VIRUS BASED OPTIMIZATION ALGORITHMS

Submitted in Partial Fulfillment of the Requirement

For the Award of the Degree of

Master of Technology

In

Computer Science & Engineering

Submitted By

SWATI JAIN
University Roll No. 2K12/CSE/31

Under the Esteemed Guidance of

Dr. (Mr.) Kapil Sharma

Associate Prof., Computer Engineering Department, DTU, Delhi

2012-2015
DELHI TECHNOLOGICAL UNIVERSITY
DELHI – 110042

Department of Computer Engineering
Delhi Technological University
Delhi-110042

CERTIFICATE

This is to certify that the dissertation titled "**Virus Based Optimization Algorithms**" is a bonafide record of work done at **Delhi Technological University** by **Swati Jain, Roll No. 2K12/CSE/31** for partial fulfilment of the requirements for the degree of Master of Technology in Computer Science & Engineering. This project was carried out under my supervision and has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma to the best of my knowledge and belief.

(Dr. (Mr.) Kapil Sharma)
Associate Professor & Project Guide
Department of Computer Engineering
Delhi Technological University

Date: _____________

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to all the people who have supported and encouraged me during the course of this project without which, this work could not have been accomplished.

First of all, I am very grateful to my project supervisor Dr. (Mr.) Kapil Sharma for providing the opportunity of carrying out this project under his guidance. I am deeply indebted to him for the support, advice and encouragement he provided without which the project could not have been a success. I am also grateful to Dr. (Mr.) OP Verma, HOD, Computer Science, DTU for his immense support. I am also thankful to my parents and spouse for being there for me at all times. Last but not the least; I am grateful to Delhi Technological University for providing the right resources and environment for this work to be carried out.

Swati Jain
University Roll no: 2K12/CSE/31
M.Tech (Computer Science & Engineering)
Department of Computer Engineering
Delhi Technological University
Delhi – 110042

ABSTRACT

Software cost estimation is one of the most important activities for software project management and all the companies, today, are focusing on incorporating new techniques to minimize any risk. Among these, Constructive Cost Model (COCOMO) is the most widely used and accepted model due to its applicability in diverse stages of software Engineering. Traditional COCOMO, however, often lacks the precision and accuracy as the estimations are largely based on the parameters such as size of the project, cost drivers, coefficients etc and a small miscalculation can lead to vast difference in the estimated effort. Hence, scientists have been focusing on optimizing the COCOMO model using various meta-heuristic algorithms. In this paper, a novel meta-heuristic algorithm, Virus Optimization Algorithm (VOA), has been used to optimize the COCOMO model in order to minimize the error in the calculations and aid in proper budgeting for software projects. The performance of the proposed algorithm was investigated by comparing it to three other well-known software cost estimation models. The results showed that proposed work outperformed other algorithms in minimizing the Mean Magnitude of Relative Error while optimizing the COCOMO II model.

Index Terms - VOA algorithm, Software Cost Estimation, COCOMO

LIST OF ABBERIVATIONS

ACRONYM	DEFINATION
VOA	Virus Optimization Algorithm
COCOMO	Constructive Cost Model
PSO	Particle Swarm Optimization Algorithm
GA	Genetic Algorithm
BA	Bat Algorithm
BFOA	Bacteria Foraging Optimization Algorithm
NASA	National Aeronautic and Space Administration
KLOC	Thousands of line of code
MMRE	Mean Magnitude of relative error
NV	New Virus
SV	Strong Virus
CV	Common Virus
EAF	Effort Adjustment Factor
MRE	Magnitude of Relative Error

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

1.1 Software Cost Estimation

During the starting phase of requirement analysis for any project or company, every project manager is supposed to determine the amount of effort required for development. This is a very critical phase of any project, as companies can lose business worth millions of dollars if they over estimate or underestimate the effort.

Software cost estimation is, therefore, the most important activity for software project management and all the companies, today, are focusing on incorporating new techniques to minimize any risk. Among these, Constructive Cost Model (COCOMO) is the most widely used and accepted model due to its applicability in diverse stages of software Engineering.

Traditional COCOMO, however, often lacks the precision and accuracy as the estimations are largely based on the parameters such as size of the project, cost drivers, coefficients etc and a small miscalculation can lead to vast difference in the estimated effort. Hence, scientists have been focusing on optimizing the COCOMO model using various meta-heuristic algorithms.

Meta-heuristic algorithms consist of a set of computer –based instructions which allow computers to simulate the real world scenarios. Extensive research in this field has lead to the development of various new optimization algorithms, some of which are inspired by nature.

In this paper, a novel meta-heuristic algorithm, Virus Optimization Algorithm (VOA), has been used to optimize the COCOMO model. VOA derives its behavior from the non-living microorganism, Virus. Viruses are large in number and have diverse structures. They can easily attach to a human living cell and use the cell's protein to replicate, change its genetic material inside the cell, and hence, are almost unstoppable. This capability of a virus can be utilized to optimize complex real world problems. In this paper, it has been used to optimize the software cost estimation model, COCOMO II in order to minimize the error in the calculations and aid in proper budgeting for software projects.

1.2 Research Objective

With the view explained in the previous section, the objective of our research work can be identified as:

- To optimize the COCOMO II model using the Virus Optimization Algorithm.
- To evaluate the performance of the proposed algorithm against other well-known software cost estimation algorithms.

1.3 Thesis Organization

We start this dissertation with Introduction in Chapter 1. A detailed description of background is presented in chapter 2 which includes reviews on Optimization algorithms; resent developments in the field of optimization, types and behavior of viruses, literature review of Virus Optimization Algorithm and various works inspired by the behavior of viruses. Chapter 3 describes the mechanics of Virus Optimization algorithm and details out the COCOMO model which are used in our proposed work. Chapter 4 explains in detail about our proposed work that is Optimizing COCOMO II model using Virus Optimization Algorithm. We evaluate the performance of the proposed technique with the existing algorithms in chapter 5. We conclude about the work done and future work in chapter 6.

In this chapter, a brief review on Optimization, evolutionary algorithms, bio-inspired computations, behavior of viruses and virus inspired algorithms has been given.

2.1 Optimization

Optimization, in computer systems, is the process of creating a fully functional and effective computer system. This can be achieved by optimizing the growth and profitability of an optimization problem simultaneously, and thereby, getting the values of the parameters, which will give the optimal value of the function which is to be optimized. Optimization algorithms are classified widely into the following categories: Approximate Algorithms, Exact Algorithms, Constructive Heuristics, Local Search Methods and Meta-Heuristic Algorithms

