Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 3752264, 10 pages
https://doi.org/10.1155/2022/3752264

Research Article

Soft Hunting Algorithm for Auto-Tuning Software Reliability Growth Models

Rajani, Naresh Kumar, and Kuldeep Singh Kaswan

School of Computing Science and Engineering, Galgotias University, Greater Noida, Uttar Pradesh, India

Correspondence should be addressed to Rajani; rajani.cs@galgotiasuniversity.edu.in

Received 20 May 2022; Accepted 23 June 2022; Published 17 August 2022

Academic Editor: Akshi Kumar

Nature-inspired algorithms are popular for auto-tuning software reliability growth models in recent decades due to their derivative-free natural tendency to circumvent the local optima problem. These methods have indeed exhibited enormous effectiveness in estimating software dependability. The goal of this study is to present a new nature-inspired approach for parameter improvement of system reliability predictions based on the hunting abilities of smoother-coated otters. The otters' most notable characteristic is their ability to hunt in groups. In this study, clever otter hunting behavior is used to enhance reliability engineering parameters of the model. Otters work well together and have a strong sense of teamwork. Matte finish coated otters' smart fish scavenging capacity distinguishes them from many other swarm intelligence-based techniques. Three stages of otter searching are accomplished: traveling in a V formation in the direction of the prey's movements, advancing forcefully via the stream, and then assaulting the prey on the beach. Three software dependability models are utilized to validate the applicability of the suggested approach. The study's findings demonstrate that the suggested algorithm outperformed the ABC, GA, and PSO algorithms by 75% and 50%, respectively, in terms of reduced SSE and lower MSE. The smooth-coated otter's cognitive foraging behavior gives great gain capabilities in parameters software cost estimation system reliability analysis. The outcomes are encouraging for auto-tuning software reliability growth models. Smooth-covered otter optimization can also be used to solve other efficiency challenges.

1. Introduction

Because of its accessibility, adaptability, derivative-free structure, and capability to circumvent the local optima difficulty, meta-heuristic algorithms have grown in popularity over the last few generations. These algorithms search for possible solutions using predefined rules that may be based on computational intelligence behavior, nature's developmental processes, human control the way, or physics principles. A number of nature-inspired algorithms have been created in the literature that may be used to solve quantitative optimization-based challenges in a variety of disciplines. The no-free lunch hypothesis [1] states that there is no best-suited meta-heuristic method for resolving all performance difficulties. This assumption has made this subject a continually developing field over the last century, motivating the authors of this paper to present a novel optimization for optimizing the process parameters of software dependability models. Nature-inspired techniques are classified into four categories [2]. Algorithmic based on the natural human evolution principle [3], swarm intelligence behavior-based algorithms [4, 5], physics patterns called algorithms [6, 7], and algorithms based on human cognitive behavior [8] are among them. Between all population-based algorithms, the most difficult challenge is to maintain the algorithm's local and global search capacities [9]. This research proposes a new swarm intelligence-based method based on the hunting skill of a smooth-covered otter. They were among the world's brightest animals, capable of using tools other than primates from an early age. The otters' most notable trait is their ability to hunt in groups while firmly coordinating and demonstrating high team cohesion. Techniques for maximizing the hyperparameters are available in the field of software model validation, via the numerous conventional methods of model parameters described [10]. In heterogeneous circumstances, these strategies are predicated on the

number of obstacles and may fall in local maxima rather than converging to worldwide maxima. Nature-inspired optimization algorithms that handle nonlinear, nondifferential, and multimodal problems provide alternatives to these standard mathematical optimization techniques. The algorithm was proposed based on the swarm intelligence behavior of smoothness coating. In this article, an otter is used to test software dependability. The algorithm's applicability is validated using common software measures undertaken. The suggested algorithm's outcomes are presented to those of artificial bee colonies [11], genetic algorithms [3], and evolutionary algorithms [5]. The statistical GO model discussed the time-dependent error-detection rate model for software reliability and other performance measures [12]. The descriptive statistics of the suggested algorithm proves that the proposed otter-based method has a substantial competence. The remainder of the paper is organized as follows: A brief summary of related work nature-inspired algorithm is given in Section 2. The otter-based optimization approach presented in this study is discussed in Section 3. Section 4 goes into the experimental setup and datasets, as well as the reliability engineering models that were employed during the experiment. Section 5 outlines the planned work's results and analyses. Section 6 includes the conclusion and references.

