I.J. Intelligent Systems and Applications, **2019, 1, 32-42**
Published Online January 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2019.01.04

Effort Estimation of Back-end Part of Software using Chaotically Modified Genetic Algorithm

Saurabh Bilgaiyan, Dhiraj Kumar Goswami, Samaresh Mishra, Madhabananda Das
School of Computer Engineering, KIIT, Deemed to be University, Bhubaneswar
E-mail: {saurabh.bilgaiyanfcs, 1415011, smishrafcs, mndas_prof}@kiit.ac.in

Received: 24 May 2018; Revised: 01 July 2018; Accepted: 15 July 2018; Published: 08 January 2019

Abstract—The focus of Software Development Effort Estimation (SDEE) is to precisely predict the estimation of effort and time required for successfully developing a software project. From the past few years, data-intensive applications with a huge back-end part are contributing to the overall effort of projects. Therefore, it is becoming more important to add the back-end part to the SDEE process. This paper proposes an Evolutionary Learning (EL) based hybrid artificial neuron termed as dilation-erosion perceptron (DEP) framework from the mathematical morphology (MM) having its foundation in complete lattice theory (CLT) for solving the SDEE problem. In this work, we used the DEP (CMGA) model utilizing a chaotically modified genetic algorithm (CMGA) for the construction of DEP parameters. The proposed method uses the ER diagram artifacts such as aggregation, specialization, generalization, semantic integrity constraints, etc. for calculating the SDEE of back-end part of the business software. Furthermore, the proposed method was tested over two different datasets, one is existing and the other one is a self-developed dataset. The performance of the given method is then evaluated by three popular performance metrics, exhibiting better performance of the DEP (CMGA) model for solving the SDEE problems.

Index Terms—SDEE, genetic algorithm (GA), evolutionary learning (EL), dilation-erosion perceptron (DEP), mathematical morphology (MM).

I. Introduction

The process of estimating the amount of effort which is expressed in final cost or person-hours for developing a software project is termed as Software development effort estimation. The inputs which are taken for this prediction process are budgets, project plans, analysis of investment, bidding rounds and plans of iterations [1, 2, 3]. From the last few decades, the process for developing software for various software related project has advanced through heterogeneous stages. Software project's requirements and types have also got completely advanced. The process of developing software is getting more and more complex day by day because nowadays the software projects are bigger in size and more complex unlike in the past where they were simple and less amount of requirements were also less in size [4, 5, 6]. Due to the increment of the size of the software projects, the developing cost has also increased parallel. While the back-end part of the software was being neglected a few years back because the contribution of the back-end part was quite negligible. But in today's scenario, the back-end part of a data-intensive application plays a vital role during the effort estimation of the software development due to its exponential growth [7].

Since the availability of research work on SDEE of back-end part of the software is quite less, therefore it is completely a new field in which more research can be done. Many of the earlier complete research work mainly focused on the SDEE of the back-end part of the software in terms of the size of the ER (Entity Relationships) diagram and the complexity [7].

The size of the back-end part of the software, their schedule a cost and also the role of the defect are hugely affected by the ER model's complexity. The overall required effort estimation gets increased if there is an increment in the complexity of the ER model [8]. The main focus of previously done researches was on the relationships among the entities, count of the entities and the attribute's count. These were the components that were recognized as being the main factors while determining the complexity of the structure of an ER diagram [7, 8].

Moreover, for precisely estimating the size and complexity of the database of the software some more factors can be taken into consideration which may be available in the EER (extended ER) or ER like aggregation knowledge and its extensions, specialization/ generalization design constraints, relationship's degree, referential integrity constraints death, entity sets relationships which include mapping cardinality and specialty of attributes present in entity set [5, 7, 8].

For the problem of SDEE, this paper has proposed an evolutionary learning algorithm. The proposed approach has been tested over two datasets. One of them is existing and the other one is a self-developed dataset. The proposed model's robustness and its performance were tested by three well-known performance metrics which are illustrated further in this paper.

Further, this paper is divided into ten subsections including the introduction part. Section-II presents the overview of back-end estimation along with the related works found in the literature. Section-III contains the basic methodologies used for the proposed technique. Section-IV presents background knowledge on MNN. Section-V describes the proposed evolutionary process. Section-VI describes the genetic operations. Section-VII contains the knowledge about performance measures. Section-VIII describes the simulation and experimental analysis. Section-IX presents the conclusion of the proposed work along with future scope which is described in section-X.

II. Related Works

Since the growth of the software projects, the back-end part of such projects also increasing exponentially. This is making it be a pivotal contributor to the overall effort of the software projects. Following are some of the related works that have been carried out in this particular field.

The proposition of four effort metrics for the database or back-end part of the software which can be derived in accordance with the complexity level of the ER diagram has been carried out by Samaresh Mishra et. Al .[7]. The proposed work includes a new model for estimating the complexity of the database and also the size of the database. The experimental analysis has shown that the total effort which is required to create a database is directly proportional to the database's complexity. Bushra Jamal et. al. [9] have proposed an empirical validation of relational database metrics which is being used for estimating the effort of the back-end part of a software project. The proposed model is used for estimating the effort as well as some database objects because of its flexibility. The results that are found in this paper have been analyzed with an actual effort which is necessary for developing a database and found its prediction is more precise. With the help of an ER diagram, a model for estimating the effort has been proposed by Samaresh Mishra et. al. [8]. The model takes the complexity of the ER diagram of the back-end part of the software for estimating the effort. The metrics used in the model include relational size, the size of the entity and the size of semantic integrity constraints. COCOMO model is used for calculating the total required effort. The results show that the accuracy of the prediction has increased up to 4% from 3%. Parida et. al. [10] analyzed models that are usually used for the estimation of the size of the back-end part and for the effort estimation. Along with the front end development, development of the back-end also plays a pivotal role. The two components which are responsible for the cost of the back-end part are the size of each and every data item and the structural size of the database. The experimental results have shown how the effort estimation is carried out for the back-end part by using ER diagram's artifacts and it also shows how to increase the productivity for the calculation of the back-end part's actual effort, of any business software. Samaresh Mishra et. al.[11] identified that for effort estimation required for the development of the database is directly proportional to the size of the relational database model and intricacies. The verification of the model has been carried out on small size projects completed by the students. The experimental analysis is done using DCS (Database Complexity and Size) where the main focus is on data. A detailed study was carried out by the software's requirements for coming to this conclusion. Functional, document and data are the three components that are allowed by the software products. Database's volume is the main factor for the effort which is needed for the database development. A bunch of verified models for the prediction of the database's size by utilizing ER and EER (Enhanced ER) diagram has been proposed.

