UNIOSUN Journal of Engineering and Environmental Sciences. Vol. 3, No. 1. March. 2021
DOI: 10.36108/ujees/1202.30.0110

The Correlation among Cognitive Complexity Metrics in Algorithm Analysis

Isola, E. O., Ogundoyin, I. K., Akanbi, C. O. and Adebayo, O. Y.

Abstract: In the early stage of software development, design complexity metrics are considered as useful indicators of a software testing effort and quality attributes. However, existing works made great efforts in establishing standardized metrics to evaluate the complexity of software, but there have not been significant efforts in finding the correlations among the cognitive complexity metrics. To address this challenge, this paper reviewed cognitive complexity metrics which includes: Improved Cognitive Complexity Measure (ICCM), New Cognitive Complexity of Program (NCCoP) and Modified Cognitive Complexity Measure (MCCM). The metrics were employed to analyse some selected sorting algorithms implemented in a procedural C programming language. The relationships among the aforementioned metrics were calculated using the Pearson Correlation Coefficient Method. The results of the comparative examination of ICCM, NCCoP and MCCM revealed that ICCM had more responsive measurements and that there exists a strong relationship among the specified metrics. ICCM had the strongest significance among the considered metrics based on the efforts in comprehending the information contained in the sorting algorithm codes. The study contributed significantly to understanding and addressing the complexity emanating from software development.

Keywords: Software complexity metrics, Algorithm, programming language, pearson correlation, C programming

I. Introduction

Software complexity is part of software engineering that deals with both the internal and external quality of software. It creates room for evaluation of system performance in term of its effectiveness, maintenance, reusability, testability, and modifiability. Most people demand for software of better features and improved qualities. Software complexity is the level to which a system or component has a design that is complex to comprehend and validate [1]. High complexity may result in more errors and technical hitches in maintenance, understandability, modification and testing effort [2]. Consequently, there has been a great effort in establishing standardized metrics to evaluate the complexity of software. While some useful metrics have been proposed to validate the software complexity [3], for instance, Isola et al. in [4] proposed an improved cognitive complexity measure (ICCM), in the work variable name used in code form an integral part in understanding the code. Arbitrarily Named Variable (ANV), Meaningful Meaning Variable (MNV) and Cognitive weight of Basic Control Structure (BCS) were employed in calculating the effort needed in understanding the information contained in the software. The authors defined ICCM as contained in eq. 1

$$CCM = \sum_{K=1}^{LOCx} \sum_{V=1}^{LOCx} (3ANV + MNV) * Wc(k) \quad (1)$$

where; the first summation is the line of code from 1 to the last Line Of Code (LOC). Premised on this, every Basic Control Structure (BCS) is assigned a cognitive weight. Either all the BCS's are in a linear layout or some BCS's are embedded in others. For the former, the sum of the weights of all n BCS's

Isola, E. O., Ogundoyin, I. K., Akanbi, C. O. and Adebayo, O. Y. (Department of Information & Communication Technology, Osun State University, Osogbo, Osun State, Nigeria)
Corresponding Author: esther.isola@uniosun.edu.ng
Telephone Number: +2348033660008
Submitted:02-Oct-2020
Accepted: 01-Feb-2021

are added and for the later, cognitive weights of inner BCS's are multiplied by the weights of external BCS's [5]. Table 1 shows different types of BCSs, with the corresponding dedicated weight.

Furthermore, Jakharand in [6] and Rajnish in [7] worked on New Cognitive Complexity of Program Measure (NCCoP). The authors explained NCCoP as a group of information enclosed in the identifiers or variables. Thus, the complexity completely depends on the variables and the internal control structure BCSs. Therefore, NCCoP method is to measure the cognitive complexity of a program [6, 7]. In the works, operators are not well thought-out, the number of variables and constants are just to be added up line by line and multiplied by its BCSs weight. The merit of this is that the weight of each LOC[8] can be used in counting the utmost weight of a unit to diminish the chances of severe errors due to the higher complexity of a module. In the work, the authors defined NCCoP as formulated in eq. 2.

$$NCCoP \quad \sum_{K=1}^{LOCs} \sum_{V=1}^{LOCs} Nv * \quad (K) \quad (2)$$

where LOCs is the number of lines in the code, Nv is the number of variables in a particular line of code, Wc is the weight (as shown in Table 1) corresponding to the particular structure of the line.

