doi: 10.2478/v10313-012-0006-7
2012 /15

The Optimization of COCOMO Model Coefficients Using Genetic Algorithms

Anna Galinina[1], Olga Burceva[2], Sergei Parshutin[3], [1-3]*Riga Technical University*

***Abstract* – Nowadays there are many models for software development cost estimation, providing project managers with helpful information to make the right decisions. One of such well-known mathematical models is the COCOMO model. To estimate costs and time, this model uses coefficients, which were determined in 1981 by means of the regression analysis of statistical data based on 63 different types of project data. Using these coefficients for a modern project, the appraisal may not be accurate; therefore, the aim of this paper is to optimize the model coefficients with genetic algorithms. Genetic algorithms are evolutionary methods for optimization. To evaluate population, the genetic algorithm will use a set of descriptive attributes of several software development projects. These attributes are the number of lines of a code, costs and implementation time of a project. Project costs estimated by means of the COCOMO model will be compared with the real ones, this way evaluating the fitness of an individual in the population of possible solutions.**

***Keywords* – COCOMO model, genetic algorithm, software cost estimation**

I. Introduction

Software cost estimation is essential for software project management. Accurate software estimation can provide good support for the decision-making process like the accurate assessment of costs can help the organization to better analyse the project and effectively manage the software development process, thus significantly reducing the risk. Once the planning is too pessimistic, it may lose business opportunities, but too optimistic planning can cause significant loss.

There are several software cost estimation models to help project managers to make the right decisions. One of such models is the COCOMO model (Constructive Cost Model). It was introduced in 1981 by Barry Boehm – the famous scientist who contributed to the development of software project management by creating a scientific approach. The COCOMO model is based on 63 different types of statistical data analysis project. The actual number of lines of code, amount of effort and time were estimated and some coefficients, which depend on the software project, were developed and identified during the regression analysis phase. Today's project evaluation based on old coefficients may not match the required accuracy; therefore, the aim of this research is to optimize the model coefficients [4].

The COCOMO model has three modes, depending on the size of the project and the project team size. The model has three levels. The accuracy of the base level is lower than in the intermediate level and detailed level because the estimation of effort uses only actual amount and information about the mode and does not use cost drivers, which include a subjective judgment of the product, project, personnel and hardware characteristics. Thus, in this article a basic level of COCOMO model will be discussed [4], [6].

Genetic algorithms are optimization algorithms in evolutionary computing techniques, proposed in 1975 by a scientist Holland. It is a natural heuristic algorithm that is used to find the exact and approximate solutions. Algorithm is based on the iterative improvement of the current solution, but a solution set is used instead of one solution. Most genetic algorithm applications are linked to a large-scale information processing and the development of prediction models [3].

II. Background

A brief description of the classic COCOMO model and genetic algorithms is given in the present section.

A. COCOMO Model

The COCOMO model has three modes to classify complexity of the system. Description of these modes is given in Table I [4]. The COCOMO model has three levels, providing increase of accuracy in each subsequent level. To calculate the effort at the base level, the equation (1) is used. It shows that effort is linearly dependant on the project size and rapidly changes if there is another mode [7], [9]. To evaluate project development time, the equation (2) is used.

$$E = a \cdot KLOC^{b}, \tag{1}$$

a, b – the COCOMO model coefficients;
$KLOC$ – the kilo-lines of code;
E – the effort (man-months).

TABLE I
MODES OF COCOMO MODEL

Software project	Project size	Description
Organic	Less than 50 *KLOC*	Small project being developed by a small team; there is no specific major innovation, the environment remains stable, and there are no strict limits and deadlines.
Semi-detached	50 – 300 *KLOC*	Medium-scale projects being developed by a relatively small team; the development process needs some innovation, the environment is characterized by low volatility; average strict limitations and deadlines are present.
Embedded	Over 300 *KLOC*	Large project being developed by a large team requires many innovations; the environment has a number of elements, which are characterized by instability, strict limitations and deadlines.

$$TDEV = c \cdot E^d , \tag{2}$$

c, d – the COCOMO model coefficients;
TDEV – the development time (months).

