Evaluating Python, C++, JavaScript and Java Programming Languages Based on Software Complexity Calculator (Halstead Metrics)

Evaluating Python, C++, JavaScript and Java Programming Languages Based on Software Complexity Calculator (Halstead Metrics)

Sabah A. Abdulkareem[1] and Ali J. Abboud[2]

[1,2] Department of Computer Engineering, University of Diyala, Diyala, Iraq

E-mail: sabahanwer55@gmail.com

Abstract. Various software organizations used software metrics to assessing and assuring operation, maintenance, and quality of software codes. Halstead is an essential type of software complexity metrics used to measure source code complexity. We presented a comparative analysis study using this metric to benefit software testing process by showing the possibility of software metrics to measure the characteristics of the software in all its aspects. Halstead metric is used to analyse the written code in python, C, JavaScript and Java programming languages. It provides a better tool to evaluate the complexity level of the language and displays the differences levels of code complexity. The conducted experiments show that python is the simplest programming language and Java is the difficulty and more complex language than others. These results benefit the automation in software metrics computation to decide which programming language can produce high quality and the less complexity software.

Keywords: Software Complexity, Halsted Metric, Programming Languages

1. Introduction

The software development process is going through several interrelated phases of planning, design, and implementation and testing. Measuring the complexity of these software phases is an important aspect of the software industry to develop an efficient and robust software applications. The software complexity means, according to the IEEE standard is the exact scientific basis for software engineering [1]. The comprehensive understanding the software code is directly linked to the degree of software complexity. Also, the hardness of analyzing software is playing an important role in software development manufacturing [2]. Then, we can define software complexity is the amount of computational efforts needed to develop, maintain and execute software code [3].
Halstead metrics and McCabe's Cyclomatic complexity are the most popular metrics for measuring the density of the code [4]. Halstead scales are based on the mathematical relationships between the number of operators and operands [5], while Cyclomatic Complexity is based on the flow control course. Halstead complexity measures are software metrics invented by Maurice Howard Halstead in 1977 [6]. It was stated that the software metrics should reflect the implementation of algorithms in different languages, however, these metrics must be independent of their implementation and statically calculated from the code. The main objective of Halstead was to define the measurable characteristics of software, and the relations among them. Halstead metrics are capable of measuring the density of the program code

efficiently and they have better support tools [4]. The rest of the paper is organized as follows: section II is used for literature survey while section III is devoted to explaining Halstead complexity metrics and their characteristics. Section IV is used to present experimental results analysis with examples. Finally, conclusions and future work are presented in section V.

2. Literature survey

The four most used programming languages in the industry and academia are C, C++, C# and Java. C is an appropriate language for developing real time applications and products for industrial automation, while C++ is suitable programming language for developing high performance applications. Also, C# is a good language for the development web applications. Finally, Java programming language has three different forms of programming, including J2SE, J2ME and J2EE. J2SE is a proper Java form for developing desktop applications and J2ME is a suitable candidate for developing wireless embedded systems applications. However, J2EE is a suitable Java form for developing server applications [7-9].
We have used these four programming languages in our study with the aim to compare their performance. The main ingredients of programming language performance are operation speed, code flexibility and efficiency. For example, Java is an object oriented programming language that lacks important features such as pointers, multiple inheritance, and templates. Such shortage makes the Java programming language less robust than C++ [10]. However, Java can be used to write programs which can run in various environments. Finally, we can conclude that C++ programs execute more quickly than Java programs, but C# or Java programs are more portable than C++ counterparts [11].
In 1979, Bjarne Stroustrup invented C++, in New Jersey, Bell Laboratories in Murray Hill. In 1983 changed this name to C++. Stroustrup constructs C++ based on C, containing all of C's attributes, features, and benefits. It is important to recognize that Stroustrupd is not constructed completely new programming language, but he promoted to strongly successful language, this language is C++. C++ is a hybrid object-oriented programming language. The object oriented of C++ attributes can be impact effective used for practical tasks of any programming. It is not used C++ for projects like databases, editors, systems of personal file, programs of communication, and networking utilities. In most cases C++ is a favorite language for Windows programming [12].
On July 1991, James Gosling starts in the project coined name "Oak"[13], this project is renamed to Java in 1995. Java is an object oriented language, portable, simple, in Sun Microsystems labs was designed by research staff and developed by James Gosling. Syntax in Java to some extent is similar to syntax in C++ [14], Java has Interfaces with multiple inheritance, typical, concurrency Support, statically typed, powerful libraries, type safe, garbage sets and etc. standard library of Java include comprehensive I/O facilities, a GUI toolkit, distributed computation support, cryptographic security, and date/time support. Java had a deep influence on the Internet. Additionally to develop web programming in general, called the applet which is a new type of networked program and is a significant type of Java [15]. Python is a robust, high level, open source, and general purpose programming language. It has advanced a massive system of useful libraries: NumPy and pandas. NumPy is one of the Python libraries that permits the effective dealing and operating on data. Pandas are another library of the Python libraries which can manage and operate tabular data structures and other on time series data. It permits implementation of even complex data analytics tasks on bigger data collections [16,17]. In1995, Brendan Eich invented JavaScript and Netscape. JavaScript is a script language that can be written in HTML. The essence JavaScript similar to C, C++, and Java. JavaScript is generally used in web browsers to achieve interactive with the user, modify the content of the document that shows within the web browser window. The JavaScript version which runs within HTML web pages called client-side JavaScript to confirm that client computer who runs scripts instead of the web server [18].
In the new research studies on the usage of complexity measures, Vard et al. [19] show by conducting survey on the universities and companies that two major characteristics affect the complexity of the software. Eventually, this increase in the complexity will increase the maintenance time of the software. Also, Hariprasad et al. [20] uses Halstead metric to quantify the complexity of two programming languages (C++ and python) in terms of execution time and bugs. Their method is limited to few

