Determining The Parameters of Object-Oriented Software Cost Estimation Using Cuckoo Search Optimization Algorithm

Seyed Hossein Hashemi *, Lecturer, Islamic Azad University, Minab Branch, Computer Engineering Department, Hashemi.Computer@Gmail.Com

Seyedeh Nafiseh Hashemi, Director Of Modern Computer School Of Minab, Nafiseh.Hashemi68@Gmail.Com

Abstract

Accurate estimation of software development costs is critical in the early stages of project creation. The estimations that exceed the real value and the estimates that go beyond real has caused companies to suffer financial losses. The purpose of this paper is to propose a simple approach for object-oriented software cost estimation. In the proposed approach, object-oriented activities such as information hiding, abstraction, inheritance and updating are considered in Class-level method. Then, the accurate weight for each of these object-oriented programming features is determined using the cuckoo search evolutionary optimization algorithm. Our goal is to accurately estimate the effort required to improve and update the activity in the object-oriented software using the learning approach and Cuckoo Search Optimization Algorithm. Since, the requirements have changed regularly during the software development period; Therefore, cost estimation process should be implemented easily. Effective and accurate cost estimation, careful decision making in program management will help to combine changes. The proposed method is presented for estimating the effort using the cuckoo search optimization algorithm and based on the previous study and with the aim of determining the optimal weights of stimuli of object-oriented programs cost. The proposed method has been simulated and the results indicate the quality of the proposed method compared to other evolutionary algorithms as well as previous studies. The proposed method has a lower magnitude error rate than other methods, so that the error of proposed method using cuckoo search optimization algorithm to determine optimal weight based on MMRE criterion is less than other evolutionary algorithms and primitive study.

In terms of PRED criteria, the proposed method has a very high accuracy and the difference in accuracy of about 29% indicates the appropriate quality of the proposed method.

Keywords: Cost Estimation, Object-Oriented Software, Cuckoo Search Optimization, Evolutionary Algorithms, Cost Stimulus Determination

1-Introduction

Effective cost and effort estimation has a significant impact on project management. The proposed method in this paper seeks to introduce a method for estimating the cost of developing object-oriented software using the cuckoo search optimization algorithm, so that it can play an effective role in estimating the effort and cost of software projects in the organizations. Estimating the effort required to build software projects is one of the major concerns of project management [1]. Providing an approximation by estimating the required cost for a project is called cost estimation. The most important output of this activity is cost estimation along with their details and cost management design. Software devices are often modified during their development life cycle for a variety of reasons, such as adding new features, troubleshooting, or quality enhancement activities [2]. An important issue for project managers is an accurate and reliable estimate of the effort for software development, especially in the early stages of the software development cycle due to the lack of estimation nature as well as rapid changes in software development methods [1-3]. Today, project failure is a topic that has been concerned, and incorrect estimation of software costs is one of the factors influencing project failure. Cost and effort estimation are two necessary requirements to prepare a software project in software engineering and project management research, respectively. In fact, effort is the most important factor for cost determination, because software cost estimation is the problem of software effort estimation [4-6].

Effort estimation for develop software is the process of predicting the effort required to develop a software system. However, effort is only a big part of software costs, and the main reason is to determine software costs. The term software cost estimation is used in most of researches for effort equivalent in software development [7]. Accurate estimation of the effort required in the early stages of software development plays an important role in project management. Although in recent years and with the aim of increasing accuracy, the use of machine learning models has become more prominent in estimating software development efforts, none of the models available in all conditions are appropriate and provide a variety of accuracy from justice to data.

Therefore, there is need to create a reliable and valid estimation model [8].

Object-oriented programming is a programming method whose basic structure is block or object; i.e. the data and functions that are supposed to act on this data are just as likely to be in the form of an object and can create a unit (object) and can be constructed in relation to its external environment. In this method, others are outside the object and are not able to change the data in that object. For example, your bank account containing your personal information and the amount of your deposit in the bank, forms an object and it is not possible to manipulate your deposit or profile due its feature. Other features of object-oriented programming include low complexity, low cost, and the ability to expand the program faster than other programming methods [9]. There has been a lot of research on software development estimations at program code level and because of the tendency of Software developers to object-oriented programs, little research has been done on of object-oriented software development estimation. The need for more studies in this area seems to be necessary.

The study on measuring the work and time as well as determining the relationship between cost and work and between work and time used a set of formulas that obtained data from the history of previously completed object-oriented projects and the derivative effect of cost on the work. This article has several parameters that make a good combination of them very effective in estimating accuracy. So we need an algorithm that can well determine these stimuli and improve estimation accuracy. Since each of these stimuli can have different value, the search space is very large. In this situation, using the evolutionary algorithms can be very useful [9-10].

