

PARAMETER ESTIMATION OF COCOMO
MODEL USING THE JAYA ALGORITHM FOR
SOFTWARE COST ESTIMATION

TAN JIE CHEE

BACHELOR OF COMPUTER SCIENCE

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : TAN JIE CHEE
Date of Birth : 02/04/1994
Title : PARAMETER ESTIMATION OF COCOMO MODEL USING
THE JAYA ALGORITHM FOR SOFTWARE COST
ESTIMATION
Academic Session : SEMESTER I 2018/2019

I declare that this thesis is classified as:

- CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)*
 RESTRICTED (Contains restricted information as specified by the organization where research was done)*
 OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:



(Student's Signature)



(Supervisor's Signature)

940402-07-5380
New IC/Passport Number
Date: 4/1/2019

Prof.Dr.Kamal Z. Zamli
Name of Supervisor
Date: 4/1/2019

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

THESIS DECLARATION LETTER (OPTIONAL)

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name	TAN JIE CHEE
Thesis Title	PARAMETER ESTIMATION OF COCOMO MODEL USING THE JAYA ALGORITHM FOR SOFTWARE COST ESTIMATION

Reasons	(i)
	(ii)
	(iii)

Thank you.

Yours faithfully,



(Supervisor's Signature)

Date: 4/1/2019

Stamp: PROF. DR. KAMAL Z. ZAMLU
Professor
Faculty of Computer Systems & Software Engineering
Universiti Malaysia Pahang
Lebuhraya Tun Razak, 26300 Gambang, Kuantan, Pahang
Tel: 09-5492544 Fax: 09-549 2144

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.



SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Bachelor of Computer Science in Software Engineering.

A handwritten signature in black ink, appearing to read 'Kamal Z Zamli', is positioned above a horizontal line.

(Supervisor's Signature)

Full Name : PROF. DR. KAMAL Z ZAMLI
Professor
Faculty of Computer Systems & Software Engineering
Universiti Malaysia Pahang
Position : Lebuhraya Tun Razak, 26300 Gambang, Kuantan, Pahang
Tel : 09-5492544 Fax : 09-549 2144
Date : 4/1/2019



STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

A handwritten signature in black ink, appearing to read 'Tan Jie Chee', is written above a horizontal line.

(Student's Signature)

Full Name : TAN JIE CHEE

ID Number : CB15033

Date : 4 JANUARY 2019

PARAMETER ESTIMATION OF COCOMO MODEL USING THE JAYA
ALGORITHM FOR SOFTWARE COST ESTIMATION

TAN JIE CHEE

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Computer Sciences (Software Engineering) with Honours

Faculty of Computer Systems & Software Engineering
UNIVERSITI MALAYSIA PAHANG

JANUARY 2019

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my gratitude to the following professionals of whom contributed to the success of this research.

Firstly, I would like to express my appreciation to my Final Year Project supervisor, Prof. Dr. Kamal Zuhairi Bin Zamli, who was a great role model through the course of this project. His guidance and encouragement allowed to me to carry out this research work to my fullest potential.

An incomparable appreciation to Miss Leu Xin Yi, Miss Tee Shih Min and Miss Wong Wei San, my friend whom I discussed and shared findings with. Without their assistance in conducting my experiments, the outcomes of this research would not be as successful.

Finally, I would like to thank my family member for their support and encourage throughout this research.

ABSTRAK

Sejajar dengan kemajuan teknologi komputer, perisian semakin banyak digunakan di samping perkakasan untuk membantu manusia melakukan tugas harian. Anggaran kos pembangunan perisian boleh menjadi isu utama (contoh: menentusahkan kos sebenar pembangunan perisian). Sehingga kini, banyak teknik telah dicadangkan untuk membantu jurutera perisian menentukan kos perisian sebenar. Tesis ini mencadangkan algoritma Jaya berdasarkan anggaran perisian berdasarkan model COCOMO I. Model Kos Konstruktif (COCOMO) adalah model anggaran kos prosedur perisian yang dibangunkan oleh Barry W. Boehm. Dalam kes ini, anggaran nilai parameter model COCOMO ditentukan untuk kos dan anggaran masa model COCOMO. Akibatnya, COCOMO Jaya Algorithm (COCOMO JA) telah dibangunkan. Projek dataset NASA 63, dataset Turki, dan dataset Kemerer telah digunakan. Keputusan eksperimen menunjukkan perbandingan antara COCOMO standard dan COCOMO JA. COCOMO standard menunjukkan lebih banyak kesilapan berbanding dengan COCOMO JA. Sebagai kesimpulan, JA sesuai untuk menilai usaha dan masa anggaran untuk model COCOMO.

ABSTRACT

In line with the advancement of computer technology, software is increasingly being used in place of hardware to help human do the daily chores. Estimate of the software development costs can be a major issue (i.e. in terms of determining the actual development costs). To date, many techniques have been proposed to help the software engineer determine the actual software cost. This thesis proposed the Jaya algorithm based on estimation of the software based on the COCOMO I model. The Constructive Cost Model (COCOMO) is a procedural software cost estimation model developed by Barry W. Boehm. In this case, the estimation of value of the COCOMO model parameters are determined for the cost and time estimation of the COCOMO model. As a result, COCOMO Jaya Algorithm (COCOMO JA) has been developed. The dataset NASA 63 project, Turkish dataset, and Kemerer dataset have been used. The experiment result shows the comparison between the standard COCOMO and COCOMO JA. The COCOMO JA demonstrated more minimize error as compare to the standard COCOMO. In conclusion, the JA suitable to evaluate the estimation effort and time for the COCOMO model.

TABLE OF CONTENT

DECLARATION	
TITLE PAGE	
ACKNOWLEDGEMENTS	ii
ABSTRAK	iii
ABSTRACT	iv
TABLE OF CONTENT	v
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF ABBREVIATIONS	xi
CHAPTER 1 INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	3
1.3 Aim and Objective	3
1.4 Scope	4
1.5 Significance	4
1.6 Thesis Organization	5
CHAPTER 2 LITERATURE REVIEW	6
2.1 Introduction	6
2.2 Software Cost Estimation Related Work	6
2.2.1 Function Point	7
2.2.2 Software Life-cycle Model (SLIM)	7

2.2.3	COCOMO I	8
2.2.4	COCOMO II	9
2.3	Existing COCOMO I Model Based On Optimization Algorithms	10
2.3.1	Cuckoo Search Algorithm	10
2.3.2	Genetic Algorithm	12
2.3.3	Firefly Algorithm	14
2.3.4	Jaya Algorithm	16
2.4	Analysis of The Related Work	18
2.5	Summary	21
CHAPTER 3 METHODOLOGY		22
3.1	Introduction	22
3.2	Research Activity	22
3.2.1	Phase 1: Problem defining	23
3.2.2	Phase 2: Concepts and Literature Review	23
3.2.3	Phase 3: User Design	24
3.2.4	Phase 4: Implementation	24
3.2.5	Phase 5: Experiment Result	24
3.2.6	Phase 6: Documentation	24
3.3	Methodology	24
3.3.1	Training Phase using Jaya Algorithm	25
3.3.2	Testing Phase using Jaya Algorithm	25
3.3.3	Testing Phase Using Standard COCOMO Model	26
3.4	Hardware and Software Requirement	26
3.4.1	Hardware Specification	26
3.4.2	Software Specification	26

3.5	Gantt Chart	27
3.6	Implementation	27
3.6.1	Jaya Algorithm	27
3.6.2	Proposed Estimation Effort for the Jaya Algorithm	29
3.6.3	Proposed Time Estimation of COCOMO Model using the Jaya Algorithm	32
3.7	Data Collection and Analysis	34
3.7.1	NASA63 Software Project Dataset	34
3.7.2	Turkish Software Industry Dataset	35
3.7.3	Kemerer Dataset	36
3.8	Summary	37
CHAPTER 4 RESULTS AND DISCUSSION		38
4.1	Introduction	38
4.2	Experimental Evaluations	38
4.3	Experimental Results	39
4.3.1	Generation Values of COCOMO Model Parameters	40
4.3.2	Effort Estimation	41
4.3.3	Time Estimation	45
4.4	Discussion	49
4.5	Summary	51
CHAPTER 5 CONCLUSION		52
5.1	Introduction	52
5.2	Research Constraint	53
5.3	Contribution	53

5.4	Future Work	54
	REFERENCES	55
	APPENDIX A GANTT CHART	58
	APPENDIX B NASA63 SOFTWARE PROJECT DATASETS	60

LIST OF TABLES

Table 2.1	Basic COCOMO Models	9
Table 2.2	Comparison the software cost estimation techniques	18
Table 2.3	COCOMO I Model Comparison between CS, GA, FA and JA	20
Table 3.1	List of Hardware Specification	26
Table 3.2	List of Software Specification	27
Table 3.3	NASA63 Dataset	35
Table 3.4	Turkish Dataset	36
Table 3.5	Kemerer Dataset	37
Table 4.1	Result of the Effort Estimation of the NASA63 Dataset	41
Table 4.2	Result of the Effort Estimation of the Turkish Dataset	43
Table 4.3	Result of the Effort Estimation of Kemerer Dataset	44
Table 4.4	Result of the Time Estimation of NASA63 Dataset	46
Table 4.5	Result of the Time Estimation of Turkish Dataset	47
Table 4.6	Result of the Time Estimation of Kemerer Dataset	48
Table 4.7	Comparison the Mean Magnitude of Relative Error (MMRE)	50

LIST OF FIGURES

Figure 2.1	Cuckoo Search Algorithm pseudo-code	11
Figure 2.2	Genetic Algorithm pseudo-code	13
Figure 2.3	Firefly Algorithm pseudo-code	15
Figure 2.4	Pseudocode of the Jaya Algorithm	17
Figure 3.1	Research Activity Phases	23
Figure 3.2	The overview of COCOMO model using the Jaya Algorithm	25
Figure 3.3	Pseudocode of the Jaya Algorithm	29
Figure 3.4	Pseudocode for Proposed of the Estimation Effort Jaya Algorithm	31
Figure 3.5	Pseudocode for Proposed of the Estimation Time Jaya Algorithm	32
Figure 4.1	Result Generating Parameters a, b, c, and d Value of COCOMO Model	40
Figure 4.2	Comparison of the Effort Estimation of NASA63 Datasets	42
Figure 4.3	Comparison of the Effort Estimation of Turkish Dataset	43
Figure 4.4	Comparison of the Effort Estimation of Kemerer Dataset	44
Figure 4.5	Comparison of the Time Estimation of NASA63 Dataset	46
Figure 4.6	Comparison of the Time Estimation of Turkish Dataset	48
Figure 4.7	Comparison of the Time Estimation of Kemerer dataset	49

LIST OF ABBREVIATIONS

COCOMO	Constructive Cost Model
JA	Jaya Algorithm
SLIM	Software Life-cycle Model
LOC	Lines of Code
PF	Productivity Factor
MBI	Manpower Buildup Index
SLOC	Source Lines of Code
E	Effort
KLOC	Kilo Line of Code
T	Time
ML	Machine Learning
CSA	Cuckoo Search Algorithm
GA	Genetic Algorithm
FA	Firefly algorithm
MMRE	Mean Magnitude of Relative Error
COCOMO JA	COCOMO Jaya Algorithm

CHAPTER 1

INTRODUCTION

1.1 Introduction

Software cost estimation is the process of evaluating the workload of software system development. Exact estimate software can supply powerful help in software management decision making. For example, precise evaluation can assist organizations to improve analyze the project cost, and efficiently manage the software development process, thereby notably reduce the risk. Ever excessive pessimistically plan may missing the business chances, but overly optimistically plans may lead major losses (Jayakumar et al, 2012).

Because this causes, exact software cost estimation for developers and customers have become crucial. In addition, implementing software cost estimation will influence the project's progress, complicity, dependability, quality and many other characteristics. During the implementation process, the assessment process needs to be completed in the full-time process and must be performed at each stage of the software development life cycle. Its purpose is to keep the change in the development process (Bingamawa, 2016).

Software cost estimation is to forecast the cost, time, and staffing levels are needed to establish a software system during the early stages of a project. Due to the limited resources available at the time, it was hard to acquire the estimated of effort in the initial stage. Based on the cost estimate requirements, one of is promoting control over time and overall cost-effectiveness in the software development lifecycle. Software cost estimates are even considered one of the three most challenging challenges in software applications. During development, the cost and time estimate for completion of the project process crude initial verification and supervising is helpful. In addition, these estimates may be helpful productivity assessment stages of the project.

Since 1960, many research papers provide many models to help calculate the cost of the software project, but due to many reasons, give a right cost estimation remains a challenge. The causes are included: 1) Collection of measurement unsureness, 2) the evaluation methods used may have many disadvantages, and 3) have different characteristics based on cost drivers of development methods (Abu-srhan, Sleit, & Sharieh, 2017; Aljahdali & Sheta, 2010; F., A., & Aljahdali, 2013). The software cost estimation has several models based which are COCOMO I, COCOMO II, Function Point (FP), and Putnam's Software Life Cycle Model (SLIM). This research discuss in Chapter 2.

In this research, Jaya Algorithm (JA) is provided as the detailed study of an optimization algorithm(F., A., & Aljahdali, 2013). It is a parameter for minimization of the COCOMO model in order to a more realistic workload can be evaluated. This research used the NASA63 software project dataset, Turkish software industry dataset and Kemerer dataset to analyse the capability assessment of the suggested approach. This research focus on COCOMO I model of the software cost estimation.

