Journal of Engineering and Applied Sciences 14 (24): 9842-9846, 2019
ISSN: 1816-949X

Analysis of Component based Metric using Non-Dominated Sorting based Genetic Algorithm

Sonal Gahlot and Rajender Singh Chhillar
Department of Computer Science, Maharshi Dayanand University, Rohtak, India
sonalght@gmail.com

Abstract: This study discusses a component based metric named as CI (Complexity of Interface), for the software complexity analysis at any stage of software development life cycle. The metric CI uses the coupling along with cohesion between different component of same or different modules, respectively. This metric also covers the anonymous classes and inner classes. The analysis of the metric is done by using the Non-dominated Sorting Genetic Algorithm 3 (NSGA-III) with discriminant analysis as the fitness function. The NSGA-III is the one of the latest and stable version modification of the Genetic algorithm used for optimization purposes. The analysis clearly shows that the CI metric computes the complexity of the interface effectively and can replace the existing coupling, cohesion metric metrics.

Key words: NSGA-III, Genetic algorithm, component based metric, coupling, cohesion, optimization

INTRODUCTION

Component based Software Engineering (CBSE) is in trend due to its immense ability to increase the reusability. The CBSE uses the existing components to develop new software's. The complexity of the software can be taken care off by selecting the appropriate component for the development (Garg and Lai, 2018). The component selection depends upon the coupling and cohesion of the component with the existing components. Coupling is the strength by which component of different modules are interconnected while the cohesion is interconnection between components of the same module. Basically, the cohesion shows the strength of the module due to its component while the coupling shows the bond between two modules. There is a trade-off between coupling and cohesion for the complexity as well as the quality of a software (Gandhi *et al*., 2018). A high-quality software should produce less complexity, coupling and high cohesion as low coupling and high cohesion reduces the testing and maintainability cost (Chowdhury and Zulkernine, 2011). That's why component having less coupling and high cohesion should be selected to improve the complexity of software (Ali *et al*., 2018). The coupling and cohesion among software component can be measured by using the software metric. Software metric plays a significant role to control the quality of software this is due to the fact that improvement needs measurement which is done through the metric. To control the software quality various software metric already has been by different authors by measuring the coupling, cohesion and the complexity of software.

LCOM is the lack of cohesion in methods is the one the CK suite metric used to measure the lack of cohesion by dividing the different methods with the total number of methods (Izadkhah and Hooshyar, 2017). LCOM metric is modified by various researchers, LCOM5 is the latest metric to define the lack of cohesion which is given by Eq. 1:

$$LCOM5 = \frac{m - nm*na}{nm - nm*na} \quad (1)$$

Here, nm, na are the number of method and number of attributes for any particular class while m gives the number of attributes used by each method of class commonly.

CAMC, Al Dallal and Briand (2010) metric is an acronym for the cohesion among methods in a class by computing the average of parameter occurrence metric. The CAMC metric is totally dependent on the parameter list of a method which generates the parameter occurrence metric by Eq. 2:

$$\partial = \sum_{a=1}^{m} \sum_{b=1}^{n} V_{ab} \quad (2)$$

Corresponding Author: Sonal Gahlot, Department of Computer Science, Maharshi Dayanand University, Rohtak, India, sonalght@gmail.com

Where:

∂ = The parameter occurrence metric

V_{ab} = The Value entry for the ath row and bth column which is 1 only if the parameter of a row a occurs the parameter list of the method of b column

The total parameter and method assumed in above are m, n, respectively. This parameter occurrence metric is used to calculate the CAMC metric by Eq. 3:

$$CAMC = average(\partial) \quad (3)$$

The CAMC calculated by Eq. 3 is for a class only and it the average values as denoted by the equation itself. The metric NHD (Counsell *et al*., 2006) stands for normalized hamming distance is the alternate method to measure the cohesion. It uses the count of methods that uses the common occurrence of the parameters. It is computed on the basis of the parameter agreement metric which is a lower triangular metric of the parameter occurrence metric. The NHD is computed by Eq. 4:

$$NHD = 1 - \frac{2}{mn(n-1)} \sum_{a=1}^{m} c_a(n - c_a) \quad (4)$$

Here, NHD is derived for any class and c_a is the number 1's in the ath column while m, n are total parameter and methods as already defined in the previous equation. Here, $2/mn(n-1)\sum_{a=1}^{m} c_a(n-c_a)$ calculates the parameter disagreement value when subtracted from 1 gives the NHD. The high cohesion gives the higher value of the NHD and NHD is more significant as compared to the CAMC metric (Izadkhah and Hooshyar, 2017). These the cohesion metric defined by various researchers, few coupling metrics to calculate the coupling among the components are as follow.

