Volume 5, Issue 6, June 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Bees Algorithm – A Tool for Case Based Software Effort Estimation

S. Bhavani
Assistant Professor (Senior)
VIT University, Vellore, India

Abstract— ***Software effort estimation has been one of the major challenge tasks in software engineering and has achieved a considerable interest from both industry and the research community. The case-base consists of several of software projects which contains the target case for which the effort has to be calculated and the history of successful projects. The analogy that has to be compared between different projects has to be found out. This paper proposed a method to retrieve project effort and find optimal number of analogies using project management activity network and bees algorithm. The bees algorithm will be used to search for analogies and features values that will be used to reduce the estimate errors.***

Keywords: Case-Base, Software Effort Estimation, Bees Algorithm

I. INTRODUCTION

One of the major challenge tasks in software engineering is software effort estimation and it has been one of the interest from both researchers and interest. So it maintains its dignity and been the soft cornered topic in software engineering. The importance of software effort estimation is because of its feasibility study, project bedding during software development for progress monitoring,risk evaluation and resource allocation. The most investigated models for solving the software effort estimationproblem is Machine Learning based prediction algorithms. Among them, Case-Based Reasoning has achieved reasonable identity because of its performance in prediction. CBR is a knowledge management technology(KMT) based on the premise that history almost repeats with itself which leads to the problem solving that can be based upon retrieval by similarities [1]. CBR has been used for several software engineering problems to solveill-defined problems or non-trivial or such as those identified by project managers including its support for software project management in predictions and lesson being learned [1]. The Figure1 illustrates the process of case based reasoning in software effort estimation. The case base consists of several numbers of cases described by a bunch of features that is divided into the two parts: problem description and the solutions. The problem description is a set of features whereas the solution can bedetermined by the effort required to accomplish the software project. However, software effort datasets that are characteristically and dramatically noisy datasets and case based reasoning methods which are more capable of handling the noisy datasets than the regression based technologies. But, like other machine learning methods, the performance of case based reasoning is dataset dependent so it has enormous space of configuration possibilities and the design options which are induced for each and every individual datasets [2]. So it is least surprise to see contradictory results and different performance figures when developing this model. Such design methodologies have much stronger impact on its accuracy and reliability of case based reasoning. The adjustment method is one of the most important parameters on case based reasoning as it fits the case in hand and minimizes the variations and deviations between a new founded case and retrieved historical cases. The adjustmentmethodology requires some of the parameters to be set such as number of analogy (i.e. Retrieved similar solutions) and the methods that are used to make the adjustment. This paper claims that, avoiding sticking to a fixed best number of performing analogies that should change from datasets to datasets which varies with the retrieved values. We can also use an alternative method to adjust case based reasoning by optimizing the number of analogies and features similarity coefficients.

This paper employs the Bees algorithm (BA) to optimize number of analogy (*K*) and the weights that are used to adjust the features similarities values between the new case and the other *K* analogies. The Bees Algorithm employs the model of population based search algorithm, which was first described inthe year of 2005. This algorithm depicts the food foraging and searching behavior of swarms of honey bees. In its zeroth version, the algorithm employs a kind of neighborhood search which will be later combined with the random search and can be used for analogy optimization. This present paper calculatesthe effect on the improvement in the effort estimation accuracy in case based reasoning when the bees algorithm method is employed to find out the optimal number of analogy as well as the coefficient values which can be used to adjust the retrieved project efforts. With respect to our knowledge the bees algorithm has not yet been used in software effort estimation and this paper depicts its significance to provide more reliable results.

II. SYSTEM OVERVIEW

Case based software effort estimation makes a newprediction for a newly started project by retrieving the previously successful completed projects that has been figured out and remembered as successful historical projects. Even though case based reasoning shows the reliable performance figures for high rated datasets, it is still being suffered from local performance tuning problems when they are to be applied over another environment.

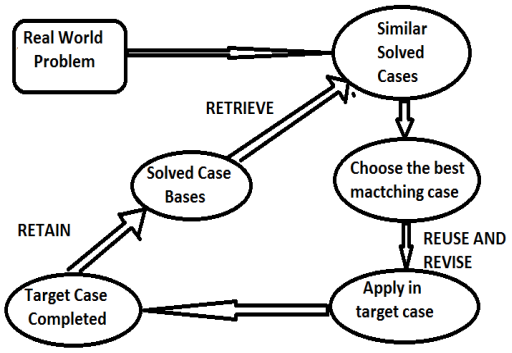

Fig.1. Case Based Reasoning- The overview process.

