OPTIMIZING NON-ORTHOGONAL SPACE DISTANCE USING ACO IN SOFTWARE COST ESTIMATION

S. RANICHANDRA
Asst.Professor, Department of Computer Science,
Dhanalakshmi Srinivasan College of Arts and Science for Women (Autonomous) , Perambalur.

ABSTRACT

This work proposes a technique to optimize the Nonorthogonal Space Distance (NoSD) based on the Ant Colony Optimization (ACO) algorithm so as to enhance estimation accuracy in analogy-based software cost estimation. NoSD is a measure of projects similarity that uses a matrix defined depends on mutual information to take both feature redundancies and feature weights into distance computation. We assume that such definition depends only on mutual information between features can hardly explain real-life software projects correctly, so we propose this new method and enhances NoSD using optimization techniques. In this proposed work, the matrix in NoSD is optimized by the ACO algorithm with the goal of reducing estimation error at training stage. Based on this optimized matrix, which well fits real-life software projects, then the distance definition can measure projects similarity more accurately and thus can greatly enhance the estimation accuracy. Experiments have been done on two real-life software projects datasets (Desharnais and ISBSG R8) using the proposed method along with some other widely used methods including Euclidean, Manhattan, Mahalanobis, Minkowski, NoSD, and weighted Euclidean distance. Experimental results show that this method brings notable improvements in estimation accuracy based on the broadly used evaluation metrics: MMRE, and PRED (0.25).

I.INTRODUCTION

One of the most efficient and significant factors in development of the software projects, is Software cost estimation. In growth of the software projects, there are many effective factors value and limits of which must be identified using the accurate estimation. The maximum costs of software projects and entirely the vast value of the costs are interrelated to the work forces. So, the most significant problem in software engineering is the cost estimation and the required effort for development of the software projects.

Software cost estimation is the process of estimating the effort required to develop software system. The important factor in selecting a cost estimation model is the accuracy of its estimates. It involves in the determination of a few estimates are cost. The target was to obtain an indepth effort, project duration and understanding of evaluation practice and to inspect factors that affect the accuracy of effort estimation. The fine accuracy in estimation can be obtained by developing the domain based helpful models that usefully explain the development life cycle and accurately estimate the effort of developing the software. Effort Estimation with good accuracy assists in managing overall budgeting and planning. The correctness of these estimates is very less and most complex to obtain, because no or very little detail about the project is known at the beginning [2].

Accurate effort estimates assist software consultancies to formulate appropriate bids when quoting for tenders, a lower estimate than the actual will direct to a loss and an unreasonably high estimate will lose the bid. Such estimation models are developed using a set of measures that illustrate the software development process, product and resources such as developer experience, system size and difficulty and the characteristics of the development environment, correspondingly [1].

The technique proposed in [3] makes feature weights using particle swarm optimization (PSO).Also Non-orthogonal Space Distance (NoSD) extends the usually used distance into a non-orthogonal space. It enhances normal distances by using a matrix which is defined based on mutual information between features to characterize the non-orthogonal space. However due to the software projects complexity, results are frequently less optimal. So this work proposes an ACO-based method to optimize NoSD matrix so as to further improve the cost estimation accuracy.

II.RELATED WORKS

Analogy-based software cost estimation is initially proposed by Shepperd and Schofield in [4] and afterward been accepted as one of the frequently used methods for cost estimation. It consist numerous important steps including data preprocessing, feature selection, case selection, and case adaptation. Case selection is a crucial part of this entire process and similarity distance is the core of this step.

Estimation by analogy [22] is the method where the proposed project is compared with previous projects of same nature with sample information in project development. The benefit of analogy method [23] in comparison to other methods is that analogy is depending on actual experience. But, it is not very effective [24] due to non existence of similar projects and the accuracy of accessible historical data. In [25], genetic algorithm is applied in analogy to reduce the time involved while selecting the historic projects. Liu et al [26] has proposed a statistical framework for the removal of noise and achieve improvement of results in analogy method.

