producer.

The main goal of a good estimating system for projects is to be accurate guess, and as near to actual effort, to overcome the failure of projects. because the estimating for projects are important to success of projects, if it is completed accurately and within the specified measurements. In this research, bird's algorithm was used because of its power in finding the optimal solution to complex problems. As in the glow worm algorithm, the EAF values were based on the 15 cost factors of each project. The law of the Euclidean distance between estimated effort and real effort of the software was used in calculation of the fitness function for this algorithm as Eqs. (15, 16) that shown below:

$$fit = |act\ effort - est\ effort|$$

Where estimate effort is calculated as follow:

$$est\ effort = KLOC\ \times EAF$$

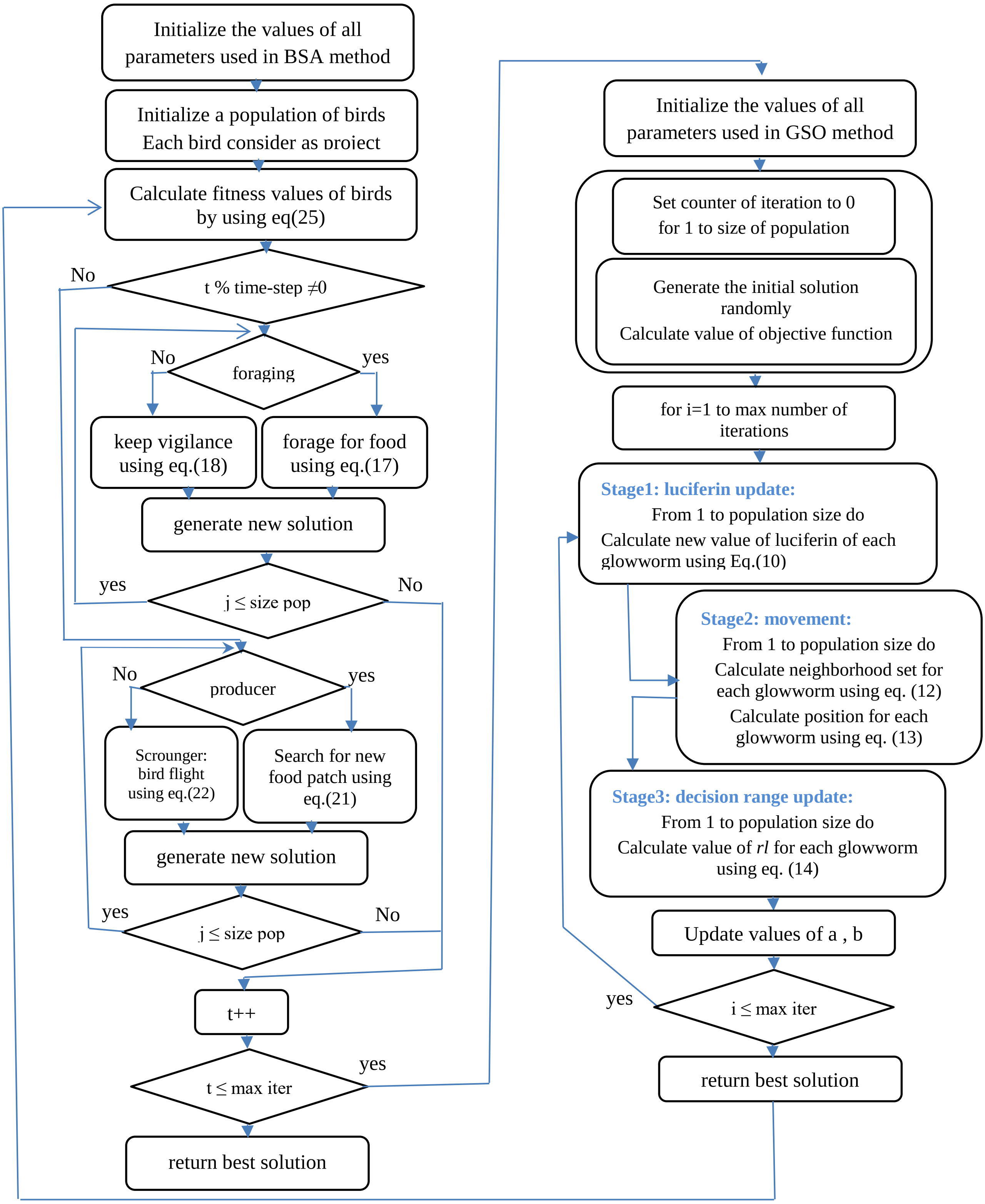
Initialize the values of all parameters used in BSA method
Initialize a population of birds Each bird consider as project
Calculate fitness values of birds by using eq(25)
No
t % time-step ≠0
No
foraging
yes
keep vigilance using eq.(18)
forage for food using eq.(17)
generate new solution
yes
j ≤ size pop
No
No
producer
yes
Scrounger: bird flight using eq.(22)
Search for new food patch using eq.(21)
generate new solution
yes
j ≤ size pop
No
t++
yes
t ≤ max iter
return best solution
Initialize the values of all parameters used in GSO method
Set counter of iteration to 0 for 1 to size of population
Generate the initial solution randomly Calculate value of objective function
for i=1 to max number of iterations
Stage1: luciferin update:
From 1 to population size do
Calculate new value of luciferin of each glowworm using Eq.(10)
Stage2: movement:
From 1 to population size do
Calculate neighborhood set for each glowworm using eq. (12)
Calculate position for each glowworm using eq. (13)
Stage3: decision range update:
From 1 to population size do
Calculate value of rl for each glowworm using eq. (14)
Update values of a , b
yes
i ≤ max iter
return best solution

