The International Journal Of Engineering And Science (IJES)
|| Volume || 3 || Issue || 6 || Pages || 30-36 || 2014 ||
ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

# A particle Swarm Optimization-based Framework for Agile Software Effort Estimation

[1]Manga I , & [2]Blamah NV
[1]Department of Computer Science, Adamawa State University, Mubi, Nigeria
[2]Department of Computer Science, University of Jos, Nigeria

**ABSTRACT**
*Software effort and cost estimation process in any software engineering project is a very critical component. The success or failures of projects depend heavily on the accuracy of effort and schedule estimations. The paper examined A Particle Swarm Optimization-Based Framework for Agile Software Effort Estimation. Traditional approaches were used to estimate effort for agile projects, but they mostly result in inaccurate estimates. This paper aimed at the application of some Particle swarm optimization framework as a soft computing technique for agile software development methodology effort estimation. The paper also identified project that uses agile development methodology, later applied Particle Swarm Optimization to minimize project duration and effort required to build software. Finally the PSO model improves the effort and time estimation accuracy by minimizing these parameters and the estimates values are close to the actual results. Generally, the acceptable target value for Mean magnitude of relative error (MMRE) is 25%. It indicates that the magnitude of relative error (MRE) for each project for the established estimation model should be less than 25% on the average. A software development effort estimation method with a smaller MMRE value than the one with bigger MMRE value gives better estimates than a model with a bigger MMRE value. The MMRE obtained from the paper indicated that the MMRE value for effort is 5.12% less than the normal established estimation model.*

***KEYWORDS:*** *Optimisation, effort estimation, agile, software*

Date of Submission: 10 June 2014 Date of Publication: 30 June 2014

## I. INTRODUCTION

Software effort and cost estimation process in any software engineering project is a very critical component. The success or failures of projects depend heavily on the accuracy of effort and schedule estimations. The introduction of agile methodology in the software development industries presented many opportunities for resarchers and practitioners.Today, software exceeds 25 million source code statements because of the complexity and size of the software. Software development organizations require more technical staff or personnel and the cost of such software may be in millions dollars. Errors in cost estimation can be very serious indeed [7]. This is because over estimating the cost of software project leads to too many resources allocated into the project and under estimating the cost of the project leads to little resources allocated into the project. Therefore, accurate estimation of the cost before the start-up of a project is essential for both the developers and the clients.The most important measure of efficiency of any software engineering projects is its ability to reach completion on time and on budget regardless of any environment the software may operate within. Software cost estimation is important when developing a system and has been a vital but difficult task since the inception of computer, [17]. Software costs are mainly refers to the effort spent in a development of a software project, which is increasingly concerned by the developers and the users. If we could make a good estimation of the software workload before the development, the software development managers may improve the quality of software products through controlling the development time and budget during software development process,[4] .Most of estimation models attempt to generate an effort estimate, which can then be converted into the project duration and cost. Although effort and cost are closely related, they are not necessarily related by a simple transformation Function.

## II. STATEMENT OF THE PROBLEM

This paper addresses agile effort estimation framework within software products that address the most frequently asked questions software development which includes:
[1] How much effort is required to complete each activity?
[2] What is the time is needed to complete each activity?
[3] The total cost of each activity?

[4] When such questions are posted, a lot of options are available as solution; this study therefore set out to design a Particle swarm Optimization-Based frame work that would be used to estimate efforts for agile software.

## III. AIM AND OBJECTIVES OF THE STUDY

The aim of this paper is to study a software cost estimation frame work for agile processes using particle swarm optimization algorithms. The specific objectives of the study are to:

[1] Identify projects that use agile processes in software development, determine the effort and schedule estimations that yield the highest degree of accuracy and reliability (Optimization); and

[2] Provide a framework to determine optimum duration, effort and schedule estimation required to build agile software using a particle swarm optimization model.

