Paper

A Novel Technique of Optimization for the COCOMO II Model Parameters using Teaching-Learning-Based Optimization Algorithm

Thanh Tung Khuat and My Hanh Le

The University of Danang, University of Science and Technology, Danang, Vietnam

Abstract—Software cost estimation is a critical activity in the development life cycle for controlling risks and planning project schedules. Accurate estimation of the cost before the start-up of a project is essential for both the developers and the customers. Therefore, many models were proposed to address this issue, in which COCOMO II has been being widely employed in actual software projects. Good estimation models, such as COCOMO II, can avoid insufficient resources being allocated to a project. However, parameters for estimation formula in this model have not been optimized yet, and so the estimated results are not close to the actual results. In this paper, a novel technique to optimize the coefficients for COCOMO II model by using teaching-learning-based optimization (TLBO) algorithm is proposed. The performance of the model after optimizing parameters was tested on NASA software project dataset. The obtained results indicated that the improvement of parameters provided a better estimation capabilities compared to the original COCOMO II model.

Keywords— COCOMO II, cost estimation, NASA software, optimization, teaching-learning-based optimization algorithm.

1. Introduction

Effort and cost estimation process in any software engineering project is an extremely important component. The success or failure of projects depends greatly on the accuracy of effort and schedule estimations. Errors in the cost estimation process can result in the serious issues [1]. Underestimating the costs may result in management approving proposed systems that then exceed their budgets, with underdeveloped functions and poor quality, and failure to complete on time. Overestimating may result in too many resources committed to the project, or, during contract bidding, result in not winning the contract, which can lead to the loss of jobs. Therefore, it is desired to find out the method to estimate the effort for software projects accurately. The introduction of the COCOMO II model has contributed significantly to the enhancement of accuracy in the software cost estimation process and currently this is one of the most commonly used models. COCOMO II has three sub-models including the Application Composition, the early design and the post-architecture (PA) models. The application composition model is used to estimate effort and schedule on projects that use integrated computer aided software engineering tools for rapid application development. The early design and the PA models are employed in estimating effort and schedule on application generator, system integration, or infrastructure developments [2]. In this work, we take into account the PA model, which is a detailed model being used once the project is ready to develop and sustain a fielded system.

Although COCOMO II is an efficient software cost estimation model, the accuracy of the model's output still relies on several constant values in the parametric-based estimation equations. These constants have not been optimized yet, and thus the accuracy of estimations on projects is not high in comparison with the actual effort and time. In this work, the constant values of COCOMO II model are optimized by using teaching-learning-based optimization (TLBO) algorithm. The proposed approach increases the efficiency of COCOMO II model when experimenting on "NASA 93" projects [3]. The test results showed that COCOMO II with optimized parameters had better performance in the software project cost estimation compared to the original COCOMO II and there was also smaller magnitude of relative error (MRE).

The remainder of paper is organized as follows. Section 2 introduces the COCOMO II model. Section 3 represents the teaching-learning-based optimization algorithm and its application into software cost estimation issues. The experiments are shown in Section 4 and finally in Section 5, the conclusion and future works are presented.

2. COCOMO II Model

COnstructive COst MOdel II (COCOMO II) [4], which was developed in 1995, is a model that allows one to estimate the cost, effort, and schedule when planning a new software development activity. It takes qualitative inputs and produces quantitative results. In COCOMO II, the effort is represented as person-months (PMs). A person-month is the amount of time one person spends working on the software development project for one month [5]. The