Figure. 3 Flowchart of proposed method hybrid BSA-GSOM1

Where the algorithm was implemented for a certain number of iterations to obtain the best fitness value as the best solution to guess the software development effort for each project depending on the effort modulation factor. From the results obtained, it was noted that this algorithm gave much better results than the previous method represented by the glowing worm method.

5.3 Hybrid BSA-GSO method1

Due to the success of the bird-swarm optimization and glow-worm optimization

Received: May 11, 2021. Revised: June 6, 2021.

algorithms, obtaining an estimate of software effort is somewhat close to actual effort. In this method, the two algorithms were hybridized, the optimized bird swarm algorithm and the glowing worm algorithm, to obtain better results than the previous two methods. Whereas, the law of calculating the estimated effort was used by adopting the COCOMO equation as an input of objective function of the new method. The new proposed method consists of two stages, first obtaining the best values of the two variables a and b from the glow worm algorithm, and then obtaining the best solution, i.e. the best estimate of the project development effort from the optimal bird swarm method. As shown in the following flowchart:

Where at the beginning random values were given for the variables a, b and the range of [2.8-3.2] for the variable a and the range of values [1.05-1.20] for the variable b, where it was observed when implementing the proposed algorithm which is a hybridization between the two algorithms and the objective function was as follows, that results were obtained Better, and the values of the variables a, b are modified under the glow worm method to obtain the best values of them for use in the bird's algorithm.

$$fit = |act\ effort - est\ effort| \tag{25}$$

Where estimate effort is calculated as follow:

$$est\ effort = a\ \times KLOC^{b}\ \times EAF \tag{26}$$

The best results were obtained for the estimated effort, when compared to the two previous methods, it was noted that its results were much better than its predecessors, and even when calculating the measures, the proposed method had the best results.

5.4 Hybrid BSA-GSO method2

Although satisfactory and fulfilling results were obtained by the proposed hybrid method, the measures were used for the project's estimated effort to improve the work of the proposed algorithm by considering the measures as a function of fitness within the algorithm. Whereas, this method is the same as the steps of the previous proposed method with adding two improvements, the first of which is when calculating the estimated effort depending on the effort modulation factor and the KLOC, the values of a, b were determined depending on the value of the EAF, where the project whose EAF value is 1.0 is considered semi- detached As for the project whose EAF value is greater than 1.0, it is considered to be subordinate to the organic type, and which is less than 1.0, it is considered to be part of the embedded type.

Secondly, all the scales mentioned in equations 2-9 were used here as a second fitness function, as the proposed method of hybridization continues until the best values of the second fitness function are obtained. Thus, we guarantee obtaining estimated effort values identical to or close to the actual project effort. When implementing this method, the best results were obtained, which are explained in the tables.

6. Results and discussion

In this research, the programmatic effort spent on each project was properly determined using a historical database prepared for past implemented and completed programs, and this data was provided by the NASA satellite agency. This database has been used because collecting such data is a very difficult process, and it is known that it is very difficult to control software projects and therefore companies are reluctant to expose their software development records. The first data set is NASA60, which is derived from 60 projects from NASA and various project centers, and the second data set is NASA63, drawn from 63 projects, and the third data set, NASA93, was drawn from 93 program projects. And each of these projects contains 15 values that represent values of cost factors shown in Table 2, where each of these fuzzy values has a numerical value corresponding to it, Therefore, based on its own cost factor, as shown in Table 3, where the fuzzy values were converted into a numerical one that was used by research here, in addition to size of project that measured in KLOC, and actual effort of project.