Numerous researchers have used soft computing approaches for software cost estimation. Idri et al have implemented the COCOMO cost model using fuzzy logic in [5] and also a fuzzy logic based analogy estimation approach in [6-8].

Case based reasoning has also been utilized by Kadoda et al in [9] they study the problem of the choice of number of analogies when making predictions. They also look at diverse adaptation strategies. The analysis is depends on a dataset of software projects collected by a Canadian software house. Their results show that selecting analogies is important but adaptation strategy appears to be less so. For this reason they urge some degree of caution when comparing competing estimation systems and only modest numbers of cases.

Myrtveit et al in [10] and Ganesan et al in [11] have also examined case based approach to development effort prediction. Bhattacharjee et al have proposed Expert Case Based Models in [12- 18].

III.ANALOGY-BASED ESTIMATION

Estimation by Analogy offered the best prediction accuracy when using a dataset of Web hypermedia applications. Estimation by Analogy is a spontaneous method and there is a fact that experts apply analogic reasoning when making estimates. Estimation by Analogy is effortless and flexible, compared to algorithmic models. Estimation by Analogy can be utilized on qualitative and quantitative data, showing closer types of datasets found in real life [19].

Analogy-based estimation is capable to deal with poorly understood domains that are complex to model, since it relies on historical data of projects related to the target project rather than on a formal model, or rules in the case of rule-based systems. It can be applied in the very premature phase of a software project when detailed information about the project is not yet accessible, and can be later improved when more detailed information is available. Analogy-based estimation has the probable to mitigate the effect of outliers in a historical data set, since estimation by analogy does not rely on calibrating a single model to fit all projects [20]. Ant colony optimization (ACO) use a population of agents or individuals to represent solutions, and the information gathered by the population influences the next generation of the search.

ESTIMATION PROCEDURE

There are several main steps in the analogy-based software cost estimation technique. The entire procedure can be illustrated as follows

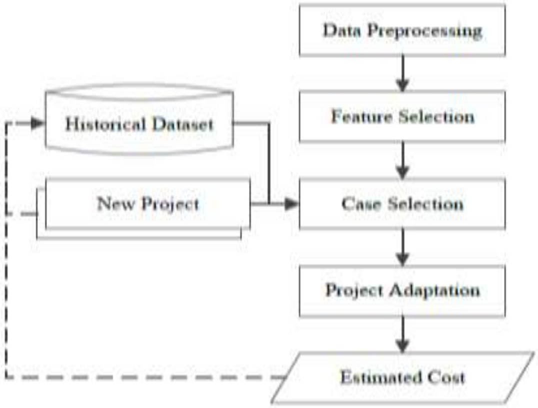


Fig1: Estimation Process of Analogy-based Software Cost Estimation

Data Preprocessing

It is the first step of the entire process. As the dataset collected might consists a lot of incomplete values, unrelated information or, complex data types, it is essential to do a preprocessing to clean up the raw data. This frequently includes removing the missed values, recoding complex types and standardizing continuous features and so on.

Feature Selection

This is a step to choose the most suitable feature subset from all the available features. As not all the features in the collected dataset are well related with the cost estimation and a few unrelated features may have bad effects on the estimation accuracy, it is essential to select a set of most related features out of the whole features. Features can be selected manually by users or experts own judgments depend on their experiences. It is also probable to adopt more sophisticated feature selection methods to select the suitable features automatically.

Case Selection

It is also sometimes called project selection as cases at this point just refer to the software projects. Case selection is a procedure of choosing the most similar past projects from the historical software projects dataset when a new project is to be predicted. It is one of the most crucial steps in the entire analogy-based estimation process as the estimation accuracy is highly influenced by the similar projects retrieved. In the case selection step, similarity distances are used to determine the similarity between projects and thus measure the similar projects selected. Therefore the similarity distance is the core of this step.

Project Adaptation

It is a step of calculating the estimated cost of a new project depends on the similar historical projects. After similar historical software projects are chosen in the previous step, these similar projects and their costs are input into a solution function which will output the predicted cost based on the input. There are numerous different solution functions and the most broadly used ones are median, mean and inverse weighted distance mean.

In common, analogy-based software cost estimation works with the beyond steps.