Table 1
Cost drivers for COCOMO-II PA model

Driver	Symbol	Very low	Low	Nominal	High	Very high	Extra high
RELY	EM_1	0.82	0.92	1.00	1.10	1.26	–
DATA	EM_2	–	0.90	1.00	1.14	1.28	–
CPLX	EM_3	0.73	0.87	1.00	1.17	1.34	1.74
RUSE	EM_4	–	0.95	1.00	1.07	1.15	1.24
DOCU	EM_5	0.81	0.91	1.00	1.11	1.23	–
TIME	EM_6	–	–	1.00	1.11	1.29	1.63
STOR	EM_7	–	–	1.00	1.05	1.17	1.46
PVOL	EM_8	–	0.87	1.00	1.15	1.30	–
ACAP	EM_9	1.42	1.19	1.00	0.85	0.71	–
PCAP	EM_{10}	1.34	1.15	1.00	0.88	0.76	–
PCON	EM_{11}	1.29	1.12	1.00	0.90	0.81	–
APEX	EM_{12}	1.22	1.10	1.00	0.88	0.81	–
PLEX	EM_{13}	1.19	1.09	1.00	0.91	0.85	–
LTEX	EM_{14}	1.20	1.09	1.00	0.91	0.84	–
TOOL	EM_{15}	1.17	1.09	1.00	0.90	0.78	–
SITE	EM_{16}	1.22	1.09	1.00	0.93	0.86	0.80
SCED	EM_{17}	1.43	1.14	1.00	1.00	1.00	–

Table 2
Scale factor values for COCOMO II model

Scale factors	Symbol	Very low	Low	Nominal	High	Very high	Extra high
PREC	SF_1	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	SF_2	5.07	4.05	3.04	2.03	1.01	0.00
RESL	SF_3	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	SF_4	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	SF_5	7.80	6.24	4.68	3.12	1.56	0.00

COCOMO II model predicts the software development effort by using the formula shown in Eq. 1.

$$PM = A \cdot Size^E \cdot \prod_{i=1}^{17} EM_i, \tag{1}$$

where A is a multiplicative constant having the value of 2.94, *Size*, which is the estimated size of software development, is the most important factor in calculating the effort of the software project and it is measured in kilo line of code (KLOC). EM_i is one of a set of effort multipliers shown in Table 1. This is the seventeen PA effort multipliers (EM) are used in the COCOMO II model to adjust the nominal effort. These multipliers are values of rating level of every multiplicative cost driver used to capture features of the software development affecting the effort to complete the project [5].

The exponent E in Eq. 1 is an aggregation of five scale factors (SF) that account for the relative economies or diseconomies of scale encountered for software projects of different sizes [4] and is computed as the following formula:

$$E = B + 0.01 \cdot \sum_{j=1}^{5} SF_j, \tag{2}$$

where B is a constant having the value of 0.91. Each scale factor has a range of rating levels, from very low to extra high. Each rating level has a weight which is presented in Table 2.

In addition to the effort, the software companies are also more interested in calculating the development time (TDEV) for projects [6]. It is derived from the effort according to the following equations:

$$TDEV = C \cdot PM^F, \tag{3}$$

$$F = D + 0.2 \cdot 0.01 \cdot \sum_{i=1}^{5} SF_j. \tag{4}$$

The values of C and D for the COCOMO II schedule equation are obtained by calibration to the actual schedule values for the 161 project currently in the COCOMO II database and results are C = 3.67 and D = 0.28.

Mean of MRE (MMRE) and prediction level (PRED) are usually used as an accurate reference value in the study of the software effort estimation. COCOMO's performance is often gauged in terms of PRED(30) [7]. PRED(30) is computed from the relative error (RE), which is the relative size of the difference between the actual and estimated values:

$$RE_i = \frac{estimate_i - actual_i}{actual_i}. \tag{5}$$

After that, the MMRE is the percentage of the absolute values of the relative errors, averaged over the T projects in the test dataset.

$$MRE_i = |RE_i|\,, \tag{6}$$

$$MMRE = \frac{100}{T} \cdot \sum_{i=1}^{T} MRE_i\,. \tag{7}$$

PRED(N) reports the average percentage of estimates that were within $N\%$ of the actual values:

$$PRED\,(N) = \frac{100}{T} \cdot \sum_{i=1}^{T} \begin{cases} 1, & \text{if } MRE_i \leq \frac{N}{100} \\ 0, & \text{otherwise} \end{cases}. \tag{8}$$

3. Teaching-Learning-Based Optimization Algorithm

In the COCOMO II model, the values of A, B, C, and D are constant and they are not tuned following the actual effort and time of new software projects. Therefore, the accuracy of estimated activities for projects is not exact. In this paper, the authors propose a novel approach to optimize these parameters of COCOMO II by using the historical software projects and TLBO algorithm.