2. Related Work

Nature-inspired algorithms are derived from two classifications: single-solution techniques and demographical solutions. In the previous class, the search process begins with a single potential answer and then advances as the number of iterations increases. Later, classes entail solving the problem using a collection of feasible solutions that begins with a randomized numerous solution and improves over the number of iterations. These two classifications are further divided into four divisions depending on their nature and include algorithms connected to the basic evolutionary principle, swarm intelligence behavior-based algorithms, physics natural process algorithms, and machine learning originated from intelligence gathering behavior. Human evolution principal-based algorithms are inspired by nature's evolution process and are primarily centered on survival of the fittest. For example, genetic algorithms [3, 13], optimization techniques [14], and evolutionary programming [15] are used to generate the next generation of individuals. The second category adheres to physical laws and includes well-known algorithms such as simulation annealing, gravity phenomenon-based search algorithm [16], black hole [6], charged system search [17], galaxy-based search optimization [18], and parallel hybrid BBO [19], among others. Swarm intelligence behavior-based algorithms are the third set of combinatorial optimization and mimic the social actions of swarms of animals, birds, and amphibian, among others. This category has the most diverse set of nature-inspired algorithms, including swarm optimization, bat method, artificial bee colony algorithm [11], flower pollinating technique, and ant colony optimization [20], among others. The last set of combinatorial optimization is influenced by biological intelligent behavior and includes a variety of generic algorithms such as teaching and learning raises the possibility [8], tabu search [21], and so on. Swarm intelligence behavior-based algorithms imitate the social activities of the flocks of animals, birds, amphibians, etc. This group is the most wide range of nature-inspired algorithms listed as follows: • particle swarm optimization (1995) inspired by the flock of birds [5] • bat-inspired algorithm (2010) inspired by the bat herd [20] • dolphin echolocation (2013) inspired from echolocation behavior of dolphins [22] • honey bee marriage based optimization (2001) inspired by the honey bees [23] • artificial fish swarm optimization (2003) inspired by the swarm of fish [24] • termite algorithm (2005) inspired by the colony of termites [25] • ACO (2006) inspired by the colony of ants [20] • ABC algorithm (2006) inspired by the bees • wasp swarm algorithm (2007) inspired by the parasitic wasps [26] • monkey search algorithm (2007) inspired by the monkeys [27] • wolf pack search algorithm (2007) inspired by the herd of wolves [28] • BEE collecting pollen algorithm (2008) inspired by the bees [29] • cuckoo search (2009) inspired by the cuckoos [30] • dolphin partner optimization (2009) inspired by the dolphins behavior [31] • firefly algorithm (2010) inspired by fire flies [32] • fruit fly optimization (2012) inspired by the fruit flies [33] • krill heard optimization (2012) inspired by the herd of krills [34]. • whale optimization algorithm (2016) inspired by the behavior of whales [2] • grey wolf optimization algorithm inspired by the hunting of wolves [9] • dynamic frequency based parallel K-bat algorithm [35]. These algorithms are doing well in their specific areas and no algorithm is applicable in all optimization problems. The authors motivated from the swarm intelligence behavior and proposed smooth-coated otter foraging behavior-based algorithm for software reliability model parameter estimation [36]. Recently, nature-inspired algorithms have been applied in 5th generation millimeter wave wireless communication propagation [19], beam forming method for a transmitting antenna [12], and 5th generation intelligent software defined atmospheric effect processing and optimal selection of software reliability models [37].

3. Smooth-Coated Otter Optimizer

Based on the actions of smoothly coating otters, this paper offers a novel auto-tuning optimization algorithm for software reliability growth models.

3.1. Otter Behavior. The (Lutrogaleperspicillata) smooth-coated otter is one of the species of otter and is only surviving symbolic of genus Lutrogale. These are generally found in Indian subcontinent and eastward to south east Asia. Smooth-coated otter as the name designates is recognized by very smooth and sleek pelage. There are a lot of interesting facts about the otters. They are among the world's brightest animals, capable of using non-primate tools from an early age. More than 100 sea otters living around the California coast were studied genetically. They observed that otters use rocks as hammers to shatter open shells and gain access to the food within. Coordination among the otters is using the vocal and non-vocal sounds. Otter, during communication with other otters in group, seems to use their mouth as the mean of non-vocal communication, mouthing, nipping, and gripping each other. Smooth-coated otters live in

group, and during fishing, they prefer a group of 4 and till now maximum group of size 11 have been noticed while fishing. A monogamous alpha pair forms an established family group having beginners of previous years. The alpha female is dominant in the group, and other mature females have been seen fighting with the alpha female and sometime leave the group and make separate groups. The dominant female regulates the group hierarchy rarely by violence unless some outsider joins the group or there is any challenge within the group. The alpha male initiates the group movement always through the territory and forays in other group, feeding ranges with conflicts. Another interesting fact about the otters is that they are generally described as fish specialist. Otters are having the intelligent foraging behavior; the diving is associated with depth and time. Otters use the tools for foraging. They hunt at time of the day and tide and place with specific depth and aquatic vegetation. These are having the most energetic foraging behavior, variation is due to fluctuating water temperature, and youngers attain the adult foraging proficiency at the age of 1-2 years. For fishing purpose, otters are using their whiskers to sense the movement in water; they also use their paws to feel for crabs in mud and for catfish in the crevices. Hunting in group is the most striking feature of the otters, strongly cooperating together, showing strong team cooperation. Repeatedly group of otters diffuses in a single, somewhat v-shaped line that points in particular direction of movement and it is found to be as wide as a creek. The dominant alpha female individual engaged in the center section. In this creation, otter ripples violently over the creek and causes panic-troubled fishes to jerk out of water a few meters forward. After 2 to 3 minutes, otter dives and appears again with a fish for about one-third in its snout. Immediately, the entire pack follows this example, while a delay appears one after the other, and many are having a fish in the muzzle. The otters then move on the shore and consume fish on grubby part of the river. The otter tosses the fish up and swallows its head in one piece. Target handling on the bank takes no more than 10 seconds. Quickly, otters start with a nose dive and distribute again for the next attack across board with the width of the creek. While hunting, the otters avoid the places where there are large size mammals as crocodiles which are the predators of otters. The hunting process is smart enough that while hunting, they never travel towards the regions where there are the fears of their lives. It appears to be the true teamwork and cooperation as shown in Figure 1. The families are hereby known to join together to make large hunting parties, ending after the hunt and returning to their own home areas. In the proposed work, smart foraging with intelligent hunting capability of the smart otters has been used for mathematical optimization. The smart and intelligent foraging behavior of otters is similar to optimization. The process of hunting fishes is analogous to finding an optimum solution to a problem.