Benala et al in [11] have suggested this for future researches. Since the use of hybrid algorithms has recently been introduced and little work has been done on it, and the efficiency of the algorithm presented in their research has also had optimistic results, it seems that other machine learning algorithms can also be used for combined algorithms. The proposed algorithms are as follows:

- Artificial Bee Colony
- Differential Evolution
- Artificial Immune System
- Bacterial foraging optimization

• Neuro Fuzzy

In [11], the authors have obtained good results by studying the machine learning methods with local data. They offer the following suggestions for further research in the field of software development estimation. This study was only based on local data. That is, they used the data of a company. So their data wasn't big because a company doesn't create a lot of projects. In the future, they plan to study their data outside the company

The authors also obtained good results in effort software estimation by providing a combined algorithm by combining clustering, neural network, and encryption algorithms based on ABE[1], although the proposed estimation model is convincing, but this method faces a number of limitations. When the dataset is consistent and normal, it is not appropriate use this model. When there are many inside-data projects, its performance is similar to neural networks and low projects is similar to ABE and using Attribute-based encryption (ABE) is not suitable. Therefore, the position of the donors should be carefully considered before using the proposed model. They intend to increase the accuracy of the estimate by combining the clustering method with other intelligent methods [12].

The authors address novel topics for software effort estimation, topics that require more researches. Titles such as criterion conversion, designing and requirements phase sizing, software device complexity analysis, analysis, review, and reassessment at functional points are known. In their proposal for the forthcoming work, they emphasize further research on the impact of novel topics for future research to optimize software effort. At the end of this review, some recommendations and guidelines are presented for professionals to implement ML models in the industry [4].

• Logical understanding of ML models and collaboration with experienced researchers in implementing .ML models

• Select ML models that estimate strengths according to contexts and objectives.

• Prioritize the inside-data when building ML models if the number of projects in the data is sufficient.

•Use models.

Given that the optimal and accurate solution of the cost estimation problem is possible only in small problems and it is almost impossible to solve the problem in large dimensions, metaheuristic algorithms can be used and optimal or near-optimal answers can be obtained in reasonable time.

[1] Attribute-based encryption (ABE)

Here, cuckoo nests represent a solution and a set of coefficients for the stimuli, and each of the stimulus coefficients represents the seed in the nest. Proper care of the host bird from the eggs placed in its nest indicates that the coefficients of the stimuli are more optimal.

The cuckoo algorithm seeks to approach the optimal cost estimate by moving the eggs in different nests. A potential solution to an optimization problem is known as a set of eggs inside a nest, thus encoded in the cuckoo's search algorithm. A population evolves from possible solutions from cuckoo nests to better solutions, and ultimately achieves the optimal solution. Typically, such solutions are encrypted in a string of values of the cost-stimulus coefficients, each value corresponds to a coefficient for the cost components of object-oriented software.

The performance of each cuckoo is measured by fitness function that measures the proximity of each nest to the solution. Accurate estimation of the required effort in the early stages of software development plays a very important role in project management. Although in recent years, with the aim of increasing accuracy, the use of machine learning models in software development efforts estimation has received more attention, but none of the models available in all conditions are suitable, and most of them are able to run for object-oriented program.

2. Research Methodology

The main purpose of this paper is to propose a simple approach to estimate object-oriented software development efforts, which changes with the requirements are often suggested. In the proposed work, object-oriented features are considered, such as information hiding, abstraction, inheritance, and class- level and method updating activities. Then, the appropriate weight for each of these object-oriented programming features is determined using Cuckoo search optimization evolutionary algorithm.

Examining the object-oriented concepts, it is clear that inheritance is one of the major concerns when updating an activity and should be considered in the effort estimation. In our work, a parameter has been calculated for inheritance. If an updated class has multiple children, each of them must be used separately, and this adds to the current effort of the update process. This article focuses on creating a visual method that is easy for effort estimation.

Our aim is to accurately estimate the effort required to improve and update activity in an object-oriented software development program using a learning approach and evolution of the cuckoo search optimization algorithm. Since the need changes frequently during the software development period, the cost estimation process should be implemented easily. Effective and accurate cost estimation and careful decision-making about program management will help to combine changes. During this work, we focused on developing an effective estimation method that should be fast, easy, and simple, and these characteristics are defined as follows:

• Quick Estimation: The manager and developer can get the result immediately

• Easy Estimation: The estimation effort method should be simple during the development process.

•Simple Estimation: The effort required to estimate should be simple for the manager and developers.

Existing models such as COCOMO are a good solution for estimating the cost of combining system design changes in software development [9]. However, our goal is different from that of COCOMO. Because COCOMO uses a regression-based curve model for estimation, it is difficult for a general programmer to generalize to object-oriented programming. The proposed method is based entirely on the characteristics of object-oriented program, in which the score estimation for each component is based on the analysis of object-oriented features and the importance of each of effective components on software development cost is based on Cuckoo Search evolutionary algorithm.

The proposed approach is very similar in some respects to the performance point method. However, the performance point method is not easy in an object-oriented development environment, because the performance point method depends on the number of pages or the number of inputs / outputs. Therefore, we need to create a very simple way to estimate efforts for a program that has been developed in an object-oriented environment.

2-1-Effort Estimation for Object-Oriented Development

In order to estimate the effort to update the program in the object-oriented development environment, we selected the following four features in the effort estimation: (1) update activities, (2) types of update objectives, (3) degree of data hiding (4) The degree of inheritance. We identified and analyzed various factors that helped us identify them and evaluated their scores according to the problem of updating them. Factors influencing the effort estimation and different characteristics of object-oriented are considered. The following factors are considered when estimating the effort of object-oriented concepts:

• Update activities

• Update objectives

• Object-oriented paradigm

1. The degree of hiding of information
2. The degree of Inheritance

2-1-1 Update Activities

The type of update activity plays a significant role in the effort estimation. In general, three operations are performed on the target (class / attribute / method). These three operations are as follows [9].