## IV. THEORETICAL FRAME WORK

According to [10] a software project is a project with high uncertainty, so that software project success is relatively low. [13]. Notes that software development is a highly complex and unpredictable task since many specialized groups are typically required to collaborate on one project.The ability to accurately and consistently estimate software development efforts, especially in early stages of the development life cycle, is required by the project managers in planning and conducting software development activities because the software price determination, resource allocation, schedule arrangement and process monitoring are dependent upon it. This issue lie in the fact that the software development is a complex process due to the number of factors involved, including the human factor, the complexity of the product that is developed, the variety of development platforms, and difficulty of managing large projects, [3].According to [17] project cost estimation and project scheduling are normally carried out together. The costs of development are primarily the cost of effort involved, so the effort computation is used in both the cost and schedule estimate. For most projects, the dominant cost is the effort cost. Software effort mainly refers to the effort spent in a development of a software project, which is increasing concerned by the developers and the users [4].Software cost estimation is not a “standalone activity”. The estimates are derived in large from the requirements of the project, and will be strongly affected by the tools, process, and their attributes associated with the project [7]. According to research conducted by [6] the Information System development process, regardless of the methodology adopted, requires effective management and planning. A large part of this planning is the creation of estimates at the beginning of a project so that resources can be appropriately allocated.

Estimating the cost of an IS development project is one of the most crucial tasks for project managers [9] but despite this it continues to be a weak link in the IS development field [2]. Information System (IS) development projects have a long history of being delivered over time, over budget and failing to satisfy requirements. The main factors that are typically estimated at the beginning of an IS development project are: cost, size, schedule, people resources, quality, effort, resources, quality, effort, resources, maintenance costs, and complexity. Estimates are produced and used for a variety of purposes and a study by [11] shows the most common uses. These are: to schedule projects for implementation, to quote the charges to users for projects, to staff projects, to audit project success, to control or monitor project implementation, to evaluate project estimators, and to evaluate project developers. According to [8] software cost estimation is the process of gauging the amount of effort required to build software project. The effort is usually represented in Person-Month (PM) and it depends upon both the size as well as the complexity of the given software project. The PM can be converted to dollar cost. The model was designed in such a manner that accommodates the COCOMO model and improves its performance. It also enhances the predictability of the software cost estimates. The model was tested using two datasets COCOMO dataset and COCOMO NASA 2 dataset. The paper was titled an adaptive leaning approach to software cost estimation.

There so many methodologies introduced in software development. Indeed, 25 years, a large number of different approaches to software development have been introduced, of which only few have survived to use today [1]. Right now, agile methodology is the most popular methodology in software development. Agile methodology emerged due to evolving and changing software requirements [14]. As this approach the requirement is not always feasible there is also a need for flexible, adaptive and agile method, which allow the developers to make late change in the specifications [1].According to [16], agile software development methods like extreme programming try to decrease the cost of change and therewith reduce the overall development costs. Agile methods try to avoid the deficits of classic software development procedures .Mostly, the following methodologies are considered to reach this aim: short release cycles, simple design, continuous testing, and refactoring, collective ownership, coding standard and continuous integration. Other characteristic of agile methodologies according to [12] includes: Effort and schedules estimates normally are computed using parametric models according to the size of the software project, whose size is measured by lines of code (LOC) or function points and so forth. There are four basic steps in software project effort and schedule estimation. They can be summarized and follows:

## V. METHODOLOGY

Models of software cost/effort are approaches that identify key contributors to cost and effort generating mathematical formulae that associate these attributes to cost and effort. [15] Identify many quantitative models from different studies by different papers which are used to estimate effort required to develop a software system.

For the purpose of this study, the paper considered Particle Swarm Optimization (PSO) model. User stories from agile velocity and project duration will be optimized and better framework for agile effort estimation will be achieved.

## VI. OPTIMIZATION PROBLEM AND MODEL FORMULATION

[1] Basic component of an optimization problem are:
[2] An objective function expresses the main aim of the model which is minimized.
[3] A set of unknowns or variables control the value of objective Function
[4] Objective function is the mathematical function that is minimized.

This is the selection of design variable, objective function and model of the design. A design variable, that takes a numeric value, will be controlled from the point of view of the design. Design variable are bounded, that is, it will have a maximum and minimum value. An objective is the numerical value that is minimized.