3.2. Phases in Smooth-Coated Otter Hunting. Otter family is divided into four groups as shown in Figure 2. The fittest among the social group is considered the alpha (α) pair that dominates the other group members in the family. Second beta (β) group is considered less fit and, in a v-shaped line, this group follows the female alpha dominant member in the group. Group gamma (γ) follows β group members in

Figure 1: Otters hunting group behavior in V formation.

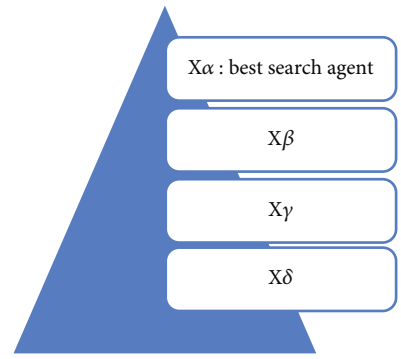

Figure 2: Hierarchy of smooth-coated otter family in terms of dominance that decreases from top to bottom the hunting process involves various phases.

their family. Maximum members in an otter family could be either 5 or 11. The rest of the family extends up to delta (δ) group. δ group follows all the previous group members. In a group, otters can communicate using their vocal and non-vocal communication capabilities which involves their chirping and growling shriek sounds.

3.2.1. Enriching the Prey. Otters prefer to hunt at the accurate places like near to the aquatic vegetation and at depths and at time when the availability of prey is high and in inactive state; this is basically at the time of day and during tide. An otter senses the prey movement very sharply in water using the whiskers that are capable of sensing even the little movements made by objects in the water. After sensing the accurate position of the suitable preys, the group of otters approaches violently towards the prey and pushes all of them towards the shore and causes panic spasms to the preys until they jump out of water. Approaching the prey and moving the prey violently to the shore can be modeled as $-Vt + 1 = Vt.w + SI\,(Xg - Xo) + (C.Xp - Xo)$ (1) and $X\,(t + 1) = Xt + V$ (2) where SI is sensation index of otters while moving and diving; SI is linearly increasing from 0 to 2 over the course of iteration. W is the magnitude of force in the direction of position of otter to prey. C is the effect of obstacles in moving otter's positions and varies

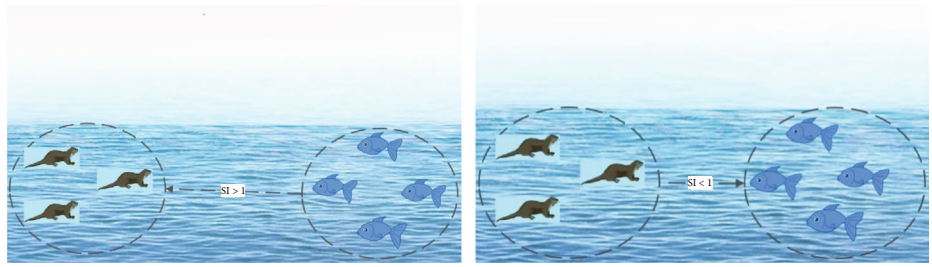

Figure 3: The use of *SI* operator.

"1. Initialize the population of otters $(Xi\,(i \cdots .n)$;
2. Initialize *SI*, *C*, and w; based on the random vector $r1$;
3. Calculate fitness capability of individual search
agent in the search space.
4. $X\alpha$ is considered the best search agent.
5. $X\beta$ is considered the second-best group of searches agent.
6. $X\gamma$ is considered the third best group of searches agent.
7. $X\delta$ is the last group of search agent that follows other
agent groups in otter family.
8. while (termination condition)
9. for each search agent
Update velocity and position of search agent by
using equations 1 and 2.
10. end for
11. Update *SI*, *C*, and w;
12. Calculate fitness of each search agent.
13. Update $X\alpha$, $X\beta$, $X\gamma$, and $X\delta$.
14. end while
15. return $X\alpha$;"

Algorithm 1: Soft Hunting Smooth-Coated Otter-Based Optimization Algorithm.

according to equation (3) $C = 2.r1$ (3) where $r1$ is random vector [0,1]. When the sensation index value is too large, then searches for the prey in different direction. When the sensation index is high and the distance value is very low, then the otters will attack the prey with a large force with a fast move, but when the distance becomes large, otters will avoid the movement towards the prey.

3.2.2. Mathematically Model Hunting the Prey. To the hunting behavior of the otter, authors have assumed that α, β, γ, and δ family members are having the most skilled sensing capabilities and they have paramount awareness of location of the prey. So according to the hierarchy, the best search agent is alpha female member and other agents follow it. So the equation is modeled as $-Vt + 1\alpha = Vt.w\,\alpha + SI\alpha\,(Xg - Xo) + (C1.X\alpha - Xo)$ (4). Second beta group follows alpha member and updates its position and velocity as $Vt + 1\beta = Vt.w\beta + SI\beta\,(Xg - Xo) + (C2.X\beta - Xo)$ (5). Similarly, third and fourth group members follow their higher members and update positions and velocities.