3.1. Fitness Function for the Software Cost Estimation Problem

In the effort and time estimation issue for software projects, if the estimated cost roughly matches the actual end cost then the project is completed successfully. This means that the lower of the value of MMRE, the higher accuracy of the estimated cost is. Therefore, this paper uses the value of MMRE on training datasets of historical projects to assess the quality of cost estimations. The fitness function is the sum of time MMRE and effort MMRE as follows:

$$f = MMRE(Time) + MMRE(Effort)\,. \tag{9}$$

3.2. Teaching-Learning-Based Optimization Algorithm

Teaching-learning-based optimization algorithm which proposed by Rao *et al.* [8] is one of the most recently developed meta-heuristics. This algorithm is the population-based algorithm inspired by learning process in a classroom. For the TLBO, the population is considered as a group of learners or a class of learners. The search process contains two phases: teacher phase and learner phase.

3.2.1. Teacher Phase

In the teacher phase, learners get knowledge from a teacher. In the entire population, the best solution is considered as the teacher ($\vec{X}_{teacher}$). In this phase, the teacher tries to improve the results of other individuals ($\vec{X}_i$) by increasing the average result of the classroom ($\vec{X}_{mean}$) towards his/her level [8]. The solution is updated according to the difference between the existing and the new mean given by:

$$\vec{X}_{new} = \vec{X}_i + r_i \cdot (\vec{X}_{teacher} - T_f \cdot \vec{X}_{mean})\,, \tag{10}$$

where T_f is a teaching factor that decides the value of mean to be changed, and r_i is a random number in the range of $0 \ldots 1$. The value of T_f can be either 1 or 2, which is again a heuristic step. Moreover, $\vec{X}_{new}$ and $\vec{X}_i$ are the new and existing solutions of the i-th learner, respectively.

3.2.2. Learner Phase

In the learner phase, learners try to increase their knowledge by interacting with others. A learner interacts randomly with other learners with the help of group discussions, presentations, formal communications, etc. [8]. A learner learns something new if another learner has more knowledge than him or her. The modification of the learner is represented as follows:

$$\vec{X}_{new} = \vec{X}_i + r_i \cdot (\vec{X}_j - \vec{X}_k) \qquad \text{if} \quad f(\vec{X}_j) < f(\vec{X}_k)\,, \tag{11}$$

$$\vec{X}_{new} = \vec{X}_i + r_i \cdot (\vec{X}_k - \vec{X}_j) \qquad \text{if} \quad f(\vec{X}_k) < f(\vec{X}_j)\,, \tag{12}$$

Algorithm 1: The TLBO pseudo code

Input:
- d is the number of variables of problems
- n is the number of students
- G is the maximal number of generations

Output: The best individual in the population: $\vec{x}_{best} = \{x^1_{best}, x^2_{best}, x^D_{best}\}$.

Generate n initial students of the classroom randomly.
Calculate fitness function $f(\vec{X}_i)$ for whole students of the classroom.
$id = 0$
while $id < n$ && all $f(\vec{X}_i) \neq 0$ **do**
  Calculate the mean of each variable $\vec{X}_{mean}$
  Identify the best solution (teacher)
  for $i = 1$ **to** n **do**
    Find teaching factor $T_f = round[1 + rand(0,1)\{2-1\}]$
    Modify solution based on teacher:
      $\vec{X}_{new,i} = \vec{X}_i + rand(0,1) \cdot (\vec{X}_{teacher} - T_f \cdot \vec{X}_{mean})$
    Calculate fitness function for new student $f(\vec{X}_{new,i})$
    if ($\vec{X}_{new,i}$ is better than $\vec{X}_i$) **then**
      $\vec{X}_i = \vec{X}_{new,i}$
    end if
    Randomly select two learners $\vec{X}_j$ and $\vec{X}_k$ ($j \neq k$)
    if ($\vec{X}_j$ is better than $\vec{X}_k$) **then**
      $\vec{X}_{new,i} = \vec{X}_i + rand(0,1) \cdot (\vec{X}_j - \vec{X}_k)$
    else
      $\vec{X}_{new,i} = \vec{X}_i + rand(0,1) \cdot (\vec{X}_k - \vec{X}_j)$
    end if
    if ($\vec{X}_{new,i}$ is better than $\vec{X}_i$) **then**
      $\vec{X}_i = \vec{X}_{new,i}$
    end if
  end for
  $id++$
end while

where $\vec{X}_k$ and $\vec{X}_j$ ($j \neq k$) are two students chosen randomly in the population, and f is the fitness function.