After making many attempts to define the characteristics that will be chosen as an input to the algorithms and research methods here, where the attempts to enter all the data (15 cost factors and the KLOC), or choose the factors most influencing the project to using them as input to the swarm algorithms here, in addition to the kloc, and this is what you do The majority of research, and many other attempts to select the data that give the best software effort. After conducting a study on these data and due to the strong convergence between the cost factor values, it was suggested in this research to use one value instead of the 15 factors, and this value is the multiplication of these factors, which is the EAF value, which used its value and the KLOC value as an input to the fitness function of the research methods. In this way, all factor values will be taken into account, and none of the factors affecting the program project will be ignored.

The following table shows the results of the four methods used in the research, which are the glowing worm method, the optimal bird swarm algorithm, and the first proposed method, which is a hybrid between the glow worm algorithm and the bird algorithm using the fitness function represented by the Euclidean distance between the estimated effort and the actual effort of the project, and the second proposed method represented by the same hybridization steps of the first method, in addition to the use of a second fitness function to improve work and reach the optimal solution, in addition to the COCOMO method. When applying all the research methods to the first data set, NASA60, it was noted from Table 4 that the second proposed hybrid method gave the best results by estimating the effort compared to other methods, as the estimated project effort values were very close to the actual effort of the project

And when all the research methods are applied to the second data set, represented by NASA63, it became clear that the second proposed method hybrid BSA-GSO method2 has the best results for the values of estimating effort, as shown in Table 5.

Finally, all research methods were applied to the third data set, represented by NASA93, and it became clear that the second proposed method had the best results for the values of estimating effort as shown in Table 6.

Table 4. The estimate effort using all methods for data set NASA60

Project no.	Actual effort	Estimate by COCOMO	GSO	BSA	BSA-GSOM1	BSA-GSOM2
1	278	364.4321	332.092	301.922	287.011	281.005
10	60	58.37976	58.921	58.998	59.503	59.907
20	239	286.1325	279.005	267.708	251.922	242.029
30	8.4	6.336011	6.989	7.022	7.891	8.014
40	60	91.99345	87.044	81.011	74.006	67.035
50	750	465.4028	489.092	591.023	691.901	745.944
60	155	99.5439	112.068	124.929	139.657	151.904

Table 5. The estimate effort using all methods for NASA63

Project no.	Actual effort	Estimate by COCOMO	GSO	BSA	BSA-GSOM1	BSA-GSOM2
1	2040	1047.997	1578.923	1721.407	1897.989	2038.925
10	321	189.9992	207.923	276.552	299.589	320.298
20	6400	4096.45	5012.074	5245.982	5978.376	6399.923
30	5.9	6.31997	6.201	6.015	6.001	5.987
40	8	10.09683	9.992	9.367	8.927	8.409
50	176	115.2676	132.044	141.298	161.209	175.001
60	57	50.11726	51.902	53.007	55.092	56.903

Table 6. The estimate effort using all methods for NASA93

Project no.	Actual effort	Estimate by COCOMO	GSO	BSA	BSA-GSOM1	BSA-GSOM2
1	117.6	122.143	121.038	120.759	118.972	117.905
10	72	39.1194	49.929	57.298	63.608	70.692
20	60	173.2488	161.021	127.294	87.383	65.091
30	62	57.85346	58.094	59.506	60.991	61.893
40	114	75.07525	78.034	89.084	101.009	112.049
50	571.4	431.3985	463.565	497.308	511.077	567.598
60	720	1209.537	1045.005	904.806	801.038	731.009
70	432	858.176	804.002	678.094	501.207	445.002
80	703	409.5006	523.809	631.003	695.008	701.507
90	8211	2714.443	3293.393	4167.044	6497.494	8205.398