CoCC Poshyvanyk and Marcus (2006) is the conceptual coupling metric which is computed on the basis of conceptual similarity, between two classes (CSBC). CSBC depends upon the conceptual similarity between the class and method which is computed on the basis similarity between the methods of a class. It means the conceptual similarity can be calculated by evaluating the conceptual similarity between the methods. The CoCC is given by Eq. 5:

$$CoCC(c) = \frac{\left(\sum_{a=1}^{n} CSBC(c, b_a) \mid c \neq b_a\right)}{n-1} \quad (5)$$

Here, CSBC is the similarity between the classes and n is the number of classes. SSCM, Wanjiku (2017) is the structural and semantic coupling metric which includes the semantic as well as the structural relation between the methods and classes. The structural relation is the dependency of other entities on evaluated method and dependency of evaluated method on the other entities. The semantic relation uses the semantic data like identifiers and comments to find the relationship between two entities. SSCM is computed by the hybrid coupling between two classes that is evaluated on the basis of method pair coupling, direct dependency and the cosine relationship. It is given by Eq. 6:

$$SSCM = \frac{\left(\sum_{a=1}^{n} HCBC(c, b_a) \mid c \neq b_a\right)}{n-1} \quad (6)$$

Where:

HCBC = The Hybrid Coupling Between Classes

n = The number of classes

These the existing metric to compute the coupling and cohesion effectively but no metric has been defined to compute the complexity directly on the basis of coupling and cohesion metric. This study designs a metric to compute the complexity of any interface discussed in next section:

CI metric: The existing metric either measure the cohesion or the coupling efficiently. The complexity of any project computed through an existing metric can be improved by focusing on the coupling as well as the cohesion. Moreover, the coupling of the attributes and methods in the inherited classes and parent classes along with static import affects the complexity of the project. The complexity of the project is also affected by the inner classes along with anonymous classes. The discussed metric considers all the above factors and computes the complexity of the project efficiently. The CI metric is given by Eq. 7:

$$CI = \frac{\sum CC_o + \sum CC_i + \sum CC_a +}{n_1 + n_2 + n_3} \quad (7)$$

Equation 7 gives the complexity of interface metric which is calculated by computing the class complexity of each outer Class (CC_o), class complexity of each inner Class (CC_i) and class complexity of each anonymous Class (CC_a) where the CC_o, CC_i, CC_a is given by Eq. 8-10, respectively. The sum of the CCo, CCi, CCa is divided by the sum of n_1-n_3 where, n_1-n_3 are number of outer classes, number of inner classes and number of anonymous classes, respectively. The value of CI varies from 0-1 and the value close to 0 exhibits low complexity projects and vice-versa:

$$CC_o = \vartheta_1*NHD+\vartheta_2*LCOM5-\vartheta_3*SSCM-\vartheta_4*CoCC+\vartheta_5*IC \quad (8)$$

Such that, $\vartheta_1+\vartheta_2+\vartheta_3+\vartheta_4+\vartheta_5 = 1$. NHD, LCOM5, SSCM, CoCC metric are given by Eq. 1, 4, 5, 6, respectively. The metric IC is computed by the coupling and cohesion between the components of parent and the child class. It is given by Eq. 9:

$$IC = \frac{O_m+a_{co}}{m_p+a_p+m_c+a_c} \quad (9)$$

Where:

O_m = The Overridden methods
a_{co} = The common attributes
m_p, a_p, m_c, a_c = The methods and attributes of parent and child class, respectively

The CC_i is given by Eq. 10:

$$CC_i = \vartheta_1*NHD_i+\vartheta_2*LCOM5_i-\vartheta_3*SSCM_o-\vartheta_4*C_oCC_o \quad (10)$$

