Validating Halstead Metrics for Scratch Program using Process Data

Zhong Chang
Beijing University of
Posts and Telecommunications
Email: viviansnow@bupt.edu.cn

RongGang Song
Yiwu Research Institute of
Education Science
Email: 26597023@qq.com

Yan Sun
Beijing University of
Posts and Telecommunications
Email:sunyan@bupt.edu.cn

***Abstract*—What will happen when traditional software complexity measures meet latest visual programming language? Are they still valid? In this paper, in order to validate classic Halstead Metrics for Scratch programming language, which is the most used language in K-12 education, we collect process data by modifying the Scratch platform. The results show a positive, significant and strong correlation between process data and Halstead Metrics, could be considered as a validation of Halstead Metrics for Scratch Program.**

I. Introduction

Halstead Metrics [1] are software complexity measures which have been widely used and adopted in software engineering. In K-12 education systems worldwide, the most common instrument for teaching programming is the use of visual programming languages based on blocks. Scratch is undoubtedly the most used language in this educational environment. However, some of the scientific approaches in conventional text programming may be invalid in visual programming. As far as we know, there is a lack of research that validates the Halstead Metrics for Scratch or other visual programming languages.

Particularly, an important application of Halstead Metrics for Scratch Program is to complement Computational Thinking(CT) assessment. Moreno-Len et al. compare the Computational Thinking score provided by Dr. Scratch [2], a free/libre/open source software assessment tool for Scratch, with Halsteads metrics. The findings of the paper [3] show positive, significant and strong correlations between them, which could be considered as a validation of the complexity assessment process of Dr. Scratch. However, the applicability of Halstead Metrics for Scratch program remain to be validated.

In this paper, we first review the Halstead Metrics and the related tool. Then, the method of collecting, processing process data was introduced in detail. Finally, we elaborate the findings from results. The results show strong positive correlations among real process data, Halstead Metrics and CT score assessed by Dr. Scratch.

II. Background and Releated Work

Halstead Metrics are software complexity measures which were first introduced by Maurice Howard Halstead in 1977. Halstead Metrics identify certain properties of a program that can be measured and the relationships among them to assess software complexity. The calculation of the Halstead Metrics in a program are based on four parameters: the number of distinct operators, the number of distinct operands, the total number of operators and the total number of operands, denoted as n_1, n_2, N_1 and N_2, respectively. Halstead Metrics in a program are defined as follows:

Vocabulary: the total number of distinct operators and operands in the program. $n = n_1 + n_2$

Length: the sum of all necessary tokens for the computation of the program. $N = N_1 + N_2$

Volume: the number of bits necessary to represent the program. $V = N * log_2 n$

Difficulty: it is used to compare different implementations of the same algorithm. $D = \frac{n_1}{2} * \frac{N_2}{n_2}$

Effort: effort required to create a program. $E = D * V$

Time: time required to create a program. $T = \frac{E}{18}$

Moreno-leon et al. developed a web app named Dr. Scratch on the basis of hairball [4], which is a plug-able static code analyzer for Scratch programs. Dr. Scratch was able to assess a Scratch program's Computational Thinking skills and Halstead Metrics.

III. Methodology

In order to collect process data during users' programming, we have modified and customized the free/open source Scratch source code[1]. Twelve kinds of operation types, including operations of "add a block", "delete a block", "double-click a block"(try to use a block), "copy a block", "add a sprite", "delete a backdrop" etc., and their operation time were saved into project.json. Project.json is a part of Scratch File Format used for storing encoded program metadata, sprites, scripts, and program media information.

The modified Scratch platform was deployed on a website named tuopinpin[2], which is a localized website created by our team. It widely used for studying, creating, sharing and exhibiting Scratch programs in primary school in China.

We collect all scratch programs which were uploaded to the website from Dec 21st, 2017 to Dec 24th. These total samples are 252. Aiming to validate Effort metric, the *T*otal *N*umber of *O*peration (TNO), the number of all operation types during

[1]https://github.com/BUPT902-Scratch/902Scratch-As
[2]http://www.tuopinpin.com/

TABLE I: Descriptive statistics of collected process data

	TNO	TPT
N	219	219
Mean	80.47	1637.68
Median	54	1508
Standard Deviation	82.01	894.09
Variance	6725.77	799405.41
Minimum	21	133
Maximum	475	5783

TABLE II: Correlation between metrics

	Effort Metric	Time Metric	CT Score	TNO	TPT
Effort Metric	1				
Time Metric	1	1			
CT Score	0.5615	0.5615	1		
TNO	0.7473	0.7473	0.4055	1	
TPT	0.7837	0.7837	0.4860	0.5884	1

the process of programming, was counted. Aiming to validate Time metric, the *T*otal *P*rogramming *T*ime (TPT, in seconds), which was the time difference between the first operation and the last operation, was calculated.