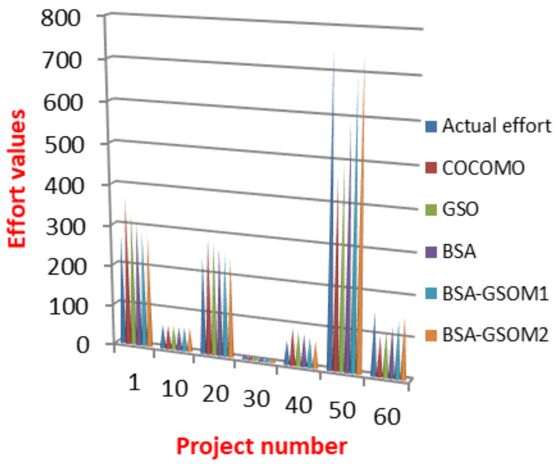

Figure. 4 Actual and estimated effort of all methods for dataset1

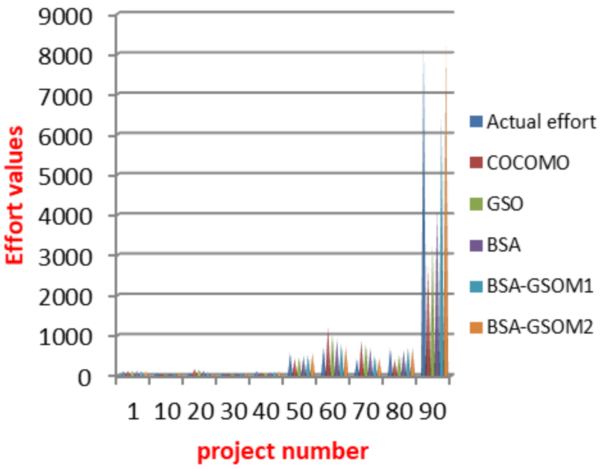

Figure. 6 Actual and estimated effort of all methods for dataset3

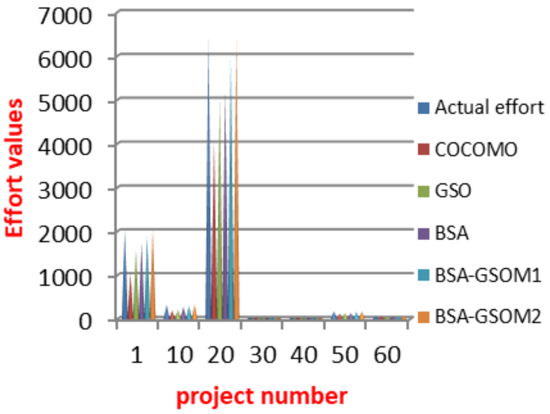

Figure. 5 Actual and estimated effort of all methods for dataset2

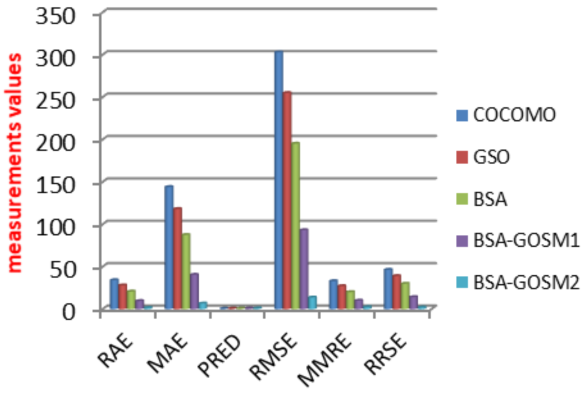

Figure. 7 The value of measurements of all methods for dataset1

From Fig. 4, the strength and success of the swarm intelligence methods used in the research is evident, as it was noted that the values of the estimated effort produced from the four intelligent methods of dataset1 are close to the value of the actual effort, and these methods have much better results than the traditional COCOMO method. The methods suggested here were the best, and we see that the BSA-GSOM2 method was the best of all.

It is evident from Fig. 5 that the second proposed method BSA-GSOM2 is the best of all in obtaining the values of the estimated effort for the second data set NASA63.

When applying the research methods to the third data set, NASA93, it was noticed that the performance of the proposed methods was better, as the second method had better results and high accuracy as shown in Fig. 6.

For the purpose of making a comparison between results of these estimates, a table was made of measures values resulting from all methods that used in the research and for all data sets. Table 7 shows the ratios of these measures on the first data set, NASA60, and it is clear from the table that the last method, which is BSA-GSOM2, achieved the best results for all measures. Regarding the MMRE measure, which is the lower its percentage, the estimate of the algorithm that used is better, as it was found that the BSA-GSOM2 method was the best among the methods that used to estimating effort of this measure, followed by the BSA-GSOM1 method, then BSA and then GSO. We note that COCOMO model achieved highest percentage, which makes it the worst for this measure. Thus, for the rest of the measures, the lower their value the better, except for the PRED measure, when the value is higher, then the method used becomes the best, and we note that the second proposed method is also the best with this measure.

Fig. 7 shows the chart of the standards values for all methods, through which it is clear that the BSA-GSOM2 method is the best.

The same applies to the second data set, NASA63, where the results of the metrics used for

Received: May 11, 2021. Revised: June 6, 2021.