languages and did not give an accurate indication of metric values. Tero et al. [21] proposed a new complexity evaluation method based on the functional diagrams to study the reliability of the software. This method consists of several functions used to check each component of software that used to protect power plant. Anh [22] proposed a systemic review of the influence of software complexity metrics on the design and testing of software systems. Makismo et al. [23] proposed novel method to analyze the source code effectiveness using software complexity metrics. Such analysis is necessary to detect the malware behavior of software applications that threatens modern computing devices and systems. Iwan et al. [24] use the Halstead metric to quantify the quality of different version of the software. They found in their study that later versions of software are more stable. Safa et al. [25] have designed a new method to predict faulty parts in the software systems using fixed analysis tools and software complexity metrics. This method can also determine the quality and reliability of application software's. Sibel and Selcuk [26] proposed a new method to measure the complexity of software by using neural networks. Their method represents a good alternative to the existing models of software complexity measurement. Madhan et al. [27] conducted a research study to prove the viability software complexity measures to quantify the complexity of genetic and Anti colony optimization algorithms. Subali and Rochimah [28] proposed novel method to assess the complexity of database systems written in the SQL programming language. Their method is divided into five stages to quantify the complexity of SQL commands.
Pawade et al use Python language to calculate Halstead metrics attributes [29]; however, we compare software in four languages to assess software Complexity.

3. Results and Discussion

We have conducted several experimental tests to show the importance of measuring the complexity of software. According to Halstead's complexity measures, the following parameters are used to calculate the metric as follows [30], [31]:

n_1 = the number of distinct operators
n_2= the number of distinct operands
N_1 = the total number of operators
N_2 = the total number of operands

These parameters are computed using the following equations:

Program vocabulary $\text{n} = n_1 + n_2$
Program length: $N = N_1 + N_2$
Estimated length: $\acute{N} = n_1 \log_2 n_1 + n_2 \log_2 n_2$
Truth program length: $\frac{\acute{N}}{N}$
Volume: $v = (N_1 + N_2)\log_2(n_1 + n_2)$ or
$v = N \ \log_2 n$
Difficulty: $D = \frac{n_1}{2} * \frac{N_2}{n_2}$
Effort: $E = D * v$
Calculating actual coding time from effort measure as follows:
Program time: $T = \frac{E}{18}$
Calculating of flaws in the execution by relation Halstead's delivered bugs (B) as follows:
Number of delivered bugs: $B = \frac{\text{V}}{3000}$
The first language in this comparative analysis study is the Python programming language as shown its code below in Figure.1.

```
def  is_odd(num):
       if  num & 1:
              return true
      return false
for i  in   range (1,100):
     print str(i) +”  :  “+ str(is_odd (i) )
```

Figure 1. Python code.

The Tables 1 and 2 show the results of applying Halstead metric on the Python language.

