Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Enhancement in Software Cost Estimation Using Ant Colony Optimization

[1]Shivani Sharma, [2]Aman Kaushik
[1]Research Scholar, [2]Assistant Professor
[1, 2] Department of Computer Science,School of Engineering and Emerging Technologies,
Baddi University of Emerging Sciences and Technology, Baddi (H.P), India

Abstract: ***The main challenge in the production and development of large and complex software projects is the cost estimation with high precision. Thus it can be said that estimating the cost of software projects play an important role in the organization productivity. With the increasing size and complexity of software projects the demand to offer new techniques to accomplish this important task increases day by day.Therefore, researchers have long attempted to provide models to fulfil this important task. Scale increasing in applications and a variety of programming languages using at the same time, manual measurement based on the LOC (Line of Code) cannot meet the estimating requirements. The emergence of Function Point resolves these difficult issues. Accurate cost estimation helps to complete project within time and budget. In this paper we are computing cost of project based on Top down approach in which we will compute function points of each module. The whole process will be done by using Ant colony optimization algorithm. To compare and evaluate the results of the proposed algorithm with K Modes algorithm and it has been observed that when we have compared with K modes then proposed work gives better results.***

Keywords: Software Effort Estimation, Software Cost Estimation, Software Cost Metrics, FP (Function Point), Top down estimation method, Ant colony optimization, K Modes.

I. INTRODUCTION

The increasing need for application software's and the growing size and complexity of these software's, requires an appropriate model for estimating the cost of software projects in software production organizations. Software's not being tangible and understandable in the early stages of production cause a low accuracy in cost estimation [1]. The success of a software project is determined by various factors that are related each other in the project. A project will be called a success if all the requirements can be fulfilled, the cost is not excessive (overflow), did not pass through the schedule (deadline) that has been planned. The accuracy of the management decisions will depend on the accuracy of the software development parameters. These parameters include effort estimation, development time estimation, cost estimation, team size estimation, risk analysis, etc. So, we need a good model to calculate these parameters. Sometimes, there is huge deviation between the actual cost and an estimated cost and hence accurate cost estimate is highly desired. The accuracy in the effort estimation is very essential because both underestimation and overestimation are harmful to the companies. [2]. If the estimate is too low then there will be pressure to complete the project as soon as possible. On other side if the estimate is too high, then more number of resources will be required. It implies that effort estimation is complex problem and it is very essential to investigate appropriate method for improvement of estimation accuracy [3].

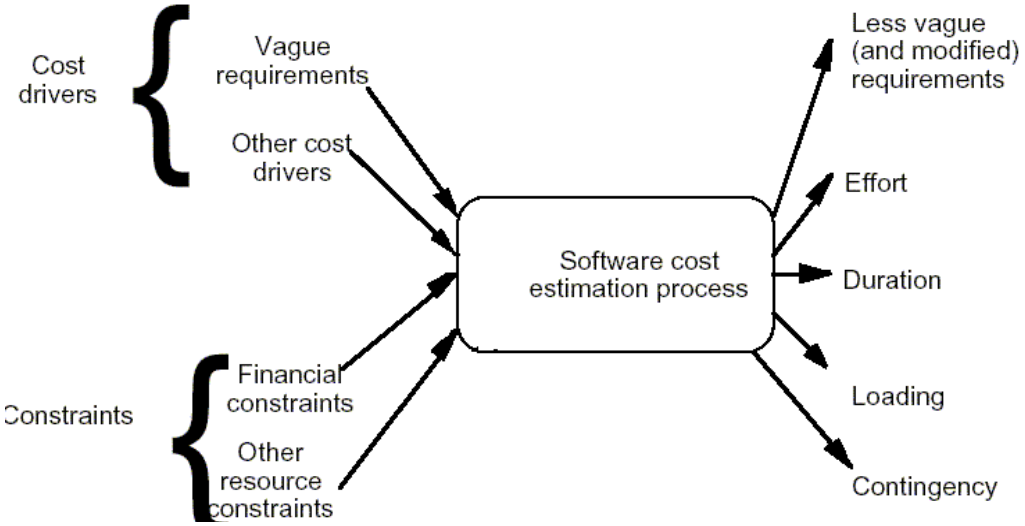

Fig.1 software cost estimation process

We have organized the paper as follows: Section 2, software cost metrics, function point estimation,Section 3, top down estimation method. Section 4 describes the related study by various researcher in this field, Section 5 describes Problem formulation and objectives, Section 6 describes Proposed Methodology. In the last conclusion of paper has described in section 7.