B. Genetic Algorithm

Genetic algorithm is based on the next 4 main components:

1. *Chromosome* – the line of numbers that could be encoded using the binary encoding, integer number encoding etc. Each position in chromosome is called a bit, gene. Chromosome is an individual representing one of task solutions [3].
2. *Initial population*. The first population is a set of task solutions that is generated randomly. The main condition of the generation process of the first population is to achieve a variety of solution sets. If this condition is false – local extreme will be achieved early. It is not good for searching of the optimal solution [3].
3. *Operator set.* Operator set allows generating new solutions on the base of current population. Operator set contains selection, crossover and mutation. Fig. 1 demonstrates the basic genetic algorithm. When selection is used, individuals are selected in the intermediate population. Different types of selection are known: Roulette wheel selection – each individual probability to be chosen in the intermediate population is proportional with it fitness function value, it is called the proportional selection; Tournament selection – all individuals have an equal probability to be chosen in the intermediate population.

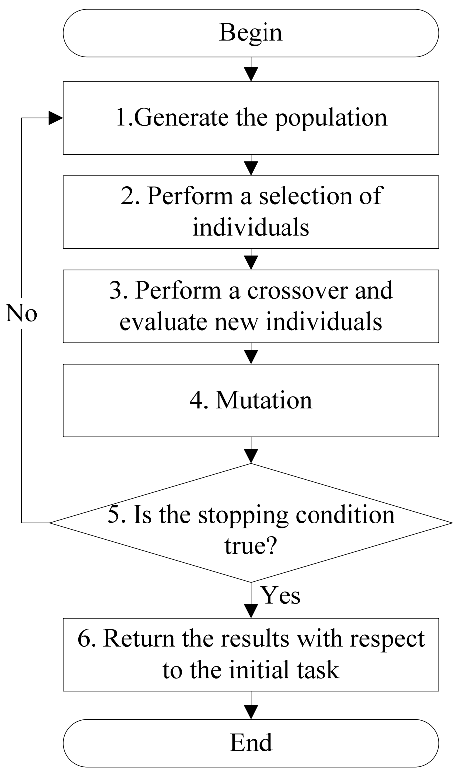

Fig. 1. The steps of the basic genetic algorithm

Respectively the crossover is chosen – the one-point crossover, two-point crossover, *n*-point crossover – individuals chosen to the intermediate population must make the exchange of chromosome parts. This process result is the generation of new individuals.
The use of the mutation chromosome gene with defined probability exchanges its value. The new value of gene is also determined with defined probability. The mutation process protects population from the local extreme points, as well as enlarges the searching solution area [8].

4. *Fitness function*. The fitness function is the individual estimation attribute. It shows the suitability for each solution. On the one hand, the fitness function allows defining solutions that are more adapted – these solutions get a chance to be chosen in intermediate population. On the other hand, the fitness function allows defining solutions that are less adapted – these individuals are removed from the solution set. Therefore, the average fitness function value of new generation is larger than the average fitness function value of previous generation [3], [8].

III. Dataset

In the current research, two datasets are used – one for each mode of the COCOMO model. The dataset for an organic mode contains 20 records, and for a semi-detached mode the dataset of 19 records is taken. Table II contains both datasets [1].

TABLE II
DATASETS

	Organic mode		**Semi-detached mode**	
No.	***LOC***	***TDEV*, minutes**	***KLOC***	***TDEV*, months**
1	10	13	59.3	21
2	17	12	60	21
3	10	15	54	15
4	24	18	52	9
5	10	13	51.9	11
6	10	14	64	16
7	10	15	51	17
8	15	13	80	14
9	10	12	54	25
10	10	12	76	38
11	17	22	51.4	30
12	11	19	66	17
13	15	18	50.2	17
14	15	19	65	10
15	13	21	55	10
16	14	20	57	28
17	15	19	72	22
18	15	20	66.5	19
19	13	15	69.1	18
20	18	19		

Datasets were split into training and test sets. The training sets of both datasets contain first 14 records; the rest records are included into test sets; 5 records – for the organic test set and 4 records – for the semi-detached test set. Each record contains information about the completed software development project. Each record is described with two attributes – the lines of code (*LOC, KLOC*) and development time (*TDEV*).

Based on the fact that the dataset with two attributes (effort and development time) was not available, the decision was taken to express the development time (2) by using the equation (1). As a result, a new equation was introduced (3). All COCOMO model coefficients (*a*, *b*, *c*, *d*) and the effort are included in this equation.

$$TDEV = c \cdot \left(a \cdot LOC^{b}\right)^{d}, \quad (3)$$

a, b, c, d – the COCOMO model coefficients;
LOC – the lines of code;
TDEV – the development time (months).