Table 1. Operators and Operands of Python

	Distinct	Total
Operators	n1=4 [' :', ' & ' , ' , ' , ' +']	N_1=7
Operands	n2=5 [' is_odd ' , ' num ' , ' True ' , ' False ' , ' i ']	N2=9

Table 2. Halstead Results of Python

Language	Python
Program vocabulary	9
Program length	16
Estimated length	19.610
Truth program length	1.226
Volume	50.719
Difficulty	3.6
Effort	182.588
Program time	10.144
Bugs	0.017

```
# include<iostream>
Using namespace std;
Bool is_odd(int num)
{       If (num % 2==1)
      {  return true ;
       }       {
     return false;}}
int main()
{          for (int i=1; i<=100; i++)
        {
         cout<<i << “ : “ << is_odd(i)
<<end1;
        }}
```

Figure 2. C++ code.

The second language in this comparative analysis study is the C++ programming language as shown its code below in Figue.2. Tables 3 and 4 show the results of applying Halstead metric on C++ language.

Table 3. Operators and Operands of C++

	Distinct	Total
Operators	n_1=5 [' % ' , ' = = ' , ' = ' , ' < = ' , ' + + ']	N1=5
Operands	n_2=7 [' is_odd ' , ' num ' , ' 2 ' , ' 1 ' , ' True ‘ , ' False ' , ' i ']	N2=13

Table 4. Halstead Results of C++

```
< Script >
function is_odd (num) {
        If (num  &  1)
        return  true ;
        else
        return  false ;
}
for (var i=1 ; i < 100 ; i ++) {
document.write (i +” :  “+is_odd ( i )
+)
}
</ Script >
```

Figure 3. JavaScript code.

Table 5. Operators and Operands of JavaScript

	Distinct	Total
Operators	n_1=5 [' & ' , ' = ' , ' < ' , ' + + ' , ' + ']	N_1=7
Operands	n_2=6 [' num ', ' 1 ‘, ' True ', ' False ', ' i ', ‘100’]	N_2=13

Table 6. Halstead Results of JavaScript

Language	JavaScript
Program vocabulary	11
Program length	20
Estimated length	27.110
Truth program length	1.356
Volume	69.189
Difficulty	5.417
Effort	374.770
Program time	20.821
Bugs	0.020

The third language in this comparative analysis study is the JavaScript programming language as shown its code in Figure.3. Tables 5 and 6 showing the results of applying Halstead metric on the JavaScript language. The fourth language in this comparative analysis study is the Java programming language as shown its code below in Figure.4. Tables 7 and 8 show the results of applying Halstead metric on Java language.

```
Public class MainFile {
Public static void main(String[] args
) {
for (int  i  =  1;  i < 100; i++)   {
System.out.printIn( i+ ”  : ”+ is_odd
(i) );
}
}
Public static boolean is_odd(int
num) {
If (num & 1) == 1)
return true;
else
return false;
}
```

Figure 4. Java code.

Table 7. Operators and Operands of Java

	Distinct		Total
Operators	n_1=6	[' = ', ' < ', ' + + ', '& ', ' = = ', ' + ']	N_1=7
Operands	n_2=6	[' i ', ' 1 ', ' 100 ', ' num ' , ' True ' , ' False ']	N_2=14

Table 8. Halstead of Java

Language	Java
Program vocabulary	12
Program length	21
Estimated length	31.020
Truth program length	1.470
Volume	75..280
Difficulty	7
Effort	526.980
Program time	29.270
Bugs	0.020

A histogram was constructed and is shown in Figure. 5. We can see an important difference between the python and Java, as, in most parameters, We can see the python has the least value from other languages while the Java has the most value from other languages, however; c++ and Javascript no an important difference between the two languages. Program time in Java is a triple of python. Program time in Java is 3 times of python, suggesting python is quicker than other languages in an implementation.

