Received: April 29, 2024. Revised: June 12, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

COCOMO II-FG-HGWO: A Modified Hunting Gray Wolf Optimization for Improving Constructive Cost Model II Performance

Rahmi Rizkiana Putri[1] **Daniel Siahaan**[1]* **Chastine Fatichah**[1]

[1]*Department of Informatics, Institut Teknologi Sepuluh Nopember Surabaya, Indonesia*
* Corresponding author's Email: daniel@if.its.ac.id

Abstract: Software effort estimation presents a significant challenge in the domain of project management. The Constructive Cost Model II (COCOMO II) is frequently applied to estimate the required software project effort. Our study proposed a Constructive Cost Model II with Fuzzy Gaussian and Hunting Grey Wolf Optimization (COCOMO II-FG-HGWO), which improves the accuracy of COCOMO II. Furthermore, the proposed method applies FG to obtain more optimal values of 11 COCOMO II effort multipliers. The GWO hunting mechanism is modified by adding tournament selection to obtain optimal alpha from populations by optimizing COCOMO II coefficients A and B. It helps each alpha explore their search abilities, thereby reducing the risk of being trapped in local optima. The experimental results show that the proposed method reduces MMRE by 0.01% on the NASA 60 dataset and achieves lower values by more than 16% compared to COCOMO II. This indicates that estimated effort is closer to actual effort, and the risk of error in calculating project costs becomes smaller and, in turn, improves the quality of software projects.

Keywords: COCOMO II, Cost estimation, FG, GWO, Hunting mechanism.

1. Introduction

In the last ten years, there has been significant progress in the software industry, leading to its fundamental role in the achievements of numerous multinational corporations [1]. Effective software operation within cost, effort, and time limitations has become essential. Both software practitioners and academic researchers are constantly investigating various strategies aimed at maintaining and improving productivity levels in the domain of software development and management. The background of our research begins with the process of efficiently resolving cost, time, and labor concerns, which is of the highest priority during the progression of a novel software project. Nevertheless, there has been a substantial increase in the costs related to software development, creating substantial obstacles for enterprises.

Consequently, precise estimation of project effort and time has become increasingly important. It is common for software development projects to exceed their initial budgets and timelines [2]. Software effort estimation (SEE) is a crucial methodology for predicting the time and resources required to create a software system. It plays a vital role in the success of software project management and other related tasks [3]. When estimating the effort required for a software project, it is important to consider various factors, including the project's duration, related expenses, and the required number of employees.

Cost estimation is a pivotal and vital factor in any software development project. It can be described as the systematic evaluation of effort, wherein estimation is conducted with regard to the number of resources necessary to complete project tasks, resulting in the delivery of a product or service that fulfills both functional and non-functional specifications while also satisfying the customer's needs [4]. The estimation of software costs involves an evaluation of numerous factors related to software development due to the fact that each factor can substantially impact the cost of software development. During the initial stages of development, companies can benefit from precise estimates. The project management process also

includes the cost estimation phase as a critical component of software development initiatives. In this phase, the level of effort is assessed and the quantity of resources necessary to accomplish every software development task is estimated. Furthermore, it is essential to provide the consumer with the required functionalities or capabilities in order to fulfill all the requirements.

Furthermore, the assessment of software quality is also possible by considering significant attributes associated with the static and dynamic properties of the system as well as the outcomes of methodical interactions [5]. Our research problem formulation includes the existence of main causes such as project failure, poor requirements, inadequate resources, unrealistic schedules, poor planning, unidentified risks, lack of management commitment, inappropriate use of technology, inappropriate project objectives, unrealistic or unarticulated, inaccurate estimates, poorly defined system requirements, poor reporting status, and unmanaged risks. In addition, specific characteristics of a project, organization, or external environment also contribute to project failure [6]. Within software projects, interrelated tasks occur synchronously or in succession. Boehm states that efforts related to requirements specifications occupy 5% of the time required to complete a project, while the allocation of approximately 8% of resources goes to creating standards documents. This implies that successful software development projects share common practices regarding allocating resources for the process of program requirements specification. Around 7-15% of available resources are allocated to activities that prioritize software requirements specification.

Moreover, evaluating Constructive Cost Model II (COCOMO) II attributes provides an overview of project-specific characteristics. Several characteristics that are taken into account in prior models include (1) project dimensions (size, complexity, and duration), (2) instability in requirements, (3) project category, (4) safety level, and (5) cost limitations. A large number of inputs are additionally accessible for each attribute through the application of the COCOMO II model. However, the accuracy value of COCOMO II is considered less accurate because there is still quite a large difference between the actual project cost and the project cost estimate (MMRE) results from COCOMO II.

Meta-heuristic algorithms are employed in an attempt to enhance forecasting results and obtain more precise estimations. Typically, genetic algorithms are incorporated into conventional optimization procedures. Integrating genetic algorithms and standard effort estimation models has yielded enhanced estimates. Khan *et al.* [7] introduced the Grey Wolf, Deep Neural Network, and Strawberry (GWDNNSB) algorithm to estimate software effort. The technique was tested using the NASA, COCOMO 81, Maxwell dataset. The model they proposed consists of Use Case Point (UCP), Expert Judgment (EJ), and Artificial Neural Network (ANN). Combines the efforts of each unique base model using linear combination rules. To validate the model's effectiveness, they applied it to a benchmark dataset, the International Software Benchmarking Standards Group (ISBSG), using three different variations to avoid bias. Next, the trained model will be applied to industrial use cases for cross-validation. Their research yielded better estimates. Their research proposed a heterogeneous ensemble effort estimation model that helps effective software development, especially in project cost and effort estimation [8]. The proposed model works in a two-stage environment based on a testing stage and a training stage. In the BABE training stage, the most appropriate weights are calculated and then used in the testing stage to evaluate the accuracy of the BABE model estimates. This research uses six real datasets, including the performance MMRE, PRED (0.25), and SA. Based on the SA values calculated for this model, it can be concluded that the proposed model is more accurate than the existing development estimation model [9].