IV. COCOMO Model Coefficient Optimization Using Genetic Algorithms

To optimize the COCOMO model coefficients, the genetic algorithm was proposed. Fig. 3 shows its main steps:

1. *Generate the initial population*. Individuals are created randomly. A specific chromosome structure is chosen. It is shown in Fig. 2.

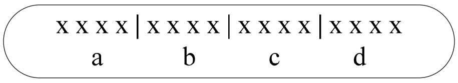

Fig. 2. Each of 4 COCOMO model coefficients is expressed by 4 genes, containing integer numbers from 0 to 9. The first gene contains an integral part of a coefficient; the remaining 3 are the fractions

2. *Calculate the predicted development time for the project.* The *TDEV* is being calculated for each project *j* in the training set, using the coefficients from an individual *i*.
3. *Calculate the project specific fitness of an individual.* To calculate individual *i* fitness for the project *j* the equation (3) is used. If the dataset contains the effort, then equation (4) is used.

$$Fitness_{ij} = \frac{TDEV_R^{j} - TDEV_P^{ij}}{TDEV_R^{j}}, \quad (3)$$

i – the individual number;
j – the project number;
$TDEV_R^{j}$ – the real project *j* development time;
$TDEV_P^{ij}$ – the predicted project development time, using individual *i*.

$$Fitness_{ij} = \frac{E_R^{j} - E_P^{ij}}{E_R^{j}}, \quad (4)$$

i – the individual number;
j – the project number;
E_R^{j} – the real project effort;
E_P^{ij} – the predicted project effort, using individual *i*.

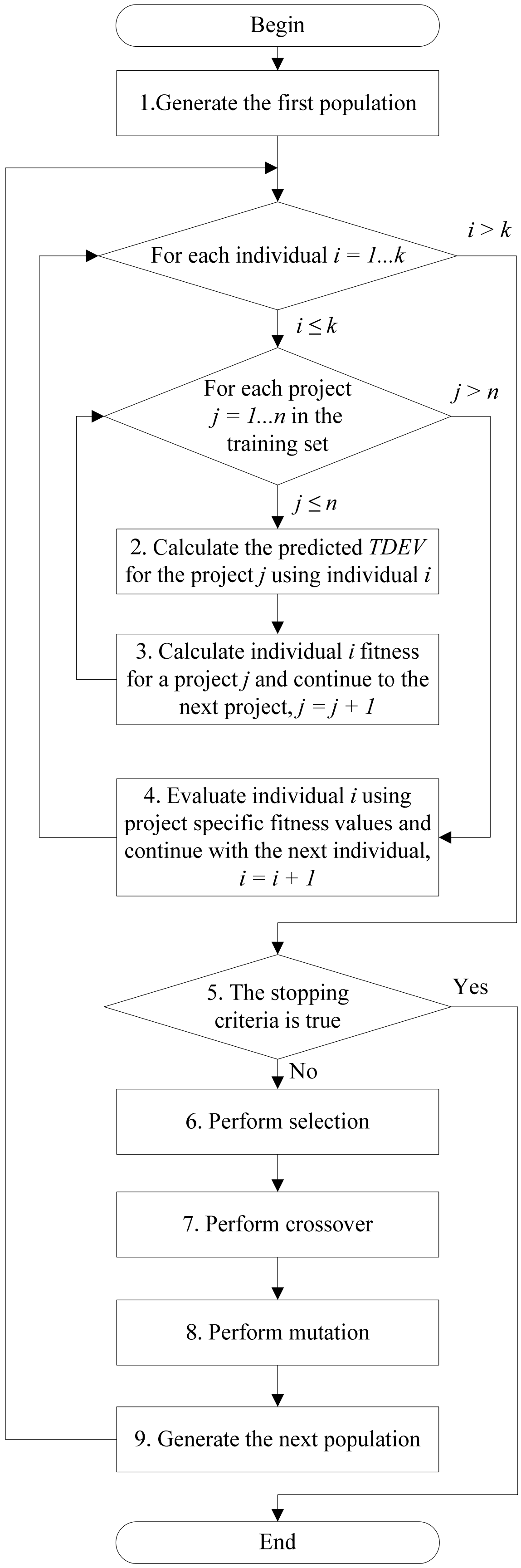

Fig. 3. The proposed algorithm for the optimization of the COCOMO model coefficients

4. *Evaluate individual.* For the calculation of the individual fitness, the equation (5) is used. It is calculated as the average value of all project specific fitness values of an individual, obtained during steps 2 and 3 in Fig. 3. In this specific task, the fitness value depends on the difference between real development time and predicted development time; therefore, the fitness function value should be minimized.

$$Fitness_i = \frac{1}{n} \cdot \sum_{j=1}^{n} Fitness_{ij} \quad (5)$$

i – the individual number;
j – the project number;
n – the number of projects.