If the new solution $\vec{X}_{new}$ is better, it is accepted in the population. The algorithm will continue until the termination condition is met. The Algorithm 1 shows the pseudo code of TLBO algorithm step by step.

4. Experimentation

The main objective of the experiment carried out is to reduce the uncertainty of current COCOMO II post architecture coefficients (A, B, C and D) and to get the best software effort estimation results being equivalent to the actual effort by using the TLBO algorithm. Experiments have been conducted on "NASA 93" dataset [3], in which 65 projects were used as training data to optimize the parameters for COCOMO II model and the other 28 projects were used for testing the performance of this model after optimizing coefficients. In this experiment, the configuration parameters for the TLBO are that the number of students is 200 and the number of generations is 2000.

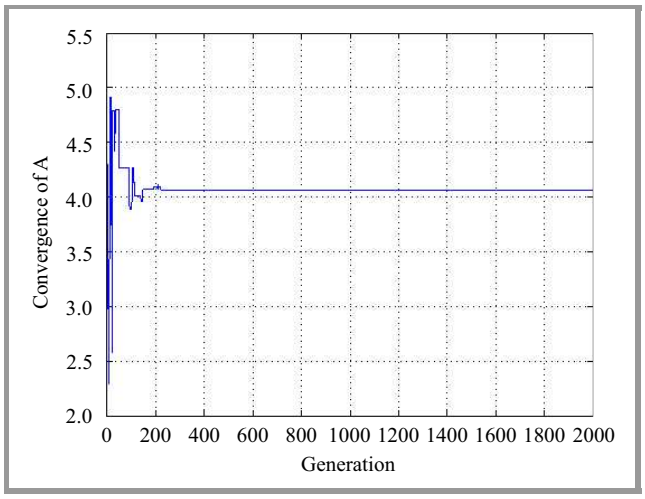

Fig. 1. Convergence of the model parameter A.

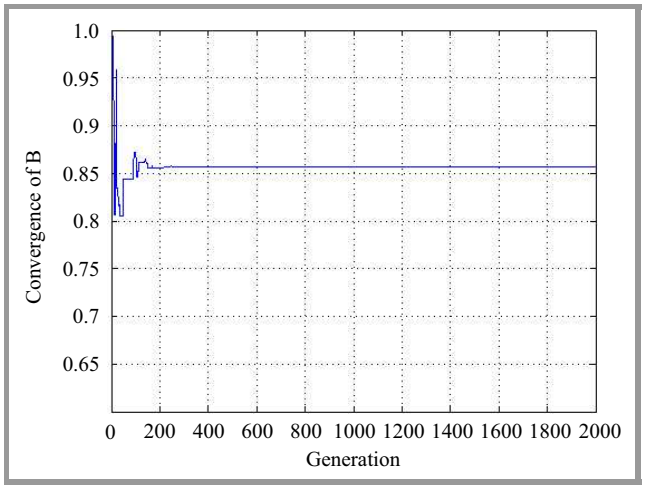

Fig. 2. Convergence of the model parameter B.

The optimized COCOMO II PA coefficients by using the TLBO are A = 4.064, B = 0.857, C = 2.938 and D = 0.357.

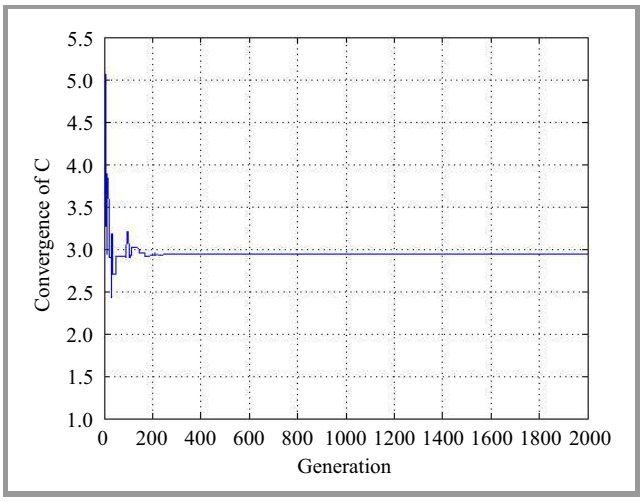

Fig. 3. Convergence of the model parameter C.