Such that, $\vartheta_1+\vartheta_2+\vartheta_3+\vartheta_4 = 1$. In the inner class SSCM, CoCC is calculated only for the outer class while NHD and LCOM is computed for inner class only as shown in Eq. 10. In the anonymous class, there is no inheritance and the coupling so only cohesion is computed as shown in Eq. 11:

$$CC_a = \vartheta_1*NHD_a+\vartheta_2*LCOM5_a \quad (11)$$

Such that, $\vartheta_1+\vartheta_2 = 1$. The NHD and LCOM5, both are computed only for the anonymous class only as given by Eq. 11. This CI metric computes the complexity of any software at any stage of SDLC.

MATERIALS AND METHODS

Genetic algorithm: Genetic Algorithm (GA) is based on the genetic selection process to exploit the search space, effectively. GA operates on a set of population instead of single/individual instances. It uses the selection, mutation, crossover like operators to complete its process. The process starts by selecting the random population/solution out of the set of possible solutions. This random solution undergoes the selection process to select a part of the initial population. This step is followed by the mutation and crossover step which introduces the randomization. The evaluation is done to select the better solution based on fitness value of current and previous solution. This process is repeated until the stopping criteria is satisfied (Tseng *et al*., 2005; Deb *et al*., 2002). The process is also shown using the flowchart in Fig. 1.

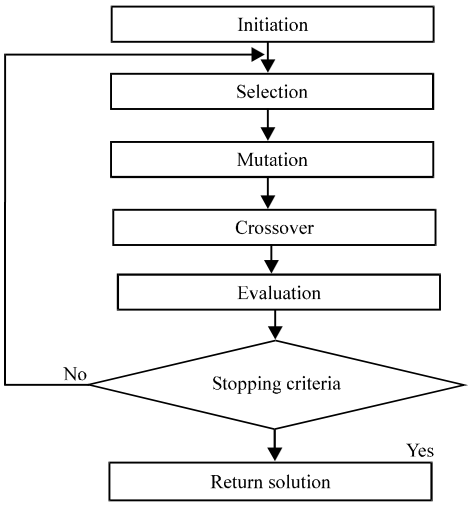

Fig. 1: Genetic algorithm

Figure 1 shows the process of Genetic algorithm. However, with the time various improved versions of genetic algorithms has been introduced. This study discusses one of the most stable version of Genetic algorithm, i.e., Non-dominated Sortingbased Genetic Algorithm (NSGA), described in next section.

NSGA: NSGA is one of the earliest multi objective evolutionary algorithm which is designed by modifying the Genetic algorithm. It performs the non-dominated sorting to place all the dominating solutions in front of the particular solution to select only dominating solution resulting diversification. This implements the exploration to improve the performance as compared to the Genetic algorithm. The NSGA consumes lot of computation time to perform the non-dominated sorting. The more stable version of NSGA are NSGA-II and NSGA-III which reduces the computation time by altering the non-dominating sorting method. NSGA-II also modifies the selection process to improve the performance. NSGA-III further improves the performance by selecting a reference point to partition the population. The complete detail of NSGA-III has been discussed in next subsection.

NSGA-III: The process of NSGA-III (Yi *et al*., 2018) initiates with generation ofreference points based on the number of objectives to be achieved. Then, the reference points are normalized and referenced to the closet objective based on the perpendicular distance of reference point to the objective. After selection of reference point, selection, crossover as well as mutation is applied to the parent and offspring population. The resultant of previous step is non dominated sorted similar

to the previous version of NSGA to select front end of the population. The complete details of the process are given in NSGA-III algorithm.