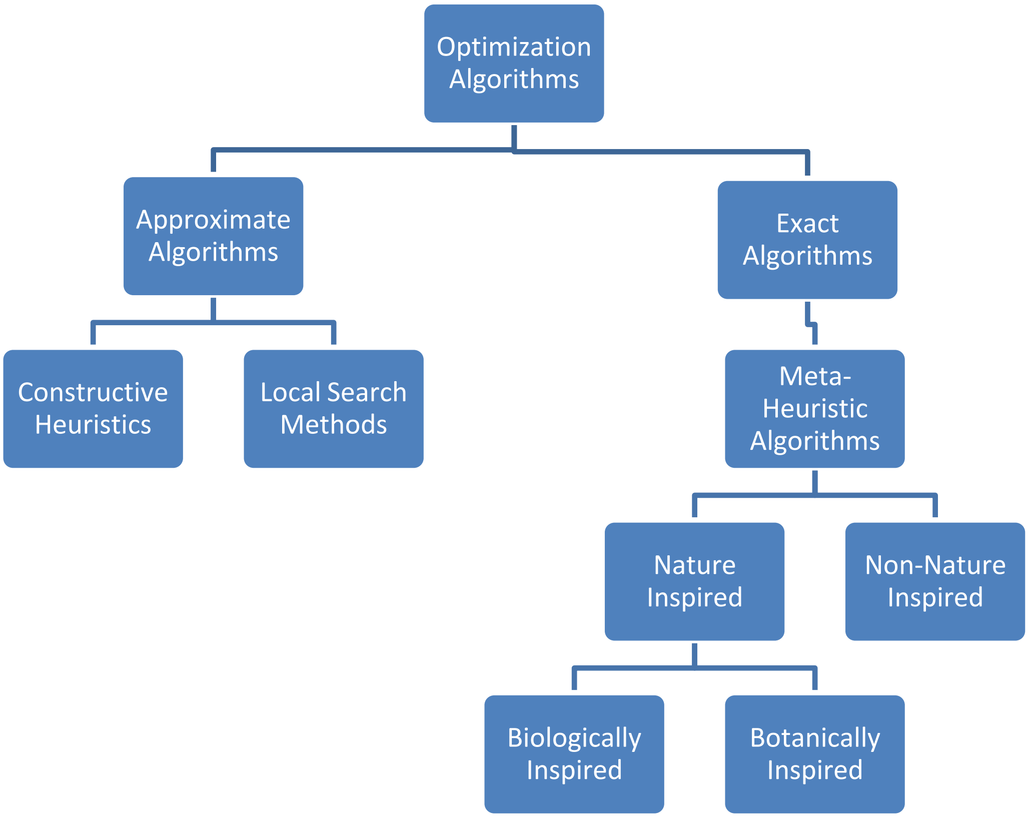


Figure 1 : Optimization Algorithms

Over the last few years, Meta-heuristic (discover solution by trial and error) approximation algorithms are widely used to solve many continuous and combinatorial optimization problems. It often finds good solution with less computation effort than exhaustive, iterative and simple heuristic methods. Some of the meta-heuristic algorithms are particle swam optimization (PSO), genetic algorithm (GA), cuckoo search (CS) , BAT algorithm etc. These algorithms are problem independent thus suits many optimization problems.

A wide number of new optimization algorithms have been proposed as well as advancements have been made in the existing optimization algorithms. Some of the recent works in the area of meta-heuristics are:

- Somayeh Allahyari, Majid Salari and Daniele Vigo [1], in May 2015, proposed a hybrid meta-heuristic algorithm for the multi-depot capacitated vehicle routing problem, in which the demand of customers could be satisfied by visiting them on the tour or by covering it. (Allahyari, Salari, & Vigo, 2015)
- Maziar Yazdani and Fariborz Jolai [2], in June 2015, proposed a new nature inspired metaheuristic algorithm-Lion Optimization algorithm which is based on the special lifestyle of lions and their cooperation characteristics. (Yazdani & Jolai, 2015)
- Tirtharaj Dash and Prabhat K. Sahu [3] in March 2015, proposed a gradient based gravitational search algorithm, which uses analytical gradients for a fast minimization to the next local minimum. (Dash & Sahu, 2015)
- Lingyun Zhou, Lixin Ding, Xiaoli Qiang, Yihan Luo [4] (Zhou, Ding, Qiang, & Luo, 2015), in July 2015, proposed an improved and a multi-population discrete firefly algorithm for the travelling salesman problem.

2.2 Nature Inspired Algorithms

An evolutionary algorithm is a subset of evolutionary computation, a generic population-based meta-heuristic optimization algorithm. An evolutionary algorithm uses mechanisms inspired by nature, such as reproduction, mutation, recombination, and selection. It starts with a randomly initialized population, which then evolves across several generations. In every generation, individuals which are fit are selected to become the parent individuals. The parents then cross-over with each other to generate new individuals (children). The child individuals are randomly selected, they might undergo mutations, after which, optimal individuals are selected for the next generation according to a fitness function. This procedure is repeated till a stopping criterion is met [5] (Wong, 2015).

2.2.1 Genetic Algorithm

Genetic algorithms are adaptive heuristic search algorithms based on the Darwin theory of natural selection. They are introduced by John Holland [6] (Holland, 1975). GAs searches the space of all possible solutions using a population of individuals which is considered as potential solutions of the problem under consideration. These solutions are computed based on their fitness. The solutions that best fit to the objective criterion survive in the upcoming generations and produce “offspring” which are variations of their parents. GAs has been successfully used in a wide variety of difficult numerical optimization problems. They have been successfully used to solve system identification, signal processing and path searching problems. [7] (Singh & Misra, 2012)

2.2.2 Particle Swarm Optimization (PSO) algorithm

This simulates the social behavior of natural creatures such as bird flocking and fish schooling to discover a place with adequate food. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*.

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its p*best* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations.

2.2.3 Bacterial Foraging Algorithm (BFOA)

BFOA is inspired by the social foraging behavior of Escherichia coli. BFOA has drawn the attention of researchers because of its good efficiency in solving the complex real-world optimization problems that are available in several application domains. The underlying biology behind the foraging strategy of E.coli is emulated in an extraordinary manner and used as a simple optimization algorithm. The Bacteria Foraging is bio inspired algorithm which estimates the cost function after each iterative step of the program as the program execution proceeds and leads to comparatively better fitness. [8] (Sharma, 2012)

2.2.4 BAT Algorithm

Bat algorithm is a meta-heuristic, nature inspired, swarm based algorithm. It’s an optimization method based on the echolocation behavior of bats. Micro bats echolocation capability helps them to detect preys, distinguish different kinds of insects, avoid obstacles, and locate their prey in the dark.[9] (Gupta & Sharma, 2015)

2.3 Virus: Types and Behavior

Viruses are non-cellular organisms, which made up of genetic material and protein that can invade living cells. These micro organisms belong to the family of viridae and Genus of virus. There are an estimated 10^{31} viruses on Earth, most of which are phages that infect bacteria.[10] (Breitbart & Rohwer, 2005)

2.3.1 Life Properties

- Viruses are considered both a living and non living things. It replicates only inside the living cells of other organisms. Viruses can infect all types of life forms, from animals and plants to microorganisms, including bacteria and archaea. [11] (Koonin, Senkevich, & Dolja, 2006)
- They have been described as "organisms at the edge of life",[12] (Rybicki, 1990) since they resemble organisms in that they possess genes and evolve by natural selection,[13] (EC., 2007)and reproduce by creating multiple copies of themselves through self-assembly.
- Although they have genes, they do not have a cellular structure, which is often seen as the basic unit of life. Viruses do not have their own metabolism, and require a host cell to make new products. They therefore cannot naturally reproduce outside a host cell. [14] (Wimmer, Mueller, Tumpey, & Taubenberger, 2009)
- Accepted forms of life use cell division to reproduce, whereas viruses spontaneously assemble within cells. They differ from autonomous growth of crystals as they inherit genetic mutations while being subject to natural selection. Virus self-assembly within host cells has implications for the study of the origin of life, as it lends further credence to the hypothesis that life could have started as self-assembling organic molecules.[11]

2.3.2 Structure of Viruses

Viruses are very small and they measured in nanometers. They can only be seen with an electron microscope. They are composed of a core of DNA or RNA surrounded by a protein coat they can only reproduce by infecting living cells. Their size ranges from 20 nanometers to 250 nanometers.