3.2.3. Exploitation (Attacking the Prey). Smooth-coated otters attack the prey in group after sensing it and strongly cooperating together, showing strong team cooperation. The most dominant alpha female individual engaged the middle section and all others follow it. To model the attacking of the prey, the value of the sensation index (SI) is set in a way that when the sensation by the otter is highest then it is assumed that the prey is very near. Value of *SI* is decreased to minimum. When the value of sensation index reaches its maximum value, then it is assumed that prey is not approachable and all otters start searching in all directions. It is also assumed that when $SI < 1$, then it forces the otter to attack the prey. When $SI > 1$, then otters start searching of the preys in all directions. The use of *SI* operator is shown in Figure 3.

3.2.4. Exploration (Searching of the Prey). Smooth-coated otter searches for the prey in a coordination with its team. When the value of $SI > 1$, then $X\alpha$: best search agent $X\beta\,X\gamma X\delta$ exploration happens as all have to search for the prey in different directions. The solution tends to diverge from the prey when $SI > 1$ and it converge towards the prey when $SI < 1$. There is also an operator *C* that reflects the obstacles in the way to attack the prey. Operator *C* is provided random values depending on the random vector $r1$. This feature also emphasizes the exploration from initial to final iterations and is helpful in stagnation of local optimum values to find the prey. Soft hunting algorithm is depicting the hunting behavior of

Table 1: Models taken under consideration and their MVF for soft hunting analysis.

Model	Mean value function
G-O NHPP model [36]	$m(t) = a\left(1 - e^{-bt}\right), a(t) = a, b(t) = b$
Inflection S-shaped model [12]	$m(t) = a\left(1 - e^{-bt}\right)/1 + \beta e^{-bt}, b(t) = b/1 + \beta e^{-bt}, a(t) = a$
Zheng-Teng-Pham [37]	$m(t) = (a/p - \beta)\left[\left(1 - (1+\alpha)e^{-bt}/1 + \alpha e^{-bt}\right)^{(c/b)(p-\beta)}\right], b(t) = c/1 + \alpha e^{-bt}, \beta(t) = \beta$

Table 2: Datasets.

Dataset	Number of faults	Time (sec/hours/days)	Type of application
DS-1	136	25 hours	Control system and real-time command
DS-2	100	10000 hours	Software projects for tandem computers
DS-3	34	849 days	Tactical data systems of US navel
DS-4	136	88682 sec	Real-time control system

Table 3: GO model result analysis DS-1.

Sr. no.	Name of algorithm	Parameter values (estimated)		SSE	MSE	Elapsed time
		a	b			
1	ABC	142.41	0.18	2.81E+05	7.76E+03	27.43
2	GA	138.97	0.17	4.03E+05	2.10E+03	69.68
3	PSO	139.41	0.18	3.17E+06	1.27E+04	4.25
4	Otter	138.82	0.19	1.94E+05	7.16E+02	16.69

Table 4: Result analysis using inflection S-shaped model on DS-1.

Sr. no.	Name of algorithm	Parameter values (estimated)			SSE	MSE	Elapsed time
		a	b	beta			
1	ABC	143.5475	0.199477	0.494249	2.14E+03	8.57E+01	28.13206
2	GA	139.4072	0.1817297	0.4938127	1.97E+05	7.57E+03	75.68854
3	PSO	135.2333	0.197125	0.478044	2.450E+05	1.00E+04	4.012182
4	Otter	134.0043	0.158224	0.370924	1.89E+05	5.17E+03	14.89633

smooth-coated otter used to provide the optimized solution of the problem discussed in this paper.

4. Experimental Setup

The smoothly coated otters' attractive and capable hunting skill is used to estimate software dependability set of parameters. The suggested approach is implemented on an Intel(R) Core (TM) i5(5th gen)-62000 CPU running at 2.40 GHz with 4 GB of RAM and a 64-bit Windows architectural, x64-based microprocessor. The proposed approach has been tested on the most commonly used dependability estimate frameworks. The following models and their mean value function are shown in Table 1, and useful for experimental work:

The suggested algorithm's results were compared to that of various meta-heuristic algorithms such as PSO, GA, and ABC. Four benchmark datasets [38] have been used for experimental analysis. Table 2 shows the dataset used for implementation.