III. Methodologies Used

A. Database size estimation based on ER-diagram (DSEBER)

The database plays a pivotal role in implementing the data-intensive software. Since the relational database management system (RDBMS) is capable of specifying many integrity constraints related to business, it is utilized for developing the back-end of the data-intensive software products. Therefore, for developing and designing a database it becomes necessary to focus on the ER Modelling approach since object-oriented methods or functions are used by RDBMS for the back-end part [7, 8, 12]. The factors which are responsible for the effort for developing a database is the relationship complexity between the data items, database size and data item numbers. Below is Lifecycle of database development:

- Conceptual or ER modeling
- Logical modeling
- Physical modeling or process of implementing

Refining of the above life cycle process of database development is done because the estimation becomes more precise with the flow of the process. For the estimation of the size, the conceptual modeling phase has been considered better phase by utilizing artifacts of the ER diagram in this suggested work [7, 8].

B. DSEBER Metric

A proposed metric is used for measuring the total relationship, entity, and size of semantic integrity constraints from the ER model's artifacts. For the measurement of the size of the database from the ER diagrams, the given factors are being considered [7, 8].

a. Total Entity Size Estimation (TESE)

The factors that are responsible for the size of the structure of the entity are associations between the entity sets, its generalization/specialization hierarchy role, and a number of entity sets. Following factors are considered for the prediction of the size of an entity [7, 8]:

- Count of similar attributes (COSA)
- Count of derived attributes (CODA) where values can be derived from the result of calculation with the generated lines of code when the

implementation is being carried out.

- Count of multivalued attributes (COMA) which results in the creation of an additional table.
- Count of many to one or one to one participation (COM1P) which exists among various sets of entities.
- The depth of inheritance (DI)

Entity size can be measured with the help of the factors described in Table 1 [7, 8].

Table 1. Factors Considered For Entity Size

Factors	Description
COSA	Similar attribute's count
CODA	Derived attribute's count
COMA	Multivalued attribute's count
COM1P	Many-to-one or one-to-one participation's count from one entity set to another entity set
DI	Inheritance depth of the entity

The vital task of COM1P is to establish a referential integrity constraint and the increment in attribute's count on the entity set by utilizing foreign keys. The relation between the foreign key and COM1P is that both are directly proportional to each other. For the indication of the count of primary key with a numeric figure, DI is used for representing them. (COMA + 1) numbers of relations are being produced by COMA for keeping the relations under 1NF [1]. By adding the simple count of the primary and foreign key which are inherited in generalization, multivalued attributes, the calculation of count of attributes can be done [7, 8]:

$$\mathrm{COA} = \mathrm{COM1P} + \mathrm{CODA} + \mathrm{DI} + \mathrm{COSA} + \mathrm{COMA} \qquad (1)$$

If the size of the primary key of an entity set is denoted by $\mathrm{Size}_{\mathrm{pk}}$, where attribute's count is denoted as$\mathrm{Size}_{\mathrm{pk}} = 1$, where the primary key is derived from a single attribute and$\mathrm{Size}_{\mathrm{pk}} = \mathrm{n}$, when a composite primary key which consists n number of attributes [7].

When there are many multivalued attributes present in the entity set and maintenance of the relation should be in 1NF, then the base relation is being decomposed into (COMA+1) count of relation.

Determination of the count of an attribute can be done by (count of attributes in base relation (COA)-COMA) and derived relation's size from the base relation can be calculated by adding the base relation's primary key and the multivalued attributes. Now the equation by which we calculate the size of derived relation is giver below as [7].

$$\mathrm{Size}_{\mathrm{DR}} = (\mathrm{Size}_{\mathrm{PK}}) + (\mathrm{Size}_{\mathrm{MA}}) \qquad (2)$$

Here, the size of the derived attributes is represented by $\mathrm{Size}_{\mathrm{DR}}$ and $\mathrm{Size}_{\mathrm{MA}}$ is the notation for the multivalued attributes. When multivalued attributes consist of a single-valued attribute then, $\mathrm{Size}_{\mathrm{MA}}$ =1 and when multivalued attributes consist of n number of attributes then, $\mathrm{Size}_{\mathrm{MA}} = n$. Entity set's size can be determined by adding all the attributes of the relations which are produced from the entity sets, as given [7, 8]:

$$Size_{ES} = (Size_{BR}) + \sum_{i=1}^{COMA}(Size_{BR})_i \qquad (3)$$

Where entity size is denoted by$Size_{ES}$, base relation size is denoted by $Size_{BR}$and derived relation size is denoted by $Size_{DR}$. Here, the COMA and the number of derived relations are equal, which have been derived while maintaining the 1NF property from the relation.

Calculation of ER diagram's total entity size which is denoted by $Total_{ES}$ can be represent as [7]:

$$Total_{ES} = \sum_{j=1}^{CE}(Size_{ES})_j \qquad (4)$$

The number of total entity sets present in the ERD is denoted by CE.

b. Total Relationship Size Estimation (TRSE)

The factors which are responsible for relationship set's structural size are a count of entity set which is associated and count of the descriptive attribute. Each many-to-many relationship results in generating a new relation. The factors mentioned above are given in Table 2. [7, 8]:

Table 2. Factors Considered For Relationship Size

Factors	Description
NODA	Number of descriptive attributes
DMMR	The degree of many to many relationships
AGG	Count of aggregation

The size of relationship set can be found by adding a count of foreign keys that are present in it and descriptive attribute which relies on the count of aggregation that takes part in this relationship and the count of the entity. They can be represented as [7]:

$$\mathrm{SizeRS} = \mathrm{NODA} + 2 \times (\mathrm{SizePK}), \text{ if DMMR} = 1 \qquad (5)$$

$$Size_{RS} = NODA + \sum_{i=1}^{DMMR}(Size_{PK})_i, \text{if DMMR} \geq \qquad (6)$$

Size of relationship set is shown by $Size_{RS}$ and entity set count is shown by i, which is associated with this relationship set.

For $DMMR = 1$and$DMMR = 2$, foreign keys count is unchanged.

Sum of the size of all the relationship in Eq. (7) is used to get the total relationship set contained in this category [7]:

$$Total_{RS} = \sum_{k=1}^{CR}(Size_{RS}) \qquad (7)$$

Count of a many-to-many relationship is shown by CR and the total size of the relationship set is shown by$Total_{RS}$ [7, 8].

c. Total Semantic Integrity Constraint Size (TSICS)

A few business constraints are found in the early development stage in form of semantic integrity which is caught while ER modeling. For performing these type of constraints, procedural language is used. So the evaluation of the size of such semantic constraints is done on the foundation of LOC which are required for performing. Thus, the overall size of all semantic integrity constraints is given in the Eq. (8) [7]:

$$Total_{SC} = \sum_{k=1}^{j} (LOC)_k \tag{8}$$

Here Total_{SC} shows the whole size of semantic integrity constraints and j shows the whole count of the semantic integrity constraints.

d. Total Database Size from ER Diagram (TDSERD)

By taking into account the count of entity set, attribute required for entity set and relationship, relationship set count with the mapping cardinality, the database size can be evaluated from the conceptual design stage and it is also reliant on structural complexity rooted in the depth of the inheritance o entity set within the hierarchy of specialization/generalization and linked count of aggregation. Eq. (9) shows the total database size acquired by ERD [7, 11].

$$\text{Total}_{\text{ERD}} = \text{Total}_{\text{ES}} + \text{Total}_{\text{RS}} + \text{Total}_{\text{CS}} \tag{9}$$

where $\text{Total}_{\text{ERD}}$ shows the total size of the ERD.