Misra in [9] proposed a Modified Cognitive Complexity Measure (MCCM), In the work, the considered cognitive[10] metrics were applied to some selected sorting algorithm codes written in C programming language and then correlation was found among the metrics. The proposed MCCM was formulated as contained in eq. 3

$$MCCM = (N_{i1} + N_{i2}) * W_c \quad (3)$$

Where: N_{i1} is the total number of operators and N_{i2} is the total number of operands.

Existing works made great effort in establishing standardized metrics to evaluate the complexity of software but there have not been effort to find the correlations among the complexity metrics. This paper therefore, addresses the problem of correlation among the complexity metrics using ICCM, NCCoP and MCCM by applying the metrics to some selected sorting algorithms implemented in C programming language in order to establish correlation among the metrics

Table 1: BCSs with its Cognitive Weights (Wc)

Category	BCS	CWU
Sequence	Sequence	1
Condition	If-else / Switch	2
Loop	For / For-in	3
	While/do…While	
Functional	Functional- call	2
activity	Alert/ prompt throw	
Exception	try-catch	1

Source: [8]

II. Materials and Methods

The ICCM, NCCoP and MCCM metrics given in eq.1, eq.2 and eq.3 were implemented on merge sort algorithm, heap sort algorithm, selection sort algorithm, insertion sort algorithm and bubble sort algorithm. Table 2 shows how ICCM was calculated for a bubble sort algorithm

I. Results and Discussion

The cognitive complexity values for ICCM, NCCoP and MCCM for five (5) sorting algorithms are shown in Table 3. The graph for comparison among the cognitive complexity metrics are depicted in Figure. 1

A. The Correlation among Cognitive Complexity Metrics

This study makes use of Pearson product-moment correlation in testing whether there is any relationship among ICCM (Improved Cognitive Complexity Measure), NCCoP (New Cognitive Complexity of Program)

Measure and MCCM (Modified Cognitive Complexity Measure).

Table 2: Evaluation of Implementation for Bubble Sort Algorithm using ICCM

S/N	CODE	3ANV + MNV	Wc	ICCM
1	#include <stdio.h>	0	1	0
2	void swap(int *xp, int *yp)	2	1	2
3	{	0	1	0
4	int temp = *xp;	2	1	2
5	*xp = *yp;	2	1	2
6	*yp = temp;	2	1	2
7	}	0	1	0
8	void bubbleSort(intarr[], int n)	5	1	5
9	{	0	1	0
10	int i, j;	7	1	7
11	for (i = 0; i < n-1;	12	3	36
12	for (j = 0; j < n-i-1; j++)	12	3	36
13	if (arr[j] >arr[j+1])	8	2	16
14	swap(&arr[j], &arr[j+1]);	8	1	8
15	}	0	1	0
16	void printArray(intarr[], int size)	4	1	4
17	{	0	1	0
18	int i;	4	1	4
19	for (i=0; i < size; i++)	10	3	30
20	printf("%d " , arr[i]);	7	1	7
21	printf("n");	3	1	3
22	}	0	1	0
23	int main()	1	1	1
24	{	0	1	0
25	intarr[] = {64, 34, 25, 12, 22, 11, 90};	2	1	2
26	int n = izeof(arr)/sizeof(arr[0]);	5	1	5
27	bubbleSort(arr, n);	4	1	4
28	printf("Sorted array: \n");	3	1	3
29	printArray(arr, n);	4	1	4
30	return 0;	0	1	0
31	}	0	1	0
	TOTAL			183

Line 1: There is no variable. 0
Line 2: There are 2MNV. 2
Line 3: There is no variable. 0
Line 4 to 6: There are 2MNV. 2
Line 7: There is no variable. 0
Line 8: There is 1ANV and 2MNV. 3(1) + 2 = 5
Line 9: There is no variable. 0
Line 10: There are 2ANVand 1MNV.= 7
Line 11: There is 4ANV. 3(4) = 12
Line 12: There is 4ANV. 3(4) = 12.
Line 13: There is 2ANV and 2MNV. 3(2) + 2 = 8
Line 14: There is 2ANV and 2MNV. 3(2) + 2 = 8
Line 15: There is no variable. 0
Line 16: There are 4MNV. = 4
Line 17: There is no variable. 0
Line 18: There is 1ANV and 1MNV. 3(1) + 1 = 4
Line 19: There is 3ANV and 1MNV. 3(3) + 1 = 10
Line 20: There is 2ANV and 1MNV. 2(3) +1 = 7
Line 21: There is 1ANV. 3(1) = 3
Line 22: There is no variable. 0
Line 23: There is 1MNV. 1
Line 24: There is no variable. 0
Line 25: There is 2MNV. 2
Line 26: There is 1ANV and 2MNV. 3(1) + 2 = 5
Line 27: There are 1ANV and 1MNV= 4
Line 28: There is 1ANV. 3(1) = 3
Line 29: There are 1ANV and 1MNV. = 4
Line 30 to 31: There is no variable = 0

"The sign and the absolute value of a correlation coefficient describe the direction and magnitude of the relationship between two variables.