Dataset: the number of flaws. Time (sec/hours/days): application type 25 hours. DS-1 136 System of direction structure in instantaneously 10000 hours. DS-2 100 Tandem computing software development projects 34 849 days. DS-3 Strategic compute clusters for the United States Navy 136 88682 sec. DS-4 Systems of ultimate control

5. Result Analysis

The suggested algorithm's effectiveness is computed and assessed using different statistical findings such as predicted model attribute values, calculated value errors, root mean squared error, and completion time in milliseconds taken by techniques. More than 1000 repetitions are used to accomplish the technique. The results of several cases are

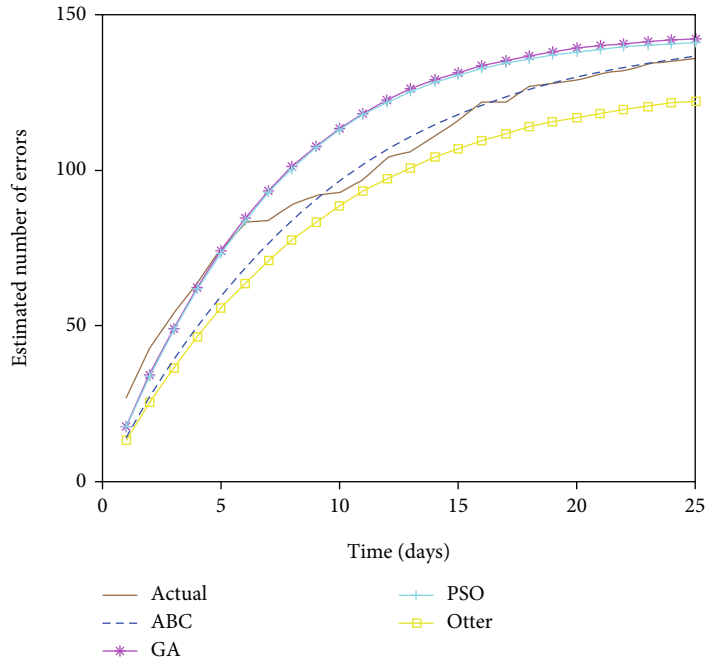

Figure 4: Estimated number of errors using GO model and inflection S-shaped model and DS-1.

Table 5: PTZ model parameter estimation DS-2.

Sr. no.	Algorithm	Parameter values (estimated)					SSE	MSE	Elapsed time
		a	*b*	*c*	alpha	beta			
1	ABC	108.789	0.0679	0.09664	1.414755	0.000490	1.02E+02	5.11E+03	25.48469
2	GA	104.7803	0.090261	1.485479	0.000361	0.000573	3.931E+05	1.97E+04	23.29652
3	PSO	109.9867	0.098521	1.692284	0.000389	0.000548	3.57E+05	1.78E+04	2.53464
4	Otter	103.4378	0.059748	0.52414	0.000386	0.00058	4.48E+04	2.241E+03	16.79884

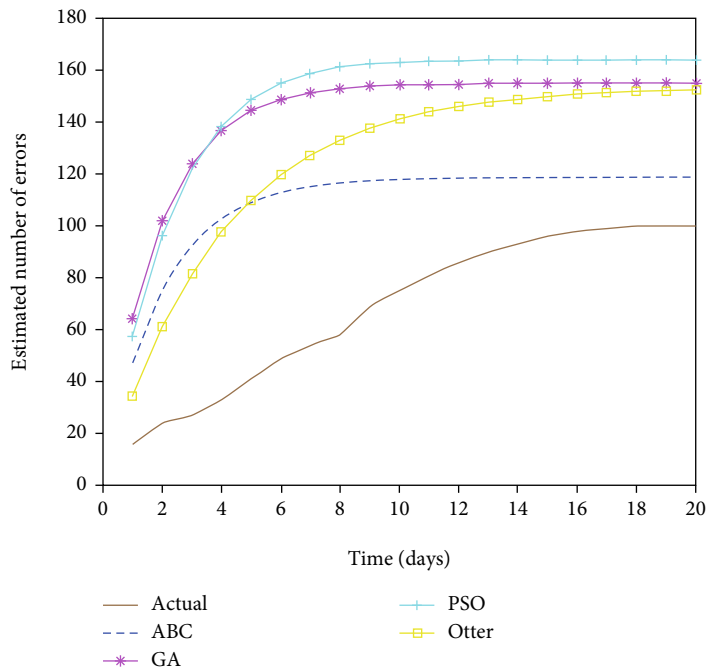

Figure 5: Estimated number of faults using PTZ model and DS-2.

Table 6: GO model result analysis DS-3.

Sr. no.	Algorithm	Parameter values (estimated)		SSE	MSE	Elapsed time
		a	*b*			
1	ABC	31.7865	0.0039	6.11E+03,	132.7339	25.15042
2	GA	34.3050	0.0068	5.07E+02	19.5016	50.8989
3	PSO	35.9554	0.0068	4.74E+02	18.22528	3.1298
4	Otter	31.0675	0.0061	6.05e+02	119.57884	16.261985