Table. 7 The values of measurements of all methods in dataset1 NASA60

Methods	**measurements**						
	MSE	**RAE**	**MAE**	**PRED**	**RMSE**	**MMRE**	**RRSE**
COCOMO	91754.600	34.238	144.104	0.483	302.910	33.159	46.497
GSO	65036.76	28.083	118.196	0.633	255.023	27.246	39.146
BSA	38130.52	20.812	87.594	0.733	195.270	20.172	29.974
BSA-GOSM1	8696.722	9.651	40.618	0.933	93.256	10.161	14.315
BSA-GOSM2	189.149	1.594	6.709	1	13.753	2.494	2.111

Table 8. The values of measurements of all methods in dataset2 NASA63

Methods	**measurements**						
	MSE	**RAE**	**MAE**	**PRED**	**RMSE**	**MMRE**	**RRSE**
COCOMO	1137652.036	39.525	352.452	0.317	1066.608	50.152	59.024
GSO	489903.330	26.739	238.440	0.492	699.931	39.861	38.733
BSA	159385.947	16.284	145.204	0.698	399.232	27.645	22.093
BSA-GOSM1	10686.738	5.080	45.302	0.873	103.377	13.929	5.721
BSA-GOSM2	660.288	0.552	4.921	1	25.696	2.178	1.422

Table 9. The values of measurements of all methods in dataset3 NASA93

Methods	**measurements**						
	MSE	**RAE**	**MAE**	**PRED**	**RMSE**	**MMRE**	**RRSE**
COCOMO	2046885.911	68.202	435.986	0.387	1430.694	64.914	126.632
GSO	1042649.805	53.349	341.038	0.452	1021.102	53.763	90.379
BSA	390410.7	33.887	216.625	0.624	624.828	34.191	55.304
BSA-GOSM1	71030.3	13.842	88.486	0.871	266.515	15.540	23.589
BSA-GOSM2	664.333	1.412	9.025	0.989	25.775	2.436	2.281

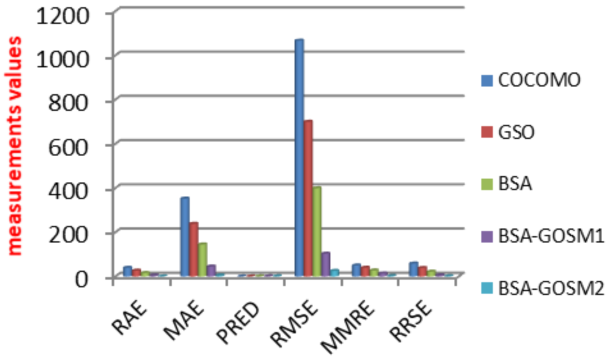

Figure. 8 The value of measurements of all methods for dataset2

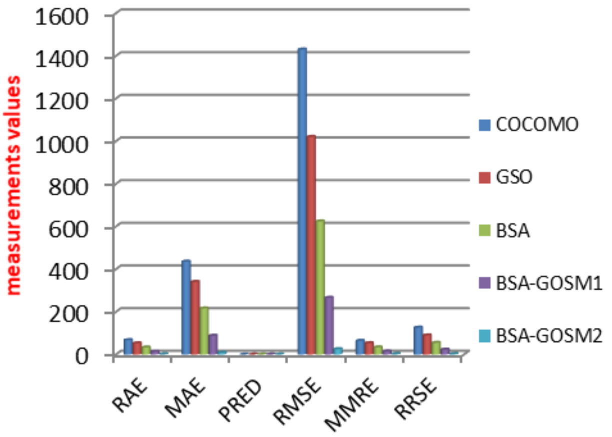

Figure. 9 The value of measurements of all methods for dataset3

all research methods were obtained, as shown in Table 8.

Fig. 8 shows that the BSA-GSOM2 method is the best when implemented on a NASA63 dataset.

Finally, the Table 9 shows the efficiency and accuracy of the proposed methods when implementing them on the third data set, and it is clear that the BSA-GSOM2 method is the best of all methods, despite the efficiency of the first proposed method, BSA-GSOM1, as well as the BSA method and the GSO method, where all of them achieved much better results than the conventional method COCOMO this is illustrated by the measures' values.

Finally, in Fig. 9 all the metrics used for all methods are clear, when implemented on the NASA93 dataset, then the BSA-GSOM2 method is the best.