C. DSEBER Process model

In Fig. 1, the portrait of the DSEBER model's diagram is given. It is used for predicting the effort required for developing the database with the help of measured suits of DSEBER over the phase of conceptual modeling of development of the database [7, 8, 9]. By measuring the size of the database with the help of the proposed DSEBER's cost metrics, the effort which is required for developing the database can be estimated. The requirements gathering and the requirements analyzing process are done at the early stage methodically, as per the model's need. The construction of the ER diagram and extended ER are carried out to capture the requirement of data as a factor of the analysis model. After that, the calculation of the total size of the database is done by using the proposed DSEBER cost metrics. Finally, by using the total size of the database which was calculated earlier, the effort which is required for developing the database is predicted with the help of the COCOMO model. In our proposed model we have used DEP (CMGA) model for predicting the effort required for building a database which is further discussed in the paper. Various objects which were unknown, during the early stages of the development of the database, showed out during the phase of implementation [7]. Cardinality's degree and mapping, relationship (based on number and type of attributes that are present in it), generalization and specialization and entities are those components which are used for the database's size estimation. Therefore the DSEBER model which is proposed here may play a vital role in the conceptual phase of the database system of any software [7, 8, 11].

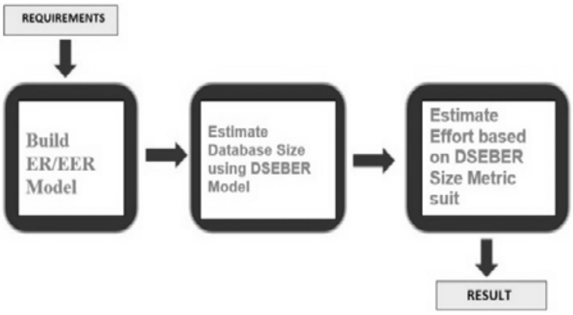

REQUIREMENTS
Build ER/EER Model
Estimate Database Size using DSEBER Model
Estimate Effort based on DSEBER Size Metric suit
RESULT

Fig.1. DSEBER Process Model

IV. Background on Mnn

Morphological Neural Network (MNN) can be described as artificial neural networks (ANNs), which carry out the basic transaction from the mathematical morphology (MM) between entire lattices at each and every network processing unit (NPU). Here we basically focus on the definition of MNNs by the algebra of lattices [13, 14].

If each and every finite and non-empty subset of Chas a supremum and infimum in C, then C (a partially ordered set) is said to be a lattice. For any $X \subseteq C$, the symbol $\wedge X$ is used for representing the infimum [14].

For any index P, $\wedge X$ can be represented as$\bigwedge_{p\in P} x^p$, when $X = \{x^p, p \in P\}$. Also, supremum o f X ($\vee X$) can be described by using same notations [1the 0, 11].

Let C and D be two lattices. Then a mapping such as $\Psi: C \rightarrow D$ is termed as increasing if and only if $\forall x, y \in C$, the following statement is satisfied [14, 15]:

$$x \leq y \Rightarrow \Psi(x) \leq \Psi(y) \tag{10}$$

By considering C as lattices, then a partial order on C^n can be described by setting [15]:

$$(x_1, \dots \dots x_n) \leq (y_1, \dots \dots y_n) \Leftrightarrow x_i \leq y_i, \quad i = 1 \dots n \tag{11}$$

The outcome of C^n(partially ordered set) derived from Eq. (11) is also termed as a lattice. It is also called as product lattice [14, 15, 16].

This is very important to take into account that the lattice C is said to be complete only when each and every nonempty subset comprises a supremum and an infimum in C. The product lattice C^n is complete only if the completeness of C lattice is achieved [14, 16, 17].

Decomposition of mappings between the complete lattices in terms of fundamental morphological operators is one of the major issues.

Let δ and ε be two operators from C to D, where both C and D are complete lattices (CLs). Then according to Banon and Barrera's decomposition theorem:

Theorem: Ψ: C → D, which is the increasing mapping, between C and D,where both C and D are complete lattices can be constructed as infimum of dilation or supremum of erosions.. For index P and Q, the existence of dilation δ^p and erosion ε^qcan be defined as [18]:

$$\Psi = \bigvee_{p \in P} \varepsilon^q = \bigwedge_{q \in Q} \delta^p \tag{12}$$

The decomposition theorem by Banon and Barrera has worked as the foundation of many MNN model's learning algorithms. The basic morphological operators in the models which occur during the decomposition process are believed to follow a special form where an extra structure of algebra is required along with the CL structure.

The main focus of this paper is on $\mathbb{R}_{\pm\infty}$ (complete lattice), because the problem of SDEE can be structured in terms of function $\mathbb{R}_{\pm\infty}^n \rightarrow \mathbb{R}_{\pm\infty}$. (where the count of so the software project's variable is represented by n).

Let $E \in R^{u \times w}$ and $F \in R^{w \times v}$ be two given matrix. Then the max-product of E and F matrixes can be represented as [14, 18]:

$$G = E \, [\![\vee]\!] \tag{13}$$

Similarly, the min-product of the above to matrixes can be computed as:

$$H = E \, [\![\vee]\!] \, F \tag{14}$$

These are described in the given Eq. (15) [15]:

$$g_{pq} = \wedge_{k=1}^{w} (e_{pk} + f_{kq}) = \vee_{k=1}^{w} (e_{pk} + f_{kq}) \tag{15}$$

Let us take up some operators ε_E, δ_E:$R_{\pm\infty}^u \rightarrow R_{\pm\infty}^v$ for $E \in R^{u \times v}$then [14, 15, 18]:

$$\varepsilon_E(\mathrm{x}) = \mathrm{E}^{\mathrm{T}} \wedge X \tag{16}$$

$$\delta_E(x) = E^T \vee X \tag{17}$$

Where T represents the term transposition of given set. Algebraic erosion is denoted by $\varepsilon_E$and algebraic dilation is denoted by δ_E from CL $R_{\pm\infty}^u$ to $R_{\pm\infty}^v$, This Eq. (12) suggests that mapping function Ψ: $R^u \rightarrow R$, which is an increasing function and can be estimated as [15, 17]:

$$\Psi \simeq \bigwedge_{p \in \bar{P}} \delta_r^p \vee \tag{18}$$

or

$$\Psi \simeq \bigvee_{q \in \bar{Q}} \varepsilon_s^q \tag{19}$$

Wherer^p and $s^q \in \mathbb{R}^u$ are the vectors for some finite index$\bar{P}$ *and* $\bar{Q}$.