Non-heuristic approaches are classified into various categories: dynamic programming, integer programming, graph theory, etc. Hidayat *et al.* [10] tested the results using NASA datasets. Testing is conducted by means of parameter testing and accuracy testing. The computational efficiency of the suggested approach is compared to the Ant Colony Optimization (ACO) method in terms of accuracy. In contrast to current optimization techniques, the cost and effort calculations of the COCOMO II model were inadequate. In light of the benefits of optimization methods and the comprehensiveness of COCOMO II attributes, this study suggests combining COCOMO II and GWO to increase the accuracy of software project costs and efforts.

Our study was mainly motivated by the following two research gaps. First, in relation to previous studies concerning optimization-based software effort estimation that requires better precision, it is important to create a COCOMO II framework that implements optimization techniques for achieving enhanced effort estimations. Results produced by the COCOMO II model using its original cost driver and effort multipliers on publicly available datasets (e.g., NASA and Turkish) have yet to be shown to be

inadequate. In addition, the cost drivers in the NASA 60 do not match the COCOMO II cost drivers, affecting the results' accuracy. Modifications to the values of the effort multipliers of COCOMO II and adjustments to its cost drivers are needed.

Secondly, in a number of past studies that employed the GWO algorithm, every individual was used in an effort to achieve the most optimal results. In the hunting mechanism, the best individual assumes the role of leader due to its position closest to the prey, and the other individuals will follow accordingly. However, this has been shown to yield solutions that are not optimal. Thus, research must be conducted in order to identify the optimal values displayed by the alpha individuals.

Therefore, this research aims to improve the accuracy of COCOMO II in estimating software project costs by proposing a COCOMO II-FG-HGWO method. The method applies Gaussian Membership Functions to change 11 COCOMO II quantitative Effort Multiplier (EM) values. Change the COCOMO II coefficient A and B values using the Gaussian Membership Function. GWO method is used to optimize the hunting GWO coefficient a, A, and C values in GWO by modifying the hunting mechanism using tournament selection.

The structure of this paper is as follows: Section 1 provides an overview of the problem's context. Section 2 details previous research relevant to software cost estimation and COCOMO II. However, we have enhanced our understanding through several studies pertaining to GWO. Section 3 elaborates on the experimental scenarios, datasets, proposed methodology, and evaluation. The results of the experimental investigation and performance comparison with previous research on COCOMO II and GWO are detailed in Section 4. Section 5 analyzes the experimental results. Section 6 presents the conclusion.

2. Related works

Several studies discussing software project cost estimates use COCOMO II modeling combined with optimization methods, such as Baiquni et al. [9], which utilized fuzzy logic and local calibration to enhance the precision of COCOMO II. Fuzzy logic with the Gaussian Membership Function, along with the integration of local calibration, was implemented. The obtained MMRE values ranged from 34% to 104%.

Effendi et al. [11] calculated COCOMO II coefficients A and B using the Bat algorithm, in which the optimal bat locations were obtained as the A and B constants. The proposed method achieved an MMRE of 34.25%. To optimize the parameters of the COCOMO model, Nandal et al. [12] utilized a combination of the Bat algorithm and GSA. The hunting and orienting capabilities of the bats in the Bat algorithm were enhanced by employing random movements during the exploration phase. GSA further improves the exploration phase because gravitational forces affect bat movement. Consequently, inaccuracies may be reduced by up to 46%. However, the Bat algorithm could be more effective when used to perform numerical operations with high precision and requires long training if the amount of data processed is large.

Langsari et al. [13] utilized Particle Swarm Optimization (PSO) to optimize COCOMO II model parameters, resulting in more precise and accurate development effort, cost, and time estimations. The experimental results showed an MMRE value of 34.19%. However, the PSO method may become stuck in a local optimum while trying to find a solution.

Sunindyo et al. [14] integrated the COCOMO II model with the K-Means clustering technique to enhance the estimation accuracy. K-Means clustering is used to determine the dataset used inside the COCOMO II calibration procedure. The proposed method decreased the COCOMO II MMRE value from 1.32 to 0.85 and enhanced the PRED value from 32% to 54%.

Fadhil et al. [15] presented an optimized estimation model that utilized two models in conjunction with the COCOMO II model. The first model implements the Dolphin algorithm, and the second model utilizes a hybrid Dolphin and Bat algorithm. On the NASA-93 dataset, an MMRE value of 50.27% and 14.57% were achieved for the NASA-93 and NASA-60 datasets, respectively.

Suherman et al. [16] examined the implementation of machine learning algorithms, namely SVR and Random Forest Regression, to adjust the parameters of the COCOMO II model. The MMRE value was reported to be 54%. However, machine learning requires data mining to provide accurate results. The results would also be accurate if incorrect or incomplete data were used. Developing complex machine learning algorithms can take a long time.

Hassan et al. [17] explored the use of a hybrid between the ACO and Bat algorithms to obtain optimized coefficients of the COCOMO II model for effort estimation. The obtained MMRE values were 2.72%, 3.47%, and 4.12% on the NASA, COCOMO 81, and KEMERER datasets. Ali et al. [18] proposed a model that consists of several base estimation models, namely Use Case Point (UCP), Expert

Judgment (EJ), and Artificial Neural Network (ANN). The efforts of each unique base model are combined using linear combination rules. A benchmark dataset, namely the International Software Benchmarking Standards Group (ISBSG) dataset, was applied to validate the model's effectiveness, using three different variations to avoid bias. For cross-validation, trained models were applied to industrial use cases. The proposed heterogeneous ensemble effort estimation model yielded enhanced estimates, which promotes a more effective software development process. However, ANN and UCP methods have several areas for improvement, namely inaccurate estimation results, ineffectiveness when used to carry out numerical operations with high precision, and long training required if the amount of data processed is large.

Meanwhile, several other studies used COCOMO II modeling combined with the GWO method without modification, illustrated in Fig. 1. Khan et al. [7] carried out other studies and proposed the GWDNNSB model that optimizes weights and learning rates using metaheuristic algorithms. The GWO algorithm is utilized for initial weight optimization, whereas SB is employed for learning rate optimization. The obtained MMRE value was 2.47%.

