ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
**Indian Journal of Science and Technology,** *Vol 8(2), 128–133, January 2015* DOI : 10.17485/ijst/2015/v8i2/57776

# A Hybrid of Ant Colony Optimization and Chaos Optimization Algorithms Approach for Software Cost Estimation

**Zahra Asheghi Dizaji* and Farhad Soleimanian Gharehchopogh**

Department of Computer Engineering, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran; Zahra_ashegi@yahoo.com, Bonab.farhad@gmail.com

## Abstract

The main challenge in the production and development of large and complex software projects is the cost estimation with high precision. Thus it can be said that estimating the cost of software projects play an important role in the organization productivity. With the increasing size and complexity of software projects the demand to offer new techniques to accomplish this important task increases day by day. Therefore, researchers have long attempted to provide models to fulfill this important task. The most documented algorithmic model is the Constructive Cost Model (COCOMO), which was introduced in 1981 by Barry W. Boehm. But due to the lack of values for the constant parameters in this model, it cannot meet the high precision for all software projects.

Nowadays, regarding the increasing researches on machine learning algorithms and the success of these studies, in this paper, we have tried to estimate the cost of software projects according to meta-heuristic algorithms. In this paper, Ant Colony Optimization (ACO) and Lorentz transformation have been used as Chaos Optimization Algorithm (COA) and NASA datasets as training and testing sets.

To compare and evaluate the results of the proposed method with COCOMO model, MARE is used, and the results show a decline in MARE to 0.078%.

**Keywords:** Ant Colony Optimization Algorithm, Chaos Optimization Algorithm, Meta-heuristic Algorithms, Software Project Cost Estimation

## 1. Introduction

The increasing need for application softwares and the growing size and complexity of these softwares, requires an appropriate model for estimating the cost of software projects in software production organizations. Softwares not being tangible and understandable in the early stages of production cause a low accuracy in cost estimation[1]. Estimating the cost of software projects plays a significant role in organization productivity. In a sense that incorrect estimates can cause enormous financial losses and in some cases can lead to total project failure[2]. Therefore, the accuracy of estimated cost is of great importance. But regarding the creative and abstract nature of software projects, the estimation of cost and duration becomes extremely difficult[3]. The process of estimating the cost of software projects is extremely important, so that before starting to produce and develop a software project every organization first deals with evaluation of human resources and available facilities[4].

The purpose of estimating the cost of software projects can be expressed as:

1. Improving the way of classification and prioritization of softwares under production according to the business plan

**Author for correspondence*

2. Improving resource management by determining resource requirements for each project
3. Increasing resource efficiency by properly scheduling resource use
4. Customer demands on equality of estimates with actual costs.

Among algorithmic models proposed for estimating the cost of software projects so far, COCOMO is the most transparent and documented model available. In this model, basically lines of code required for software production are estimated according to the concept of Function Point, and based on that the amount of activities required for the project is estimated[3]. But considering the increasing demands for softwares with more complexity and better quality we need to present innovative techniques to fulfill this important task. In this paper, we have used ACO algorithm and COA which are considered as meta-heuristic algorithms to estimate the cost of software projects.

We have organized different sections of this paper as follows: section 2: an overview of previous works, section 3: material and methods, section 4: the proposed method, section 5: result and discussion of the proposed method, and section 6: conclusion and future works.

## 2. Previous Works

Estimating the cost of software projects has long been the focus of attention of many researchers, in a way that the model-based methods from the late 1970s continued by presentation of the models such as: SLIM by Putnam and Myers in 1992[6], Checkpoint model by Jones in 1997[7], PRICE-S model by Park in 1988[8], and COCOMO by Boehm in 1981[3]. Sometime after the presentation of algorithm models, several studies have been conducted to use non-algorithm methods such as machine learning methods as an alternative to model-based methods. Some of these papers are mentioned briefly.