The rest of this research is following. Chapter 2 which discusses the cost estimation related work. Chapter 3 shows methodology to the software cost estimation process. Chapter 4 will illustrate the results of proposed method. Lastly, Chapter 5 is summarization of research.

1.2 Problem Statement

Many software industries facing failure in project effort estimation and finally give rise to the consequence of an enable to arrange the exact software release day. The company will face the consequences of losing money, especially when the company meets deadlines and forced to give a low quality of the deliverables (Sheta, A., Rine, D., & Ayesh, A., 2008).

Many research papers have applied the techniques and found the basic shape of the graph denoted by COCOMO equations is borne out. Specifically, the COCOMO model parameters are optimized to ensure accurate estimation of software development effort and time.

The hybrid algorithm of cuckoo search algorithm and genetic algorithm (CSGA) is difficult to master, strong learning curve as it uses the unfamiliar metaphor. Also, the CSGA need tuning, however, the Jaya algorithm does not need tuning as the parameters are adaptive (Abu-srhan et al., 2017).

1.3 Aim and Objective

The goal of this research is using the Jaya algorithm to find out the minimum of parameter value for COCOMO model parameters, then to calculate the estimated effort and time. The objective of this research is as follows:

- i. To review the state-of-the-art on software cost estimation focus on COCOMO I Model.
- ii. To implement the Jaya Algorithm (JA) for improving the COCOMO I Model based on estimation effort and time.
- iii. To evaluate the proposed of the estimation effort and time using the Jaya algorithm (JA).

1.4 Scope

The scope of this research are as follows:

- i. The study focuses on parameter estimation of the COCOMO I model.
- ii. The NASA 63 software project dataset, Turkish dataset, and Kemerer dataset used in this research.
- iii. The estimating of the parameter NASA 63 software project dataset, Turkish dataset, and Kemerer dataset are using the Jaya algorithm.

1.5 Significance

The significance of this research are as follows:

- i. Enable accurate software effort estimation can reduce the risk
- ii. Use the Jaya algorithm to improve the accuracy of estimation effort and time.
- iii. Supply good evaluation ability contrapositive with other models.

1.6 Thesis Organization

This thesis consists of five chapters. Below is the summary of all the chapter in the thesis:

Chapter one is the introduction. This chapter discusses the introduction, problem statement, aim, and objective, scope, significance and thesis organization of the research.

Chapter two is the literature review. The thesis will discuss the research and literature review that is related to the research.

Chapter three is methodology. It will explain the methodology used in this research.

Chapter four is the implementation. The result of the research, which will be discussed based on the COCOMO I model.

Chapter five is the conclusion. It will make a conclusion of the research that had been done and summarize this research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

At the chapter 1, the discussion has done about this research's introduction where cover problem statement, objectives, scope, significance and thesis organization. Hence in this chapter will be discussing the previous research about that related to this thesis and literature review that existed.

2.2 Software Cost Estimation Related Work

The project manager wants to distinctly determine the cost evaluates for the software development in order to evaluate the progress of the project based on the anticipate budget, anticipate timetable, and potential for improved utilization of resources. The study finds the major cost drivers of software development were efforts to converted efforts into costs.

Some software cost estimation models are developed, and an advanced software cost estimation model is established. These models include the Function Points, SLIM model, COCOMO I, and COCOMO II (Kumar et al, 2012). This research discussed the COCOMO I model and used the meta-heuristic algorithm which is Jaya algorithm to estimate the effort and time in this session.

2.2.1 Function Point

The function point measurement method was developed by Allan. Albrecht of IBM and was first published in 1976. The suggested function point has no dimension. Functional Point is calculated based on analysing the needs of the project. These requirements help determine the number of functions to be developed and the complicity of every function. Therefore, there is no requirement to calculate the size of the Lines of Code (LOC), only the project functions need to be considered. Once the number of function points is determined, the average monthly function points and the monthly labour cost are estimated; the overall budget can be calculated. Albrecht initially suggested four categories such as the file, input, output, and inquiry, including a set of related weights and 10 General System Characteristics (GSC) (F., A., & Aljahdali, 2013).

2.2.2 Software Life-cycle Model (SLIM)

The Putnam's Software Life-cycle Model (SLIM) is an empirical software effort estimation model. The SLIM model developed by Putnam is ground on Norden / Rayleigh's human resource allocation and his findings in many projects completed in the analysis. (Suri & Ranjan, 2012). SLIM relies on Source Lines of Code (SLOC) estimation of the general size of the project and then changes it by using the Rayleigh Curve model in order to generate the effort estimation. The pattern of the curve can be affected by 2 main parameters: the beginning gradient of the curve and an element of productivity. The SLIM users have two options for selecting their values. The first way is users can input data to adjust the model of the done project through, they can reply a set of 22 questions, of which SLIM to be offered the recommendation of PF and MBI. The second way selected for this study, because there is no formerly gather standardization database, and that is sense more exactly returns the general user experience (Kemerer, 1987).

2.2.3 COCOMO I

A lot of software cost estimation models are suggesting help to provide high-quality estimates to aid project managers make the best make policy for the project (Abu-srhan et al., 2017; Aljahdali & Sheta, 2010; F., A., & Aljahdali, 2013). COCOMO model is widely used cost estimation model for algorithm software. It uses a fundamental regression formula, which contains the parameters obtained from historical project data, as well as at present features and next projects (Structures, n.d.).

The Constructive Cost Model (COCOMO) is one of the well-known model structures used to evaluate software effort. COCOMO model was researched and developed by Barry W. Boehm in 1981(A. F. Sheta, 2006). The model uses sixty three software projects for evaluation. The COCOMO model aids define the accurate economics formula used to determine software development time, effort, and maintenance (Jayakumar et al, 2012; A. F. Sheta, 2006).

COCOMO I is composed of three more and more in detail and precise forms of composition. They are basic, intermediate and detailed. Boehm put forward three project modes (Chawla, 2016; Nisar et al, 2009):

1. Organic mode – straightforward projects allow a few teams to work in the known and stability environment.
2. Semi-detached mode – The project attracted a team with extensive experience. It's between organic mode and embedded mode.
3. Embedded mode – complicated projects are developed within the restricting of keep changing requirement.

According to Boehm's the basic COCOMO shown in Equation 2.1 and Equation 2.2, the following form is adopted.

$$E = a * (KLOC)^b \quad 2.1$$

$$T = c * (E)^d$$

2.2

Where E presents the software effort calculated in person-months, KLOC stands Kilo Line Of Code. The values of the parameters a and b primarily determined by the category of the software projects. Where T presents the project duration calculated in months. The values of the parameter c and d are constants, and E is effort. The basic COCOMO models show in Table 2.1. Depending on the type of software projects, these models are estimated to have distinct results.

Table 2.1 Basic COCOMO Models

Model Name	Project Size	Effort (E)	Time (T)
Organic	Less than 50 KLOC	$E = 2.4 * (\text{KLOC})^{1.05}$	$T = 2.5 * (E)^{0.38}$
Semi-Detached	50-300 KLOC	$E = 3.0 * (\text{KLOC})^{1.12}$	$T = 2.5 * (E)^{0.35}$
Embedded	Over 300 KLOC	$E = 3.6 * (\text{KLOC})^{1.20}$	$T = 2.5 * (E)^{0.32}$

The major problem COCOMO model is that it cannot supply actual effort in the recent development environment. By exploring non-algorithmic techniques, such as Jaya algorithms or other natural heuristic algorithms, the limitations of COCOMO models can be overcome. Software effort evaluation based on existing parameters does not always give accurate results; therefore, we need to tune the parameters for more precise consequences. In this research, parameters a and b are optimized.

2.2.4 COCOMO II

The COCOMO II model was updated with COCOMO I in order to improve the software development practices in the 1990s and 2000. There are 3 submodels COCOMO II. They are applications composition, early design, and post-architecture. It may be conducted in different of manners to handle diverse software environments. The application composition model is used to evaluate the workload with progress of items that are usually developed as the rapid application. Early design models involved inquiry of option system framework and operational ideas. When the top-level design is completed, the post-architecture model is used, and more detail regarding the project can be obtained, and the software framework is well-defined (Estimation et al., 2008; Kumari & Pushkar, 2013).

2.3 Existing COCOMO I Model Based On Optimization Algorithms

In the literature, many machine learning (ML) methods are imposed to improve software cost estimation. The machine learning optimization algorithm inspiration comes from nature has gained more focus, thus finding more precise evaluation for software effort. Machine learning algorithms inspired by nature contain cuckoo search algorithm, genetic algorithm, firefly algorithm, and so on.

2.3.1 Cuckoo Search Algorithm

The cuckoo search (CS) is a natural population-based metaheuristic algorithm originally presented by Xin-she Yang and Suash Deb in 2009 to solve the optimization problem (Gómez, Iglesias, 2014; Yang & Deb, 2009). This algorithm is the inspiration for the cuckoo's reproductive strategies (Ali, 2015; Gómez, Iglesias, 2014). The cuckoo search (CS) algorithm is on account of the under 3 idealization regulations (Abu-srhan et al., 2017; Gómez, Iglesias, 2014; Yang & Deb, 2009):

- 1) Every cuckoo only has one egg at a time and pours it into a casually selected nest.
- 2) High-quality egg nests are considered the best nest for the next generation.
- 3) The number of host nests available is fixation, and the host may find the alien egg with a probability of $p_a \in (0, 1)$. Under the circumstances, the host bird can toss the egg out or give up the nest, so that a new nest can be built in a new place.

Based on these 3 regulations, Cuckoo Search algorithm (CSA) elementary procedure can be summarized in pseudocode shown in Figure 2.1.

Algorithm 1 A Cuckoo Search Algorithm Pseudo-code

```
begin
Objective function  $f(\mathbf{x})$ ,  $\mathbf{x} = (x_1, \dots, x_D)^T$ 
Generate initial population of  $n$  host nests  $x_i$  ( $i = 1, 2, \dots, n$ )
while ( $t < \text{MaxGeneration}$ ) or (stop criterion)
    Get a cuckoo (say,  $i$ ) randomly by Lévy flights
    Evaluate its fitness  $F_i$ 
    Choose a nest among  $n$  (say,  $j$ ) randomly
    if ( $F_i > F_j$ )
        Replace  $j$  by new solution
    end
    A fraction ( $p_a$ ) of worse nests are abandoned and new ones are built
    via Lévy
    Keep the best solutions (or nests with quality solutions)
    Rank the solutions and find the current best
end while
Postprocess results and visualization
end
```

Figure 2.1 Cuckoo Search Algorithm pseudo-code

Source: (Gálvez, Iglesias, 2014; Yang & Deb, 2009).

2.3.2 Genetic Algorithm

Genetic algorithm (GA) is search algorithm base on nature choosing and natural hereditist mechanism. It was developed by John Holland in the 1970s, his co-workers, and his pupils at the University of Michigan (Ali, 2015). Genetic algorithm (GA) is ground on the following four major constituent parts (Bhatia, Bawa, & Attri, 2015; Galinina, Burceva, & Parshutin, 2012; Jayakumar et al, 2012):

- 1) Chromosome. It is an independently represents the mission solution. And can use binary encoding, integer encoding and other encoded digital lines. Every situation in the chromosome is called a bit or gene.
- 2) Initial population. The first is a series of tasks population of solutions randomly generated. The major criteria for the first population generation procedures is to obtain various solutions.
- 3) Operator set. Operator settings allow new solutions to be generated based on the current population. The set of operators involves selection, crossover, and mutation. Figure 2.2 shows the genetic algorithm pseudo-code.
 - a) Selection: In the alternative, the individual is chosen in the middle group. There are known diverse types of choices such as Roulette wheel selection and Tournament selection.
 - b) Crossover: It concerns two strings mate randomly chosen. The result of this procedure is the creation of new individuals. The most simple type is the n-type crossover changes may be made by on the other hand type of cross.
 - c) Mutation: The use of the mutant chromosome gene changes its value with the probability of the definition. The new values of the gene are also decided by the probability of the definition. Using the uniform one-bit mutation function, in this function, the bits may change its state from 0 to 1 or from 1 to 0.
- 4) Fitness function. It is individual estimation attribute. The fitness function is used to determine individual goals. Individuals with larger adaptive values should be more stable and approach the solutions.

Algorithm 2 A Genetic Algorithm Pseudo-code
Objective function $f(x)$, $x = (x_1, \dots, x_D)^T$ Generate an initial population of individuals Evaluate the fitness of all individuals while termination condition not met do Select fitter individuals for reproduction Recombine between individuals Mutate individuals Evaluate the fitness of the modified individuals Generate a new population end while

Figure 2.2 Genetic Algorithm pseudo-code

Source: (Galinina et al., 2012; Jayakumar et al, 2012)

2.3.3 Firefly Algorithm

Firefly algorithm (FA) is the multi-mode optimize algorithm, that part of the realm of natural, motivated by the fireflies gleaming behaviour. FA was first introduced by Xin-She at Cambridge University in 2007. Firefly algorithm experience has proven that it can solve problems more naturally and it is possible to over perform other metaheuristic algorithms (Ghatasheh, Faris, Aljarah, & Al-Sayyed, 2015).

The Firefly Algorithm (FA) follows the following idealization regulations (Yang, 2009):

- 1) Each of fireflies are gender-neutral, in this way a firefly will be engaged to other, no matter what of their sex.
- 2) The attraction force is proportionate to their luminance, for any 2 flashes, therefore, the low luminance brightness closer to the brightness. Attraction is proportionate to the lightness and decreases with increasing distance.
- 3) Firefly luminance is influenced or decided by the goal function. For maximum problems, the luminance can be simply proportionate to the value of the goal function. Other modality of similar luminance can be defined genetic algorithm fitness function.