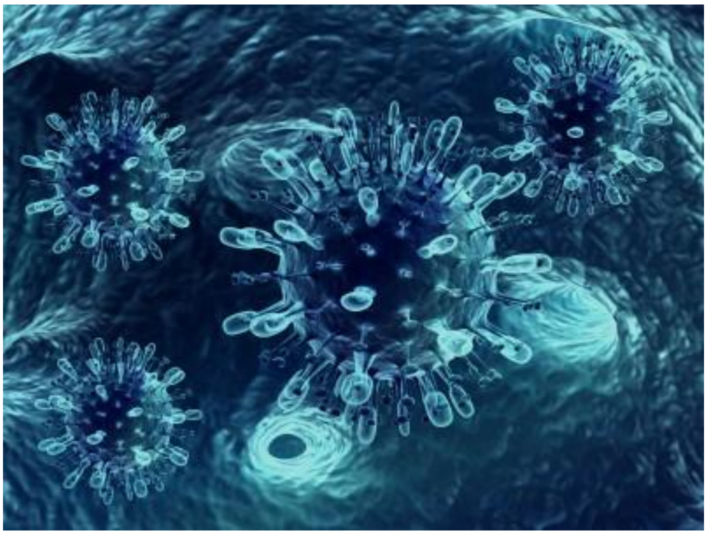


Figure 2 : Structure of Viruses

2.3.3 Shapes of Viruses

- They are helical in shape like the Ebola virus.

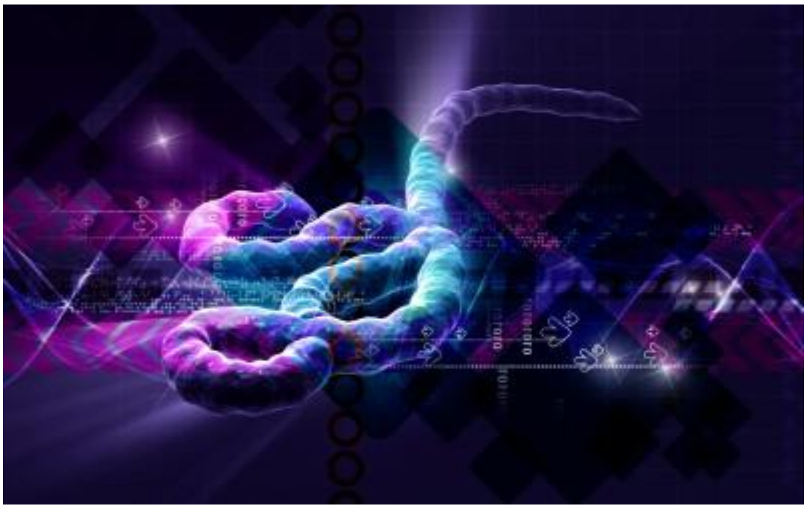


Figure 3 : Structure of Ebola Virus

- They are polyhedral shapes like the influenza virus

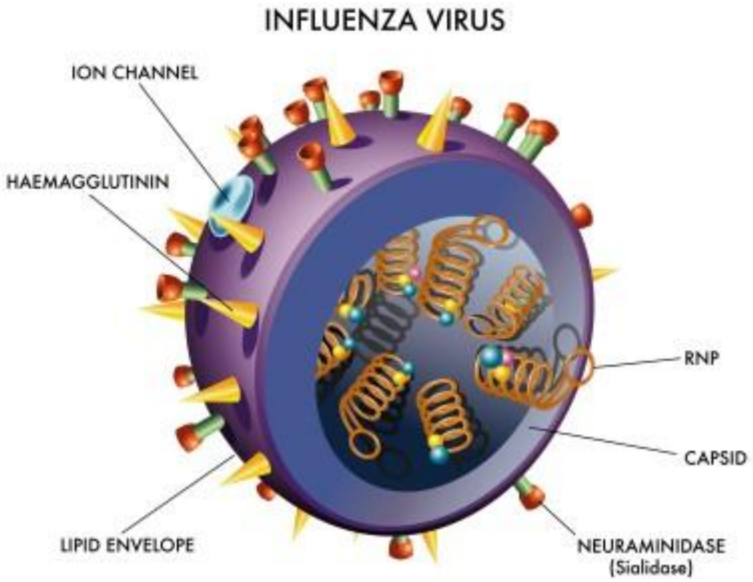


Figure 4 : Structure of Influenza Virus

- They are complex in shapes like bacteriophages.