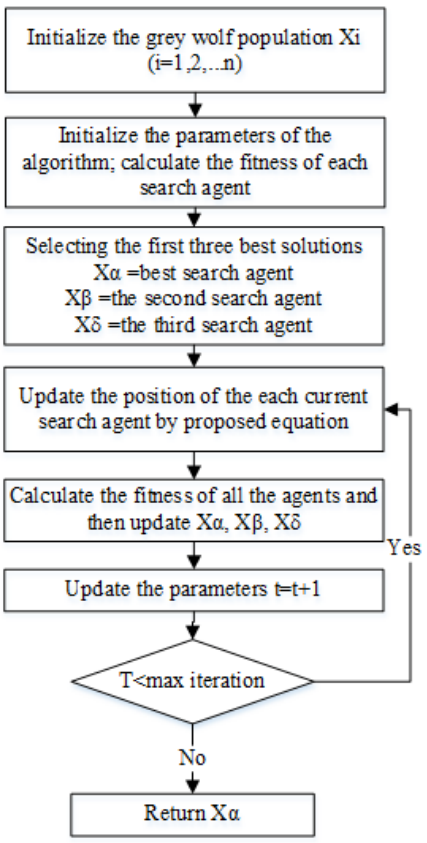

Figure. 1 GWO Algorithm

Al-sheikh et al. [19] applied existing GWO to enhance the coefficients of the basic COCOMO model and two COCOMO-based models with the NASA 18 dataset. Moreover, the performance of existing GWO in finding the optimal value for effort estimation was compared with that of metaheuristic algorithms, including Moth Flame Optimization (MFO), Zebra Optimization (ZOA), Prairie Dog Optimization (PDO), and White Shark Optimization (WSO). Several metrics, namely VAF, MSE, MAE, MMRE, RMSE, and R2, were used to evaluate the models. Based on the experimental results, it was established that GWO outperformed the other metaheuristic algorithms. They use GWO without modification, which can make it difficult to determine the best alpha individual because they also focus on finding solutions from beta and delta individuals. So, each individual's exploration is limited and vulnerable to being trapped in local optima.

Putri et al. [20] applied existing GWO and FG to obtain more optimal values of nine COCOMO II effort multipliers to enhance the coefficients of the basic COCOMO II model and with the NASA 93 dataset. Moreover, the performance of the existing GWO in finding the optimal value for effort estimation. They use GWO without modification, which can make it difficult to determine the best alpha individual because they also focus on finding solutions from beta and delta individuals. Therefore, each individual's exploration is limited and vulnerable to being trapped in local optima.

Putri et al. [21] applied extended Gamma GWO to enhance the coefficients of the basic COCOMO II model and with the NASA 93 dataset. The performance of extended Gamma GWO is not better than that of the previous population, namely alpha, beta, and delta. They suggested increasing the gamma population, which obviously did not result in a lower total error (MMRE). This is because the alpha individual, the best search agent in the GWO population, always achieves optimal results.

Based on previous researches, our research aims to improve the accuracy of COCOMO II in estimating software project costs by proposing a COCOMO II-FG-HGWO method. The method applies Gaussian Membership Functions to change the eleven COCOMO II quantitative Effort Multiplier (EM) values. Using the Gaussian Membership Function, the method also changes the A and B COCOMO II coefficient values. GWO method is used to optimize the hunting coefficient values a, A, and C in GWO by modifying the hunting mechanism using tournament selection. Also, the COCOMO II model requires adjustments to its cost

drivers when used on the NASA 60 benchmark dataset, and consequently, the effort multipliers also need to be optimized.

3. Materials and method

The architectural framework presented in this research is assessed and verified using publicly available datasets. The Turkish dataset used in this study consists of projects from the Turkish software sector. In addition, two NASA datasets, namely NASA 60 and NASA 93, are also used. The accuracy of cost estimations is measured using the MMRE metric. The initial stage is to prepare the dataset for effort estimation. This process of preparing the publicly available software projects in the dataset involves determining the cost driver values, which include effort multipliers and scale factors based on the respective value of each level. This task is performed manually. Once the dataset has been prepared, the subsequent task is to choose between the quantitative and qualitative effort multipliers. This is necessary since the 11 quantitative effort multipliers will undergo value modifications using the Gaussian Membership Function method.

Furthermore, in the proposed method, a modified hunting mechanism of the GWO method is applied to adjust the constant COCOMO II coefficient A and B model. The hunting mechanism involves selecting the alpha individual with the lowest fitness via tournament selection and selecting the minimum value among the best alpha individuals. The overall flow of our proposed method is illustrated in Fig. 2, as well as the respective values of each level. This task is performed manually. Once the dataset has been prepared, the subsequent task is to choose between the quantitative and qualitative effort multipliers. This is necessary since the eleven quantitative effort multipliers will undergo value modifications using the Gaussian Membership Function method. Furthermore, in the proposed method, a modified hunting mechanism of the GWO method is applied to adjust the constant values of the