II. SOFTWARE COST METRICS

The continuous application of measurement based techniques to the software development process and its products to supply meaningful and timely management information together with the use of those techniques it improve that process and its products. Software metrics are all about measurements which in turn, involve numbers, the use of numbers to make things better, to improve the process of developing software and to improve all aspects of the management of that process.

LOC (LINES OF CODE)

- **Source Lines of Code (SLOC)**is software metric used to measure the size of software program by counting the number of lines in the text of the program's source code. This metric does not count blank lines, comment lines, and library. SLOC measures are programming language dependent. They cannot easily accommodate nonprocedural languages. SLOC also can be used to measure others, such as errors/KLOC, defects/KLOC, pages of documentation/KLOC, cost/KLOC.
- **Deliverable Source Instruction (DSI)** is similar to SLOC. The difference between DSI and SLOC is that "if-then-else" statement, it would be counted as one SLOC but might be counted as several DSI.
- LOC is presented as a measurement technique for quantifying the size of a software product. LOC is more of a measurement technique than a counting technique.[6,23]

FUNCTION POINT

Albrecht [4, 5] introduced the concept of function points as a software size measure while considering the more general problem of measuring application development productivity. His objective was to develop a software size measure that was independent of the implementation technology. To measure productivity, a measure of work product (or output) must be defined. For this Albrecht chose the function value to be delivered to the user. To calculate the function value delivered to the user, the number of inputs, outputs, inquiries, and files (including interfaces to other programs) from the user perspective are counted, weighted, and summed. This function count is then adjusted for the processing complexity of the application to produce a function point number. The function point metric (FP) proposed by Albrecht can be used effectively as a means for measuring the functionality delivered by a system using historical data. FP can then be used to estimate the cost or effort required to design, code and test the software, predict the number of errors that will be encountered during testing and forecast the number of components and/or the number of projected source lines in the implemented system.

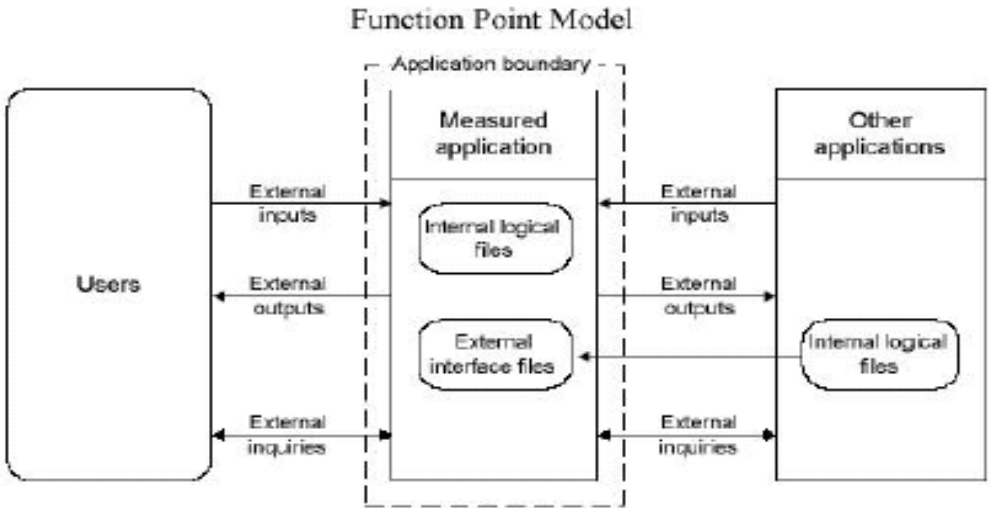

Fig. 2 function point model

- **External Inputs (EI)** These are end-user actions such as putting in a login or executing a mouse click.
- **External Outputs (EO)** The system provides the end-user output or interface such as a GUI display or items in a report.
- **External Enquiries (EQ)** This function is initiated by the end-user. For example, the end-user wishes to submit a query to a database or requests on-line help.
- **Internal Logical Files (ILF)** These files are the master or transaction files that the system interacts with during its session.
- **External Interface Files (EIF)** Unlike logical internal files, where the application uses solely for its purpose, these files are or databases are shared with other applications or systems.[6]

III. TOP-DOWN ESTIMATING METHOD

The top-down method of estimation is based on the whole uniqueness of the software project. The project is partitioned into lower-level components and life cycle phases beginning at the highest level. This method is more appropriate to early cost estimates when only global properties are known. Top-down estimating method is also called Macro Model. Using top-down estimating method, an overall cost estimation for the project is derived from the global properties of the software project, and then the project is partitioned into various low-level mechanism or components.The leading method using this approach isPutnam model. This method is more applicable to early costestimation when only global properties are known. In the earlyphase of the software development, it is very useful becausethere is no detailed information available.The top-down method is usually faster, easier to implement and requires minimal projectdetail. However, disadvantages are that it can be less accurateand tends to overlook lower-level components and possibletechnical problems. It also provides very little detail forjustifying decisions or estimates.[7, 8]

IV. LITERATURE REVIEW

Several works has been completed in the field of software size, cost estimation in software engineering and still lots need to be complete on it. Various researchers have suggested their work in this arena from that some of the most significant works are displayed in this section.