In 2011, T.R. Benala et.al.[9] used Artificial Neural Network (ANN) for classifying and training the datasets, Genetic Algorithms for initializing the parameters, adapted ANN for testing and COCOMO II as a base model for comparisons. The results show improvement in accuracy of cost estimates in comparison to ANN. Mittal and Parkash have used COCOMO to estimate the cost of software projects. In their research, they have considered KLOC, which indicates the size of project, as a fuzzy number. And to evaluate the performance of their proposed method they have used Mean Absolute Relative Error (MARE) and Prediction (Pred). The results show improvement of the proposed method compared to COCOMO II [10].

Fuzzy Logic used to estimate the cost of software projects[11]. In this research, triangular Fuzzy numbers are used as input parameters for intermediate COCOMO. The data were first converted into Fuzzy model and after the estimation is completed are converted back from the Fuzzy mode. The results were compared with intermediate COCOMO and Alaa Sheta model according MMRE, PRED (n) and VAF. These comparisons indicate an improvement in accuracy of estimation compared to the models discussed.

A. Mittal et.al. used Fuzzy Logic (FL) model to estimate the cost of software projects[12]. They have introduced the cost estimation of software projects as one of the most challenging and important activities in software development. Their method showed that FL model can be used in software development. They have used 14 projects from the KEMERE set. According to the results the MARE and PRED (n) in the proposed model is better than the algorithm models. Also according to these researchers' claims, it can be said that the cost function has many parameters in software projects. Some of the factors that directly affect the cost estimation can be stated as:

1. Lines of Code (COD)
2. Kilo Lines of Code (KLOC).

## 3. Material and Methods

### 3.1 Intermediate COCOMO

The amount of effort and time required to produce software projects can be estimated as a function of performance degree or line of code. These types of functions have been determined by experience. COCOMO is considered as the most common algorithm model in software projects' cost estimation. It is used to determine the overall cost of the project as well as the cost of different stages of the project. The procedure of this model is shown in Table 1. In Table 1, the size of the project is in KLOC.

As seen in Table 1, COCOMO classifies the projects into three categories:

- Organic projects: These are conducted by small teams familiar with the work environment. They usually have specific objectives, and no sophisticated qualitative requirements are present.

**Table 1.** Software Cost Estimation with COCOMO II

| Class of Projects | COCOMO Model Formula |
|---|---|
| Organic | $PM = 2.4 * (size)^{1.05} * \prod_{i=1}^{15} EM_i$ |
| Semidetached | $PM = 3 * (size)^{1.12} * \prod_{i=1}^{15} EM_i$ |
| Embedded | $PM = 3.6 * (size)^{1.20} * \prod_{i=1}^{15} EM_i$ |

- Semi-detached projects: These are relatively larger than simple projects.
- Embedded projects: These projects are considered for large systems. Time constraints are involved and they need innovation

### 3.2 ACO Algorithm

ACO algorithm was introduced for the first time by Marco Dorigo et.al in 1992 as a multi-agent solution with the aim of solving optimization problems like traveling salesman problem [13].In 1990, Deneubourg and later Goss showed in an experiment that after putting obstacles on their way, ants are capable of discovering the shortcut routes very quickly[14]. An ant on the move leaves some pheromone (in varying quantities) on the ground and specifies the path by the smell of this material. When an ant moves alone and randomly, by facing a path with more pheromone effect, most likely chooses the path and strengthens it by leaving more Pheromone. This material will evaporate quickly however, but in short term it will remain as the ant's path on the ground. Accidents and probability play an important role in ACO. Another issue is the evaporation of the left pheromone. Evaporation of pheromone and probability allow the ants to find a shorter path. The probability of transition from state $i$ to state $j$ by $k$-th ant is calculated by Formula (1).

Formula (1):

$$P_{i,j}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{i,j}(t)\right]^{\alpha} * \left[\eta_{i,j}\right]^{\beta}}{\sum \left[\tau_{i,j}\right]^{\alpha} * \left[\eta_{i,j}\right]^{\beta}} & \text{if } K \in \text{allowed}_k \\ 0 & \text{otherwise} \end{cases}$$

$\tau_{i,j}$ is the intensity of pheromone , $\eta_{i,j}$ is the desirability of transition, α and β define the influence of pheromone on the edge and domain. Formula (2) is used to improve the searching ability and to avoid the premature convergence.