In case of $\bar{P} = 1$ *and* $\bar{Q} = 1$, the vectors r,s$\in \mathbb{R}^u$ can be used for the approximation of the mapping function Ψ: $R^u \rightarrow R$ and can be represented as [14, 15, 17]:

$$\Psi \simeq \delta_r \tag{20}$$

Or

$$\Psi \simeq \varepsilon_s \tag{21}$$

Eq. (20) & (21) represent a hypothesis which is used as a basis for our SDEE problem using DEP.

V. The Process of Evolutionary Learning

One thing that should be noted down is that the variables like a, b and λ needs to be tuned for the DEP model. Therefore, the factor of weight which is going to be used for the training's process is mentioned in Eq. (18) [14, 19, 20].

$$W = (a, b, \lambda) \tag{22}$$

Until the CMGA iteration's convergence is found out, adjustments are being carried out with respect to the criteria defined for error. The representation of chromosomes weight vector of j^{th} individual from g^{th} generation is done by$w_j^{(g)}$. For determining the fitness function which is denoted by$ff(w_j^{(g)})$, the weight vector must be altered. This should show the quality of solution which is achieved by configuring the variable on the system. Fitness function is determined as given in the Eq. (23).

$$ff(w_j^{(g)}) = \frac{1}{N} \sum_{k=1}^{N} e^2(k) \tag{23}$$

The number of input patterns is represented by N and the error (instantaneous error (IE)) is represented by $e(k)$ and can be computed as [14, 20]:

$$e(k) = d(k) - y(k) \tag{24}$$

Here, the desired outcome signal is denoted by $d(k)$ and obtained output is denoted by $y(k)$ which is for a training sample that is represented by k.

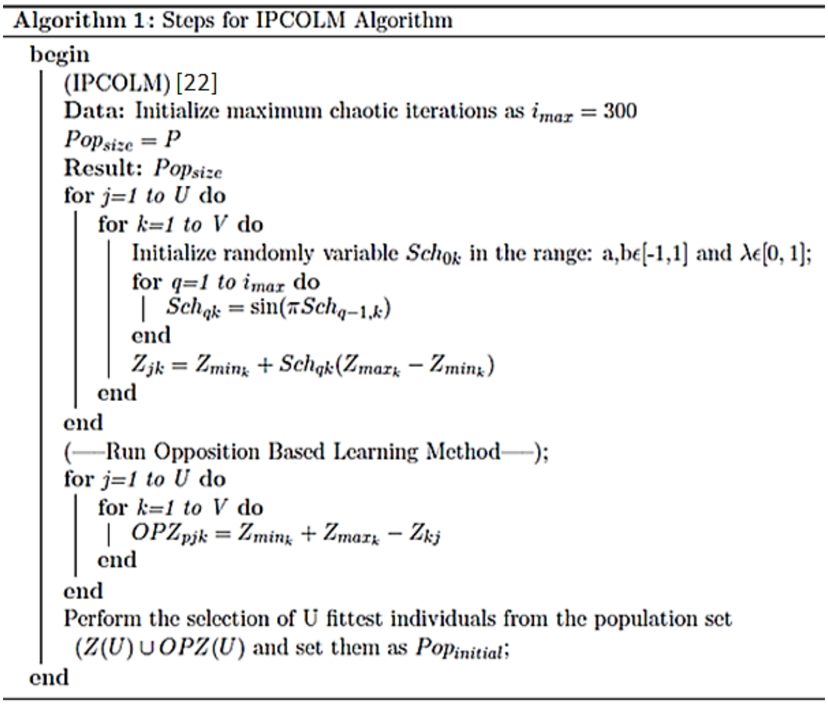
For parameters adjustment of DEP model and predicting values of the model, the below-mentioned stages are followed in DEP (CMGA) process.

A. *Initialization of Population using Chaotic Opposition based Learning Method (IPCOLM)*

Initial population values always affect the convergence speed and the likely predicted output of the evolutionary algorithms. A better convergence speed can save the iterations of algorithms from getting trapped in local optima problem. Hence, instead of using a random initialization of the population using chaotic opposition based learning method also represented as IPCOLM has been used for gathering more likable output. The sensitivity and randomness are directly dependent on conditions defined initially by chaotic maps. These maps can be utilized to initialize the initial population for increasing the population diversity through extraction of search space information. After using this approach, the CMGA algorithm's convergence speed also gets increased. For this approach, the sinusoidal iterator has been used which is given in the Eq. (25) [22].

$$Sch_{i+1} = \sin(\pi Sch_i), Sch_i \in [0,1], i = 0,1..i_{max} \quad (25)$$

Here, i denotes the count of iterations and i_{max} denotes the maximum possible chaotic iterations. The steps for IPCOLM can be referred from Algorithm 1. (referred as Fig. to 2).



```
Algorithm 1: Steps for IPCOLM Algorithm
begin
    (IPCOLM) [22]
    Data: Initialize maximum chaotic iterations as $i_{max} = 300$
    $Pop_{size} = P$
    Result: $Pop_{size}$
    for j=1 to U do
        for k=1 to V do
            Initialize randomly variable $Sch_{0k}$ in the range: a,bϵ[-1,1] and λϵ[0, 1];
            for q=1 to $i_{max}$ do
                $Sch_{qk} = \sin(\pi Sch_{q-1,k})$
            end
            $Z_{jk} = Z_{min_k} + Sch_{qk}(Z_{max_k} - Z_{min_k})$
        end
    end
    (—Run Opposition Based Learning Method—);
    for j=1 to U do
        for k=1 to V do
            $OPZ_{pjk} = Z_{min_k} + Z_{max_k} - Z_{kj}$
        end
    end
    Perform the selection of U fittest individuals from the population set
     $(Z(U) \cup OPZ(U))$ and set them as $Pop_{initial}$;
end
```

Fig.2. Steps for IPCOLM Algorithm

The process of CMGA starts a loop which consists of the stages for an objective which is minimizing the fitness function$ff: \mathbb{R}^n -> R$. This is defined in the Eq. (23). The dimensionality defined for weight vector of the DEP model is denoted by h and it can be computed as 2h+1. The loop consists of the process of selection followed by the genetic operators such as crossover and section. The crossover operator and mutation operator is applied on a pair of the parent of chromosomes opted from the selection process for further reproduction of new off-springs. Primary focus here is to select the best individual from the given population. Let the value of $Pop_{size} = 10$ [14, 21, 22, 23, 24].