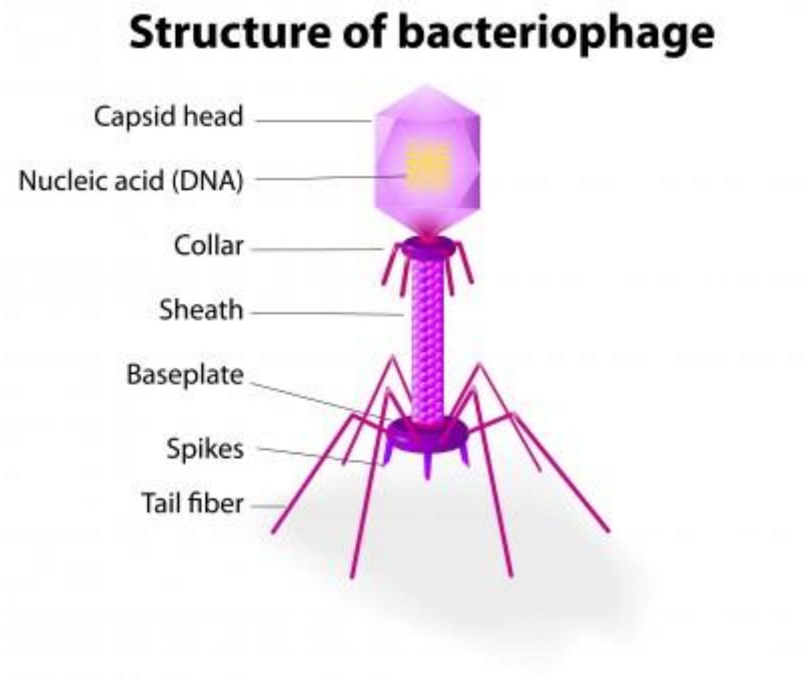


Figure 5 : Structure of Bacteriphages

- Structure of HIV Virus

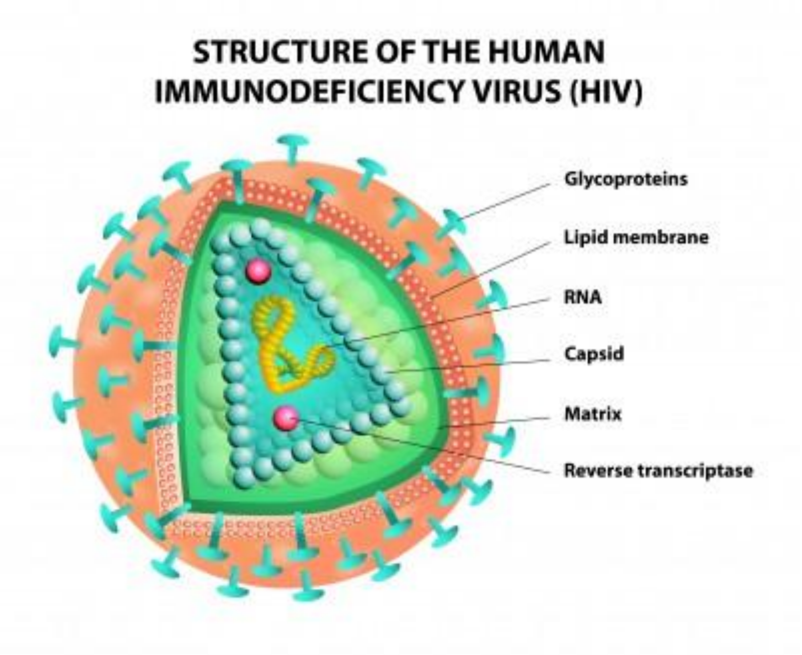


Figure 6 : Structure of HIV Virus

2.3.4 Advantages and Disadvantages of Viruses

Advantages of Viruses

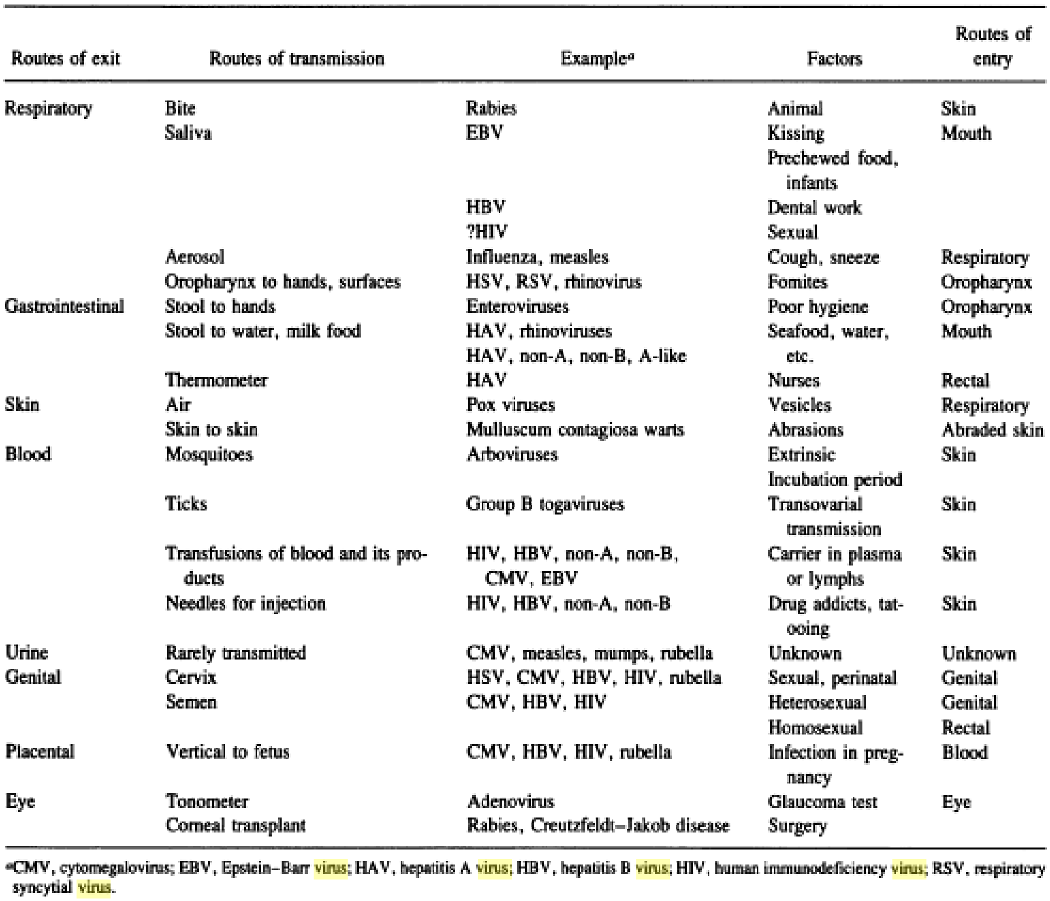
They are use full in delivering genes to target cells and play a vital role in and gene therapy researches.

Disadvantages of Viruses

There are many pathogenic viruses, which causes harm for human beings, plants and animals. In human beings the diseases caused by viruses are: HIV, influenza, herpes, hepatitis small pox, cowpox, etc. The diseases caused by viruses in plants are tobacco mosaic viruses, etc.

The major routes of transmission of viral infections in humans are listed in Table 1.[15] (Evans, 1989)

Table 2-1 : Transmission of Viral Infections



Routes of exit	Routes of transmission	Example[a]	Factors	Routes of entry
Respiratory	Bite	Rabies	Animal	Skin
	Saliva	EBV	Kissing	Mouth
			Prechewed food, infants	
		HBV	Dental work	
		?HIV	Sexual	
	Aerosol	Influenza, measles	Cough, sneeze	Respiratory
	Oropharynx to hands, surfaces	HSV, RSV, rhinovirus	Fomites	Oropharynx
Gastrointestinal	Stool to hands	Enteroviruses	Poor hygiene	Oropharynx
	Stool to water, milk food	HAV, rhinoviruses	Seafood, water, etc.	Mouth
		HAV, non-A, non-B, A-like		
	Thermometer	HAV	Nurses	Rectal
Skin	Air	Pox viruses	Vesicles	Respiratory
	Skin to skin	Mulluscum contagiosa warts	Abrasions	Abraded skin
Blood	Mosquitoes	Arboviruses	Extrinsic	Skin
			Incubation period	
	Ticks	Group B togaviruses	Transovarial transmission	Skin
	Transfusions of blood and its products	HIV, HBV, non-A, non-B, CMV, EBV	Carrier in plasma or lymphs	Skin
	Needles for injection	HIV, HBV, non-A, non-B	Drug addicts, tattooing	Skin
Urine	Rarely transmitted	CMV, measles, mumps, rubella	Unknown	Unknown
Genital	Cervix	HSV, CMV, HBV, HIV, rubella	Sexual, perinatal	Genital
	Semen	CMV, HBV, HIV	Heterosexual	Genital
			Homosexual	Rectal
Placental	Vertical to fetus	CMV, HBV, HIV, rubella	Infection in pregnancy	Blood
Eye	Tonometer	Adenovirus	Glaucoma test	Eye
	Corneal transplant	Rabies, Creutzfeldt–Jakob disease	Surgery	

[a] CMV, cytomegalovirus; EBV, Epstein–Barr virus; HAV, hepatitis A virus; HBV, hepatitis B virus; HIV, human immunodeficiency virus; RSV, respiratory syncytial virus.

2.4 Optimization Algorithms inspired by the Virus behavior

2.4.1 A susceptible-infected Removal (SIR) Epidemic Model

P.K.Das and S.S. DE [16] (Das & DE, 2000), in 2000, applied the SIR mathematical model for the study of diseases like Cholera and non-choleric diarrhea in Greater Calcutta.

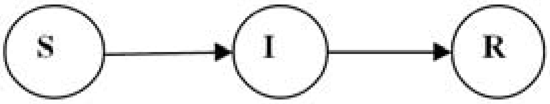


Figure 7 : Basic SIR Model of disease transmission

2.4.2 Swine influenza inspired optimization algorithm and its application to multimodal function optimization and noise removal

Shyam S. Pattnaik, Devidas G. Jadhav, Swapna Devi and Radha Kanta Ratho [17] (Pattnaik, Jadhav, Devi, & Ratho, 2012), in 2012, proposed a search algorithm, Swine Influenza Inspired Optimization (SIIO), to find the optimal solution. This is based on the SIR (susceptible - infectious-recovered) virus spread model of Swine Influenza to develop the new evolutionary algorithm named as SIIO. SIR model is used to frame optimization algorithm following the spread and control phenomenon of the swine flu virus in the human population. The fitness based classes viz. susceptible (S), infectious (I) and recovered (R) of the individuals are made and treatment is used for the affected individuals by imitating the health information from the best fitness individual.