We conclude from the above tables and results that the swarm intelligence methods used by research to estimate the effort of the projects had much better results than the traditional COCOMO model, where the results of the first GSO method were better than the traditional method, and then the second method, BSA, had better results than the previous one. Then hybridize the first and second method to obtain the proposed method of BSA-GSOM1, where when implementing it on NASA data sets, better results were obtained than the two previous methods, and to improve its performance, it was developed and a second proposed method was obtained, BSA-GSOM2, whose results were excellent, where an estimated effort close to or equal to the actual effort was obtained.

When comparing this work (our work) with previous works, it was noticed that our research methods were better in finding the estimated effort. Where in reference [12] the projected estimating values are as follows: from project 1,10,20,30,40,50,60 the estimated effort are 287.204, 42.182, 185.565, 4.648, 110.885, 444.591, 61.845 respectively. Where in our research the estimating effort of the same projects are 281.005, 59.907, 242.029, 8.014, 67.035, 745.944, 151.904 respectively and the actual effort as 278, 60, 239, 8.4, 60, 750, 155. We note that our research methods were more efficient and accurate than others in finding the values of the estimated effort.

In reference [13], the best mean absolute errors MAE value was: 247.047 for the dataset NASA60, the MAE value: 874.477 for the dataset NASA63, and the MAE value: 430.727 for the dataset NASA93. In our research, the value of the MAE measure for the dataset NASA60 and for all research methods used was as follows: In the first method, GSO, the value of MAE was: 118.196, and in the second method, BSA, it was MAE: 87.594, and in the third method, BSA-GSOM1 MAE: 40.618, and in the fourth method, BSA-GSOM2 the value of MAE: 6.709. It was noted that it is the last method, and it is the second suggested method for our work, BSA-GSOM2, its results were the best among all the methods used. Likewise, for the NASA63 dataset and the NASA93 dataset, the results of our methods were much better than the reference method [13], as shown in Tables 8 and 9.

In reference [14], when implementing the two models from networks on NASA data with 60 projects, the best values obtained for the metrics were MAE: 48.757, root mean squared error RMSE value: 69.732, relative absolute error RAE value: 27.044, and root relative squared error RRSE: 25.752. In our research, much better values for these measures were obtained when implementing our methods on the same data set NASA60, where in the second proposed method, BSA-GSOM2, in our research the following values were obtained: MAE: 6.709, RMSE value: 13.753, RAE scale value: 1.594 and RRSE: 2.111 Where our research results are the best.

In reference [15], the MMRE value for the first data set (NASA60) 20.1620 and the second data set (NASA63) 16.0480 and the third data set (NASA93) 53.7338 were obtained, and this indicates that there are differences and not exactly match the values of the estimated effort with the actual effort. While in our research, the values of the mean magnitude of relative error MMRE measure were obtained as follows by using the first method GSO were 27.246, 39.861 and 53.763, respectively. When using the second method BSA the MMRE values were as follows 20.172, 27.645 and 34.191, and when using the first proposed method BSA-GSOM1 in our research, the values of MMRE were obtained as follows 10.161, 13.929 and 15.540. When using the second proposed method BSA-GSOM2 the measure values were as follows: 2.494, 2.178 and 2.436. It was noted that the second proposed method in our research was the best among the methods, and an estimated effort was obtained very near to actual effort of NASA projects.

7. Conclusion

In this research, a new tool was presented to solve the problem of software effort and estimate the cost, using swarm intelligence methods. As these proposed models are used to assist project managers in managing and planning projects, as well as avoiding unforeseen risks, as well as avoiding delays that occur during the project period. Four intelligent methods were presented, the first of which was the glowing worm method GSO and the bird swarm method BSA, and then these two methods were hybridized to produce a new hybrid method BSA-GSOM1 with better results, and to improve the work of this proposed method and reach the best solution for estimating the project effort. The measures of the estimated effort were used as a second fitness function to produce a new hybrid method BSA-GSOM2. It has high efficiency and excellent accuracy. Where performance measures were calculated for methods of estimated effort. According to the results obtained, the second proposed method, BSA-GSOM2, was the best among the methods.

The research dealt with many different methods to solving the problem of finding the programming estimate that comes closest to the real effort, and this work can be used as a springboard for launching more projects in the future. For example, in addition to the methodologies utilized in this study, other swarm intelligence methodologies could be applied. Also, the work can be developed by obtaining a different database for projects that have been applied in various areas of life.

Conflicts of Interest

Declare conflicts of interest or state “The author declares no conflict of interest.” Authors must identify and declare any personal circumstances or interest that may be perceived as inappropriately influencing the representation or interpretation of reported research results.