B. Process of Selection

For going through the genetic operations, two numbers of chromosomes are selected from the population. For reproduction of new off-springs. For this spinning, the roulette wheel technique has been applied. Better child chromosomes are said to be produced by the parents with high potential. The chromosomes which will have a better fitness value would have more chance of getting selected. The selection method of the chromosome C_j, can be executed by using the probability P_j. Follow the given Eq. (26) [20, 23].

$$p_j = \frac{ff(C_j)}{\sum_{k=1}^{Pop_{size}} ff(C_k)}, \text{ j =1,2,...,} Pop_{size} \quad (26)$$

Cumulative probability CP_j of the chromosomes C_j is determined in the Eq. (27).

$$\widehat{CP_j} = \sum_{k=1}^{j} CP_k, j = 1,2,\dots,Pop_{size} \quad (27)$$

The process of selection starts by generating a random number$d \in |0,1|$ which is a non-zero number (floating point). The chromosome $C_i$only selected, if and only if the following condition given in Eq. (28) is satisfied:

$$CP_{j-1} < d \leq CP_{j-1}, (CP_0 = 0) \quad (28)$$

The observation that is done through this process of selection is that, the chromosomes which have a greater $ff(C_j)$ will possess a greater opportunity for getting chosen. Therefore the best chromosomes among all the other chromosomes will generate number of offspring and the average would live while the worst ones will die. The genetic operations only undergoes in the selection process on two chromosomes only, which are chosen.

VI. Genetic Operations

New off-springs are created by applying the genetic operation (operator) on the selected parent chromosomes. The genetic operation consists of a process of crossover and mutation process which are described below:

A. *Process of Crossover*

The crossover operation is utilized for helping in the exchange of information between the two selected parents (Vectors $\mathrm{W}_\mathrm{a}^{(\mathrm{g})}$, $\mathrm{W}_\mathrm{b}^{(\mathrm{g})} \in \mathbb{R}$, where a and b are indices in the range [1, P]), with the help of the roulette wheel approach. For the recombining process, the crossover operators are utilized, which help in the reproduction of four new child chromosomes or offspring ($offs_1$, $offs_2$, $offs_3$, $offs_4 \in \mathbb{R}^h$) which is mentioned in the Eq. (29), (30), (31) & (32) given as follows [14, 20]:

$$offs_1 = \frac{\mathrm{W}_\mathrm{a}^{(\mathrm{g})}+\mathrm{W}_\mathrm{b}^{(\mathrm{g})}}{2} \tag{29}$$

$$offs_2 = \max\left(\mathrm{W}_\mathrm{a}^{(\mathrm{g})}, \mathrm{W}_\mathrm{b}^{(\mathrm{g})}\right)u + (1-u)w_{max} \tag{30}$$

$$offs_3 = \min\left(\mathrm{W}_\mathrm{a}^{(\mathrm{g})}, \mathrm{W}_\mathrm{b}^{(\mathrm{g})}\right)u + (1-u)w_{min} \tag{31}$$

$$offs_4 = \frac{u\left(W_a^{(g)}+W_b^{(g)}\right)+(1-u)(w_{max}+w_{min})}{2} \tag{32}$$

The vectors min($\mathrm{W}_\mathrm{a}^{(\mathrm{g})}$, $\mathrm{W}_\mathrm{b}^{(\mathrm{g})}$) and max($\mathrm{W}_\mathrm{a}^{(\mathrm{g})}$, $\mathrm{W}_\mathrm{b}^{(\mathrm{g})}$) are used for denoting the minimum and maximum of individual respectively. The crossover weight is given by $u \in [0,1]$. Here the value of u is 0.9 (Closer value to 1 represents the direct contribution of parents to the reproduction process using crossover). The representation of the vectors having min and the max possibility of the values of the gene are done by W_{min} & w_{max} respectively [14, 20].

After the conclusion of the crossover, process produce, the fitness of each newly generated off-spring is calculated and off-spring having best fitness value (i.e. least value) will be selected as best of all. The representation of the resultant best crossover off-spring is done by the vector $offs_{best} \in \mathbb{R}^n$, which then performs the replacement process by replacing the individual which comprises the worse fitness in the given population (i.e. $W_{worse}^{(g)}$with the $worse \in [1, P]$ [14, 20].

B. *Process of Mutation*

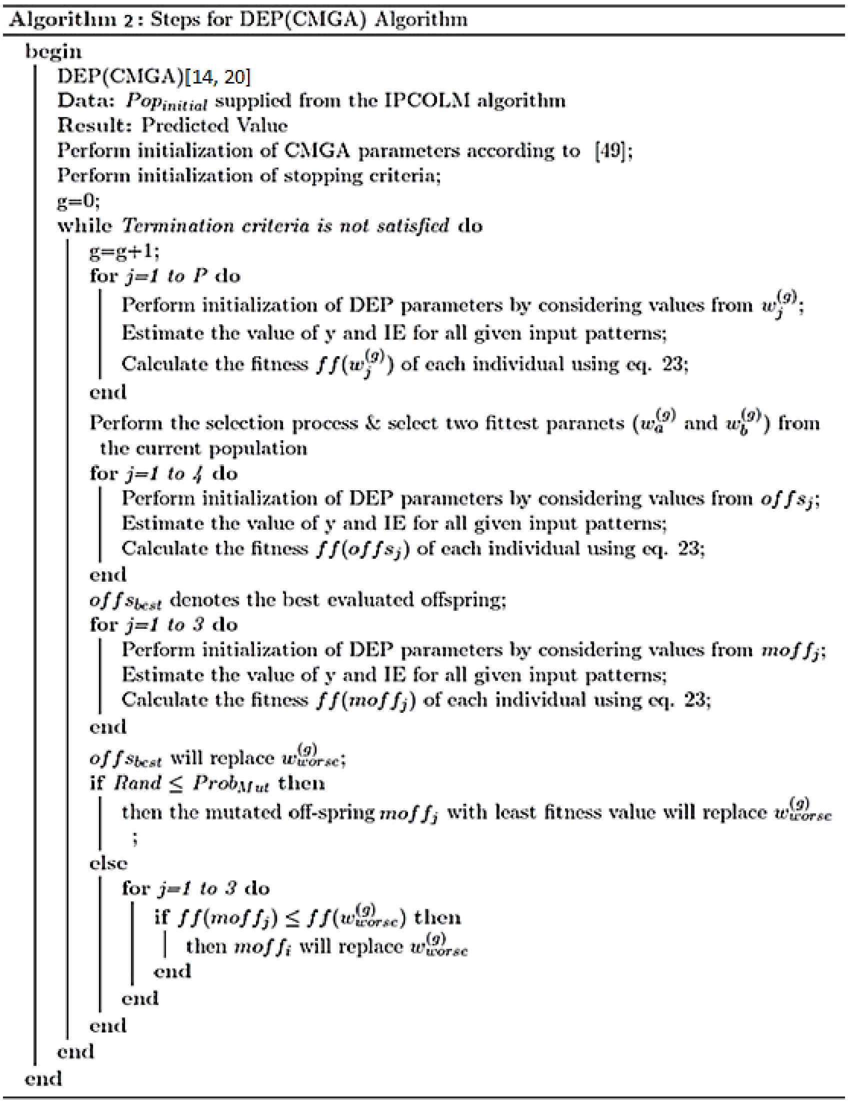
Let set the $Prob_{Mut} = 0.1$ as probabithe lity of mutation. Then$mo_1, mo_2, mo_3, \in \mathbb{R}^h$ are the three new mutated offspring that are created by using$offs_{best}$ and can be computed as [14, 20]:

$$moff_j = offs_{best} + t_j v_{j,} \quad j = 1,2,3 \tag{33}$$

Vector t_i satisfies the inequalities$W_{min} \leq offs_{bes}$t $+ t_i \leq W_{max,}$ where i=1, 2, 3. The range of vectorv_i is chosen among[0,1]. It also satisfies following condition: vector v_1 comprise only one non-zero chosen entry (selected randomly), v_2 has a randomly selected binary non-zero entry and $v_3$has constant vector value 1. After the production of a a newly mutated off-springs, again the fitness of all three has been calculated and then for association of them withthe in the population fothe the llowing scheme has been takenwith the help of a randomly generating numbers (Rand) from the range of [0,1] : If Rand $\leq Prob_{Mut}$ then the replacement of $W_{worse}^{(g)}$within population range i.e. [1, P] is done with the mutated offspring having least fitness value, else the mentioned steps are being followed $for\, j = 1,2,3 :$ $W_{worse}^{(g)}$if the fitness value of the $mo_j \leq W_{worse}^{(g)}$ then the mo_j will replace the $W_{worse}^{(g)}$ in order to form the new population [14, 20].

Moreover, the stopping criteria used for the proposed DEP (CMGA) process is given as: (i). $CMGA_{gen} = 10000$ which is the number of maximum generation (ii). $TEP \leq 10^{-6}$ which is training error process of fitness function.

The steps of DEP (CMGA) process is given in Algorithm 2. (can be referred to as Fig. 3):



```
Algorithm 2: Steps for DEP(CMGA) Algorithm
begin
    DEP(CMGA)[14, 20]
    Data: $Pop_{initial}$ supplied from the IPCOLM algorithm
    Result: Predicted Value
    Perform initialization of CMGA parameters according to [49];
    Perform initialization of stopping criteria;
    g=0;
    while Termination criteria is not satisfied do
        g=g+1;
        for j=1 to P do
            Perform initialization of DEP parameters by considering values from $w_j^{(g)}$;
            Estimate the value of y and IE for all given input patterns;
            Calculate the fitness $ff(w_j^{(g)})$ of each individual using eq. 23;
        end
        Perform the selection process & select two fittest paranets ($w_a^{(g)}$ and $w_b^{(g)}$) from
         the current population
        for j=1 to 4 do
            Perform initialization of DEP parameters by considering values from $offs_j$;
            Estimate the value of y and IE for all given input patterns;
            Calculate the fitness $ff(offs_j)$ of each individual using eq. 23;
        end
        $offs_{best}$ denotes the best evaluated offspring;
        for j=1 to 3 do
            Perform initialization of DEP parameters by considering values from $moff_j$;
            Estimate the value of y and IE for all given input patterns;
            Calculate the fitness $ff(moff_j)$ of each individual using eq. 23;
        end
        $offs_{best}$ will replace $w_{worse}^{(g)}$;
        if $Rand \leq Prob_{Mut}$ then
            then the mutated off-spring $moff_j$ with least fitness value will replace $w_{worse}^{(g)}$
             ;
        else
            for j=1 to 3 do
                if $ff(moff_j) \leq ff(w_{worse}^{(g)})$ then
                    then $moff_i$ will replace $w_{worse}^{(g)}$
                end
            end
        end
    end
end
```

Fig.3. Steps for DEP (CMGA) Algorithm

VII. Performance Measures

For the performance evaluation of the proposed system after successfully implementing the idea, this paper has used three popular measures of performance (i.e. mean magnitude of relative error, prediction and evaluation function) which have been utilized as a benchmark [14].

Literature for evaluation of prediction comprises of various criteria of performance. Mean squared error (MSE) is quite popular among them for providing a prediction model for the direction. Another application of MSE to estimate the comparisons between models and processes alternatively. But MSE is not considered as a reliable performance metric because of its lack of scale preserving linear transformation and non-singular's invariance model lacking [14, 25, 26].

A. Prediction Accuracy (PRED(s)):

The percentage of prediction is represented by PRED(s) which can be determined by the following Eq. (34) & (35) [14, 25].

$$PRED(s) = \frac{100}{N}\sum_{j=1}^{N} S_j \tag{34}$$

Here, N determines the number of input patterns.

$$S_j = \begin{cases} 1, & if\left(MMRE_j\right) < \frac{s}{100} \\ 0, & Otherwise \end{cases} \tag{35}$$

B. Mean Magnitude of Relative Error (MMRE):

Literature shows that MMRE is the first metric which is able to detect the model deviation precisely which can be calculated as Eq. 36 [14, 26]:

$$MMRE = \frac{1}{N}\sum_{j=1}^{N}\frac{|Target_j - Predicted_j|}{Target_j} \tag{36}$$

Where N determines the number of input patterns and Target represents the output which is being predicted.

C. Evaluation Function (Eval$_f$):

Out of combining PRED(s) and MMRE, a more robust and consistent global indicator of performance has been created which is termed as evaluation function (also represented as Eval$_{f)}$.and it can be calculated as Eq. (37). [14].

$$Eval_f = \frac{PRED(s)}{(1+MMRE)} \quad (37)$$

VIII. Simulation and Experimental Analysis

The proposed model was tested on two different datasets. The first dataset is proposed by the Samaresh Mishra et. al (referred as dataset 1) [7] and it is built over five different student projects (small size projects) while the second data set is a self-developed dataset. Both the datasets consists of the following attributes: total entity size estimation (TESE), total relationship size estimation (TRSE), total semantic integrity constraint size (TSICS), the total database size of ERD (TDSERD), actual effort in person-hour (AE-PH).

Out of these attributes TESE, TRSE, TSICS, and TDSERD is taken as input to the CMGA-DEP model. The second dataset which is self-developed can be referred from the Table 3.

Table 3. Self-Developed Dataset

Project No.	TESE	TRSE	TSICS	TDSERD	AE
1	28	18	95	141	67.791
2	72	14	123	209	115.683
3	28	4	72	104	70.358
4	34	7	108	149	109.106
5	39	9	112	160	110.09

For maintaining the large variations of predictions and getting closer values, the normalization of the datasets have been carried out for which the datasets have been divided into three distinct sets where training set includes 50% of data, validation set includes 25% of data and test set includes 25% of data. The values of a, b were initialized in the interval $[-1,1]$ and λ in the range $[0,1]$ and given as initial input to the IPCOLM algorithm. The method of leave one out cross validation (LOOCV) has been used for generalization of error [14].