## V. FUNCTIONS THAT ARE MINIMIZED

In order to estimate duration needed to complete a project, it is calculated as

$$T = \frac{E}{V}(Days) \qquad 3.1$$

$$= \sum_{I=i}^{n} \frac{(ES)_i}{(V)_i^D}(Days) \qquad 3.2$$

The unit of T in this calculation is Days which can be then converted to months, dividing by number of working days per month. Thus

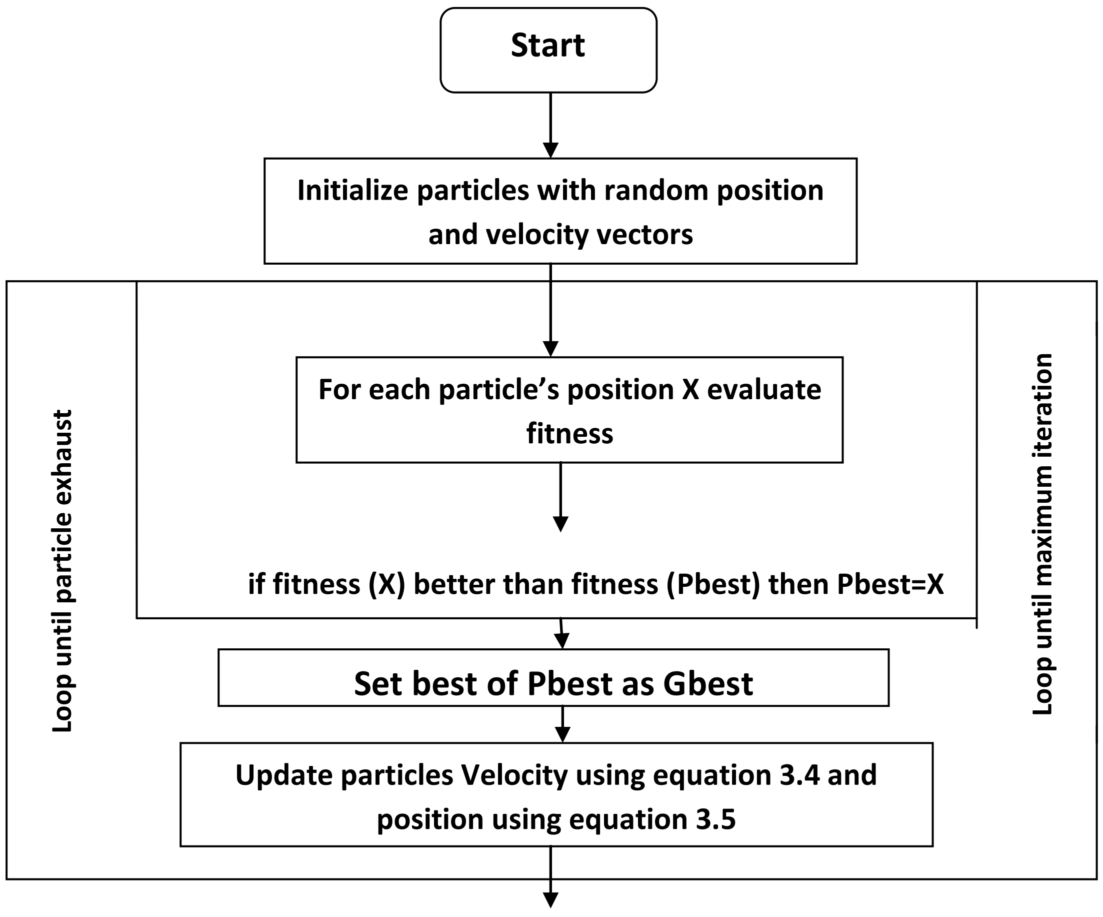
$$T = \sum_{i=1}^{n} \frac{(ES)_i}{(V)_i^D}( * (\frac{1}{W}) Months$$

Where WD is work days per month, V is the project velocity, E is the effort, ES is the effort of user story, T is the project duration or time and D is critical and is called project deceleration.

## VI. PARTICLES SWARM OPTIMIZATION (PSO)

Particle Swarm Optimization (PSO) is a population based search algorithm developed, the technique is based on the movement and intelligence of swarms. It uses a concept of social interaction for problem solving. The Population contains set of particles each of which represents a solution for a given optimization problem. These particles are normally initialized randomly as most evolutionally computation techniques (for example, genetic algorithms). During the evolutionary process, each particle, based on some evaluation criterion, updates its own position with certain velocity. The velocity is compiled based on both the best experience of the particle itself and that of the entire population. This update process is repeated for a number of generations. The update process stops either when the objectives are reached or when the maximum number of generation is reached. Summary of general concept of PSO is:

[1] It consists of a swarm of particles.
[2] Each particle resides at a position in the search space.
[3] The fitness of each particle represents the quality of its position.
[4] The particles fly over the search space with a certain velocity.
[5] Each particle is treated as a point in an N – dimensional space which adjusts its “flying” experience of other particles.
[6] The velocity (both direction and speed) of each particle is influenced by its own position found so far and the best solution that was found so far by its neighbours.
[7] Eventually the swarm will converge to optimal position.