Formula (2): $\tau_{i,j}(t) = (1-p) * \tau_{i,j}(t)$

$P$ lies in the range[1,0], growth in the value of this variable causes the searches to become more random. To use the information of the best ants we need to update the pheromone, and to do so Formula (3) is used.

Formula (3) :

$$\tau_{i,j}(t+1) = \tau_{i,j}(t) + \sum_{k=1}^{n_k} \Delta\tau_{i,j}^{k}(t) \qquad \Delta\tau_{i,j}^{k}(t) = \frac{Q}{L_k}$$

$L_k(t)$ is the length of the route made by $k$-th ant in time $t$ and $n_k$ is the number of ants. ACO algorithm procedure is shown in Figure 1:

## 4. Chaos Optimization Algorithm

COA was introduced for the first time in 1980 by Henri Poincare[15]. As the description of discrete dynamical systems in time is done by the help of iterative maps, in this type of systems there is a relation like $x_{n+1}=F(x_n)$ between the points that the system selects. These points form a circuit together. Based on this, the purpose of mapping is a functional relation like $F:R \rightarrow R$ where $R$ is a set of real

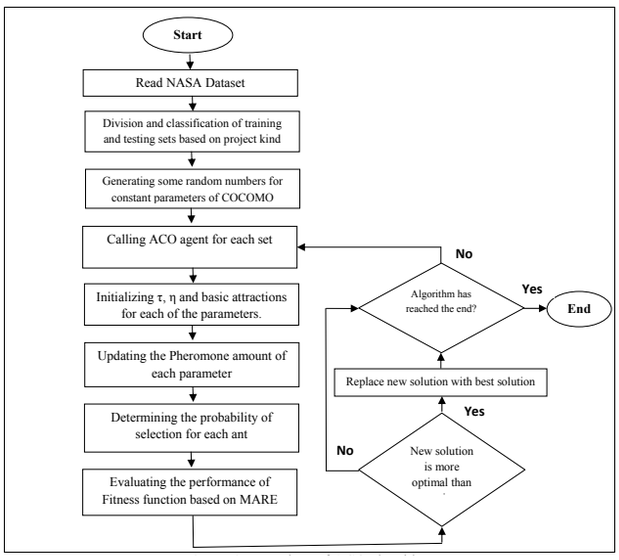


**Figure 1.** Procedure of ACO algorithm.

points by which the circuit $O(x_0)$ of $x_0$ points (belonging to the $R$ number set) is defined as a group of points. $O(x_0)=(x_0, F^2(x_0), F^3(x_0),\ldots)$[15].

First order equation is expressed as $x_{n+1}=F(x_n)$ considering $x_n = F_n(x_0)$. Maps can be classified based on linearity like Lorentz transformation, Tent maps, etc. or non-linearity like Logistic maps, Henon maps, etc.[16]. We have used Lorentz transformation in this paper. Formula (4) is used to implement this mapping.

Formula (4): $$x_{n+1} = \begin{cases} 2x_n, & 0 \le x_n \le \frac{1}{2} \\ 2-2x_n & \frac{1}{2} \le x_n \le 1 \end{cases}$$

COA and ACO algorithm are combined through Formula (5).

Formula (5): $$\tau_{i,j}(t+1) = \tau_{i,j}(t) + \sum_{k=1}^{n_k} \Delta\tau_{i,j}^k(t) * CM_1$$

In formula (5), an extra variable ($CM_1$) is added to the ACO algorithm in Formula (3). Its value is obtained from COA by Formula (4). The combination is shown in Formula (5).

## 5. Proposed Method

Software projects' cost estimation models are actually project management techniques that are used to control the effort and time of software project production. Estimation models have significant differences with each other, and experience shows that an estimate must be carried out with several models. So in this paper, we have used ACO algorithm and lorentz transformation as COA. First, the dataset is classified according to project types. Then the data are sent to ACO and hybrid ACO & COA and each of these algorithms are trained based on their type of operation and return the optimal solutions. Then the most optimal solution is selected from these solutions and is applied to the test data. The procedure of this method is shown in Figure 2.