• Create: Create new classes, methods and features in a class.

• Delete: delete classes, features, and exit methods.

• Modify: Modify a class, existing attributes and methods (i.e. change data type, change return type, change access to function, change access variable for classes).

Trying to "add a new class" is different from "updating an existing class". Therefore, the update (W_{upd}) factor indicates problems in the update activities.

2-1-2 Update Objectives

The effort required also depends on the type of objectives. The following are categories that should be considered [10]:

•Void type: Void is a type of data such as int, char that does not have a certain amount. In C ++ and Java, void is used as a return type when the method does not return any value called a function.

• Library type: A logical unit of process that is defined and consumed as third-party methods. For example, target modification with a different initial type is different from the type defined by the user. Type (W_{type}) is assigned to this type to differentiate their required effort.

Type 2: Primitive data-type includes data types that are in object-oriented language, such as int and char in C ++ and Java.

•Custom type: The logical unit of process defined within the developing program.

2-1-3- Object-Oriented Paradigm

Features of the object-oriented parameter for estimation include encapsulation and inheritance. All other features can be expressed in terms of these two features.

• Data Encapsulation

Encapsulation or information hiding is a feature that controls the target view of external classes. The following is a summary of the information hiding in C ++: [9]

- Private: Features and methods that are defined privately are only available in the same class in which the method and feature are defined.
- Protected: Features and methods that are protected as accessible only to inherited classes.
- Public: Features and methods that are publicly defined are available outside of classrooms.

When changing, the common feature / method changes access to another modifier or vice versa. Then more effort is needed to change the classes and target methods of the reference, so there is a difference in difficulty depending on the amount of data hiding. This is determined by the weight of W_{inf-h}.

• Inheritance Degree

Inheritance is a way to reuse and expand existing class assets and methods without modification, as well as creating a hierarchical relationship between the base class and the derived classes. In addition, inheritance plays an important role in estimating update efforts when required changes and design for a software system. Because inheritance involves the relationship between parents and children between classes (basic class and derived classes), updating to basic classes can change changes in derived classes.

We examined the heritance properties of object-orientated for effort estimation and assigned the weight to the inheritance, as required for effort estimation. As the inheritance degree in the system increases, so estimated effort increases. The difficulty they encounter due to the heritance degree is shown by W_{inht}.

2-2- Formula for Effort Estimation

There are some changes in the designing features; therefore, P program should be updated to change this features. Based on the changing features of Ri designing (required changes), Cj class should be updated. In more details, the features of Ak in Cj (class) and M1 (method)in Cj class should be updated. Total required effort for updating is total efforts required for updating separated class. Here, C1…. Cm are classes whose updating [9].

$$E(P,\sigma) = \sum_{i=1}^{n} E_{requirement}(R_i)$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{m} E_{class}(C_i) \tag{1}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{p} E_{attribute}(Ak) + \sum_{l=1}^{q} E_{method}(M_l))$$

$$E_{req}(R) = \sum_{j=1}^{n} E_{cl}(C_j) \tag{2}$$

Trying to update a class is the sum of the effort required to update its features and methods, which are calculated as follows [9]:

$$E_{cl}(C) = \sum_{k=1}^{p} E_{attr}(Ak) + \sum_{l=1}^{q} E_{meth}(M_l) \tag{3}$$

That E'meth(Ml) and Eattr(Ak) represent the effort to update the adjective and method. The estimated effort to update the program is as follows:

$$E_{attr}(A) = \alpha \times W_{upd} \times W_{type} \times W_{\inf - h} \tag{4}$$

Here α is the base score for an update feature.
According to the inheritance, it can be written as follows:

$$E_{meth}(M) = \beta \times (1 + W_{inht} \times NOC(C) + WCC(M) + WCM(M) + WPM(M)) \times W_{upd} \times W_{type} \times W_{\inf - h} \quad [9]\ (5)$$

Here, β is the base score for the method update. The variables W_{upd}, W_{type}, h-W_{inf}, W_{inht} are constants to show the characteristics in the previous section and WCC (M), WCM (M), WPM (M), NOC (C) are object-oriented fundamental criteria.
It is objectivism. In Equation (3-5), WPM (M), WCM (M), WCC (M) indicate internal characteristics in one method are independent of their definition. As the number of child classes

effort increases to achieve this. Therefore, a separate factor is required to demonstrate an inheritance relationship.

2-2-1-Fundamental Criteria

The criteria set out in Article [9] of the WCC, WCM, WPM and NOC are the reference for the complexity of a program.