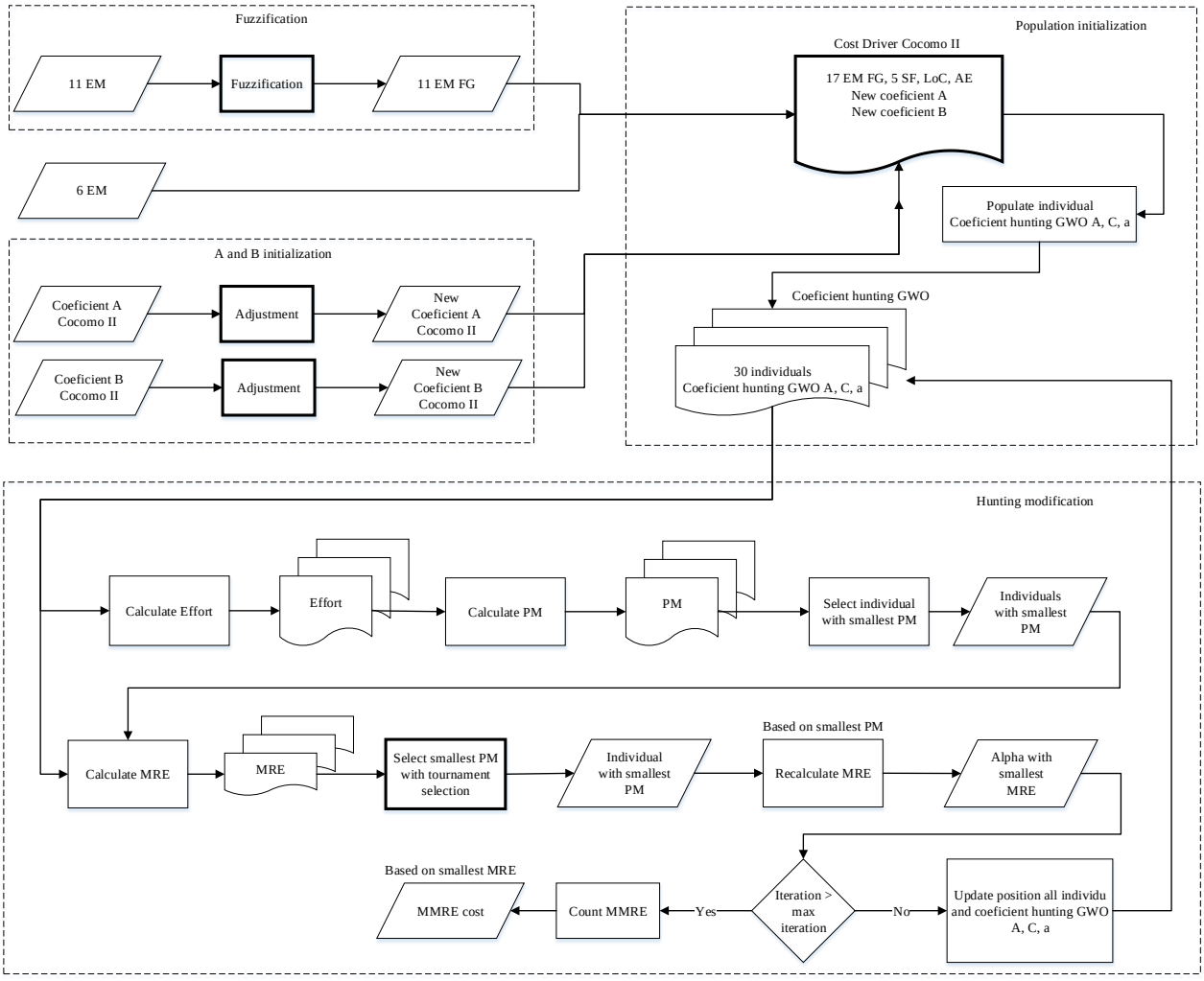

Figure. 2 Proposed method for improving COCOMO II performance

COCOMO II coefficient A and B model. The hunting mechanism involves selecting the alpha individual with the lowest fitness via tournament selection and selecting the minimum value among the best alpha individuals.

3.1 Fuzzification

The COCOMO II Cost Driver has 25 attributes, which are categorized into 17 effort multipliers, five scale factors, and the attributes of lines of code (LoC) and actual effort (AE). To enhance the precision of COCOMO II, a novel approach was employed, in which 11 quantitative effort multipliers were subjected to fuzzy logic modeling. This process aims to generate more optimal values than the previous 11 predefined quantitative effort multipliers. The modeling process involves adding the 11 quantitative effort multipliers into the fuzzy system. The Gaussian Membership Function is commonly used in fuzzy logic for modeling purposes. An example of a quantitative effort multiplier is LTEX, which represents the level of experience of the software development team in applying the tool for development over one year. The range of the newly obtained values of the 11 quantitative effort multipliers is shown in Table 1. Previously, LTEX had a range between 0.84% to 1.2%, which was then changed to possess a range between 0.85% to 1.16%. These changes to the range of effort multiplier values influence COCOMO II's accuracy. The fuzzy model is used to modify the effort multiplier values and uses the cost driver description as fuzzy logic input. For example, the PCAP effort multiplier has an interval from very low to very high, in which the input value for very low is 15%, for low is 35%, and so on. The difference between each level in PCAP is the percentage of the programmer's ability to work in a team. This research utilizes a Gaussian Membership Function (GMF), represented by the following in Eq. (1). Where C_i is the center, and σ_i is the curve's width. Which contains the parameters of the center of the value (C_i) and the curve's width (σ_i).

Table 1. Result of fuzzy implementation on effort multiplier

Cost Driver	Effort Multiplier	Range
Product	DATA	0.90-1.24
Platform	TIME	1.00-1.59
	STOR	1.00-1.45
	PVOL	0.87-1.23
Personnel	ACAP	0.74-1.41
	PCAP	0.76-1.34
	PCON	0.81-1.26
	LTEX	0.85-1.16
	APEX	0.81-1.11
	PLEX	0.85-1.13
Project	SCED	1.00-0.31

$$\mu_{Ai}(\mathrm{X}) = \mathrm{Gaussian}(\mathrm{x}, c_i, \sigma_i) = e^{-(x-ci)\wedge 2/2\,\sigma_i 2} \quad (1)$$

This equation is different from the GMF carried out by Maleki et al.[22] in the curve width calculation section.

3.2 A and B initialization

Implementing the Gaussian Membership Function to COCOMO II coefficients A and B randomly aims to improve the effort calculation of the COCOMO II model by minimizing the error value and identifying the most optimal solution.

3.3 Population initialization

The COCOMO II cost driver receives initial values during the population initialization stage. These cost drivers include the 11 quantitative effort multipliers subjected to fuzzy logic modeling, the other six effort multipliers, the five scale factors, LoC, AE, and the new COCOMO II coefficients A and B values. The determination of these values occurs without depending on subjective factors. Next, the process of forming populations consisting of 30 individuals is carried out. The values of the coefficient vectors of the GWO method, namely A, C, and a, are initialized randomly. Each individual consists of the COCOMO II cost drivers and hunting coefficients.