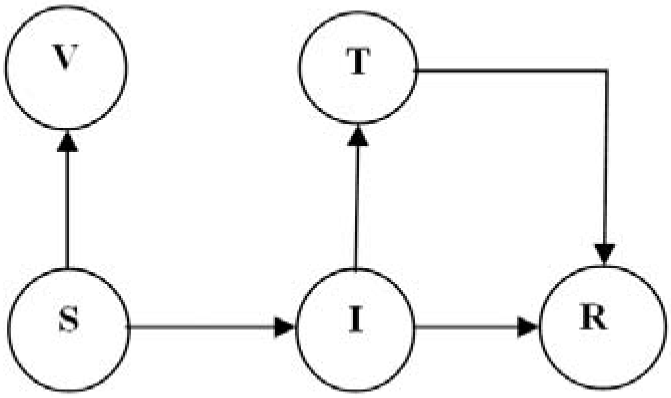


Figure 8: Extended Susceptible-Infectious-Recovered Model of disease transmission

2.4.3 Dynamic Multi-objective Optimization Algorithm Based On GEP and Virus Evolution

Weihong Wang, Yanye Du, Qu Li and Zhaolin Fang [18] (Wang, Du, Li, & Fang, 2012), in 2011, proposed a new dynamic multi-objective algorithm based on Gene Expression Programming and Virus Evolution. Many systems need to consider dynamic scheduling problems, and these constraints are called dynamic constraints. Mathematical models abstracted from problems with multiple objectives and related with time factors are Dynamic Multi-objective Optimization (referred to as DMO). As a major component of the biological immune system, virus system has many information processing mechanisms and functional features, and it has a great significance for further improvements in genetic algorithm, gene expression programming, and so on.

2.4.4 Virus Optimization Algorithm

Inspired by the behavior of viruses, Y.C. Liang and J.R.C. Juarez [19] (Liang & Juarez, 2015) proposed a novel meta-heuristic algorithm for continuous optimization problems, Virus Optimization Algorithm.
The algorithm simulates the behavior of a virus when it attacks the host cell.

2.4.5 Multilevel Image Thresholding Using Relative Entropy and Virus Optimization Algorithm

Y.C. Liang and J.R.C. Juarez[20] (Liang & Juarez, 2012), in 2012, proposed a new approach based on the Kullback-Leibler information distance, also known as Relative Entropy. The approach minimizes a mathematical model, which will determine the number of image thresholds automatically. The optimization of the mathematical model is achieved by using Virus Optimization Algorithm (VOA).

In this chapter, we have studied in detail about the Virus Optimization algorithm, COCOMO II and have used it to evaluate the Effort Estimation Problem (using NASA 63 dataset).

3.1 Virus Optimization Algorithm (VOA)

Virus Optimization Algorithm was proposed by Y.C. Liang and J.R.C. Juarez in November 2014. This algorithm is a continuous population-based optimization algorithm, which is derived from the ways in which a virus attacks a human cell.

A virus is a non-living microorganism, which infects or attaches itself to the human cell, also known as the host cell. Virus, then, exploits the cell metabolism and alters the production of protein inside the cell, leading to the creation of more viruses eventually leading to the death of the host cell. The cell death starts from the outer membrane of the cell, extending towards the nucleololus.

Figure 9 illustrates the host cell (in yellow), black dots represent the virus population, and the white region represents the cell nucleolus.

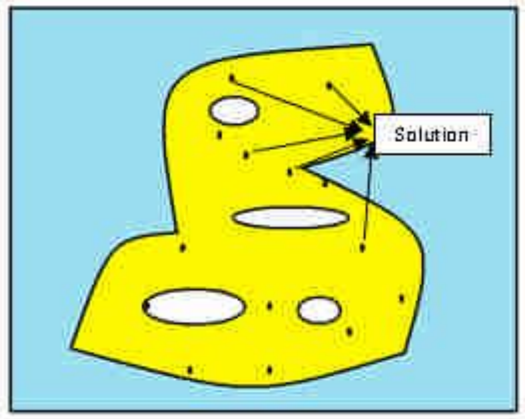


Figure 9 : Initial Virus population inside the host cell

Viruses replicate inside the host cell at different rates. The viruses with strong DNA/RNA structure, also known as strong viruses, grow at a rate larger than the replication rate of the lesser strong viruses, also known as common viruses .

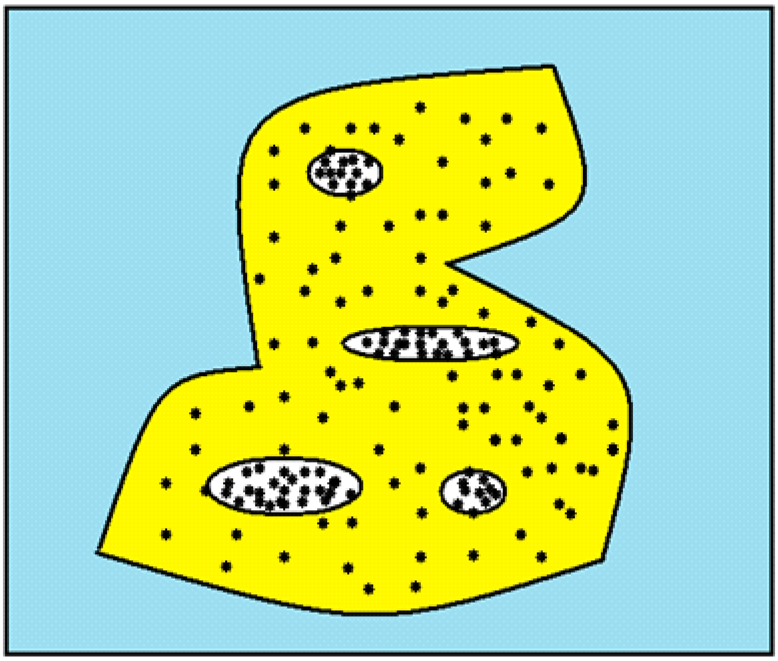


Figure 10 : Viruses generated after the replication process

Over time, immunity is developed within the humans against the viruses with the activation of B-lymphocytes (or white blood cells). B-Lymphocytes then trigger the development of antibodies which limits the spread of infection by killing certain types of viruses. This mechanism is known as antivirus mechanism.

Analogy

1. The host cell is considered as the solution space
2. The global optimum of the problem will be located inside the host cell (represented by white region in Figure 9).
3. Each virus location represents one complete solution.
4. The replication of strong viruses represents the exploitation and the replication of common viruses represents the exploration. Strong and common viruses are classified by evaluating the objective function.

5. The immune system of an organism, or the antivirus mechanism, is used to kill the common viruses so that a fixed population size can be maintained after each replication.

Procedure

1. **INITIALIZATION** of each virus within the population

   The initial virus population is randomly generated for each dimension d

   $$X_{ij} = x_{min\,j} + rand(0,1)\,(x_{max\,j} - x_{min\,j}) - (1)$$

   Where i=1, 2...N, j=1, 2….d, $x_{min\ j\ and}$ $x_{max\ j\ are}$ lower and upper boundaries for dimension j respectively.

2. **REPLICATION**

   Classification: Evaluating the fitness function of each virus. The viruses are then sorted on the basis of their fitness functions. The top 30% viruses of the total population are then classified as strong viruses. The remaining population of viruses is classified as common viruses.

   Exploitation: Strong viruses replicate at a higher rate as compared to the common viruses and generate new viruses closer to the global optimum.

   $$NV_{ij}=SV_{ij} +- rand()/intensity * SV_{ij} -(2)$$

   Where NV=New Virus, SV = Strong Virus, i and j represent the ith member in the population on the jth dimension, intensity is used to reduce the random perturbation while creating new members from the strong viruses. Initially, intensity is set to one, and is increased when the objective function of the new population does not improve as compared to the previous generation.