• WCC represents weighted pair classes. M method includes n classes that are C1, C2 ... Cn In M. The weight of the WCi indicates the amount of classes used in the M method. Here, WCC as following:

$$WCC(M) = \sum_{i=1}^{n} w_{ci} \qquad (6)$$

WCM represents the weighted pair members. M method includes attributes that are A1, A2 ......which is defined in the same class C and is used inside the same class M method. The weight of the WAi indicates the degree where attribute inside the M method is used for an Ai attribute and is defined as follows:

$$WCM(M) = \sum_{i=1}^{n} w_{Ai} \qquad (7)$$

WPM indicates the weight parameters of the method. M method includes n parameters P1, P2 ...... Pn. The weight indicates the degree to which the Pi parameter is used within the M method. WPM is defined as follows:

$$WPM(M) = \sum_{i=1}^{n} w_{pi} \qquad (8)$$

• Number of children (NOC)

Indicates the degree of subclasses that want to inherit basic classroom methods, and in fact the NOC is equal to the number of required subsets.

2-3- Cuckoo Search Optimization Algorithm to Determine Optimal Coefficients

The variables W_{upd} ‘W_{type}, W_{inf-h} ‘W_{inht} are constants to show the importance of the characteristics provided to estimate the cost of developing object-oriented software and WCC(M), WCM(M), WPM(M) are coefficients for some characteristic; these characteristic are fundamental criteria of object-orientation and were described in the previous section. The importance of each of these components in estimating software development efforts is enormous, and their optimal determination is vital and complex at the same time. Due to the continuous and unlimited of the problem space to find the values of these coefficients, it is not possible to use definite algorithms and we need evolutionary algorithms to find the optimal values of these coefficients.

In the basic article of this research, i.e. the method presented in [9], the author's experiences have been used to determine the coefficients of these factors, which may not be effective. In this paper, the learning power of the cuckoo evolutionary search optimization algorithm is used to determine the coefficients of these variables affecting cost. The cuckoo search algorithm, as heuristic optimization technique is highly efficient for detecting relevant traits in high-dimensional search spaces based on the natural behavior of the cuckoo bird. As a result, the cuckoo's search algorithm for finding optimal solutions to find the optimal coefficients of the cost components of object-oriented software is possible.

Specifically, a potential solution to an optimal problem, encoded as a gene sequence known as cuckoo, is encrypted in the cuckoo search algorithm. A population of cuckoo’s possible solutions evolves to better solutions, and ultimately to the optimal solution. Typically, such solutions are encoded in a string of values, each corresponding to a coefficient for the cost components of object-oriented software. The performance of each cuckoo is examined by a fitness function that measures the proximity of each cuckoo to the solution. We will now examine the steps of this algorithm in more detail, which will be described in more detail below.

2-3-1- Mapping The Problem of Finding the Coefficients of Cost Components to The Cuckoo Search Algorithm

In this section, a new method for determining the optimal coefficients of cost components of object-oriented software using the cuckoo search algorithm is presented. The steps of this proposed algorithm are described in full detail in the following sections.

✓ The initial value of cuckoos

In the cuckoo search algorithm, each solution is represented by an array. In the vocabulary of the genetic algorithm, this array is called the "chromosome", while in the cuckoo search algorithm, it is called the "cuckoo's nest" and plays a similar role. In an optimal dimensional problem, a cuckoo's nest is a 1 x N. This array is defined as follows:

$$Cuckoo\_Nest = [p_1, p_2, ..., p_j]$$

pi is also a variable that needs to be optimized.

In the proposed cuckoo search algorithm, each cuckoo is a string of numbers corresponding to the coefficients of cost components of object-oriented software. Table 1 shows the presentation of the feature as an arrangement of eggs inside the initial cuckoo's nest.

Table 1: Example of Cost Coefficient Coding Using Cuckoo Search Algorithm

Creating				Deleting				Updating			
W_{inf-h}	W_{type}	W_{upd}	W_{inht}	W_{inf-h}	W_{type}	W_{upd}	W_{inht}	W_{inf-h}	W_{type}	W_{upd}	W_{inht}

The cost of a cuckoo nest is defined as the accuracy of the cost estimate. The cost function indicates the objective function, which is stated in the next section. This algorithm starts with the original cupola Npop, which is randomly selected from the population, and the best of them are selected as the best solution.

In the cuckoo search algorithm, each solution of the problem is a 1xN dimensional N problem that represents a cuckoo to solve the problem. In the proposed approach to determine the optimal values of the cost estimation parameters of object-oriented programs, each cuckoo is a string of numbers or weights. The profit or competence of a cuckoo is defined under a fitness function that is described below.

$$MREi = \frac{Estimated\ Effort - Real\ Effort}{Estimated\ Effort}$$

4.2 The Objective Function of the Cuckoo Search Optimization Algorithm for Cost Estimation

Given that Magnitude of Relative Error (MRE) (a general criterion for cost estimation models) can be used to identify the best prediction model, this criterion is used.

Magnitude of Relative Error(MRE) is calculated for all observations i whose efforts have been predicted. The aggregation of relative error intensity based on multiple observations (in the number of N) can be obtained by Mean of Magnitude of Relative Error (MMRE) using equation (16-3):

$$MMRE = \left(\frac{1}{N}\right) \sum_{i=1}^{N} \mathrm{MRE}_i \qquad (10)$$

(MMRE) is more appropriate than (MRE), because the relative error intensity measures the error for evaluation. The results of the mean relative error intensity have better results, and therefore the mean relative error intensity has been used in this study. The accuracy of an estimation technique has an inverse relationship with the mean relative error intensity. Relative error intensity for estimation and mean relative error intensity used as an accuracy measurement criterion.