As very few works are available to provide a more robust comparison, the work is compared with only one existing technique (effort estimation model through ERD (EMERD)) which is a mathematical model of estimation based on complexity and size of ERD. It can be observed from the table 4. (for dataset 1) the value for EMERD technique is $eval_f \cong 31.4960$. But proposed the DEP (CMGA) model outperforms the EMERD model with $Eval_f = 85.3232$. The proposed DEP (CMGA) model has better performance values in all three performance metrics, $PRED$ (25), MMRE and $Eval_f$ with values 95.69, 0.1215 and 85.3232 respectively. There was an massive improvement of 139.2250%, 55.00% and 170.9017% with respect to $PRED$ (25), MMRE and $Eval_f$ in the DEP (CMGA) model over the EMERD model which is observed as only the existing model in the similar working environment. Fig. 4. shows the prediction results for individual projects considered for dataset 1.

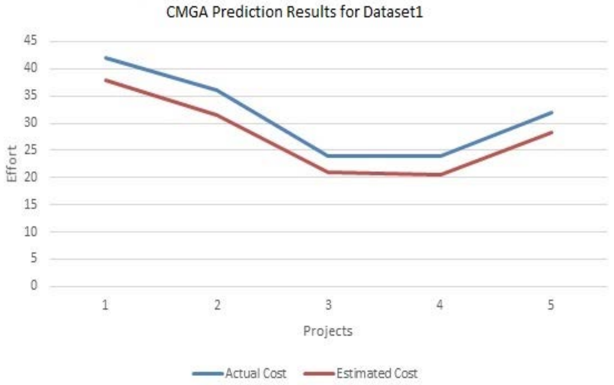


Fig.4. CMGA Prediction Results for Dataset1

Table 4. Prediction Results For Dataset 1

Model Type	PRED(25)	MMRE	Eval$_f$
EMERD [4]	40.00	0.2700	31.4960
CMGA (DEP)	95.69	0.1215	85.3232

Table 5. shows the prediction results for the self-developed dataset (referred to as dataset 2). Table 5. Shows that EMERD achieved only $eval_f \cong 14.5623$. But proposed the DEP (CMGA) model outperforms the EMERD model with $Eval_f \cong 84.1918$. The proposed DEP (CMGA) model has better performance values in all three performance metrics. $PRED$ (25), MMRE and $Eval_f$ with values 95.12, 0.1298 and 84.1918 respectively. There was a massive improvement of 375.60%, 65.2383% and 478.1490% with respect to PRED (25). MMRE and $Eval_f$ in the DEP (CMGA) model over the EMERD model which is observed as the only existing model in the similar working environment. Fig 5. shows the prediction results for individual projects considered for dataset 2.

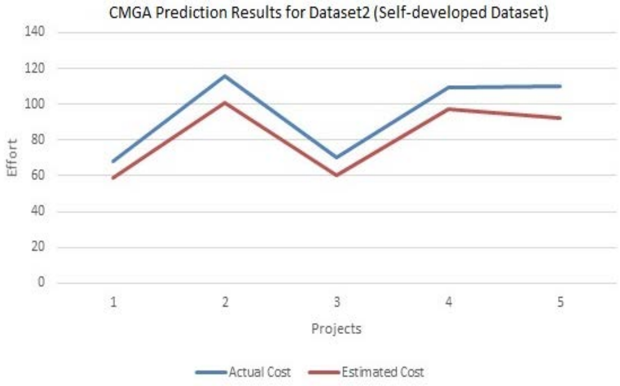


Fig.5. CMGA Prediction Results for Dataset2 (Self-developed Dataset)

Table 5. Prediction Results for Dataset 2

Model Type	PRED(25)	MMRE	$\mathrm{Eval_f}$
EMERD [4]	20.00	0.3734	14.5623
CMGA (DEP)	95.12	0.1298	84.1918

IX. Conclusion

In this paper, authors have manifested a chaotically modified evolutionary morphological approach to solve the SDEE problem. The evolutionary CMGA algorithm has been applied as a DEP (CMGA) model for parameter optimization of DEP perceptron. This leads to a more precise solution towards SDEE problem.

For the performance of the proposed model, two different performance metrics i.e. MMRE and PRED (25) have been used. A global indicator (evolution function ($Eval_f$)) has been created for a more robust and consistent measure of performance by using these two statistic measures of error. An empirical validation of the proposed model has been carried out using one existing dataset (referred as dataset1) and a self-developed dataset (referred as dataset2), displaying the strength and stability of the model which has been proposed in this paper, through an analytical comparison of existing work found in the literature (i.e. EMERD). The experimental validation confirms a more consistent and improved global performance of DEP (CMGA) model, having around 324.0507% of massive average improvement over EMERD model which is the only existing model available.

The DSEBER model helped in identifying the higher priorities and important factors of ER and EER diagrams of a system.

The proposed model's major advantages which possibly notice other than its excellent predictive capability in comparison to EMERD model are: it consists of some small components which are nonlinear (i.e. a distinct amount of nonlinear components can be used by the model with respect to the problem). That makes the model computationally simpler and reduces the complexity.

X. Future Work

More development studies would be considered with the model which has been proposed, as a part of future work along with the other regression and classification problems (such as maintainability and fault prediction) in software engineering.

References

[1] M. Padmaja, D. Haritha. "Software Effort Estimation Using Grey Relational Analysis", *International Journal of Information Technology and Computer Science (IJITCS)*, ISSN: 2074-9015, Vol. 9, Issue: 5, pp. 52-60, May-2017.

[2] S. Goyal, A. Parashar, "Machine Learning Application to Improve COCOMO Model using Neural Networks", *International Journal of Information Technology and Computer Science*, Vol. 3, pp. 35-51, March-2018.

[3] S. Bilgaiyan, S. Sagnika, S. Mishra, M N. Das, "A Systematic Review on Software Cost Estimation in Agile Software Development", *Journal of Engineering Science and Technology Review*, ISSN: 1791-2377, Vol. 10, Issue: 4, pp. 51-64, September-2017.

[4] P. Pospieszny, B. Czarnacka-Chrobot, A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms", *The Journal of Systems and Software*, Vol. 137, pp. 184–196, March-2018.

[5] Y. Zao, H B K. Tan, W. Zhang, "Software Cost Estimation through Conceptual Requirement", *In Third International Conference on Quality Software, IEEE*, pp. 141-143, November-2003.

[6] G J. Kennedy, "Elementary Structures in Entity-Relationship Diagram: A New Metric of Effort Estimation", *In International Conference on Software Engineering: Education and Practice. IEEE*, pp. 86-92, January-1996.

[7] S. Mishra, P. Pattnaik, R. Mall, "Early Estimation of Back-End Software Development Effort", *International Journal of Computer Applications*, ISSN: 0975–8887, Vol. 33, Issue: 2, pp. 6-11, November-2011.

[8] S. Mishra, R. Mall, "Estimation of Effort Based on Back-End Size of Business Software Using ER Model", In 2011 *World Congress on Information and Communication Technologies, IEEE*, ISSN: 2074-9015, pp. 1098-1103, December-2011.

[9] B. Jamil, J. Ferzund, A. Batool, S. Ghafoor, "Empirical Validation of Relational Database Metrics for Effort Estimation", *In 6th International Conference on Networked Computing, IEEE*, ISSN: 2074-9015, pp. 1-5, May-2010.