5. *Checking the stopping condition.* The stopping condition defines when the algorithm must be finished. The iteration number is defined as a stopping condition. As an alternative condition the best (minimal) and average fitness of the population can be used.
6. *Selection.* The roulette wheel selection is used to form a set of individuals that will take part in crossover.
7. *Crossover.* Using individuals, selected in a previous step, new individuals are generated. The *n*-point crossover is used, which is the simplest crossover and can be easily replaced by another crossover type.
8. *Mutation.* During the mutation step, none or a small number of individuals is randomly selected. The probability to be selected should be low, typically about 10%. For each individual a random scheme of mutating genes is defined. As the integer number encoding is used (see Fig. 2), the implemented mutation concept is very simple – mutating genes are assigned new randomly generated integer numbers in a range from 0 to 9.
9. *Generate the new population.* During this step, the new population is formed. A number of strategies for running this step exist. The proposed algorithm applies the strategy that is based on concurrency among parents and children. The best individuals among parents and children are chosen using the same selection method applied to the step 6. As an option the elitism can be added to the strategy, insuring that the best solution will not be lost.

V. Results

During the experiments, the initial population of 10 individuals was generated. Then the optimization of the COCOMO model coefficients was performed using the proposed algorithm. Based on the fact that each of the three modes of COCOMO model has its own coefficients, experiments were performed using datasets according to each mode. Experiments were performed by changing the genetic algorithm parameters (number of iterations, crossover points).

A. Organic mode experiments

In experiments using organic mode datasets, the best result was achieved using 1500 iterations and the three-point crossover strategy.

As a result of algorithm execution, a solution set was received from which the best individual was chosen – a solution with the best fitness function value, the Mean Relative Estimation is 14.79 (training set):

2 3 1 2 | 1 1 1 4 | 3 0 2 0 | 0 4 2 1.

According to this individual, the resulting optimized COCOMO model coefficients are the following: *a=2.312; b=1.114; c=3.02; d=0.421*. Current COCOMO model coefficients are the following: *a=2.4; b=1.105; c=2.5; d=0.38*.

Using the optimized coefficients for the test sets records Mean Relative Estimation value is 18.66. It is 2.48 times less than the Mean Relative Estimation value obtained with the current COCOMO models coefficients, which equals 46.32. Table III contains actual and predicted *TDEV* values for the organic mode dataset; using the genetic algorithm optimised and current COCOMO model coefficients.

TABLE III
PREDICTED DEVELOPMENT TIME VALUES FOR THE ORGANIC MODE

No.	Actual *TDEV*	Calculated *TDEV* using coefficients optimized by GA	Calculated *TDEV* using COCOMO model current coefficients
1	21	18	12
2	20	18	12
3	19	17	11
4	20	18	12
5	15	15	10
6	19	17	11

The data in Table III is depicted in Fig. 4, showing the comparison among the organic mode real development time, the predicted development time using GA coefficients and the predicted development time using current COCOMO model coefficients.

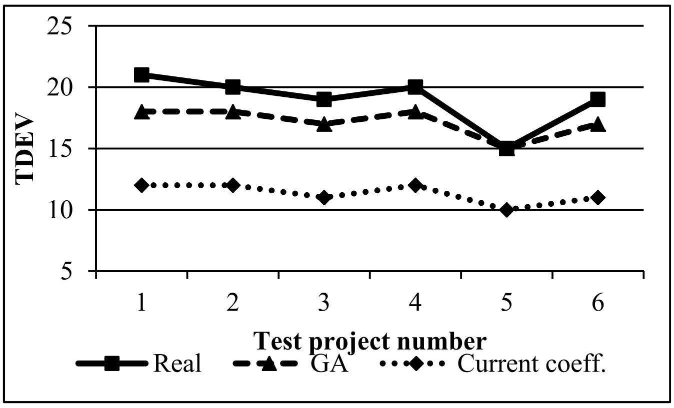

Fig. 4. The real organic mode project development time (*TDEV*) and the calculated one using the coefficients optimized by GA and the current COCOMO model coefficients

Fig. 4 shows that the development time predicted by current COCOMO model coefficients is much lower than the actual data. At the same time, in the results obtained using the coefficients optimized by the genetic algorithm, the error is much lower, but it still persists.