3.4 Hunting modified

Effort (E) for each individual within a population of 30 individuals is calculated. Then, the Person-Months (PM), which represents the number of hours worked within a month for each individual, is calculated. Next, the individuals with the lowest PM values are selected. Subsequently, to obtain the error value, the MRE value is computed using the PM values of the selected individuals. Tournament selection determines a single candidate with the lowest PM value to become the alpha. To determine the alpha, the individual that obtains the lowest MRE value is selected. The location of each individual and the hunting coefficients are updated if the process still needs to be completed after the specified maximum number of iterations, namely 500 iterations. Until the specified condition is fulfilled, an alpha individual with the minimum MRE is obtained, and computing E is repeated. Once the iteration procedure is

complete, the MMRE value is calculated in order to find the Alpha individual with the lowest MMRE. The algorithm assigns the alpha wolf based on the location of the individual closest to the prey. The other wolves within the pack are required to obey the alpha wolf's lead. Initially, each individual's initialization process is marked as Xi. Next, it is necessary to initialize the coefficient hunting vectors a, A, and C to surround the prey. The algorithm is based on the assumption that the alpha wolf has greater knowledge compared to the other wolves on the location of the prey. Thus, the remaining wolves within the pack are required to stick to a sequential pattern of following the alpha wolf. Each individual wolf adjusts its location by considering the positions of all the other wolves that were picked beforehand. To clarify, the n-th wolf updates its position by considering the position of the n-1-th wolf. Therefore, the speed of expansion and selecting an optimal location for the first wolf are crucial. The next wolf within the pack independently adjusts its location based on the location of the alpha wolf, and this process continues iteratively, as shown in Fig. 3. Tournament selection is used to identify the alpha wolf in each iteration of the process of optimizing COCOMO II coefficient A and B parameters. Initially, several individuals are randomly picked. The MRE value, used as the picked individuals' fitness function, is calculated. Xα is determined by utilizing the COCOMO II coefficient A and B parameter values of the picked individuals, where Xα in Eq. (3) represents the search agent with the optimal MRE value. The proposed method is shown in Fig. 3. The method contains many parameters.

$$\vec{a}=2\left(1-\frac{t^j}{Tj}\right) \tag{2}$$

The encircling behavior is modeled by Eq. (3) and Eq. (4). Eq. (3) is the coefficient vector, namely C, is used to guide the wolves in hunting for the prey and aid the algorithm in finding the optimal solution and avoid being trapped in a local optimal and also coefficient vector A and D in Eq. (4). The impact of the coefficient vector $\vec{a}$ in Eq. (3) is observed in the amount of motion, which guides the algorithm in its search for a solution. The variable j in Eq. (2) is established to improve the number of iterations allocated to the exploration process optima and coefficient vectors A and D in Eq. (4).

$$\vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}\right| \tag{3}$$

$$X_a = X_a - A_a \,.\, D_a \tag{4}$$

$$X_n(t+1) = \frac{1}{n-1}\sum_{i=1}^{n-1} Xi(t); n = 2, \ldots m \tag{5}$$

In Eq. (5), n represents the currently selected wolf, m represents the total number of wolves in the pack, t represents the iteration, and i parameter starts from the first wolf and progresses until before the last wolf has been selected and updated. A value is reduced from 2 to 0 by $\vec{a}$. That demonstrates the exploration

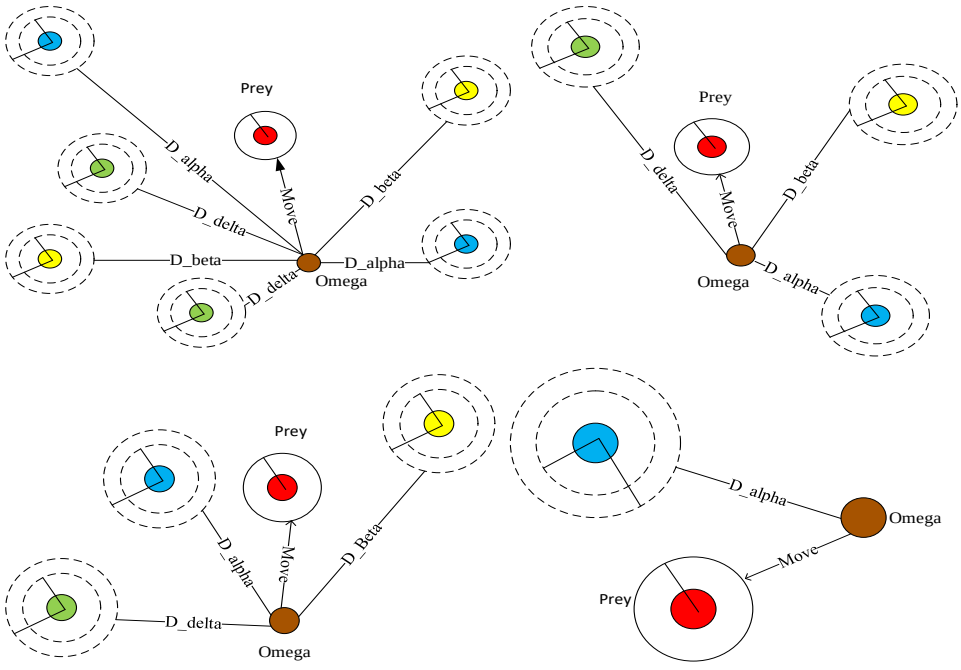

Figure. 3 The process of updating each wolf's position in capturing the prey in the proposed method

and exploitation aspects of the hunting mechanisms. C parameter can generate random values in each repetition [23]. Those equations are different from those carried out by Mirjalili et al.[24] in hunting prey. Their research conducted alpha, beta, and delta individuals, who were assessed using a hunting procedure to identify the most proficient individuals. It is common knowledge that alpha represents the prime candidate solution within the group of individuals. Consequently, the scope of our study is primarily directed towards the pursuit of alpha individuals through the hunting process, aiming to determine the most superior alpha individual among various other alphas.

4. Experimental result

In the first experiment, an extension of COCOMO II, namely the COCOMO II-FG model, is tested, and its performance is compared to the original COCOMO II model. In the COCOMO II-FG model, the effort multiplier values are modified by means of the Gaussian Membership Function. The COCOMO II-GWO model is tested in the second experiment, and its performance is compared to the original COCOMO II model. In the COCOMO II-GWO model, the GWO method with the proposed hunting mechanism is used to optimize COCOMO II coefficients A and B. In the third experiment, the proposed model is implemented. Our research proposes changes to the NASA 60 cost driver to suit the COCOMO II, Turkish, and NASA 93 datasets. In the NASA 60 dataset, there are only four levels of Effort Multiplier, which should be six levels according to COCOMO II: very low (VL) to extra high (EH). Apart from that, the LoC values in the NASA 60 dataset are smaller compared to the other two datasets. For comparison, the LoC values in the NASA 60 dataset are within the range of 5.5%-177.9%. Meanwhile, the LoC values for the Turkish and NASA 93 23 datasets are between 1611-114280 and 900-352000, respectively. LoC is used to calculate PM. Consequently, LoC affects the resulting MRE value. In the proposed model, the Gaussian Membership Function is used to optimize the values of quantitative effort multipliers, and the GWO method with the proposed hunting mechanism is used to optimize the COCOMO II coefficient A and B parameters. The performance of the proposed model is compared to COCOMO II. For performance evaluation, the MRE and MMRE metrics are used [15]. MRE is widely used for evaluating effort estimating models and is calculated using Eq. (6). MRE is computed for each project within the dataset. Subsequently, MMRE is calculated using Eq. (7), the average of MRE over n number of projects.