Data preprocessing and feature selection are used mostly to cleanup and process the data for later estimation and they are comparatively independent of the cost estimation process. Case selection and project adaptation are the crucial parts of this technique. Finally, when a new project is completed, the estimation accuracy can be assessed.

DISTANCE MEASURES

First, the basic Euclidean distance is the most broadly used similarity distance in software cost estimation. It is so much adopted owing to its simplicity and transparency. Likewise, Manhattan distance and Chebyshev distance are variants of the Euclidean distance and they have also been utilized in many studies. Basically, they are all the unique cases of the Minkowski distance. These distances are basically similar and their cost estimation accuracies are also similar. Their definitions are quite easy and they have fine computation efficiency. But the disadvantages are that they too much simplify the problem and make some assumptions that are not aligned with original situation. On the one hand, they assume that all features are of same importance but in fact different features have different impact on cost estimation.

On the other hand, they assume that features to be independent to each other. But the real case is that most features are highly associated with each other. The weighted distance resolves some of the problems of the Euclidean distance. By adding up weights to project features, weighted distance gives features different weights in distance computation based on their importance to cost estimation. This assists reduce noises in collected data and can enhance estimation accuracy. But it also has some difficulties. Similar to Euclidean distance, weighted distance also assumes features to be independent with each other which remain a difficulty to solve. Also, assigning of feature weights becomes another problem and the estimation results might be seriously affected by the different dataset distribution. Mahalanobis distance utilizes the inverse of covariance matrix to integrate data distribution into distance computation. It no longer considers project features to be independent which is more closed to real life situation. But feature weights are not well considered. Besides, the use of covariance matrix can help in troubles like outlier detection but it is not so suit for the software cost estimation problem. To manage with these problems, this study proposes a non-orthogonal space similarity distance (NoSD). It defines feature weights using mutual information between each feature and to-be-estimated cost and defines the features non-orthogonal relationship using mutual information among features. Finally these two factors are joint into a matrix and the matrix is used in the distance computation to assist measure projects similarity more accurately.

IV.NON-ORTHOGONAL SPACE DISTANCE BASED COST ESTIMATION

NoSD is an enhanced similarity distance that extends the introduced Euclidean distance, weighted distance, Manhattan distance, and Mahalanobis distance. It tries to resolve the problems of these distances and offers a non-orthogonal approach of measuring distance between software projects. This makes it illustrate real life software project similarity better.

The main assumption of this non-orthogonal space distance is that it assumes that in real life software projects, project features are typically correlated with each other rather than independent and also project features are of different significance to each other. The current methods typically make a simplification of the model and assume either software projects to be independent variables or to have same importance. So to enhance the estimation accuracy, we required to make the similarity distance more closed to real life situation.

A.ACO-BASED NoSD OPTIMIZATION

For the goal of enhancing estimation accuracy in software cost estimation problem, the major challenge lies in the complexity of real-life software projects. We assume that NoSD, which is based only on mutual information between features, is inadequate to be optimal due to the real projects difficulty. Thus this ACO-based NoSD optimization method is proposed to formulate the distance definition better fit real situation. The fundamental idea of this ACO-based method is to use ACO algorithm to optimize the NoSD matrix so as to adapt the distance definition to different software projects dataset. Based on the initial matrix offered by NoSD method, ACO is used to train and optimize the matrix with the goal of minimizing the estimation error. Then the optimized matrix can better illustrate projects similarities in different kinds of real-life projects and thus can significantly improve estimation accuracy.

B.ANT COLONY OPTIMIZATION

ACO is one of the most well-known meta-heuristic algorithms. ACO algorithm was first offered by Dorigo in 1996 [21]. ACO algorithm is motivated from the natural life of the ants. Ants leave a scented stuff on the path named pheromone. This stuff evaporates however is left in short time as the ant trace on the earth. The ants are able to generate pheromone to find the nearest path to the food. The ants choosing the nearest path create more pheromone than the ones selecting the longer paths. As the more pheromone attracts various ants, the more and the more ants will choose the shorter path and then all ants will locate the shorter path and move on it. To examine the subject more, we assume that there are two paths on the way to the food which are dissimilar in length. The ants choose both paths by the same probability. The ants which have been on the shorter path will generate the most pheromone before the others. So the other ants will choose this path and will enhance the pheromone of this path. At last all ants will locate the shortest path to the food.