Based on these 3 regulations, firefly algorithm elementary procedure can be summarized in pseudocode shown in Figure 2.3.

Algorithm 3 A Firefly Algorithm Pseudo-code
<p>Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, \dots, x_d)^T$</p> <p>Generate initial population of fireflies \mathbf{x}_i ($i = 1, 2, \dots, n$)</p> <p>Light intensity I_i at \mathbf{x}_i is determined by $f(\mathbf{x}_i)$</p> <p>Define light absorption coefficient γ</p> <p>while ($t < \text{MaxGeneration}$)</p> <p> for $i = 1: n$ all n fireflies</p> <p> for $j = 1: n$ all n fireflies (inner loop)</p> <p> if ($I_i < I_j$), Move firefly i towards j; end if</p> <p> Vary attractiveness with distance r via $\exp[-\gamma r]$</p> <p> Evaluate new solutions and update light intensity</p> <p> end for j</p> <p> end for i</p> <p> Rank the fireflies and find the current global best g^*</p> <p> end while</p> <p>Postprocess results and visualization</p>

Figure 2.3 Firefly Algorithm pseudo-code

Source: (Ghatasheh et al., 2015; Yang, 2009)

There are two main issues with the attraction of flies in firefly algorithms; attractive model and various light intensity models. For a particular firefly at the place X luminance I is formula for $I(X) = \alpha f(X)$. The attraction β is proportionate to the flies, the distance $R_{i,j}$, between fireflies i and j relating to the firefly (Ghatasheh et al., 2015). Furthermore, FA has two main advantages: first, it is automatic segmentation capability, and second is the ability to deal with the multi-mode state. This automatic subdivision capability makes it suitable for highly nonlinear, multi-mode optimization problems (Shukla & Singh, 2017).

2.3.4 Jaya Algorithm

The Jaya is a simplicity and formidable global optimization algorithm already triumphantly imposed restriction and unrestraint problems benchmark function (Du, Vinh, Trung, Hong Quyen, & Trung, 2017; Pandey, 2016). The Jaya algorithm proposed by Venkata Rao is a population-based method based on global search (Pandey, 2016). It is ground on the truth that a given problem can be solved to the optimal solution and avert a failure solution. The advantage of the algorithm needs a few control parameters, for example, the greatest of generations, population scale, and the number of design variables, the majority of cases, any algorithm is universal. The absence of specific algorithm and control parameters that demand adjustment is feasible for the real computation to be carried. (Pandey, 2016).

The purpose is to change the objective function $f(x)$ to a minimum or maximum value. In any iteration i , assuming there are ‘ m ’ design variable (i.e. $j = 1, 2, \dots, m$), ‘ n ’ candidate solutions (i.e. population size, $k = 1, 2, \dots, n$). Throughout the solution candidate, the best candidate get $f(x)$ (i.e. $f(x)_{best}$) is the optimum value, while the failure candidate obtained $f(x)$ (i.e. $f(x)_{worst}$) the failure value. If $X_{j,k,i}$ is the value of the j^{th} variable for the k^{th} candidate during the i^{th} iteration, then this value will be altered according to Equation 2.3 below (Venkata Rao, 2016).

$$X'_{j,k,i} = X_{j,k,i} + r_{1j,i} (X_{j,best,i} - |X_{j,k,i}|) - r_{2j,i} (X_{j,worst,i} - |X_{j,k,i}|) \quad 2.3$$

Where, $X_{j,best,i}$ is the value of the variable j for the best candidate and $X_{j,worst,i}$ is the value of the variable j for the worst candidate. $X'_{j,k,i}$ is the updated value of $X_{j,k,i}$ and $r_{1j,i}$ and $r_{2j,i}$ are 2 random numbers for the j^{th} variable during the i^{th} iteration in the range $[0,1]$. The term “ $r_{1j,i} (X_{j,best,i} - |X_{j,k,i}|)$ ” represents the trend toward a solution that tends to be closer to the best solution and the “ $- r_{2j,i} (X_{j,worst,i} - |X_{j,k,i}|)$ ” express the solution tendency to avoid the worst solution. If it provides a better function value then $X'_{j,k,i}$ is usable. At the end of the iteration, all acceptable function values will remain unchanged, and these previous values will be the next iteration of input. (Pandey, 2016; Venkata Rao, 2016). Below the Figure 2.4 will show the pseudo-code of the Jaya algorithm.

Algorithm 4 Jaya Algorithm Pseudo-code

Start

Initialize the random population, maximum iteration number and iteration count variable iter=0.

While (iter < max iter number)

Iter = iter +1

Evaluate functional value of the each one of Population

Obtain new solutions using Equation 2.3 as follow:

$$X'_{j,k,i} = X_{j,k,i} + r_{1j,i} (X_{j,best,i} - |X_{j,k,i}|) - r_{2j,i} (X_{j,worst,i} - |X_{j,k,i}|)$$

Accept new solution if it gives a better

Functional value.

Conduct the local search.

Start Mutation strategy

End mutation strategy

End while

End

Figure 2.4 Pseudocode of the Jaya Algorithm

Source : (Banerjee & Sarkar, 2017)

2.4 Analysis of The Related Work

Comparison the software cost estimation techniques shown in Table 2.2.

Table 2.2 Comparison the software cost estimation techniques

Related Works	(F. & Aljahdali, 2013)	(Suri & Ranjan, 2012)	(Chawla, 2016)
Feature			
Research Title	Comparative analysis Between FPA and COCOMO Techniques for Software Cost Estimation	Comparative Analysis of Software Effort Estimation Techniques	Software Development Effort Estimation Techniques: A Review
Cost Estimation Models	Function Point Analysis (FPA)	Putnam's Software Life-cycle Model (SLIM)	COCOMO I, COCOMO II
Advantage	<ul style="list-style-type: none"> It gives a dependable effort relationship. 	<ul style="list-style-type: none"> Quick and simple. Appropriate for major scale projects. There are fewer characteristics and inputs for evaluation than for COCOMO. 	<ul style="list-style-type: none"> Most frequently accustomed model. Local project data can be calibrated and customized.
Disadvantage	<ul style="list-style-type: none"> Low evaluation accuracy. 	<ul style="list-style-type: none"> Absence of element needed to evaluate some of the software. 	<ul style="list-style-type: none"> Imprecise size and cost drives mensuration will result in poor estimates.

Based on the existing research more focus on software cost estimation techniques. Software cost estimation can be regarded as a basic activity, which requires correct methods and techniques to complete a good estimate of the results. Several cost estimation methods are studied and compared in this paper. These three methods are Function point Analysis, Putnam SLIM and Constructive Cost Model (COCOMO). No method is consequentially better or worse than the other one. In reality, their advantages and disadvantages are always supplementary. The COCOMO I model is easy to use and most frequently accustomed model. So, this research will focus on the COCOMO I model.

COCOMO I model comparison the algorithm between Cuckoo Search Algorithm, Genetic Algorithm, Firefly Algorithm, and Jaya algorithm shown in Table 2.3.

Table 2.3 COCOMO I Model Comparison between CS, GA, FA and JA

Related Works	(Kumari S., Pushkar S., 2017)	(Galinina et al., 2012)	(Ghatasheh et al., 2015)	Current Study
Features				
Research Title	Software Cost Estimation Using Cuckoo Search	Optimizing Basic COCOMO Model using Simplified Genetic Algorithm	Optimizing Software Effort Estimation Models Using Firefly Algorithm	Parameter Estimation Of COCOMO Model Using The Jaya Algorithm
Methods	Cuckoo Search Algorithm	Genetic Algorithm	Firefly Algorithm	Jaya Algorithm
Advantage	<ul style="list-style-type: none"> • Low parameter. • It satisfies the fast convergence. • It supports both local and global search capability. 	<ul style="list-style-type: none"> • It acts on the number of optional scheme, not just as a one solution. 	<ul style="list-style-type: none"> • Parameter tuning is an algorithm that learns good parameter values to balance the opportunities between exploration and development. • It can effectively deal with highly nonlinear and multi-model optimization problems. 	<ul style="list-style-type: none"> • Fast forward to the optimal solution.
Disadvantage	<ul style="list-style-type: none"> • It's easy to fall into the local optimum. 	<ul style="list-style-type: none"> • It's easy to get stuck in a local minimum. 	<ul style="list-style-type: none"> • For high-dimensional and nonlinear problems, it will have excessive restriction. 	<ul style="list-style-type: none"> • It does not require adjusting any algorithm-specific control parameter

Based on the existing research, other researchers focus more on software cost estimation using optimization algorithms. In software cost estimation, COCOMO coefficient is optimized by the Genetic algorithm. An evolutionary model evolved by genetic algorithms for estimating efforts. The results show that the new model triumphantly advances the execution by minimizing MMRE. The value of the COCOMO I model coefficient is acquired by combining the Genetic algorithm and Firefly Algorithm. The Genetic algorithm and Firefly Algorithm triumphantly advance the precise contrasted to COCOMO. The Firefly Algorithm and Cuckoo Search Algorithm are very rival and the result very proximate. For another Genetic algorithm has the minimum results and slowest convergence. So, the Cuckoo Search Algorithm as a meta-heuristic optimization algorithm over performs Firefly Algorithm and Genetic Algorithm in terms of higher estimation precise for the software cost estimation COCOMO I model based model. The Jaya algorithm can find the optimal solution and the optimum the MMRE. The Jaya algorithm and the Firefly algorithm result very close. So, Jaya algorithm is chosen in this research.

2.5 Summary

In this chapter, the literature review has been completed. The discussion is carried on the introduction of COCOMO model and refer to the problem. The existing COCOMO model based on optimization algorithms will provide in this chapter such as CSA, GA, FA and JA. After that, the analysis comparison the software cost estimation and optimization algorithm are shown in the Table 2.2 and Table 2.3.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In chapter 2, this research was explained regard COCOMO model and the literature review will concentrate on the existing COCOMO model based on optimization algorithm which are Cuckoo Search Algorithm, Genetic Algorithm, Firefly Algorithm and Jaya Algorithm.

In this chapter was discussed on the research methodology using the Jaya Algorithm to calculate estimate the effort. The datasets are used in the training process. The Jaya algorithm will be explained in describes in this chapter. Then, the hardware and software specifications required for this study are also defined.

3.2 Research Activity

Research activity is one procedure to solve the research problem. In this research activity, we should not only describe the ways to be applied in the research but also think the logic behind the method logic. In Figure 3.1 show the research activity composed of six phases such as problem defining, concepts and literature review, user design, implementation, experiment results, and documentation.

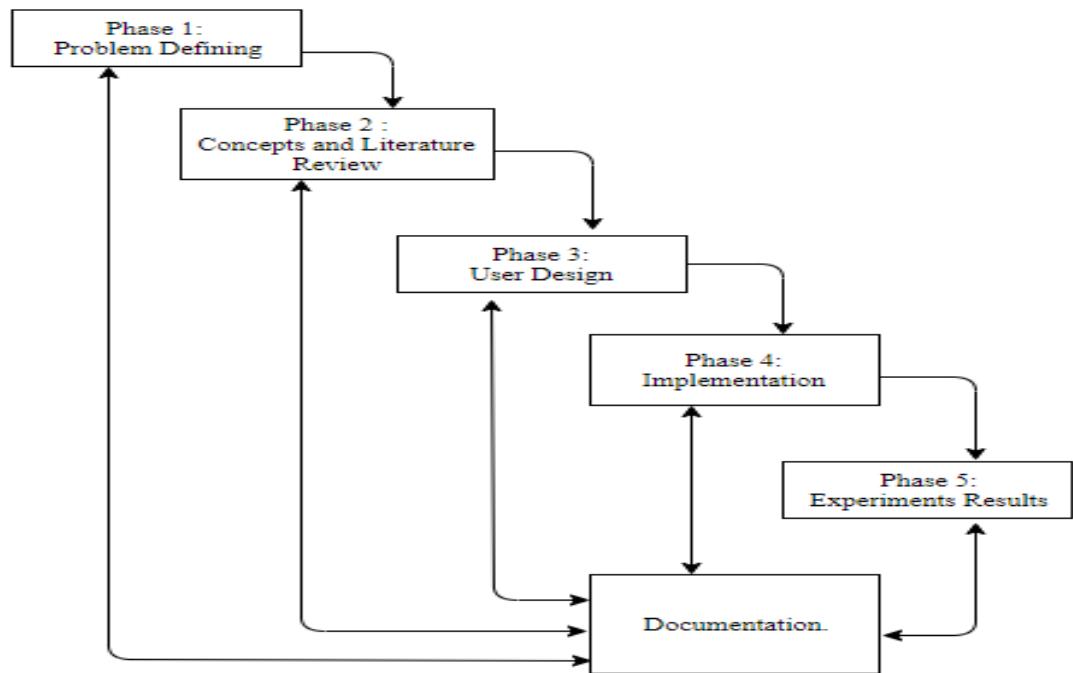


Figure 3.1 Research Activity Phases

3.2.1 Phase 1: Problem defining

At this phase, the research topic is chosen and the problem is defined in Chapter 1. The problem is to find out the minimum of parameter values (a and b), then calculated the estimated effort based on Jaya Algorithm (JA) and we will analyse the project before it is confirmed. Debates on the objectives, scope, and significance of the research will be carried out prior to the next stage for purpose of having a clear instruction in this research. This phase will be recorded in documentation.