Table 2. Effect of FG and HGWO on increasing MMRE

Method	MMRE (%)		
	Turkish	**NASA 60**	**NASA 93**
COCOMO II	733.14	0.99	1243.23
COCOMO II-FG	51.08	0.99	716.85
COCOMO II-GWO	1.49	4.78	3.15
COCOMO II-FG-HGWO	**0.82**	**0.01**	**0.26**

$$MRE = \frac{Predicted\ value - Actual\ Value}{Actual\ Value} \quad (6)$$

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} MRE \quad (7)$$

Based on the experimental, Table 2 shows the results of comparison accuracy in MMRE. The first trial, COCOMO II with the Fuzzy Gaussian (COCOMO II-FG) model, was compared with COCOMO II using Turkish, NASA 60, and NASA 93 datasets. The MMRE value showed that Turkish decreased by 682.06%, NASA 93 decreased by 526.38%, and NASA 60 showed no changes. This means there is still a less significant decrease in estimated values (MMRE) between COCOMO II and COCOMO II-FG from the Turkish and NASA 93 datasets, except for MMRE NASA 60, where there is no change in value. The second trial, the COCOMO II-GWO model, was compared with COCOMO II using the Turkish datasets NASA 60 and NASA 93. The MMRE value showed that Turkish decreased by 731.65%, NASA 60 increased by 3,79%, and NASA 93 decreased by 1240.08%. This means there is a significant decrease in the estimated value (MMRE) between COCOMO II and COCOMO II-GWO. The proposed model was assessed using the Turkish, NASA 60, and NASA 93 datasets. The MMRE values obtained by the proposed model on the Turkish, NASA 60, and NASA 93 datasets were lower by 732.32%, 0.98%, and 1242.97%, respectively, compared to those obtained by COCOMO II. The proposed model on the Turkish, NASA 60, and NASA 93 datasets was lower by 0.67%, 4.77%, and 2.89%, respectively, compared to those obtained by COCOMO II-GWO. The MMRE value obtained by the proposed model on the NASA

Table 3. Comparison results using Turkish and NASA datasets

Author	Method	MMRE (%)		
		Turkish	NASA 60	NASA 93
Sheikh et al.	COCOMO II-GWO	-	-	3.45
Putri et al.	COCOMO II-GWO	-	-	3.15
Putri et al.	COCOMO II-enhance gamma GWO	-	-	51.09
Proposed	COCOMO II-FG-HGWO	**0.82**	**0.01**	**0.26**

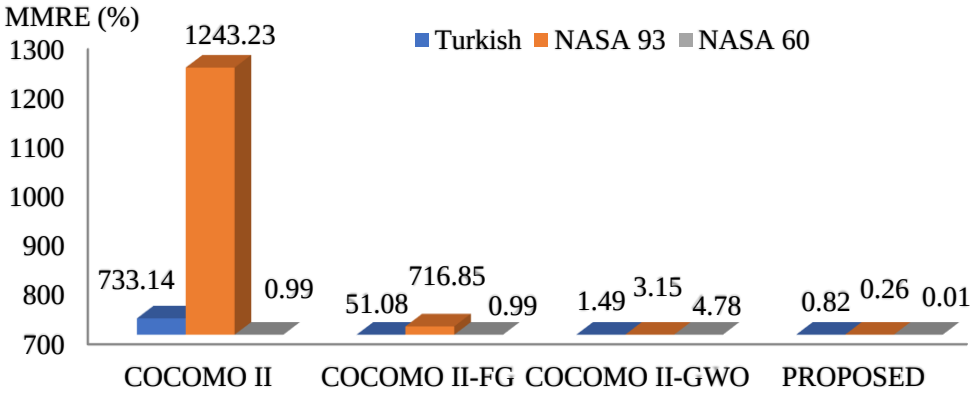

Figure. 4 MMRE value of effort estimation applying the proposed method and other methods

60 dataset was significantly lower than all the other models, which signifies that the use of Fuzzy Gaussian and the modified coefficient hunting vectors a, A, and C of the GWO method significantly enhanced the performance of COCOMO II. Table 2 shows that the proposed model obtained significantly lower MMRE values on all the datasets compared to COCOMO II and COCOMO II-GWO, which are marked in bold.

Two studies demonstrate similarities to our investigation but also highlight differences in the COCOMO model, GWO method, and dataset. Table 3 compares the findings of our study and those of Khan et al. [7], who utilized COCOMO I-GWO without any modifications in addition to Deep Neural Network and Strawberry optimization on the NASA 93 dataset. Their analysis produced an MMRE of 3.45%. On the other hand, research conducted by Sheikh et al. [25] utilized COCOMO I and COCOMO II-GWO without modifications on the NASA 18 dataset, resulting in an MMRE of 0.13%. Nevertheless, their study omitted the use of Fuzzy Gaussian and refrained from altering any mechanisms within the GWO method. In contrast, our study incorporates FG to convert 11 COCOMO II quantitative EMs. It also modifies the GWO hunting mechanism by adjusting coefficient hunting vectors a, A, and C of GWO and employing tournament selection.

5. Analysis

Based on Table 2, our proposed method obtained a significantly lower MMRE value than other studies, especially on the NASA 60 dataset, namely 0.01%. This MMRE value is much smaller than that obtained by COCOMO II on the NASA 60 dataset, namely 0.98%, and much smaller than obtained by COCOMO II-GWO on the NASA 60 dataset, namely 4.78%. Furthermore, the proposed model obtained a lower MMRE value on the NASA 60 dataset than the other datasets. This is because the LoC in the NASA 60 dataset tends to be smaller compared to that in the other two datasets. The experimental results indicate that the method proposed in this study achieved the best performance indicated by lower MMRE values on all the datasets, as shown in Fig. 4.