C.ESTIMATION PROCEDURE

The default analogy-based software cost estimator is a type of lazy learner and there is no preparation required before it starts to estimate the cost of a new software project. But in this case, the ACO-optimized similarity distance ACO-NoSD requires to be obtained as described in the optimization process. The optimization procedure makes the analogy-based cost estimator an active learner. Therefore the whole estimation procedure needs to be adjusted.

The historical software projects dataset is divided in to two subsets which are the training set and the testing set. Training set is utilized to train and optimize the estimator parameters. The validation set is used for fine-tuning estimator parameters. The testing set contains the software projects that required to be estimated and it will also be used evaluate the final estimation accuracy. As the estimator wants to actively optimize the similarity distance before estimating cost of new projects, a training process wants to be added to the default analogy-based estimation process. Therefore the estimation procedure can be chiefly divided into two stages which are the training stage and the testing stage.

Training Stage

This stage is utilized to train and to attain the definition of the ACO-optimized distance ACO-NoSD based on the training set. First, the NoSD matrix is estimated .Then ACO algorithm is utilize to iteratively optimize the matrix. For each iteration of the optimization process, a candidate solution can be got and thus a candidate similarity distance can be obtained according to the ACO-NoSD formula. Then the estimation error can be computed dependent on the training set and the obtained candidate similarity distance. The ACO algorithm iteratively optimizes the distance matrix with the aim of minimizing this estimating error and at the end returns an optimized matrix and this will be utilized in the optimized NoSD for cost estimation.

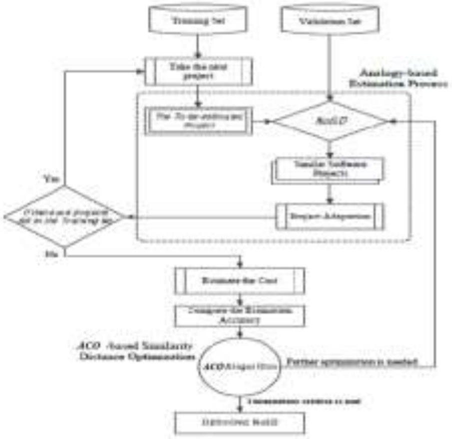

Training Set
Validation Set
Take the next project
Analogy-based Estimation Process
The To-be-estimated Project
NoSD
Similar Software Projects
Project Adaptation
Yes
If there are projects left in the Training set
No
Estimate the Cost
Compute the Estimation Accuracy
ACO -based Similarity Distance Optimization
ACO Algorithm
Further optimization is needed
Termination criteria is met
Optimized NoSD

Fig2: The training stage of the ACO-NoSD based estimation procedure

Testing Stage

This stage is intended at estimating software project cost and computing the cost estimation accuracy. The testing stage is much effortless compared with the training stage. It is related with the default analogy-based software cost estimation process except that here the ACO-optimized distance will be used instead of the default one. Based on this optimized matrix, the testing stage evaluates costs of testing set projects and finally the estimation accuracy can be determined.

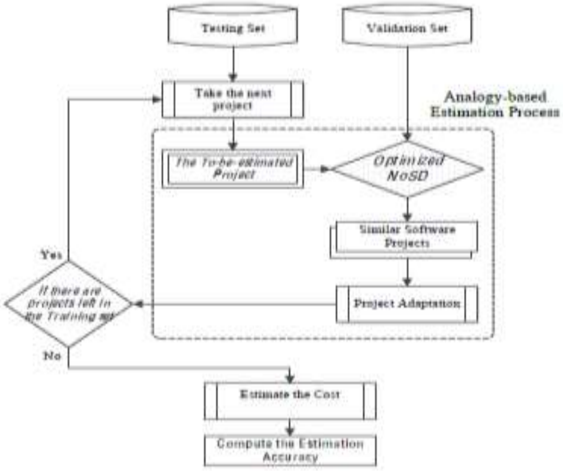
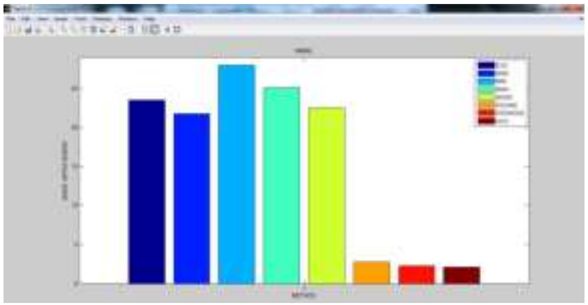