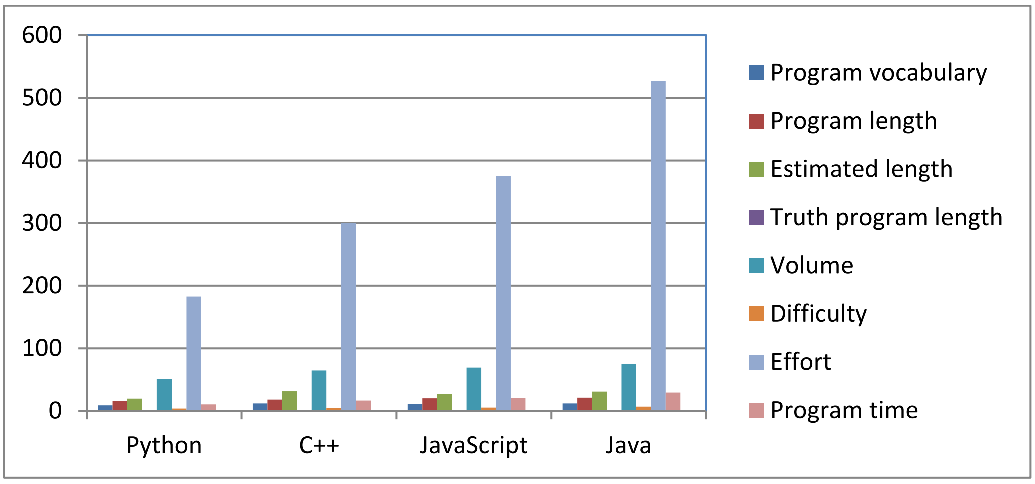

Figure 5. Comparative analysis.

4. Conclusion

In this paper, it has described a study that evaluates the performance of Halstead’s effort, a complexity measure in four programming languages (Python, C++, JavaScript and Java). It used the same program in these Languages to explain the rate of the complexity measure in each language. It shows that python is very user-friendly software and the simplest languages while Java is a powerful, modern, quicker and is more complex than other languages. Actual tests were executed for all languages and datasets under the same conditions. Although it is known that the different commands complexity in various languages cannot be calculated by the number of lines of code, this complexity measure is not simple to obtain.
It concluded from the above that the software metrics are important in the project life cycle stages and tested, when applying metrics of the program early and help overcome the existence of errors to a large extent and thus save time and cost. In contrast, Halstead complexity measures cannot be used in large programs. The use of software metrics is preferred when the requirements phase begin to keep up with every stage of the project. In the future, we will test the complexity of multifactor security application software’s to know the best real time programming language for such applications [32]. Also, more complex software’s will be tested by using other metrics other than Halsted metric [33].

References

[1] Boehm, Barry W. "Verifying and Validating Software Requirements and Design Specifications." IEEE software 1, no. 1 (1984): 75.
[2] A. Madi, O. K. Zein, S. Kadry, On the Improvement of Cyclomatic Complexity Metric, vol. 7, no. 2, 2013.
[3] D. Y. Pawade, , M. Metha, K. Shah, and J. Rathod, “ Python Based Software Complexity Calculator using Halstead Metrics”, Int. J. Adv. Found. Res. Comput. 2, no. Special Issue (NCRTIT 2015), pp. 390-394, 2015.
[4] Moreno-León, Jesús, Gregorio Robles, and Marcos Román-González. "Comparing computational thinking development assessment scores with software complexity metrics." In 2016 IEEE global engineering education conference (EDUCON), pp. 1040-1045. IEEE, 2016.
[5] Cooper James , “ Java™ Design Patterns: A Tutorial “, 2017.
[6] Flater, David, and David Flater. Software Science Revisited: “Rationalizing Halstead's System Using Dimensionless Units”. US Department of Commerce, National Institute of Standards and Technology, 2018.
[7] H. Chen, “Comparative Study of C C++ C# and Java Programming Languages”, Vaasan Ammattikorakeakoulu Vasa Yrkeshogskola university of applied sciences Information Technology, 2010.
[8] A. M. Alnaser, O. AlHeyasat, A.A.-K. Abu-Ein, H. Moh'd, S. Hatamleh, A. A. M. Sharadqeh ,