2-5- Data Used

Some researchers in various companies used inter-organizational data and some external companies used extra-organizational data to build models for software development cost estimation. Researchers who have done research on this topic have found that extra-organizational models are significantly worse than inter-organizational models, and of course, because different methods are used in different studies, it is impossible to fully analyze studies.

Finally, they concluded that the studies conducted on inter-organizational data with less than 21 project samples using a one-way validation method were all better than extra-organizational models. To evaluate the proposed method, we used 70% of the data to teach the proposed method and 30% to test it. That's why this article uses a model of data within a company called NASA 93, which includes 93 software projects with 17 features from NASA.

These features include information about the size of the software project. This information can be provided in the form of code lines, function points, or many other criteria. Size variables are often considered important features for estimating effort.

2-5-1 Accuracy Criterion

As mentioned in the previous discussion, the mean criterion of relative error intensity and PRED criterion in this paper have been used for the objective function of the proposed algorithm.

• MMRE Criterion

Relative intensity error is as follows:

$$MMRE = \left(\frac{1}{N}\right)\sum_{i=1}^{N} MRE_i \qquad (11)$$

N also indicates the total number of samples evaluated.

• PRED criteria

In most research studies, another criterion used to assess the accuracy of the algorithms tested is PRED, which indicates the percentage of samples that have a low estimation error or equal to x in the estimation process. The equation used to calculate this probability is as follows:

$$PRED(X) = \frac{K}{n} \qquad (12)$$

• X indicates the desired difference, which in most research works is 0.25.

• K indicates the number of samples for which the difference in the estimated cost by the algorithm being evaluated for them is less than or equal to the actual value of X.

• n also indicates the total number of samples evaluated. Therefore, the higher the PRED value (0.25), the lower the error rate of the evaluated algorithm, and the estimated cost for the larger number of samples of the evaluated data was less than or equal to 0.25.

2-5-2- Cuckoo Search Algorithm Parameters

The proposed model of this paper has been tested on software projects of NASA93 data collection described in the previous section, and the results of the proposed model have been compared based on the cuckoo search optimization algorithm. Because the performance of the proposed algorithm is well demonstrated, we have compared software projects on the proposed cuckoo search optimization algorithm to determine the optimal weights.

In implementing the proposed metaheuristic algorithms, it is very important to apply the value of the parameters to achieve the optimal solution and can have a significant effect on the performance of an optimal solution. Metaheuristic algorithms are very sensitive to their parameters, and strict compliance with these parameters can lead to rapid response to achieve the answer. The specified value of these parameters is obtained by repeating the algorithms and their repeated functions. Accordingly, the adaptation of the parameters is one of the important factors for achieving the optimal solution. Table (2) shows the optimization value of the parameters.

Table 2: The value of parameter optimization in simulation

simulation parameters	Value
The population of the cuckoo search algorithm	100
Number of eggs per cuckoo	12
The number of iterations of the cuckoo algorithm	100
The objective function	Mean of Magnitude of Relative Error(MMRE)

2-5-3- Evaluation Results

The results obtained in Table (3) show that the proposed model has fundamentally reduced the MMRE criteria by optimizing cuckoo search. Accordingly, the proposed model is suitable for work estimation and has a lower estimation error.

2-5-3-1- Evaluate The Proposed Method According to The Objective Function Obtained in Comparison with Other Evolutionary Methods

Given that the optimal and accurate solution to the problem of software projects cost estimation is possible only in small issues, and now the problem is almost impossible in large dimensions. In this paper, Metaheuristic are used to obtain optimal or near-optimal answers in a reasonable amount of time. As mentioned in the previous discussion, evolutionary algorithms mainly have one weakness, and that is to get stuck in the local or global optimal. The Cuckoo Search Optimization Algorithm removes random and post-exploratory problems around the problem, prevents the algorithm from getting stuck in the local and global optimization, and is more efficient than other evolutionary algorithms. In this section, the efficiency of the proposed method is analyzed and compared with two other evolutionary algorithms. To do this, the value obtained from the objective function is normalized for comparison. This comparison is shown by the number of different generations in each of the algorithms.

Certainly, in software projects cost estimation, we are faced with a large number of candidate solutions, not all of which are feasible, and a cuckoo search optimization algorithm is used to solve this problem. In this section, a comparison of the MMRE objective function based on the number of generations between the proposed method and two other evolutionary algorithms, namely the genetic algorithm and the particle swarm optimization, is shown. For this evaluation, these methods have been implemented and the results have been evaluated based on the same data. These results are shown in Table 3.