Testing Set
Validation Set
Take the next project
Analogy-based Estimation Process
The To-be-estimated Project
Optimized NoSD
Similar Software Projects
Yes
If there are projects left in the Training set
Project Adaptation
No
Estimate the Cost
Compute the Estimation Accuracy

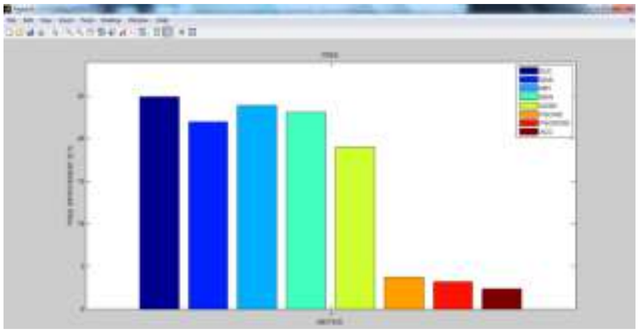
Fig3: The testing stage of the ACO-NoSD based estimation procedure

V.EXPERIMENTAL RESULTS

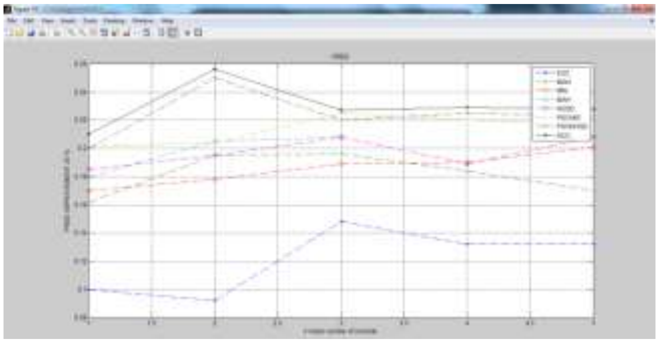
On the whole comparison, the results of our proposed ACO-NoSD is compared with other methods based on the average value of MMRE, PRED and the percentage of improvements of MMRE and PRED, the proposed ACO-NoSD performs well than all other tested methods. Particularly for those without optimization process, the developments are significant. The improvements of the proposed ACO-NoSD over other methods are also examined and the percentages of enhancement for each method in MMRE, PRED are shown. The experiment results on ISBSG R8 and Desharnais Dataset are illustrated as follows



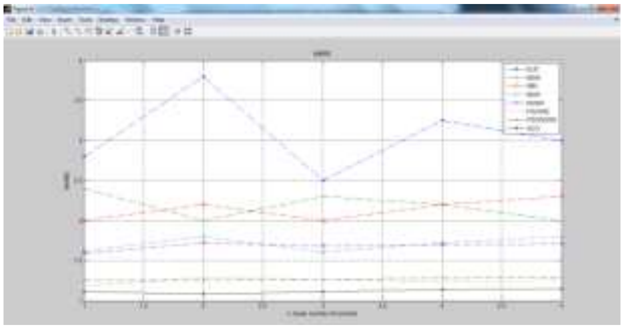
MMRE improvements on Desharnais dataset



PRED improvements on Desharnais dataset



PRED results in ISBSG R8 dataset



MMRE results in ISBSG R8 dataset

The evaluation criteria MMRE, and PRED are computed for each tested method on each dataset. As MMRE show the estimation error and PRED shows the accuracy, larger PRED and smaller MMRE mean fine results. This shows that optimizing the non-orthogonal space matrix has the capability of achieving better estimation accuracy than optimizing only the feature weights. Optimization-based methods show much improved results than others and ACO-NoSD shows higher estimation accuracy in most cases.