   Exploration: Common viruses replicate at a slower rate as compared to strong viruses and generate new solutions away from the global optimum.

   $$NV_{ij}=CV_{ij} +- rand() * CV_{ij} -(3)$$

   Where NV=New Virus, CV = Common Virus, i and j represent the ith member in the population on the jth dimension.

3. **UPDATING/MAINTAINANCE**

The old and new virus population is combined and sorted on the basis of the objective function value. If the average objective function does not improve within replications, the value of intensity is increased by one.

Antivirus mechanism works as the population maintenance mechanism. This is triggered after each replication to kill a given number of viruses as given by the equation below:

$$\textit{Amount}=\textit{rand}(0, \textit{population\_size} - \textit{strong members}) - (4)$$

Where population_size is the number of viruses inside the host cell. Application of the antivirus eliminates the common viruses from the population. It also helps in making sure that the population size remains fixed after every replication. If the population size exceeds 1000(total number of viruses that can exist within the host cell at a given time), the excess population is discarded.

4. STOPPING CRITERION

The above steps are repeated till the stopping criterion is met i.e. the maximum number of replications is achieved.

Pseudo code of VOA

```
//Initialization
Virus_population ←  generate_initial_population (parameter_values);

Viruses_strength ← Evaluate_objective_function_value(Virus_population);

While(TRUE)
[Strong_viruses, Common_viruses] ←Classification(Virus_population, Viruses_strength);

[New_strong, New_common] ←Replication(Strong_viruses, Common_Viruses); //Replication

New_members ←Storage(New_Strong, New_Common);

New_members_strength ←Evaluate_objective_function_value(New_members);

Virus_population ←Combine(Virus_population, New_members);//Updating

Population_performance ←Average_objective_function_value(Virus_population);

If(Population_performance did not improve) →Intensify_exploitation();

Apply Antivirus(); //Maintenance

If(populationsize exceeds 1000 members)→Reduction(Virus_population, Virus_strength,initial_value);

Stop ← Evaluate_stopping_criterion();
```

```
If(Stop==True) → BREAK WHILE;

End While
```

3.2 COCOMO II Model

The Constructive Cost Model (COCOMO) is a regression-based software cost estimation model developed by Barry W. Boehm in 1981 which is also called in references as COCOMO 81. COCOMO-81 is said to be the best known, best documented and it reflects most software development practices on that time.

One of the problems with the use of COCOMO I today is that it does not support modern software development processes like desktop development, code reusability, rapid-development, object-oriented approaches etc. Therefore, in 1997, Boehm developed the COCOMO II for estimating modern software development projects.

COCOMO consists of a hierarchy of three increasingly detailed and accurate forms:

1. **Basic COCOMO** computes software development effort (and cost) as a function of program size and it holds up until a certain point, usually for projects that can be reasonably accomplished by small teams of two or three people.

2. **Intermediate COCOMO** provides more accurate estimates by taking into account software development environment through 15 cost drivers.

3. **Detailed COCOMO** computes effort as a function of program size and a set of cost drivers given according to each phase of software life cycle i.e. analysis and design of the software engineering Process.

The COCOMO estimated software effort is given by below equation and is measured in calendar months

Effort(Person -month) = a´[LOC]b ´(EM1´EM2´EM3´ ´EM15) … (5)

Here the coefficient “a” is known as productivity coefficient and the coefficient “b” is the scale factor. They are based on the different modes of project as given in Table 3.1

Table 3-1 : Software Project Mode

Software Project	**Project Size**	**a**	**b**
Organic	Less than 50 KLOC	3	1.1
Semi-detached	50 – 300 KLOC	3	1.1
Embedded	Over 300 KLOC	3	1.2

And EMi are effort multipliers (Cost drivers) which have up to six levels of rating: Very Low, Low, Nominal, High, Very High, and Extra High. Each rating has a corresponding real number based upon the factor and the degree to which the factor can influence productivity as given in Table 3.2

Table 3-2 : Software Cost Drivers

Cost Drivers	**Rating**					
	Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
acap	1.46	1.19	1	0.86	0.71	-
pcap	1.42.	1.17	1	0.86	0.7	-
aexp	1.29	1.13	1	0.91	0.82	-
modp	1.24.	1.1	1	0.91	0.82	-
tool	1.24	1.1	1	0.91	0.83	-
vexp	1.21	1.1	1	0.9	-	-
lexp	1.14	1.07	1	0.95	-	-
sced	1.23	1.08	1	1.04	1.1	-
stor	-	-	1	1.06	1.21	1.56
data	-	0.94	1	1.08	1.16	-
time	-	-	1	1.11	1.3	1.66
turn	-	0.87	1	1.07	1.15	-
virt	-	0.87	1	1.15	1.3	-
rely	0.75	0.88	1	1.15	1.4	-
cplx	0.7	0.85	1	1.15	1.3	1.65

These effort multipliers fall into three groups: those that are positively correlated to more effort, those that are negatively correlated to more effort and the third group containing just schedule information.

3.3 NASA 63 Data Set

The proposed algorithm is evaluated on 63 NASA projects from different centers from the years of 1971 to 1987. As shown in table 3.3, this dataset consist of development mode (embedded, organic, semidetached), EAF of 15 cost drivers, size of each project in kilo source line of code and actual effort.

Mode	EAF	LOC	Effort
Embedded	2.28811	113	2040
Embedded	0.53128	6.9	8
Embedded	5.50991	22	1075
Embedded	2.01377	30	423
Embedded	1.73015	29	321
Embedded	1.73015	32	218
Embedded	0.93626	37	201
Embedded	4.94502	3	60
Embedded	3.04353	3.9	61
Embedded	2.37496	6.1	40
Embedded	1.94746	3.6	9
Embedded	3.27117	320	11400
Embedded	3.48791	299	6400
Embedded	0.84607	252	2455
Embedded	0.96816	118	724
Embedded	0.7025	90	453
Embedded	1.1639	38	523
Embedded	0.95249	48	387
Embedded	0.99439	1.98	5.9
Embedded	0.56909	390	702
Embedded	2.30187	42	605
Embedded	1.47674	23	230
Embedded	0.30168	91	156
Embedded	0.3401	6.3	18
Embedded	2.66087	27	958
Embedded	3.30632	17	237
Embedded	1.05362	9.1	38
Organic	0.32046	132	243
Organic	0.99814	60	240
Organic	0.65617	16	33
Organic	1.86504	4	43
Organic	0.85243	25	79
Organic	1.6573	9.4	88
Organic	0.68887	15	55
Organic	0.37224	60	47

Organic	0.3588	15	12
Organic	0.38774	6.2	8
Organic	0.9649	3	8
Organic	0.25445	5.3	6
Organic	0.58734	45.5	45
Organic	1.06981	28.6	83
Organic	1.33662	30.6	87
Organic	0.87268	35	106
Organic	0.82473	73	126
Organic	1.28037	24	176
Organic	2.30456	10	122
Organic	1.15428	5.3	14
Organic	0.77736	4.4	20
Organic	1.08961	25	130
Organic	1.00697	23	70
Organic	2.12549	6.7	57
Organic	0.38613	10	15
Semidetached	0.84227	293	1600
Semidetached	0.67554	1150	6600
Semidetached	0.90842	77	539
Semidetached	2.81069	13	98
Semidetached	0.99439	2.14	7.3
Semidetached	3.43917	62	1063
Semidetached	2.17879	13	82
Semidetached	0.38067	23	36
Semidetached	0.75808	464	1272
Semidetached	1.37602	8.2	41
Semidetached	0.4466	28	50

Chapter 4 Proposed Work

This chapter gives the understanding of the proposed work “Optimizing Intermediate COCOMO Model using Virus Optimization Algorithm”. The work can be divided into two parts. In part 1, we basically derive the new values of coefficients of the COCOMO II model, for all types of systems i.e. organic, semi-detached and embedded, and in next part, we have applied Virus Optimization Algorithm to each section in order to derive the new values of ‘a’ and ‘b’ for all the three systems i.e. organic, semi-detached and embedded.