Table 3: Comparison of the proposed method with basic algorithms (MMRE)

Number of generation	Proposed algorithm	Genetic algorithm	Particle Swarm Optimization
100	11.7	16.1	15.2
200	10.89	15.7	14.6
300	9.23	15.1	14.0
400	8.98	14.8	13.4
500	8.72	14.8	13.1
600	8.45	14.8	12.7

From Table (3) it can be seen that the proposed algorithm based on the cuckoo search optimization algorithm at the end of iterations can create a better solution compared to other methods. A closer look has shown that the genetic algorithm, as stated, is stuck in the local optimal, and the result indicates that this method has led to a local optimization after 400 generations, which has been shown to be a relatively flat area. The order is much earlier than the algorithm provided in Particle Swarm Optimization. However, the particle swarm optimization produces better quality responses than the genetic algorithm, despite not getting stuck in the local optimization. These results are also illustrated in Figure 1.

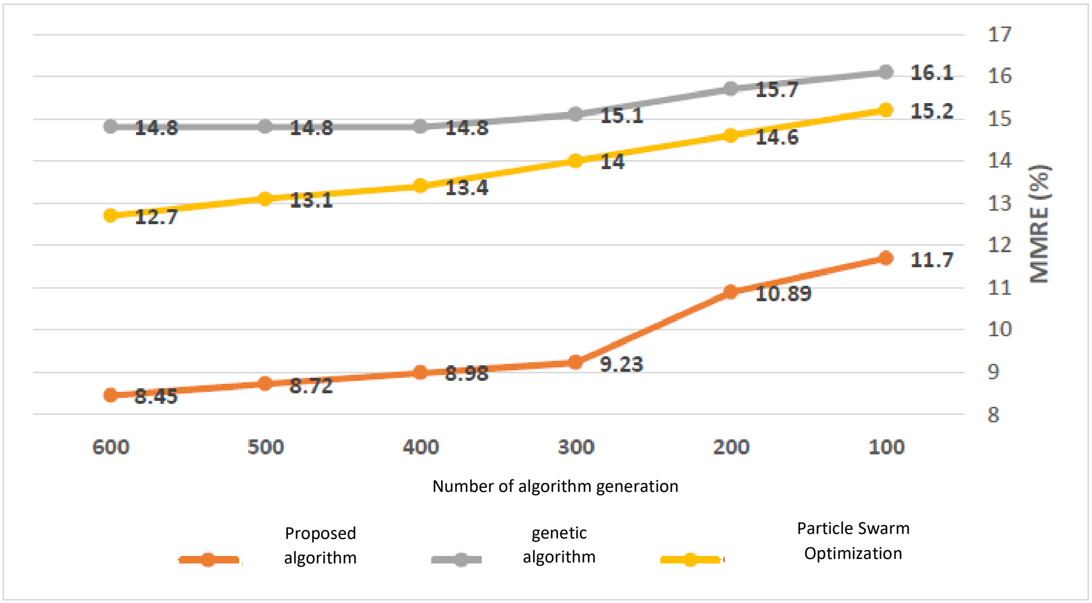


Figure 1: Comparison of The Proposed Method with Basic Algorithms in Terms of MMRE

The results indicate that the proposed method covers well the weaknesses of methods such as particle swarm optimization and genetic algorithm and does not get stuck in local optimization and has always been improving with increasing number of generations. Also, in the proposed algorithm, cuckoo optimization search has resulted in better performance than genetic algorithms and particle swarm optimization. The results indicate that the proposed method has obtained a better solution at the end of the iterations and has achieved a suitable progress in terms of energy consumption compared to the previous algorithms.

In the following, other similar comparisons have been made according to the PRED criterion. In this section, comparisons have been made with genetic evolutionary algorithms and particle swarm optimization. The results of this comparison are shown in Table 4.

Table 4: Comparison of The Proposed Method with Primitive Algorithms (PRED)

Number of generation	Proposed algorithm	Genetic algorithm	Particle Swarm Optimization
100	81.9	78.6	79.2
200	82.2	78.8	79.4
300	82.6	79.3	79.7
400	82.9	79.7	80.1++++++++++
500	83.1	79.8	80.3
600	83.3	79.8	80.5

From Table 4, we can see that the proposed algorithm based on the cuckoo search optimization algorithm at the end of the iterations can create a better solution compared to other methods. These results are also illustrated in Figure 2.