VI.CONCLUSION

This study mostly based on the similarity distance in analogy-based software cost estimation. The current correlated works are studied. A new non-orthogonal space similarity distance is projected and further an ACO based non-orthogonal space distance optimization method is introduced. Experiments are done and results show that the proposed similarity distance can be a better choice for analogy-based software cost estimation since it can help considerably improve estimation accuracy.A key research of the related works in analogy-based software cost estimation is conducted. Following the study of Euclidean distance, Minkowski distance, Manhattan distance, weighted Euclidean distance, and Mahalanobis distance, the Non-orthogonal Space Similarity Distance is proposed in this study. NoSD assumes software project features together to be dependent with each other and to have dissimilar importance in the computing the distance between software projects. Based on NoSD, more similar historical projects can be recognized in the case selection step and thus the final estimation accuracy can be enhanced. Based on this study another ACO-NoSD (ACO Optimized NoSD) method is proposed in this study. ACO (ant colony optimization), a typical space search problem optimization algorithm, is adopted in ACO-NoSD to optimize the matrix in the distance definition. After the training process, the optimized matrix is used in the NoSD distance and the resulted similarity distance can assess software projects similarities better. ACO-NoSD helps the similarity distance to be more adapted to real life software project datasets and can explain the data distribution better. Thus it can further increase the estimation accuracy in analogy-based software cost estimation. The proposed NoSD and ACO-NoSD method can solve some of the existing problems in the analogy-based software cost estimation. The experimental results show that this non-orthogonal space similarity distance can be a good alternative to the currently used similarity distances such as Euclidean distance in cost estimation as it offers the possibility of further improving estimation accuracy.

REFERENCES

[1]. K. Vinay Kumar, V. Raves, Mahil Carr and N. Raj Kiran," Software development cost estimation using wavelet neural networks," The Journal of Systems and Software, 2008.

[2]. Kirti Seth, Arun Sharma and Ashish Seth," Component Selection Efforts Estimation– a Fuzzy Logic Based Approach," International Journal of Computer Science and Security, (IJCSS) Vol. 3, No.3, pp. 210-215, 2009.

[3] Vahid Khatibi Bardsiri,Dayang Norhayati Abang Jawawi,Siti Zaiton Mohd Hashim,Elham Khatibi, “A PSO-based model to increase the accuracy of software development effort estimation” ,Software Quality Control .

[4] M. Shepperd and C. Schofield, "Estimating software project effort using analogies," Software Engineering, IEEE Transactions on, vol. 23, no. 11, pp. 736--743, 1997.

[5] A. Idri, L.Kjiri, and A Abran. (2000), “COCOMO Cost Model Using Fuzzy Logic”, In Proceedings of the 7th International Conference on Fuzzytheory and Technology, pp.219-223. Atlantic City, NJ, USA.

[6] A. Idri and A Abran. (2000b), “Towards A Fuzzy Logic Based Measures for Software Project Similarity”, In Proceedings of the 6th Maghrebian Conference on Computer Sciences, pp. 9-18, Fes Morroco.

[7] A. Idri and A. Abran. (2001), “A Fuzzy Logic Based Measures For Software Project similarity: Validation and Possible Improvements”, In Proceedings of the 7th International Symposium on Software Metrics, pp. 85-96, England, UK, IEEE.

[8] A. Idri, A. Abran and T.M. Khoshgoftaar. (2001c), “Fuzzy Analogy: A new Approach for Software Cost Estimation”, In Proceedings of the 11th International workshop on software Measurements, pp.93-101, Montreal, Canada.

[9] G.Kadoda, M Cartwright, L Chen, and M.shepperd.(2000), “Experiences Using CaseBased Reasoning to Predict Software Project Effort”, In Proceeding of EASE, p.23-28, Keele,UK.