Table 1 shows the descriptive statistics of collected process data. Some invalid or error samples (e.g. remix of a program or an empty program) were discarded. Meanwhile, in 219 valid Scratch programs (denoted as *N* in the table), there are 195 Scratch programs successfully assessed by Dr. Scratch.

IV. Findings

Table II shows the Pearson correlation coefficient between each of metrics. As shown, TNO and TPT have a positive, significant, strong correlation with Effort Metric and Time Metric, while having a moderate correlation with CT score assessed by Dr. Scratch. The results also validate the conclusion that Halstead Metrics have a positive, significant and strong correlation with CT score, which was found by Moreno-Len et al. before. In our cases, Time Metric and Effort Metric have a moderate to strong positive correlation with CT score.

Fig. 1 shows the scatter plot of TNO and Halstead's Effort Metric with its best fitting line. The coefficient of determination is $r^2 = 0.56$, which indicates that 56% of the variance of Halstead's Effort of a Scratch program can be predicted from its TNO. The relationship between these two variables is better represented by the linear model for TNO between 25 and 100.

A similar case is depicted in Fig. 2. In this case, we set $r^2 = 0.61$, which states a better accuracy of the linear model than the prior one. However, for those programs with a TPT under 800 seconds, it is clear that the model does not represent the relationship between the metrics with the same accuracy.

V. Conclusions

In this paper, we introduce the process data collected through customed Scratch platform. Then the TPT and TNO were calculated from the process data. The positive, significant and strong correlations between TPT, TNO and Halstead's Time Metric, Halstead's Effort Metric validate the Halstead Metrics for Scratch Program.

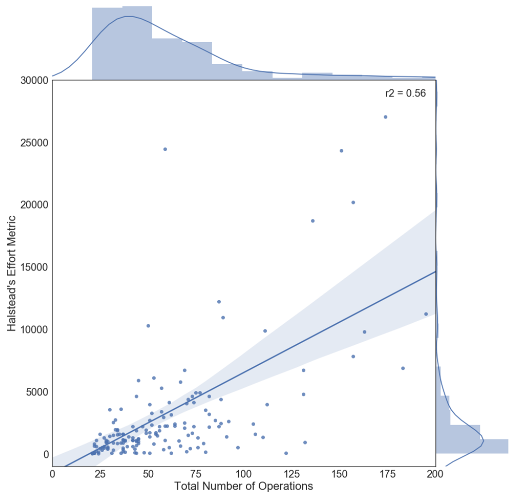

Fig. 1: Scatter plot of TNO and Effort Metric with its best fitting line $r^2 = 0.56$

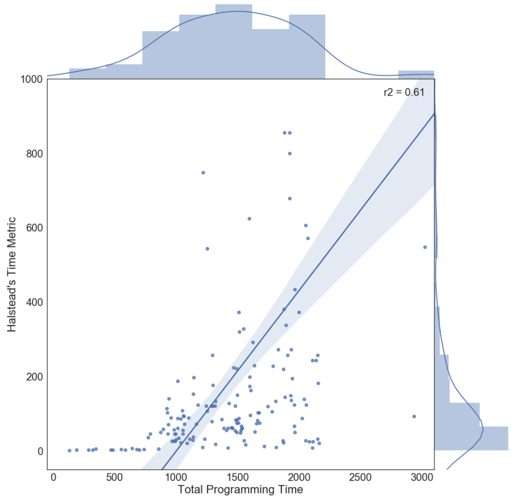

Fig. 2: Scatter plot of TPT and Time Metric with its best fitting line $r^2 = 0.61$

Future research could extend and complement criteria for assessing CT score by adding scope from Halstead Metrics and other Computer Science approaches.

Acknowledgment

This work is partly supported by the National Natural Science Foundation of China under Grant 61672109, 61772085, 61532012.

References

[1] M. H. Halstead, "Elements of software science," *Elsevierence*, 1977.
[2] J. Moreno-León, G. Robles, and M. Román-González, "Dr. scratch: Automatic analysis of scratch projects to assess and foster computational thinking," *RED. Revista de Educación a Distancia*, no. 46, pp. 1–23, 2015.
[3] J. Moreno-Len, G. Robles, and M. Romn-Gonzlez, "Comparing computational thinking development assessment scores with software complexity metrics," in *Global Engineering Education Conference*, 2016.
[4] B. Boe, C. Hill, M. Len, G. Dreschler, P. Conrad, and D. Franklin, "Hairball:lint-inspired static analysis of scratch projects," in *Proceeding of the ACM Technical Symposium on Computer Science Education*, 2013.