Author Contributions

For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used as follows: "Conceptualization, Shahbaa; methodology, Shahbaa; software, Shahbaa; validation, Shahbaa; formal analysis, Shahbaa; investigation, Shahbaa; resources, Shahbaa; data curation, Shahbaa; writing—original draft preparation, Shahbaa; writing—review and editing, Shahbaa; visualization, Shahbaa; supervision, Shahbaa; project administration, Shahbaa; funding acquisition, Shahbaa", etc. Authorship must be limited to those who have contributed substantially to the work reported.

Acknowledgments

The author would like to thank the University of Mosul/ College of computer sciences and mathematics in Iraq for providing facilities to completion of this research.

References

[1] P. Rijwani and S. Jain, "Enhanced Software Effort Estimation using Multi Layered Feed Forward Artificial Neural Network Technique", *Procedia Computer Science*, Vol. 89, pp. 307-312, 2016.

[2] H. Carvalho, M. Lima, W. Santos, and R. Fagunde, "Ensemble Regression Models for Software Development Effort Estimation: A comparative Study", *International Journal of Software Engineering & Applications (IJSEA)*, Vol. 11, No. 3, pp. 71-86, 2020.

[3] R. Mehdi, "Software Effort Estimation Using Adaptive Fuzzy-Neural Approach", *International Journal of Computer Applications Technology and Research*, Vol. 6, No. 7, pp. 329-332, 2017.

[4] R. Sharma and R. Kishore, "COCOMO II Implementation Using Perceptron Learning Rule", *International Journal of Engineering Research & Technology (IJERT)*, Vol. 2, No. 6, pp. 3247-3256, 2013.

[5] B. Seref and N. Barisci, "Software Effort Estimation Using Multilayer Perceptron and Adaptive Neuro Fuzzy Inference System", *International Journal of Innovation, Management and Technology*, Vol. 5, No. 5, 2014.

[6] B. Singh and A. Misra, "Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects", *International Journal of Computer Applications*, Vol. 59, No. 9, pp. 22-26, 2012.

[7] B. Khaleel, "Image Clustering based on Artificial Intelligence Techniques", *AL-Rafidain Journal of Computer Sciences and Mathematics*, Vol. 11, No. 1, pp. 99-112, 2014.

[8] T. Kalaiselvi, P. Nagaraja, and Z. Basith, "A Review on Glowworm Swarm Optimization", *International Journal of Information Technology (IJIT)*, Vol. 3, No. 2, pp. 49-56, 2017.

[9] S. Khaleel, "Image Compression Using Swarm Intelligence", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 1, pp. 257-269, 2021.

[10] Y. Sinkar and C. Rjabhushanam, "Glowworm swarm optimization algorithm discrete applications", *Journal of Critical Reviews*, Vol. 7, No. 8, pp. 1752-1756, 2020.

[11] B. Khaleel, "Using Artificial Intelligence Techniques for Image Compression", *AL-Rafidain Journal of Computer Sciences and Mathematics*, Vol. 11, No. 2, pp. 65-81, 2014.

[12] D. Nandal and O. Sangwan, "Software Cost Estimation by Optimizing COCOMO Model Using Hybrid BATGSA Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 11, No. 4, pp. 250-263, 2018.

[13] B. Marapelli, "Software Development Effort Duration and Cost Estimation using Linear Regression and K-Nearest Neighbors Machine Learning Algorithms", *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, Vol. 9, No. 2, pp. 1043-1047, 2019.

[14] A. Baareh, "Optimizing Software Effort Estimation Models Using Back-Propagation Versus Radial Base Function Networks", *Journal of Computer Science*, Vol. 15, No. 3, pp. 321-331, 2019.

[15] B. Khan, R. Naseem, M. Binsawad, M. Khan, and A. Ahmad, "Software Cost Estimation Using Flower Pollination Algorithm", *Journal of Internet Technology*, Vol. 21, No.5, pp. 1243-1251, 2020.

[16] A. Kumar, B. Patro, and B. Singh, "Parameter Tuning for Software Effort Estimation Using Particle Swarm Optimization Algorithm", *International Journal of Applied Engineering Research*, Vol. 14, No. 2, pp. 139-144, 2019.

[17] S. Chirra and H. Reza, “A Survey on Software Cost Estimation Techniques”, *Journal of Software Engineering and Applications*, Vol. 12, pp. 226-248, 2019.

[18] M. Bosu and S. Macdonell, “Experience: Quality Benchmarking of Datasets Used in Software Effort Estimation”, *ACM Journal of Data and Information Quality*, Vol. 11, No. 4, pp. 19:1-19:38, 2019.