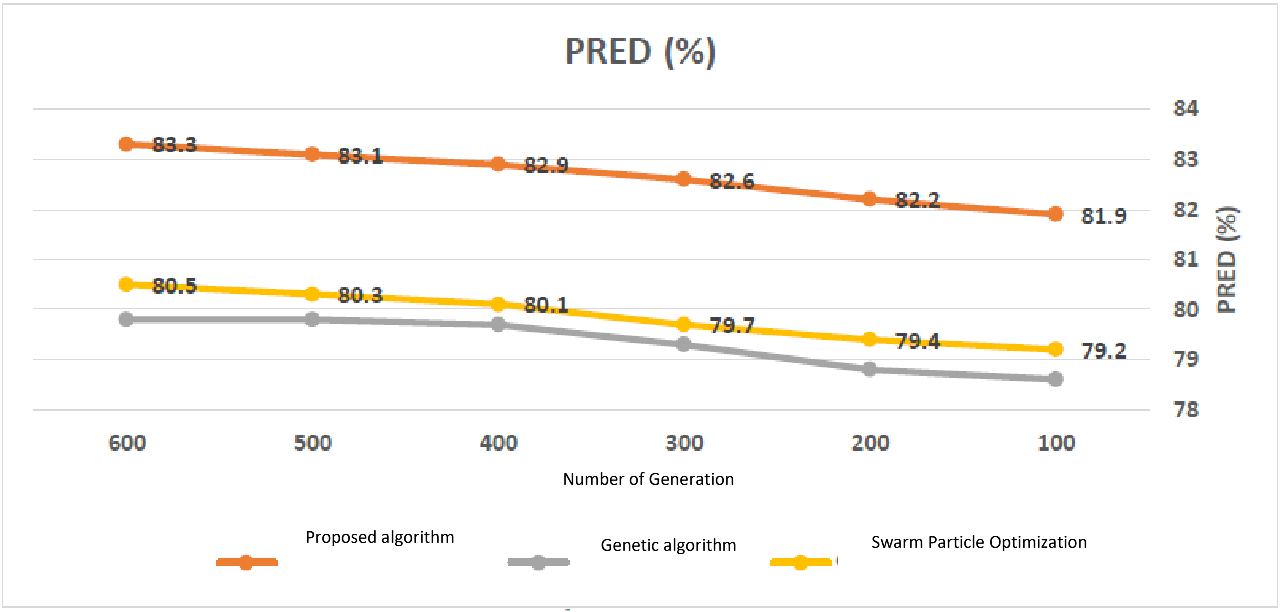


Figure 2: Comparison of the proposed method with basic algorithms (PRED)

2-5-4-Comparing with Previous Studies

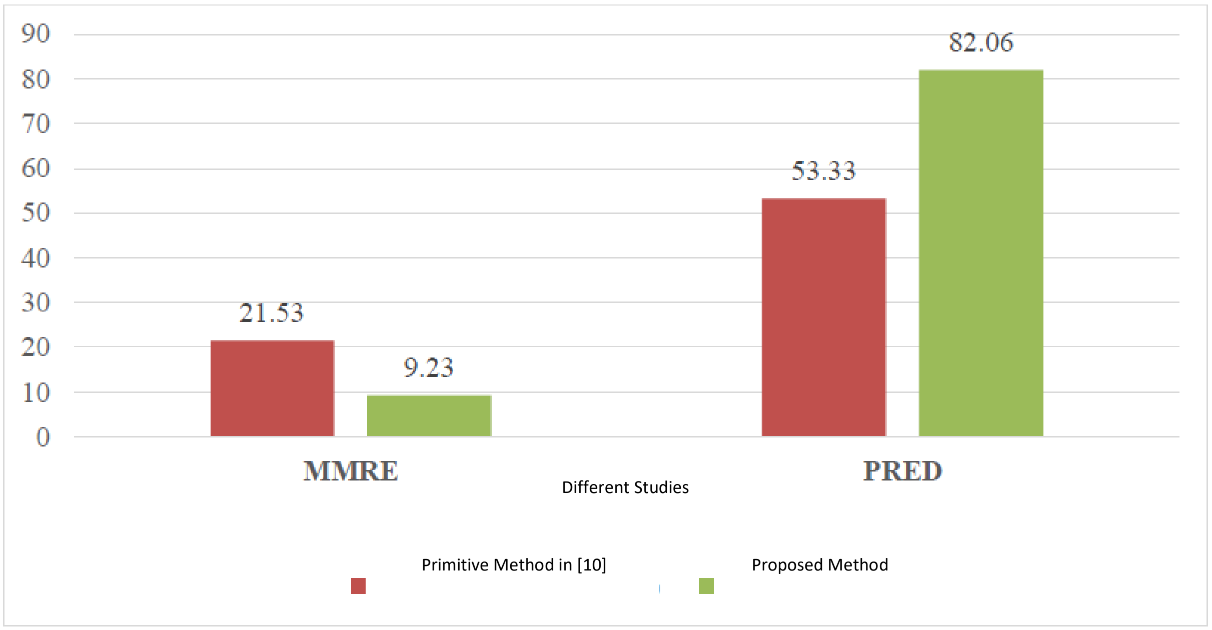
In the previous section, the quality of the proposed method of this study was compared with evolutionary algorithms. In this section, we compare with other previous studies. The results in Table 5 show that the proposed model has a better estimation accuracy compared to the method presented in [10], which is the basic article of this study.

Table 5: Implementation Results

	Compared models	
Evaluation criterion	Primitive method	Proposed method
MMRE	21.53	9.23
PRED	53.33	82.06

As can be clearly seen from the results presented in Table 5, the proposed method has a lower mean magnitude error than other methods, so that the proposed error rate is aligned with the cuckoo search optimization algorithm to determine the optimal weights. In terms of PRED criteria, the proposed method has a very high accuracy and the difference in accuracy of about 29%, indicates the appropriate quality of the proposed method. Figure 3 shows the MMRE comparison chart in the proposed model and other methods. As can be seen, the proposed model with the cuckoo search optimization algorithm for determining the weights of cost estimating factors of object-oriented software has greatly reduced the MMRE error rate compared to the baseline study.

Figure 3: Comparison of accuracy in different methods



As can be seen in Figure 3, the percentage of MMRE error in the proposed model is less than the primitive model. The reason for improving the proposed method compared to the basic article is that in the basic article, the same set of coefficients have been used for each of the cost components. By examining several values and selecting the best one, which is certainly due to the high volume of available solutions and the large size of the problem space, this is inefficient.