[10] S. Parida, S. Mishra, "Review report on Estimating the Back-End Cost of Business Software Using ER- Diagram Artifact", *International Journal of Computer Science and Engineering Technology (IJCSET)*, ISSN: 2229-3345, Vol. 2, Issue: 1, pp. 233-238, March-2014.

[11] S. Mishra, E. Aisuryalaxmi, R. Mall, "Estimating Database Size and its Development Effort at Conceptual Design Stage", *In Global Trends in Information Systems and Software Applications. Springer*, Vol. 270, Issue: 1, pp. 120-127, 2012.

[12] S. Mishra, K C. Tripathy, M K. Mishra, "Effort Estimation Based on Complexity and Size of Relational Database System", *International Journal of Computer Science and Communication*, ISSN: 2074-9015, Vol. 1, Issue: 2, pp. 419-422, July-December-2010.

[13] S. Bilgaiyan, K. Aditya, S. Mishra, M. Das, "A Swarm Intelligence based Chaotic Morphological Approach for Software Development Cost estimation", International Journal of Intelligent Systems and Applications, ISSN: 2074-9058, Vol. 10, Issue: 9, pp. 13-22, September-2018.

[14] R de A. Araujo, A L I. Oliveira, S. Soaresand, S. Meira, "An Evolutionary Morphological Approach for Software Development Cost Estimation" *Neural Networks. Elsevier*, Vol. 32, Issue: 1, pp. 285-291, August-2012.

[15] P. Sussner, E L. Esmi, "Morphological Perceptrons with Competitive Learning: Lattice-Theoretical Framework and Constructive Learning Algorithm", *Information Sciences, Elsevier*, Vol. 181, Issue: 10, pp. 1929-1950, May-2011.

[16] R de A. Araujo, S. Soares, A L I. Oliveira, "Hybrid Morphological Methodology for Software Development

Cost Estimation", *Expert Systems with Applications, Elsevier*, Vol. 39, Issue: 6, pp. 6129-6139, May-2012.

[17] A L I. Oliveira, P L. Braga , R M F. Lima, M L. Cornélio, "GA-based Method for Features Selection and Parameters Optimization for Machine Learning Regression applied to Software Effort Estimation", *Information and Software Technology, Elsevier*, Vol. 51, Issue: 11, pp. 6129-6139, November-2010.

[18] G J F. Banon, J. Barrera, "Decomposition of Mappings between Complete Lattices by Mathematical Morphology, part 1. General lattices", *Signal Processing, Elsevier*, Vol. 30, Issue: 3, pp. 299–327, February-1993.

[19] S. Bilgaiyan, K. Aditya, S. Mishra, M N. Das, "Chaos-based Modified Morphological Genetic Algorithm for Software Development Cost Estimation", *Progress in Computing, Analytics and Networking*, Vol. 710, pp. 31-40, April-2018.

[20] F. Leung, H. Lam, S. Ling, "Tuning of the Structure and Parameters of a Neural Network Using an Improved Genetic Algorithm", *IEEE Transactions on Neural Networks*, ISSN: 1045-9227, Vol. 14, Issue: 1, pp. 79-88, February-2003.

[21] S. Bilgaiyan, S. Sagnika, S. Mishra, M N. Das, "Study of Task Scheduling in Cloud Computing Environment Using Soft Computing Algorithms", *International Journal of Modern Education and Computer Science*, ISSN: 2075-017X, Vol. 7, Issue: 3, pp. 32-38, March-2015.

[22] W F. Gao, S Y. Liu, L L. Huang, "Particle Swarm Optimization with Chaotic Opposition-Based Population Initialization and Stochastic Search Technique", *Communications in Nonlinear Science and Numerical Simulations. Elsevier*, Vol. 17, Issue: 11, pp. 4316-4327, November-2012.

[23] A. Hussain, Y. S. Muhammad, M. N. Sajid, "An Efficient Genetic Algorithm for Numerical Function optimization with Two New Crossover Operators", International Journal of Mathematical Sciences and Computing, ISSN: 2310-9025, Vol. 4, Issue. 4, pp. 41-55, November-2018.

[24] A. Raha, M. K. Naskar, et al. "A Genetic Algorithm Inspired Load Balancing Protocol for Congestion Control in Wireless Sensor Networks using Trust Based Routing Framework (GACCTR)", International Journal of Computer Network and Information Security, ISSN: 2074-9090, Vol. 9, Issue. 9, pp. 9-20, July-2013.

[25] R de A. Araujo, A L I. Oliveira, S. Soares, "A Shift-Invariant Morphological System for Software Development Cost Estimation" *Expert Systems with Applications, Elsevier*, ISSN: 2074-9015, Vol. 38, Issue: 4, pp. 4162-4168, April-2011.

[26] M P. Clements, D F. Hendry, "On the Limitations of Comparing Mean Square Forecast Errors" *Journal of Forecasting*, ISSN: 2074-9015, Vol. 12, Issue: 8, pp. 617-637, December-1993.

Authors' Profiles

Saurabh Bilgaiyan is currently working as a Teaching Associate in KIIT deemed to be University, Bhubaneswar, India. He is also pursuing Ph.D. (CSE) at KIIT, deemed to be University, Bhubaneswar, India. He obtained his Bachelor's degree of B.E. (I.T.) in 2012 from B.I.R.T., Bhopal, India and Master's degree of M.Tech. (CSE) in 2014 from KIIT, deemed to be University, Bhubaneswar, India His area of interests includes soft computing, cloud computing, image processing, distributed database systems and software engineering.

Dhiraj Kumar Goswami is currently pursuing Dual Degree (B.Tech & M.Tech) (CS&E) in KIIT, deemed to be University, Bhubaneswar, India. His area of interests includes soft computing and software engineering.

Samaresh Mishra M.Tech, Ph.D. (Computer Science) is the Dean of School Computer Engineering, KIIT, deemed to be University, Bhubaneswar, India. He has a rich experience of teaching in the field of Computer Science at both UG and PG degree level and has also published many articles in International Conferences and Journals of repute. His area of interest includes Database Engineering, Cost Estimation, Software Reliability and Cloud Computing.

Madhabananda Das M.Tech., Ph.D. is a Senior Professor at the School Computer Engineering, KIIT, deemed to be University, Bhubaneswar, India. He has a rich experience of teaching in the field of Computer Science at both UG and PG degree level and has also published many articles in International Conferences and Journals of repute. His area of interest includes soft computing, computational intelligence and image processing.

How to cite this paper: Saurabh Bilgaiyan, Dhiraj Kumar Goswami, Samaresh Mishra, Madhabananda Das, "Effort Estimation of Back-end Part of Software using Chaotically Modified Genetic Algorithm", International Journal of Intelligent Systems and Applications(IJISA), Vol.11, No.1, pp.32-42, 2019. DOI: 10.5815/ijisa.2019.01.04