3.2.2 Phase 2: Concepts and Literature Review

Research on literature review is being conducted at this phase. We studied the concepts of the software cost estimation models and the existing COCOMO I model based on the optimization algorithm which are CSA, GA, FA, and JA. In literature review, we focus on the COCOMO I model based on the optimization algorithms that have been done by previous research. This phase will be recorded in documentation.

3.2.3 Phase 3: User Design

At this phase, an overview of the research activity and the process of the COCOMO I model diagram will be created for the thesis phase. This phase will be recorded in documentation.

3.2.4 Phase 4: Implementation

At this phase, there will be an implementation process. The fourth and fifth chapter of the Final Year Project 2 will cover this stage. The Java programming language is going to build and test by using NetBeans IDE 8.1 tool. All the implementation are documented.

3.2.5 Phase 5: Experiment Result

At this phase, we will collect the results of the experiment. This stage also ensures that the proposed algorithm will use the Jaya algorithm and test it to optimize the values of parameters a, b, c, and d. The comparison of the estimation effort and time using the Jaya algorithm. All the experimental result will be documented.

3.2.6 Phase 6: Documentation

The main goal of this documentation is to process the document and interpret how our research is conducted which is parameter estimation of COCOMO I model using the Jaya Algorithm. All documentation will be recorded in this paper. In the final year project 2, this research report needs to be submitted, so it is necessary to record and determine all phases of activities.

3.3 Methodology

The COCOMO model is used to estimate the software cost. This section will further discuss the parameter estimation of the COCOMO model using the Jaya algorithm. The overview of COCOMO I model shown in Figure 3.2. It is divided into three phases. The first phase is training phase using the Jaya Algorithm. The second phase is testing phase using Jaya algorithm and lastly is testing using standard COCOMO.

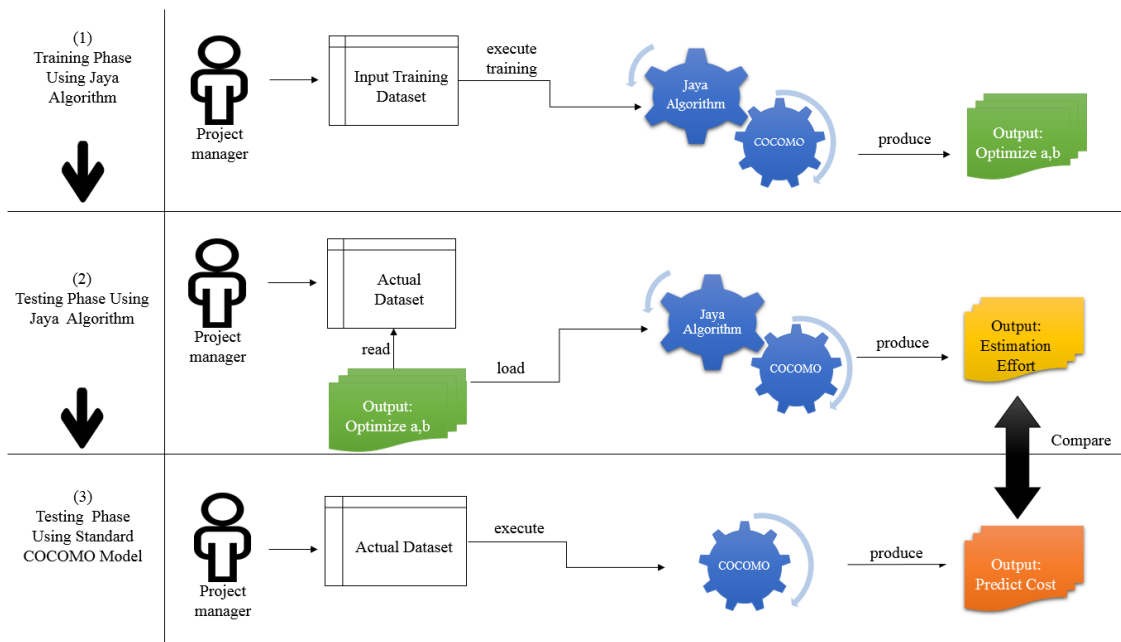


Figure 3.2 The overview of COCOMO model using the Jaya Algorithm

These phases and algorithms will be further debated and expounded in the next three subsections.

3.3.1 Training Phase using Jaya Algorithm

During the training phase, the optimal values of the COCOMO model parameters a , b , c and d are estimated by using the training dataset. The Jaya algorithm was used to train the dataset, and new estimation of COCOMO model parameter a , b , c and d was obtained. According to the training result, the optimal value is selected as the value of the parameter a , b , c and d . Optimization parameter can minimize MMRE (Mean Magnitude Relative Error) and achieve the accurately estimation effort and time.

3.3.2 Testing Phase using Jaya Algorithm

During the testing phase, we implement the COCOMO model with the optimal values of the parameters obtained in the training phase. The basic COCOMO model Equation 2.1 and Equation 2.2 (refer in Chapter 2) to executed the optimal values of the parameter a , b , c and d . Based on this, a new Jaya algorithm model equation is proposed. Then, we used the Jaya algorithm equation to compute the estimated effort and time of the project.

3.3.3 Testing Phase Using Standard COCOMO Model

In the last phase, only the COCOMO model is applied to assess the test phase of the standard COCOMO. The actual dataset will execute the COCOMO model only and does not touch through the Jaya Algorithm. Estimate the effort of the project in actual effort using the basic COCOMO model equations and established the values of the parameter. After that, the comparison between the estimation effort and actual effort.

3.4 Hardware and Software Requirement

This section describes the hardware and software used in this research.

3.4.1 Hardware Specification

The following Table 3.1 shows the hardware specifications required for this research.

Table 3.1 List of Hardware Specification

Hardware	Purpose
Laptop Asus	To make a proposal documentation.
Hard disk	Back up the files
Printer	Print the document

3.4.2 Software Specification

The following Table 3.2 shows the hardware specifications required for this research.

Table 3.2 List of Software Specification

Software	Purpose
Microsoft Word 2013	Used to write all required documents.
Microsoft Power Point 2013	Prepare the presentation slide and diagram.
Flowchart Maker	Draw flow chart.
Google Chrome	Search some information concerning the research.
NetBeans IDE 8.1	Build and implement the Jaya Algorithm code.
Microsoft Project 2010	Draw Gantt chart.

3.5 Gantt Chart

The Gantt chart shows the estimated duration from the beginning of the project to the end of the project (refer to Appendix A).

3.6 Implementation

In this research, we have collected COCOMO I datasets from 63 projects of NASA software project, Turkish dataset and Kemerer dataset. The Jaya Algorithm would discuss in detail how to optimize the values of parameter a, b, c, and d will be implemented.

3.6.1 Jaya Algorithm

The Jaya is a simplicity and formidable global optimization algorithm already triumphantly imposed restriction and unrestraint problems benchmark function (Du et al., 2017; Pandey, 2016). The Jaya algorithm proposed by Venkata Rao is a population-based method based on global search (Pandey, 2016). It is ground on the truth that a given problem can be solved to the optimal solution and avert a failure solution. The advantage of the algorithm needs a few control parameters, for example, the greatest of generations, population scale, and the number of design variables, the majority of cases, any algorithm is universal. The absence of specific algorithm and control parameters that demand adjustment is feasible for the real computation to be carried. (Pandey, 2016).

The purpose is to change the objective function $f(x)$ to a minimum or maximum value. In any iteration i , assuming there are 'm' design variable (i.e. $j = 1, 2, \dots, m$), 'n' candidate solutions (i.e. population size, $k = 1, 2, \dots, n$). Throughout the solution candidate,

the best candidate get $f(x)$ (i.e. $f(x)_{\text{best}}$) is the optimum value, while the failure candidate obtained $f(x)$ (i.e. $f(x)_{\text{worst}}$) the failure value. If $X_{j,k,i}$ is the value of the j^{th} variable for the k^{th} candidate during the i^{th} iteration, then this value will be altered according to Equation 3.1 below (Venkata Rao, 2016).

$$X'_{j,k,i} = X_{j,k,i} + r_{1j,i} (X_{j,\text{best},i} - |X_{j,k,i}|) - r_{2j,i} (X_{j,\text{worst},i} - |X_{j,k,i}|) \quad 3.1$$

Where, $X_{j,\text{best},i}$ is the value of the variable j for the best candidate and $X_{j,\text{worst},i}$ is the value of the variable j for the worst candidate. $X'_{j,k,i}$ is the updated value of $X_{j,k,i}$ and $r_{1j,i}$ and $r_{2j,i}$ are 2 random numbers for the j^{th} variable during the i^{th} iteration in the range $[0,1]$. The term “ $r_{1j,i} (X_{j,\text{best},i} - |X_{j,k,i}|)$ ” represents the trend toward a solution that tends to be closer to the best solution and the “ $- r_{2j,i} (X_{j,\text{worst},i} - |X_{j,k,i}|)$ ” express the solution tendency to avoid the worst solution. If it provides a better function value then $X'_{j,k,i}$ is usable. At the end of the iteration, all acceptable function values will remain unchanged, and theseprone values will be the next iteration of input. (Pandey, 2016; Venkata Rao, 2016). Below the Figure 3.3 will show the pseudo-code of the Jaya algorithm.

Algorithm 4 Jaya Algorithm Pseudo-code

Start

Initialize the random population, maximum iteration number and iteration count variable iter=0.

While (iter < max iter number)

Iter = iter +1

Evaluate functional value of the each one of Population

Obtain new solutions using Equation 3.1 as follow:

$$X'_{j,k,i} = X_{j,k,i} + r_{1j,i} (X_{j,best,i} - |X_{j,k,i}|) - r_{2j,i} (X_{j,worst,i} - |X_{j,k,i}|)$$

Accept new solution if it gives a better

Functional value.

Conduct the local search.

Start Mutation strategy

End mutation strategy

End while

End

Figure 3.3 Pseudocode of the Jaya Algorithm

Source : (Banerjee & Sarkar, 2017)

3.6.2 Proposed Estimation Effort for the Jaya Algorithm

The model seeks by the application Jaya algorithm to supply the best precision in software cost estimation procedure. The major objective here is to optimize the Equation 2.1 (refer in Chapter 2) shown in the effort estimation of the parameter a and b.

Before Jaya algorithm is applied to a given project, the random population size, maximum iteration, and iteration count variable for each parameter was initialization. The next step is to use the Jaya algorithm for adjustment parameter. The goal is to optimize the parameter a and b. After that, using the Jaya algorithm Equation 3.1 (refer in Chapter 3) to calculate the estimation effort. If the values of the parameter a and b are better, then it will keep the new values of parameter a and b. After that, use the best a and best b to train and test the dataset, to find out the estimation effort. Finally, we use the

accuracy of the estimation effort compared with the actual effort. In order to optimize COCOMO I model parameters a and b, Jaya algorithm is proposed shown in Figure 3.4.

Our purpose is to the minimum the error which is called MMRE (Mean Magnitude of Relative Error) between the estimation effort and actual effort. This is commonly used to measure the property of the model in the procedure of software cost estimation. According to the following is the MMRE Equation 3.2:

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{Actual\ Effort - Estimated\ Effort}{Actual\ Effort} \right| \quad 3.2$$

Where n, is the total number of projects. The Actual Effort is taken from three different datasets, Estimation Effort is an estimated effort which is the effort we get from our algorithm.

Pseudocode for Proposed of the Estimation Effort Jaya Algorithm
Initialize the random population size, maximum iteration number and iteration count variable iter=0 While (<i>iter</i> < maximum iteration number) For <i>i</i> =0 : n all n population size Modify the solution using the Equation 3.1 Evaluate fitness of the each one of population using the fitness function using Equation 3.2 If ($X'_{j,k,i} > X_{j,k,i}$) Then Update the previous solution Else No update in the previous solution End If End For End while Display the optimum result

Figure 3.4 Pseudocode for Proposed of the Estimation Effort Jaya Algorithm

1. Initialize the random population size, maximum iteration number and iteration count variable iter=0

Randomly the population size and maximum iteration number. Then the iteration count variable start to 0. When the number is recognized as the maximum iteration number, which acts as a stop criterion. Jaya Algorithm terminates when the stop condition is satisfied.

2. Modify the solution using the Equation 3.1 (refer in Chapter 3)

Modify the solution using Jaya algorithm Equation 3.1

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i} (X_{j,best,i} - |X_{j,k,i}|) - r_{2,j,i} (X_{j,worst,i} - |X_{j,k,i}|) \quad 3.1$$

3. Evaluate fitness of the each one of population size using the fitness function using Equation 3.2 (refer in Chapter 3)

Evaluation the fitness functions of each population size by fitness function,

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{Actual\ Effort - Estimated\ Effort}{Actual\ Effort} \right| \quad 3.2$$

After the fitness evaluation, if a better solution is produced in the population, the new location will be updated.

4. Update the previous solution

If the best parameters a and b will update and keep the values

5. No update in the previous solution

If the worst parameters a and b will no update

6. Display the optimum result

The algorithm parameter estimate terminates when the stop condition is met. Alternatively, repeat step 2 until step 5. Finally, at the end of the calculation, the optimal population is selected, that is, the optimal value of the result.

3.6.3 Proposed Time Estimation of COCOMO Model using the Jaya Algorithm

The objective of this program is to calculate and optimize the values of the parameter c and d present in the COCOMO model Equation 2.2 (refer in Chapter 2). In order to optimize COCOMO model values of the parameter c and d , the Jaya algorithm is proposed. The step is as follows shown in Figure 3.5.