Table 4 shows that the relationship between ICCM and NCCoP is strong and positive. The analysis presented in Table 4 reveals that the

relationships between ICCM and NCCoP are significant since the P-value is 0.0001.

Table 3: Complexity values for different measures

ALGORITHM	ICCM	MCCM	NCCoP
Merge sort algorithm	279	201	116
Heap sort algorithm	236	177	99
Selection sort algorithm	162	108	66
Bubble sort algorithm	183	74	61
Insertion sort algorithm	127	100	54

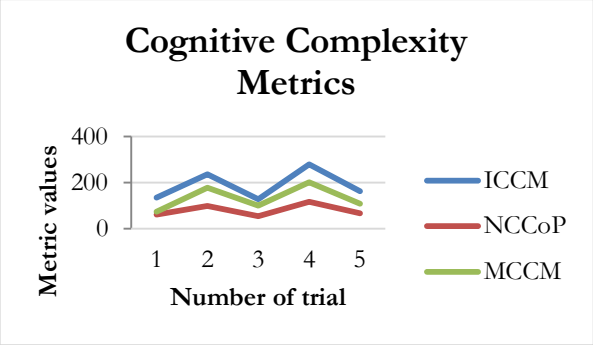

Figure 1: Relative comparison graph of Cognitive Complexity Metrics

A strong positive relationship (r=0.976) is also recorded between ICCM and MCCM as presented in Table 4. The (0.005) P-value shows that the relationships between ICCM and MCCM are significant. Since the three (3) metrics have a strong relationship with each other, this implies that any combination of the metrics can be used for the reusability and maintenance of the software.

B. Regression Coefficient of Sorting Algorithm

In regression with multiple independent variables, the coefficient tells about how much the dependent variable is expected to increase when the independent variable is expected to increases by one, holding all other independent variables constant. Multiple linear regression has more than two independent variables are used to predict the value of a dependent variable. The strengths of effect of Improved Cognitive Complexity metrics applied to sorting algorithm written in C language was determined by the multiple regression coefficients as presented in Table 5, NCCoP has the strongest significant effect on ICCM with a standardized estimate of 0.785. The strengths of the effect of New Cognitive Complexity of Program metrics applied to sorting algorithm written in C language was determined by the multiple regression coefficients as presented in Table 6. Where ICCM has the strongest significant effect on NCCoP with a standardized estimate of 1.179.

The strengths of the effect of Modified Cognitive Complexity Measure applied to sorting algorithm written in C language was determined by the multiple regression coefficients as presented in Table 7. According to Table 7 ICCM has the strongest significant effect on MCCM with a standardized estimate of 2.247.

C. Discussion

In this work, a series of experiments were carried out to investigate the relationship among some selected cognitive complexity metrics. The cognitive complexity values of the five (5) selected sorting algorithms were summarized in Table 3. Table 5 showed the statistics that were calculated in analysing C codes to appraise ICCM, NCCoP and MCCM measures. Merge sort algorithm had the maximum value of complexity which was (ICCM = 279), this indicated that the Merge sort algorithm had the highest complexity information among the selected sorting algorithm codes.

Table 4: Pearson Correlation of Cognitive Complexity Metrics

		ICCM	NCCoP	MCCM
ICCM	Pearson Correlation	1	.996**	.976**
	Sig. (2-tailed)		.000	.005
	N	5	5	5
NCCoP	Pearson Correlation	.996**	1	.963**
	Sig. (2-tailed)	.000		.008
	N	5	5	5
MCCM	Pearson Correlation	.976**	.963**	1
	Sig. (2-tailed)	.005	.008	
	N	5	5	5

Correlation is significant at the 0.01 level (2-tailed)

Table 5: Regression Coefficient of Dependent Variable ICCM for each sorting Algorithm written in C Language

Model		**Unstandardized Coefficients**		**Standardized Coefficients**		**Sig.**
		B	Std. Error	Beta T		
1	**(Constant)**	-2.978	9.710		-.307	.788
	NCCoP	1.955	.392	.785	4.990	.038
	MCCM	.271	.194	.219	1.394	.298