" Time Comparing between Java and {C}++ Software", vol. 5, no. 8, pp. 630-633, 2012.
[9] Warqaa Shaher AlAzawee, Ahmed M. Jasim and Sabah Anwer Abdulkareem," Design and Implementation of Database Management for Presidency of Diyala University", Diyala Journal of Engineering Sciences Vol. 13, No. 02, June 2020, pages 34-42.
[10] Christopher Kormanyos, " Real-time C++: efficient object-oriented and template microcontroller programming", Springer, 2018.
[11] Andrew Johansen, "C++: The Ultimate Beginner's Guide", Create Space, 2015.
[12] Gosling, Bill Joy, and Guy Steele," The Java language specification", Addison-Wesley Professional, 2000.
[13] Herbert, Schildt. "The Complete Reference Java." ,2019.
[14] Sultan S. Al-Qahtani, Pawel Pietrzynski, Luis F. Guzman, Rafik Arif, and Adrien Tevoedjre, "Comparing selected criteria of programming languages java, php, c++, perl, haskell, aspectj, ruby, cobol, bash scripts and scheme revision 1.0-a team cplgroup comp6411-s10 term report", arXiv preprint arXiv:1008.3434 (2010).
[15] Shi, Siyu. "Introduction to Python and Its Statistical Applications." In Open Source Software for Statistical Analysis of Big Data: Emerging Research and Opportunities, pp. 162-196. IGI Global, 2020.
[16] Deitel, Harvey, Paul Deitel, and Paul J. Deitel. "Python for Programmers". Prentice Hall, 2019.
[17] Flanagan, David. JavaScript-The Definitive Guide, 7e. O'Reilly Media, Incorporated, USA, 2020.
[18] Coimbra, Rodrigo Tavares, Antônio Resende, and Ricardo Terra. "A Correlation Analysis between Halstead Complexity Measures and other Software Measures." In 2018 XLIV Latin American Computer Conference (CLEI), pp. 31-39. IEEE, 2018.
[19] Vard Antinyan, Miroslaw Staron, and Anna Sandberg, "Evaluating code complexity triggers, use of complexity measures and the influence of code complexity on maintenance time", Empirical Software Engineering , Vol. 22, No. 6, pp. 3057-3087,2017.
[20] Hariprasad T, Seenu K.,Vidhyagaran G,Chaudrasegar Thirumalai, "Software complexity analysis using halstead metrics", 2017 International Conference on Trends in Electronics and Informatics (ICEI). IEEE, 2017.
[21] Tero Tyrväinen, Ola Bäckström, Jan-Erik Holmberg, and Markus Porthin. "SICA–A Software Complexity Analysis Method for The Failure Probability Estimation." In 13th International Conference on Probabilistic Safety Assessment and Management. International Association for Probabilistic Safety Assessment and Management IAPSAM, 2016.
[22] Anh Nguyen-Duc, " The impact of software complexity on cost and quality-A comparative analysis between Open source and proprietary software", arXiv preprint arXiv:1712.00675, 2017.
[23] Maksim O. Shudrak and Vyacheslav V. Zolotarev,"Improving fuzzing using software complexity metrics", ICISC 2015. Springer, Cham, 2015.
[24] Iwan Binanto, Harco Leslie Hendric Spits Warnars, Bahtiar Saleh Abbas, and Nesti Fronika Sianipar, "Halstead Metric for Quality Measurement of Various Version of Statcato", In 2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), pp. 276-280. IEEE, 2018.
[25] Safa Omri, Pascal Montag, and Carsten Sinz," Static Analysis and Code Complexity Metrics as Early Indicators of Software Defects", Journal of Software Engineering and Applications 11, No. 04, pp. 153-166, 2018.
[26] Sibel Senanand Selcuk Sevgen, " Measuring Software Complexity Using Neural Netwroks ", Istanbul University Journal of Electrical and Electronics Engineering 17, No. 2, pp. 3503-3509, 2017.
[27] Madhan M., I. Dhivakar, T. Anbuarasan, and Chandrasegar Thirumalai. "Analyzing complexity nature inspired optimization algorithms using halstead metrics." In 2017 International Conference on Trends in Electronics and Informatics (ICEI), pp. 1077-1081. IEEE, 2017.
[28] Made Agus Putra Subali and Siti Rochimah , " A new model for measuring the complexity of SQL commands", In 2018 10th International Conference on Information Technology and

Electrical Engineering (ICITEE), pp. 1-5. IEEE, 2018.

[29] Pawade, D. Y., et al. “Python based software complexity calculator using halstead metrics.” Int. J. Adv. Found. Res. Comput. 2.Special Issue (NCRTIT 2015) (2015): 390-394.

[30] Zuse, Horst. “Software complexity: measures and methods. “ ,Vol. 4. Walter de Gruyter GmbH & Co KG, 2019.

[31] Ahmad, Johanna, and Salmi Baharom. “Comparison of Software Complexity Metrics in Measuring the Complexity of Event Sequences.” In International Conference on Information Science and Applications, pp. 615-624. Springer, Singapore, 2017.

[32] Ali J. Abboud, “Multifactor Authenticator for Software Protection”. Diyala Journal for Engineering Sciences, Vol. 08, No. 04, Special issue, 2015.

[33] Ali N. Albu-Rghaif, Abbood K. Jassim and Ali J. Abboud, “ A Data Structure Encryption Algorithm based on Circular Queue to Enhance Data Security”, 2018 1st IEEE International Scientific Conference of Engineering Sciences - 3rdScientific Conference of Engineering Science (ISCES). pp. 24-29, 2018.