In previous studies, modifications to the GWO algorithm are advantageous. Improvements to the position-updating mechanism of GWO enhanced its performance [26]. Furthermore, controlling the coefficient of GWO to balance the exploration and exploitation process better can also be done to increase the performance of GWO [27]. Modifying the encircling behavior and position-updating equation of GWO has been demonstrated to yield effective results and identify the optimal solution [28]. Moreover, the use of fuzzy sets has been shown to

enhance the performance of COCOMO II [17]. Fuzzy sets are used instead of crisp sets to quantify uncertainty [29]. It is able to adequately manage the uncertainty and imprecision associated with business process complexity metrics.

6. Conclusion

Our study aims to improve the accuracy of COCOMO II in estimating software project costs by proposing a COCOMO II-FG-HGWO method. The method applies Gaussian Membership Functions to change 11 COCOMO II quantitative Effort Multiplier (EM) values. And also change COCOMO II coefficient A and B values using the Gaussian Membership Function. GWO method is used to optimize the coefficient hunting values a, A, and C in GWO by modifying the hunting mechanism using tournament selection. The hunting mechanism is modified using the GWO method in the proposed model, and the alpha wolf is selected using the Tournament Selection method. As a result, optimal values of A and B are obtained. The performance of the proposed model is compared to other effort estimation models using the Turkish, NASA 93, and NASA 60 datasets. Using the Gaussian Membership Function to adjust the effort multiplier values and the modified GWO method to optimize COCOMO II coefficient A and B parameters, significantly lower MMRE values were obtained**.**

Based on the experimental, our research proposal is compared with COCOMO II using the Turkish, NASA 60, and NASA 93 datasets. The experimental results show that the proposed model obtained MMRE values on the Turkish, NASA 93, and NASA 60 datasets that were lower by 723.04%, 1241.26%, and 0.98%, respectively, compared to those obtained by COCOMO II. This indicates that the proposed model produces more accurate effort estimations compared to COCOMO II and the other estimation models. This is further signified by the lowest MMRE value obtained by the proposed model on the NASA 60 dataset compared to COCOMO II and the other estimation models. Even though adjustments to cost drivers were required on the NASA 60 dataset, a significantly low MMRE was obtained using the proposed method. The COCOMO II-FG-HGWO model obtained a lower MMRE value on the NASA 60 dataset than the other datasets. This is because the LoC in the NASA 60 dataset tends to be smaller compared to that in the other two datasets. For comparison, the smallest LoC value for the NASA 60 dataset is 5.5 to 177.9, the smallest LoC value for the Turkish dataset is 1611 to 114280, and the smallest LoC value for the NASA 93 dataset is 900 to 352000. This LoC is used as one of the values for calculating PM and will affect whether the MRE and MMRE values are large or small.

This indicates that estimated effort is closer to actual effort, and the risk of error in calculating project costs becomes smaller and, in turn, improves the quality of software projects. For future research, it is possible to enhance the performance of COCOMO II by modifying the dataset's attributes and implementing modifications to the attributes of COCOMO II. Additionally, further modifications can be made to the optimization method mechanism to achieve more enhanced estimations.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Rahmi Rizkiana Putri and Daniel Siahaan; methodology, All Authors; software, Rahmi Rizkiana Putri; validation, Daniel Siahaan and Chastine Fatichah; writing—original draft preparation, All authors; writing—review and supervision, Daniel Siahaan dan Chastine Fatichah.

Acknowledgments

The authors thank the Ministry of Education, Culture, Research, and Technology through the PPS-PDD program. With grant number 009/E5/PG.02.00.PL/2023.

References

[1] N. Rankovic, D. Rankovic, M. Ivanovic, and L. Lazic, “A New Approach to Software Effort Estimation Using Different Artificial Neural Network Architectures and Taguchi Orthogonal Arrays”, *IEEE Access*, Vol. 9, pp. 26926-26936, 2021, doi: 10.1109/ACCESS.2021.3057807.

[2] P. Rai, D. K. Verma, and S. Kumar, “A hybrid model for prediction of software effort based on team size”, *IET Softw.*, Vol. 15, No. 6, pp. 365-375, 2021, doi: 10.1049/sfw2.12048.

[3] M. N. Mahdi, *et al.*, “Software Project Management Using Machine Learning Technique—A Review”, *Appl. Sci.*, p. 39, 2021, doi: 10.3390/app11115183.

[4] M. F. Diego, E. R. Méndez, F. G. L. de Guevara, S. Abrahão, and E. Insfran, “An update on effort estimation in agile software development: A systematic literature review”, *IEEE Access*, Vol. 8, pp. 166768-166800, 2020, doi: 10.1109/ACCESS.2020.3021664.

[5] I. M. S. Raharja, and D. O. Siahaan,

"Classification of non-functional requirements using fuzzy similarity KNN Based on ISO / IEC 25010", In: *Proc. of 2019 Int. Conf. Inf. Commun. Technol. Syst. ICTS 2019*, pp. 264-269, 2019, doi: 10.1109/ICTS.2019.8850944.

[6] A. Nizam, "Software Project Failure Process Definition", *IEEE Access*, Vol. 10, pp. 34428-34441, 2022, doi: 10.1109/ACCESS.2022.3162878

[7] M. S. Khan, F. Jabeen, S. Ghouzali, Z. Rehman, S. Naz, and W. Abdul, "Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation", *IEEE Access*, Vol. 9, pp. 60309-60327, 2021, doi: 10.1109/ACCESS.2021.3072380.

[8] M. A. Shah, D. N. A. Jawawi, M. A. Isa, M. Younas, A. Abdelmaboud, and F. Sholichin, "Ensembling Artificial Bee Colony with Analogy-Based Estimation to Improve Software Development Effort Prediction", *IEEE Access*, Vol. 8, pp. 58402-58415, 2020, doi: 10.1109/ACCESS.2020.2980236.

[9] M. Baiquni, R. Sarno, Sarwosri, and Sholiq, "Improving the accuracy of COCOMO II using fuzzy logic and local calibration method", In: *Proc. of 2017 3rd Int. Conf. Sci. Inf. Technol. Theory Appl. IT Educ. Ind. Soc. Big Data Era, ICSITech 2017*, Vol. 2018-Janua, pp. 284-289, 2017, doi: 10.1109/ICSITech.2017.8257126.