Table 6: Regression Coefficient of Dependent variable NCCoP for each sorting algorithm written in C language

Model		**Unstandardized Coefficients**		**Standardized Coefficients**		**Sig.**
		B	Std. Error	Beta T		
1	**(Constant)**	2.613	4.527		.577	.622
	MCCM	-0.93	.117	-.187	-.791	.512
	ICCM	.473	.095	1.179	4.990	.038

Table 7: Regression Coefficient of Dependent variable MCCM for each sorting algorithm written in C language

Model		**Unstandardized Coefficients**		**Standardized Coefficients**		**Sig.**
		B	Std. Error	Beta T		
1	**(Constant)**	-5.679	25.451		-.223	.844
	ICCM	1.820	1.306	2.247	1.394	.298
	NCCoP	-2.574	3.252	-1.276	-.791	.512

MCCM with a value of 201 and NCCoP with a value of 116 were also able to show that, but ICCM consider the effort for comprehending the code and the information contained in software. ICCM for Insertion sort algorithm had the least value of 127 which implied lesser complexity information in how the user can simply comprehend some functions in the code, MCCM with a value of 100 was able to show it too but NCCoP with value of 54 was not able to show this. ICCM gave exact result compared to NCCoP and MCCM because ICCM considered the effort for comprehending the code and information contained in software. Since the three (3) metrics had strong relationship with each other, this implied that any combination of the metrics can be used for the reusability and maintenance of the software.

III. Conclusion

The result of correlation among cognitive complexity showed that ICCM demonstrated the complexity of the program undoubtedly and accurately than other considered cognitive measures. The metrics were evaluated by dissimilar online sorting algorithm codes written in C programming language to establish reusability and maintenance

measures and also, that there exists a level of correlation among the measures. The comparative examination of the implementation of ICCM against NCCoP and MCCM revealed that ICCM had more responsive measurement and that there exists a strong relationship among the specified metrics. ICCM had the strongest significant effects on MCCM, NCCoP and ICCM considering the efforts in comprehending the information contained in the code.

References

[1] Kehinde, A.S., Monsurat, O.B., Isola, E. O., Olabiyisi, S.O., Omidiora, E.O. and Oyeleye, C.A. "Object Oriented Programming Languages For Search Algorithms in Software Complexity Metrics", *International Research Journal of Computer Science* , vol. 6, no. 4, 2019, pp. 90–101.

[2] Syed, T.R. and Maheswaran, K. "Software Cognitive Complexity Metrics for OO Design: A Survey", *International Journal of Scientific Research in Science, Engineering and Technology (ijsrset.com),* vol.3, issue 3, 2017, pp. 691 – 698.

[3] Kushwaha D.S. and Misra A.K (2006): "A modified cognitive information complexity measure of software", https://doi.org/10.1145/1108768.1108776, Accessed January 2006.

[4] Isola, E.O, Olabiyisi, S.O, Omidiora, E.O, Ganiyu, R.O, Ogunbiyi, D.T and Adebayo, Y.O "Development of an Improved Cognitive Complexity Metrics for Object Oriented Codes", *British Journal of Mathematics & Computer Science*, vol. 18, no. 2, 2016, pp. 1 – 11.

[5] Isola, E., Olabiyisi, S., Omidiora, E. and Ganiyu, R. "Performance Evaluation of Improved Cognitive Complexity Metric and Other Code Based Complexity Metrics", *Anale. Seria Informatică.* vol. XVI, series 16th, 2018, pp. 114–119.

[6] Jakhar, A.K., and Rajnish, K. "A New Cognitive Approach to Measure the Complexity of Software", *International Journal of Software Engineering and its Applications*, vol. *8*, no. 7, 2014, pp. 185–198.

[7] Jakhar, A.K. and Rajnish, K. "Measuring Complexity, Development Time and Understandability of a Program: A Cognitive Approach", *International Journal of Information Technology and Computer Science*, vol. 6, no. 12, 2014, pp. 53–60.

[8] Arockia, S. and Aloysius, A. "Aspect Oriented Programming - Cognitive Complexity Metric Analysis Tool", *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 3, Issue 1, 2018, pp. 480 – 486.

[9] Misra, S. "Modified Cognitive Complexity Measure", *Computer and Information Sciences – International Symposium,* Istanbul Turkey, November 1 – 3, 2006, pp. 1050-1059.

[10] Jakhar, A.K and Kumar, R., "A cognitive measurement of Complexity and Comprehension for Object -oriented Code", *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol.10, no.3, 2016, pp.643-650.