DATASET

We have used NASA 63 dataset and divided it as follows:

- Dataset for organic system
- Dataset for Semi-Detached system
- Dataset for Embedded system

This dataset consists of development mode, 15 cost drivers, size of each project in KLOC and actual effort

4.1 Working Steps

To primary aim of this work is to optimize the intermediate COCOMO drivers such that the calculated efforts are approximately same as the actual efforts for NASA 63 dataset. Virus Optimization Algorithm solves the problem of finding those values for the 15 Effort Multipliers such that the following equations are minimized:

- MRE : Magnitude of Relative Effort(MRE) can be defined as :

  $$MRE=|actual\_effort_i - estimated\_effort_i|/ actual\_effort_i \text{--- } (7)$$

- MMRE : The mean magnitude of relative error (MMRE) is achieved through the summation of MRE over N observations :

  $$MMRE=\sum_{i=1 \text{ to } N} MRE \text{---- } (8)$$

The main steps to optimized COCOMO II using VOA are given below :

Step 1: Generate Initial Virus Population

The initial virus population is randomly generated for each dimension d

$X_{ij} = x_{min\,j} + rand\,(0, 1)\,(x_{max\,j} - x_{min\,j}) - (9)$

Step 2: Evaluate NASA 93 DATASET using COCOMO Model

Calculate MRE for each 63 projects in NASA dataset using Eq. (7) where Actual Effort is given and Estimated Effort is COCOMO calculated effort using Eq. (5). Then overall fitness (MMRE) is calculated using Eq. (8).

Step 3: Evaluate initial population on NASA 63 DATASET and select best solution

For each virus in population, fitness (MMRE) is calculated by taking average of MRE's for each NASA project. Then best virus is selected from them having min fitness i.e. min. MMRE.

Step 4 : Generate new solution for each Virus i in Generation j, either by exploration or exploitation using equation 2 and 3 respectively.

Step 5: Evaluate the fitness (MMRE) of newly generated individual of virus.

Step 6: Accept the new solution, if its fitness is better that old solution.

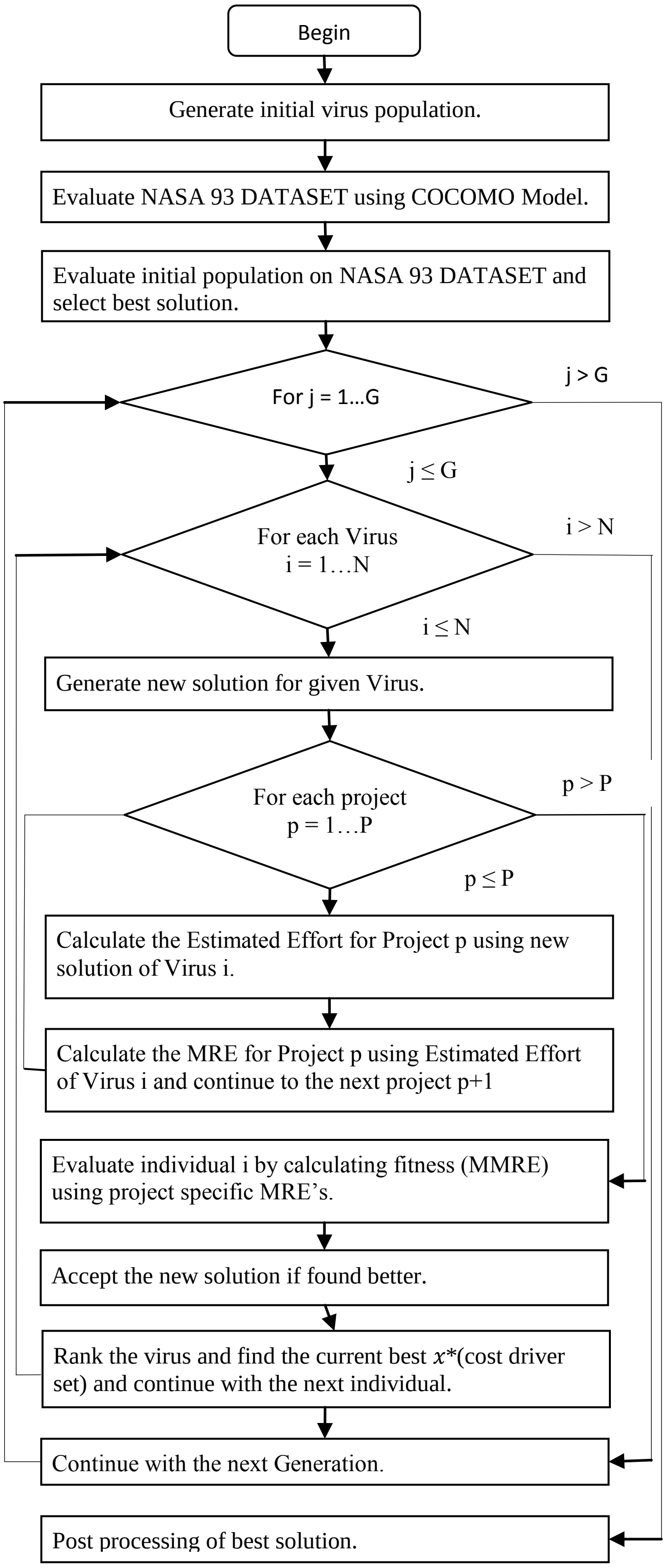
Step 7: Rank the virus and find the current best solution x*.

Step 8: Repeat step 4 to 7 for each individual in generation j.

Step 9: After the N generations, final x* would act as the optimized solution to the problem.

Step 10: Post process the result.

4.2 Flowchart For The Proposed Work


Begin
Generate initial virus population.
Evaluate NASA 93 DATASET using COCOMO Model.
Evaluate initial population on NASA 93 DATASET and select best solution.
For j = 1…G
j > G
j ≤ G
For each Virus i = 1…N
i > N
i ≤ N
Generate new solution for given Virus.
For each project p = 1…P
p > P
p ≤ P
Calculate the Estimated Effort for Project p using new solution of Virus i.
Calculate the MRE for Project p using Estimated Effort of Virus i and continue to the next project p+1
Evaluate individual i by calculating fitness (MMRE) using project specific MRE's.
Accept the new solution if found better.
Rank the virus and find the current best x*(cost driver set) and continue with the next individual.
Continue with the next Generation.
Post processing of best solution.

The proposed model is implemented in Java for all modes.

Chapter 5 Result and Analysis

In this project, we have implemented Virus Optimization Algorithm to optimize the COCOMO II parameters (a,b), such that the calculated efforts are almost same as the actual efforts for NASA 63 datasets. We have used VOA to minimize the MRE and MMRE. The efficiency of the algorithm has been measured by comparing the results of our algorithm with various nature inspired algorithms like Genetic Algorithm, BAT Algorithm and BFO Algorithm.

5.1 Experimental Setup

For this implementation, we have used the following parameters for Virus Optimization Algorithm:

1. d=2
2. Population size=1000
3. Number of strong virus = 300
4. Number of common viruses = 700

As all the three modes of COCOMO model have their own coefficients a and b, we have performed experiments on each mode individually.