Another experiment was performed to evaluate the genetic algorithm efficiency in coefficient optimization for the effort forecasting. The following results were obtained: Mean Relative Estimation value using the coefficients optimized by GA was 33.252; Mean Relative Estimation value using current COCOMO model coefficients was 41.387. This shows that the optimized coefficients more accurately predict the project effort than the current COCOMO model coefficients. The obtained experimental effort prediction results for the organic mode are shown in Table IV.

TABLE IV

PREDICTED EFFORT VALUES FOR THE ORGANIC MODE

No/	*KLOC*	Actual Effort	Calculated *TDEV* using coefficients optimized by GA	Calculated *TDEV* using COCOMO model current coefficients
1	14	60	44	38
2	10.4	50	31	28
3	12.8	62	40	35
4	15.4	70	49	42
5	16.3	80	52	45

The graphical comparison of project effort prediction results for the organic mode is displayed in Fig. 5.

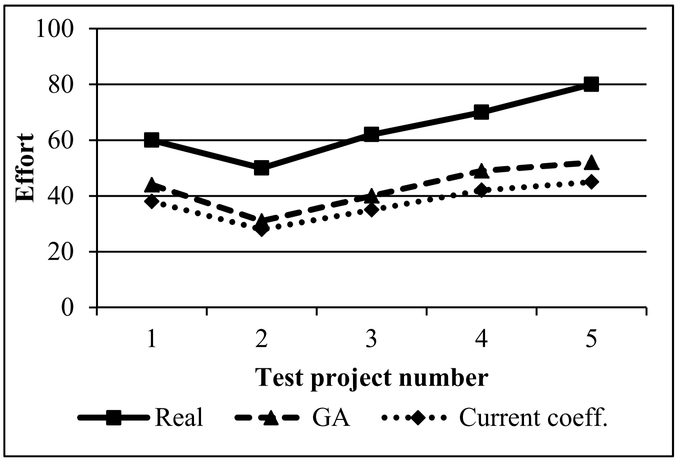

Fig. 5. The real organic mode project effort and the predicted one using the coefficients optimized by GA and the current COCOMO model coefficients

Fig. 5 shows that despite the fact that the COCOMO model coefficients were improved and the error reduced, the forecasted effort is far from the real one. Having conducted the experiments, it can be stated that the use of the COCOMO organic mode coefficients optimized by genetic algorithms to forecast the project effort and development time produces more accurate results than the use of the current COCOMO model coefficients.

B. Semi-Detached Mode Experiments

Optimization of coefficients according to datasets for a semi-detached mode was performed using a similar genetic algorithm that was used in the organic mode experiments. The best result for the semi-detached mode was obtained using 1000 iterations and the three-point crossover strategy. The resulting individual is the following:

3 0 2 0 | 1 0 7 0 | 2 5 0 0 | 0 3 5 0.

According to this individual, the resulting optimized COCOMO model coefficients for the semi-detached mode are the following: $a=3.020;$ $b=1.070;$ $c=2.500;$ $d=0.350$. The fitness function value equals 30.75, which is less than the fitness function value of the current COCOMO model semi-detached mode coefficients ($a=3;$ $b=1.12;$ $c=2.5;$ $d=0.35$), which equals 33.15. Mean Relative Estimation value for optimized coefficients is 21.52, but for current coefficients – 21.82. As may be seen, only coefficients a and b slightly differ, the same time coefficients c and d remain constant. The obtained results, using the COCOMO model coefficients optimized by the genetic algorithm, and the current ones, are shown in Table V.

TABLE V

PREDICTED DEVELOPMENT TIME VALUES FOR THE SEMI-DETACHED MODE

No/	Actual *TDEV*	Calculated *TDEV* using coefficients optimized by GA	Calculated *TDEV* using COCOMO model current coefficients
1	10	17	18
2	28	17	18
3	22	18	20
4	19	18	19
5	18	18	19

Fig. 6 shows the results from Table V. As may be seen, the results produced by GA optimized coefficients are close to the ones produced by using the current COCOMO model coefficients, but remain slightly worse.