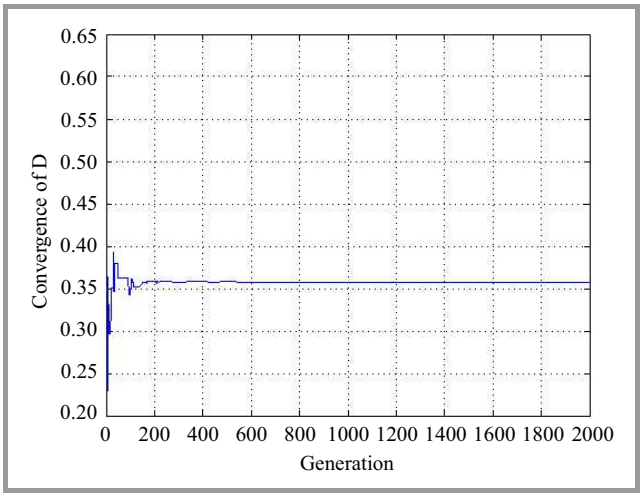

Fig. 4. Convergence of the model parameter D.

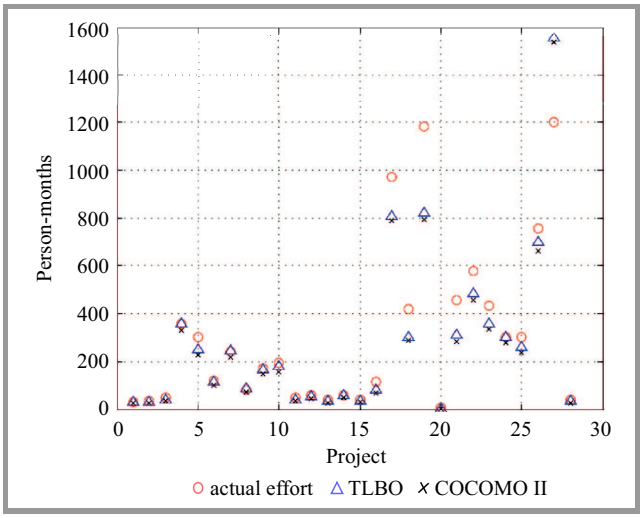

Fig. 5. Actual effort and estimated effort using TLBO and COCOMO II.

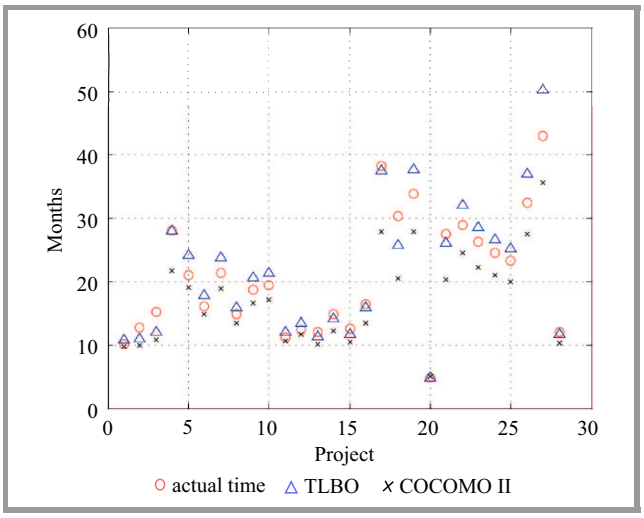

Fig. 6. Actual time and estimated time using TLBO and COCOMO II.

The convergence of the model parameters after each generation is described in Figs. 1–4.