## 6. Results and Discussion

With the increasing rate of production and development of software projects, there is a need for a methodology

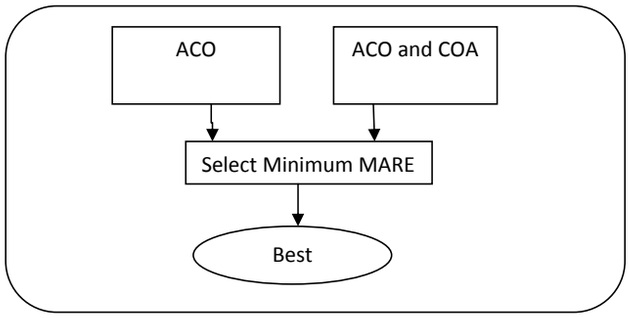


**Figure 2.** Procedure of The Proposed Method.

to estimate the costs of these projects. Therefore, in this paper we have used datasets from 60 NASA projects and ACO algorithm and the proposed hybrid algorithm to estimate the costs of software projects. We have also used MARE to compare the proposed method with COCOMO. In this paper, 80% of NASA datasets are used for training and 20% for testing. The calculation of MARE is shown in Formula (6) and Formula (7).

Formula (6): $$MARE_i = \frac{|Actual_i - Estimate_i|}{Actual_i}$$

Formula (7): $$MMARE = \frac{1}{N}\sum_{i=1}^{N} MARE_I$$

According to Figure 3, it can be said that cost estimation on the training sets based the proposed method has lowers MARE compared to intermediate COCOMO.

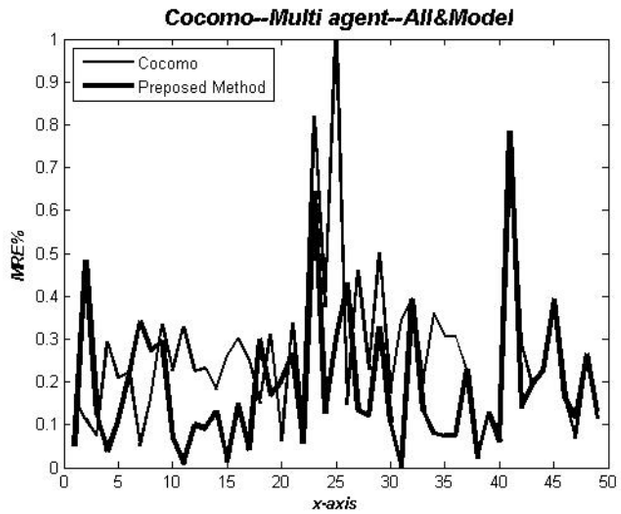


**Figure 3.** Comparison between the proposed method and COCOMO, according to MARE on training sets.

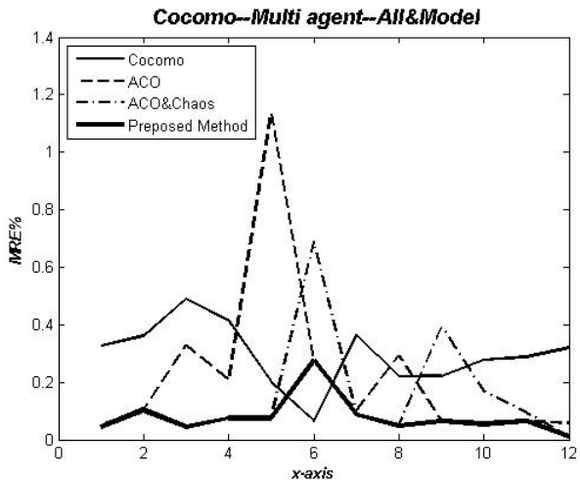


**Figure 4.** Comparison between the proposed method and COCOMO, according to MARE on testing sets.

**Table 2.** MARE on testing sets

| Model Name | MARE |
|---|---|
| COCOMO | 0.29 |
| ACO | 0.22 |
| COA_ACO | 0.14 |
| Proposed Method | 0.078 |

In Figure 4 and Table 2, ACO algorithm, combination of ACO and COA, intermediate COCOMO and the proposed method are evaluated and the results indicate optimal performance from the proposed method.