[19] M. Aljohani and R. Qureshi, “Comparative Study of software estimation techniques”, *International Journal of Software Engineering & Applications (IJSEA)*, Vol. 8, No. 6, pp. 39-53, 2017.

[20] A. Kaushik, A. Chauhan, D. Mittal, and S. Gupta, “COCOMO Estimates Using Neural Networks”, *Intelligent Systems and Applications*, Vol. 9, pp. 22-28, 2012.

[21] P. Rama, D. Prasad, and R. Sudha, “Hybrid Neuro-Fuzzy Systems for Software Development Effort Estimation”, *International Journal on Computer Science and Engineering (IJCSE),* Vol. 4, No. 12, pp. 1924-1932, 2012.

[22] A. Zakrani, M. Hain, and A. Idri, “Improving software development effort estimation using support vector regression and feature selection”, IAES *International Journal of Artificial Intelligence (IJ-AI),* Vol. 8, No. 4, pp. 399-410, 2019.

[23] V. Venkataiah, M. Ramakanta, and M. Nagaratna, “Review on Intelligent and Soft Computing Techniques to Predict Software Cost Estimation”, *International Journal of Applied Engineering Research*, Vol. 12, No. 22, pp. 12665-12681, 2017.

[24] P. Singal, A. Kumari, and P. Sharma, “Estimation of Software Development Effort: A Differential Evolution Approach”, *Procedia Compute Science1,* Vol. 67, No. 2, pp. 643-652, 2020.

[25] S. Khaleel and A. A. Thanoon, “Design a Tool for Generating Test Cases using Swarm Intelligence”, Vol. 10, No. 1, pp. 421-444, 2013.

[26] J. Odili, M. Kahar, A. Noraziah, and S. Kamarulzaman, “A comparative evaluation of swarm intelligence techniques for solving combinatorial optimization problems”, *International Journal of Advanced Robotic Systems,* pp. 1-11, 2017.

[27] S. Khaleel and R. khaled, “Selection and Prioritization of Test Cases by using Bees Colony”, *AL-Rafidain Journal of Computer Sciences and Mathematics*, Vol. 11, No. 1, pp. 179-201, 2014.

[28] J. Xue and B. Shen, “A novel swarm intelligence optimization approach: sparrow search algorithm”, *Systems Science & Control Engineering: An Open Access Journal*, Vol. 8, No. 1, pp. 22-34, 2020.

[29] S. Khaleel and K. Saleh, “Detection of Network Anomaly Based on hybrid intelligence techniques”, *AL-Rafidain Journal of Computer Sciences and Mathematics,* Vol. 9, No. 2, pp. 81-98, 2012.

[30] P. Oramus, “Improvements to glowwrom swarm optimization algorithm”, *Computer Science*, Vol. 11, pp. 1-20, 2010.

[31] Z. Yu and X. Yang, “Full Glowworm Swarm Optimization Algorithm for Whole-Set Orders Scheduling in Single Machine”, *Hindawi Publishing Corporation the Scientific World Journal*, Vol. 2013, p. 6, 2013.

[32] Y. Wang, Z. Cui, and W. Li, “A Novel Coupling Algorithm Based on Glowworm Swarm Optimization and Bacterial Foraging Algorithm for Solving Multi-Objective Optimization Problems”, *Algorithms*, Vol. 12, No. 61, pp. 1-15, 2019.

[33] Z. Li and X. Huang, “Glowworm Swarm Optimization and Its Application to Blind Signal Separation”, *Hindawi Publishing Corporation Mathematical Problems in Engineering*, Vol. 2016, p. 8, 2016.

[34] D. Zhang, J. Yang, and P. Yang, “An Improved Chaos Bird Swarm Optimization Algorithm”, *IOP Conf. Series: Journal of Physics: Conf. Series*, pp. 1-8, 2019.

[35] G. Sabarinath and T. Manohar, “Application of Bird Swarm Algorithm for Optimal Allocation of Renewable Energy Driven Distributed Generation”, *International Journal of Engineering and Advanced Technology (IJEAT),* Vol. 8, No. 3S, pp. 167-171, 2019.

[36] M. Lin, Y. Zhong, J. Lin, and X. Lin, “Discrete Bird Swarm Algorithm Based on Information Entropy Matrix for Traveling Salesman Problem”, *Hindawi: Mathematical Problems in Engineering*, Vol. 2018, p. 15, 2018.

[37] C. Zhou, L. Mei, R. Chellali, and Y. Zhao, “An Improved Bird Swarm Algorithm with Adaptive Characteristics”, *Advances in Computer Science Research*, Vol. 86, pp. 230-235, 2018.