Jeffery, et al. (1993) have explored there has been a growing demand that the software development process be effectively managed, producing cost effective, reliable software within specified time constraints. Critical to the effective management of a software development is the ability of management to forecast the total development effort and cost. [10]

Zheng, Y., et al. (2009) have proposed an estimating method for software effort based on function point. It helps to estimate software effort more accurately without considering the languages or developing environment you choose. Firstly, use actual project records to obtain the linear relation between function point and software effort. Then determine the parameters of linear equations by maximum likelihood estimating method. [11]

Zeinab Abbasi Khalifehlou (2012**)** have presented SCE models based on data mining techniques for the choice of suitable AI techniques for essential effort in new projects. The techniques considered are LR, ANN, SVR and K-NN. That trained and tested instances are considered with these methods. [12]

Pauline et *al.,* (2013)have explored"An Enhanced model to estimate effort, performance and cost of the software projects", in this research authors have offered a layered model for the calculation of performance, effort and cost of given software. In this method authors discovered 3 simple but key software engineering metrics- function-points, lines of code, and person-months in the lower layer. [13]

Batra, Barua (2013)have Introduced the new approach is supportive for the reduction of cost and effort. Software development in this era is at demanding phase, and estimation of cost and effort in this filed always remains an open challenge and considered to be a complex task. Software engineering and SDLC have their significance presence in the estimation. [14]

Maleki, Isa, et al., (2014) have discussed one of the issues that helps the correct use of software and prevent the software projects from failure is the accurate estimate of cost. Accurate estimates helps the developmental companies that have better analyse of software projects feasibility and effectively manage software development process. [15]

Malathi and Lijin (2014)have designed "an Efficient Method of for the Estimation Effort in Software Cost", The proposed method is an integrated approach of combining analogy with the fuzzy, based on reasoning by analogy for handling both numerical and categorical variables where the uncertainty and imprecision solution is also identified by the behaviour of linguistic values utilized in the software projects. [16]

Zhang et al. (2014) have proposed BREM for software effort prediction and two embedded strategies for handling missing data. Inspired by the formulation of Bayesian regression and its superiority over other predictive models in machine learning, they combine Bayesian regression and EM algorithm to make full use of training and test data in constructing the predictive model. [17]

Primandari and Sholiq (2015) have presented the approach The Use Case Points (UCP) has been a method that often used as a reference to calculate effort estimation, the amount of worker and time required in software development project. Result of the effort estimation using UCP is the number of effort required to develop the software as a whole. [18]

Mittas et al. (2015**)** have designed a framework for comparing multiple cost estimation methods using an automated visualization toolkit. The purpose of this research is to present a framework for visualization and statistical comparison of the errors of several cost estimation methods. It is based on an automated tool which can facilitate strategies for an intelligent decision-making. [19]

Nilesh Choursiya et al. (2015**)**have evaluated to create good software required good software size estimation tool or method because software size is a most vital component. Hence, can enhance the accuracy of software by improving the exactness of software size estimation and it ultimately results in enhanced software effort estimates and software cost estimates. [20]

Balaji *et al.*,(2013) have introduced "Software Cost Estimation using Function Point with Non Algorithmic Approach", by Effort is a function of size. For estimating effort first face sizing problem.In their research the authors have offered a combined and efficient technique which utilizes both lines of code (LOC) and Function Points (FP) for estimation of effort of software project. [21]

V. PROBLEM FORMULATION

In this paper we are computing the cost estimation of the project based on Top down approach in which we will compute function points of each module. The whole process will be done by using Ant colony optimization (ACO) algorithm. In existing research they have developed an algorithm k-modes by which they are computing the cost of project based. But there is a drawback in their research that is algorithm taking more time to compute the cost of project to overcome this in our research work we are computing the cost of project based on Top down approach in which we will compute function points of each module . The whole process will be done by using Ant colony optimization algorithm.

OBJECTIVES

- To develop a highly optimized algorithm to calculate the cost of project based on function point in less time.
- Compare our research (ACO) with k modes algorithm.