NSGA-III algorithm (P, RFP):

//Here, Pop is the Parent population and RFP is the Reference Points

1. iter = 0
2. Pop_{iter} = P
3. While termination criteria not satisfied
   a. mi = 1
   b. $sol_{iter} = \varnothing$
   c. $Opop_{iter}$ = Crossover and mutation (Pop_{iter})
   d. $TPOP_{iter} = POP_{iter} \cup OPP_{iter}$
   e. $TPOP_{iter}$ = Non dominated sort ($TPOP_{iter}$)
   f. While (mi<=N)
   i. $Sol_{iter} = Sol_{iter} \cup TPOP_{iter}(mi)$
   ii. mi = mi+1

   end while
g. if(Length(Sol_{iter} == N)
   i. $Pop_{iter+1} = Sol_{iter}$

   else
   i. $p_1 = TPop_{iter}(mi-1)$
   ii. NRFP = Select reference point with respect to p_1
   iii. NRFP = Normalize(NRFP)
   iv. Pop_{iter+1} = Select members on the basis of NICHING by using p_1 and NRFP

   end if
h. iter = iter+1

   end while
4. Return Pop_{iter-1}

The above specified algorithm executes the process of NSGA-III until the termination criteria is not satisfied and generates a new population in each iteration. In each iteration, crossover and mutation is applied to the parent population which is proceeded by non-dominated sorting to select the front N members to generate a new solution. If the new solution is not sufficient then a reference point based population is generated. The reference point is selected on the basis of last member, i.e., dominating member and the normalization is followed by NICHING process to generate a new population.This process gives the better solution and complete process gives the optimized solution. This study applies the concept of NSGA-III in the field of software engineering to ensure the quality of product discussed in next section.

RESULTS AND DISCUSSION

Implementation and analysis: The implementation of the proposed metric has been done by using the JPeek and Code MR tool. The JPeek tool is used to compute the cohesion metric value, i.e., NHD, LCOM5. The SSCM and CoCC is computed by MATLAB program with takes the input from Code MR tool. The validation using NSGA-III is done using the MATLAB. The implementation has been done on different academic and industrial projects. The details of the projects are given in Table 1.

Table 1: Projects used for analysis

Project name	Description (Project)	Application
AutoCAD-D	Academic	Mechanical drawing
Facebook-D	Academic	Social media
Friends Tracker-D	Academic	Social media
Web Intranet Manager	Academic	ERP
jFreeChart	Industrial	Representation and charting
Finding Bugs	Industrial	Code analyzer
HTML Parser	Industrial	Code parser

Table 1 shows the name of projects for analysis. The description represents that whether the project is academic or industrial. All the academic projects have been designed by the students of DPG institute of technology and management, gurugram. Different academic project covers the project of different application area's like AutoCAD-D is a replica of the original AutoCAD with limited functions used to design the different machine drawings. Similarly, the Facebook-D is the replica of original Facebook with less functionality covers the social media application. The Friends Tracker-D are the applications for social media and face recognition as shown in Table 1. The jFree Chart is the industrial project available over internet to draw the charts on given data. Web intranet manager is a ERP project design by a startup company to provide the intranet interaction and sharing between any company employees. Finding bugs ia code analyzer project to find the bugs in a code. This project is also availed through internet. These project has been evaluated using the tools specified above to determine the NHD, LCOM5, MMAC SSCM, CoCC metric. The details of the projects are mentioned in Table 2.

Table 2 shows the number of classes along with the number of packages in the project being evaluated. The classes which are imported through the external packages are also given in Table 2. The line of code to estimate the size of project is also shown in Table 2. The overall analysis shows that evaluation project varies in size from 446 line of code to 90152 line of code. This is done to show the scaling factor that evaluation can be done easily on any project. Table 3 shows the value of proposed metric for the projects shown in Table 1. The evaluation of the projects for the CI metric is shown in Table 3.

CI metric calculated for the mentioned seven projects computes the complexity of the projects. The evaluation of the projects also has been done on existing state of art coupling and cohesion metric given in Table 4. The correlation between the metric has been derived using the NSGA-III shown in Table 5.

The correlation values derived for each metric are higher than the 0.6 which shows the high correlation between the CI and the other existing state of art metric. This shows that the CI metric can easily replace all existing state of art metric. This proves the significance of the CI metric.