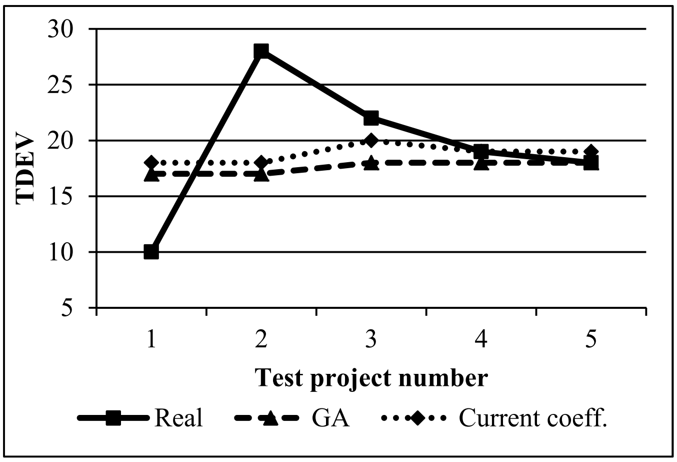

Fig. 6. The real semi-detached mode project development time (*TDEV*) and the calculated one using the coefficients optimized by GA and the current COCOMO model coefficients

As a result of the effort prediction experiment for the semi-detached mode, the next results are obtained: Mean Relative Estimation value using the coefficients optimized by GA equals 28; using the current COCOMO model coefficients – 14.4. Predicted effort values are shown in Table VI.

TABLE VI
PREDICTED EFFORT VALUES FOR THE SEMI-DETACHED MODE

No.	***KLOC***	**Actual Effort**	**Calculated *TDEV* using coefficients optimized by GA**	**Calculated *TDEV* using COCOMO model current coefficients**
1	115.8	480	488	614
2	66.6	300	270	331
3	50.5	270	201	243
4	79	400	324	400
5	77	315	315	389

The graphical comparison of the real and predicted effort using the coefficients optimized by GA with the current COCOMO model coefficients for the semi-detached mode is shown in Fig. 7.

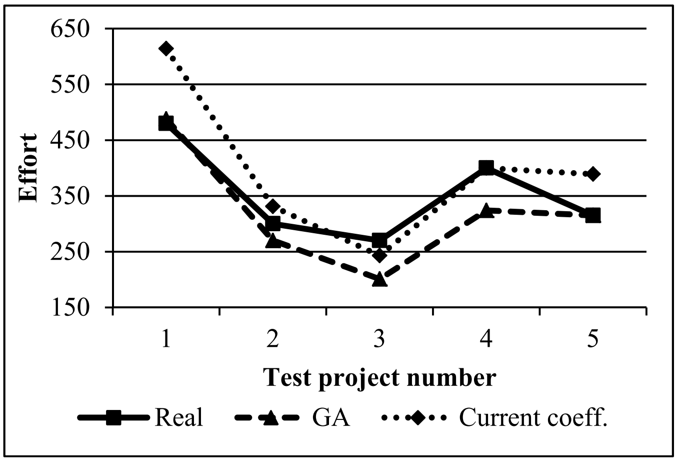

Fig. 7. The real semi-detached mode project effort and the predicted one using the coefficients optimized by GA and the current COCOMO model coefficients

Fig. 7 shows that the coefficients optimized by the genetic algorithm and the current COCOMO model coefficients produce inaccurate results. Error values are almost the same in both cases; however, GA coefficients are slightly better.

VI. CONCLUSIONS

The objective of this research was to optimize the COCOMO model coefficients using the genetic algorithms. The task of the COCOMO coefficient optimization is not new; different methods such as neural networks, fuzzy algorithms, object-oriented methods etc. were applied to it by a number of scientists.

The current research proposes a genetic algorithm based method for optimization of the COCOMO model coefficients both for organic and semi-detached modes. In a series of experiments, the proposed algorithm was tested and the obtained results were compared with the ones obtained using the current COCOMO model coefficients. The results show that in most cases the results obtained using the coefficients optimized by the proposed algorithm are close to the ones obtained using the current coefficients. Comparing organic and semi-detached COCOMO model modes, it can be stated that use of the coefficients optimized by the GA in the organic mode produces better results in comparison with the results obtained using the current COCOMO model coefficients. At the same time, coefficients for the semi-detached mode produced by the proposed algorithm do not result in the high forecasting accuracy, which is the same or slightly worse than the accuracy obtained using the current COCOMO model coefficients.