SIMULATION ENVIORNMENT

In the current scenario for the implementation of proposed methodology Net Beans version 8.1 is used. Net Beans is a software development platform written in Java. The Net Beans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the Net Beans Platform, including the Net Beans integrated development environment (IDE). The Net Beans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++and HTML5.
The short list below enumerates some of its most prevalent features

- User interface management (e.g. menus and toolbars)
- Storage management (saving and loading any kind of data)
- Wizard framework (supports step-by-step dialogs)
- Net Beans Visual Library

PARAMETER FOR COMPARISON OF RESULTS

We have compared duration (run/ execution time) parameter ofproposed algorithm with K MODES algorithm in which we will compute function points. It has been observed that when we have compared proposed work with K modes then proposed work gives better results.

DURATION/ RUN TIME OF ALGORITHM

- This duration parameter is run/execution time of algorithm in which computing the start time and end time of algorithm.
    - Start time=System. Current time Millis ();
    - End time= System. Current time Millis () +20000;
    - End time – start time /1000;

VI. THE PROPOSED METHODOLOGY

This section discusses the methodology that was used for this paper. This work is conducted by applying Ant colony optimization algorithm. In this paper we are computing the cost of project based on Top down approach in which we will compute function points of each module. The whole process will be done by using Ant colony optimization algorithm.

ANT COLONY OPTIMIZATION

It was introduced for the first time by Marco Dorigo et.al in 1992 as a multi-agent solution with the aim of solving optimization problems like traveling salesman problem. An ant is a simple computational agent in the ant colony optimization algorithm. It iteratively constructs a solution for the problem at hand. The intermediate solutions are referred to as solution states. At each iteration of the algorithm, each ant moves from a state x to state y, corresponding to a more complete intermediate solution. When an ant moves alone and randomly, by facing a path with more pheromone effect, most likely chooses the path and strengthens it by leaving more Pheromone. [9]

Advantages of the Ant Colony Optimization

- Inherent parallelism
- Positive Feedback accounts for rapid discovery of good solutions
- Can be used in dynamic applications (adapts to changes such as new distances, etc.).

Disadvantages of the Ant Colony Optimization

- Theoretical analysis is difficult
- Sequences of random decisions (not independent)
- Probability distribution changes by iteration

STEPS FOR COUNT FUNCTION POINTS USING ANT COLONY OPTIMIZATION ALGORITHM

Length=vector array of Files; counter=0; i=0;

Step 1: Traverse from the first index of array (counter) using i Update visited node status for current node from 0 to 1.

Step 2: Set Dr is number of function exist in ith index of class then Apply Foreach (k=0 k< Dr) find the attributes in respected function, sum=sum+attributes.

Step 3: Update Pheromone stack of the selected function point get extracted from stack then after that we will empty that stack.

Step 4: k=k++

Step 5: If i<=length set visit status (v) of all node to 0, sum = 0, set current node = "start" and go to step 2.

Step 6: End of algorithm.