[10] D. T. Hidayat, C. Fatichah, and R. V. Ginardi, "Pattern reduction enhanced ant colony optimization clustering algorithm", In: *Proc. of 2016 Int. Semin. Appl. Technol. Inf. Commun. ISEMANTIC 2016*, pp. 317-322, 2017, doi: 10.1109/ISEMANTIC.2016.7873858.

[11] Y. A. Effendi, R. Sarno, J. J. Prasetyo, Y. A. Effendi, R. Sarno, and J. J. Prasetyo, "Implementation of Bat Algorithm for COCOMO II Optimization", In: *Proc. of 2018 Int. Semin. Appl. Technol. Inf. Commun. Creat. Technol. Hum. Life, iSemantic 2018*, pp. 441-446, 2018, doi: 10.1109/ISEMANTIC.2018.8549699.

[12] D. Nandal, O. P. Sangwan, V. K. Attri, and S. I. Khaleel, "Software cost estimation by optimizing COCOMO model using hybrid BATGSA algorithm", *Int. J. Intell. Eng. Syst.*, Vol. 11, No. 4, pp. 250-263, 2018, doi: 10.22266/ijies2018.0831.25.

[13] K. Langsari, R. Sarno, and Sholiq, "Optimizing time and effort parameters of COCOMO II using fuzzy Multi-objective Particle Swarm Optimization", *Telkomnika (Telecommunication Comput. Electron. Control.*, Vol. 16, No. 5, pp. 2199-2207, 2018, doi: 10.12928/TELKOMNIKA.v16i5.9698.

[14] W. D. Sunindyo, and C. Rudiyanto, "Improvement of COCOMO II Model to Increase the Accuracy of Effort Estimation", In: *Proc. of Int. Conf. Electr. Eng. Informatics*, Vol. 2019-July, pp. 140-145, 2019, doi: 10.1109/ICEEI47359.2019.8988909.

[15] A. A. Fadhil, R. G. H. Alsarraj, and A. M. Altaie, "Software Cost Estimation Based on Dolphin Algorithm", *IEEE Access*, Vol. 8, pp. 75279-75287, 2020, doi: 10.1109/ACCESS.2020.2988867.

[16] I. C. Suherman, "Implementation of Random Forest Regression for COCOMO II Effort Estimation", *IEEE*, pp. 476-481, 2020, doi: 10.1109/iSemantic50169.2020.9234269.

[17] C. A. U. Hassan *et al.*, "Optimizing Deep Learning Model for Software Cost Estimation Using Hybrid Meta-Heuristic Algorithmic Approach", *Comput. Intell. Neurosci.*, Vol. 2022, 2022, doi: 10.1155/2022/3145956.

[18] S. S. Ali, J. Ren, K. Zhang, J. Wu, and C. Liu, "Heterogeneous Ensemble Model to Optimize Software Effort Estimation Accuracy", *IEEE Access*, No. February, pp. 1-1, 2023, doi: 10.1109/access.2023.3256533.

[19] N. M. Alsheikh, and N. M. Munassar, "Improving Software Effort Estimation models using Grey Wolf Optimization Algorithm", *IEEE Access*, Vol. 11, No. September, pp. 143549-143579, 2023, doi: 10.1109/ACCESS.2023.3340140.

[20] R. R. Putri, D. O. Siahaan, and C. Fatichah, "Improve the Accuracy of Software Project Effort and Cost Estimates in COCOMO II Using GWO", In: *Proc. of 2021 5th International Conf on Informatics and Computational Sciences (ICICoS)*, pp. 128-133, 2021, doi: 10.1109/ICICoS53627.2021.9651845.

[21] R. R. Putri, D. Siahaan, and C. Fatichah, "Improving the Accuracy of COCOMO II Using Extended Gamma GWO", In: *Proc. of 2022 Int. Semin. Intell. Technol. Its Appl. Adv. Innov. Electr. Syst. Humanit. ISITIA 2022*, pp. 145-150, 2022, doi: 10.1109/ISITIA56226.2022.9855265.

[22] I. Maleki, L. Ebrahimi, S. Jodati, and I. Ramesh, "Analysis of Software Cost Estimation Using Fuzzy Logic", *Int. J. Found. Comput. Sci. Technol.*, Vol. 4, No. 3, pp. 27-41, 2014, doi: 10.5121/ijfcst.2014.4303.

[23] A. Seyyedabbasi and F. Kiani, "I-GWO and Ex-GWO: improved algorithms of the Grey Wolf Optimizer to solve global optimization problems", *Engineering with Computers*, Vol. 37, No. 1. pp. 509-532, 2021, doi:

10.1007/s00366-019-00837-7.
[24] S. M. Mirjalili, S. M. Mirjalili, A. Lewis, M. Seyedali, M. Seyed Mohammad, and L. Andrew, "Grey Wolf Optimizer", *Adv. Eng. Softw.*, Vol. 69, pp. 46-61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.
[25] N. Al-sheikh, and N. Munassar, "Improving Software Effort Estimation Models Using Grey Wolf Optimization Algorithm", *IEEE Access*, Vol. 11, pp. 143549-143579, 2023, doi: 10.1109/ACCESS.2023.3340140.
[26] F. Yan, X. Xu, and J. Xu, "Grey Wolf Optimizer with a Novel Weighted Distance for Global Optimization", *IEEE Access*, Vol. 8, pp. 120173-120197, 2020, doi: 10.1109/ACCESS.2020.3005182.
[27] H. Almazini, and K. R. K. Mahamud, "Grey Wolf Optimization Parameter Control for Feature Selection in Anomaly Detection", *Int. J. Intell. Eng. Syst.*, Vol. 14, No. 2, pp. 474-483, 2021, doi: 10.22266/ijies2021.0430.43.
[28] N. Singh, and S. B. Singh, "A Modified Mean Gray Wolf Optimization Approach for Benchmark and Biomedical Problems", *Evol. Bioinforma.*, Vol. 13, 2017, doi: 10.1177/1176934317729413.
[29] N. Y. Setiawan, and R. Sarno, "Multi-Criteria Decision Making for Selecting Semantic Web Service Considering Variability and Complexity Trade-Off," *J. Theor. Appl. Inf. Technol.*, Vol. 86, No. 2, pp. 316-326, 2016, [Online]. Available: http://www.jatit.org/volumes/Vol86No2/16Vol86No2.pdf.