Table 3
MRE values for estimations using TBLO and COCOMO II

Project ID	MRE of effort		MRE of time	
	TLBO	COCOMO II	TLBO	COCOMO II
3	0.0085	0.2008	0.0722	0.0367
13	0.1477	0.2989	0.1475	0.2294
15	0.1744	0.3000	0.2029	0.2870
16	0.0009	0.0774	0.0009	0.2244
22	0.1593	0.2403	0.1449	0.0940
23	0.0481	0.1760	0.1120	0.0768
28	0.0302	0.0845	0.1133	0.1219
29	0.0397	0.1279	0.0757	0.0953
31	0.0158	0.1436	0.1000	0.1118
32	0.0533	0.1725	0.1000	0.1163
34	0.1712	0.3212	0.0805	0.0462
35	0.0778	0.2327	0.0893	0.0551
36	0.2082	0.3675	0.0622	0.1650
37	0.0005	0.1716	0.0468	0.1798
39	0.1163	0.2862	0.0610	0.1682
40	0.2831	0.3993	0.0315	0.1835
44	0.1688	0.1887	0.0172	0.2723
47	0.2810	0.3131	0.1502	0.3256
56	0.3031	0.3279	0.1152	0.1783
58	0.5435	0.6716	0.0006	0.0187
61	0.3202	0.3841	0.0528	0.2619
69	0.1571	0.2065	0.1130	0.1525
70	0.1758	0.2221	0.0853	0.1529
72	0.0003	0.0778	0.0906	0.1435
73	0.1333	0.2066	0.0855	0.1402
76	0.0748	0.1236	0.1392	0.1549
77	0.2956	0.2789	0.1714	0.1724
93	0.0373	0.2618	0.0273	0.1510
MMRE	**14.38%**	**24.51%**	**8.89%**	**15.41%**

The graph in Fig. 5 illustrates the results of the effort estimation using the parameters optimized by TLBO and the original coefficients of COCOMO II compared with the actual effort. Figure 6 is the graph of values of estimated time by employing the parameters optimized by the TLBO and the original coefficients of COCOMO II in comparison with the actual time.

Based on these results, it can be seen that the COCOMO II with optimized parameters by the TLBO gived the higher estimated results compared to the original one because the estimated effort and time of the improved COCOMO II were more close to actual effort and time than the original model.

Table 3 shows the comparison of MRE between the improved COCOMO II model with optimized parameters by the TLBO and original model in terms of effort and time for 28 projects from NASA software project datasets. The obtained results indicated that the improved model have had lower MRE error compared to the original COCOMO model. As also can be seen that the model with optimized parameters has reduced MMRE error value for both the effort and time and it can be said that these are helpful methods for the software cost estimation process.

Another criterion to assess the effectiveness of the improved model is the value of PRED. From Table 3, the values of PRED(30) by using Eq. (8) for models as presented in Table 4 can be computed.

Table 4
The values of PRED(30) using TBLO and COCOMO II

	Time	Effort
TLBO	100%	89.29%
COCOMO II	96.43%	75%

Actually, the proposed method has considerably enhanced the accuracy of the software cost estimation in terms of effort and time.

5. Conclusion and Future Work

Accurate software cost estimation is a critical activity in the project planning. The authors found that the use of TLBO Algorithm to optimize the parameters of the COCOMO II model has resulted in the predicted effort and time of this model closing to the real effort. Thus, the proposed algorithm has effectively addressed the complicated optimization problem and achieved more accurate results by optimizing the coefficients of the COCOMO II model. The obtained results will contribute to the development of software projects within time and budgets.

However, there still exists some drawbacks in presented study. Experiments are only carried out on NASA projects which are characterized by lines of code, a number of scale factors and effort multipliers. The obtained results indicate that the improved model is more accurate on NASA

projects than traditional COCOMO II. Authors firmly believe that the proposed model is also more efficient than the conventional COCOMO II model for non-NASA projects influenced by factors as mentioned above. Due to the difficult in the project dataset collection, this has not yet been proven by experiments. Therefore, authors intend to apply the improved model for experimental studies on non-NASA projects in the future.