Start
Initialize particles with random position and velocity vectors
For each particle's position X evaluate fitness
if fitness (X) better than fitness (Pbest) then Pbest=X
Set best of Pbest as Gbest
Update particles Velocity using equation 3.4 and position using equation 3.5
Loop until particle exhaust
Loop until maximum iteration

$$V_i^{k+i} = V_i^k + c_1 r_1 \left(Pbest_i^k - X_i^k\right) + c_2 r_2 \left(Gbest_i^k + X_i^k\right) \qquad 3.4$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \qquad 3.5$$

In a physical *n*-dimensional search space, the position and velocity of each particle i are represented as the vectors:

$$X_i = \left(x_{i1}, \ldots, x_{in}\right) \quad and \quad V_i = \left(v_{i1}, \ldots, v_{in}\right)$$

$C_1$ and $C_2$ are acceleration (weighing) factors known as cognitive and social scaling parameters. Determine the magnitude of the random forces in the direction of Pbest and Gbest.
$r_1$ and $r_2$ are random numbers between 0 and 1.
K is the iteration index
The acceleration coefficient should be set sufficiently high.
Higher acceleration coefficients result in less stable systems in which the velocity has a tendency to explode.
$V_{max}$ was introduced to control the velocity exposition.The motivation behind introducing the inertia weight ( $\omega$ ) was the desire to better control the scope of the search and reduce the importance of (or eliminate) $V_{max.}$

The inertia weight can be used to control the balance between exploration and exploitation. When $\omega$ is big, particle swarm tend to global search while they tend to local search when it is small?
Hence, suitable selection of the inertia weight $\omega$ can provide a balance between global and local exploration abilities and thus require less iteration on average to find the optimum.
With the introduction of inertia weight $\omega$ , the equation to update particle velocity becomes:

$$V_i^{k+i} = \omega V_i^k + c_1 r_1 \left(Pbest_i^k - X_i^k\right) + c_2 r_2 \left(Gbest_i^k + X_i^k\right) \qquad 3.6$$

While the equation to update particle position remains the same.

$$X_i^{k+1} = X_i^k + V_i^{k+1}$$

**Performance Indicators on PSO Application:** In the paper of software cost and effort estimation, the performance indicators used is usually using the Mean of Magnitude of Relative Error (MMRE) or Prediction level (Pred) as accuracy reference. Therefore in this study, MMRE was used to determine the effectiveness of

PSO application. $E$ is the effort and the equation used for the computation was based on the Effort Mean Magnitude of Relative Error using the equation:

$$EMMRE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{AE-EE}{AE}\right| \qquad 3.7$$

$EMMRE$ is the Effort Mean Magnitude of Relative Error, $AE$ is the Actual Effort, $EE$ is the Estimated Effort, $n$ is the number of projects and $i$ is a number (No). $MRE$ is the magnitude of relative error and is computed as:

$$MRE = \left|\frac{AE-EE}{AT}\right|$$

$T$ is the Time and the Time Mean Magnitude of Relative Error is determined from the equation:

$$TMMRE = \frac{1}{n}\sum_{1}^{n}\left|\frac{AT-ET}{AT}\right| \qquad 3.8$$

$TMMRE$ is Time Mean Magnitude of Relative Error is, $AT$ is the Actual Time and $ET$ is the Estimated Time. $MRE$ Is the Magnitude of Relative Error which is computed the eqution:

$$MRE = \left|\frac{AT-ET}{AT}\right|$$

## VII. DISCUSSION

The summary of the results obtained from the implementation of the particle swarm optimization framework to optimize the project duration (Time) and effort is shown below. Table 4.1 shows the data obtained from ten past agile projects, their respective velocities (V), the work days per month in the projects, project deceleration (D) and their respective actual effort. Estimated effort (EE) using PSO framework from the implementation of the algorithm in C-Sharp and the computed Magnitude of Relative Error for effort is also presented in appendix. Table 4.1 in appendix shows the project number, agile project velocity ($V_i$), Project work days, agile sprint size, and the actual efforts over ten historical completed agile projects, the estimated effort (EE) using PSO framework and the effort MRE. Table 4.2 shows the actual time over ten agile past completed projects, the estimated time using PSO and the Time Magnitude of relative error (MRE).The evaluation consists in comparing the accuracy of the estimated effort with the actual effort. There are many evaluation criteria for software effort estimation; among them the paper considered the most frequent one the Magnitude of Relative Error (MRE) and Mean Magnitude of Relative Error (MMRE). The Effort Mean Magnitude of Relative Error (EMMRE) and Time Mean Magnitude of Relative Error (TMMRE) are defined as in equation3.7 and equation 3.8 respectively. The Mean Magnitude of Relative Error (MMRE) computes the average of MRE over N projects. Using equation 3.7, the computed value for EMMRE was found to be 0.1988 which is about 19.88% and TMMRE was 0.2310 which is 23.10%Generally, the acceptable target value for MMRE is 25%. It indicates that the MRE for each project for the established estimation model should be less than 25% on the average [5].
A software development effort estimation method with a smaller MMRE value than the one with bigger MMRE value gives better estimates than a model with a bigger MMRE value. The MMRE obtained from the paper indicated that the MMRE value for effort is 5.12% less than the normal established estimation model. This works in conformity with [5] that the less is the MMRE value than the MMRE value of the established estimation model the more accurate is the effort

## VIII. CONCLUSION

The level of complexity of software projects today has drawn much attention to the need for methods of estimating how much effort will be required, how long it will take, and how many people will be needed to build software. Therefore, software costing should be carried out objectively with the aim of accurately predicting the effort, time and staff level to develop software.Accurate and reliable software project estimates such as time, effort in the early phase of software development is one of the crucial objectives in software project management.

## IX. RECOMMENDATIONS

After careful considerations of the results obtained from the tables, the following recommendations are made:

[1] Developing software products should require taking into consideration factors such as environment, size of the products, project velocity, users' stories and the model to be used.

[2] Because software size is the key input for most software parametric estimating models, it is critical that accurate estimating techniques be used by agile software team instead of estimating program size based on opinions of one or more experts.

[3] Due to inherent limitations of non-parametric models adopted by some agile development team, this study recommends that software developers should adopt the newest models that will give reliable effort estimation that is based on current development.

## REFERENCES

[1] P. Abraham, O. S., Ronkanen & J .Warster. (2002). *Agile software development method*. Retrieved October ,2013, from VTT Home Page: http://www.vtt.fi/inf/publications/2002/478.pdf

[2] R. Agarwal, M. Mumar, G. Yogesh, S. Malliick, R.M. Bharadwaj & D. Anantwar (2001). Estimating software projects. *AGM SIGSOFT, Software Engineering Notes*. 26, 60-67.

[3] I. Attarzedah & S.H. Ow (2010). A novel soft computing to increase the accuracy of software cost estimation. *IEEE*.

[4] B. Baskeles, , U., Boyazici, B. Turam & A Bener. (2007). *Software effort estimation using machine learning methods*. Computer Information Sciences *ISCIS*. IEEE.

[5] A. Kaushik, R Soni. & A.K. Soni, (2012). An Adaptive Learning Approach to Software Cost Estimation. *2012 National Conference on Computing and Communication Systems (NCCCS).* IEEE.

[6] K. Coonboy & A. Fitzgeneral, (2004). Towards a Conceptual Framework for Agile Methods. *Proc. AGM Workshop on Interdisciplinary Software Engineering Paper*.

[7] C. Jones (2007). Why Flawed Software Projects are not Cancelled in Time. *Cutter IT Jounal, 16*, 12-17.

[8] M. Jorgensen & M Shepperd. (2007). "A systematic Review of Software Development Cost Estimation Studies". *IEEE Transaction on Software Engineering,*1, 201-205.

[9] J. Keung, R Jeffery. & B. Kitchenham,. (2004). The challenge of introducing a new software cost estimation technology into a small software organisation. *Proceedings of the 2004 Australian Software Engineering Conference.* Sydney, Australia.

[10] T.B. Kusumasari,, , I. Supriana , S Surendro. &,H. Sastramihardja (2011). Collaboration model of software development. *2011 International Conference on Electrical Engineering and Informatics.* Bandung, Indonesia.

[11] A.L. Lederer,& J. Prasad, (1995). Perceptual and information system cost estimation. *AGM SIGCPR Conference on Supporting Teams, Groups, and Leaning Inside and Outside the IS Function REINVENTING IS.* Nashville Tennessee.