Table 2: Detail of projects used for analysis

Project name	No. of classes	No. of packages	No. of external packages	No. of external classes	Line of code
AutoCAD-D	19	9	21	149	1629
Facebook-D	6	1	11	45	446
Friends Tracker-D	47	8	26	112	2063
Web intranet manager	15	1	5	17	1327
jFreeChart	650	40	38	269	71692
Finding Bugs	1426	55	60	714	90152
HTML Parser	14	1	4	16	648

Table 3: Proposed metric for projects

Project name	CC_o	CC_i	CC_a	CI
AutoCAD-D	0.315	0.080	0.047	0.147
Facebook-D	0.764	0.193	0.115	0.357
Friends Tracker-D	0.949	0.240	0.142	0.444
Web intranet manager	0.568	0.144	0.085	0.266
jFreeChart	0.062	0.016	0.009	0.029
Finding Bugs	0.137	0.035	0.021	0.064
HTML Parser	0.230	0.058	0.034	0.108

Table 4: Existing state of art metric analysis

Project name	NHD	LCOM5	MMAC	SSCM	CoCC
AutoCAD-D	0.00	0.00	0.00	0.00	0.10
Facebook-D	0.00	0.50	0.00	0.50	0.12
Friends Tracker-D	7.12	0.79	0.50	2.85	4.45
Web Intranet Manager	2.50	0.50	0.50	0.50	0.85
jFreeChart	7.06	0.50	0.54	0.00	4.02
Finding Bugs	5.39	1.28	0.79	5.25	9.50
HTML Parser	1.17	0.72	0.50	4.12	3.87

Table 5: Correlation of CI with existing state of art metric

Project name	Correlation with CI
NHD	0.87
LCOM5	0.67
MMAC	0.78
SSCM	0.56
CoCC	0.76

CONCLUSION

This study discusses the CI metric to calculate the complexity of any project at any stage of SDLC and calculate the same for the seven industrial and academic projects. The study also, calculates the existing coupling and cohesion metric namely NHD, LCOM5, MMAC, SSCM, CoCC for the same projects and computes the correlation of the CI metric by using the NSGA-III. The NSGA-III uses the discrimination analysis as the objective function to compute the correlation of the CI metric with the existing metric. The high correlation clearly signifies that the CI metric effectively computes the complexity of any project.

REFERENCES

Al Dallal, J. and L.C. Briand, 2010. An object-oriented high-level design-based class cohesion metric. Inf. Software Technol., 52: 1346-1361.

Ali, S., M. Abdellatief, M.A. Elfaki and A. Wahaballa, 2018. Complexity metrics for component-based software systems: Developer perspective. Indian J. Sci. Technol., Vol. 11, 10.17485/ijst/2018/v11i32/123093

Chowdhury, I. and M. Zulkernine, 2011. Using complexity, coupling and cohesion metrics as early indicators of vulnerabilities. J. Syst. Archit., 57: 294-313.

Counsell, S., S. Swift and J. Crampton, 2006. The interpretation and utility of three cohesion metrics for object-oriented design. ACM Trans. Software Eng. Methodol., 15: 123-149.

Deb, K., A. Pratap, S. Agarwal and T. Meyarivan, 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput., 6: 182-197.

Gandhi, P., K. Vashisht, K. Dawra, S. Jaitly and P. Banerjee, 2018. Component based software development- An efficient approach. Intl. J. Sci. Eng. Sci., 2: 26-32.

Garg, M. and R. Lai, 2018. A method for selecting a model to estimate software reliability at the design phase of component-based real-time system development. J. SoftWare, 13: 317-334.

Izadkhah, H. and M. Hooshyar, 2017. Class cohesion metrics for software engineering: A critical review. Comput. Sci. J. Moldova, 25: 44-74.

Poshyvanyk, D. and A. Marcus, 2006. The conceptual coupling metrics for object-oriented systems. Proceedings of the 22nd IEEE International Conference on Software Maintenance, September 24-27, 2006, Philadelphia, PA., pp: 469-478.

Tseng, T.L.B., W.Y. Liang, C.C. Huang and T.Y. Chian, 2005. Applying genetic algorithm for the development of the components-based embedded system. Comput. Standards Interf., 27: 621-635.

Wanjiku, R.N., 2017. Software product quality assessment using Scoped Class Cohesion Metric (SCCM). MSc Thesis, Jomo Kenyatta University of Agriculture and Technology, Juja, Kenya.

Yi, J.H., S. Deb, J. Dong, A.H. Alavi and G.G. Wang, 2018. An improved NSGA-III algorithm with adaptive mutation operator for big data optimization problems. Future Gener. Comput. Syst., 88: 571-585.