Finding appropriate analogies which fits the procedure of adjustment and depicts the dataset characteristics is done by local tuning, where this process is a hectic process on its own kind of environment. The application characterizes the project with input as project tasks with duration. The Project Management Activity Network will output the three dimensions such as duration, distance and Fitness value. Apart from this it also gives the critical path value for the given input project. From this three dimensional outputs the Bees Algorithm will find the analogy. Using the various information provided the user can make the analysis.The memory used for this application must be in order that it is used in an optimized way. The various other applications might also be running in the system, it must see that the application developed takes a less time and uses the memory available in the system in an optimized way that it does not slow down the process of the other applications.

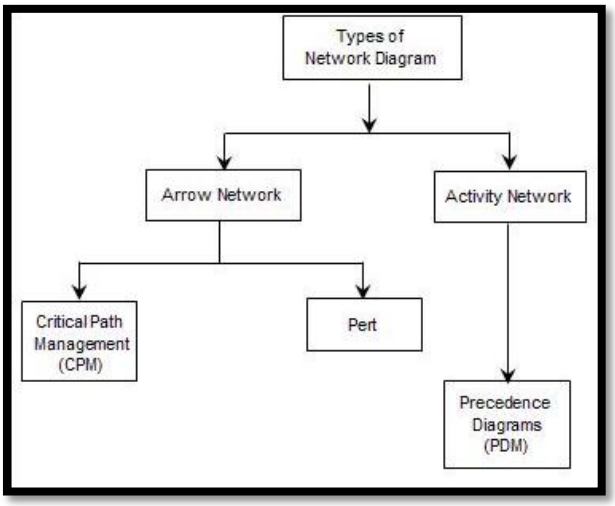

Fig.2. Activity of project management activity network

III. MODULAR DESCRIPTION

A. Proposed Method Of Case Based Reasoning

The adjustment of case based reasoning is a technique which is used to derive a newly founded value by minimizing the gap between the differences between the retrieved number of analogies and close target project for which the effort estimation is required to be calculated. Now It must be noted that this is an important and foremost step in Case based effort estimation as it showsthe structure and strength of analogies on the target case project. This technique should be represented mathematically as a function formula that identifies the differences between problem description of the identified target project and its number of analogies and also it shows the description in an attempt to generate the more reasonable and reliable solution set. In Case based effort predication the function is formulated as shown in Eq. 1:

$$Effort(p_t) = F(p_1, p_2, p_3, \cdots, p_K) \quad (1)$$

wherePt is the project used for target and P1 to PN is the top most familiar K similar projects close to the target one. The similarity degree is calculated using Euclidean distance. F is the adjustment function procedure that is used to view the differences between Pt and all other top similar projects, and then convert these identified differences into the amount of changes in the effort value set. The adjustment function shown above is illustrated in equations 2 and 3, where wj is the optimization coefficient value. The proposed case based reasoning method exploits the searching capability of the bees algorithm to overcome the local tuning problems of effort estimation adjustment. More specifically, the job is to search for appropriate number of weights (wj) and K number of values such that the difference between the performance measures is minimized.

$$Effort(p_{ti}) = Effort(p_i) + \sum_{j=1}^{M} w_j \times (f_{tj} - f_{ij}) \quad (2)$$

$$Effort(p_t) = \frac{1}{K} \sum_{i=1}^{K} Effort(p_{ti}) \quad (3)$$

where M is the feature number, ftj is the jth feature value of the close target project. fij is the jth feature value of the analogy project Pi.

Before initiating, the bees algorithm parameters must be set carefully , these parameters are as follows: size of the problem is represented as Q, scout bees is represented as n, sites selected out of n visited sites is represented as s, the best sites out of s selected sites is represented ase, bees recruited for best e sites is represented asnep, bees recruited for the other selected sites is represented asnsp, other bees number is represented asosp and patch initial size is represented asngh which includes its site and its neighborhood in addition to Stopping condition in which our study is to minimize MMRE value.