Pseudocode for Proposed of the Estimation Time Jaya Algorithm
Initialize the random population size, maximum iteration number and iteration count variable $iter=0$ While ($iter < \text{maximum iteration number}$) For $i = 0 : n$ all n population size Modify the solution using the Equation 3.1 Evaluate fitness of the each one of population using the fitness function using Equation 3.2 If ($X'_{j,k,i} > X_{j,k,i}$) Then Update the previous solution Else No update in the previous solution End If End For End while Get the estimate effort result Use the time Equation 2.2 $Time = c * (E)^d$ Display the optimum result

Figure 3.5 Pseudocode for Proposed of the Estimation Time Jaya Algorithm

1. Initialize the random population size, maximum iteration number and iteration count variable iter=0

Randomly the population size and maximum iteration number. Then the iteration count variable start to 0. When the number is recognized as the maximum iteration number, which acts as a stop criterion. Jaya Algorithm terminates when the stop condition is satisfied.

2. Modify the solution using the Equation 3.1

Modify the solution using Jaya algorithm Equation 3.1 (refer in Chapter 3)

$$X'_{j,k,i} = X_{j,k,i} + r1_{j,i} (X_{j,best,i} - |X_{j,k,i}|) - r2_{j,i} (X_{j,worst,i} - |X_{j,k,i}|) \quad 3.1$$

3. Evaluate fitness of the each one of population size using the fitness function using Equation 3.2 (refer in Chapter 3)

Evaluation the fitness functions of each population size by fitness function,

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{Actual\ Effort - Estimated\ Effort}{Actual\ Effort} \right| \quad 3.2$$

After the fitness evaluation, if a better solution is produced in the population, the new location will be updated.

4. Update the previous solution

If the best parameters a and b will update and keep the values

5. No update in the previous solution

If the worst parameters a and b will no update

6. Get the estimate effort result

Take the estimation effort result of each project number.

7. Use the time Equation 2.2

Evaluation the time estimation of each project number using the Equation 2.2 (refer in Chapter 2).

$$T = c * (E)^d \quad 2.2$$

8. Display the optimum result

The algorithm parameter estimate terminates when the stop condition is met. Alternatively, repeat step 2 until step 7. Finally, at the end of the calculation, the optimal population is selected, that is, the optimal value of the result of estimation effort and time.

3.7 Data Collection and Analysis

In this research, there have three different datasets which are NASA63 software project dataset, Turkish dataset, and Kemerer dataset. The NASA63 dataset need data conversion because the NASA63 project data point uses the COCOMO I model data format. A data conversion is not needed for the Turkish dataset.

3.7.1 NASA63 Software Project Dataset

By experimenting with the data sets provided by Bailey and Basili, we were able to build an effort estimation model (A. F. Sheta, 2006). The dataset consists of the Kilo Line of Code (KLOC) and actual effort variables. NASA63 Software Project Dataset start from project 1 to project 15 given in Table 3.3 and start form project 16 to project 63 given in Appendix B.

Table 3.3 NASA63 Dataset

Project No.	KLOC	Actual Effort
1	113.0	2040.0
2	293.0	1600.0
3	132.0	243.0
4	60.0	240.0
5	16.0	33.0
6	4.0	43.0
7	6.9	8.0
8	22.0	1075.0
9	30.0	423.0
10	29.0	321.0
11	32.0	218.0
12	37.0	201.0
13	25.0	79.0
14	3.0	60.0
15	3.9	61.0

Source : (Sachan et al., 2016)

We set the dataset of 70% use to training and 30% use to testing. The datasets from the first 44 projects were used to estimate the parameters, while the other 19 projects were used to test their performance.

Typically, parameters ground on the type of software project is fixed. Our purpose is using Jaya algorithms to make the new estimate for COCOMO model parameters. This will permit us to calculate the effort of NASA software projects. The estimated parameters will greatly extend the calculation of all projects development effort. We developed the following model using the Jaya algorithm.

3.7.2 Turkish Software Industry Dataset

Optimization experiments were carried out the Turkish dataset provided by Ekananta Manalif (Dhiman & Diwaker, 2013). The dataset consists of the Kilo Line of Code (KLOC) and actual effort variables. The Turkish dataset given in Table 3.4. We set the dataset of 70% use to training and 30% use to testing. The datasets from the first 8 projects were used to estimate the parameters, while the other 4 projects were used to test their performance.

Table 3.4 Turkish Dataset

Project No.	KLOC	Actual Effort
1	3.00	1.20
2	2.00	2.00
3	4.25	4.50
4	10.00	3.00
5	15.00	4.00
6	40.53	22.00
7	40.50	2.00
8	31.85	5.00
9	114.28	18.00
10	23.11	4.00
11	1.37	1.00
12	1.61	2.10

Source : (Manalif, Capretz, & Ho, 2013)(Menzies, Tim, n.d.)

3.7.3 Kemerer Dataset

The Kemerer dataset consists of 15 software projects, described by five independent features and one independent feature (Azzeh, n.d.). On Kemerer dataset, we apply the Jaya Algorithm method to find an estimate effort and display the optimum error rate. The Kemerer dataset consists of the Kilo Line of Code (KLOC) and considers the actual effort of each project.

The Kemerer dataset is given in Table 3.5. We set the dataset of 70% use to training and 30% use to testing. The datasets from the first 10 projects were used to estimate the parameters, while the other 5 projects were used to test their performance.

Table 3.5 Kemerer Dataset

Project No.	KLOC	Actual Effort
1	253.6	287.0
2	40.5	82.5
3	450.0	1107.3
4	214.4	86.9
5	449.9	336.3
6	50.0	84.0
7	43.0	23.2
8	200.0	130.3
9	289.0	116.0
10	39.0	72.0
11	254.2	258.7
12	128.6	230
13	161.4	157
14	164.8	246.9
15	60.2	69.9

Source : (Gharehchopogh, Maleki, & Reza Khaze, 2014)

3.8 Summary

This chapter discusses the methodology that will be used to conduct this research. The methodology divided three phases to discuss the overview of the COCOMO I model. This chapter also clearly lists the hardware and software specifications for this study.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter will discuss the implementation of parameter estimation of the COCOMO model using the Jaya Algorithm. This study was developed the NetBeans IDE version 8.1 and completed this study using Java as the programming language. The experimental results will be collected and compared to the results of the basic COCOMO model, random and Jaya algorithm. After that, the results will be discussed. The experiments used three different test datasets used by Jaya algorithm. The three datasets are NASA 63 software project, Turkish dataset and Kemerer dataset. The results of each test dataset are stored in the table to be compared.

4.2 Experimental Evaluations

The experimental evaluation of COCOMO I model is focusing on two main goals:

- i) To estimate the effort and time of the COCOMO I model using the Jaya algorithm.
- ii) To compare the effort estimation and time estimation with the original COCOMO and actual estimate.

Based on the before-mentioned goals, the Jaya algorithm evaluation is divided into two part. In the first part, the size performance to estimate the effort for the COCOMO I model. The second part is calculate the time using the estimate effort to calculate. The best values with the number of iteration (10000) and population size (100) is acquired.

The COCOMO I model described in this thesis is implemented using the Java programming language. The experiment was conducted in a desktop environment using NetBeans IDE 1.8 to perform the COCOMO I model. The machine specifications are as follows:

- i) Windows 8.1 Single Language and 64 bits Operating System
- ii) Intel (R) Core (TM) i5-5200U CPU @ 2.20GHz
- iii) 8 GB of RAM (Memory)
- iv) Java running environments (JRE) version 1.8

The experimental results are compared and rendered in several tables and graphs. the Tables 4.1 until the Table 4.6 shows the experimental results.

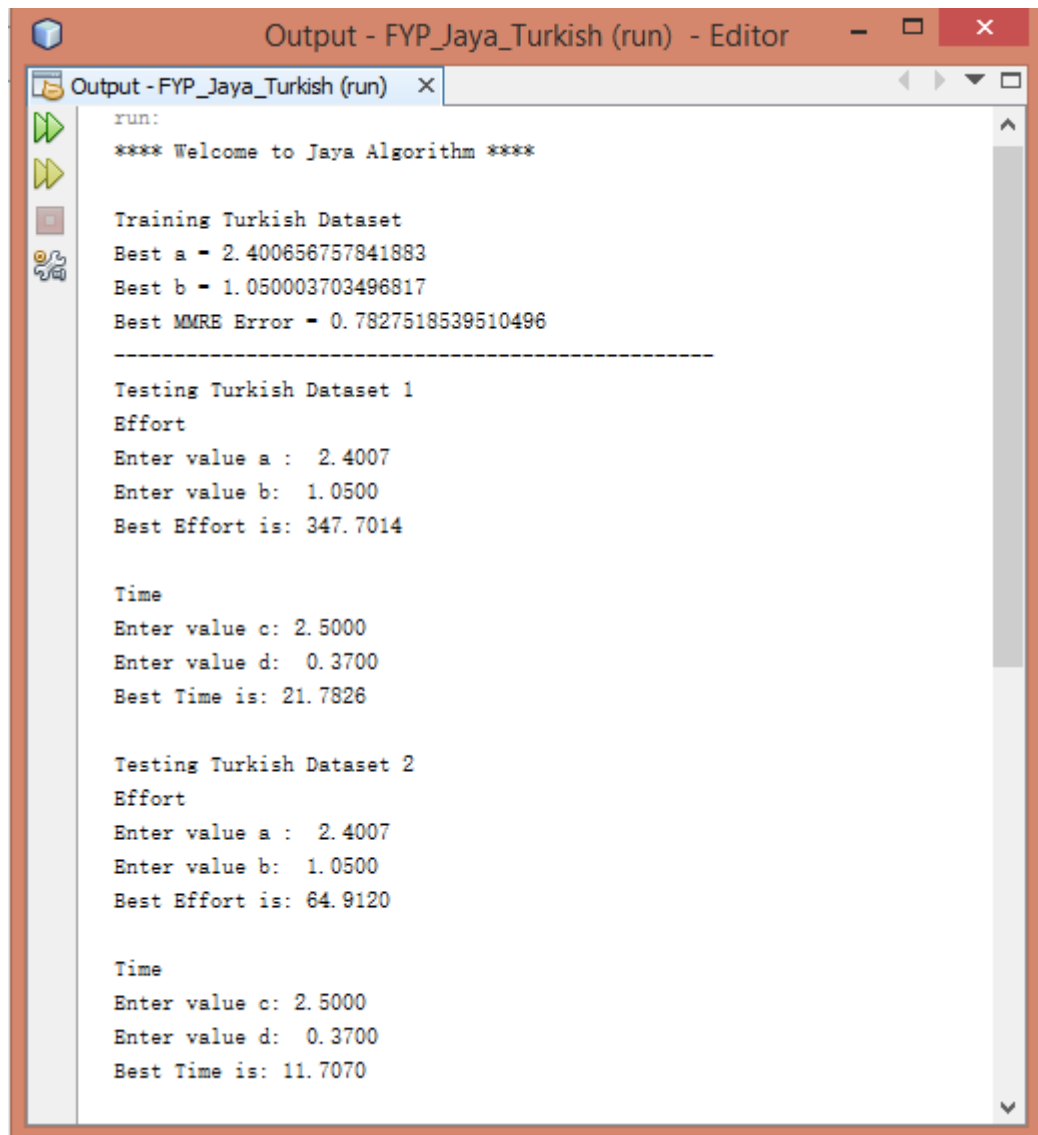
4.3 Experimental Results

In this experiment, the Jaya algorithm was used to optimize the a and b values of the basic COCOMO model shown in Equation 2.1 (refer to Chapter 2). The basic COCOMO model permits us to evaluate software development efforts for the NASA63 dataset, Turkish dataset, and Kemerer dataset. The fitness function is shown in Equation 3.2 (refer to Chapter 3). The calculated parameters can greatly predigest the evaluation of software effort for all projects.

In order to balances between the executive time and estimation accuracy. This research various tests combinations the iteration and population size were performed. The values of the various tests combinations which is (1000, 10), (1000, 50), (1000, 100), (10000, 10), (10000, 50), (10000, 100). Thereafter run the Jaya algorithm for each various tests combinations, the most appropriate combination is selected which is 10000 for the maximum iteration number and 100 for the population size. This combination can obtain the actual effort is not much different from the estimated effort and repetition of execute time is 18 minutes 26 seconds for Turkish dataset.

4.3.1 Generation Values of COCOMO Model Parameters

Running the program generate following the values there are the snapshot.



```
run:
**** Welcome to Jaya Algorithm ****

Training Turkish Dataset
Best a - 2.400656757841883
Best b - 1.050003703496817
Best MMRE Error - 0.7827518539510496
-----
Testing Turkish Dataset 1
Effort
Enter value a : 2.4007
Enter value b: 1.0500
Best Effort is: 347.7014

Time
Enter value c: 2.5000
Enter value d: 0.3700
Best Time is: 21.7826

Testing Turkish Dataset 2
Effort
Enter value a : 2.4007
Enter value b: 1.0500
Best Effort is: 64.9120

Time
Enter value c: 2.5000
Enter value d: 0.3700
Best Time is: 11.7070
```

Figure 4.1 Result Generating Parameters a, b, c, and d Value of COCOMO Model

The Figure 4.1 shows sample execution of generating a, b, c, and d values to be used the COCOMO model. This experiment executes the program design from Chapter 3 given the following values of effort and time. These values will be used to the subsection to generate the estimates.