According to the findings of the research, it should be stated that having the appropriate statistical data describing the software development projects, genetic algorithms can be used to optimize the COCOMO model coefficients.

REFERENCES

[1] R. Bhatnagar, V. Bhattacharjee, M.-K. Ghose, Software Development Effort Estimation – Neural Network Vs. Regression Modeling Approach, Vol. 2, International Journal of Engineering Science and Technology, 2010, pp. 2950-2956.

[2] R. Chandrasekaran and R.-V. Kumar, On the Estimation of the Software Effort and Schedule using Constructive Cost Model – II and Functional Point Analysis, Vol. 44, Department of Statistics, Tambaram: 2012, pp. 38-44.

[3] J. Horn, N. Nafpliotis and E. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization. Orlando, USA: IEEE; 1994, pp. 82-87.

[4] X. Huang, D. Ho, J. Ren, L.F. Capretz, Improving the COCOMO model using a neuro-fuzzy approach. Vol. 7, Applied Soft Computing, 2007, pp. 29-40.

[5] A. Kaushik and A. Chauhan, COCOMO Estimates Using Neural Networks, Vol. 9, Intelligent Systems and Applications, Delhi: Modern Education and Computer Science Press, 2012, pp. 22-28.

[6] Y. Miyazaki, K. Mori, COCOMO evaluation and tailoring. Proc. Eighth Int. Conf. Soft. Eng., London, UK, 1985, pp. 292-299.

[7] T. Potok and M. Vouk, Development Productivity for Commercial Software Using Object-Oriented Methods, Vol. 29, Software – Practice and Experience, North Carolina: IBM Press, 1999, pp. 833-837.

[8] R. Sarker, K-H. Liang, C. Newton, A new multiobjective evolutionary algorithm. Eur J Oper Res 2002, pp. 12–23.

[9] J. Singh and B. Sahoo, Software Effort Estimation with Different Artificial Neural Network, Vol. 2, Computing, Communication and Sensor Network, Rourkela: CCSN Press, 2011, pp. 13-17.

Anna Galinina, MSc student, Riga Technical University, the Institute of Information Technology, 1 Kalku Street, Riga, LV - 1658, Latvia, gestija18@inbox.lv.
Anna Galinina received her BSc degree in Computer Management and Computer Science from Riga Technical University in 2011. Currently she is an MSc student at the Faculty of Computer Science and Information Technology, Riga Technical University. Her research interests include data mining and knowledge discovery, genetic algorithms.

Olga Burceva, MSc student, Riga Technical University, the Institute of Information Technology, 1 Kalku Street, Riga, LV - 1658, Latvia,
olgaa.burceva@gmail.com
Olga Burceva received her BSc degree in Computer Management and Computer Science from Riga Technical University in 2011. Currently she is an MSc student at the Faculty of Computer Science and Information Technology, Riga Technical University. Her research interests include data mining and knowledge discovery, genetic algorithms.

Sergei Parshutin,
Lecturer, Research Fellow, Riga Technical University, the Institute of Information Technology, 1 Kalku Street, Riga, LV - 1658, Latvia, sergei.parshutin@rtu.lv.
Dr. sc.ing. Sergei Parshutin is a Lecturer and Research Fellow at the Faculty of Computer Science and Information Technology, and the Department of Modelling and Simulation, Riga Technical University. His research interests include data mining and knowledge discovery, intelligent information systems, intelligent agent technology, evolutionary computing and decision support.