FLOW CHART OF COUNT FUNCTION POINTS USING ACO ALGORITHM

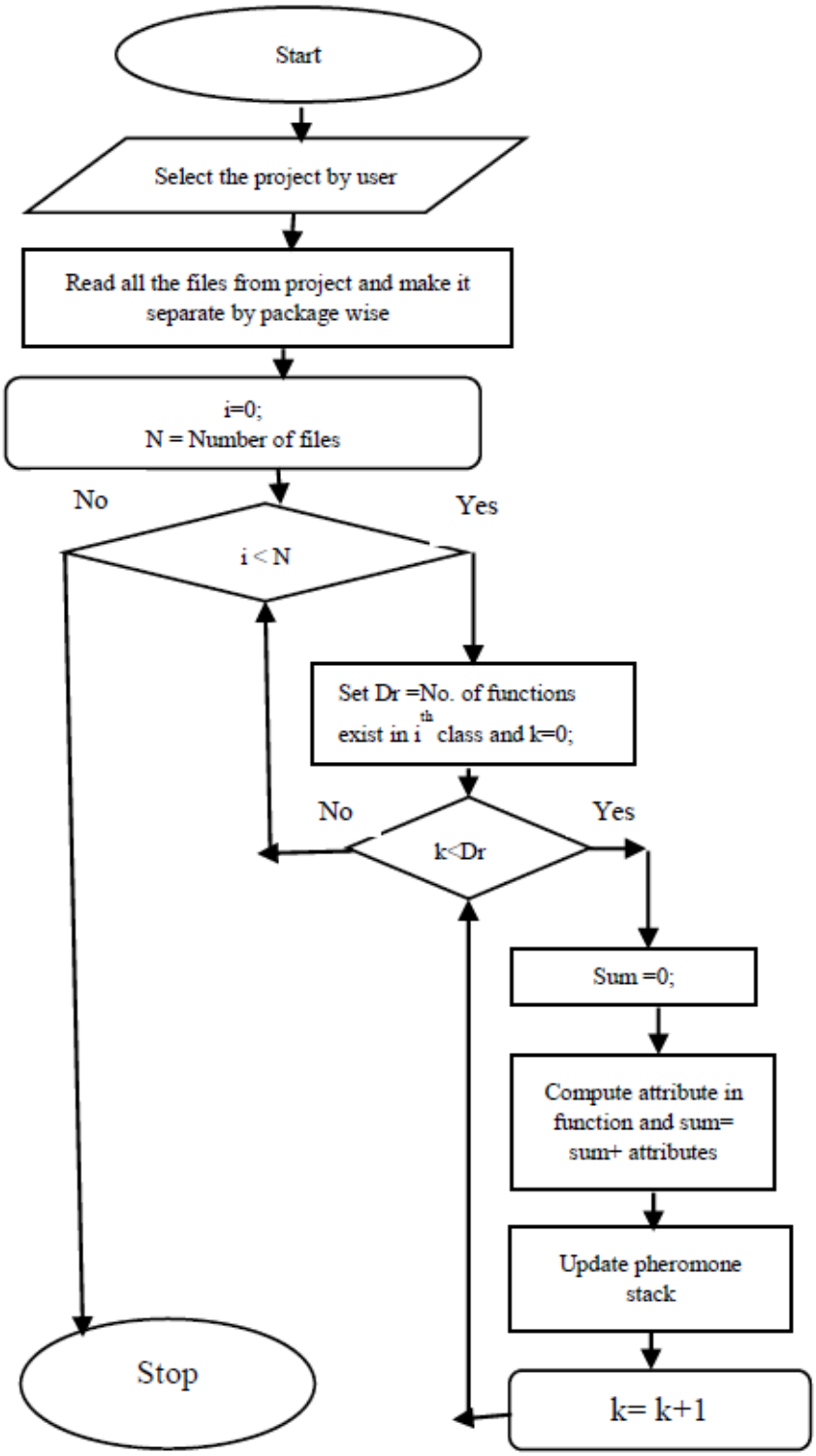

Fig. 3 flow chart of count function points via ant colony optimization algorithm

WE HAVE COMPARED OUR PROPOSED WORK WITH K MODES ALGORITHM

K MODES

The k-modes approach modifies the standard k-means process for clustering categorical data by replacing the Euclidean distance function with the simple matching dissimilarity measure, using modes to represent cluster centers and updating modes with the most frequent categorical values in each of iterations of the clustering process. These modifications guarantee that the clustering process converges to a local minimal result. Since the k-means clustering process is essentially not changed, the efficiency of the clustering process is maintained. In k-modes clustering, the cluster centers are represented by the vectors of modes of categorical attributes. In statistics, the mode of a set of values is the most frequent occurring value.

K-Modes Advantages

- If variables are huge, then K-Means most of the times computationally faster.
- K-Means produce tighter clusters than hierarchical clustering, especially if the clusters are globular.
- There are always K clusters.[22]

STEPS FOR COUNT FUNCTION POINT USING K MODES ALGORITHM

Let X = {x1, x2, x3… xn} be the set of files.

Step1: First set k=0, t=0;

Step 2: For each while i <x and for each k not equal to size D where D is number of function in each class

Step 3: Calculate the function point of function then k=k+1

Step 4: Sum=sum+Fp the data point to the cluster center (File).

Step 5: If i>x length return false.