[10] I. Myrtveit and E. Stensrud. (1999), "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", IEEE transactions on software Engineering, vol 25,no. 4, pp. 510-525.

[11] K. Ganeasn, T.M. Khoshgoftaar, and E. Allen. (2002), "Case-based Software Quality Prediction", International journal of Software Engineering and Knowledge Engineering, 10 (2), pp. 139-152.

[12] S. Kumar and V.Bhattacharjee,(2007),"Analogy and Expert Judgment: A Hybrid Approach to Software Cost Estimation", In Proceedings of the National Conference on information Technology: Present practice and Challenge, Sep'07, NewDelhi, India.

[13] V. Bhattacherjee and S. Kumar,(2004),"Software cost estimation and its relevance in the Indian software Industry", In Proceedings of the International Conference on Emerging Technologies IT Industry, Nov'05, PCTE, Ludhiana India.

[14] V. Bhattacherjee and S .Kumar,(2006),"An Expert- Case Based Frame work for Software Cost Estimation", In Proceedings of the National Conference on Soft Computing Techniques for Engineering Application (SCT-2006), NIT Rourkela.

[15] V. Bhattacherjee,(2006),"The Soft Computing Approach to Program Development Time Estimation In Proceeding of the International Conference on Information Technology, ICIT 06, Dec'06, Bhubneshwar, India, IEEE Computer Society.

[16] V. Bhattacherjee, S .Kumar and Ekbal Rashid (2008), "Estimation of Software Development Effort in University Setting: A Case Study", Accepted for Presentation at National Conference on Architecturing Future IT Systems (NCAFIS '08), Indore, October 2008.

[17] V. Bhattacherjee, S .Kumar and Ekbal Rashid (2008), "Case Based Estimation Model using Project Feature Weights", In Proceedings of The National Seminar on Recent Advances on Information Technology (RAIT-2009) ISMU, Dhanbad.

[18] V. Bhattacherjee, S. Kumar and E. Rashid, (2008),"A Case Study on Estimation of Software Development Effort.", Accepted for presentation and publication in ICACT-2008 Conference proceedings, Gokaraju Rangaraju Institute of Engineering & Technology, Hydrabad. (A.P.), India.

[19] R. Belbin, Team Roles at Work, Butterworth- Heinemann, Oxford, 1983

[20]Jingzhou Li, Guenther Ruhe, Ahmed Al-Emran and Michael M. Richter, "A flexible method for software effort estimation by analogy", Journal Empirical Software Engineering, Vol. 12, No. 1, pp.65-106, Feb 2007.

[21] M. Dorigo, V. Maniezzo, A. Colorni, "Ant system: optimization by a colony of cooperating agents", IEEE Trans. on Systems, Man and Cybernetics, Part B, Vol.26, No.1, pp.29-41, 1996.

[22] Wai, J., B. Keung, A. Kitchenham and D.R. Jeffery, 2008." Analogy-X: Providing statistical inference to analogy-based software cost estimation ", IEEE Transactions Software Eng., Vol. 34, pp. 471-484.

[23] Jianfeng Wen, Shixian Li, Linyan Tang, 2009."Improve Analogy-Based Software Effort Estimation using Principal Components Analysis and Correlation Weighting," 16th Asia-Pacific Software Engineering Conference.

[24] Joon-kil Lee, Ki-Tae Kwon, 2009. " Software Cost Estimation using SVR based on Immune Algorithm," 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing

[25] Y.Li, M. Xie, and T.Goh, 2009. "A study of project selection and feature weighting for analogy based software cost estimation, "Journal of systems and software, vol.82, pp.241-252.

[26] Q. Liu, W.Z. Qin, R. Mintram, M. Ross,2008. " Evaluation of preliminary data analysis framework in software cost estimation based on ISBSG R9 data," Software quality journal, 16(3): 411-458.

FORMAT FOR BIOGRAPHIES

Each author may include his or her biography at the end of the paper. The paragraph begins with the author's name (font size 9 point, bold, indentation of 0.125" space).

First Author personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.

Second Author personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.

Third Author personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.