Anna Galiņina, Olga Burceva, Sergejs Paršutins. COCOMO modeļa koeficientu optimizācija ar ģenētiskiem algoritmiem
Pašlaik pastāv daudzi programmatūras izmaksu novērtēšanas modeļi, kuri dod projektu vadītājiem noderīgu informāciju, lai pieņemtu pareizos lēmumus. Viens no šādiem labi zināmiem matemātiskiem modeļiem ir COCOMO modelis. Modelī darbietilpīguma un laika izvērtēšanai tiek izmantoti koeficienti, kuri ir noteikti regresijas analīzes posmā, balstoties uz 1981. gadā 63 dažādu projektu statistisko datu analīzi. Šodien projektu novērtēšana ar dotiem koeficientiem var nedot nepieciešamo precizitāti, tāpēc darba mērķis ir optimizēt modeļa koeficientus. COCOMO modelim ir trīs līmeņi un trīs režīmi, kuri ir atkarīgi no projekta apjoma un izstrādātāju komandas lieluma. Dotajā pētījumā tiek apskatīts tikai viens bāzes līmenis un divi režīmi – organiskais un starpposmu. COCOMO modeļa koeficientu optimizācijas uzdevums nav jauns. Uzdevuma risināšanai jau tika pielietoti neironu tīkli, izplūdušie algoritmi, objektu orientētas metodes un citi. Šajā darbā koeficientu optimizācijai tika nolemts pielietot ģenētiskos algoritmus. Ģenētiskie algoritmi ir evolūcijas metode optimizācijai. Lai novērtētu populācijas, ģenētiskie algoritmi izmanto datu kopu ar aprakstošiem atribūtiem no vairākiem programmatūras izstrādes projektiem. Tie ir – programmas koda garums, darbietilpīgums un projekta patērētais laiks. Kopumā tika izmantotas divas datu kopas, viena datu kopa uz katru COCOMO modeļa režīmu. COCOMO modeļa prognozētais projekta darbietilpīgums tika salīdzināts ar reālām vērtībām, šādā veidā izvērtējot indivīda piemērotību populācijas iespējamiem risinājumiem. Eksperimentu gaitā tika ģenerēta sākuma populācija no desmit indivīdiem un tika veikta koeficientu optimizācija. Izmantojot trīspunktu krustošanas stratēģiju, organiskajā režīmā labākais rezultāts tika sasniegts 1500. iterācijā, starpposmu režīmā labākais rezultāts tika sasniegts 1000. iterācijā. Kodēšanai tika pielietoti veselie skaitļi no nulles līdz deviņi. Salīdzinot prognozēto darbietilpīgumu un izstrādes laiku, kuri tika iegūti, izmantojot esošos koeficientus un jaunos koeficientus ar reālajiem datiem, tika konstatēts, ka jaunie koeficienti dod precīzāku rezultātu.

Анна Галынина, Ольга Бурцева, Сергей Паршутин. Оптимизация коэффициентов COCOMO модели с помощью генетических алгоритмов
В настоящее время существует множество моделей для оценки стоимости разработки программного обеспечения, которые предоставляют полезную информацию менеджерам для принятия правильных решений. Одной из таких известных математических моделей является модель COCOMO. Для оценки трудозатрат и времени модель использует коэффициенты, которые были найдены в 1981 году методом регрессивного анализа статистических данных, основанных на 63 различных типах проектных данных. Использование этих коэффициентов для современной оценки проектов может не дать точных результатов, поэтому целью данной работы является оптимизация коэффициентов, используя генетические алгоритмы. У модели COCOMO есть три уровня и три режима, которые зависят от размера проекта и размера команды разработчиков. В данной работе рассмотрен только один базовый уровень и два режима – органический и полуразделённый. Оптимизация коэффициентов модели COCOMO не является новой задачей. Для решения задачи уже использовались нейронные сети, нечёткие алгоритмы, объектно-ориентированные методы и другие. В данной работе для оптимизации коэффициентов используются генетические алгоритмы. Генетические алгоритмы являются эволюционным методом для оптимизации. Для оценки популяций генетический алгоритм будет использовать набор атрибутов из нескольких проектов по разработке программного обеспечения. Это количество строк кода, трудозатраты и время реализации проекта. В целом, было использовано два набора данных, по одному на каждый режим модели COCOMO. Прогнозируемые моделью COCOMO трудозатраты будут сравниваться с реальными данными, таким образом оценивая функцию приспособленности индивида в популяции возможных решений. В ходе экспериментов была сформирована начальная популяция из десяти индивидов и произведена оптимизация коэффициентов. Используя стратегию трёхточечного скрещивания, в органическом режиме лучший результат был достигнут на 1500 итерации, в полуразделённом режиме лучший результат был достигнут на 1000 итерации. Для кодировки использовались целые числа от нуля до девяти. Сравнивая прогнозируемые трудозатраты и время разработки, которые были получены, используя существующие коэффициенты и новые коэффициенты, было констатировано, что новые коэффициенты дают более точные результаты.