4.3.2 Effort Estimation

The calculated values of effort estimation are rendered in Table 4.1, Table 4.2, and Table 4.3. The model with the best set parameters a and b is rendered in Equation 4.1, Equation 4.2 and Equation 4.3 for each dataset. The Figure 4.2 shows the measured effort comparison between NASA63 dataset, basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The Figure 4.3, shows the measured effort comparison between Turkish dataset, basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The Figure 4.4 shows the measured effort comparison between Kemerer dataset, basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The result shows that the effort estimated by the proposed Jaya Algorithm gives poor results compared to the actual effort.

$$E = 3.3904 * (KLOC)^{1.1963} \quad 4.1$$

$$E = 2.4001 * (KLOC)^{1.0500} \quad 4.2$$

$$E = 2.4006 * (KLOC)^{1.0500} \quad 4.3$$

4.3.2.1 NASA 63 Software Project Dataset Results

Table 4.1 Result of the Effort Estimation of the NASA63 Dataset

Project No.	KLOC	Actual Effort (Sachan et al., 2016)	Basic COCOMO (Effort)	Standard COCOMO (Effort)	COCOMO JA (Effort)
1	35.0	106.0	100.3420	241.5361	238.4975
2	73.0	126.0	217.1202	579.6598	574.6748
3	23.0	36.0	64.5692	146.5003	144.3253
4	464.0	1272.0	1513.7534	5244.1439	5251.9325
5	91.0	156.0	273.6558	753.6359	748.0565
6	24.0	176.0	67.5201	154.1167	151.864
7	10.0	122.0	26.9284	54.3335	53.2834
8	8.2	41.0	21.8633	42.8974	42.0227
9	5.3	14.0	13.8261	25.5103	24.9305
10	4.4	20.0	11.3720	20.4393	19.9544
11	6.3	18.0	16.5775	31.3407	30.6573
12	27.0	958.0	76.4088	177.3233	174.8441
13	17.0	237.0	47.0092	102.2121	100.5282
14	25.0	130.0	70.4771	161.794	159.4647
15	23.0	70.0	64.5692	146.5003	144.3253
16	6.7	57.0	17.6844	33.7245	33.0003
17	28.0	50.0	79.3829	185.1718	182.6191
18	9.1	38.0	24.3896	48.5614	47.5983
19	10.0	15.0	26.9284	54.3335	53.2834

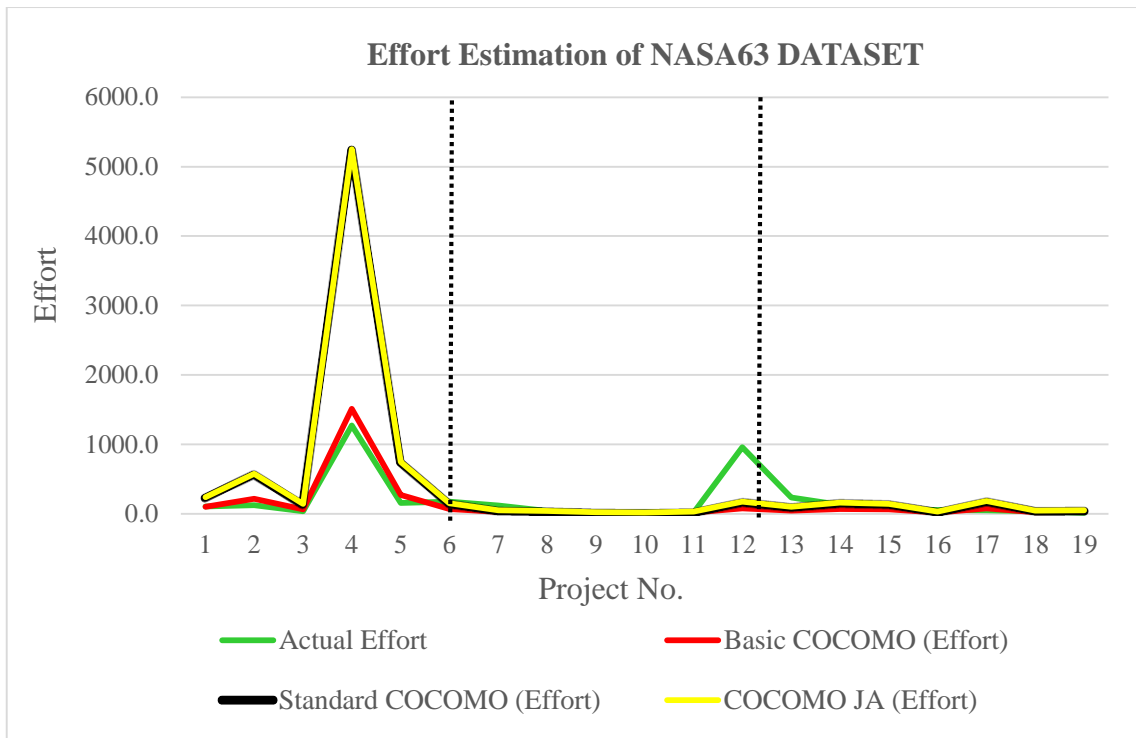


Figure 4.2 Comparison of the Effort Estimation of NASA63 Datasets

In Figure 4.2, the effort estimation pattern of NASA63 dataset for Actual effort, basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.2 is divided into three part. In the first part, we clearly see that the highest estimated effort for the Standard COCOMO and COCOMO JA, meanwhile the project 4 of the values of effort estimation for the both of this two COCOMO are most closely. While project 4 the actual effort is the lowest compared to the estimation effort of this three COCOMO. After, project 6 of the value of the effort estimation for the basic COCOMO, standard COCOMO, COCOMO JA, and actual effort will slightly decrease. In the second part, the project 7 to the project 12 the values of the estimation effort for the basic COCOMO, standard COCOMO and COCOMO JA more closely to the actual effort. While project 11 and project 12 the values of actual effort higher the values of estimation effort for the basic COCOMO, standard COCOMO, and COCOMO JA. In the third part, the values of the estimation effort actual effort, basic COCOMO, standard COCOMO, and COCOMO JA are more closely.

4.3.2.2 Turkish Dataset Results

Table 4.2 Result of the Effort Estimation of the Turkish Dataset

Project No.	KLOC	Actual Effort (Manalif, Capretz, & Ho, 2013)	Basic COCOMO (Effort)	Standard COCOMO (Effort)	COCOMO JA (Effort)
1	114.28	18.00	347.6002	349.4545	347.6924
2	23.11	4.00	64.8935	65.1595	64.9064
3	1.37	1.00	3.3402	3.3466	3.3404
4	1.61	2.10	3.9571	3.9652	3.9575

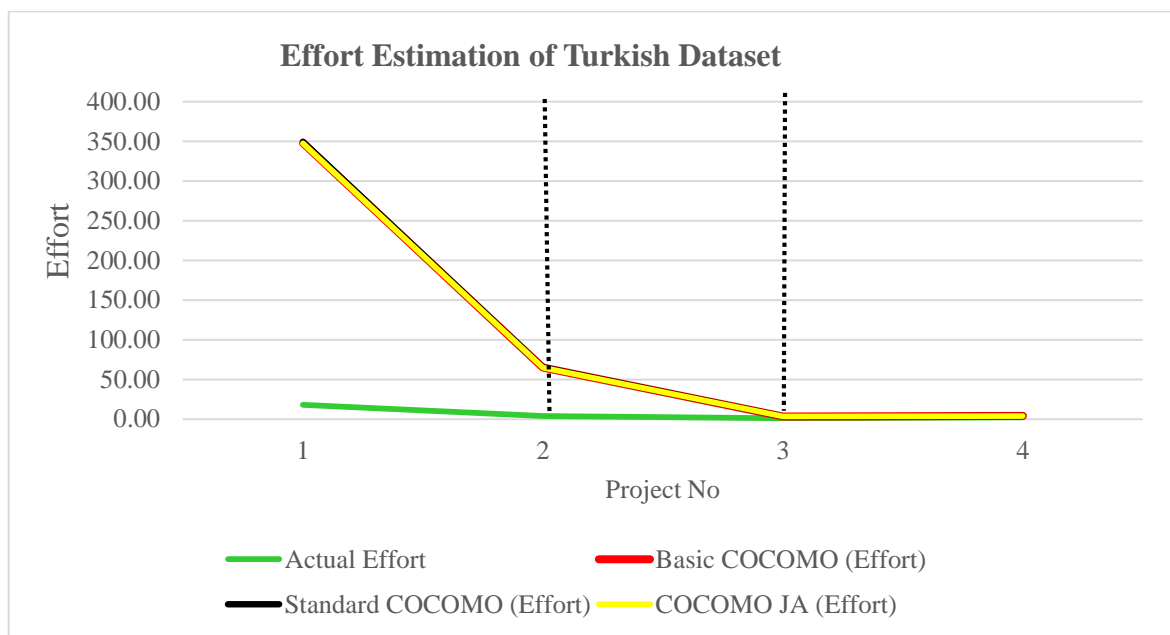


Figure 4.3 Comparison of the Effort Estimation of Turkish Dataset

In Figure 4.3, the effort estimation pattern of Turkish dataset for Actual effort, basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.3 is divided into three part. In the first part, we clearly see that the highest values of the estimated effort of the basic COCOMO, standard COCOMO, and COCOMO JA, meanwhile the value of both of that three COCOMO are most closely of the project 1, but the values of the estimation effort for the actual effort difference far. While project 1 to the project 4 the values of the estimation effort for the actual effort is lowest. In the second part, we clearly see that the graph will slightly decrease and close to the values of the estimation effort for the actual effort. The project 2 the values of the estimation effort of the basic COCOMO, standard COCOMO, and COCOMO JA are very close to the values of estimation effort for the actual effort. In the third part, the

project 3 and project 4 the values of the estimation effort for the actual effort and the values of estimation effort for the basic COCOMO, standard COCOMO, and COCOMO JA are almost close the effort.

4.3.2.3 Kemerer Dataset Results

Table 4.3 Result of the Effort Estimation of Kemerer Dataset

Project No.	KLOC	Actual Effort (Gharehchopogh, Maleki, & Reza Khaze, 2014)	Basic COCOMO (Effort)	Standard COCOMO (Effort)	COCOMO JA (Effort)
1	254.2	258.7	804.7214	812.273	804.9877
2	128.6	230	393.4724	397.1473	393.5987
3	161.4	157	499.4706	504.1429	499.6326
4	164.8	246.9	510.5242	515.3006	510.6899
5	60.2	69.9	177.3322	178.9796	177.3872

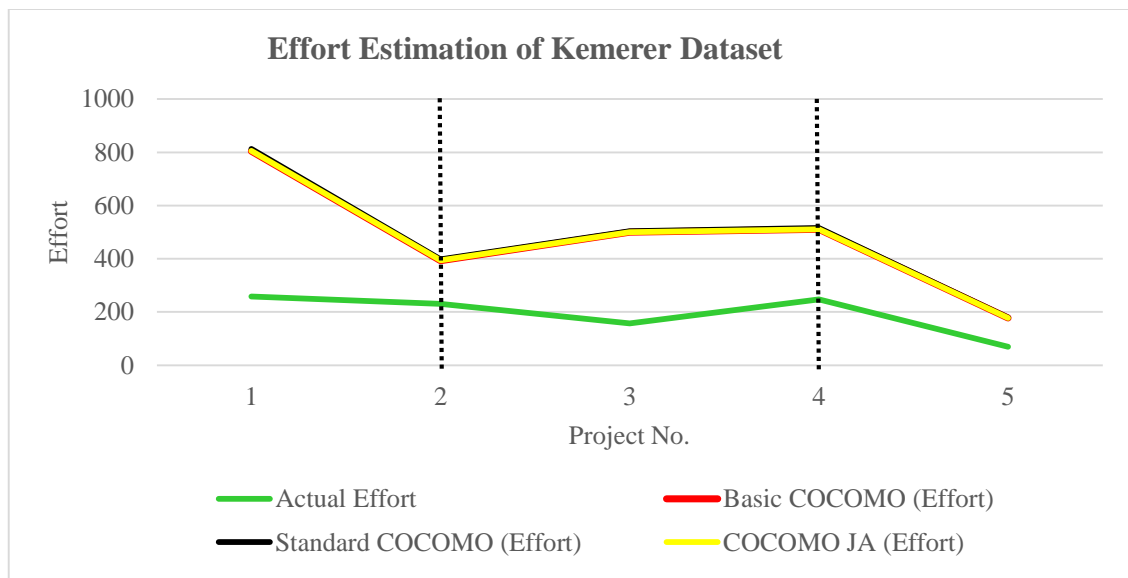


Figure 4.4 Comparison of the Effort Estimation of Kemerer Dataset

In Figure 4.4, the effort estimation pattern of Kemerer dataset for Actual effort, basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.4 is divided into three part. In the first part, we clearly see that the highest values of estimated effort for the estimation effort of the basic COCOMO, standard COCOMO, and COCOMO JA model, meanwhile the estimated effort of both of that three COCOMO are most closely of the project 1 and project 2. While the project 1 to the project 4 the values of estimation effort of the actual effort is lowest compared to the estimation effort of the basic COCOMO, standard COCOMO, and COCOMO JA. In

the second part, project 2 will slightly be increased. The project 3 the values of the estimation effort of the basic COCOMO, standard COCOMO, and COCOMO JA is slowly stability and mostly close the estimate effort , but project 2 the values of estimation effort for the actual effort falling slowly. Then, project 3 estimate effort of the actual effort will slightly be increased. In the third part, project 4 and project 5 the values of estimation effort for the actual effort, basic COCOMO, standard COCOMO, and COCOMO JA falling slowly.