The algorithm starts with an initial population of N scout bees which are to be employed to search for food. Each and every bee represents functional solution as a set of *K* analogy coefficient set of values. The scout bees are placed randomly in the given initial search space. The fitness computation process is computed, using Leave one out by cross validation, for each site which is visited by a scout bee by calculating the Mean Magnitude Relative Error (MMRE). This step is very essential for colony communication which shows the direction of flower patch which will be found, then its distance from the bee hive and its fitness value. This information helps the beescolony to send its bees to flower patches exactly, without using guides or maps. So, the best sites which isvisited by the highest fittest bees are being selected for its neighborhood search. The neighborhood search area is calculated by identifying the radius or diameter of the search area from best site which is considered as the important step in bees algorithm.

The algorithm still continues in searching the neighborhood of the selected sites, so therefore the need to recruit more bees to search the best sites which may have promising results. The bees that can be selected directly according to the fitness value associated with the sites that are being visited. Alternatively, the fitness values that are used to calculate the probability of the bees that are being selected. Therefore the fittest and healthiest bee from each and every patch is targeted to form the next bee population value. Our focus is to reduce the number of points to be visited with more promising value. Finally the remaining number of bees are assigned to search for randomly new potential solutions. These steps are repeated and circulated until the end condition is met or iteration number has been finished. At the end of each and every iteration, the colony of bees hive will have two parts of its new population – those that were the fittest representatives bees from a patch and those that have been sent out randomly from bees hive which failed to satisfy the fitness value. The pseudo code of bees algorithm is shown in Figure 3.

```
Input:Q,n,s,e,ngh,nep,nsp
Output:BestBee
Population ← InitializePopulation(n, Q)
While(! StopCondition())
      MMRE=EvaluatePopulation(Population)
      BestBee  ← GetBestSolution(Population)
      NextGeneration ← ø
      ngh ← ( ngh  X PatchDecreasefactor)
      Sites_best  ← SelectBestSites(Population, s)
      for(Site_i ∈ Sites_best)
            nsp ← ø
            if(i<e)    nsp  ←  nep
            else     nsp ← osp
            Neighborhood← ø
            for( j To  nsp )
              Neighborhood← CreateNeighborhoodBee(Site_i,ngh )
            end
            NextGeneration←  GetBestSolution(Neighborhood)
      end for
      RemainingBeesnum ← ( n - s)
      for(j To RemainingBeesnum)
          NextGeneration←  CreateRandomBee()
      end for
      Population←  NextGeneration()
  end while
  Return( BestBee )
```

Fig. 3.Bees algorithm workflow

B. Methodology

The prediction accuracy of several techniques is assessed using MMRE, PREDperformance measure. MMRE computes the mean of the percentage of error between actual and calculated project effort values as shown in Equation 4. PRED(0.25) is used for complementary condition to count the percentage of estimates that is calculated within less than 0.25 of the actual values.

$$MMRE = \sum_{i=1}^{N} \frac{|Effort(p_i) - \overline{Effort(p_i)}|}{Effort(p_i)} \quad (4)$$

$$PRED(0.25) = \frac{\lambda}{N} \times 100 \quad (5)$$

where λ is the number of projects that have magnitude relative error less than the value of 0.25 and N is the number of observations.

IV. OBSERVATION

Before stepping into case based reasoning the parameter values of the bees algorithm must be set ina order such that it yields optimal analogy.

Table 2.Bees algorithm sample values

Parameter	Value
problem size (*Q*)	Number of features + 1 for value of K
number of scout bees (***n***)	30
number of sites (***s***)	30
number of best sites out of ***s*** selected sites (***e***)	20
number of bees recruited for best ***e*** sites (***nep***)	15
number of bees recruited for the other selected sites	30
(***nsp***)	
other bees number (***osp***)	20
and initial size of patches (***ngh***)	15
Stopping criterion	Minimize *MMRE* of CBR+

But, there is no way to identify the best number of parameters values of bees algorithm therefore we did some of the investigations to select the best number of values for all datasets used in this environment. The parameter values used in this paper are presented in Table 2:

The obtained results show that there is some evidence that isfound using adjustment techniques which is better than that without adjustment as original case based reasoning.

