International Journal of Academic Research and Development
ISSN: 2455-4197
Impact Factor: RJIF 5.22
www.academicsjournal.com
Volume 3; Issue 1; January 2018; Page No. 197-199

Estimation of effort using nature inspired optimization techniques

Mandeep Kaur
Assistant Professor, Department of Computer Science, Khalsa College for Women, Civil Lines, Ludhiana, Punjab, India

Abstract
Software effort estimation is an integral part of software development process. Accurate estimates help in finding the time, schedule, and manpower and in turn the cost that would be required to develop the software. Lines of Code and Function Points are the software metrics which determine the size of the software and are hence used in the process of software management. The success of a project depends upon the accurate and timely estimates and this also help in reducing the risk involved with the development of the project. In the recent years various optimization techniques have been used to accurately and efficiently determine the software effort. This paper presents some of the popular nature inspired optimization techniques like particle swarm optimization, bee colony optimization, ant colony optimization, firefly optimization, bat optimization, etc. used by the researchers for software effort estimation.

Keywords: COCOMO, mean absolute relative error, mean magnitude of relative error, software effort estimation, optimization

1. Introduction
Effort Estimation is the process of finding the amount of effort required to develop or maintain software product in person months. Accurately estimating the software effort is one of the most challenging tasks for software developers. It is one the most important process in software development and should be done before programming the software projects. This estimate helps in estimating the project cost, time duration, maintenance effort early in the software development life cycle which in turn helps the project manager to efficiently utilize the resources, evaluate the progress of project, and provides powerful assistance for software project management decisions. Improper budgets, functions, staffing, low quality are major considerations that are the results of underestimation or overestimation of the software effort [1]. Software metrics like Lines of Code (LOC) and Function Points are most widely used to estimate software effort. LOC measures the size of the project by counting the number of source instructions ignoring the comments and header lines. The most widely used COCOMO (Constructive Cost Estimation Model) makes use of KLOC (Kilo Lines of Code) to measure the software size, where Effort is calculated as:

$$\text{Effort} = a(\text{KLOC})^b \quad (1.1)$$

Other software metric is function point which measures the software size in terms of functionality provided to the user making it independent of language, database management system or proceeding hardware [2].
This paper presents the use of various optimization techniques like Genetic algorithms, particle swarm optimization, bee colony optimization, ant colony optimization, firefly optimization, bat optimization, etc.

2. Optimization Techniques
Optimization is an art or methodology of making something as perfect, as functional as possible. It is a mathematical discipline that is concerned with the finding of the extreme (minima and maxima) of numbers, functions, or systems [3]. Optimization techniques help in finding the best solution from amongst the set of feasible solutions. The steps for performing the optimization of any problem are:

1. Select the objective function for a given problem
2. Optimize the solution using various optimization techniques.
3. Repeat the process if the stop criteria is not satisfied.
4. Process ends when the stop criterion is met.

2.1 Particle Swarm Optimization
Particle swarm optimization (PSO) was developed by Dr. Eberhart and Dr. Kennedy [4] in 1995, inspired by social behavior of bird flocking or fish schooling. In this the given problem is optimized by iterative improvement in the candidate solution with regard to a given measure of quality [5]. Initially, all particles are set to random positions in the space and small initial random velocities. The position of each particle is updated based on its best known global position in the problem space and the best position known to a particle and velocity. The objective function is sampled after each position update [6].
Each particle has a position (2.2.1) and a velocity (2.2.2) which are calculated as follows:

$$V_{i,d}(it+1) = V_{i,d}(it) + C1*Rnd(0,1)*[pb_{i,d}(it) - X_{i,d}(it)] + C2*Rnd(0,1)*[gb_d(it) - X_{i,d}(it)] \quad (2.2.1)$$

$$X_{i,d}(it+1) = X_{i,d}(it) + V_{i,d}(it+1) \quad (2.2.2)$$

where, ‘i’ is particle's index, used as a particle identifier, ‘d’ is dimension being considered, each particle has a position and a velocity for each dimension, ‘it’ is iteration number, the algorithm is iterative, ‘Xi, d’ is position of particle i in dimension d, ‘Vi, d’ is velocity of particle i in dimension d,

'C1' is acceleration constant for the cognitive component, 'Rnd' is stochastic component of the algorithm, a random value between 0 and 1.

2.2 Bee Colony Optimization

Bee colony optimization consists of two phases: forward pass and backward pass. In forward pass, the search space is explored by every artificial bee. It applies a predefined number of moves, which construct and/or improve the solution, yielding to a new solution. Next starts the backward phase where all the bees move back to the hive and share information about the quality of their solutions i.e. the value of the objective functions is computed. Then every bee decides randomly whether to continue its own exploration and become a recruiter or a follower. For every follower a solution from the recruiters is chosen and if the solutions are still not complete the process is repeated again until the best solution is found.

2.3 Ant Colony Optimization

Ant colony optimization algorithm is a meta-heuristic algorithm which is inspired by the real ant colonies where the survival of the colony is more important that the survival of individual component. Hence for this, the ants move out in search of food and when they find the food they try to find the shortest distance from their nest. And during this procedure, the ants leave a chemical called, pheromone, forming a pheromone trail from the source of food to the nest and vice versa. When other ants choose the path, they choose the one which have strong concentrations of pheromone.