4.3.3 Time Estimation

The calculated values of time estimation are rendered in Table 4.4, Table 4.5, and Table 4.6. The model with the best set parameters c and d is rendered in Equation 4.4, Equation 4.5 and Equation 4.6 for each dataset. The Figure 4.5 shows the NASA63 dataset measured time comparison between basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The Figure 4.6 Turkish dataset shows the measured time comparison between basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The Figure 4.7 shows the Kemerer dataset measured time comparison between basic COCOMO, Standard COCOMO and proposed Jaya algorithm. The result shows that the time estimated by the proposed Jaya Algorithm gives poor results compared to the Basic COCOMO.

$$E = 2.5 * (KLOC)^{0.3244} \tag{4.4}$$

$$E = 2.5 * (KLOC)^{0.3383} \tag{4.5}$$

$$E = 2.5 * (KLOC)^{0.3553} \tag{4.6}$$

4.3.3.1 NASA63 Software Project Dataset Results

Table 4.4 Result of the Time Estimation of NASA63 Dataset

Project No.	Basic COCOMO (Time)	Standard COCOMO (Time)	COCOMO JA (Time)
1	14	15	15
2	19	20	20
3	12	13	13
4	40	42	40
5	21	22	21
6	12	13	13
7	9	9	9
8	8	9	8
9	7	7	7
10	6	7	7
11	7	8	8
12	13	14	13
13	11	11	11
14	13	13	13
15	12	13	13
16	7	8	8
17	13	14	14
18	8	9	9
19	9	9	9

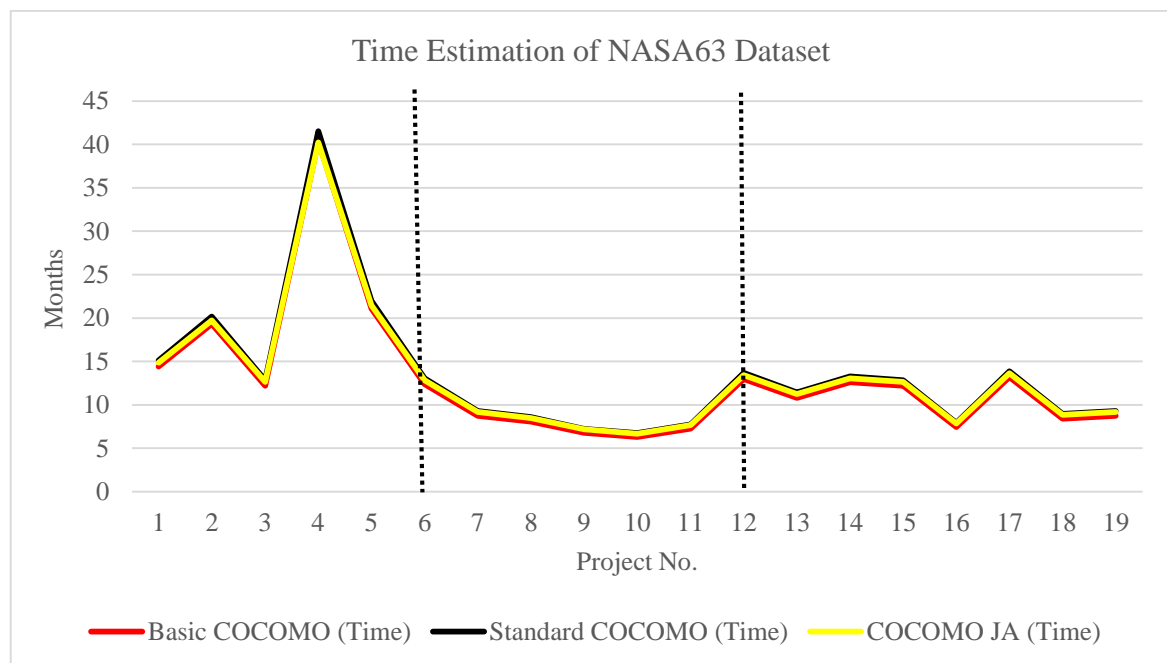


Figure 4.5 Comparison of the Time Estimation of NASA63 Dataset

In Figure 4.5, the time estimation pattern of NASA63 dataset for basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.5 is divided into three part. In the first part, the time estimation for the basic COCOMO, standard COCOMO, and COCOMO JA are very close. It will be increased the time estimation of Project 1 and project 2. At project 3 the time estimation started to fall slowly, suddenly at project 4 the time estimation growing rapidly. After that, the time of the estimated of the project 5 and project 6 will be fall slowly. In the second part, the time of the estimated of the project 7 to project 11 will be fall slowly. Then, the time of the estimated of the project 12 slightly increased. In the third part, the time of the estimate of the project 13 to project 17 will be fluctuation. After, the projects 18 and project 19 mostly same the time of estimate.

4.3.3.2 Turkish Dataset Results

Table 4.5 Result of the Time Estimation of Turkish Dataset

Project No.	Basic COCOMO (Time)	Standard COCOMO (Time)	COCOMO JA (Time)
1	23	20	18
2	12	11	10
3	4	4	4
4	4	4	4

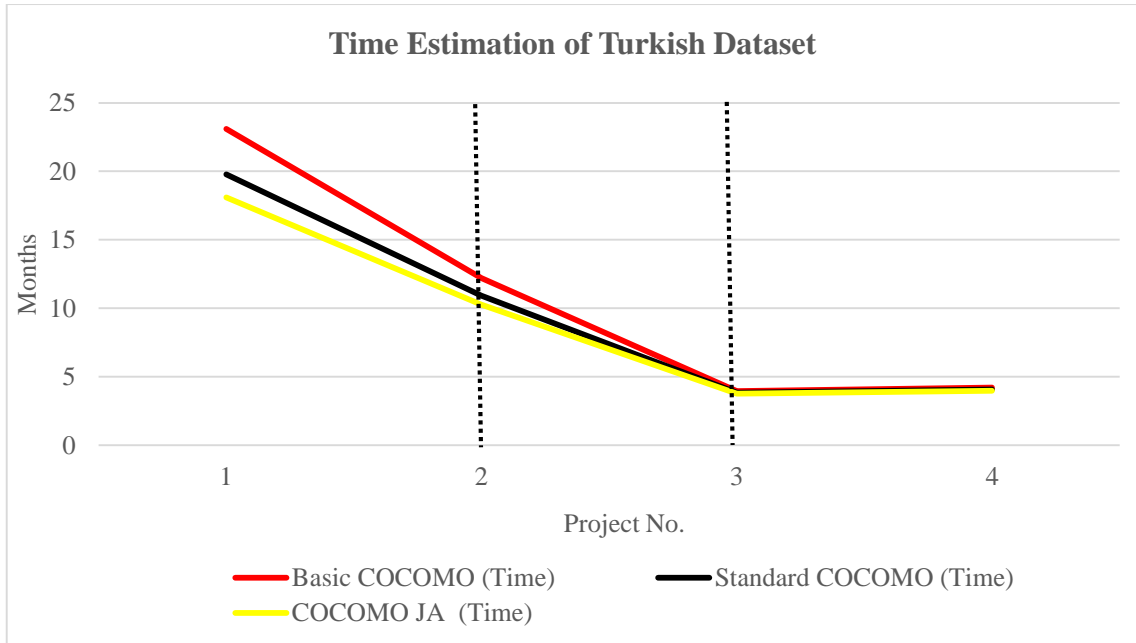


Figure 4.6 Comparison of the Time Estimation of Turkish Dataset

In Figure 4.6, the time estimation pattern of NASA63 dataset for basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.6 is divided into three part. In the first part, project 1 the time estimation for the basic COCOMO is the highest, but the lowest is COCOMO JA. At the same time, project 1 to project 2 both of this three COCOMO reduce rapidly. In the second part, the time of the estimated of the project 2 to project 3 will be fall slowly. Then, the time of the estimated of the project 12 is slightly increased. In the third part, the time of the estimate of project 3 and project 4 are almost the same.

4.3.3.3 Kemerer Dataset Results

Table 4.6 Result of the Time Estimation of Kemerer Dataset

Project No.	Basic COCOMO (Time)	Standard COCOMO (Time)	COCOMO JA (Time)
1	32	27	27
2	24	21	21
3	27	23	23
4	27	23	23
5	18	16	16

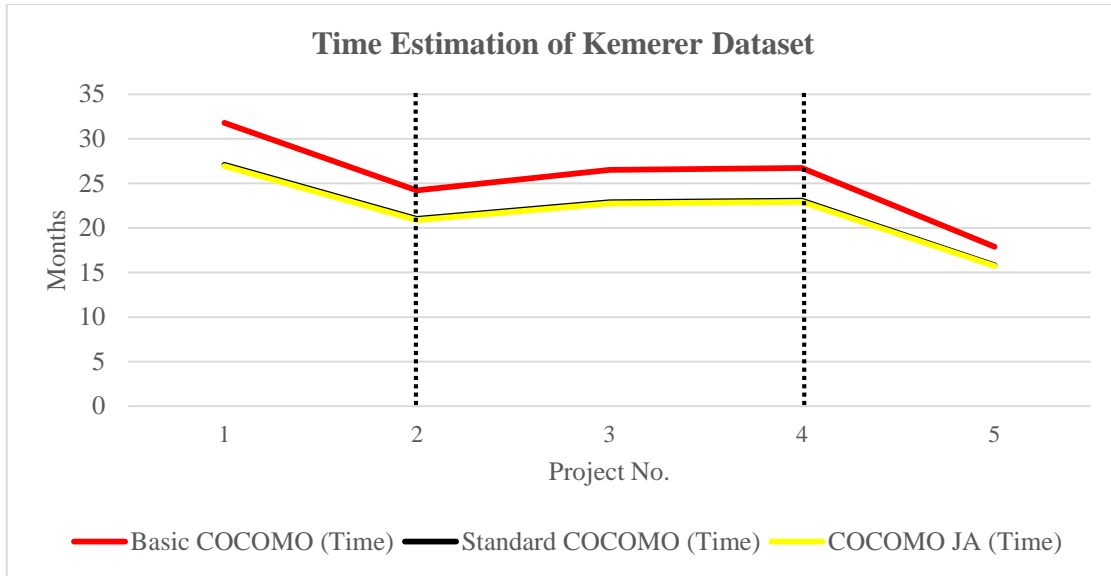


Figure 4.7 Comparison of the Time Estimation of Kemerer dataset

In Figure 4.7, the time estimation pattern of NASA63 dataset for basic COCOMO, Standard COCOMO and COCOMO JA of the project was highlighted.

The Figure 4.7 is divided into three part. The time of estimate basic COCOMO is higher than the standard COCOMO and COCOMO JA. But the standard COCOMO and COCOMO JA are the same time of the estimate. In the first part, project 1 the time estimation for the basic COCOMO will slightly decrease. In the second part, the time of the estimated of the project 2 to project 3 will be slightly increased. In the third part, the time of the estimate of project 4 and project 5 will slightly decrease.

4.4 Discussion

This section discusses the experimental results in detail. The experiment results from 4.3.2.1 until 4.3.2.3 show the comparison the effort estimation of each dataset. Conversely, the experimental results from 4.3.3.1 until 4.3.3.3 show the comparison the time estimation of each dataset which is NASA63 dataset, Turkish dataset, and Kemerer dataset.

For the purpose of this study, we believe that the dataset is sufficient. The dataset is divided into two portions; the training set is about 70% and the testing set is about 30%. This study assesses two main variables, the Millennium Line (KLOC) and the actual effort of the project size.

We developed the COCOMO JA model for effort evaluation. JA was used to evaluate the best parameters a and b of Equation 2.1 (refer to Chapter 2), and the values of a and b were selected to minimum the error between the actual effort and estimated effort. We called it MMRE. From Table 4.7, we notice that we have acquired great MMRE values.

Table 4.7 Comparison the Mean Magnitude of Relative Error (MMRE)

Dataset	Standard COCOMO	COCOMO JA
NASA63	1.2253E-6	3.5614E-9
Turkish	0.7861	0.7827
Kemerer	0.1437	0.1415

The Table 4.7 shows the comparison between standard COCOMO with COCOMO JA. The MMRE results for the standard COCOMO and COCOMO JA more nearly. It is clearly shown that the COCOMO with using the algorithm will be better compared with the random COCOMO. The COCOMO JA demonstrate a more minimize error as compared to the standard COCOMO.

The overall result shows that the effort estimated by the proposed Jaya Algorithm gives poor results compared to the actual effort. Only the instruction can be regarded as the approximation of the best solution and the avoidance of the worst solution. However the convergence speed is increased, the population multiplicity may not be efficiently sustained, thus leading to the local optimal solution.(Yu, Liang, Qu, Chen, & Wang, 2017). Another reason may be that there is no strategy to enhance the best solution for each generation, which may result in poor quality of the final solution.

4.5 Summary

In this chapter, the estimation effort, and time estimation using the basic COCOMO is organic type, standard COCOMO, and COCOMO JA has been presented. The dataset from NASA63, Turkish, and Kemerer are used as case study.

Based on the existing content of this chapter, the next chapter will summarize all the research results, conclusion, and provide some idea for future research guidance.

CHAPTER 5

CONCLUSION

5.1 Introduction

In this chapter, we will summarize the work done in the whole research process. This chapter also discuss the implementation of the objectives of the research, constraints, contributions and recommendation improving this research in the future work. The following is the realization of the objectives:

- i) To review the state-of-the-art on software cost estimation focus on COCOMO I model.
- ii) To implement the Jaya Algorithm (JA) for improving the COCOMO I Model based estimate effort and time.
- iii) To evaluate the proposed of the estimation effort and time using the Jaya algorithm (JA).