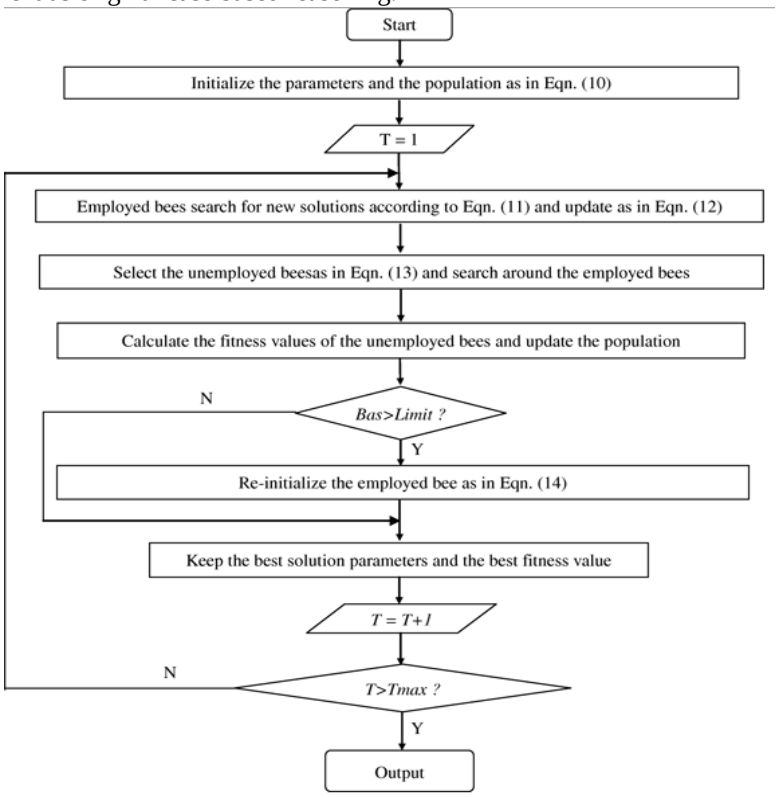

Fig. 4.Bees algorithm - pseudo code

V. CONCLUSION

This paper presents us a new and efficient method for calculating software effort estimation using combination of case based reasoning and Bees algorithm. We have used the bees algorithm to automatically specify the appropriate set of K analogy coefficient value set that are being used to adjust the retrieved closest number of targeted analogies. The results that we obtained shows much improvements on the prediction accuracy for Case based effort estimation. One of the most important advantages of the proposed method is that it does not prone to locally optimal solutions. This is due to the strength of the bees algorithm to perform local and global search randomly. While we are not yet guaranteed that the obtained performance values are the global optimum, the results obtainedis the best performance ever when all features are used without repetitions. The results of the current study proves the scope for further investigation, through better performance when adjustment to Case based effort estimation is experimented when compared with other set of results. Nevertheless, theresults without adjustment are significant as it will be useful for the further research to investigate and find out whether the use of feature weights can also viable in obtaining accurate estimated values.

REFERENCES

[1] Mohammad Azzeh. "A replicated assessment and comparison of adaptation techniques for analogy-based effort estimation", EmpiricalSoftware Engineering, 09/23/2011

[2] Mohammad Azzeh. "Adjusted Case-Based Software Effort Estimation Using Bees Optimization Algorithm", Lecture Notes in Computer Science, 2011

[3] Auer M, Trendowicz A, Graser B, Haunschmid E, Biffl S (2006) Optimal project feature weights in analogybased cost estimation: Improvement and limitations. IEEE Trans SoftwEng 32:83–92. doi:10.1109/ TSE.2006.1599418

[4] Azzeh, M., Neagu, D., Cowling, P., 2010. Fuzzy grey relational analysis for software effort estimation, EmpirsoftwEng, 60-90 DOI. 10.1007/s10664-009-9113-0.

[5] Azzeh, M., Neagu, D. & Cowling, P. (2008a), Fuzzy Feature Subset Selection for Software Effort Estimation, in proceedings of International workshop on software predictors PROMISE'08 (part of ICSE'08), Leipzig, Germany, 71-78.

[6] Boehm, B. (1981), *Software Engineering Economics*, Prentice-Hall, NY.