The proposed method of this article, in order to solve the problem in the basic article of the evolutionary algorithm search optimization that works well in continuous problems, is able to identify the optimal coefficients for each of the factors influencing the software development cost and with good dynamics, it detects near-optimal values for the coefficients of each of the factors affecting the cost, thereby improving the efficiency of the cost estimation method and reduce the error prediction of object-oriented development cost.

3- Conclusion

Accurate estimates are vital in the early stages of project creation. Estimates that are higher than the real value and estimates that are lower than the real values have caused companies to incur financial losses. If the cost estimates and manpower are too low for the project manager, the production team will be under pressure to carry out the project, and if the estimates are too high to complete on time, more resources are needed for projects.

In planning and scheduling a software project, one of the first tasks is to estimate the project time. Time constraints are very important in software projects and have led to the project being completed over a period of time. Proper planning allows the project to progress faster, in which case more effort is required in the early stages, which eliminates potential risks and operational weaknesses.

Accordingly, accurate cost estimations are very important for controlling the software project for budgeting and contracting between the parties. The experience of previous studies shows that the use of only one technique is not suitable for all conditions. Therefore, the use of estimates that use several methods will lead to estimates closer to reality. For this reason, in this study, software cost estimation is done based on cuckoo search optimization.

The purpose of this paper is to propose a simple approach to evaluating the of development software efforts that changes with requirements are often suggested. In the proposed work, we have considered object-oriented features such as information hiding, abstraction, inheritance, and class and method optimization activities. The appropriate weight for each of these object-oriented programming features is then determined using the Cuckoo Search Optimization Evolutionary Algorithm. Examining the object-oriented concepts, it is clear that inheritance is one of the major concerns when updating activity and should be considered in the effort estimation.

In our work, a parameter for inheritance has been calculated. If an updated class has multiple children, each of them must be used separately, and this adds to the current effort of the update process. This article focuses on creating a visual method that is easy to estimate. The main goal is to accurately estimate the effort required to modify and update activity in an object-oriented software development program using the learning approach and evolution of the cuckoo search optimization algorithm. Since the need changes frequently during the software development period, the process of cost estimation must be implemented easily.

Cost-effective and accurate estimates, careful decision-making about program management will help to combine changes. In doing so, we focused on developing an effective estimation method that should be quick, easy, and simple. Accordingly, software cost estimation is very effective in increasing efficiency, reducing costs, speeding and reducing time to perform operations, improving communication, ensuring software development and identifying project-related risks. In this paper, software cost estimation is performed using cuckoo search optimization.

4. References

1.Iranmanesh, Elnaz and Vahid Khatibi Bardsiri, Application of Algorithmic Models in Software Project Estimation, First National Conference on Computer Engineering Research, Tehran, Farzin Center for Sustainable Development of Science and Technology, 2014

2.Kumar, G., Bhatia, P.K.: A detailed analysis of software cost estimation using COSMIC-FFP. PAK Publishing Group J. Rev Comput. Eng. Res.2(2), 39–46 (2015)

3. Kumar, G., Bhatia, P.K.: Automation of software cost estimation using neural network technique. Int. J. Comput. Appl. 98(20), 11–17 (2014)

4. Rao, G.S, Krishna, C.V.P., Rao, K.R.: Multi objective particle swarm optimization for software cost estimation. In: ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India, vol. I, pp. 125–132. Springer International Publishing (2014)

5. Maleki, I., Ghaffari, A., Masdari, M.: A new approach for software cost estimation with hybrid genetic algorithm and ant colony optimization. Int. J. Innov. Appl. Stud. 5(1), 72–81 (2014)

6. Kaur, R., Kumar, R., Bhondekar, A.P., Kapur, P.: Human opinion dynamics: an inspiration to solve complex optimization problems. Scientific reports 3, (2013)

7. Sheta, A.F., Aljahdali, S.: Software effort estimation inspired by COCOMO and FP models: a fuzzy logic approach. Int. J. Adv. Comput. Sci. Appl. (IJACSA) 4, 11 (2013)

8. Kaushik, A., Chauhan, A., Mittal, D., Gupta, S.: COCOMO estimates using neural networks. Int. J. Intell. Syst. Appl. 9, 22–28 (2012)

9. Bardsiri, V.K., Jawawi, D.N.A., Hashim, S.Z.M., Khatibi, E.: A PSO-based model to increase the accuracy of software development effort estimation. Software Qual. J. 21(3), 501– 526 (2013)

10. Sharma Y., (2016) Effort Estimation for Program Modification in Object Oriented Development, International Conference on Computational Science and Its Applications, 2016

11. Li, B., Sun, X., Leung, H., Zhang, S. (2013) A survey of code-based change impact analysis techniques. Softw. Test. Verif. Reliab. 23(8), 613–646

12.Hosseini S., Khaled A. Al, (2014) "A survey on the Imperialist Competitive Algorithm metaheuristic: Implementation in engineering domain and directions for future research", Applied Soft Computing, vol. 24, pp. 1078-1094.