COCOMO II expands the capabilities of the original model and can estimate applications using modern development methods [9]. In the report of Jones [10], he pointed out that COCOMO II was one of the most widely used estimation tools in 2013. In [11], Menzies *et al.* analyzed the experiments and compared COCOMO II to other software effort estimation models to find the answer for the question "*Are the old parametric calibrations relevant to more recent projects*?". Authors concluded that COCOMO II calibration is relevant to more recent projects. These figures indicate that the improved COCOMO II model still counts in estimating the effort for contemporary software projects. Therefore, the proposed model in this paper might be utilized for predicting the effort of the current software projects. Authors plan to carry out experiments to verify the effectiveness of the improved COCOMO II on the modern projects. This is an important area that requires further research.

In the future work, authors also intend to apply the TLBO Algorithm for Agile Software Effort Estimation. The various nature-inspired algorithms will be employed to optimize the parameters of the COCOMO II model as well.

References

[1] C. Jones, "Why flawed software projects are not cancelled in time", *Cutter IT J.*, vol. 10, no. 12, pp. 12–17, 2003.

[2] B. Clark, S. Devnani-Chulani, and B. Boehm, "Calibrating the COCOMO II post-architecture model", in *Proc. 20th Int. Conf. Softw. Engin. ICSE*, Kyoto, Japan, 1998, pp. 477–480.

[3] T. Menzies, The Tera-PROMISE Repository for COCOMO 93, 2015 [Online]. Available: http://openscience.us/repo/effort/cocomo/nasa93.html

[4] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0", *Annals of Softw. Engin.*, vol. 1, no. 1, pp. 57–94, 1995.

[5] C. Abts, B. Clark, S. Devnani-Chulani, E. Horowitz, R. Madachy, D. Reifer, R. Selby, and B. Steece, "COCOMO II Model Definition Manual", Tech. Rep., Center for Software Engineering, University of Southern California, Los Angeles, CA, USA, 1998.

[6] J. Kaur and R. Sindhu, "Parameter estimation of COCOMO II using tabu search", *Int. J. Comp. Sci. Inform. Technol.*, vol. 5, no. 3, pp. 4463–4465, 2014.

[7] Z. Chen, T. Menzies, D. Port, and B. Boehm, "Feature Subset Selection Can Improve Software Cost Estimation Accuracy", in *Proc. Int. Worksh. Predic. Models in Softw. Engin. PROMISE 2005*, St. Louis, MO, USA, 2005, pp. 1–6.

[8] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-Learning-Based optimization: A novel method for constrained mechanical design optimization problems", *Computer-Aided Design*, vol. 43, pp. 303–315, 2011.

[9] B. Boehm, C. Abts, and S. Chulani, "Software development cost estimation approaches – A survey", *Annals of Softw. Engin.*, vol. 10, no. 1–4, pp. 177-205, 2000.

[10] C. Jones, "A short history of software estimation tools", Tech. Rep., VP and CTO, Namcook Analytics LLC, Narragansett, RI, USA, 2013.

[11] T. Menzies, B. Boehm, Y. Yang, J. Hihn, and N. Lekkalapudi, "Just how good is COCOMO and parametric estimation", in *Proc. 29th Int. Forum on COCOMO and Syst. Softw. Cost Model.*, Los Angeles, CA, USA, 2014 [Online]. Available: http://csse.usc.edu/new/events/cocomo-2014/program

Thanh Tung Khuat completed the B.Sc. degree in Software Engineering from University of Science and Technology, Danang, Vietnam, in 2014. Currently, he is participating in the research team at DATIC Laboratory, University of Science and Technology, Danang. His research interests focus on software engineering, software testing, evolutionary computation, intelligent optimization techniques and applications in software engineering.

Email: thanhtung09t2@gmail.com
The University of Danang
University of Science and Technology
54 Nguyen Luong Bang, Lien Chieu
Danang, Vietnam

My Hanh Le is currently a lecturer of the Information Technology Faculty, University of Science and Technology, Danang, Vietnam. She gained M.Sc. degree in 2004 and completed the Ph.D. program in Computer Science at the University of Danang in 2015. Her research interests are about software testing and more generally application of heuristic techniques to problems in software engineering.

Email: ltmhanh@dut.udn.vn
The University of Danang
University of Science and Technology
54 Nguyen Luong Bang, Lien Chieu
Danang, Vietnam