2.4 Firefly Optimization

Firefly optimization technique is inspired by the natural flashing behavior of fireflies. Fireflies use their flashes for communication, to attract prey, and as protective warning mechanism. The less bright fireflies are always attracted towards brighter ones i.e. attractiveness is proportional to the intensity of brightness and this brightness decreases with the distance. Movement of fireflies towards the brighter ones is given by the following formula:

$$x_i=x_i+\beta_0 e^{-\gamma r_{ij}^2}(x_j-x_i)+\alpha(rand-1/2) \quad (2.4.1)$$

Where β_0 is the maximum coefficient of attraction between ith and jth fireflies, α is the coefficient of random displacement vector, γ is the light absorption coefficient for the environment, and is the Euclidean distance between two fireflies. Every firefly is compared to all others and the ones with fitness values less than the other are attracted according to equation (2.5.1). This process continues until the final optimum solution is obtained.

2.5 Bat Optimization

Bat optimization is a meta- heuristic technique inspired from the social behavior of bats and phenomenon of echolocation. It was developed in 2010 by Xin-She Yang [7]. In this algorithm the bat flies with a random velocity, v_i, at a position, x_i, with varying frequency f_{min} and loudness A_0.

Following equations are used to find new solutions:

$$f_i= f_{min}+(f_{max}-f_{min})\beta \quad (2.5.1)$$

$$v_i^t=v_i^t-1+(x_i^t-x^*)f_i \quad (2.5.2)$$

$$x_i^t=x_i^t-1+v_i^t \quad (2.5.3)$$

Where $\beta \varepsilon [0,1]$ is a random vector drawn from uniform distribution.

X* is current best global solution which is located after comparing all the solutions.

3. Related Work

Gharehchopogh *et al.* [8] presented a new effort estimation model for software projects using Particle Swarm Optimization (PSO) and studied the effective parameters on effort estimation using the PSO algorithm. The results of the paper showed that the proposed model gave better estimation in comparison to the COCOMO model for effort.

Ghatasheh *et al.* [9] proposed the Firefly Algorithm for optimizing the parameters of three COCOMO-based models. These models included the basic COCOMO model and other two models were the extensions of the basic COCOMO model. The developed estimation models were evaluated using different evaluation metrics. The results showed that the firefly algorithm had higher accuracy and significant error minimization as compared to Genetic Algorithms and Particle Swarm Optimization.

Dewan and Sehra [10] suggested a method of optimizing the COCOMO model coefficients, which were determined by means of the regression analysis of statistical data based on 63 different types of project data, with ant colony optimization algorithm. Firstly, the data was collected and cleaned. The outliers were removed during analyzing the data. Then different paths were generated by ants. The best path was selected by using Ant colony optimization algorithm. The performance of the system was analyzed by mean magnitude of relative error. The percent improvement of proposed model over COCOMO model was 92.29%.

Padmaja and Haritha [11] proposed one of the meta-heuristic approaches such as bat algorithm to improve the accuracy of software cost estimation with existing optimization techniques. The result obtained using the proposed model was compared with other model i.e. GRA to minimize the error rate. The experimental observation showed better results, high accuracy.

Khuat and Le [12] proposed a directed artificial bee colony algorithm in order to tune the values of model parameters based on past actual effort. The proposed methods were verified with NASA software dataset and the obtained results were compared to the existing models in other literatures. The results indicated that the proposed model had significantly improved the performance of the estimations.

4. Conclusion

Accurate software effort estimation is one of the important tasks of software development which needs to be estimated in the very early stages. Though basic software effort estimation techniques have existed since a long time but of late many researchers have used different nature inspired optimization techniques to accurately estimate the software effort. This paper presents some of the most popular optimization techniques to estimate effort which showed better results in terms of mean magnitude of absolute, relative error and

accuracy. This paper would help the researchers in their future studies and would help in adding more knowledge in this field.

5. References

1. Mall R. Fundamentals of Software Engineering, PHI Learning Private Limited, New Delhi, 2008.
2. Walston CE, Felix CP. A method of programming measurement and estimation, IBM Systems Journal, 1977; 16(1):54-73.
3. Kiranyaz S, Ince T, Gabbouj M. Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition, Adaptation, Learning, and Optimization, Springer-Verlag Berlin Heidelberg, 2014.
4. Bai Q. Analysis of Particle Swarm Optimization Algorithm, Computer and Information Science, 2008; 3(1):180-184.
5. Sheta AF. Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects, Journal of Computer Science, 2006; 2:118-123.
6. Kaur M. A Comprehensive Literature Survey on the Use of Particle Swarm Optimization Technique for Software Effort Estimation, International Journal for Research in Applied Science & Engineering Technology, 2017; 5(9).
7. Ibrahim M. A New Model for Software Cost Estimation Using Bat Algorithm, International Journal of Academic Research in Computer Engineering, 2016; 1(1):53-60.
8. Gharehchopogh FS, Maleki I, Khaze SR. A Novel Particle Swarm Optimization Approach for Software Effort Estimation, International Journal of Academic Research, 2014; 6(2):69-76.
9. Ghatasheh N, Faris H, Aljarah I, Al-Sayyed RMH. Optimizing Software Effort Estimation Models Using Firefly Algorithm, Journal of Software Engineering and Applications, 2015; 8(03).
10. Dewan N, Sehra SK. Ant Colony Optimization Based Software Effort Estimation, IJCST, 2014; 5(3).
11. Padmaja M, Haritha D. Software Effort Estimation using Meta Heuristic Algorithm, International Journal of Advanced Research in Computer Science, 2017; 8(5).
12. Khuat TT, Le MH. Optimizing Parameters of Software Effort Estimation Models using Directed Artificial Bee Colony Algorithm, Informatica, 2016; 40:427-436.