## 7. Conclusion and Future Works

The process of estimating the cost of software projects is extremely important, so that each organization before starting the production or development of a software first evaluates the required costs according to human resources and existing facilities. The increasing size and complexity of software projects entails a suitable model for estimating the cost of these projects. Therefore in this paper we have first tried to classify the projects in the NASA datasets according to project type (Organic, Semi-detached or Embedded). Then, ACO algorithm and combination of this algorithm with COA is used to estimate the cost of software projects.

According to the results, it can be said that when ACO algorithm is combined with COA, its performance improved. Also using the proposed method due to the selection of optimal solutions causes an improvement in estimated costs of software projects compared to COCOMO according to MARE. According to the results, MARE is 0.29 percent for COCOMO and it is 0.078 percent for the proposed method. Thus, based on the results the proposed method can be used to solve different optimization problems.

## 8. References

1. Kim G-H, An S-H, Kang K-I. Comparison Of Construction Cost Estimating Models Based On Regression Analysis, Neural Networks, And Case-Based Reasoning. Build Environ. 2004 Oct; 39(10):1235–42.
2. Sharma A, Kushwaha DS. Estimation of Software Development Effort from Requirements Based Complexity. Procedia Technology. 2012; 4:716–22.
3. Boehm BW. Software Engineering Economics. Prentice-Hall; 1981.
4. Park H, Baek S. An Empirical Validation of a Neural Network Model for Software Effort Estimation. Expert Syst Appl. 2008; 35(3):929–37.
5. Leungh, Zhangf. Software cost estimation. Handbook of Software Engineering and Knowledge Engineering. World Scientific Pub Co; 2001.
6. Jones C. Estimating Software Costs. Tata Mc-Graw, Hill Edition; 2007.
7. Khatibi V, Jawawi .DNA Software Cost Estimation Methods: A Review. J Emerg Trends Comput Inform Sci. 2010-11; 2(1):21–9.
8. Kumari S, Pushkar S. Performance Analysis of the Software Cost Estimation Methods: A Review. Int J Adv Res Comput Sci Software Eng. 2013 Jul; 3(7):229–38.
9. Benala TR, DehuriS , Satapathy SC, Sudha Raghavi CH. Genetic Algorithm for Optimizing Neural Network Based Software Cost Estimation. Lecture Notes in Computer Science. 2011; 7076:233–9.
10. Mittal A, Parkash K, Mittal H. Software Cost Estimation Using Fuzzy Logic.ACM SIGSOFT Software Engineering Notes. 2010 Nov; 35(1):1–7.
11. Ziauddin N, Kamal S, Khan S, Abdul J. A Fuzzy Logic Based Software Cost Estimation Model. International Journal of Software Engineering and Its Applications. 2013 Mar; 7(2):7–18.

12. Mittal A, Parkash K, Mittal H. Software Cost Estimation Using Fuzzy Logic. ACM SIGSOFT Software Engineering. 2010 Nov; 35(1)1–7.
13. Dorigo M, Gambardella LM. Ant Colony System: A cooperative learning approach to the traveling salesman roblem. IEEE Trans Evol Comput. 1997; 1:53–66.
14. Deneubourg JL, Aron S, Goss S, Pasteels JM. The self-organizing exploratory pattern of the Argentine ant. Insect Behaviour. 1990; 3:159–164.
15. Peitgen H-O, Jurgens H, Saupe D. Chaos and Fractals. United States of America: Springer Science and Business Media; 2004.
16. Gharehchopogh FS, Dizaji ZA. A new chaos agent based approach in prediction of the road accidents with hybrid of pso optimization and chaos optimization algorithms: a case study. Int J Acad Res. 2014 Mar; 6(2):108.
17. Gharehchopogh FS, Maleki I, Khaze SR. A Novel Particle Swarm Optimization Approach for Software Effort Estimation. Int J Acad Res. 2014 Mar; 6(2):69.