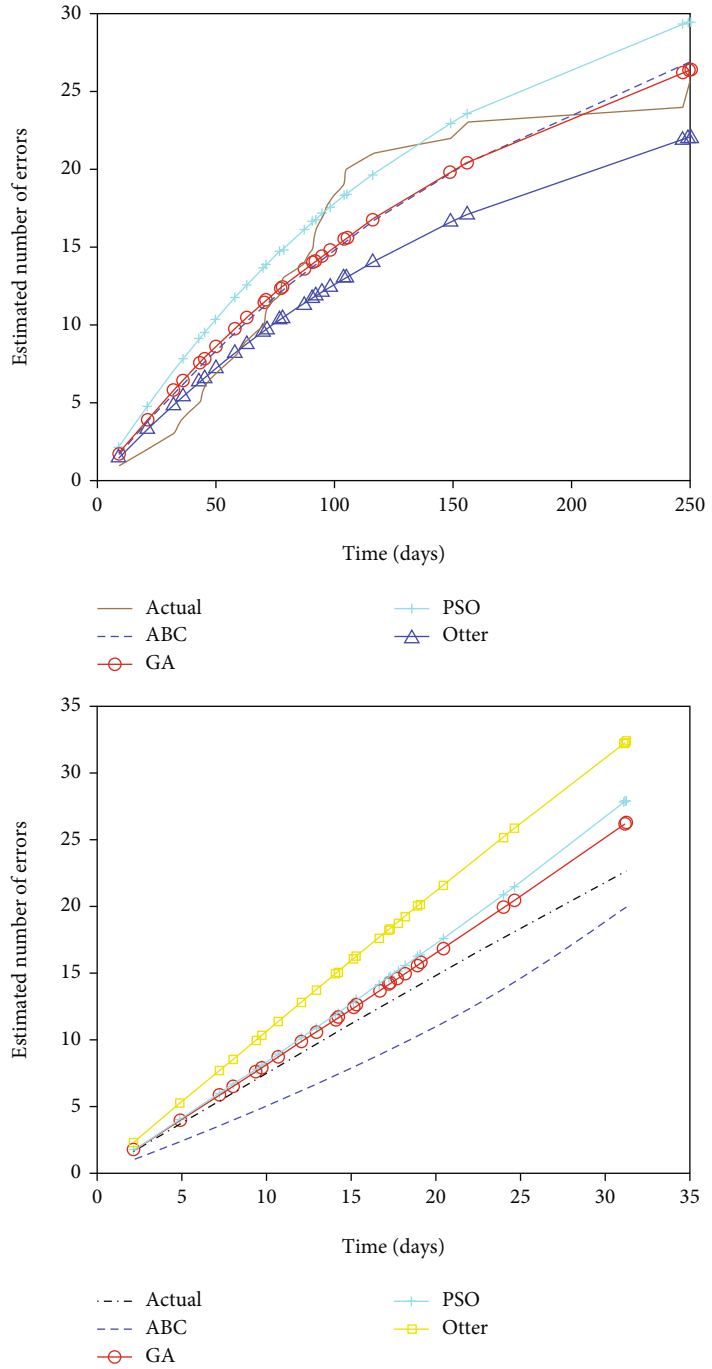

Figure 6: Estimated number of errors using GO model and inflection S-shaped model and DS-3.

Table 7: Inflection model result analysis DS-3.

Sr. no.	Algorithm	Parameter values (estimated)			SSE	MSE	Elapsed time
		a	*b*	beta			
1	ABC	31.78	7.24E-05	8.80E-05	5.41E+03	2.08E+02	27.58383
2	GA	35.77057	0.006895	0.000333	5.056E+02	19.46594	81.44662
3	PSO	31.08892	0.006895	0.000333	5.85E+03	9.78414	1.502138
4	Otter	30.95481	9.44E-05	0.000106	3.93E+02	2.33E+02	14.52722

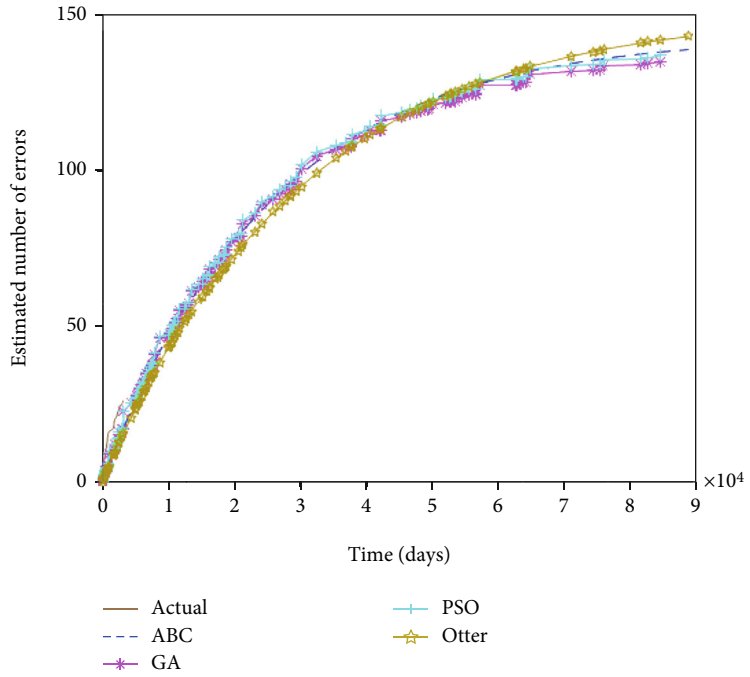

Figure 7: Estimated number of faults using PTZ model and DS-4.

analyzed using three of the most well-known software dependability frameworks and four databases.

5.1. Example 1. Analysis of the data using the GO model and the DS-1. In this example, the results obtained meta-heuristic strategies that is investigated and contrasted to an otter-based approach suggested utilizing DS-1 and the Goel-Okumotto model. Table 3 displays the findings in terms of calculated attribute values and goodness of fit of compared algorithms of SSE, MSE, and elapsed time. The suggested computation projected values obtained for the GO model for DS-1 are quite close to the real dataset values. Performance on the basis of SSE and MSE outperforms other techniques and has the lowest error rate when compared to other algorithms. Figure 3 depicts the estimated amount of mistakes using the GO model and DS-1.