The experiment was conducted on Windows 7 operating system with 2 GB RAM and 2.40 GHz Intel Pentium Dual Core Processor. The code is written in JAVA and run on Eclipse tool.

5.2 Experimental Results

We have performed the following analysis on the three modes of COCOMO model:

5.2.1 Organic Mode:

In experiments using organic mode, the best results were achieved with 100 iterations. In all the runs of the algorithm, we found that the MMRE with Virus Optimization Algorithm is the lowest when compared to COCOMO, genetic algorithm, and BAT algorithm.

5.2.2 Semi-detached Mode:

In experiments using semi-detached mode, the best results were achieved with 150 iterations. The results gave the value of MMRE which was much less than those obtained using COCOMO, Genetic Algorithm or BAT Algorithm.

5.2.3 Embedded Mode:

In the experiments with embedded mode, the best results were obtained with 80 iterations. MMRE converged to an optimal solution very fast. Also, the result obtained is the lowest when compared to COCOMO, Genetic Algorithm or BAT Algorithm.

5.3 Comparisons

We have concluded the final results of the experiment in the table below:

Table 4-1 : MMRE calculated using different models

	MMRE for COCOMO	MMRE for Genetic Method	MMRE for BAT	MMRE for VOA
Organic	0.876	0.8	0.3093	0.2918
Semidetached	0.51	0.51	0.2157	0.2119
Embedded	0.82	0.72	0.3826	0.3723

The following graph depicts the MMRE obtained through various methods listed in Table above.

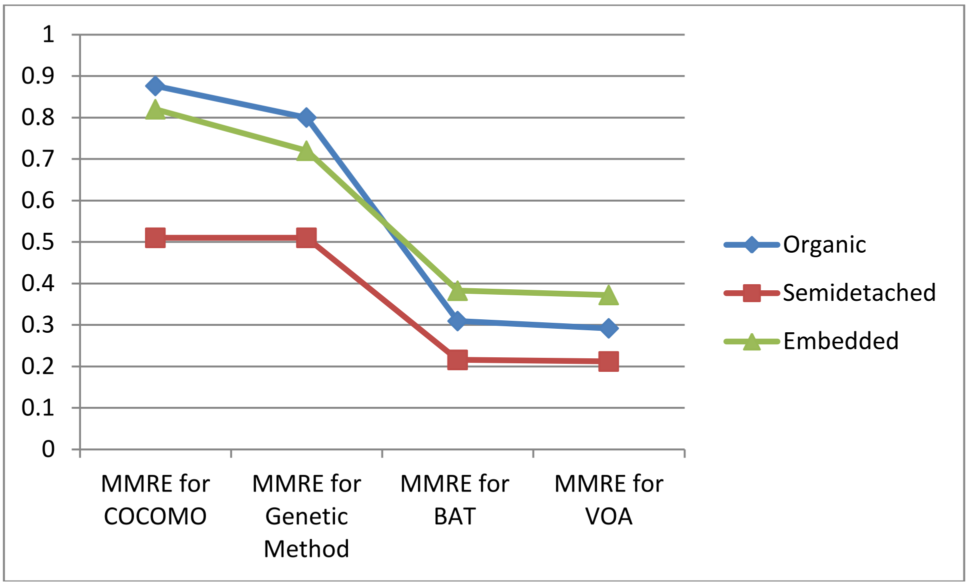


Figure 11 : Comparison of MMRE

Chapter 6 Conclusion and Future Work

The current research proposes a Virus Optimization Algorithm approach for optimization of the coefficients of intermediate COCOMO model. The proposed algorithm gave better results than the traditional COCOMO model, Genetic algorithm and BAT algorithm. The algorithm also shows better convergence as compared to the traditional algorithms.

In future, Virus optimization algorithm can be used to solve complex problems like feature selection. We can also derive new algorithms based on the diverse behavior of viruses, the fact that viruses are of various types, mutate easily, and cannot be killed by animal’s antibodies completely, hence, giving an optimal solution to complex real world problems.

References

[1] Allahyari, S., Salari, M., & Vigo, D. (2015). A hybrid metaheuristic algorithm for the multi-depot covering tour vehicle routing problem. *European Journal of Operational Research* , 756-768.

[2] Yazdani, M., & Jolai, F. (2015). Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm. *Journal of Computational Design and Engineering* .

[3] Dash, T., & Sahu, P. K. (2015). Gradient gravitational search: An efficient metaheuristic algorithm for global optimization. *Journal of Computational Chemistry* , Pages 1060-1068.

[4] Zhou, L., Ding, L., Qiang, X., & Luo, Y. (2015). An Improved Discrete Firefly Algorithm for the Traveling Salesman Problem. *Journal of Computational and Theoritical Nanoscience* , pp. 1184-1189(6).

[5] Wong, K.-C. (2015). Evolutionary Algorithms : Concepts, Designs, and Applications in Bioinformatics. *arXiv:1508.00468v1 [cs.NE]* .

[6] Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI:University of Michigan Press.

[7] Singh, B. K., & Misra, A. K. (2012). Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects. *International Journal of Computer Applications* , 0975-8887.

[8] Sharma, K. (2012). *Software Cost Estimation: Swarm Intelligence Approache.*

[9] Gupta, N., & Sharma, K. (2015). Optimizing intermediate COCOMO model using BAT algorithm. *Computing for Sustainable Global Development (INDIACom), 2015* (pp. 1649-1653). New Delhi: IEEE.

[10] Breitbart, M., & Rohwer, F. (2005). Here a virus, there a virus, everywhere the same virus? In *Trends in Microbiology* (pp. 278-284). Cell Press.

[11] Koonin, E., Senkevich, T., & Dolja, V. (2006). The ancient Virus World and evolution of cells. In *Biology Direct* (pp. doi:10.1186/1745-6150-1-29). PMID 16984643.

[12] Rybicki, E. (1990). he classification of organisms at the edge of life, or problems with virus systematics. . S Afr J Sci.

[13] EC., H. (2007). Viral evolution in the genomic age. *doi:10.1371/journal.pbio.0050278* .

[14] Wimmer, E., Mueller, S., Tumpey, T., & Taubenberger, J. (2009). Synthetic viruses: a new opportunity to understand and prevent viral disease. In *Nature Biotechnology* (pp. 27(12):1163–72). PMID 20010599.

[15] Evans, A. S. (1989). *Viral Infections of Humans: Epidemiology and Control.* New York: Plenum Publishing Corporation.

[16] Das, P., & DE, S. (2000). A Susceptible-Infected Removal (SIR) epidemic model. *Indian Journal of Pure and Applied Mathematics* .

[17] Pattnaik, S. S., Jadhav, D. G., Devi, S., & Ratho, R. K. (2012). Swine influenza inspired optimization algorithm and its application to multimodal function optimization and noise removal. *International Peer-reviewed and Open Access Journal for the Artificial Intelligence Specialists* .

[18] Wang, W., Du, Y., Li, Q., & Fang, Z. (2012). Dynamic Multi-objective Optimization Algorithm Based. *Research Journal of Applied Sciences, Engineering and Technology* , 4(2): 90-92.

[19] Liang, Y.-C., & Juarez, J. R. (2015). A novel metaheuristic for continuous optimization problems: Virus optimization algorithm. *Engineering Optimization.* Taylor & Francis.

[20] Liang, Y. C., & Juarez, J. (2012). Multilevel Image Thresholding Using Relative Entropy and Virus Optimization Algorithm. *IEEE World Congress on Computational Intelligence.* Brsibane, Australia.