[12] G Miller,. (2001). The Characteristics of agile software processes. *Proceedings of the 39th Int'l Conf. and Exhibition on Technology of Object Oriented Languages and Systems.*

[13] K.S. Na, X. Li,. J.T. Simson & K.Y .Kim. (2004). uncertainity profile and software project performance:A Cross National Comparison. *The Journal of System and Software,70*, 155-163.

[14] M. Omar, S.L Syed-Abdullahi. & A.Yasin,. (2003). The Impact of Agile Approach on Software Cost Estimation.

[15] R. Racovic, (2004). Towards a Methodology to Estimate Cost of Object-Oriented Software Developpment Projects. *ComSIS, 1 (2*) 173-194.

[16] A. Schmietendorf, M Kunz. & R Dumke. (2008*). Effort estimation for agile software development projects*. 5th Software Software Measurement European Forum. Milan.

[17] I. Sommerville. (2008). *Software Engineering.* USA: Addison Wesley.

[18] S. Ziuddin, T Kamal. & Z. Shahrukh, (2012). An Effort Model for Agile Software Development. *Advances in Computer Science and its Application (ACSA),* 2(1), 315-324

**APPENDIX**

**Table 4.1 PSO Based Effort**

| Project No | $V_i$ | Work Days | Sprint Size | D | Actual Effort | PSO based Effort (EE) | Effort MRE |
|---|---|---|---|---|---|---|---|
| | 4.20 | 22.00 | 10.00 | 0.6871 | 156.00 | 152.00 | 0.0256 |
| | 3.70 | 21.00 | 10.00 | 0.7013 | 202.00 | 197.00 | 0.0248 |
| | 4.00 | 22.00 | 10.00 | 0.8789 | 173.00 | 178.00 | 0.0289 |
| | 3.80 | 22.00 | 10.00 | 0.8868 | 331.00 | 325.00 | 0.0272 |
| | 4.90 | 22.00 | 10.00 | 0.9034 | 124.00 | 121.00 | 0.0242 |
| | 4.10 | 22.00 | 10.00 | 0.9034 | 339.00 | 347.00 | 0.0236 |
| | 4.20 | 22.00 | 10.00 | 0.8601 | 97.00 | 94.00 | 0.0309 |
| | 3.80 | 22.00 | 10.00 | 0.8332 | 257.00 | 254.00 | 0.0117 |
| | 3.90 | 22.00 | 10.00 | 0.6750 | 84.00 | 85.00 | 0.0119 |
| | 4.60 | 22.00 | 10.00 | 0.7632 | 211.00 | 214.00 | 0.0142 |

Table 4.2 shows the result of the PSO based or estimated time using PSO, the actual time and Magnitude of Relative Error (MRE) for time over ten completed projects.

**Table 4.2 Project Duration**

| **Project No** | **Actual Time (Days)** | **PSO based (Estimated time) (Days)** | **Time MRE** |
|---|---|---|---|
| **1.** | 63 | 62 | 0.0159 |
| **2.** | 92 | 91 | 0.0109 |
| **3.** | 56 | 57 | 0.0178 |
| **4.** | 86 | 84 | 0.0232 |
| **5.** | 32 | 31 | 0.0313 |
| **6.** | 91 | 93 | 0.0220 |
| **7.** | 35 | 34 | 0.0286 |
| **8.** | 93 | 91 | 0.0215 |
| **9.** | 36 | 37 | 0.0278 |
| **10.** | 62 | 59 | 0.0317 |

**Table 4.3 Computed values of MMRE**

| **Parameter** | **MMRE Value** | **MMRE %** |
|---|---|---|
| **Effort** | 0.1988 | 19.88 |
| **Time** | 0.2310 | 23.10 |

## BIOGRAPHY

**Manga Ibrahim: Is a lecturer with Adamawa State University, Mubi, Nigeria. He is a masters student in the department of computer science at Adamawa State University, Mubi, Nigeria. He is a member of Nigeria Computer society, Member IEEE. He obtained his Bsc in computer science in 2007 from Same University Mubi, Nigeria. His areas of research interest are Computational Intelligence Software Computer Algorithms.**

**Nachamada Vachaku Blamah :Is a Senior Lecturer with the University of Jos, Nigeria. He obtained his Bachelors of Technology, Master of Science, and Doctorate degrees in Computer Science. Dr. Blamah is a member of the IEEE Computational Intelligence Society and the Computer Professionals (Registration Council of Nigeria), and his research interests are mainly in the areas of computational intelligence and multi agent systems.**