5.2. Example 2. Statistical analysis using inflection S-shaped model and DS-1. This example uses inflection S-shaped model and DS-1 to illustrate how the proposed algorithm behaves in terms of parameter estimation, SSE, MSE, and elapsed time values. Data in Table 4 is showing that values of parameters estimated using the proposed algorithm are much close to actual number of errors in the dataset DS-1. In terms of SSE and MSE values, the proposed algorithm behaves significantly better than other used algorithms. The value of elapsed time is little bit more than PSO algorithm because of a greater number of parameters in the proposed algorithm. Figure 4 is depicting the behavior of DS-1 in estimating the number of faults using inflection S-shaped model.

5.3. Example 3. Statistical analysis using PTZ model and DS-2. In this example, the proposed algorithm is analyzed using five parameters PTZ model and DS-2 as shown in Table 5. The number of parameters is five in the PTZ model. Here, the proposed algorithm is tested with a large number of parameters.

The proposed algorithm is performing well in terms of parameter estimation but its SSE and MSE values are little bit higher than PSO and GA algorithms. In terms of elapsed time taken by the algorithms, PSO algorithm behaves well than another algorithm. Figure 5 is showing the estimated number of faults using DS-2.

5.4. Example 4. Statistical analysis using GO model and DS-3. This example tests the validity of proposed algorithm using time domain dataset DS-3 as shown in Table 6. In this case, GO

Table 8: PTZ model result analysis DS-4.

Sr. no.	Algorithm	Parameter values (estimated)					SSE	MSE	Elapsed time
		a	*b*	*c*	alpha	beta			
1	ABC	139.9816	0.000602	3.99E-05	8.05E-05	9.69E-05	8.74E+03	64.29368	45.512
2	GA	138.3598	0.000988	3.98E-05	0.009964	0.000348	7.25E+03	53.31243	156.644
3	PSO	139.7796	0.000998	3.99E-05	0.009884	0.007576	7.90E+03	57.25215	51.426
4	Otter	136.0941	0.000933	3.30E-05	0.008679	9.53E-05	1.22E+03	38.49756	35.229

model is used again but with time domain dataset DS-3. The proposed algorithm is beating other algorithms in terms of estimated parameter values, but in terms of MSE, SSE, and elapsed time, PSO algorithm works well than other algorithms. Figure 6 is showing the behavior of DS-3 using GO model and DS-3.

5.5. Example 5. Statistical study with the intonation S-shaped models and the DS-3. This example confirms the proposed methodology using a two-parameter bifurcation S-shaped model using DS-3. In this scenario, the suggested approach outperforms all other metrics except time duration and has acceptable parameter, SSE, and MSE values. Figure 6 depicts the estimated amount of mistakes using the inflection S-shaped model and DS-3 result analysis is shown in Table 7. Example 6: Statistical analysis with the PTZ paradigm and DS-4. This example employs the PTZ model once more, but this time using dataset DS-4, and the results show that the suggested approach outperformed all other algorithms and is capable of parameter estimation, SSE, MSE, and delayed time values. Figure 7 depicts the estimated amount of mistakes using the PTZ model and DS-4, whose result analysis is shown in Table 8.

From all results in all examples, it is found that the proposed algorithm is having better performance in most of the cases. Parameter estimation done by the proposed algorithm proves that it has estimated parameters very precisely than other algorithms. The little difference in parameter values depicts that the algorithm saves imperfect debugging behavior of the models. The number of fault values is having a little bit of deviation from the actual values due to the reason that developers and testers are not perfect initially and they gain knowledge by learning with time. Goodness of fit of the algorithm in terms of SSE and MSE values are showing major beating criterion than other algorithms where PSO and GA algorithms are behaving well. In terms of elapsed time, the proposed algorithm is found to be satisfactory even with a greater number of parameters than PSO and ABC algorithms.

6. Conclusion

This paper proposes a novel method based on the search algorithm behavior of a smoothly covered otter and it is useful for auto-tuning the software reliability growth models. A comparison of the proposed work with several other nature-inspired algorithms on the basis of statistical inference SSE, MSE, and elapsed time demonstrates the suggested algorithm's substantial optimization potential over the ABC, GA, and PSO in numerous ways. The suggested approach may be more difficult in certain circumstances, but it outperforms in others. In the future, the suggested approach might be employed as a generic algorithm for topology optimization across several domains like intelligent computing techniques. Artificial intelligence and optimization algorithm is useful for real-life applications like autonomous driving, healthcare, and recommender systems. It can also be applicable for ubiquitous and pervasive smart healthcare systems.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We thank those who have contributed to the article, especially our family and friends for their patience and support.

References

[1] A. J. Lockett, "No free lunch theorems," *Natural Computing Series*, vol. 1, no. 1, pp. 287–322, 2020.

[2] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[3] J. H. Holland, "Genetic algorithms-computer programs that 'evolve' in ways that resemble natural selection can solve complex problems even their creators do not fully understand," *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.

[4] M. N. Ab Wahab, S. Nefti-Meziani, and A. Atyabi, "A comprehensive review of swarm optimization algorithms," *PLoS One*, vol. 10, no. 5, p. e0122827, 2015.

[5] R. Eberhart and J. Kennedy, "New optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, 1995.