The first objective, software cost estimation technique have many method such as the function point analysis, SLIM, COCOMO and so on. The COCOMO model has three type which is basic, intermediate and detailed. This research focus on the COCOMO I model and selected the basic COCOMO. The COCOMO I model is easy to use. And then the basic COCOMO can used for fast and slightly rough computing the software costs. After this research the COCOMO model the different from other researchers. This research is using the new meta-optimization algorithm which is Jaya algorithm in this research.

The second objective, the parameter estimation employs the Jaya algorithm to find the maximum number of iteration and population size by the fitness function. Jaya

Algorithm is integrated with parameter estimation to improve the optimal finding the values of parameters a and b. Jaya Algorithm is also modified to suitable our realization.

The last objective, the Jaya algorithm has been successfully employed to undertake all the experimentations given. In the conducted evaluation, the COCOMO JA results are successfully compared to the actual effort. However, the experiment result not good compared to the actual effort.

In this research, the estimation effort is a challenging issue for the software project managers. Modify JA to evaluate the parameter of the COCOMO I model. The developed COCOMO JA model was tested using the NASA63 dataset, the Turkish Dataset, and the Kemerer dataset.

5.2 Research Constraint

Throughout this research, the constraint in this research is about the dataset. The dataset hard to find out the complete, some of the research paper just give a little the dataset. In addition, another constraint in this research is the time limited completely. Something cannot complete based on the plan and time management, such as bug fixes and test the result.

5.3 Contribution

The research contributions made in this research work can be explained from different perspectives as follows:

- i) Value parameters of a, b, c, and d for COCOMO model based on NASA63 dataset, Turkish dataset, and Kemerer dataset.
- ii) The Jaya algorithm implementation (COCOMO JA) for cost and time estimation for COCOMO model.

5.4 Future Work

For the future research of the following effort can be pursued:

- i) New data case study from the real cost estimation problem.
- ii) Tuning of Jaya algorithm with proper equation time, population size and the execution repetition.
- iii) Using the Jaya to solve other cost estimation problem beside the COCOMO model, for example like the COCOMO II.
- iv) To increase the training size.

In conclusion, this research presents a small step of applying Jaya algorithm for software cost estimation.





















REFERENCES

- Abu-srhan, A., Sleit, A., & Sharieh, A. (2017). Parameters Estimation of the COCOMO Model Using Hybrid Algorithm of Genetic Algorithm and Cuckoo Search Algorithm, (April).
- Ali, S. M. (2015). Hybrid Cuckoo Search - Genetic Algorithm (CSGA): An Approach to Solve Some Combinatorial Problems, 5(4), 123–132.
- Aljahdali, S., & Sheta, A. F. (2010). Software effort estimation by tuning COOCMO model parameters using differential evolution. *ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010*, 1–6.
<https://doi.org/10.1109/AICCSA.2010.5586985>
- Azzeh, M. (n.d.). Dataset Quality Assessment : An extension for analogy based effort estimation, 1–20.
- Banerjee, S., & Sarkar, D. (2017). Comparative Analysis of Jaya Optimization, 5(Viii), 909–922.
- Bhatia, S., Bawa, A., & Attri, V. K. (2015). A Review on Genetic algorithm to deal with Optimization of Parameters of Constructive Cost Model. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(4), 405–408.
<https://doi.org/10.17148/IJARCCE.2015.4491>
- Bingamawa, M. T. (2016). A Review of Software Cost Estimation : Tools , Methods , and Techniques A Review of Software Cost Estimation : Tools , Methods , and Techniques, (November). <https://doi.org/10.13140/RG.2.2.18980.48008>
- Chawla, R. (2016). Software Development Effort Estimation Techniques : Software Development Effort Estimation Techniques : A Review, (February), 1–16.
- Dhiman, A., & Diwaker, C. (2013). Optimization of COCOMO II Effort Estimation using Genetic Algorithm, 208–212.
- Du, D.-C., Vinh, H.-H., Trung, V.-D., Hong Quyen, N.-T., & Trung, N.-T. (2017). Efficiency of Jaya algorithm for solving the optimization-based structural damage identification problem based on a hybrid objective function. *Engineering Optimization*, 273, 1–19.
<https://doi.org/10.1080/0305215X.2017.1367392>
- Estimation, C., Parametric, M., The, S. T. B. S., The, S. T. B. D., Li, C., They, T. S., & Li, C. (2008). Draft 1.1., 1–8.
- F., A., & Aljahdali, S. (2013). Software Effort Estimation Inspired by COCOMO and FP

- Models: A Fuzzy Logic Approach. *International Journal of Advanced Computer Science and Applications*, 4(11). <https://doi.org/10.14569/IJACSA.2013.041127>
- Galinina, A., Burceva, O., & Parshutin, S. (2012). The Optimization of COCOMO Model Coefficients Using Genetic Algorithms. *Information Technology and Management Science*, 15(1), 45–51. <https://doi.org/10.2478/v10313-012-0006-7>
- Gálvez, Iglesias, & C. et al. (2014). Cuckoo search with Lévy flights for weighted bayesian energy functional optimization in global-support curve data fitting. *Scientific World Journal*, 2014. <https://doi.org/10.1155/2014/138760>
- Gharehchopogh, F. S., Maleki, I., & Reza Khaze, S. (2014). A novel particle swarm optimization approach for software effort estimation. *International Journal of Academic Research*, 6(2), 69–76. <https://doi.org/10.7813/2075-4124.2014/6-2/A.12>
- Ghatasheh, N., Faris, H., Aljarah, I., & Al-Sayyed, R. M. H. (2015). Optimizing Software Effort Estimation Models Using Firefly Algorithm. *Journal of Software Engineering and Applications*, 8(3), 133–142. <https://doi.org/10.4236/jsea.2015.83014>
- Jayakumar et al. (2012). International Journal of Advanced Research in Computer Science and Software Engineering. *International Journal*, 2(9), 62–70. Retrieved from http://www.ijarcsse.com/docs/papers/July2012/Volume_2_issue_7/V2I700161.pdf%5Cnh http://www.ijarcsse.com/docs/papers/9_September2012/Volume_2_issue_9/V2I900140.pdf
- Kemerer, C. F. (1987). An empirical validation of software cost estimation models. *Commun. ACM*, 30(5), 416–429. <https://doi.org/10.1145/22899.22906>
- Kumar et al. (2012). Modified Cocomo Model for Maintenance Cost Estimation of Real Time System Software. *Euroasiapub.Org*, 1(1), 1–7.
- Kumari, S., & Pushkar, S. (2013). Performance Analysis of the Software Cost Estimation Methods : A Review. *International Journal of Advanced Research in Computer Science and Software Engineering Research*, 3(7), 229–238.
- Manalif, E., Capretz, L. F., & Ho, D. (2013). Software Project Risk Assessment and Effort Contingency Model based on COCOMO Cost Factors. *Journal of Computations & Modelling*, 3(1), 113–132.
- Menzies, Tim, Z. C. (n.d.). COCOMO NASA.
- Nisar et al. (2009). Software Development Effort Estimation Using Fuzzy Logic, (February). Retrieved from <http://www.scialert.net/pdfs/itj/2009/347-353.pdf;%5Cnh> <http://www.doaj.org/doaj?func=openurl&genre=article&issn=18125638&date=2009&volume=8&issue=3&spage=347>

- Pandey, H. M. (2016). Jaya a novel optimization algorithm: What, how and why? *Proceedings of the 2016 6th International Conference - Cloud System and Big Data Engineering, Confluence 2016*, 728–730. <https://doi.org/10.1109/CONFLUENCE.2016.7508215>
- Sachan, R. K., Nigam, A., Singh, A., Singh, S., Choudhary, M., Tiwari, A., & Kushwaha, D. S. (2016). Optimizing Basic COCOMO Model Using Simplified Genetic Algorithm. *Procedia Computer Science*. <https://doi.org/10.1016/j.procs.2016.06.107>
- Sheta, A. F. (2006). Estimation of the COCOMO model parameters using genetic algorithms for NASA software projects. *Journal of Computer Science*, 2(2), 118–123. <https://doi.org/10.3844/jcssp.2006.118.123>
- Sheta, A., Rine, D., & Ayesh, A. (2008). Development of software effort and schedule estimation models using Soft Computing Techniques. *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, 1283–1289. <https://doi.org/10.1109/CEC.2008.4630961>
- Shukla, R., & Singh, D. (2017). Selection of parameters for advanced machining processes using firefly algorithm. *Engineering Science and Technology, an International Journal*, 20(1), 212–221. <https://doi.org/10.1016/j.jestch.2016.06.001>
- Structures, D. (n.d.). Chapter 2 Literature Review on Vibration, 11–44.
- Suri, P. K., & Ranjan, P. (2012). Comparative Analysis of Software Effort Estimation Techniques. *International Journal of Computer Applications*, 48(21), 975–8887.
- Venkata Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, 7(1), 19–34. <https://doi.org/10.5267/j.ijiec.2015.8.004>
- Yang, X. S. (2009). Firefly algorithms for multimodal optimization. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5792 LNCS, 169–178. https://doi.org/10.1007/978-3-642-04944-6_14
- Yang, X. S., & Deb, S. (2009). Cuckoo search via Levy flights. *2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings*, 210–214. <https://doi.org/10.1109/NABIC.2009.5393690>
- Yu, K., Liang, J. J., Qu, B. Y., Chen, X., & Wang, H. (2017). Parameters identification of photovoltaic models using an improved JAYA optimization algorithm. *Energy Conversion and Management*, 150(August), 742–753. <https://doi.org/10.1016/j.enconman.2017.08.063>

APPENDIX A GANTT CHART

	 Task	Task Name	Duration	Start	Finish	February	March	April	May	June	July	August	September	October	November	De
1		PARAMETER ESTIMATION OF COCOMO MODEL USING THE JAYA ALGORITHM														
2		Planning	7 days	27/2/2018	7/3/2018											
3		Identify the introduction, problem statement	4 days	27/2/2018	2/3/2018											
4		Identify the objective, scopes, significance and thesis organization	4 days	3/3/2018	7/3/2018											
5		Concepts and literature review	21 days	9/3/2018	6/4/2018											
6		Review related works about the research title	5 days	9/3/2018	15/3/2018											
7		Study and analyse the software cost estimation	5 days	14/3/2018	20/3/2018											
8		Review the existing COCOMO model Based on the optimization Algorithms	5 days	16/3/2018	22/3/2018											
9		Study and analyse the existing COCOMO model based on the optimization Algorithms	5 days	23/3/2018	29/3/2018											
10		Comparison the software cost estimation	3 days	31/3/2018	3/4/2018											

	i	Task	Task Name	Duration	Start	Finish	February	March	April	May	June	July	August	September	October	November	Dec
10		✦	Comparison the software cost estimation	3 days	31/3/2018	3/4/2018			■								
11		✦	Comparison the existing COCOMO model based on the optimization algorithms	4 days	3/4/2018	6/4/2018			■								
12		✦	Methodology	19 days	6/4/2018	2/5/2018			■	■							
13		✦	Design the research activity	2 days	6/4/2018	9/4/2018			■								
14		✦	Design the overview of the COCOMO model using the Jaya Algorithm	5 days	10/4/2018	16/4/2018			■								
15		✦	Implementation of proposed algorithm	7 days	17/3/2018	26/3/2018		■									
16		✦	Presentation PSM1	2 days	14/5/2018	15/5/2018				■							
17		✦	Study on PSM 2 implementation	67 days	10/6/2018	10/9/2018					■	■	■	■	■	■	■
18		✦	Data collection phase	6 days	1/10/2018	8/10/2018									■		
19		✦	Finalize the dataset	6 days	1/10/2018	8/10/2018									■		
20		✦	Development and implementation Phase	30 days	8/10/2018	16/11/2018									■	■	■
21		✦	Development and implement Jaya algorithm code	30 days	9/10/2018	19/11/2018									■	■	■
22		✦	Evaluation	43 days	1/10/2018	28/11/2018									■	■	■
23		✦	Carry out the experiment result	39 days	1/10/2018	22/11/2018									■	■	■
24		✦	Discuss the training and testing result	5 days	22/11/2018	28/11/2018											■
25		✦	Documentation	10 days	29/11/2018	12/12/2018											■
26		✦	Report writing	10 days	29/11/2018	12/12/2018											■

APPENDIX B
NASA63 SOFTWARE PROJECT DATASETS

Project No.	KLOC	Actual Effort
16	6.1	40.0
17	3.6	9.0
18	320.0	11400.0
19	1150.0	6600.0
20	299.0	6400.0
21	252.0	2455.0
22	118.0	724.0
23	77.0	539.0
24	90.0	453.0
25	38.0	523.0
26	48.0	387.0
27	9.4	88.0
28	13.0	98.0
29	2.1	7.3
30	2.0	5.9
31	62.0	1063.0
32	390.0	702.0
33	42.0	605.0
34	23.0	230.0
35	13.0	82.0
36	15.0	55.0
37	60.0	47.0
38	15.0	12.0
39	6.2	8.0
40	3.0	8.0
41	5.3	6.0
42	45.5	45.0
43	28.6	83.0
44	30.6	87.0
45	35.0	106.0
46	73.0	126.0
47	23.0	36.0
48	464.0	1272.0
49	91.0	156.0
50	24.0	176.0
51	10.0	122.0
52	8.2	41.0
53	5.3	14.0

54	4.4	20.0
55	6.3	18.0
56	27.0	958.0
57	17.0	237.0
58	25.0	130.0
59	23.0	70.0
60	6.7	57.0
61	28.0	50.0
62	9.1	38.0
63	10.0	15.0