Step 6: End of algorithm

FLOW CHART OF COUNT FUNCTION POINTS USING K MODES ALGORITHM

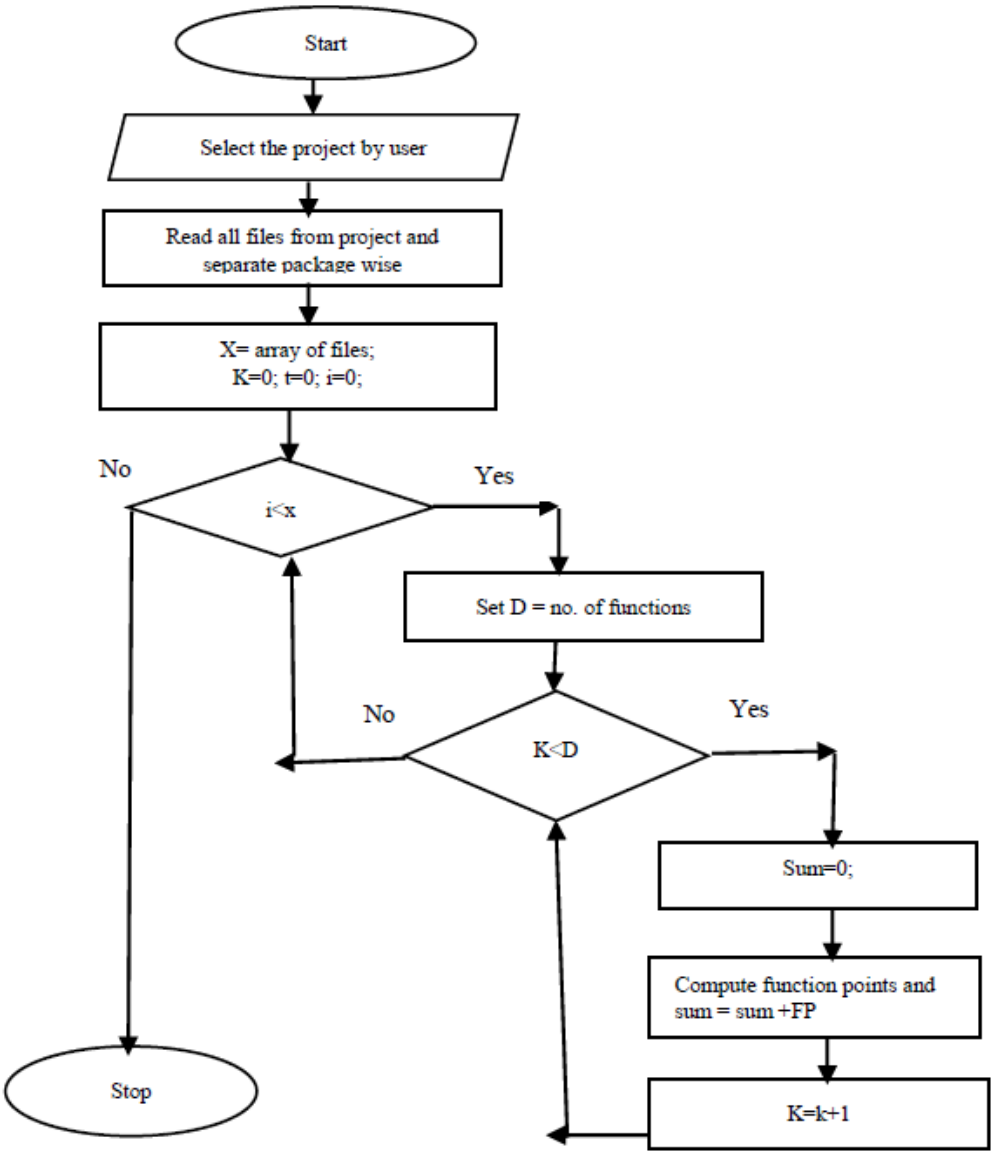

Fig. 4 Flow chart of count function points via K MODES algorithm

FUNCTION POINTS COUNT BY ANT COLONY OPTIMIZATION AND K MODES ALGORITHM.

We have taken six projects in which count function point for each module via ANT COLONY and K MODES Algorithm. Basically function point count by Ant colony optimization is showing algorithm taking less time to compute function point as compare K MODES algorithm.

Table 1 showing count function point results via ant colony optimization (ACO), K modes algorithm

FUNCTION POINTS

Sr. No.	ACO (PROPOSED ALGORITHM	K MODES ALGORITHM
PROJECT 1	2599	
PROJECT 2	7509	
PROJECT 3	277	
PROJECT 4	832	
PROJECT 5	57	
PROJECT 6	3112	

This table 1 showing count function point results via ant colony (ACO) and K Modes algorithm. We have taken six projects and count function point for individual project. Function point will remain same for ACO and K MODES.

RESULTS

We have plotted the six projects results basis on duration (execution time of algorithm) parameters. In which showing run /execution time of algorithm when compute function points by ACO and K MODES algorithm. Which is showing proposed algorithm taking less time as compared K MODES algorithm to count function points. It has been observed that when we compared with K modes then proposed work gives better results.