[6] A. Hatamlou, "Black hole: a new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175–184, 2013.

[7] W. Wiltschko and R. Wiltschko, "Magnetic orientation and magnetoreception in birds and other animals," *Journal of Comparative Physiology A*, vol. 191, no. 8, pp. 675–693, 2005.

[8] W. H. Lim and N. A. Mat Isa, "Teaching and peer-learning particle swarm optimization," *Applied Soft Computing*, vol. 18, pp. 39–58, 2014.

[9] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[10] S. K. G. Numerical, M. For, and S. K. Gupta, "Numerical Methods for Engineers," *New Academic Science*, 2013, 3rd edition.

[11] X. Yu, W. Chen, and X. Zhang, "An Artificial Bee Colony Algorithm for Solving Constrained Optimization Problems," in *2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)*, pp. 2663–2666, Xi'an, China, 2018.

[12] A. L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," *IEEE Transactions on Reliability*, vol. R-28, no. 3, pp. 206–211, 1979.

[13] A. B. Book, *Book Review*, vol. 33, no. 93, pp. 69–73, 1994.

[14] J. Grandgirard, D. Poinsot, L. Krespi, J. P. Nénon, and A. M. Cortesero, "Costs of secondary parasitism in the facultative hyperparasitoid Pachycrepoideus dubius: does host size matter?," *Entomologia Experimentalis et Applicata*, vol. 103, no. 3, pp. 239–248, 2002.

[15] P. Bangert, "Optimization: simulated annealing," *Optimization for Industrial Problems*, vol. 220, no. 4598, pp. 165–200, 2012.

[16] E. Rashedi, E. Rashedi, and H. Nezamabadi-pour, "A comprehensive survey on gravitational search algorithm," *Swarm and Evolutionary Computation*, vol. 41, pp. 141–158, 2018.

[17] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3–4, pp. 267–289, 2010.

[18] H. S. Hosseini, "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation," *International Journal of Computational Science and Engineering*, vol. 6, no. 1/2, p. 132, 2011.

[19] X. S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.

[20] S. Fidanova and S. Fidanova, "Ant colony optimization," *Ant Colony Optimization and Applications*, vol. 947, pp. 3–8, 2021.

[21] G. Barbarosoglu and D. Ozgur, "A tabu search algorithm for the vehicle routing problem," *Computers and Operations Research*, vol. 26, no. 3, pp. 255–270, 1999.

[22] A. Kaveh and N. Farhoudi, "A new optimization method: dolphin echolocation," *Advances in Engineering Software*, vol. 59, pp. 53–70, 2013.

[23] H. A. Abbass, "MBO: marriage in honey bees optimization a haplometrosis polygynous swarming approach," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, vol. 1, pp. 207–214, Seoul, Korea (South), 2001.

[24] M. Neshat, A. Adeli, G. Sepidnam, M. Sargolzaei, and A. N. Toosi, "A review of artificial fish swarm optimization methods and applications," *International Journal of Smart Sensing and Intelligent Systems*, vol. 5, no. 1, pp. 107–148, 2012.

[25] M. Roth and S. Wicker, "Termite: a swarm intelligent routing algorithm for mobilewireless Ad-Hoc networks," *Studies in Computational Intelligence*, vol. 31, pp. 155–184, 2006.

[26] B. Pröll and H. Werthner, "E-commerce and web technologies," in *Lecture Notes in Computer Science: Preface*, Springer Berlin, Heidelberg, 2005.

[27] A. Mucherino and O. Seref, "Monkey search: a novel metaheuristic search for global optimization," in *AIP Conference Proceedings*, vol. 953no. 1, pp. 162–173, 2007.

[28] C. Yang, J. Chen, and X. Tu, "Algorithm of marriage in honey bees optimization based on the Nelder-Mead method," *Proceedings on Intelligent Systems and Knowledge Engineering (ISKE2007)*, 2007.

[29] X. Lu and Y. Zhou, "A novel global convergence algorithm: bee collecting pollen algorithm," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, pp. 518–525, Springer-Verlag Berlin Heidelberg, 2008.

[30] X. Yang, S. Deb, and A. C. B. Behaviour, "Cuckoo search via L ´ evy flights," in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, pp. 210–214, Coimbatore, India, 2009.

[31] S. Yang, J. Jiang, and G. Yan, "A dolphin partner optimization," in *2009 WRI Global Congress on Intelligent Systems*, pp. 124–128, Xiamen, China, 2009.

[32] W. T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69–74, 2012.

[33] X. S. Yang, "A new metaheuristic bat-inspired algorithm," *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284, pp. 65–74, 2010.

[34] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.

[35] A. K. Tripathi, K. Sharma, and M. Bala, "Dynamic frequency based parallel k-bat algorithm for massive data clustering (DFBPKBA)," *International Journal of Systems Assurance Engineering and Management*, vol. 9, no. 4, pp. 866–874, 2018.

[36] X. Zhang, X. Teng, and H. Pham, "Considering fault removal efficiency in software reliability assessment," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 33, no. 1, pp. 114–120, 2003.

[37] D. E. Ventzas and C. Koriatopoulos, "Software reliability modeling," *International Statistical Review / Revue Internationale de Statistique*, vol. 62, no. 3, p. 289, 1994.