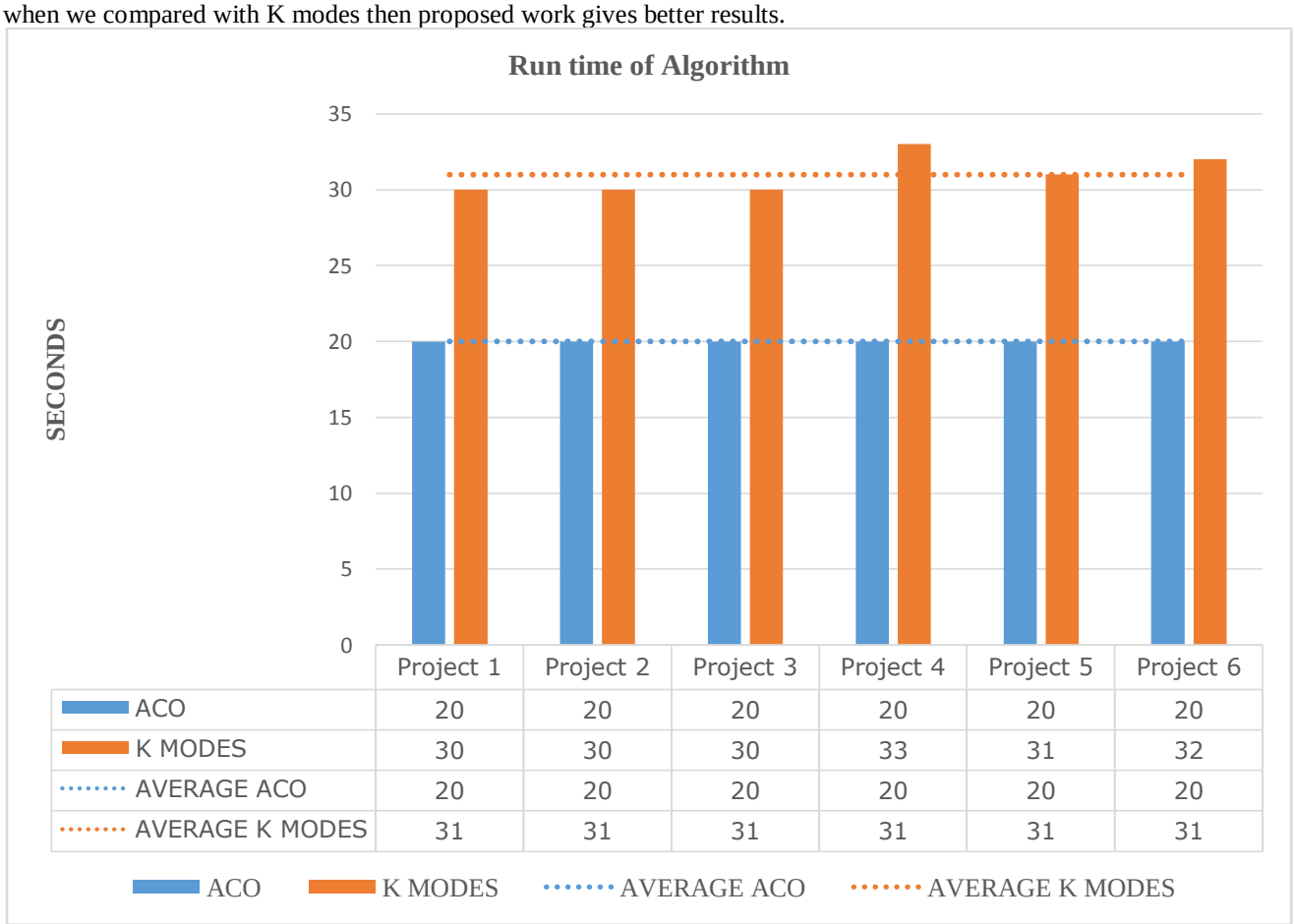

Fig. 5 Chart showing the run time of algorithm for ACO and K modes for six projects

This fig.5 is showing the duration (run time of algorithm) for six projects. And duration shows us the execution/runtime for all projects. This chart is showing result for ACO and K MODES in which average line showing how on an average the proposed ACO Algorithm is better than average of the other algorithm which is K MODES algorithm by which the average runtime in the project are identified. The average runtime by the proposed algorithm gives much improved results as in comparison with the other such technique and algorithm.

VII. CONCLUSION

The accurate estimation of software development costs and efforts is a critical issue to take good management decisions. The cost estimation problem Varies considerably among organisation that do their estimation under very different constraints. This paper deals with computing cost of project based on Top down approach in which we will compute function points of each module. The whole process will be done by using Ant colony optimization algorithm. Though it's difficult to figure out the count of function point, it will greatly simplify the process of softwareestimation, and then help the manager to have an accurate software effort measurement as well as cost measurement.We are predicting the cost estimation based on computing the function points in software project and in future scope of this paper we can use another algorithm to compute function point in place of Ant Colony Optimization and compute function point in less time. And can enhancement in software cost estimation process.

ACKNOWLEDGMENT

I would like to thanks Mr. Aman Kaushik and specially Department of Computer Science and Engineering for encouraging me to do this work and also the anonymous referees for their helpful comments and suggestions.

REFERENCES

[1] Kim G-H, an S-H, Kang K-I. Comparison Of Construction Cost Estimating Models Based On Regression Analysis, Neural Networks, And Case-Based Reasoning. Build Environ. 2004 Oct; Vol.39, pp.:1235–42.

[2] P.C. Pendharkar, "Probabilistic Estimation of Software Size and Effort", Expert Systems with Applications, Vol. 37, pp. 4435-4440, 2010.

[3] R. Agarwal, M. Kumar, S. Malick, R. M. Bharadwaj and D. Anantwar, "Estimating Software projects", Software Engineering Notes, ACM SIGSOFT, Vol. 26, No. 4, pp. 60–67, 2001.

[4] Allan J. Albrecht, "Measuring Application Development Productivity", Proceeding of the Joint SHARE/GUIDE/IBM Applications Development Symposium, pp. 83 – 92, 1979.

[5] -, AD/M Estimating and Productivily Measurement Guidelines, IBM Corp. Information Systems, 1984

[6] Pichai Jodpimai, Peraphon Sophatsathit, and Chidchanok Lursin- sap, "Analysis of Effort Estimation based on Software Project Models", IEEE, 2009.

[7] "COCOMO II Model definition manual", version 1.4, University of Southern California.

[8] Caper Jones, "Estimating software cost" tata Mc- Graw -Hill Edition 2007.

[9] [28] Dorigo M, Gambardella LM. Ant Colony System: A coop¬erative learning approach to the traveling salesman problem. IEEE Trans Evol Comput. 1997; 1 pp.:53–66

[10] Jeffery, D. Ross, Graham C. Low, and M. Barnes. "A comparison of function point counting techniques," IEEE Transactions on Software Engineering, Vol. 19, pp. 529-532, May 1993

[11] Zheng, Y., Wang, B., Zheng, Y., & Shi, L. "Estimation of software projects effort based on function point," Proceedings of 2009 4th International Conference on Computer Science & Education, IEEE. pp. 941-943, July 2009

[12] Khalifelu, Zeynab Abbasi, and Farhad Soleimanian Gharehchopogh, "Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation," Procedia Technology, pp. 65-71, Dec.2012.

[13] Pauline et al., "Comparison of available Methods to Estimate Effort, Performance and Cost with the Proposed Method", International Journal of Engineering Inventions, Volume 2, and Issue 9, PP: 55-68, May 2013.

[14] Batra, Geetika, Kuntal Barua, and M. Tech Scholar. "A Review on cost and effort estimation approach for software development," International Journal of Engineering and Innovative Technology Vol.3, pp. 290-293, October 2013.

[15] Maleki, Isa, et al., "Analysis of Software Cost Estimation Using Fuzzy Logic," International Journal in Foundations of Computer Science & Technology, Vol.4, pp. 27-41, May 2014.

[16] Malathi et al., "An Efficient Method for the Estimation of Effort in Software Cost," International Journal of Advance Research in Computer Science and Management Studies Volume 2, pp. 330-335, February 2014.

[17] W. Zhang, Y. Yang, Q. Wang, "Using Bayesian Regression and EM algorithm with missing handling for software effort prediction," Information and Software Technology, pp. 58-70, February 2015.

[18] Putu Linda Primandari A and Sholiq, "Effort Distribution to Estimate Cost in Small to Medium Software Development Project with Use Case Points," Procedia Computer Science, Vol. 72, pp.78 – 85, 2015.

[19] Mittas, Nikolaos, Ioannis Mamalikidis, and Lefteris Angelis. "A framework for comparing multiple cost estimation methods using an automated visualization toolkit," Information and Software Technology Vol. 57, pp.310-328, 2015.

[20] Nilesh Choursiya et al., "An Enhanced Function Point Analysis Method for Software Size Estimation," International Journal of Computer Science and Information Technologies, Vol. 6, pp. 2797-2799, 2015.

[21] Balaji et, al., "Software Cost Estimation using Function Point with Non Algorithmic Approach", Global Journal of Computer Science and Technology Software & Data Engineering Volume 13 Issue 8 Version 1.0 Year 2013

[22] Huang, Joshua Zhexue. "Clustering Categorical Data with k-Modes." (2009): pp. 246-250.

[23] Kiumi Akingbehin and Bruce Maxim, "A Three-Layer Model for Software Engineering Metrics", Proceedings of the Seventh ACIS International Conference on Software Engineering Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 17 – 20, 2006.

